A New Block-based Reinforcement Learning Approach for Distributed Resource Allocation in Clustered IoT Networks

Fatima Hussain, *Senior Member, IEEE*, Rasheed Hussain, *Senior Member, IEEE*, Alagan Anpalagan, *Senior Member, IEEE*, and Abderrahim Benslimane, *Senior Member, IEEE*

***Abstract*—Resource allocation and spectrum management are two major challenges in the massive scale deployment of Internet of Things (IoT) and Machine-to-Machine (M2M) communication. Furthermore, the large number of devices per unit area in IoT networks also leads to congestion, network overload, and deterioration of the Signal to Noise Ratio (SNR). To address these problems, efficient resource allocation play a pivotal role in optimizing the throughput, delay, and power management of IoT networks. To this end, most of the existing resource allocation mechanisms are centralized and do not gracefully support the heterogeneous and dynamic IoT networks. Therefore, distributed and Machine Learning (ML)-based approaches are essential. However, distributed resource allocation techniques also have scalability problem with large number of devices whereas the ML-based approaches are currently scarce in the literature. In this paper, we propose a new distributed block-based Q-learning algorithm for slot scheduling in the smart devices and Machine Type Communication Devices (MTCDs) participating in clustered IoT networks. We furthermore, propose various reward schemes for the evolution of Q-values in the proposed scheme and, discuss and evaluate their effect on the distributed model. Our goal is to avoid inter- and intra-cluster interference, and to improve the Signal to Interference Ratio (SIR) by employing frequency diversity in a multi-channel system. Through extensive simulations, we analyze the effects of the distributed slot-assignment (with respect to varying SIR) on the convergence rate and the convergence probability. Our theoretical analysis and simulations validate the effectiveness of our proposed method where, (i) a suitable slot with acceptable SIR levels is allocated to each MTCD, and (ii) IoT network can efficiently converge to a collision-free transmission causing minimum intra-cluster interference. The network convergence is achieved through each MTCD's learning ability during the distributed slot allocation.**

***Index Terms*—MTCDs, Clustered IoT network, Machine Learning, Block Q-learning, Resource allocation**

I. Introduction

INTERNET of Things (IoT) and Machine-to-Machine (M2M) communication networks leverage a large number of heterogeneous devices with sensing, computing, and communication capabilities. These devices are also referred to as Machine-Type Communication Devices (MTCDs) that generate and transmit massive amount of data [1]. IoT is poised to create new business opportunities through futuristic applications in various domains such as, but not limited to, smart transportation, e-health, smart city, and industrial automation, to name a few. However, the communication among different MTCDs, and with the infrastructure results in simultaneous access of the limited channel resources. This phenomenon causes channel access problems due to the large number of MTCDs and their communication characteristics. Furthermore, the periodic and bursty data transmission, low power and low proximity communication, and contextual information exchange among MTCDs also cause channel access problems.

In essence, MTCDs usually transmit small-size data in each allocated time slot. However, the frequency of data transmission is much higher as compared to the traditional communication devices whereas the traditional communication devices transmit large amount of data but less frequently. Moreover, the mobility of MTCDs makes the communication more challenging. Therefore, efficient and delay-sensitive channel access is required for mobile MTCDs [2]. In this context, sophisticated resource allocation and spectrum management techniques are essential to meet the versatile resource requirements of the MTCDs in IoT networks.

In this paper, we address the network overload and congestion problems in IoT networks through clustering on the basis of spatial distribution of MTCDs. Clustering can efficiently reduce the network congestion and increase the energy efficiency by grouping a large number of MTCDs on various traits [3], [4]. We also propose a Q-learning algorithm for distributed slot assignment in the IoT networks. We aim at the slot assignment in Random Access Network (RAN) of MTCDs in a clustered IoT environment and define a strategy for controllers to select appropriate slot in a frame. Since all the controllers use the same time slots, there will be inter-cluster interference. Our goal is to assign the MTCDs in each cluster in a way that the inter-cluster interference is minimized. Additionally, we perform allocation on a block of slots simultaneously instead of the individual slots. In the following subsections, we abstractly discuss the resource allocation in IoT and basic ML concepts that are leveraged in our solution.

A. Resource Allocation in IoT Networks

Efficient resource allocation is essential in networks because it affects the overall network performance. The available

F. Hussain is with the Royal Bank of Canada, Toronto, Canada. E-mail: fatima.hussain@rbc.com

R. Hussain is with the Institute of Information Security and Cyber-Physical Systems, Innopolis University, Innopolis, Russia. E-mail: r.hussain@innopolis.ru

A. Anpalagan is with the WINCORE Lab, Department of Computer Science, Ryerson University, Canada. Email: alagan@ee.ryerson.ca

A. Benslimane is with University of Avigon, Email: abderrahim.benslimane@univ-avignon.fr

hyper-dimensional radio resources in the network (such as frequency bands, time slots, orthogonal codes, transmit power, and transmit-receive beams) must be efficiently managed and made dynamically adaptive to the fluctuation of wireless channels and traffic loads. Furthermore, it is at par important to fairly support the devices according to their Quality of Service (QoS) requirements in the network with scarce radio resources. Resource allocation can be mainly performed in two ways, i.e., scheduling and Random Access (RA). To this end, Contention-based Carrier-Sense Multiple Access (CSMA) and schedule-based Time-Division Multiple Access (TDMA) schemes are the obvious choices for channelization (scheduling and random access) of the MTCDs. CSMA is a natural choice due to its simplicity and flexibility; however, its performance can be significantly degraded in case of high contention due to the high overhead caused by resolving the collisions. Furthermore, additional packet delay is incurred in case of a competition for the channel access which increases the probability of collisions and re-transmissions. In contrast to CDMA, TDMA generally achieves higher network throughput even in case of high interference and contention scenarios [5]. Since sensor network-based IoT applications have heterogeneous nodes with variable traffic load, neither CSMA nor TDMA alone is a good choice. Therefore, it is very difficult to achieve time slots scheduling in a distributed fashion along with the tight synchronization.

In the wake of the afore-mentioned challenges, efficient and robust resource allocation algorithms are essential for heterogeneous IoT networks and more so when the channel conditions and traffic loads are intermittent. In such environments, distributed resource allocation techniques are more appropriate than the medium access scheduling [6]. Furthermore, conventional methods of resource allocation are not sufficient to meet the ever-increasing QoS requirements of the MTCDs with limited energy and the scarce radio resources. Recent research results show that Machine Learning (ML)-based resource allocation techniques outperform the conventional methods [7], [8]. In the following, we abstractly describe the ML from a bird's eye view.

B. Machine Learning (ML) from a Bird's Eye View

ML is a breed of Artificial Intelligence (AI) that helps MTCDs learn without explicit programming. ML differs from the traditional computational approaches in a way that ML algorithms train MTCDs on input data and use statistical analysis for predicting the output values. Thus, ML techniques facilitate communication devices in building models from training data, and the trained models automate the decision making based on the input data. ML can be used for modeling, optimization, prediction and forecasting the future outputs of the smart systems including IoT networks [9]–[11]. In the context of networks, ML-based techniques have been extensively used for channel estimation in Multiple Input Multiple Output (MIMO) systems to improve the end-to-end performance by learning statistical properties of the wireless channels [12], [13].

ML techniques are generally classified into three broad categories based on how the learning is performed, and how and what type of feedback is received from the environment on that learning. These categories are supervised, unsupervised and reinforcement learning. Supervised learning is performed when specific targets are set to achieve from certain inputs. While in unsupervised learning, the environment only provides inputs without any desired outputs. On the other hand, Reinforcement Learning (RL) is between supervised and unsupervised learning, and is performed when specific results at intermediate states are not defined and only the collective outcome is defined. In other words, the agent learns from the feedback received after performing some interaction with the environment. Q-learning is one of the most popular RL technique [14] that does not require any knowledge of the environment for making decisions. Agents learn with experience by performing actions and reward is assigned to the agents on the basis of their good or bad actions.

The rest of the paper is organized as follows: Section II summarizes the existing work addressing resource allocation in M2M and IoT networks using ML techniques. Section III describes our system model and Section IV presents centralized and distributed methods for resource allocation using block-based Q-learning in IoT networks. Simulation results are presented in Section V and Section VI concludes the paper.

II. Related Work

To date, many techniques such as scheduling, game theory, graph theory and ML have been used for efficient resource allocation in IoT networks [12], [15]. In [15], the authors analyzed resource allocation problem through both non-cooperative and cooperative games to maximize their data rate and minimize the power utilization. Similarly, the performance of coordinated and uncoordinated transmission strategies for multiple access is analyzed in [12]. Whereas, in [16], the authors proposed a predictive resource allocation scheme employed at the e-Node B (eNB) based on the propagation characteristics of the M2M applications. Furthermore, in [17], classification of the M2M scheduling techniques is presented from the perspective of versatile traffic requirements. Similarly, Fast Adaptive Slotted Aloha (FASA) is proposed in [18] that takes into account the knowledge of the previous states of slots that could be either idle, successful, or collided. This information is exploited to improve the performance of the access control protocol. Additionally, slotted ALOHA using Successive Interference Cancellation (SIC), also called Frame-less ALOHA, is proposed in [19].

