Distributed Learning for Low Latency Machine Type Communication in a Massive Internet of Things

Taehyeun Park\textsuperscript{1} and Walid Saad\textsuperscript{1}
\textsuperscript{1}Wireless@VT, Bradley Department of Electrical and Computer Engineering, Virginia Tech, Blacksburg, VA, USA,
Emails:\{taehyeun, walids\}@vt.edu

Abstract

The Internet of Things (IoT) will encompass a massive number of machine type devices that must wirelessly transmit, in near real-time, a diverse set of messages sensed from their environment. Designing resource allocation schemes to support such coexistent, heterogeneous communication is hence a key IoT challenge. In particular, there is a need for self-organizing resource allocation solutions that can account for unique IoT features, such as massive scale and stringent resource constraints. In this paper, a novel \textit{finite memory multi-state sequential learning} framework is proposed to enable diverse IoT devices to share limited communication resources, while transmitting both delay-tolerant, periodic messages and urgent, critical messages. The proposed learning framework enables the IoT devices to learn the number of critical messages and to reallocate the communication resources for the periodic messages to be used for the critical messages. Furthermore, the proposed learning framework explicitly accounts for IoT device limitations in terms of memory and computational capabilities. The convergence of the proposed learning framework is proved, and the lowest expected delay that the IoT devices can achieve using this learning framework is derived. Furthermore, the effectiveness of the proposed learning algorithm in IoT networks with different delay targets, network densities, probabilities of detection, and memory sizes is analyzed in terms of the probability of a successful random access request and percentage of devices that learned correctly. Simulation results show that, for a delay threshold of 1.25 ms, the average achieved delay is 0.71 ms and the delay threshold is satisfied with probability 0.87. Moreover, for a massive network, a delay threshold of 2.5 ms is satisfied with probability 0.92. The results also show that the proposed learning algorithm is very effective in reducing the delay of urgent, critical messages by intelligently reallocating the communication resources allocated to the delay-tolerant, periodic messages.

I. INTRODUCTION

The Internet of Things (IoT) is an emerging networking technology that promises to intercon-
nect a massive number of devices such as wearables, sensors, smartphones, and other machine type devices [2]. The IoT will impact multiple application domains including home automation [3], smart grids [4], [5], drone-based systems [6], and industrial monitoring [7], [8]. To support such innovative IoT applications, there is a need for new wireless technologies that can enable a large-scale connectivity among IoT devices. However, integrating the IoT ecosystem into existing wireless networks faces many challenges that include coexistence with human-type devices, self-organizing operation, and limited communication resources [9]. Moreover, the IoT devices are typically machine-type devices that differ significantly from conventional human-type devices, such as smartphones, in terms of performance requirements, memory, computation, and energy constraints as well as traffic patterns [2]. In addition, the IoT devices will require low latency, ultra-reliable, and short packet transmissions. Therefore, existing wireless networks must be re-designed to meet these IoT challenges.

The IoT devices deployed in an existing wireless network will need to deliver diverse applications and, thus, they will be heterogeneous in terms of their performance requirements and traffic patterns. For example, IoT devices used for smart metering may not require ultra-low latency and will communicate periodically. In contrast, IoT devices used for industrial monitoring will require ultra-low latency and communicate sporadically. Furthermore, such heterogeneous IoT devices with different properties and requirements will have to coexist and share limited communication resources appropriately to satisfy their quality-of-service (QoS) requirements. To incorporate such heterogeneous IoT devices into existing wireless networks with limited resources, a distributed method is necessary to allocate limited resources appropriately depending on the QoS needs of the IoT devices. The IoT devices must be able to obtain their communication resources autonomously, because it is impractical to assume that they can communicate frequently with the base station, given their stringent resource constraints. Moreover, in a massive IoT, the base station will not be able to manage the communication resources of all devices in a timely manner. Therefore, resource allocation in the IoT must be distributed and must also consider the limited capabilities of the IoT devices in terms of computation and memory. A resource allocation framework satisfying the aforementioned requirements will accelerate the deployment of the IoT over existing networks with limited communication resources.

Resource allocation in the IoT has attracted significant research recently [10]–[19]. For instance, different multiple access schemes have been proposed in order to maximize energy
efficiency [10] or throughput [11]. Furthermore, non-orthogonal multiple access schemes for IoT networks are analyzed in [12], in terms of channel capacity, latency, and connectivity. The authors in [13] propose a framework based on cognitive edge computing to optimize the use of distributed cloud resources for the IoT. In [14] and [15], the authors propose different methods to integrate IoT devices into existing cellular networks and share communication resources with existing devices while avoiding network congestion and minimizing system capacity in a massive network. Meanwhile, the work in [17] introduces new communication protocols for IoT applications that require ultra-reliability and ultra-low latency. Moreover, the authors in [18] propose a resource allocation framework exploiting cloud radio access network for joint channel selection and power allocation. In [19], the authors compare the performance of dynamic time division duplex with centralized and distributed resource allocation schemes in a dense IoT. The authors in [20] provide an overview on resource allocation challenges and random access for narrow band IoT. In [21], we developed a qualitative survey that discusses the potential of learning in the IoT. A framework based on sequential learning to learn the number of urgent, critical messages and to allocate corresponding uplink resources is studied in [16]. Although message type heterogeneity and the resource constraints of the IoT devices are considered in [16], this work requires a learning sequence for each critical message. Therefore, an IoT device must be part of all learning sequences to accurately learn the number of critical messages, which is impractical for resource-constrained IoT devices. Meanwhile, the works in [15], [17], and [19] do not account for the heterogeneity in communication requirements, traffic patterns, and message types. Moreover, most of the prior art in [10], [14], [15], and [18] relies on centralized solutions, which may be impractical for a massive IoT. Further, even though the energy constraints of IoT devices are typically considered, such as in [10] and [18], the IoT devices also have limited capabilities in terms of memory and computation, which are ignored in existing works, [10], [11], [14], [18], and [19].

One promising approach to resource management in the IoT is via the use of learning tools [21]. In particular, sequential learning models [22] can offer distributed solutions using which the IoT devices can seize limited communication resources depending on their communication requirements and their computational and memory constraints. Sequential learning algorithms can use a series of observations on an environment to learn its hidden binary state [22]. Furthermore, sequential learning can enable the IoT devices to learn a hidden binary state using only a finite
number of observations, and this is called finite memory sequential learning. The hidden binary state in an IoT can capture the existence of a critical message, which will cause the IoT devices to properly adjust their resource usage to minimize the delay of critical messages. In an IoT, there may be multiple critical messages simultaneously, and knowing the number of critical messages is necessary to enable the system to allocate the scarce communication resources appropriately. However, the number of critical messages cannot be learned using existing finite memory sequential learning schemes such as in [22]. Although sequential learning is suitable for IoT resource allocation, it cannot be readily applied to the IoT as it is limited to binary state learning. Therefore, there is a need to extend existing binary-state learning algorithms to multi-state learning algorithms suitable for the IoT.

The main contribution of this paper is a multi-state sequential learning framework with finite memory that can enable the IoT devices to allocate the limited communication resources in a self-organizing manner while satisfying their heterogeneous latency requirements. In particular, we consider a massive IoT in which there are IoT devices with limited capabilities transmitting either periodic or critical messages. The periodic messages, such as meter readings, are delay-tolerant messages with predictable traffic pattern, while the critical messages, such as system failures, are randomly occurring, delay-intolerant messages. When some devices transmit critical messages, the communication resources that are normally allocated for the periodic messages must be reallocated for the critical messages for their timely transmission. We propose a novel multi-state sequential learning for the IoT devices to learn the number of existing critical messages and, then, to reallocate communication resources appropriately depending on the delay requirement of critical messages. The proposed multi-state sequential learning is suitable for the IoT devices as it only requires finite number of observations to learn in a distributed way. Then, we analyze the memory, limited observation capabilities of devices, and expected delay resulting from our proposed algorithm. We show that our proposed learning framework converges to the true state of the environment (i.e., it autonomously learns the true number of existing critical messages). We also derive the lowest expected delay of a critical message that can be achieved with our learning framework. Simulation results show the learning effectiveness of the proposed framework as well as its ability to realize low-latency transmissions across a massive IoT network.

The rest of this paper is organized as follows. Section II introduces the system model and Section III presents the proposed learning algorithm and analyzes its properties. Section IV
analyzes the simulation results, while Section V draws conclusions.

II. SYSTEM MODEL

Consider the uplink of a wireless IoT system consisting of one base station (BS) serving $N$ IoT devices. To transmit their messages to the BS, the IoT devices must first request uplink communication resources via a time-slotted random access channel (RACH) using one of $p$ random access (RA) preambles [23] that are typically divided into two types: $p_c$ contention-based RA preambles (RAPs) and $p_f = p - p_c$ contention-free RAPs [23]. For using a contention-based RAP, the IoT devices choose one out of the $p_c$ preambles randomly. However, if more than one IoT device choose a given contention-based RAP, there will be a collision resulting in a failure to request uplink resources. Therefore, a device will successfully be allocated uplink resources using contention-based RAP, only if it is the sole device using that preamble at that given time slot. For using contention-free RAPs, the devices are assigned one of the $p_f = p - p_c$ preambles by the BS. The preamble usage and allocation among the IoT devices will depend on their message types as discussed later in this section.

