Dynamic Tree-Splitting Algorithm for Massive Random Access of M2M Communications in IoT Networks

Huda Althumali, *Graduate Student Member, IEEE*, Mohamed Othman, *Senior Member, IEEE*, Nor Kamariah Noordin, *Senior Member, IEEE*, and Zurina Mohd Hanapi, *Member, IEEE*

***Abstract*—Enabling machine-to-machine (M2M) communications on cellular networks will provide a promising future for smart cities and the Internet of Things. M2M systems involve a huge number of connected devices that may synchronously be activated to react to some event. This massive synchronous access causes intensive congestion and collisions in the random access channel (RACH), which is used as a first step to access network resources. In this article, we introduce a dynamic tree-splitting (DTS) algorithm to resolve RACH collisions for delay-sensitive devices during burst arrival scenarios. The DTS algorithm assigns a specific number of preambles to the collided devices for their next access attempt. The number of preambles is determined based on the mean number of collisions in each random access opportunity, with the aim of increasing the utilization of preambles. A mathematical analysis of the proposed algorithm is presented as well as the derivations of throughput and access delay. The analysis and simulation results show that DTS reduced access delay and increased RACH throughput by approximately 12%, compared to recent benchmarks, with a mean of three preamble transmissions and a success rate above 0.98, which indicates the efficiency and reliability of the proposed algorithm.**

***Index Terms*—Cellular Internet of Things (IoT) networks, collision resolution, dynamic tree-splitting (DTS), machine-to-machine (M2M), massive random access.**

I. Introduction

MACHINE-TO-MACHINE (M2M) communication involves thousands of devices that interact with each other without human intervention. These autonomous connections represent the cornerstone of the Internet of Things (IoT), in which every single object is connected to the Internet. M2M communication has several applications, such as smart grids, smart homes, industry monitoring, e-healthcare, and e-transportation. The Third-Generation Partnership Project (3GPP) has recommended cellular networks such as long-term evolution (LTE), LTE-advanced (LTE-A), narrow-band (NB) IoT, and fifth generation (5G) as adequate infrastructure for M2M communications due to their extended coverage and large capacity. In cellular networks, the random access channel (RACH) is used to acquire initial access to the base station, or what is called eNodeB. The RACH performs well with human-to-human uniform arrivals, but it may suffer from excessive congestion and collisions with the burst arrivals of M2M traffic, which cause most access requests to be dropped. The problem becomes more complicated when considering delay-sensitive services, which have strict deadline requirements. Several studies have investigated this issue and introduced numerous solutions to avoid RACH collisions. Most of these solutions rely on distributing massive arrival requests over time. However, this type of solutions is not appropriate for delay-sensitive applications, which require a minimum access delay. Therefore, this article will focus on resolving RACH collisions as soon as possible instead of wasting time avoiding them. A new collision resolution scheme is introduced, based on the tree-splitting concept in which each colliding group of devices is divided into several groups with a lower number of devices for their next access contention. Specifically, the contributions of this work are as follows.

1) The design of a dynamic tree-splitting (DTS) algorithm to resolve RACH collisions by dynamically adjusting the number of tree branches at each level based on collision intensity.
2) The optimization of the number of splitting groups based on the collision coefficient to maximize the access success rate and RACH throughput, which in turn reduces access delay.
3) A mathematical analysis of the DTS algorithm, in which we derive the number of branches on each tree level as well as the derivations of the throughput and the access delay.

The remainder of this article is organized as follows: Section II offers an overview of RACH design and procedure. The related

Manuscript received December 13, 2020; revised June 1, 2021; accepted July 12, 2021. This work was supported in part by the Saudi Ministry of Education and the Malaysian Ministry of Education through the Research Management Center, Universiti Putra Malaysia, under the Grant 9001103, and in part by Putra Grant with High Impact Factor under Grant UPM/7002/1/GPB/2017/9557900. *(Corresponding authors: Huda Althumali; Mohamed Othman.)*

Huda Althumali is with the Department of Computer Science, College of Science and Humanities, Imam Abdulrahman Bin Faisal University, Jubail 31961, Kingdom of Saudi Arabia (e-mail: hdalthemaly@iau.edu.sa).