The afore-mentioned techniques can ideally guarantee high performance in M2M scenario in terms of throughput. However, they have the following shortcomings: 1) energy efficiency and complexity aspects are not considered, 2) these techniques incur higher storage and processing overhead for eNB, 3) since the energy consumption in MTCDs is a major concern, frame size has major contribution to energy loss and has not been taken into account in the existing techniques, 4) the redundant data and control packets that are sent to eNB as a part of communication process, further increase the frame size and the network traffic, and 5) IoT applications are dynamic in nature, i.e., the devices have the ability to quickly

join and leave the network any time. Therefore, change in the network topology and network size make resource allocation more challenging. Due to these limitations of the existing techniques and dynamic nature of smart devices, ML-based techniques are recommended [20].

To this end, ML is used as a promising solution for various challenges faced by the densely populated heterogeneous IoT networks. These challenges include security, data analytics, resource allocation, and information routing [5], [21]. ML algorithms can predict and effectively schedule the available resources in IoT networks by considering the constituent factors that contribute to the efficient resource scheduling. Hence, such algorithms consider fluctuation of the wireless channels, variable traffic loads, QoS requirements, energy consumption of the smart nodes, and dynamically changing resources such as transmit power, frequency bands, time slots, and orthogonal codes. In [22], various ML algorithms are applied for data analysis in heterogeneous IoT devices and their performance is evaluated in terms of data processing, energy and efficiency. In another work [23], the authors discussed sequential learning and applied it on the IoT devices with stringent memory and computational constraints. Sequential learning enables IoT devices to change their transmission parameters adaptive to the changes in environment. Furthermore, in [24], the authors proposed a deep learning-based traffic load prediction and intelligent channel assignment algorithm to avoid congestion in IoT networks.

Reinforcement learning (RL) is one of the breeds of ML inspired by behaviorist psychology of agents, i.e., the way agents take actions and interact with the environment [25]. RL is also recently used for resource management in IoT networks. In [26], the authors used RL-based approach for interference mitigation in a macro-cell network underlaid with self-organized femto-cells. In this scenario, each femto-cell adapts its strategy and gradually learns by interacting with its environment. Furthermore, the authors in [27] proposed an RL algorithm that continuously adapts to the changing network traffic in deciding which action to take in order to maximize the energy saving at eNB. In [28], RL framework is presented for traffic offloading in a stochastic heterogeneous cellular network. Similarly, in [29], the authors used QoS performance measures for base station selection in a typical LTE environment. They used the ratio between the device throughput and its delay as a selection criterion to switch from one base station to another. In the same spirit, RL-based base station selection algorithm is proposed in [30] which allows MTCDs to choose base station in a self-organizing fashion.

Q-learning algorithms, a category of ML, adapt to the optimum action by gaining experience with the number of trials after learning about the environment [31]. Q-learning based MAC with Informed Receiving (ALOHA-QIR) for WSN is presented in [32] where frame-based slotted ALOHA and Q-learning are used to find slots such that the nodes have certain intelligence to access slots having lower probability of collision. A decentralized Q-learning technique is proposed in [33] to manage the interference generated by multiple devices in the network. Furthermore, Q-learning is employed in [34] to choose different transmission parameters and to make an efficient assignment of spectrum and transmit powers in the cognitive radios. In [35], a distributed Q-learning algorithm is proposed for sharing spectrum among femtocells and macrocells in a decentralized manner. Another Q-learning RACH access scheme (QL-RACH) is proposed in [36] to control the M2M traffic in order to reduce its impact on a cellular network. It uses ALOHA and an intelligent slot assignment strategy to avoid collisions among the MTCDs.

However, the performance of these schemes is dominated by the network traffic especially at the upper load limit. Most of these schemes have an assumption and restriction of using a virtual M2M frame having length (in time slots) equal to the number of MTCDs in the network. This imposes the upper bound on the number of MTCDs being served simultaneously. Additionally, every node maintains a Q-value for each slot in the M2M frame to record the transmission history on that slot in consecutive frames, which is not an energy efficient mechanism for the energy-constrained nodes.

We, on the other hand, reduce the computational and time complexity by introducing the concept of block-based or clustered slot allocation using Q-learning, which is efficient as compared to the works presented in [36], [37]. We improve the efficiency by performing distributed slot assignment that does not only minimize the inter-cluster interference, but also benefits from frequency reuse among all the clusters. In essence, we propose a scalable solution that adapts itself to the changing conditions of the network, traffic, number of devices and the available resources. Furthermore, the concept of Q-learning is applied on a block of slots instead of individual slots. In contrast to the conventional Q-learning method, our proposed mechanism conserves node energy because the Q-value records are maintained for multiple slots instead of an individual slot.

A. Our Contributions

In our work, we implement the clustering of MTCDs with cluster heads (or controllers) to facilitate the hierarchical control of resource allocation in massive IoT networks. We first consider the centralized slot allocation for MTCDs in our clustered network followed by the distributed assignment.

This work is a continuation of our previous work [5], [38] in which we first studied various channel allocation strategies followed by distributed channel assignment in clusters in a way that inter-cluster interference is minimized. In our previous work, we compared our distributed channel allocation with the random access protocol (Aloha). However, it is inefficient because one slot was utilized by only one cluster and no spatial reuse of time slots was realized. To address this issue, in this work, slots can be used simultaneously by more than one cluster; however, it gives rise to interference. Therefore, the interference is minimized by using block-based Q-learning algorithm.

We summarize our contributions as follows:

- We employ an energy-efficient clustering technique to overcome the congestion and overload problem in IoT network by applying K-means algorithm.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

- A centralized graph coloring-based algorithm is proposed for slot allocation in MTCDs such that minimum inter-cluster interference is experienced by all the devices.
- Q-learning-based distributed slot allocation is proposed for clustered IoT network.
- We perform comparative analysis of the centralized and distributed slot allocation in terms of average SIR and admittance rate for the clustered IoT network.
- We further analyze the convergence capabilities (convergence time and convergence probability) of the proposed approach with respect to different parameters.
- Finally, we propose various reward schemes for the Q-learning algorithm and perform comparative analysis of the incentive schemes, and show the dependencies of system output on the choice of reward scheme.

III. System Model and Problem Formulation

In this section, we outline the system model in detail. We consider a clustered IoT network with downlink communication as shown in Fig. 1. We assume densely populated smart MTCDs per unit area where a single eNB is not able to serve all the MTCDs in an efficient manner. Therefore, we use K-means clustering algorithm to make four clusters of smart MTCDs. A single eNB is considered which communicates only with the cluster heads instead of each device. MTCD controllers are responsible for data aggregation and transmission to their respective MTCDs as shown in Fig. 1. When more than one controllers transmit in the same time slot, interference is experienced as a result of such uncoordinated transmission. For instance, cluster heads 2 and 4 tend to transmit data to their associated MTCDs in slot 1 and experience the interference as shown by "Data" in red color. While cluster heads 1 and 3 transmit in slot 4 and slot 2, respectively and perform interference-free data transmission as shown by "Data" in black color.

We consider a TDMA-based IoT network having a frame of T slots and a controller for each cluster sends data to its associated devices in these slots. Following is the detail of problem formulation.

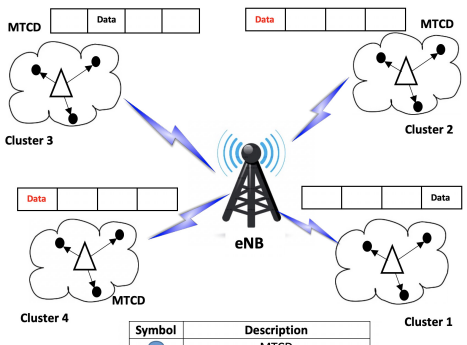

Fig. 1. Clustered IoT/M2M network.

Let K be the total number of clusters and D be the maximum number of devices in each cluster.

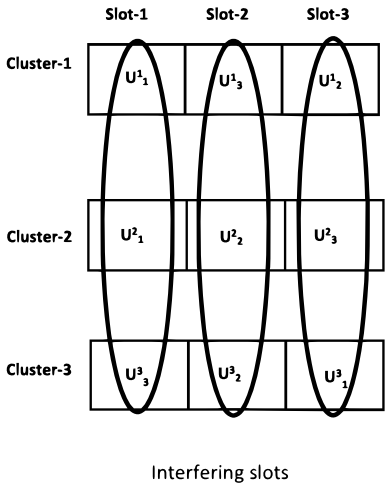

Fig. 2. Frame allocation and interference. ($T = 3, K = 3, D = 3$)

Each device in a cluster will be assigned a time slot for data transmission. Let i^{th} device in j^{th} cluster is denoted by U_i^j. In the Fig. 2, slot 1 in cluster 1 is assigned to user 1. Similarly, slot 1 in cluster 2 is assigned to user 1 of cluster 2, whereas slot 1 in cluster 3 is assigned to user 3 of the cluster 3. Let the r^{th} frame is denoted as F_r, and frames with user assignment can be explained as: frame 1 for cluster 1 is $F_1 = [U_1^1, U_3^1, U_2^1]$. Similarly, in the previous example, $F_2 = [U_1^2, U_2^2, U_3^2]$ and $F_3 = [U_3^3, U_1^3, U_2^3]$ are frame 1 of cluster 2 and cluster 3, respectively.

We use TDMA within the cluster and all devices (in each cluster) share the same frequency but are scheduled at different times. Therefore, there is no intra-cluster interference. However, since each time slot is used (simultaneously) by all the controllers (or clusters), there will be inter-cluster interference. Our aim is to assign slots to the devices in each cluster in a way that the inter-cluster interference is minimized. To achieve this goal, we use Q-learning algorithm.