One of the prominent features of the IoT is heterogeneity in terms of device types, functionalities, and transmitted messages. For instance, some IoT devices, such as smart meters and environment sensors, will be periodically transmitting small packets, such as meter readings, observation reports, and system status reports. However, such devices may need to also transmit urgent, critical messages when necessary, such as critical system state condition, power outage, and fire detection reports. These message types differ in terms of quality-of-service (QoS) requirements, such as delay, and traffic patterns, such as periodic or bursty. To model heterogeneous messages, we consider the co-existence of periodic messages, which are infrequent, periodic transmissions, and critical messages, which are critical, urgent transmissions that require very small delay. Depending on the message type, the IoT devices will transmit using most appropriate RAP type, because different RAPs have different properties.

Since periodic messages are non-critical and recurring, they are typically delay-tolerant with predictable traffic patterns [5]. Hence, the IoT devices transmitting periodic messages will request uplink resources using an RAP once every period $T$, and their traffic pattern will be predictable. We let $\tau_i$ for $i = 1, \cdots, T$ be the possible time slots when the IoT devices first transmit their periodic messages. The devices that first transmit in slot $\tau_i$ will transmit in slots $\tau_i + kT$
for \( k \in \mathbb{Z}_+ \). Moreover, the IoT devices that first transmit simultaneously will always transmit simultaneously using different RAPs, and, thus, it is sufficient to only consider the RAP allocation among the IoT devices that have the same \( \tau_i \).

Contention-free RAPs are more suitable for periodic message transmissions than contention-based RAPs, because they can guarantee successful uplink resource request for appropriate values of \( T \). If \( p_f \) contention-free RAPs are used to transmit periodic messages, the minimum period \( T_{\text{min}} \) of the periodic messages that can be achieved is:

\[
T_{\text{min}} = \left\lceil \frac{N}{p_f} \right\rceil,
\]

and \( T_{\text{min}} \) can be achieved when an approximately equal number of devices transmit periodic messages in each slot. For \( T \geq T_{\text{min}} \), at most \( p_f \) IoT devices will be transmitting at any given slot to avoid collisions. If the required period of the periodic messages is less than \( T_{\text{min}} \), there will be more than \( p_f \) IoT devices transmitting simultaneously in some time slots and collisions will occur. For \( T \geq T_{\text{min}} \), the BS will allocate different contention-free RAPs to IoT devices with equal \( \tau_i \) and may allocate the same contention-free RAP to multiple IoT devices with different \( \tau_i \). This allocation scheme helps avoiding RAP collision, while maximizing the number of IoT devices that can request uplink resources. The use of contention-free RAPs takes advantage of the predictable traffic patterns of the periodic messages and guarantees that the IoT devices successfully request their uplink resources.

Note that contention-based RAPs are not suitable for the periodic messages, because the probability of successful transmission is low, particularly for a massive IoT with large \( N \). Assuming that \( \tau_i \) are chosen randomly, if the IoT devices transmitting periodic messages use contention-based RAPs, the expected number \( N_p \) of IoT devices transmitting periodic messages at any given time slot will be \( N_p = \frac{N}{T} \). The probability \( p_{p,s} \) of a IoT device successfully transmitting the contention-based RAP in a given time slot will then be:

\[
p_{p,s} = \left( \frac{p_c - 1}{p_c} \right) \left( \frac{N_p - 1}{N_p} \right) .
\]

Since the IoT will be massive (i.e., \( N \) is large), then, \( p_{p,s} \) is small, and, thus, only few, if any, periodic messages will be successfully transmitting in a given time slot. For periodic messages with \( T \geq T_{\text{min}} \), the use of contention-free RAPs guarantees that the IoT devices successfully request uplink resources, while the probability of successful request is low using contention-based
RAPs. Therefore, it is more advantageous for the IoT devices transmitting periodic messages to use contention-free RAPs. Hereinafter, we assume that the period $T$ of periodic messages is $T_{\text{min}}$ and that $N$ is a multiple of $p_f$.

Next, for critical IoT device messages, since they contain critically information about an abnormal event, it is necessary to have them reach the BS in a timely manner. Therefore, critical messages are delay-intolerant messages that need retransmissions in case of collision. Moreover, since critical messages are triggered by an unpredictable abnormal event, such as system failure or forest fire, the system cannot know beforehand when and which IoT devices will have critical messages to transmit. Furthermore, critical messages can be highly correlated since an abnormal event will typically trigger many critical messages simultaneously. For instance, a forest fire will trigger various devices monitoring different environmental parameters, such as temperature, humidity, or carbon monoxide. Since the network cannot predict when the critical messages will be triggered, how many will be triggered, and which IoT devices will have critical messages to transmit, the BS cannot assign pre-determined RAPs, and, thus, IoT devices with critical messages must use contention-based RAPs to request uplink transmission resources.

We let $N_a$ be the number of IoT devices having critical messages to send. For IoT devices with critical messages using $p_c$ contention-based RAPs, the probability $p_{a,s}$ of a device being successfully allocated an uplink resource for sending a critical message will be:

$$p_{a,s} = \left( \frac{p_c - 1}{p_c} \right)^{N_a - 1}.$$  (3)

Given $p_{a,s}$, the expected delay $D_c$ of a critical message under a contention-based RAP is defined as the expected number of time slots needed until the first successful acquisition of an uplink communication resource using contention-based RAP, as follows:

$$D_c = \left( \frac{p_c}{p_c - 1} \right)^{N_a - 1}.$$  (4)

Since periodic messages are always transmitted and critical messages may not exist, it is inefficient for $p_c$ to be large, because $T_{\text{min}}$ in (1) will be large with small $p_f$. Therefore, the number of contention-based RAPs $p_c$ will generally be chosen to be small. Moreover, critical messages are correlated, which means that an abnormal event will trigger many critical messages, and, thus, the value of $N_a$ may be large. Therefore, the expected delay $D_c$ can be long, but critical messages must be delivered with low latency given their urgent nature. For an IoT
device \( i \) having a critical message, the BS will allocate a contention-free RAP for time slots \( \tau_i + kT \) for \( k \in \mathbb{Z}_+ \) for periodic message transmission, since the BS does not know beforehand about the critical message. In such cases, instead of using the assigned contention-free RAP for the periodic message, IoT devices with critical messages will use the assigned contention-free RAP for sending their critical messages. Since \( T = T_{\text{min}} \) and \( \tau_i \) are chosen randomly for all devices, the expected delay \( D_f \) of a critical message under a contention-free RAP is defined as the expected number of time slots needed until an IoT device with critical message is allocated with contention-free RAP, as follows:

\[
D_f = \frac{1}{T_{\text{min}}} \sum_{j=0}^{T_{\text{min}}-1} j = \frac{T_{\text{min}} - 1}{2}.
\]  

(5)

Therefore, the expected delay of a critical message is \( D = \min(D_c, D_f) \).

To minimize the expected delay \( D \) of the critical messages, the value of \( D_c \) must be minimized as \( D_f \) is determined by system parameters. Moreover, in a massive IoT network with large \( N \) and limited communication resources with small \( p \), \( T_{\text{min}} \) will be large, and, thus, \( D_f \) will also be large. Since the value of \( N_a \) cannot be controlled, the value of \( p_c \) must be maximized to minimize \( D_c \). Since the number of RAPs \( p \) is fixed, the only way to increase the value of \( p_c \) is to reallocate RAPs from contention-free to contention-based. The method used for such reallocation cannot be centralized since a centralized method requires the BS to have already received the critical messages and to be aware of the abnormal event, which is impractical. Therefore, RAP reallocation must be done in a distributed manner.