Mohamed Othman and Zurina Mohd Hanapi are with the Department of Communication Technology and Network, Universiti Putra Malaysia, 43400 Seri Kembangan, Malaysia (e-mail: mothman@upm.edu.my; zurinamh@upm.edu.my).

Nor Kamariah Noordin is with the Department of Computer and Communication Systems, Faculty of Engineering, Universiti Putra Malaysia, 43400 Seri Kembangan, Malaysia (e-mail: nknordin@upm.edu.my).

Digital Object Identifier 10.1109/JSYST.2021.3097715

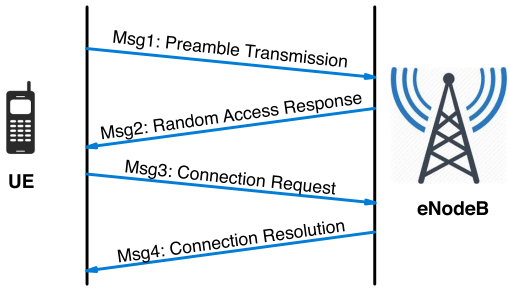


Fig. 1. Standard random access (SRA) procedure in cellular networks.

literature is presented in Section III, while Section IV discusses the system model. After that, the DTS design and analysis is presented in Section V. The performance evaluation is covered in Section VI, with the evaluation results being discussed in Section VII. Finally, Section VIII concludes this article.

II. RACH Overview

The standard random access (SRA) procedure in cellular networks is contention based, where devices are competing to conserve resources [1]. The SRA procedure consists of four message transmissions between the user equipment (UE) and the eNodeB, as shown in Fig. 1. The first message (Msg. 1) includes a preamble transmission that is randomly chosen by the UE in the nearest available RAO. There are 54 orthogonal preambles available for random access contention in each RAO. The transmission is considered successful if the preamble has been chosen by only one UE. If more than one UE transmits the same preamble, a collision occurs. However, the eNodeB at this stage is not able to detect the collision. The eNodeB then sends the second message (Msg. 2) as a random access response (RAR) to all detected preambles within t_{RAR} ms, informing the UEs about the uplink resources that must be used for the third message (Msg. 3) transmission. The UE then sends a connection request message to the eNodeB using the uplink resource that had been defined in the previous step. If more than one UE have chosen the same preamble, their Msg. 3 transmissions collide and the eNodeB detects this collision. The eNodeB replies to Msg. 3 by transmitting the fourth message (Msg. 4) to all successful requests, informing them that they have successfully gained access to the network. If the UE does not receive Msg. 4 within t_{CRT} ms it retries another access attempt after a random backoff time of $T_{BO} \in [0, BI]$, where BI is the maximum value of the backoff time and ranges from 1 to 960 ms. If the UE has exceeded the maximum number of transmission attempts $k_{\max}$, the UE request is dropped.

III. Related Work

Several solutions have been introduced to improve the performance of the RACH for massive arrivals of M2M communications. These solutions have been classified based on their objectives into two groups, namely congestion control techniques and collision resolution techniques [2]. Congestion control techniques aim to convert the burst arrivals of M2M traffic into uniform arrivals, while the collision resolution techniques focus on resolving the conflict between devices after a failed access attempt.

The 3GPP introduced the access class barring (ACB) scheme as a congestion control scheme in order to reduce massive arrivals of M2M traffic [3]. With ACB, the eNodeB broadcasts an ACB factor between 0 and 1 to all devices through a system information block type 2 (SIB2) message. When a device is activated, it generates a random number between 0 and 1. The device proceeds to the random access procedure if the generated number is less than or equal to the ACB factor. Otherwise, the device has to make another access attempt after an ACB time.