Let the interference between different clusters in slot t is denoted by E_t on a particular frequency channel. This interference is the function of different devices assigned to the slot t in each cluster. In case of Fig. 2, slot $t = 1$ is assigned to devices $1, 1, 3$ associated with cluster $1, 2, 3$, respectively. Let γ_t be the resulting SIR of devices in the slot t. The SIR can be written as: $\gamma_t = f(U_1^1, U_1^2, U_2^3)$, where $f()$ is the function of slot assignment to MTCDs in each cluster. We calculate the actual power transmitted by each controller to its associated devices. This transmitted power is function of the distance between the controller and associated devices, and also the received power which is used to calculate the interference received by devices sharing the same time slot. Afterwards, we calculate the SIR for each device in all the clusters. Our goal is to find the combination of devices assigned to slot t in such a way that the interference is minimum and SIR (achieved by each device) is maximum. In the above example with three clusters, it can be written as:

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

$$\begin{aligned} \max_{U_i^j} \quad & \gamma_t, \forall t, i \in \{1,2,3\}, j \in \{1,2,3\} \\ \text{subject to} \quad & \\ & |F_r| = T, \forall r \end{aligned} \tag{1}$$

where F_r is the number of devices assigned in r^{th} frame. The above constraint means that all devices are assigned with slots in all the frames and no frame should be left empty. In a general case where there are T slots, R frames, K clusters and D MTCDs, the problem can be written as: At each time instant t,

$$\max \quad \gamma_t, \ \forall t = 1, 2, ..., T \tag{2}$$

$$\text{s.t. } |F_r| = T, \ \forall r = 1, 2, ..., R \tag{3}$$

$$U = [(U_1^1, ..., U_D^1), (U_1^2, ..., U_D^2), ..., (U_1^K, ..., U_D^K)] \tag{4}$$

Equation 2 is used to maximize the SIR in all time slots achieved by all the devices. We present a solution to this maximization problem using block-based Q learning. In our proposed method, each cluster head learns with experience about a specific slot that is assigned to a specific device within a cluster and it will maximize the SIR achieved by that device. This should hold true for all the devices during all the frames. U is the vector that consists of all the devices in each cluster. Here, each MTCD is numbered from 1 to D, and we assume equal number of devices in each cluster.

IV. Proposed Solution for Efficient Resource Allocation

In this section we discuss our proposed resource allocation techniques. Slots are assigned either in centralized or distributed way. In the centralized method, one controller or eNB is responsible for slot allocation to all of its associated devices. While in the distributed method, all the controllers assign slots to associated devices independently (without considering other controller assignments) and distributively, and modify their slot allocation according to the interference experienced by the associated devices. We first perform centralized slot allocation using conflict graph method followed by distributed slot assignment using RL technique. In the centralized method, eNB or controller is responsible for the formation of conflict graph and it broadcasts the conflict free slot assignment to all the controllers/heads or MTCDs. We use graph coloring method for conflict-free and least interference slot assignment, and afterwards this slot schedule is advertised by the controller to all the MTCDs in its cluster. While in the distributed slot assignment, each controller is responsible for the formation and advertisement of the slot schedules. In the following, we explain these methods in detail.

A. Centralized Slot Allocation

Downlink data transmission from controllers to their respective MTCDs is analogous to many independent point-to-point flows in the network. A best scheduling assignment is the conflict-free assignment which is sometimes very hard due to the existence of large number of MTCDs per unit area. In this regard, we try to assign slots where all MTCDs are able to maintain the required threshold of SIR. The conflicting node transmissions are determined based on an interference graph in the centralized assignment.

In essence, IoT network is represented by a graph $G = (V, E)$, where V is the set of nodes in the graph. These nodes correspond to $D + K = |V|$, the total number of MTCDs and controllers in the network. E represents the transmission links that are to be established and scheduled between K controllers and D associated MTCDs.

A node in a network may interfere with, or overhear another node, therefore these nodes should not transmit and receive simultaneously. The interference graph $C = (V, I)$ is formed by the estimation of SIR level among various nodes. $I \subset V \times V$ is the set of edges such that $(i, p) \in I$, and i and p belong to n and m clusters respectively, if either i or p can overhear each other and potentially interfere each other. Therefore, if i is currently an active receiver, p should not be scheduled to receive from the controller at the same time. To this end, any conflicting nodes are not colored with the same color.

Coloring a graph (nodes) is analogous to assigning a time slot to various MTCDs. The controller corresponding to each cluster should take into account the interferes within its range while generating schedules. After assigning the colors to the nodes, the controllers broadcast this information to the neighboring controllers. Neighboring controllers assign the time slots to the nodes associated to them, taking into account the already assigned potentially conflicting nodes which are associated with previous controllers. In Algorithm 1, we outline the steps for the proposed slot assignment followed by their explanation using interference graph coloring. In our solution, we consider the frame size of 8 slots as an example.

Algorithm 1 SIR-based slot allocation

1: Input: (i) Controllers: $1, 2, \ldots, K$, (ii) MTCDs in each clusters: $1, 2, \ldots, D$ and (iii) Time slots: $1, 2, \ldots, T$.
2: Output: Assignment of T slots for D MTCDs in K clusters
3: Initialize: $k^* \leftarrow 0$, $k^{new} \leftarrow 0$
4: $k^* \leftarrow$ the controller that has highest number of MTCDs
5: Randomly assign slots to MTCDs in k^*
6: **while** Time slots are available **do**
7: $k^{new} \leftarrow$ the farthest controller from k^*
8: Compute the SIR received by each MTCD in k^{new} for each available time slots
9: Assign the time slots to the MTCDs that maximizes their received SIR.
10: $k^* \leftarrow k^{new}$
11: **end while**

It can be seen that for T number of total available time-slots and D number of devices in each cluster, the complexity of Algorithm 1 is $\mathcal{O}$ $(T \times D)$.

1) Interference graph and coloring: In Fig. 3, three clusters are shown, and these clusters are served by cluster heads n, m and l, respectively. We note that MTCD i is served by cluster

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

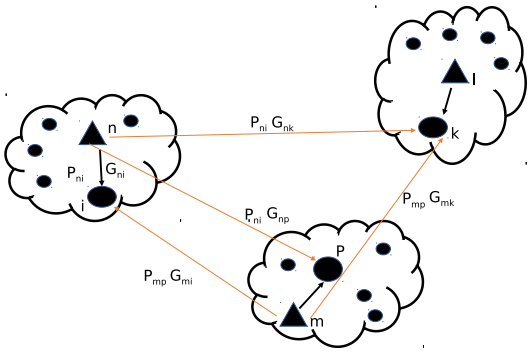

Fig. 3. Interference graph.

head n and G_{ni} denotes the channel gain between them. P_{ni} denotes the transmission power of cluster head n. Expression for the SIR at MTCD (with neglected noise) i is given by:

$$\gamma^{i}_{t\,(i\longrightarrow n)} = \frac{P_{ni}G_{ni}}{\sum_{\substack{j=1\\ j\neq n\\ O\in I_i}}^{K} P_{jO}G_{jO}}, \tag{5}$$

where I_i is the set of MTCDs in a cell that can potentially interfere with MTCD i. Let P_{mp} and P_{lk} be the transmitted power from m and l controllers intended for p and k MTCDs in their own clusters, respectively. This transmission is overheard by the MTCD i as three of them are using the same time slot t. Channel from controllers m and l to MTCD i is given by G_{mi} and G_{li}, respectively. Therefore, Eq. 5 is modified as:

$$\gamma^{i}_{t\,(i\longrightarrow n)} = \frac{P_{ni}G_{ni}}{P_{mp}G_{mi} + P_{lk}G_{li}} \tag{6}$$

Similarly, SIR of the MTCD p served by cluster head m is given by:

$$\gamma^{p}_{t\,(p\longrightarrow m)} = \frac{P_{mp}G_{mp}}{\sum_{\substack{j=1\\ j\neq m\\ O\in I_i}}^{K} P_{jO}G_{jO}} \tag{7}$$

2) Slot assignment: To further explain our model, let us assume that there are 3 slots in a frame, 12 MTCDs are served simultaneously by the controller and 13 onwards will experience denial of service. No two MTCDs will have the same slot within the cluster but two MTCDs of different cluster share the same slot. In the following, we explain various steps of slot assignment.

- We start with the cluster of maximum MTCDs and we assume that the time slots are sequentially assigned to all members.
- Distance to all the controllers is calculated. After that, we search for the potential interfering MTCDs starting from the cluster of the closest controller.
- Time slots are assigned to MTCD i in an iterative way to calculate the SIR. Time slot giving the optimum value of SIR is assigned to the respective MTCD. From interference perspective, both MTCDs i and p can safely communicate (transmit and receive) in the same time slot and interference caused by them is below the threshold. Furthermore, the overhearing is not significant to obstruct the reliable communication. This obstruction or interference is not significant and both the MTCDs are able to maintain the required SIR. If it is not the case, then both MTCDs are declared as interfering neighbors and are assigned different slots.
- After slot assignment is carried out for all the MTCDs in a second cluster, next cluster is picked up whose controller is nearest to the former, and the same process is repeated. This continues for all the clusters till least interference slot is assigned to the entire network.

We used the above centralized allocation technique to obtain conflict-free slot assignment as shown in Fig. 4, where x-axis and y-axis denote the coordinates of a grid. We obtain Fig. 4 by using 8 slots per frame, to better explain the conflict graph and the color assignment. MTCDs using the same time slots are shown with same color. This slot assignment is carried out in a way that minimum interference is experienced by all the MTCDs sharing same time slots. Color assignment is shown in Table I.