To reallocate an appropriate number of contention-free RAPs, the IoT devices must be able to observe the number of critical messages \( N_a \) in the system. We assume that the overhead signaling for the periodic messages and critical messages are different such that the IoT devices can observe the number of critical messages \( N_a \) being transmitted in a given time slot [24]. However, the IoT devices may not be able to accurately observe \( N_a \), since they cannot observe all of the critical messages in the network. For instance, an IoT device that is located far from the abnormal event may not be able to observe the event and its corresponding critical messages. IoT devices may also be prone to missed detection even when they are in proximity to the abnormal event. To capture the limited observation capability of the IoT devices, we assume that IoT devices within a detection range \( r_d \) have a probability of missed detection \( p_{01} \), while IoT devices outside of the detection range \( r_d \) have a much higher probability of missed detection,
In order for the IoT devices to accurately know $N_a$ despite their limited observation capability, they can employ a distributed learning process \cite{25} using which the devices can collectively learn the number of critical messages $N_a$ in a self-organizing manner. For instance, IoT devices with limited observation capabilities can use the observations of other IoT devices to have a more accurate estimate of $N_a$. For the reallocation of RAPs, we assume that there is a globally known sequence of contention-free RAPs that will be reallocated to contention-based RAPs. When an IoT device having a periodic message learns that there are $N_a$ critical messages, it will stop using the first $\beta$ contention-free RAPs of the sequence. This means that even if the IoT device with periodic message is assigned to use one of the $\beta$ RAPs by the BS, it will not transmit using any RAP that has been reallocated to contention-based RAPs. When an IoT device having a critical message learns that there are $N_a$ critical messages, it may transmit using the first $\beta$ RAPs of the sequence of contention-free RAPs in addition to the contention-based RAPs. Therefore, the number of contention-based RAPs will increase from $p_c$ to $p'_c = p_c + \beta$, and the expected delay $D'_c$ from using contention-based RAP and the expected delay $D'$ of critical message after learning will, respectively, be given by:

$$D' = \min(D'_c, D_f) = \min\left(\left(\frac{p'_c}{p_c - 1}\right)^{N_a}, D_f\right).$$

(6)

The value of $\beta$ is determined to satisfy $D' \leq D_{th}$, where $D_{th}$ is a design parameter and a threshold delay that the critical messages should satisfy.

To develop an effective learning approach, one must take into account specific characteristics of the IoT devices. As the IoT devices are small, low cost devices, their computational capability will be limited. Moreover, they are constrained by very limited battery life, and, thus, they cannot communicate frequently with the BS or with other IoT devices. Further, the IoT devices may also be limited in terms of how much information they can store and process. Therefore, they may not have sufficient information for accurate learning, and their learning method must account for the lack of memory. Also, the IoT devices may have inaccurate information about $N_a$ due to their limited observability of the environment. For applications that require low latency, the expected delay of critical messages must be reduced in relatively real time, and, thus, the IoT devices must quickly and accurately learn $N_a$. Therefore, the learning scheme used by the IoT devices must be computationally simple, distributed, quick, and accurate without requiring excessive memory.
and frequent communication, as developed next.

III. Finite Memory Multi-state Sequential Learning

For our model, the IoT devices must learn to satisfy their heterogeneous QoS requirements, in presence of both known periodic messages and unknown critical messages. To this end, we propose a novel multi-state sequential learning framework for enabling the IoT devices to learn, given their inherent limited memory and computation capabilities. However, even with an appropriate learning method, some IoT devices may not learn the true value of $N_a$ due to their limited observability and lack of memory. The IoT devices learning the true value of $N_a$ with limitations can be mapped to agents learning the true state of a system with multiple states. For instance, the different values of $N_a$ that the IoT devices can learn can be mapped to different system states, while the true value of $N_a$ is the true, underlying state that the IoT devices should learn. By mapping different values of $N_a$ to different states, we propose a multi-state learning framework for dynamic reallocation of RAPs, while taking into account the requirements of the IoT devices. Furthermore, our learning method will require an IoT device to only communicate with one neighboring device within the communication range $r_c$ to obtain a finite memory of observations of other devices.

As shown in Fig. 1, we propose a learning algorithm composed of two phases, which are used to choose a state and to test whether the chosen state is the true underlying state $H_T$. A state is chosen out of all possible states during the first phase, and the chosen state is called a favored state. During the second phase, it is repeatedly tested whether the favored state should be changed by going back to the first phase. The sequential learning is implemented individually by all devices, and the IoT devices will learn as they get necessary information. Moreover, an IoT device belongs to either first phase or second phase as an IoT device learns only once, and the phase that an IoT device belongs to will depend on the learning sequence. In our model, the IoT devices learn the number of critical messages based on their observations and the observations of neighboring devices during the first phase. During the second phase, the IoT devices will be informed about the state $N_a$ learned during the first phase and will repeatedly test whether the learned value of $N_a$ is a true value or not. If it is determined that the learned value is not the true value of $N_a$, then the learning method reverts back to the first phase. For both phases, the observations are critical in learning correctly, and more observations typically result
in higher probability of learning correctly. However, it is unrealistic to assume that an IoT device is able to know, store, and process observations of all other IoT devices, which corresponds to infinite memory sequential learning [22]. Therefore, our proposed learning framework has finite memory [25] as it assumes that an IoT device only utilizes some of the observations of other IoT devices. In other words, the finite memory learning requires finite number of observations, while the infinite memory learning requires infinite number of observations. The convergence of our finite memory learning framework is shown in Subsection III.C.

A. First Phase of Sequential Learning

For an appropriate reallocation of contention-free RAPs, the IoT devices must learn the true value of the number of critical messages $N_a$. However, the devices may learn wrong values of $N_a$ due to limited observability. We let $\mathcal{S}$ be the set of all possible values of the state $N_a$. Here, the true value of $N_a$ is the true underlying state $H_T \in \mathcal{S}$. During the first phase, the IoT devices chose a favored state $s_f \in \mathcal{S}$ after $K$ observations, which provides information about $H_T$. $s_f$ is a value of $N_a$ that the devices learn to be true. In our model, the IoT devices observe how many critical messages exist depending on their limited observation range $r_d$. For simplicity, we model an observation of the environment $e$ to be an element of $\mathcal{S}$ such that, for any given $H_T \in \mathcal{S}$,

$$0 < \frac{\Pr(e = j \mid H_T)}{\Pr(e = k \mid H_T)} < \infty \forall j, k \in \mathcal{S},$$

$$\Pr(e = H_T \mid H_T) \gg \Pr(e \neq H_T \mid H_T), \quad \Pr(e = i \mid i) \gg \Pr(e = i \mid j) \forall i, j \in \mathcal{S},$$

$$\Pr(e = j \mid i) \approx \Pr(e = k \mid i) \forall i, j, k \in \mathcal{S},$$

where $\Pr(e = i \mid j)$ is probability of observing $i$ when $j$ is the true underlying state. (7) is necessary to ensure that the learning process is not trivial by preventing the likelihood ratio of
observations from being 0 or $\infty$. In (8), we assume that a given device is more likely to observe the true underlying state $H_T$ than any other state in $\mathcal{S}$ and that it is most likely to observe $i \in \mathcal{S}$ when the true underlying state is $i$. Furthermore, in (9), the probabilities of observing the states that are not true underlying state are approximately equal, and we assume that the observations of the environment are independent.

The first phase is initiated by an abnormal event, and the devices that are first to observe the abnormal event will have to transmit critical messages. Furthermore, those devices propagate the learning sequence to their neighboring devices within $r_c$ by transmitting their observations of the environment. Those neighboring devices constitute the next set of devices that learn in the first phase, and these devices learn $N_a$ based on their observations and the information from the device that learned first. Therefore, the first phase propagates sequentially to the various IoT devices within $r_c$. For each device, the learning sequence for the first phase stops when this device obtains $K$ observations and then initiates the second phase by choosing $s_f$ based on $K$ observations. To prevent the learning sequence from oscillating between IoT devices, an IoT device that already learned $N_a$ will not learn again and no longer participate in any other learning sequence. If an IoT device that has not yet learned receives the necessary information for learning from more than one neighboring device simultaneously, then it will arbitrarily choose one of the learning sequences to continue.

We assume that $r_c < r_d$ as the communication links between IoT devices are usually short-ranged. Moreover, it is desirable for $K$ to be small so that $s_f$ is chosen quickly and that the learning method converges faster to $H_T$. For small $K$, the memory and computational requirements for the IoT devices will also be small due to the reduced amount of information that must be stored and processed, and, thus, the critical messages will be successfully transmitted with small delay as the learning converges faster to $H_T$. Hence, the IoT devices during the first phase will most likely be within $r_d$. Since the probability of missed detection $p_{01}$ within $r_d$ is usually close to 0 [26], the IoT devices participating in the first phase are most likely to observe $e = H_T$. Therefore, (8) holds true for the IoT devices within $r_d$, where the first phase always occurs.

While the IoT devices accumulate $K$ independent observations of the environment, they can make an intelligent decision on which state is most likely to be the true underlying state $H_T$ (i.e., the true value of $N_a$). This decision will be based on maximum likelihood, whose computational
complexity can be simplified given (8). For a set of $K$ observations $\{e_1, \cdots, e_K\}$, the favored state $s_f$ will be:

$$s_f = \arg\max_{s \in S} \text{Pr}(e_1, \cdots, e_K \mid s) = \arg\max_{s \in S} \prod_{j=1}^{K} \text{Pr}(e_j \mid s),$$

(10)

$$= \arg\max_{s \in S} \prod_{j \in S} \text{Pr}(j \mid s)^{k_j} = \arg\max_{s \in S} \sum_{j \in S} k_j \ln(\text{Pr}(j \mid s)),$$

(11)

where $k_j$ is the number of times that state $j \in S$ is observed out of $K$ observations. If the likelihoods $\text{Pr}(e = i \mid j) \forall i, j \in S$ are known by the devices, then $s_f$ can be determined easily using (11). However, when the likelihoods $\text{Pr}(e = i \mid j) \forall i, j \in S$ are unknown, the maximum likelihood can be simplified using (8) and (9) and, then, the favored state $s_f$ will be $a \in S$ if:

$$\sum_{j \in S} k_j \ln(\text{Pr}(j \mid a)) \approx \sum_{j \in S, j \neq a} k_j = K - k_a \leq 1 \forall b \in S.$$

(12)

Therefore, with simplifications, the decision process in the first phase will be such that the devices will be able to designate one of the observed states as a favored state $s_f$.