The ACB scheme has been improved in several directions in recent researches. For example, the priority of the devices is considered in the extended ACB: low-priority devices are barred during the massive access scenarios [4]. The ACB factor has been optimized using timing advance information to increase the number of successful requests per RAO [5]. A dynamic ACB scheme is proposed in [6], in which the ACB factor is optimized based on an estimation of the backlog devices in the next RAO. The performance of the RACH under the ACB control is analyzed in several works, such as [7], which compares the performance of the optimal ACB factor with the fixed one. The optimal combination of ACB factors and ACB time were determined for different scenarios in [8]. The ACB factor is also optimized using a reinforcement learning-based scheme to dynamically adapt to the burst arrivals of M2M communications [9]. Another approach to controlling massive arrivals of the M2M traffic is the prebackoff scheme [10]. In this scheme, the burst arrivals are uniformly distributed over a prebackoff window. The size of the window is determined by the prebackoff parameter. The performance of the prebackoff scheme has been analyzed for group paging scenarios with different prebackoff parameters [11], [12]. The results demonstrated that the increased value of the prebackoff parameter increases the access success probability at the expense of increasing the access delay.

Furthermore, the 3GPP has introduced the backoff scheme as a collision resolution scheme for the massive M2M random access [1]. In this scheme, the collided devices have to wait for a specific time period before they try another access attempt. The backoff time period depends on the setting of the backoff indicator (BI), which is chosen randomly by the base station in the range of 0–960 ms. Our previous work has investigated the backoff procedure and optimized the value of the BI [13]. We introduced a dynamic backoff scheme that optimizes the value of the BI based on the number of collided devices and the available resources. The results show that the dynamic backoff scheme achieves around a 99.9% access success probability compared to the standard backoff scheme in [1]. However, all the previous solutions are not appropriate for the delay-sensitive services that require minimum access delays. For this reason, researchers have introduced other solutions for delay-sensitive services. For example, the dynamic allocation scheme proposes the assignment of additional RACH resources to the M2M traffic in the massive arrivals scenarios [14]. Another approach is coordinated random access, in which critical information is transmitted by one or a few representatives of each group

of devices [15]. Moreover, a priority-based ACB scheme has been proposed in [16] to consider the different priorities of access classes. In this approach, M2M devices are divided into several classes based on their delay requirements and a different ACB factor is used for each class with the aim of reducing the access delay for high-priority devices. The issue of the increased access delay of the ACB scheme was investigated in [17]. The authors sought to reduce the access delay using deep reinforcement learning to dynamically adjust the barring time and the periodicity of the RAOs. The results show that this scheme reduced the mean access delay compared to the optimal ACB scheme in [6]. Furthermore, there are some solutions that focus on resolving the RACH collisions instead of delaying the arrivals to avoid them, such as the q-ary tree-splitting technique, in which the devices that collided in one preamble are directed to a group of q preambles for their next access attempt [18]. This scheme relies on the concept of the Capetanakis contention tree algorithms that was introduced for multiaccessing broadcast communication channels [19]. This scheme resolves the conflict between devices by splitting them into several groups, where each group contains only one active device. This concept was adopted in the q-ary algorithm for the random access procedure, in which collided devices are divided into q groups in the next contention attempt. Thus, the number of collisions is reduced in each group of contending devices. Specifically, the collided devices are directed to a number of preambles that are reserved only for these devices to perform the next access attempt. As the contending devices are uniformly randomly distributed over the q reserved preambles, the number of collisions is reduced in each access attempt. This process is repeated until all devices are successful or the maximum number of transmissions is reached. The q-ary technique shows an efficient performance in reducing the access delay and the number of preamble transmissions while keeping a high access success rate. The performance of the q-ary algorithm is affected by the value of the number of splitting groups q. As the q is reduced, the access delay is reduced, while the number of transmissions is increased. For this reason, the binary tree-splitting algorithm was adopted as a collision resolution scheme in hybrid random access protocols, which combine a congestion control method with a collision resolution technique to efficiently improve the RACH performance [20]. In this approach, the prebackoff scheme and the dynamic ACB scheme are used as outer techniques to reduce the massive arrivals of the M2M traffic. After that, the binary tree-splitting algorithm is used as a collision resolution scheme for the collided devices only. This protocol achieved a good performance in increasing the RACH throughput and reducing access delay. However, the prebackoff scheme adopted in the hybrid protocol requires prior knowledge of the total number of contending devices, which may not be available to the eNodeB. Furthermore, it has been shown that binary tree-splitting is optimal for Poisson arrivals traffic with a mean of 1.15 requests per preamble [20]. However, binary tree splitting is not efficient for burst arrivals where a massive number of collisions is expected. Splitting large numbers of collided devices into two groups in each transmission will definitely result in new collisions and most of the devices will reach the maximum number of transmissions and will be dropped. Moreover, the q-ary scheme divides each group of collided devices into q groups without considering the number of collided devices in each group. This scheme actually wastes the preambles if the number of collided devices is less than q because there will definitely be some idle preambles. For this reason, this article proposes a DTS algorithm, in which the number of preambles assigned to each group is determined based on the mean number of the collided device, which helps to improve preamble utilization, access success probability, and access delay. More specifically, the splitting of the collided devices into several contention groups helps to reduce collision probability in the next access attempt, and the optimization of the number of preambles assigned to each group helps to increase preamble utilization (RACH throughput), and therefore, increase the access success rate. Moreover, DTS helps to reduce the access delay, which is the most important parameter for delay-sensitive devices, by directing the collided devices to the nearest available group of preambles for their next access attempt.