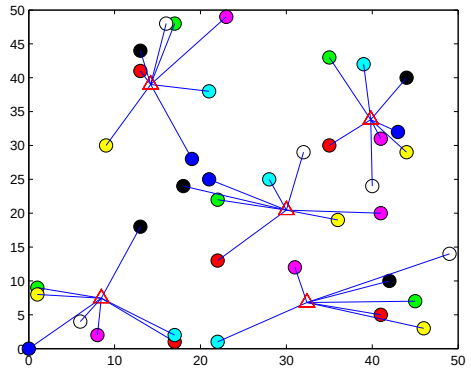

Fig. 4. Slot Assignment

TABLE I
COLOR ASSIGNMENT

Slot no.	Color assigned	Slot no.	Color assigned
1	yellow	5	green
2	blue	6	white
3	cyan	7	black
4	magenta	8	red

3) Shortcomings of the centralized approach: A centralized network architecture has single server or controller that handles all the major processing tasks. Centralized resource allocation techniques harness the benefits of consistency, efficiency, and firm control over the individual devices and network. However, lack of scalability and single point of failure does not make it a good choice for the resource allocation in distributed and heterogeneous IoT networks. Also, there is marked increase in the traffic due to exchange of control

messages for managing central control among the controller and associated nodes. On the other hand, decentralized network architecture distributes workloads among several devices instead of relying on central server or controller.

B. Distributed Slot Allocation

Without loss of generality, decentralized resource allocation methods have a visible edge over the conventional centralized counterparts. Various benefits such as increased system reliability, scalability, and privacy are achieved through distributed resource allocation. Also, there is no single point of failure and the privacy is achieved by passing data through various devices instead of single server or a controller. Keeping in mind the distributed nature of most of the IoT applications, distributed resource allocation techniques seem to be better choice as compared to its counterpart.

In the distributed implementation of slot assignment, joint learning and actions are carried out by each controller independent of each other. As controllers are independent of each other and eNB is not involved in any traffic exchange, we refer to it as a distributed approach. There are multiple agents with multiple actions in the considered scenario; therefore, we use Multi-Agent Reinforcement Learning (MARL) algorithm to perform distributed slot assignment [39], [40]. There are two distinct ways in which Q-learning can be applied to a multi-agent system, MARL and Join Action Learner (JAL). MARL algorithm is an Independent Learner (IL) algorithm because the agents perform their actions, obtain rewards and update their Q-values independent of the actions performed by other agents [41]. On the other hand, in JAL, an agent learns Q-values by performing joint action and it is also influenced by the actions of other agents. In our case, each controller takes actions independent of each other; therefore, it is MARL with IL. But individual Q-value is affected by the decisions of other controllers, therefore, they are also JALs. As there is not any joint Q-value and joint reward update, our problem falls into both categories. More precisely, our problem is identified as distributed and decentralized MARL with the combination of JAL and IL.

1) Multi-Agent Reinforcement Learning (MARL): RL is a mathematical tool for modeling the interactions between agents, i.e., controllers and cluster heads, providing them with the capability of learning which can enable them to make certain decisions [42]. In this work, we deal with a multi-agent scenario in which each controller learns from its environment about the suitability of its slot assignment. Each controller transmits data to its associated MTCD during a specific time slot and on a specific frequency channel, which are shared by all the controllers. This results in interference during transmission. Every controller learns from previous experience and schedules its MTCDs in a way that minimum interference is caused to the neighboring controller with whom it shares the slot and frequency channel.

2) Block Q-Learning for IoT network: We introduce a new concept of block Q-learning or clustered Q-learning. In essence, we consider distributed slot assignment in which controllers are independent of each others' decisions and actions. If slot-wise learning is performed, the complexity space increases exponentially with increase in the number of MTCDs because each device has to learn with respect to every other device in the network. We propose block-based slot assignment in which each controller assigns slots to all of its associated devices simultaneously (and not on individual basis). Q-learning is employed to learn with experience as to which slots are suitable for all the devices in the cluster, in one go, i.e., all the devices are assigned with slots at the same instant and under one action (for all). This way, the computational complexity will decrease and the efficiency will be improved because each cluster learns about the best suitable slots with experience which is influenced by the actions performed by other clusters.

C. Distributed Block-based Q-learning

The proposed distributed block-based Q-learning approach is based on the learning experience of controllers on a set or block of slots simultaneously. Reward function is assigned to the block of slots instead of an individual slot, and Q-values are also updated for the block of slots in contrast to the conventional Q-learning mechanism. Furthermore, devices are also combined into a group, i.e., each controller assigns reward values and acquires indirect feedback and experience (of good or bad slot allocation in terms of the obtained SIR) for all the devices instead of individual device. Figure 5 illustrates the flow-chart of our proposed block-based Q-learning algorithm. Data is transmitted by controllers and each controller has a number of blocks (slot combinations) available for data transmission to its associated MTCDs as shown in Eq. 11. Each block is initialized with zero Q-values and is updated on the basis of the achieved reward. The process is repeated for all the available blocks and the Q-values of all the blocks are compared. A block with the highest Q-value is selected only if achievable SIR (for each slot) is more than the acceptable threshold.

It can be seen that for L number of iterations and N number of frames, the complexity of the distributed scheme is $\mathcal{O}(L \times N)$.

1) Block description: Let $(D)^K$ be the total number of blocks available to be used by all the controllers, where K is the number of clusters and D is the number of devices in each cluster. From this set, each controller will select one block on the basis of previous Q-values and then calculates the reward on the basis of the received SIR.

We define a matrix $\boldsymbol{\gamma}$, where rows represent the number of cluster heads and columns represent the SIR of each associated device of all controllers.

By considering the device assignment in Fig. 2, the matrix $\boldsymbol{\gamma}$ is obtained as follows:

$$\boldsymbol{\gamma} = \begin{bmatrix} \gamma_{11} & \gamma_{13} & \gamma_{12} \\ \gamma_{21} & \gamma_{22} & \gamma_{23} \\ \gamma_{33} & \gamma_{31} & \gamma_{32} \end{bmatrix}, \tag{8}$$

where $\boldsymbol{\gamma_{i,j}}$ represents SIR of the i^{th} device in the j^{th} cluster. After calculating the SIR for each device, the controller checks if it meets the required $\boldsymbol{\gamma^{th}}$. or not. Using the relation in Eq. 12, $'0'$ and $'1'$ are assigned and the matrix $\hat{\boldsymbol{\gamma}}$ is obtained.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

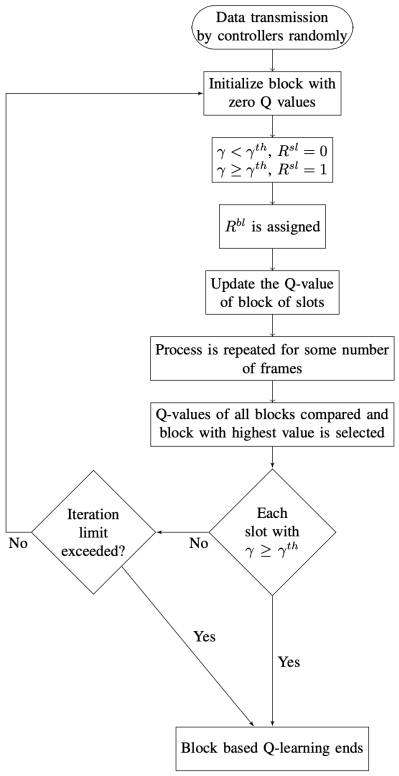

Fig. 5. Flow chart of Block Q-learning

$$\hat{\boldsymbol{\gamma}} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \tag{9}$$

The discrete values for SIRs depend on whether they meet the threshold or not, and are assigned in $\hat{\boldsymbol{\gamma}}$. For instance, second and third device of cluster 3, and second device of cluster 2 did not meet the required threshold; therefore, 0 is assigned. While the rest of the devices in all the clusters meet the required threshold, therefore, 1 is assigned. After obtaining the matrix $\hat{\boldsymbol{\gamma}}$, reward R^{bl} is obtained using Eq. 16 as explained below.