B. Second Phase of Sequential Learning

Once the IoT devices choose one of the observed values of $N_a$ to be the true value of $N_a$, the IoT devices must repeatedly test whether the chosen value is the truth, given the possibility of learning error due to limited observability. In terms of learning, the favored state $s_f$ chosen in the first phase is repeatedly tested to determine whether $s_f$ is $H_T$, and, if it is found that $s_f \neq H_T$, then $s_f$ must be changed to some other state in $S' = S \setminus \{s_f\}$. Since the second phase tests whether $s_f$ must be changed or not, this can be modeled as binary hypothesis testing between $s_f = H_T$ and $s = H_T$ for some state $s \neq s_f$, $s \in S'$, and the IoT devices in second phase each test with their available information. Binary hypothesis testing uses binary environmental observations to repeatedly test whether the favored hypothesis should be changed from 0 to 1 or from 1 to 0. The favored hypothesis will converge to the underlying truth by design.

As our learning method is multi-state, the devices will observe more than two states. However, if a device observes any state in $S'$, it will be equivalent to observing an unfavored state $s'_f$. Therefore, the number of possible observations reduces to two: $s_f$ and $s'_f$, during the second phase. Here, observing $s_f$ is equivalent to 1 and observing $s'_f$ is equivalent to 0. Similar to the
observations in first phase, an environment observation \( e \) in the second phase is such that, for an underlying true state \( H_T \in S \),
\[
\Pr(e = s_f \mid H_T) + \Pr(e \in S' \mid H_T) = 1 \quad \text{and} \quad 0 < \frac{\Pr(e = s_f \mid H_T)}{\Pr(e = s'_f \mid H_T)} < \infty. \tag{13}
\]

If \( s_f \neq H_T \) is chosen during the first phase and the environment observations repeatedly indicate \( s'_f \), then the favored state must be changed. Therefore, a mechanism for returning to the first phase to adjust the choice of \( s_f \) is necessary. We set a threshold \( \alpha \) such that if the environment observations indicate \( s'_f \) for a total of \( \alpha \) consecutive times, then the learning method will go back to the first phase to choose \( s_f \) again with some probability \( p_b \), which will be further discussed later. Using a threshold, our learning method may go back to the first phase both when \( s_f = H_T \) and when \( s_f \neq H_T \), and the probabilities of returning to the first phase for both cases are critical for the convergence of the proposed learning method.

The probability of observing \( s'_f \) for \( \alpha \) consecutive times when \( s_f \neq H_T \) is
\[
\Pr\{\{e_{n_1-\alpha+1}, \ldots, e_{n_1}\} = \{s'_f, \ldots, s'_f\} \forall n_1 \geq \alpha \mid s_f \neq H_T\} = (1 - \Pr(e = s_f \mid s_f \neq H_T))^\alpha, \tag{14}
\]
where \( e_j \) is \( j \)-th observation of the environment and \( n_1 \in \mathbb{Z}_+ \) is the number of observations necessary for \( \alpha \) consecutive observations of \( s'_f \) to occur when \( s_f \neq H_T \). The number of observations \( n_1 \) necessary to go back to the first phase when \( s_f \neq H_T \) is a random variable and will be further discussed later. Moreover, it is essential that the learning method quickly reverts to the first phase if \( s'_f \neq H_T \) and, thus, the expected number of observations needed to observe \( s'_f \) for \( \alpha \) consecutive times is of high interest. With \( p_1 = \Pr(e = s_f \mid s_f \neq H_T) \), the expected value of \( n_1 \), for any environment observation \( e \), will be:
\[
\mathbb{E}[n_1 \mid s_f \neq H_T] = p_1 \left( \sum_{j=0}^{\alpha-1} (1 - p_1)^j \left( \mathbb{E}[n_1 \mid s_f \neq H_T] + j + 1 \right) \right) + \alpha(1 - p_1)^\alpha, \tag{15}
\]

\[
= \mathbb{E}[n_1 \mid s_f \neq H_T]p_1 \left( \sum_{j=0}^{\alpha-1} (1 - p_1)^j \right) + p_1 \left( \sum_{j=0}^{\alpha-1} (1 - p_1)^j(j + 1) \right) + \alpha(1 - p_1)^\alpha, \tag{16}
\]

\[
= \mathbb{E}[n_1 \mid s_f \neq H_T]p_1 \frac{1 - (1 - p_1)^\alpha}{1 - (1 - p_1)} + p_1 \left( \sum_{j=0}^{\alpha-1} (1 - p_1)^j(j + 1) \right) + \alpha(1 - p_1)^\alpha, \tag{17}
\]
\[
p_1 \left( \sum_{j=0}^{\alpha-1} (1-p_1)^j (j+1) \right) + \alpha (1-p_1)^\alpha
= \frac{\left( \sum_{j=0}^{\alpha-1} (1-p_1)^j (j+1) \right) + \alpha (1-p_1)^\alpha}{(1-p_1)^\alpha}.
\]

Similarly, the probability of observing \(s_f'\) for \(\alpha\) consecutive times when \(s_f = H_T\) is

\[
\Pr(\{e_{n_1-\alpha+1}, \ldots, e_{n_1}\} \forall n_2 \geq \alpha \mid s_f = H_T) = (1-\Pr(e = s_f \mid s_f = H_T))^\alpha,
\]

where \(n_2 \in \mathbb{Z}_+\) is the number of observations necessary for \(\alpha\) consecutive observations of \(s_f'\) to occur when \(s_f = H_T\). Similar to \(n_1\), the number of observations \(n_2\) necessary to go back to the first phase when \(s_f = H_T\) is a random variable and will be further analyzed later. Unlike the previous scenario in which \(s_f \neq H_T\), the expected number of observations necessary to observe \(s_f'\) for \(\alpha\) consecutive times should be very large when \(s_f = H_T\). With \(p_2 = \Pr(e = s_f \mid s_f = H_T)\), the expected value of \(n_2\), for any environment observation \(e\), will be:

\[
\mathbb{E}[n_2 \mid s_f = H_T] = p_2 \left( \sum_{j=0}^{\alpha-1} (1-p_2)^j \left( \mathbb{E}[n_2 \mid s_f = H_T] + j + 1 \right) \right) + \alpha (1-p_2)^\alpha,
\]

\[
= \mathbb{E}[n_2 \mid s_f = H_T] p_2 \left( \sum_{j=0}^{\alpha-1} (1-p_2)^j \right) + p_2 \left( \sum_{j=0}^{\alpha-1} (1-p_2)^j (j+1) \right) + \alpha (1-p_2)^\alpha,
\]

\[
= \mathbb{E}[n_2 \mid s_f = H_T] p_2 \frac{1 - (1-p_2)^\alpha}{1 - (1-p_2)} + p_2 \left( \sum_{j=0}^{\alpha-1} (1-p_2)^j (j+1) \right) + \alpha (1-p_2)^\alpha,
\]

\[
= \mathbb{E}[n_2 \mid s_f = H_T] \frac{p_2}{1 - (1-p_2)^\alpha} \left( \sum_{j=0}^{\alpha-1} (1-p_2)^j (j+1) \right) + \alpha (1-p_2)^\alpha.
\]

The threshold \(\alpha\) can be seen as a design parameter to adjust the expected number of necessary iterations in (18) and (23). Moreover, there are advantages and disadvantages of choosing a high value and a low value for \(\alpha\). Given (8), a high value of \(\alpha\) will imply that it is highly unlikely to incorrectly go back to the first phase. However, when choosing \(s_f \neq H_T\), it will take many observations to go back to the first phase, and, thus, the learning method converges slowly to \(s_f = H_T\). A low value of \(\alpha\) will imply that the learning method can quickly go back to the first phase to choose \(s_f\) again if \(s_f \neq H_T\). Meanwhile, a small \(\alpha\) also implies that it is more likely to incorrectly go back to the first phase even when \(s_f = H_T\).
Similar to the first phase, the IoT devices in the second phase will individually learn after receiving $m$ bits of information from any neighboring IoT device that learned and, then, they will transmit $m$ bits of updated information to neighboring IoT devices. The memory that IoT device $i$ needs to learn $N_a$ is essentially the size of the received information set $\{e_{i-m+2}, \ldots, e_{i-1}, F_{i-1}, Q_{i-1}\}$, where $e_{i-1}$ is the environment observation of IoT device $i-1$, $F_{i-1}$ is the currently favored hypothesis, and $Q_{i-1}$ indicates whether $F_{i-1}$ should be changed. The value of $m$ must be at least two bits to capture $\{F_{i-1}, Q_{i-1}\}$, which are necessary for learning \cite{22}, and the second phase reduces to the finite memory sequential learning introduced in \cite{22} if $m = 2$.