IV. System Model

This article considers a massive number of machine devices N previously registered with a single eNodeB in an LTE network. Following the 3GPP specifications in [21], the burst arrivals model represents massive synchronous arrival of M2M access requests. According to the standards of the 3GPP, the burst traffic of M2M access requests is generated according to the beta distribution within a specific time period T_A as follows [21]:

$$g(t) = \frac{t^{(\alpha-1)}(T_A - t)^{(\beta-1)}}{T_A^{(\alpha+\beta-1)} B(\alpha, \beta)}, \qquad 0 \le t \le T_A \tag{1}$$

where $B(\alpha, \beta)$ denotes the beta function $B(\alpha, \beta) = \int_0^1 t^{(\alpha-1)}(1-t)^{(\beta-1)} dt$ and t is a single time slot ranging from 0 to T_A. According to [21], $\alpha = 3$ and $\beta = 4$.

The considered RACH is based on frequency-division duplexing, where time is divided into frames, and each frame is divided into ten subframes of a 1 ms duration. It is assumed that there is only one RAO available in each subframe. Therefore, the length of each RAO is equal to 1 ms. Each RAO contains M preambles used by the UEs for the initial random access. The details of the model notations are shown in Table I. Furthermore, we assume that devices can immediately proceed to perform the random access procedure once they are activated without performing any ACB process. Even though ACB has performed well in reducing the RACH congestion, it causes a great increase in access delay since all devices have to wait for an ACB time before they can proceed to the first access attempt. For this reason and since our work considers delay-sensitive systems, we discarded the use of the ACB to avoid the access delay increase. Instead, we allow all the arrivals to immediately proceed with the first access attempt, and then, only the collided devices join the proposed DTS for collision resolution.

It is worth mentioning that our proposed solution can also be implemented in LTE-A and 5G NR networks since they have the same physical structure for the RACH and the same RACH capacity [22].