2) Q-Value update: We use stateless Q-learning in our proposed scheme to obtain the learning experience. Each controller has individual Q values for every block and the controllers select the block such that maximum number of MTCDs are able to meet the required SIR threshold. Q values are denoted by $Q(j, T_{bl}, w)$ and these values represent the action taken by the controller j on the block of slots T_{bl} to obtain the frequency channel w, and α is the learning rate. We use $w = 1$ and same approach can be used for multiple frequencies as well. All previous Q values and the current reward contribute to the Q value update using the following expression:

$$Q_{n+1}(j, T_{bl}, w) = Q_n(j, T_{bl}, w) + \alpha(R^{bl}_{n+1} - Q_n(j, T_{bl}, w)) \tag{10}$$

To this end, the reward is calculated when actions are performed by the agents, followed by the Q value update, as follows:

- Agent: MTCD controller (or cluster heads) $j, \forall 1 \leq j \leq K$, are the agents running Q-learning algorithm. These controllers perform the selection of the best block of slots for their associated devices.
- Action: $A(n) = a^{j,\hat{T_{bl}}}(n), T_{bl} \in [T_{bl^1}, ..., T_{bl^D}]$, where $a^{j,T}(n)$ is defined as the action of j^{th} controller at time instance n and the aim is to choose a block $\hat{T_{bl}}$ out of T_{bl}, where $|T_{bl}| = (D!)$ is the number of blocks available to each controller while $(D)^K$ is the total number of blocks. For instance, block combinations for controller 1 having three associated devices are $3! = 6$, and they are:

$$T_{bl} = \begin{cases} U_1U_2U_3 \\ U_1U_3U_2 \\ U_2U_1U_3 \\ U_2U_3U_1 \\ U_3U_2U_1 \\ U_3U_1U_2 \end{cases} \tag{11}$$

  The order of each block gives the allocation sequence for each associated device. For example, first block indicates that the first device is assigned to first slot, second device to the second slot, and third device to the third slot.
  Exploration strategy: Actions are performed by cluster heads using un-directed exploration technique [43]. Considering the pros and cons of various un-directed exploration techniques, we used the combination of the naive algorithm and the greedy algorithm. Cluster heads start with a random selection (naive algorithm) of one block from a set of available blocks and afterwards select block on the basis of Q-learning preferences (greedy algorithm). It will pick the block with highest Q-value. If the reward of current action is less than the action in the previous step, it will follow the naive strategy and sequentially pick the next block in the available blocks.
- Reward: is defined as the benefit that controllers will get in terms of SIR as described below.

Reward calculation: We assign the reward in two stages:

1) In the first step, calculate SIR for the individual slots and interpret SIR into discrete values as follows:

$$R^{sl} = \begin{cases} 1, & \gamma \geq \gamma^{th}, \\ 0, & \gamma < \gamma^{th}, \end{cases} \tag{12}$$

   where γ^{th} is the required threshold of SIR by the MTCDs for proper functionality. Using this reward function and Eq. 9, matrix $\hat{\boldsymbol{\gamma}}$ is obtained.
2) In the second step, actual reward allocation on a block of slots is followed by Q-value update. R^{bl} is calculated by counting the number of 1's in the $\hat{\boldsymbol{\gamma}}$ as follows (for the above example):

$$R^{bl} = \begin{cases} 1 & \text{all } 1's, \\ -1 & \text{one } 0, \\ -2 & \text{two } 0's, \\ -3 & \text{all } 0's. \end{cases} \tag{13}$$

Block reward is 1 if the reward for all slots is equal to 1, while block reward is -3 when the total reward for all the slots is 0. If two slots have 0 reward, then block reward is -2.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

Whereas block reward is -1 if only one slot has reward equal to 0.

3) Rewards schemes: We propose three different reward schemes as follows.

- Pessimistic reward (R1): If all the MTCDs of a cluster reach the required SIR threshold, 1 is assigned to each slot in a block, and then block reward is assigned as 1. If only 2 devices get the required SIR, block will have one 0 and two $1's$, and block reward is assigned as -1. Hence, for our reward scheme, number of $0's$ are counted to determine the reward. MTCDs are penalized more for wrong selection of the slots and rewarded less for the good choice in this scheme. This choice of reward scheme depends on the type of IoT application scenario. Therefore, some applications require more strict selection than others. For the given example in matrix $\hat{\gamma}$, reward assignment is given below:

$$R^{bl} = \begin{cases} 1, \\ -1, \\ -2. \end{cases} \tag{14}$$

  Additionally, there are two more reward schemes explained below:
- Optimistic reward (R2):

$$R^{bl} = \begin{cases} 3 & \text{all } 1's, \\ 2 & \text{two } 1's, \\ 1 & \text{one } 1, \\ -1 & \text{no } 1's. \end{cases} \tag{15}$$

- Balanced reward (R3):

$$R^{bl} = \begin{cases} 1 & \text{all } 1's, \\ 0.5 & \text{one } 0, \\ -0.5 & \text{two } 0's, \\ -1 & \text{no } 1's. \end{cases} \tag{16}$$

After obtaining R^{bl}, Q values are updated using Eq. 10. Controllers will learn by experience which block is good for the data transmission by comparing their Q values. Block with higher Q values will always be preferred by the controller j for frequency w, i.e.,

$$I^{j,w} = \max_{T_{bl},j,w} Q_t(j, T_{bl}, w) \tag{17}$$

At the bootstrapping phase, all the Q values are initialized to 0. If multiple blocks have the same Q value, controllers randomly select one of them.

All three schemes can be better explained and compared with the example in Table II. We have three clusters and each cluster has three users. Three slots in a frame have to be assigned by controllers to their MTCDs in way to minimize the interference. We consider the block allocation in one cluster with the same situation for all three reward schemes. In the pessimistic scheme, as there is only a single positive reward, only the combination of slots which gives that reward, will lead to an increasing Q-value. All other combinations will decrease the Q-values and hence should not be selected. In the optimistic scheme, as there are multiple rewards, it is possible that a two-slot combination will always be selected because it can increase the Q-values. Hence, in the optimistic scheme, we can have multiple slot combinations that can increase the Q-value with a possibility that the Q-value is only maximum for that reward, but not the maximum over all rewards. This is the similar case with the balanced scheme, although it will happen less frequently due to a higher difference between the two positive rewards.

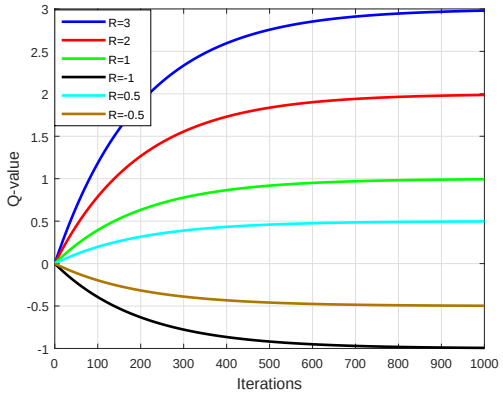

Fig. 6. Evolution of Q-values vs iteration for various rewards

4) Evaluation and comparison of reward schemes: We compare various reward schemes in Table II and illustrate the performance of our algorithm with respect to different rewards in Fig. 6. We show the evolution of Q-values versus iterations for different reward values. For better illustration, we assume a low value of α (learning rate) and assume that only one reward is selected till the end of all iterations. This way, we obtain six evolving Q-values, one for each reward over 1000 iterations. As it can be seen, positive rewards lead to increasing and negative rewards lead to decreasing Q-values. If the difference of rewards is less, difference between the evolving Q-values

TABLE II
COMPARISON OF REWARD SCHEMES

Block selection	Pessimistic reward		Optimistic reward		Balanced reward		Time
	Block	Q-value	Block	Q-value	Block	Q-value	
$U_1U_2U_3$	√	-0.5	√	1	√	0.25	t=1
$U_1U_3U_2$		0		0		0	
$U_2U_1U_3$		0		0		0	
$U_2U_3U_1$		0		0		0	
$U_3U_1U_2$		0		0		0	
$U_3U_2U_1$		0		0		0	
$U_1U_2U_3$		-0.5	√	1.5	√	-.125	t=2
$U_1U_3U_2$	√	-1				0	
$U_2U_1U_3$		0				0	
$U_2U_3U_1$		0				0	
$U_3U_1U_2$		0				0	
$U_3U_2U_1$		0				0	
$U_1U_2U_3$		-0.5	√	0.5		-.125	t=3
$U_1U_3U_2$		-1		0	√	0.25	
$U_2U_1U_3$	√	1		0		0	
$U_2U_3U_1$		0		0		0	
$U_3U_1U_2$		0		0		0	
$U_3U_2U_1$		0		0		0	

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

is also less, e.g., for reward R = 3 and R = 2, the Q-values are close, whereas for R = 1 and R = 0.5, the difference is more. In fact, for R = 0.5, a similar Q-value as R = 1 takes much more iterations as compared to R = 2 and R = 3.

V. Performance Evaluation and Discussion

In this section, we present the simulation results and discuss the performance of our proposed distributed algorithm. We first discuss the simulation environment followed by discussion on the obtained results.

A. Simulation Environment

We use MATLAB software to model our system. The simulation parameters and their values are summarized in Table III. We use a grid of 50m×50m, where x and y axis denote the distance units over which 4 cluster heads and 12 devices are uniformly distributed. Clusters are formed on the basis of spatial distribution.

TABLE III
Parameters and Values

Symbol	Definition
$K = 4$	Number of clusters/MTCD controllers
$D = 3$	Number of devices in each cluster
$T = 3$	Number of slots in each frame
U_j	Set of MTCDs in j^{th} cluster
$\gamma^{th} = 2, 4, .., 10$	SIR Threshold
$P_r = 1$ mW	Received power

B. Performance Analysis

We analyze our proposed algorithm in heterogeneous M2M and IoT network scenarios. First of all, we compare the centralized and distributed techniques and then analyze the convergence time (one time unit is one time slot) and convergence probability at various SIR threshold levels. We further analyze the effect of varying SIR thresholds on the admittance rate of MTCDs with the help of cumulative success probability. Admittance rate of MTCDs is defined as the number of MTCDs able to access the channel achieving acceptable SIR threshold per unit time. Moreover, we show that our proposed algorithm achieves acceptable SIR efficiency with minimum interference in distributed scenario.