For $m > 2$, an IoT device $i$ computes its private belief $x_i$ based on maximum likelihood using the observations of previous devices in learning sequence $\{e_{i-m+2}, \ldots, e_{i-1}\}$ and its own observation $e_i$. Using private belief $x_i$ instead of its own observation $e_i$ effectively extends the limited observation range $r_d$ such that the devices outside of $r_d$ will have a probability of missed detection of $p_{01}$ instead of $p_{01}'$. Therefore, the assumptions in \cite{6} hold true for more IoT devices, and more IoT devices will learn correctly. Moreover, the use of $x_i$ instead of $e_i$ is very effective in the second phase, because the learning progresses sequentially away from the abnormal event and the IoT devices participating in the second phase will most likely be outside of $r_d$. Therefore, without using $x_i$, such IoT devices will be subject to high probability of missed detection $p_{01}'$. In this case, the assumptions in \cite{6} may not hold true for the IoT devices participating in the second phase. The effective detection radius $r_d'$ can therefore be defined as follows:

$$r_d' = r_d + (m - 2)r_c,$$  \hfill (24)

where $m$ is the memory size of a device (in bits). For a larger memory size $m$, the effective radius $r_d'$ in \cite{22} will be bigger, and, thus, during the second phase, more IoT devices are subject to $p_{01}$ instead of $p_{01}'$. However, as the deployment region is finite, it is unnecessary to have very large values of $m$, which can render $r_d'$ much greater than the dimensions of the deployment region.

Moreover, for larger values of $m$, more energy is required to transmit, which may be impractical for IoT devices with energy constraints. Therefore, it is necessary to choose an appropriate value of $m$ depending on network dimensions, energy constraints, and delay requirements.

An IoT device $i$ in the second phase learns using $x_i$, $F_{i-1}$, and $Q_{i-1}$ by repeatedly testing whether the currently favored hypothesis $F_{i-1} \in \{0, 1\}$ should be changed. Moreover, a number
of consecutive devices in the learning sequence need to collectively test and decide whether $F_{i-1}$ should be changed. Here, we define the notion of an S-block as a set consecutive devices testing whether $F_{i-1}$ should be changed from 0 to 1 and an R-block as a set of consecutive devices testing whether $F_{i-1}$ should be changed from 1 to 0. Every device in a learning sequence belongs to either an S-block or an R-block, and both blocks are alternating in a learning sequence to repeatedly test $F_{i-1}$. As shown in [22], the lengths of the S and R-blocks must be chosen to ensure that $F_{i-1}$ will converge to true hypothesis.

Since observing $s_f$ is equivalent to 1 and observing $s'_f$ is equivalent to 0, the IoT devices belonging to an S-block decide that $F_{i-1}$ should be changed from $s'_f$ to $s_f$ if all devices in that S-block observe $s_f$. Similarly, the IoT devices belonging to an R-block decide that $F_{i-1}$ should be changed from $s_f$ to $s'_f$ if all devices in that R-block observe $s'_f$. $Q_{i-1}$ is used to track if all of the observations are $s_f$ in S-block or $s'_f$ in R-block, and $F_{i-1}$ is changed accordingly by the IoT devices. After learning, IoT device $i$ updates $m$ bits of information by replacing $\{F_{i-1}, Q_{i-1}\}$ with $\{F_i, Q_i\}$ and the oldest observation $e_{i-m+2}$ with its own observation $e_i$. Moreover, IoT device $i$ propagates the updated information to its neighboring devices for them to learn.

C. Convergence of the Proposed Learning Method

Our proposed learning method consists of two phases during which a favored state $s_f$ is chosen and then is repeatedly tested whether $s_f$ should be changed. For the first learning phase, it must be shown that the favored state $s_f = H_T$ can be chosen correctly during the first phase with nonzero probability. Moreover, for the second phase, it must be shown that the learning method will not revert back to first phase with probability approaching 1 if $s_f = H_T$. As $s_f \neq H_T$ can be chosen during the first phase, it must also be shown that the learning method will revert back to first phase with probability 1 when $s_f \neq H_T$. If the aforementioned properties of both first and second phases of the learning method can be shown, then the learning method will be guaranteed to converge to the correct underlying state $H_T$, as shown next.

**Theorem 1.** For a memory size $m$ such that $(m - 2) \geq \alpha$, if (8) holds, then the proposed finite memory learning method will converge in probability to the true, underlying state $H_T$.

**Proof.** See Appendix A.

The result of Theorem 1 applies whenever (8) holds true for all agents participating in the
learning method and none of the agents suffers from the limited observation capability. However, the IoT devices in our model have limited observation capabilities, and, thus, (8) may not hold true for all of them. However, with sufficiently large memory size $m$ such that $r'_d$ in (24) is large enough to circumscribe the entire deployment region, (8) will hold true for all IoT devices. Therefore, with an additional condition on the value of $m$, the proposed finite memory learning method will converge in probability to $H_T$ for the IoT devices. Furthermore, the IoT devices are limited in memory, and, thus, the effect of having small or big memory sizes on the performance and convergence of learning will be further analyzed via simulations in Section IV.

D. Delay of Critical Messages

For larger sizes $m$ of the devices’ memory, (8) holds true for more IoT devices that can learn $\beta$ correctly during the second phase, and, thus, the delay of the IoT devices with critical messages will be smaller. As the learning propagates during $t$ time slots, an IoT device, which learned and is located furthest from a device with critical messages, is located at a distance of $tr_c$ from the abnormal event. In other words, after $t$ time slots, only the IoT devices that are located closer than $tr_c$ are part of a learning sequence. Therefore, whether an IoT device was able to learn correctly depends on both $m$ and $t$ as the IoT devices that are located within $r_l = \min(tr_c, r'_d)$ are likely to learn correctly.

Assuming a massive IoT network with limited number of RAPs with $N \gg p$, $T_{min}$ is very large, and, thus, $D$ and $D'$ are dictated by $D_c$ and $D'_c$ respectively. Since $D'_c$ approaches 1 as $\beta$ increases, $D_{th}$ must be greater than 1 for a feasible value of $\beta$. For a given threshold delay $D_{th} < D_c$, the value of $\beta$ is, given system parameters $p_c$ and $N_a$:

$$D'_c = \left(\frac{p_c + \beta}{p_c + \beta - 1}\right)^{N_a} \leq D_{th},$$  \tag{25}

$$p_c + \beta \leq \sqrt[N_a]{D_{th}(p_c + \beta - 1)} \quad \text{and} \quad \beta(1 - \sqrt[N_a]{D_{th}}) \leq \sqrt[N_a]{D_{th}(p_c - 1) - p_c},$$  \tag{26}

$$\frac{\sqrt[N_a]{D_{th}(p_c - 1) - p_c}}{1 - \sqrt[N_a]{D_{th}}} \leq \beta = \left\lfloor \frac{\sqrt[N_a]{D_{th}(p_c - 1) - p_c}}{1 - \sqrt[N_a]{D_{th}}} \right\rfloor.$$  \tag{27}

For the value of $\beta$ in (27), the delay threshold $D_{th}$ will be satisfied for the critical messages for $m$ and $t$ approaching $\infty$. However, it is impractical to have very large $m$ considering the restricted resources of IoT devices. Furthermore, it is of interest to analyze the transient phase of our learning method, because the IoT devices with critical messages try to transmit as other
devices are still learning. Therefore, the maximum number of reallocated RAPs $\beta_t$ at time $t$, for finite values of $m$ and $t$, is of interest to determine the expected delay of critical messages for realistic values of $m$ and $t$.