C. Centralized versus Distributed Implementation

Before we discuss the simulation results, we discuss the comparison between centralized and distributed schemes. This comparison is based on inter- and intra-cluster interference, complexity and the number of users per slot. Inter-cluster interference is avoided in fully distributed scheme and minimized in rest of the two. While the intra-cluster interference is avoided in all three schemes. Number of users transmitting in one slot is more than one in the centralized and partially distributed schemes while only one user per slot can transmit in fully distributed scheme. Centralized scheme has the highest communication complexity due to duplex communication among eNB and controllers, and among the controller and MTCDs. Also, this communication increases network traffic and it may cause network congestion. Partially distributed scheme has moderate communication complexity and communication occurs only between controllers and MTCDs. Whereas fully distributed scheme has lowest communication complexity than the other two schemes. There is no direct communication among MTCDs, controllers and eNB. Only ACK and NACK signals are exchanged between controllers and the associated devices.

D. Simulation Results

Through extensive simulations, we compare the centralized and distributed slot allocation and show the superiority of latter in terms of admittance rate. In Fig. 7, admittance rate for both types of slot allocation at various SIR thresholds is shown. It is evident that more MTCDs satisfy the required SIR threshold in distributed allocation. This is due to the repeated reward calculation and resulting slot allocation. The convergence criteria for the block-based Q-learning algorithm is to maximize the number of MTCDs qualifying the given SIR threshold. Figure 8 shows the average SIR for both types of allocation for various SIR threshold. Centralized allocation outperforms the distributed allocation which is due to less number of MTCDs admitted at the same SIR level as compared to the distributed allocation. This is due to the fact that centralized allocation emphasizes only on maximum SIR rather than maximizing the number of admitted users. While in the distributed allocation, both admitted users and SIR are considered.

To this end, distributed allocation performs well in terms of computational complexity and time. The block-based Q-learning approach enables controllers to allocate slots to all of their associated MTCDs simultaneously and thus improves the efficiency. Additionally, there is no involvement of eNB in slot allocation by the controllers which reduces the burden on eNb and also significantly reduces the network traffic.

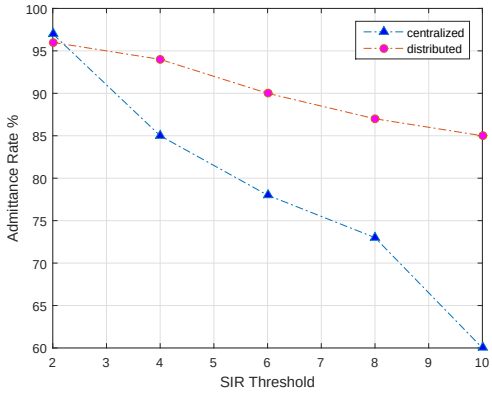

Fig. 7. Admittance rate for centralized and distributed allocation

1) Distributed implementation: Distributed resource allocation is better than the centralized in many aspects. In the centralized approach, a central controller has access to the global channel and network knowledge, and hence it

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

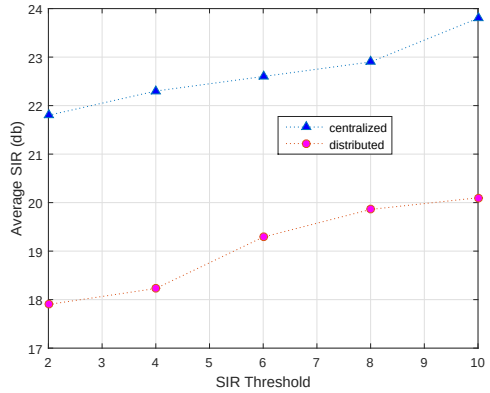

Fig. 8. Average SIR for centralized and distributed allocation

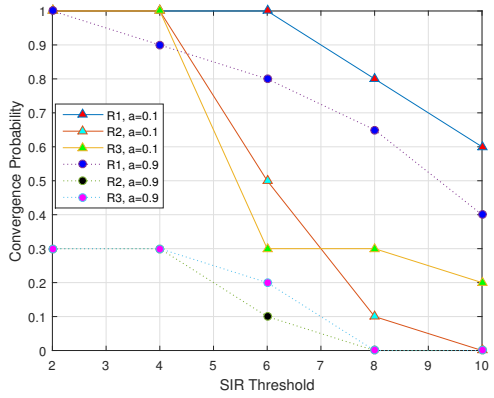

Fig. 9. Convergence probability at various reward schemes; R1:Pessimistic, R2: Optimistic, R3:Balanced

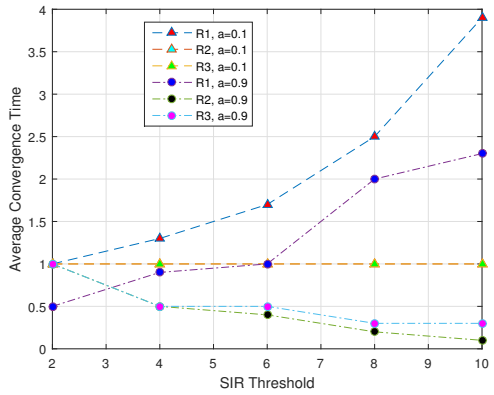

Fig. 10. Convergence time at various reward schemes; R1: pessimistic, R2: optimistic, R3: balanced

is capable of making coordination and resource allocation decisions. Through this approach, the resources are assigned to all the devices in the network. Without an efficient and fast infrastructure, centralized allocation is a challenging task due to the exchange of inter-cell scheduling information and large amount of feedback required by the controllers to send all the information. Also, complexity increases exponentially with the increased number of devices. The distributed strategy reduces both signaling and feedback requirements as compared to its counterpart. Distributed schemes are scalable and the information exchange among devices and the resulting overhead is according to the size of the network (number of associated devices). Hence, in the following, we further analyze the distributed scheme using Q-learning).

2) Reward schemes: Figures 9 and 10 show the comparison among various reward schemes. We used convergence probability and convergence time as the comparison metrics for various schemes. Performance of the algorithm depends upon the underlying reward scheme. We can see in Fig. 9 and Fig. 10 that pessimistic reward (R1) outperforms the rest of two schemes. It has highest convergence probability as immediate rewards are biased towards negative values which enables controllers to try different combinations of slot allocation. This increases the possibility of convergence. While for the optimistic reward scheme, immediate rewards are biased towards positive values and there are more chances that controllers are biased to pick up the same combinations (as it gives positive reward to them). Therefore, there is a small probability that the controllers will pick up new combinations for slot allocations. Hence, there are less chances of convergence as controllers will keep on adopting the same combination of slot allocation. Most of the times, the algorithm is unable to allocate the slots, the only time it is able to do that is when the initial slot allocation matches (or is very close to) the required slot allocation. Thus, when the initial slot allocation is close to the required allocation, the algorithm converges quickly.

The reason for such performance and the dependency on reward schemes can be explained by observing the degree of adaptation to positive or negative rewards in each scheme. In the pessimistic scheme, there is only a single positive reward and that is awarded only when all three slots meet the required SIR threshold. In other words, Q-value is only increased when all three slots meet the acceptable SIR criteria. In the optimistic scheme, there are multiple positive rewards which means that even when there is only a single slot that meets the SIR threshold, the reward is still positive which will increase the Q-value. In the balanced scheme, the Q-value will only increase when a minimum of two slots meet the required SIR threshold. The proposed algorithm selects the block that has the maximum Q-value. In case of pessimistic scheme,selected block can only be the one where all slots meet the required threshold. In case of optimistic and balanced schemes, there can be multiple blocks that will lead to a higher Q-value. Thus, it means that the solution given by the optimistic and balanced schemes is not always unique and is dependent on the initialization.

From above numerical and simulation results, pessimistic reward scheme appears to be the best choice in the current

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TVT.2020.2965796, IEEE Transactions on Vehicular Technology

settings. We apply this reward scheme for the further analysis of our algorithm.

3) Convergence time: In Fig. 11, convergence time is shown with respect to various SIR thresholds. Convergence time is defined as the time taken by the controllers to learn with repeated data transmission and resulting Q-values for the selection of best block and best time slots for their data transmission. Convergence is declared when there is no further change in the block selection and slot assignment. We calculate the convergence time obtained over 1000 iterations and average the results. Different values of SIR varying from 2 to 10 in a step-size of 2 are used to study the change in convergence time. Convergence time varies from 2 to 9 slots and it increases with the increase in the SIR threshold. This is because of the fact that with increasing threshold, there are less configurations of MTCDs that can meet the required threshold. This is an expected behavior as it is difficult for all the MTCDs to meet the required SIR threshold. Furthermore, the controllers take more time to learn with experience to figure out how to pick up the best block.

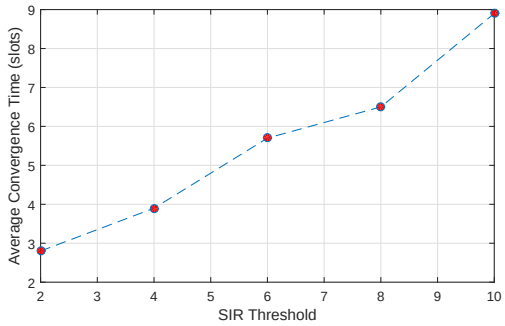

Fig. 11. Convergence time at various SIR thresholds

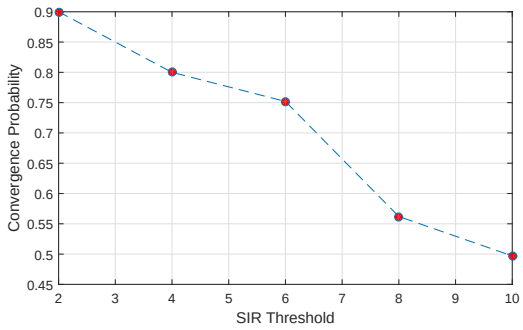

Fig. 12. Convergence probability at various SIR thresholds

4) Convergence probability: Fig. 12 shows the convergence probability for various SIR thresholds. Convergence probability is defined as ratio of the number of times the simulation converges and the total number of iterations. As mentioned, the SIR is varied from 2 to 10 in a step size of 2. A decreasing trend in convergence probability can be observed between 0.5 and 0.9. We can see that when SIR threshold is increased, the convergence probability is decreased. It is because with the increase in SIR threshold, less configurations of MTCDs can fulfill the required thresholds (as shown in Fig. 11). Hence, the simulation converges less number of times as the SIR threshold increases, indicating that it is difficult for all MTCDs to get the desired threshold simultaneously.