**Theorem 2.** For finite values of $t$ and $m$, the maximum expected number of reallocated RAPs $\beta_t$ at time $t$ used to reduce the delay of critical messages, that can be achieved using our learning approach is:

$$E[\beta_t] = \frac{(pf-\beta)!}{pf!} \sum_{b=0}^{\beta} bP(n_l,b)P(n'_l,\beta-b)C(\beta,b).$$

(28)

**Proof.** See Appendix B.

Theorem 2 derives the maximum expected number of contention-free RAPs that will be reallocated to contention-based RAPs to reduce the delay of critical messages in a realistic IoT with finite $t$ and $m$. Moreover, it is interesting to note that when $m = \infty$ and $t = \infty$, $n_l = pf$ as $r'_d$ will be large enough to include the entire deployment region. Therefore, the probability of having $\beta_t$ reallocated RAPs at time $t$ in (34) is 1 for $\beta_t = \beta$ and zero otherwise, and, thus, $E[\beta_t] = \beta$. Moreover, for higher values of $m$, $t$, and, thus, $n_l$, $E[\beta_t]$ will approach $\beta$ and the critical messages are more likely to satisfy $D_{th}$. With finite values of $m$ and $t$, the lowest expected delay that can be achieved with our proposed learning method is

$$\left(\frac{pe+E[\beta_t]}{pe+E[\beta_t]-1}\right)^{N_a}.$$  

**IV. SIMULATION RESULTS AND ANALYSIS**

For our simulations, we consider a rectangular area of width $w$ and length $l$ within which the IoT devices are deployed following a Poisson point process of density $\lambda$ and an expected number of IoT devices $wl\lambda$. We let $w = l = 100$ m, $r_c = 2$ m, and $r_d = 10$ m, and we choose a time slot duration of 0.25 ms [27]. A random location within the deployment region is chosen to be the location of an abnormal event, and the IoT devices within a distance of 1 m will have critical messages and initiate the learning procedure. The number of RAPs $p$ is 64 [23], which includes $pf = 63$ contention-free RAPs and $pc = 1$ contention-based RAPs. We also set $K = 3$ and $\alpha = 5$. Moreover, the probability of missed detection $p'_{01}$ outside of $r'_d$ is set to 0.9 [26]. All statistical results are averaged over a large number of independent runs.

Fig. 2 shows snapshots of learning with different memory sizes progressing in a small-scale network at $t = 0.25$ ms, 1.75 ms, and 4.25 ms. For both scenarios, learning converges by $t = 4.25$
Figure 2. Snapshots showing how the proposed learning scheme progresses in small scale networks at $t = 0.25$ ms, $1.75$ ms, and $4.25$ ms, respectively.

At $t = 0.25$ ms, for both values of $m$, only the neighboring devices of the IoT devices with critical messages had a chance to learn in the first phase, and they all learned correctly. At $t = 1.75$ ms, for learning with $m = 3$ bits shown in Fig. 2(a), IoT devices outside of $r'_d$ started to learn incorrectly due to the limited observability. However, for learning with $m = 10$ bits shown in Fig. 2(b), IoT devices are still within $r'_d$ and are learning correctly. At $t = 4.25$ ms, even with larger memory size of $m = 10$ bits, some IoT devices are outside of $r'_d$ and are unable to learn correctly. For both memory sizes, there are few IoT devices that did not have a chance to learn as they are not within $r_c$ of any other IoT devices.

Fig. 3 shows the cumulative distribution function for the delay of critical messages for different delay thresholds $D_{th}$ with $\lambda = 2$, $p_{11} = 0.9$, and $m = 5$ bits. Fig. 3 shows that the delay threshold is satisfied, which means that the achieved delay is less than or equal to $D_{th}$, with probability 0.80 for $D_{th} = 0.75$ ms, 0.87 for $D_{th} = 1.25$ ms, 0.95 for $D_{th} = 2.5$ ms, and 0.99 for $D_{th} = 5$ ms. Moreover, the average achieved delay is 0.59 ms for $D_{th} = 0.75$ ms, 0.72 ms for $D_{th} = 1.25$ ms, 0.99 ms for $D_{th} = 2.5$ ms, and 1.35 ms for $D_{th} = 5$ ms. Although the largest $D_{th} = 2.5$ ms has the longest average delay, it has the highest probability of satisfying the delay threshold.
Figure 3. Cumulative distribution function of delay of critical messages for different $D_{th}$.

This is because the IoT devices with critical messages have more time available to satisfy $D_{th}$ for larger $D_{th}$, and more IoT devices can learn $\beta$ with more time available. Moreover, with more time, the IoT devices with critical messages are more likely to be allocated contention-free RAP by the BS for successful RA request. Here, it is important to note that $D_{th}$ may not be satisfied, because $t$ and $m$ are finite.

Fig. 4 shows the cumulative distribution function for the delay of critical messages for different network densities $\lambda$ with $D_{th} = 2.5$ ms, $p_{11} = 0.9$, and $m = 5$ bits. As our proposed learning
method relies on the IoT devices communicating and propagating information, it is important that the IoT devices have at least one neighboring device to be able to learn $\beta$. Therefore, a massive IoT network with higher values of $\lambda$ is better for learning. However, with smaller value of $\lambda$, $T_{\text{min}}$ is smaller, and, thus, $D_f$ is smaller. The value of $D_f$ is 19.75 ms for $\lambda = 1$, 39.63 ms for $\lambda = 2$, and 79.25 ms for $\lambda = 4$. Therefore, a small value of $\lambda$ increases the delay by increasing the number of isolated IoT devices and decreases the delay by decreasing the value of $D_f$. From Fig. 4, we can see that the probability of achieving a delay less than or equal to $D_{\text{th}}$ is 0.97 for $\lambda = 1$, 0.95 for $\lambda = 2$ and 0.92 for $\lambda = 4$. Therefore, it is more likely to satisfy $D_{\text{th}}$ with lower $\lambda$, and, thus, the effect of having smaller $D_f$ is more significant than having more isolated IoT devices. Furthermore, this shows the effectiveness of our learning method in a massive network with devices that have limited resources and capabilities.

Fig. 5 shows the cumulative distribution function for the delay of critical messages for different $p_{11}$. Since the observation of an IoT device affects its learning as well as that of all IoT devices later in the sequence, the performance of our proposed method significantly depends on the quality of observations. For low $p_{11}$, the IoT devices are prone to false alarms as $p_{01} = 1 - p_{11}$, and, thus, they do not learn $N_a$ correctly, which causes higher critical message delays. The delay threshold is satisfied with probability 0.98 for $p_{11} = 0.99$, 0.96 for $p_{11} = 0.9$, 0.85 for $p_{11} = 0.7$, and 0.57 for
Fig. 6 shows the cumulative distribution function for the delay of critical messages for different memory sizes $m$ with $D_{th} = 2.5$ ms, $\lambda = 2$, and $p_{11} = 0.9$. The effect of having larger values of $m$ is not significant in reducing the delay of critical messages even though the memory size $m$ is critical for the convergence of proposed learning method. This is because the RA request is successfully done during first few time slots as shown in Fig. 6 when the learning method is still in an early transient phase. The effect of having larger values of $m$ is more clear when the learning has progressed to the IoT devices at the edge of detection range $r_d$ as the memory size $m$ extends $r_d$ to $r'_d$. Therefore, the effect of $m$ on the performance of the learning method will be more pronounced after more devices learned as is later shown in Fig. 7.

The convergence of the proposed learning technique is analyzed via simulations by studying the percentage of IoT devices that learn correctly after all devices had a chance to learn $N_a$. The parameters that affect the performance of the proposed learning technique include memory size $m$, network density $\lambda$, and probability of detection $p_{11}$ within $r'_d$. The network density $\lambda$ affects the number of IoT devices that can learn such that an IoT device is more likely to have no neighboring device within $r_c$ to get necessary information for learning for lower values of $\lambda$. 

$p_{11} = 0.5$. Unlike $D_{th}$ and $\lambda$, which indirectly affect the performance of learning, $p_{11}$ affects the performance of learning method directly and more significantly, because the probability of a successful RA request achieves 0.99 after 11 time slots.
Such isolated device will not be able to learn using our proposed learning method. However, if the network is dense enough, the performance of learning is limited not by $\lambda$ but by memory size $m$ and probability of detection $p_{11}$. The memory size $m$ determines the effective detection radius $r'_d$ and $p_{11}$ controls how accurately the devices can observe within $r'_d$. For the subsequent simulations, we take $D_{th} = 3$, while varying $m$ and $p_{11}$.

Fig. 7 shows the average percentage of devices that learned $N_a$ correctly out of all $N$ IoT devices for different memory sizes $m$ with $\lambda = 2$. As the IoT devices can learn $N_a$ correctly within $r'_d$ and $r'_d$ depends on $m$, the average percentage of devices that learned correctly is clearly limited for finite memories $m = \{3, 10, 15, 30\}$ bits, while the average percentage approaches 100% for infinite memory $m = \infty$ bits. Moreover, the average percentage of devices that learned correctly is an increasing function of $m$ as $r'_d$ is an increasing function of $m$. The maximum average percentage of devices that learned correctly is 4.67% for $m = 3$ bits, 18.79% for $m = 10$ bits, 29.84% for $m = 15$ bits, and 66.07% for $m = 30$ bits. This is because as the learning progresses further, the IoT devices outside of $r'_d$ will learn incorrectly, and, thus, the percentage of devices that learn correctly out of all $N$ will no longer increase. Moreover, the maximum average percentages for different values of $m$ are achieved at different time slots since a larger $m$ requires more time to achieve the maximum average percentage.