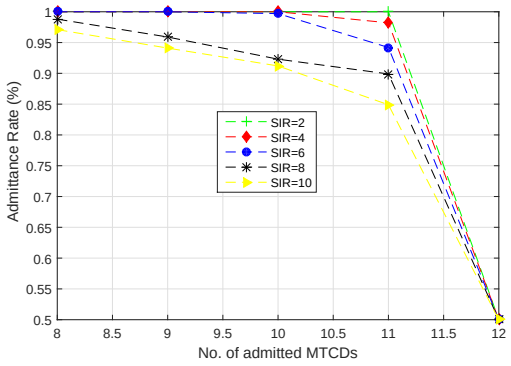

Fig. 13. Successful smart devices/MTCDs at various SIR thresholds

5) Admittance rate: In Fig. 13, cumulative distribution function is used to study the effect of SIR threshold on the number of MTCDs meeting the required SIR level. It shows only the number of successful MTCDs at each threshold level provided that all of them meet the convergence criteria. For example at $\gamma = 10$, all of the MTCDs meet the required SIR criteria (i.e., all the MTCDs will get the SIR greater than SIR threshold) 50% of time. We can also see that as the SIR level increases, convergence probability decreases and all the devices do not meet the required threshold. As the SIR threshold is decreased, increase in the number of satisfying MTCDs is observed. When the number of MTCDs is 8 and the admittance rate is 0.5, it means that only 4 MTCDs are admitted. For $\gamma = 2$, all the MTCDs are admitted as the threshold is lower enough. This can also be seen from Fig. 11 and Fig. 12, where at $\gamma = 2$, the convergence time is minimum and convergence probability is maximum. For $\gamma = 4$ to 8, the admittance rate starts decreasing from 10 MTCDs and goes to 0.5 for 12 MTCDs. $\gamma = 10$ has the lowest admittance rate. A decreasing trend of admittance with respect to SIR threshold can be seen which shows that, with an increase of SIR threshold, lower number of MTCDs will be admitted.

Figure 14 shows the percentage of the MTCDs satisfying the given SIR threshold criteria. Three SIR levels, 2, 6, and 10 are used to perform the analysis. We can see that for $\gamma = 2$, 90% of times all the MTCDs reach the required level while for $\gamma = 6$, this number reduces to 75% and for $\gamma = 10$, it is reduced to 30%.

VI. Conclusion and Future Work

In this paper, we proposed a distributed slot allocation mechanism in random access network of MTCDs and presented a strategy for the selection of time slots by MTCD controllers. We showed that distributed slot assignment outperforms the centralized mechanism in the IoT networks in terms of computational complexity and time. Our proposed resource allocation mechanism employs block-based Q-learning. We

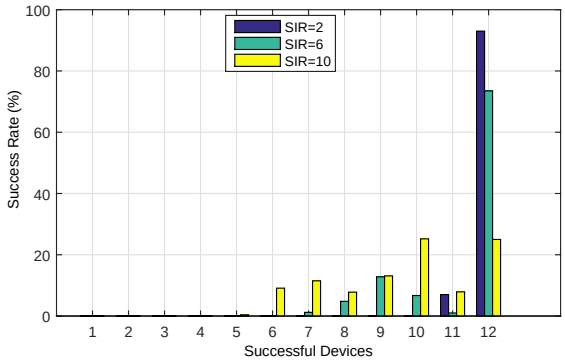

Fig. 14. Successful smart devices/MTCDs at SIR threshold $2, 6, 8$

also investigated both analytically and through simulation, the relationship between SIR threshold and the convergence probability. It was observed that the convergence probability decreases with increase in the SIR threshold. We further showed that it takes longer to converge at higher values of SIR threshold. For the future work, we aim at extending the same work for various reward schemes explained in this paper. Also, we aim to use Q-learning within a distributive shared slot OFCDM environment. Controllers will learn with experience as to which code and spreading is preferred for them for the lowest interference and frequency diversity.

REFERENCES

[1] F. Hussain, *Internet of Things: Bilding Blocks and Business Models*. Springer International Publishing, 2017. [Book].

[2] S. Verma, Y. Kawamoto, and N. Kato, "Energy-efficient group paging mechanism for qos constrained mobile iot devices over lte-a pro networks under 5g," *IEEE Internet of Things Journal*, pp. 1–1, 2019.

[3] G. Kumar and H. Mehra, "An Hybrid Clustering Algorithm for Optimal Clusters in Wireless Sensor Networks," *IEEE Student Conference on Electrical, Electronics and Computer Science*, pp. 1–6, March 2014.

[4] H. Harb and A. Makhoul, "K-Means Based Clustering Approach for Data Aggregation in Periodic Sensor Networks," *IEEE Conference on on Wireless and Mobile Computing, Networking and Communications*, pp. 435– 441, December 2014.

[5] F. Hussain, A. Anpalagan, A. S. Khwaja, and M. Naeem, "Resource Allocation and Congestion Control in Clustered M2M Communication using Q-Learning," *Wiley Transactions on Emerging Telecommunications Technologies*, 2015.

[6] F. Hussain, A. Anpalagan, and R. Vannithamby, "Medium Access Control Techniques in M2M Communication: Survey and Critical Review," *Wiley Transactions on Emerging Telecommunications Technologies*, September 2014.

[7] S. Chinchali, P. Hu, T. Chu, M. Sharma, M. Bansal, R. Misra, M. Pavone, and S. Katti, "Cellular network traffic scheduling with deep reinforcement learning," in *AAAI*, 2018.

[8] J. S. Jang, Y. L. Kim, and J. H. Park, "A study on the optimization of the uplink period using machine learning in the future iot network," in *2016 IEEE International Conference on Pervasive Computing and Communication Workshops (PerCom Workshops)*, pp. 1–3, March 2016.

[9] S. Maghsudi and S. Staczak, "Channel selection for network-assisted d2d communication via no-regret bandit learning with calibrated forecasting," *IEEE Transactions on Wireless Communications*, vol. 14, pp. 1309–1322, March 2015.

[10] U. S. Shanthamallu, A. Spanias, C. Tepedelenlioglu, and M. Stanley, "A Brief Survey of Machine Learning Methods and their Sensor and IoT Applications ," *IEEE Conference on Information, Intelligence, Systems and Applications*, March 2018.

[11] G. Stampa, M. Arias, D. Snchez-Charles, V. Munts, and A. Cabellos-Aparicio, "A deep-reinforcement learning approach for software-defined networking routing optimization," in *The ACM CoNEXT Student Workshop*, Dec 2017.

[12] H. Huang, J. Yang, H. Huang, Y. Song, and G. Gui, "Deep learning for super-resolution channel estimation and doa estimation based massive mimo system," *IEEE Transactions on Vehicular Technology*, vol. 67, pp. 8549–8560, Sep. 2018.

[13] Huang and et al, "Deep Learning for Super-Resolution Channel Estimation and DOA Estimation Based Massive MIMO System," *IEEE VTC*, August 2018.

[14] J. Zhu, Y. Song, D. Jiang, and H. Song, "A new deep-q-learning-based transmission scheduling mechanism for the cognitive internet of things," *IEEE Internet of Things Journal*, pp. 1–1, 2017.

[15] H. Safdar, N. Fisal, R. Ullah, W. Maqbool, F. Asraf, Z. Khalid, and A. Khan, "Resource Allocation for Uplink M2M Communication: A Game Theory Approach," *IEEE Symposium on Wireless Technology and Applications (ISWTA)*, pp. 48–52, September 2013.

[16] J. Brown and J. Y. Khan, "Predictive Allocation of Resources in the LTE Uplink Based on Maximum Likelihood Estimation of Event Propagation Characteristics for M2M Applications ," *IEEE Global Communications Conference (GLOBECOM)*, pp. 4965 – 4970, December 2014.

[17] E. Ahmed and G. Yasser, "BAT: A Balanced Alternating Technique for M2M Uplink Scheduling over LTE," *IEEE Vehicular Technology Conference*, pp. 1–6, May 2015.

[18] H. Wu, C. Zhu, R. J. La, X. Liu, and Y. Zhang, "Fasa: Accelerated s-aloha using access history for event-driven m2m communications," *IEEE/ACM Transactions on Networking*, vol. 21, pp. 1904–1917, Dec 2013.

[19] C. Stefanovic and P. Popovski, " ALOHA Random Access that Operates as a Rateless Code," *IEEE Transactions on Communications*, vol. 61, pp. 4653–4662, November 2013.

[20] T. Park, N. Abuzainab, and W. Saad, "Learning How to Communicate in the Internet of Things: Finite Resources and Heterogeneity ," *IEEE Access; Special Ssction on Optimization for Emerging Wireess Networks: IoT, 5G and Smart Grid Communications*, vol. 4, pp. 7063 – 7073, November 2016.