Fig. 8 shows the average percentage of devices that learned $N_a$ correctly out of all $N$ IoT
devices for different probabilities of detection \( p_{11} \) with \( m = 15 \) bits and \( \lambda = 2 \). For lower \( p_{11} \), the devices are more likely to observe and learn \( N_a \) incorrectly. Learning incorrectly will affect subsequent devices in their learning as the proposed approach relies on the observation of previous IoT devices. For \( p_{11} = 0.5 \), less than 1% of IoT devices learn correctly, and about 17.5% of IoT devices learn correctly even for \( p_{11} = 0.9 \). Even for a \( p_{11} \) close to 1, Fig. 8 shows that the average percentage of devices that learn \( N_a \) correctly does not approach 100%. Hence, we can conclude that \( m \) is much more critical in increasing the percentage of devices that learned correctly than \( p_{11} \).

The percentage of devices that learned correctly does not approach 100% in Fig. 8, and the proposed learning method does not seem to converge to \( H_T \). This is because the value of \( m \) is not large enough to let \( r'_d \) circumscribe the deployment region. For small \( m \), some IoT devices do not satisfy (8) due to their limited observation range. In other words, for those devices, the effective detection range \( r'_d \) is not extended enough to include the IoT devices located far away from the abnormal event, and the IoT devices located outside of \( r'_d \) will not learn correctly. However, it is unnecessary to have all IoT devices learn correctly to satisfy a given delay threshold as shown in Fig. 3, Fig. 4, and Fig. 5. Furthermore, since the memory size \( m \) will be finite or even small in practical IoT systems, it is important to analyze the convergence of learning method with finite values of \( m \) as shown in Fig. 7.

Figure 8. Average percentage of devices that learned correctly out of \( N \) devices with varying \( p_{11} \).
V. CONCLUSION

In this paper, we have proposed a novel learning framework for enabling IoT devices to reallocate limited communication resources depending on the types and number of messages existing in the IoT. We have introduced a finite memory multi-state sequential learning using which the heterogeneous IoT devices can reallocate RAPs appropriately to reduce the delay of critical messages. We have shown that our proposed learning framework is suitable for the IoT devices with limited computational capability and finite memory and effective against the limited observation capability of IoT devices. Furthermore, we have proved the convergence of our proposed learning framework and derived the lowest expected delay of critical messages that can be achieved with our proposed learning framework. Simulation results have shown that the cumulative distribution function of delay of critical messages and average percentage of devices that learned correctly are functions of memory size, network density, and probability of detection. In particular, our proposed learning framework has shown to be effective in massive network with low delay threshold.

APPENDIX

A. Proof of Theorem 1

The first phase of the proposed learning method is based on maximum likelihood estimation, and, thus, the chosen $s_f$ depends on the $K$ observations. Given the assumption in (8), if all of the $K$ observations are $H_T$, then $s \in S$ that will maximize (11) will be $H_T$ and, thus, $s_f$ will be $H_T$. Moreover, given (7), $\Pr(e = H_T|H_T)$ cannot be zero. Since the observations are independent, the probability of observing $K$ $H_T$ is $\Pr(e = H_T|H_T)^K$, and, thus, the probability of having $K$ observations of $H_T$ is not zero. Therefore, the favored state $s_f$ will be chosen correctly such that $s_f = H_T$ with nonzero probability from the first learning phase.

If $s_f = H_T$ is chosen from first phase, then the true underlying state is $s_f$ and, thus, second phase will converge to $H_T$. However, if $s_f \neq H_T$ is chosen from the first phase, then the true underlying state is $s_f'$ and, thus, the second phase will not converge to $H_T$. Therefore, there is a returning condition, which is $\alpha$ consecutive observations of $s_f'$, using which the learning method can go back to the first phase if $s_f \neq H_T$ seems to be chosen from first phase.

When $s_f \neq H_T$, the probability of observing $s_f'$ for $\alpha$ consecutive times, which may cause
the learning method to go back to the first phase, is:

\[
Pr(n_e|s_f \neq H_T) = Pr(\{e_{n'-\alpha+1}, \ldots, e_{n'}\} = \{s'_f, \ldots, s'_f\} \forall n' \leq n_e \mid s_f \neq H_T)
\]

\[
= \begin{cases} 
0 & \text{if } n_e < \alpha, \\
q_1^\alpha & \text{if } n_e = \alpha, \\
q_1^\alpha + (N - \alpha)p_1q_1^\alpha - p_1q_1^\alpha \left( \sum_{i=0}^{n_e-(\alpha+1)} Pr(i|s_f \neq H_T) \right) & \text{if } n_e > \alpha,
\end{cases}
\]

(30)

where \( q_1 \) is \((1 - p_1) \) and \( n_e \) is number of observations. Similarly, for the case when \( s_f = H_T \), the probability of observing \( s'_f \) for \( \alpha \) consecutive times after \( n_e \) observations is:

\[
Pr(n_e|s_f = H_T) = Pr(\{e_{n'-\alpha+1}, \ldots, e_{n'}\} = \{s'_f, \ldots, s'_f\} \forall n' \leq n_e \mid s_f = H_T)
\]

\[
= \begin{cases} 
0 & \text{if } n_e < \alpha, \\
q_2^\alpha & \text{if } n_e = \alpha, \\
q_2^\alpha + (N - \alpha)p_2q_2^\alpha - p_2q_2^\alpha \left( \sum_{i=0}^{n_e-(\alpha+1)} Pr(i|s_f = H_T) \right) & \text{if } n_e > \alpha,
\end{cases}
\]

(32)

where \( q_2 \) is \((1 - p_2) \). Here, we note that \( Pr(n_e|s_f \neq H_T) \) and \( Pr(n_e|s_f = H_T) \) are cumulative probability distributions. Since \( p_2 > p_1 \), \( Pr(n_e|s_f \neq H_T) \) increases to 1 faster than \( Pr(n_e|s_f = H_T) \). However, as \( n_e \) increases to infinity, \( Pr(n_e|s_f \neq H_T) \) and \( Pr(n_e|s_f = H_T) \) will both approach 1. Thus, as \( n_e \) increases to infinity, the learning method will go back to the first phase regardless of \( s_f \). Therefore, the probability density function \( p_b \), which determines the probability of going back to Phase 1 once \( s'_f \) is observed \( \alpha \) consecutive times, must be designed appropriately to ensure that learning will only go back to Phase 1 when \( s_f \neq H_T \).

An appropriate choice for \( p_b \) is such that \( p_b \) is high for lower values of \( n_e \), while \( p_b \) approaches 0 as \( n_e \) increases. Since \( Pr(n_e|s_f \neq H_T) \) increases to 1 faster than \( Pr(n_e|s_f = H_T) \forall n_e \) as \( p_2 > p_1 \), \( Pr(n_e|s_f \neq H_T) \) is much greater than \( Pr(n_e|s_f = H_T) \) for small values of \( n_e \), and, thus, it is much more likely to observe \( s'_f \) for \( \alpha \) consecutive times when \( s_f \neq H_T \) than when \( s_f = H_T \) for small values of \( n_e \). Therefore, for small \( n_e \), \( p_b \) must be high for small values of \( n_e \) so that the learning method is likely to go back to the first phase when \( s_f \neq H_T \) as \( Pr(n_e|s_f \neq H_T) \) is big, while the learning method is unlikely to go back to the first phase as \( Pr(n_e|s_f = H_T) \) is small. However, as \( Pr(n_e|s_f = H_T) \) approaches 1 as \( n_e \) goes to infinity, \( p_b \) must approach 0 as \( n_e \) goes
to infinity to prevent going back to first phase when \( s_f = H_T \). One such choice for \( p_b \) will be a sigmoid function that quickly decreases to 0 for \( n_e > a \) for some value \( a \). For instance, one such sigmoid function is:

\[
C \left( 1 - \frac{n_e - a}{1 + |n_e - a|} \right),
\]

(33)

where \( C \) is a scaling factor to make (33) an appropriate probability distribution. Similar to \( \alpha \), the value of \( a \) is a design parameter that determines the value of \( n_e \) at which it becomes unlikely for the learning method to go back to the first phase. Furthermore, the value of \( a \) should be chosen between the values of \( n_e \) where \( \Pr(n_e \mid s_f \neq H_T) \) is close to 1, while \( \Pr(n_e \mid s_f = H_T) \) is close to 0, and where \( \Pr(n_e \mid s_f = H_T) \) starts to significantly increase to 1.