[21] A. LHeureux, K. Grolinger, H. F. Elyamany, and M. A. M. Capretz, "Machine Learning With Big Data: Challenges and Approaches," *IEEE Access*, vol. 5, pp. 7776 – 7797, April 2017.

[22] W. Cui, Y. Kim, and T. S. Rosing, "Cross-Platform Machine Learning Characterization for Task Allocation in IoT Ecosystems ," *IEEE Computing and Communication Workshop and Conference (CCWC)*, pp. 1 – 7, March 2017.

[23] T. Park and W. Saad, "Learning with Finite Memory for Machine Type Communication ," *IEEE Conference on Information Science and Systems*, pp. 1 – 6, October 2016.

[24] F. Tang, Z. M. Fadlullah, B. Mao, and N. Kato, "An intelligent traffic load prediction-based adaptive channel assignment algorithm in sdn-iot: A deep learning approach," *IEEE Internet of Things Journal*, vol. 5, pp. 5141–5154, Dec 2018.

[25] A. Biral, M. Centenaro, A. Zanella, L. Vangelista, and M. Zorzi, "The challenges of M2M massive access in wireless cellular networks," *ELSEVIER Digital Communications and Networks*, pp. 1–19, March 2015.

[26] M. Nazir and M. Bennis, "Learning based mechanisms for interference mitigation in self-organized femtocell networks," *Asilomar Conference*, pp. 1886 – 1890, November 2010.

[27] K. Peng-Yong and P. Dorin, "Reinforcement learning approach to dynamic activation of base station resources in wireless networks," *IEEE Personal Indoor, Mobile Radio Communications*, pp. 3264 – 3268, September 2013.

[28] X. Chen, J. Wu, and Y. Cai, "Energy-Efficiency Oriented Traffic Offloading in Wireless Networks: A Brief Survey and a Learning Approach for Heterogeneous Cellular Networks ," *IEEE Journal on Selected Areas in Communications*, pp. 627–640, January 2015.

[29] A.-H. Mohammed and A. Anpalagan, "Base Station Selection in M2M Communication Using Q-learning Algorithm in LTE-A Networks," *In Proc. IEEE International Conference on Advanced Information Networking and Applications*, 2015.

[30] M. Hasan and E. Hossain, "Random Access for Machine-to-Machine Communication in LTE-Advanced Networks: Issues and Approaches," *ieeecm*, pp. 90– 97, June 2013.

[31] M. A. Alsheikh, D. Niyato, and H.-P. T. Lin, "Machine Learning in Wireless Sensor Networks: Algorithms, Strategies, and Applications," *IEEE Communication Surveys and Tutorials*, vol. 16, pp. 1996– 2018, November 2014.

[32] Y. Chu, P. Mitchell, and D. Grace, "ALOHA and Q-Learning Based Medium Access Control for Wireless Sensor Networks," *International Symposium on Wireless Communication Systems*, pp. 511–515, August 2012.

[33] G. Serrano and L. Giupponi, "Distributed Q-learning for aggregated interference control in cognitive radio networks," *IEEE Transactions on Vehicular Technology*, vol. 59, pp. 1823–1834, May 2010.

[34] C. Wu and M. D. Felice, "Spectrum management of cognitive radio using multi-agent reinforcement learning," *International Conference on Autonomous Agents and Multiagent Systems*, pp. 1705–1712, 2010.

[35] M. Benis and D. Niyato, "A Q -learning Based Approach to Interference Avoidance in Self-Organized Femtocell Networks," *IEEE Globecom*, pp. 706–710, October 2010.

[36] B. L. Mohammed and M. P. D. Grace, "Application of Q-Learning for RACH Access to Support M2M Traffic over a Cellular Network," *IEEE Proceedings of European Wireless Conference*, pp. 1–6, May 2014.

[37] B. L. Mohammed and M. P. D. Grace, "Frame based Back-off for Q-learning RACH access in LTE networks," *Australasian Telecommunication Networks and Applications Conference (ATNAC)*, pp. 176–181, November 2014.

[38] F. Hussain and A. Ferworn, "Distributed Slot Allocation in Capillary Gateways for Internet of Things Networks.," *IEEE VTC*, September 2016.

[39] L. Rucco, A. Bonarini, C. Brandolese, and W. Fornaciari, "A bird's eye view on reinforcement learning approaches for power management in WSNs ," *IEEE Wireless and Mobile Networking Conference (WMNC)*, pp. 1–8, April 2013.

[40] N. J. Patel and R. H. Jhaveri, "Detecting packet dropping nodes using machine learning techniques in Mobile ad-hoc network: A survey ," *IEEE International Conference on Signal Processing And Communication Engineering Systems (SPACES)*, pp. 468–472, January 2015.

[41] C. Claus and C. Boutilier, "The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems ," *American Association for Artificial Intelligence*, 2000.

[42] W. Qiang and Z. Zhongli, "Reinforcement learning model, algorithms and its application," *IEEE International Conference on Mechatronic Science, Electric Engineering and Computer*, pp. 1143–1146, August 2011.

[43] S. N. Sasikumar, *Exploration in Feature Space for Reinforcement Learning*. PhD thesis, Australian National University, 2017.

BIOGRAPHIES

Fatima Hussain [SM] is working as Security Analyst in "API Security and Governance" squad, Royal Bank of Canada (RBC), Toronto Canada. She is leading the development and promotion of new APIs and API development learning curriculum along with API security and governance duties. She is the technical lead in Ryerson-RBC research collaborative lab initiative. She is also an Adjunct Professor at Ryerson University, Toronto and her role includes the supervision of graduate research projects. Dr Hussain's background includes number of distinguished professorships at Ryerson University and University of Guelph where she has been awarded for her research, teaching and course development accomplishments within Wireless Telecommunication and Internet of Things. Her research interests include API Security, Cyber Security and Machine Learning. She is a prolific author with various conference and journal publications to her credit. Dr. Hussain received her PhD and MASc degrees in Electrical and Computer Engineering from Ryerson University, Toronto. Upon graduation she joined the Network-Centric Applied Research Team (N-CART) as a post doctoral fellow where she worked on various NSERC-funded projects in the realm of the Internet of Things.

Rasheed Hussain [SM] is currently working as an Associate Professor and Director of Institute of Information Security and Cyber-Physical Systems at Innopolis University, Innopolis, Russia. He is also the Director of Networks and Blockchain Lab at Innopolis University. Furthermore, he is ACM Distinguished Speaker, senior member of IEEE, member of ACM, and certified trainer for the Instructional Skills Workshop (ISW). His research interests include Information Security and Privacy, Vehicular Ad Hoc NETworks (VANETs), vehicular clouds, and vehicular social networking, applied cryptography, Internet of Things, Content-Centric Networking (CCN), cloud computing, blockchain, and Machine Learning in cybersecurity.

Alagan Anpalagan (S98-M01-SM04) is a Professor in the ELCE Department, Ryerson University, Canada. He served the department in administrative positions as Associate Chair, Program Director for Electrical Engineering, and Graduate Program Director. He directs a research group working on radio resource management, and radio access and networking areas within the WINCORE Lab. He currently serves as Vice Chair, IEEE SIG on Green and Sustainable Networking and Computing with Cognition and Cooperation. Dr. Anpalagan was the recipient of the IEEE Canada J.M. Ham Outstanding Engineering Educator Award (2018), SGS Outstanding Contribution to Graduate Education Award (2017), Deans Teaching Award (2011), Faculty Scholastic, Research and Creativity Awards (2019) from the Ryerson University. He is a Fellow of the Institution of Engineering and Technology (FIET) and Fellow of the Engineering Institute of Canada (FEIC).

Abderrahim Benslimane [SM] is Full Professor of Computer-Science at the Avignon University/France since 2001. He has the French award for Doctoral supervision and Research 2017-2021. He has been recently an International Expert at the French Ministry of Foreign and European affairs (2012-2016). He served as a coordinator of the Faculty of Engineering and head of the Research Center in Informatics at the French University in Egypt. He was attributed the French award of Scientific Excellency (2011-2014). He has been as Associate Professor at the University of Technology of Belfort-Montbliard since September 1994. He obtained the title to supervise researches (HDR 2000) from the University of Cergy-Pontoise, France. He received the PhD degree (1993), DEA (MS 1989) from the Franche-Comte University of Besanon, and BS (1987) from the University of Nancy, all in Computer Science. He is Chair of the ComSoc Technical Committee of Communication and Information Security. He is EiC of Inderscience Int. J. of Multimedia Intelligence and Security (IJMIS), Area Editor of Security in IEEE IoT journal, Area Editor of Wiley Security and Privacy journal and editorial member of IEEE Wireless Communication Magazine, Elsevier Ad Hoc, IEEE Systems and Wireless Networks Journals. He is founder and serves as General-Chair of the IEEE WiMob since 2005 and of iCOST and MoWNet international conference since 2011. He served as a Symposium co-chair/leader in many IEEE international conferences such as ICC, Globecom, AINA and VTC. He was Guest Editor of many special issues. He participates to the steering and the program committee of many IEEE international conferences. He was Board committee member, Vice-chair of Student activities of IEEE France section/Region 8; he was Publication Vice-chair and Conference Vice-Chair of the ComSoc TC of Communication and Information Security. He participates to the steering and the program committee of many IEEE international conferences.