With appropriate choice for \( p_b \), when \( s_f \neq H_T \), the learning method is highly likely to go back to the first phase even when \( n_e \) is low, because both \( p_b \) and \( \Pr(n_e \mid s_f \neq H_T) \) are high. However, when \( s_f = H_T \), the learning method is highly unlikely to go back to the first phase, because either \( p_b \) or \( \Pr(n_e \mid s_f = H_T) \) is high, while the other is low. Therefore, as learning progresses and the number of observations \( n_e \) increases to infinity, the learning method goes back to the first phase to change \( s_f \) with probability approaching 0 when \( s_f = H_T \). However, \( n_e \) increasing to infinity does not require the learning method to have infinite memory as our finite memory learning method updates the memory by replacing the oldest observation by newest observation. With \( m \) such that \((m - 2) \geq \alpha \), the finite memory will have enough observations to check if \( \alpha \) consecutive observations of \( s'_f \) have occurred. Since the first phase will choose \( s_f = H_T \) with nonzero probability, the learning method will converge to \( H_T \) in probability.

\[ B. \text{ Proof of Theorem 2} \]

For finite values of \( t \) and \( m \), the maximum ratio of IoT devices that learn correctly is \( \frac{\pi r_l^2}{A} \), where \( A \) is the area of geographical region on which the IoT devices are deployed and \( r_l = \min(tr_c, r'_d) \) is a range within which the devices are likely to learn correctly. It assumes that \( r_l \) is entirely within the network and all IoT devices learn correctly within \( r_l \), which is best scenario for our learning method. In a given time slot at time \( t \), the expected number of IoT devices with periodic messages transmitting in that time slot that learned correctly is \( n_t = \frac{mt \pi r_l^2}{A} \) and, the expected number of IoT devices with periodic messages transmitting in that time slot that did not learn correctly is \( n'_t = p_f - n_t \).
Since there are IoT devices outside of $r_l$ that do not learn correctly with finite $m$ and $t$, there will be varying number of reallocated RAPs $\beta_t$ in different time slots, and the value of $\beta_t$ may range from 0 to $\beta$. For a value of $\beta_t \in [0, \beta]$, the probability of having $\beta_t$ reallocated RAPs at any given time $t$ is:

$$\frac{(p_f - \beta)\!}{p_f!} P(n_t, \beta_t) P(n'_t, \beta - \beta_t) C(\beta, \beta_t),$$  

(34)

where $P(n, k)$ is $k$-permutations of $n$ and $C(n, k)$ is $k$-combinations of $n$. The probability in (34) considers different cases at which $\beta_t$ IoT devices that learned correctly are assigned to use the first $\beta$ contention-free RAPs, which are to be allocated first, for their periodic messages. Furthermore, the expected number $E[\beta_t]$ of contention-free RAPs that will be reallocated to contention-based RAPs is:

$$E[\beta_t] = \frac{(p_f - \beta)\!}{p_f!} \sum_{b=0}^{\beta} b P(n_t, b) P(n'_t, \beta - b) C(\beta, b).$$  

(35)

REFERENCES

[1] T. Park and W. Saad, “Learning with finite memory for machine type communication,” in Proc. of the IEEE 50th Annual Conference on Information Science and Systems, Princeton, USA, Mar. 2016.

[2] Z. Dawy, W. Saad, A. Ghosh, J. G. Andrews, and E. Yaacoub, “Toward massive machine type cellular communications,” IEEE Wireless Communications, vol. 24, no.1, pp. 120–128, Feb. 2017.

[3] S. Kelly, N. Suryadevara, and S. Mukhopadhyay, “Towards the implementation of IoT for environmental condition monitoring in homes,” IEEE Sensors Journal, vol. 13, no. 10, pp. 3846–3853, Oct. 2013.

[4] H. Farhangi, “The path of the smart grid,” IEEE Power and Energy Magazine, vol. 8, no.1, pp. 18–28, Jan. 2010.

[5] J. J. Nielsen, G. C. Madueno, N. K. Pratas, R. B. Sorensen, C. Stefanovic, and P. Popovski, “What can wireless cellular technologies do about the upcoming smart metering traffic?” IEEE Communications Magazine, vol. 53, no. 9, pp. 41–47, Sep. 2015.

[6] M. Mozaffari, W. Saad, M. Bennis, and M. Debbah, “Unmanned aerial vehicle with underlaid device-to-device communications: Performance and tradeoffs,” IEEE Transactions on Wireless Communications, vol. 15, no. 6, pp. 3949–3963, Jun. 2016.

[7] M. Weiner, M. Jorgovanovic, A. Sahai, and B. Nikolic, “Design of a low-latency, high-reliability wireless communication system for control applications,” in Proc. of IEEE International Conference on Communications, Sydney, Australia, June 2014.

[8] L. Xu, W. He, and S. Li, “Internet of things in industries: A survey,” IEEE Transactions on Industrial Informatics, vol. 10, no. 4, pp. 2233–2243, Nov. 2014.

[9] R. Khan, S. U. Khan, R. Zaheer, and S. Khan, “Future internet: The internet of things architecture, possible applications and key challenges,” in Proc. of 10th International Conference on Frontiers of Information Technology, Islamabad, Pakistan, Dec. 2012.
[10] H. Dhillon, H. Huang, H. Viswanathan, and R. Valenzuela, “Power-efficient system design for cellular-based machine-to-machine communications,” IEEE Trans. Wireless Commun., vol. 12, no.11, pp. 5740–5753, Nov. 2013.

[11] ——, “Fundamentals of throughput maximization with random arrivals for M2M communications,” IEEE Trans. Commun., vol. 62, no.11, pp. 4094–4109, Nov. 2014.

[12] L. Dai, B. Wang, Y. Yuan, S. Han, C. I. I, and Z. Wang, “Non-orthogonal multiple access for 5G: Solutions, challenges, opportunities, and future research trends,” IEEE Communications Magazine, vol. 53, no. 9, pp. 74–81, Sep. 2015.

[13] A. Amjad, F. Rabby, S. Sadia, M. Patwary, and E. Benkhelifa, “Cognitive edge computing based resource allocation framework for internet of things,” in Proc. of the Second International Conference on Fog and Mobile Edge Computing, Valencia, Spain, May 2017.

[14] M. Hasan, E. Hossain, and D. Niyato, “Random access for machine-to-machine communication in LTE-Advanced networks: Issues and approaches,” IEEE Communications Magazine, vol. 51, no. 6, pp. 86–93, June 2013.

[15] G. C. Madueno, C. Stefanovic, and P. Popovski, “Reliable reporting for massive M2M communications with periodic resource pooling,” IEEE Wireless Communications Letters, vol. 3, no. 4, pp. 429–432, May 2014.

[16] T. Park and W. Saad, “Resource allocation and coordination for critical messages using finite memory learning,” in Proc. of the 2016 IEEE Globecom Workshops, Dec. 2016.

[17] G. Durisi, T. Koch, and P. Popovski, “Toward massive, ultrareliable, and low-latency wireless communication with short packets,” Proc. of the IEEE, vol. 104, no. 9, pp. 1711–1726, Sep. 2016.

[18] Z. Zhou, M. Dong, K. Ota, G. Wang, and L. T. Yang, “Energy-efficient resource allocation for D2D communications underlaying cloud-RAN-based LTE-A networks,” IEEE Internet of Things Journal, vol. 3, no. 3, pp. 428–438, June 2016.

[19] V. Venkatasubramanian, M. Hesse, P. Marsch, and M. Maternia, “On the performance gain of flexible UL/DL TDD with centralized and decentralized resource allocation in dense 5G deployments,” in Proc. of IEEE 25th Annual International Symposium on Personal, Indoor, and Mobile Radio Communication, Washington DC, USA, Sep. 2014.

[20] M. Chen, Y. Miao, Y. Hao, and K. Hwang, “Narrow band internet of things,” IEEE Access, vol. PP, no. 99, pp. 1–1, Sep. 2017.

[21] T. Park, N. Abuzainab, and W. Saad, “Learning how to communicate in the internet of things: Finite resources and heterogeneity,” IEEE Access, vol. 4, pp. 7063–7073, Nov. 2016.

[22] T. Cover, “Hypothesis testing with finite statistics,” The Annals of Mathematical Statistics, vol. 40, no. 3, pp. 828–835, Jun. 1969.

[23] M. Ali, E. Hossain, and D. Kim, “LTE/LTE-A random access for massive machine-type communications in smart cities,” IEEE Communications Magazine, vol. 55, no.1, pp. 76–83, Jan. 2017.

[24] C. Yu, L. Yu, Y. Wu, Y. He, and Q. Lu, “Uplink scheduling and link adaptation for narrowband internet of things systems,” IEEE Access, vol. 5, pp. 1724–1734, Feb. 2017.

[25] K. Drakopoulos, A. Ozdaglar, and J. N. Tsitsiklis, “On learning with finite memory,” IEEE Transactions on Information Theory, vol. 59, no.10, pp. 6859–6872, May 2013.

[26] B. C. Arrue, A. Ollero, and J. R. M. de Dios, “An intelligent system for false alarm reduction in infrared forest-fire detection,” IEEE Intelligent Systems and their Applications, vol. 15, no. 3, pp. 64–73, June 2000.

[27] D. Catania, M. Sarret, A. Cattoni, F. Frederiksen, G. Berardinelli, and P. Mogensen, “The potential of flexible UL/DL slot assignment in 5G systems,” in Proc. of IEEE 80th Vehicular Technology Conference, Vancouver, Canada, Sep. 2014.