TABLE I
SUMMARY OF MODEL NOTATIONS

Symbol	Description
N	Total number of UE requests (devices)
N_s	Total number of successful requests
M	Total number of preambles per initial RAO
$M_{i,k}$	Total number of preambles for the k^{th} level of the i^{th} subtree
k	Number of transmissions or number of tree levels
k_{max}	Maximum number of transmissions
T_A	Activation time period
$CC_{i,k}$	Collision coefficient for the k^{th} level of the i^{th} subtree
$b_{i,k}$	Number of branches (preambles) per group for the k^{th} level of the i^{th} subtree
$l_{i,k}$	Length of the k^{th} level of the i^{th} subtree
n_i	Number of arrival requests in the i^{th} subtree
$Ps_{i,k}$	Success probability for the k^{th} level of the i^{th} subtree
$Pd_{i,k}$	Idle probability for the k^{th} level of the i^{th} subtree
$Pc_{i,k}$	Collision probability for the k^{th} level of the i^{th} subtree
$ns_{i,k}$	Number of successful requests for the k^{th} level of the i^{th} subtree
$nc_{i,k}$	Number of collided requests for the k^{th} level of the i^{th} subtree
$mc_{i,k}$	Number of collided preambles for the k^{th} level of the i^{th} subtree
G_i	Number of preamble groups for the i^{th} subtree
$T(m_{i,k})$	Time of the next access attempt of preamble m for the k^{th} level of the i^{th} subtree
$g(m_{i,k})$	Group index for the next access attempt of preamble m for the k^{th} level of the i^{th} subtree
d_i	Delay of the i^{th} subtree
D	Total delay of DTS
Tp	Throughput of DTS

V. DYNAMIC TREE-SPLITTING ALGORITHM

In this section, the detailed process of the proposed DTS algorithm is explained. We start with a high-level description of DTS, after which mathematical analysis is presented.

A. DTS Description

The DTS algorithm is being proposed to efficiently resolve random access collisions during massive access scenarios by splitting the collided devices into a number of groups to reduce the number of contending devices in each contention frame. The preamble is considered collided if it has been sent by more than one device, even if it has been successfully decoded by the eNodeB on Msg. 1. Therefore, we assume that the devices that are collided in Msg. 3 have already been considered collided in Msg. 1. Thus, the devices that have collided in the same preamble are directed to a specific group of preambles for their next access attempt. The collided devices receive information about the next access attempt in a feedback message after each access attempt. For this purpose, this article adopts the new type of Msg. 4, denoted as Msg. 4b, as introduced in [18]. Usually, in the standard random access procedure, Msg. 4 is sent to the successful devices informing them about the uplink resources that will be used for their data transmission. However, the proposed Msg. 4b is an alternative to Msg. 4, and it is sent to the collided devices only informing them about the time and the group of preambles to be used in their next access attempt. Therefore, Msg. 4b is processed similarly to Msg. 4 and takes the same processing time, usually 5 ms according to the standard configuration of the RACH. Therefore, the delay of the collided devices is not affected, since they receive feedback message Msg. 4b after each access attempt as the successful devices receive Msg. 4. The signalling overhead and the limited capacity of downlink is resolved through the bundling transmission scheme, which will be discussed at the end of this section.

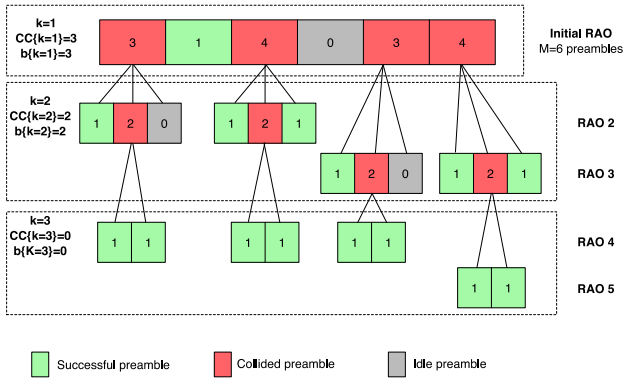


Fig. 2. Example of the DTS algorithm with $M = 6$ preambles and $N = 15$ devices.

The main aim of DTS is to optimize the number of preambles in each group according to the collision intensity of each RAO. This is different from the approaches in [18] and [20], where the number of preamble groups is determined in each RAO without considering the collision intensity, which definitely affects the utilization of preambles, and in turn, the overall performance of the RACH. In this article, the number of preamble groups G in each RAO is dynamically determined based on the number of preambles or branches b assigned to each group. Considering that there is a total of M preambles in a given RAO i, the number of groups in each RAO is $G_i = M/b_i$. The number of branches b_i for each RAO is changed dynamically based on the collision coefficient CC_i of the ith RAO where the collision was initiated. The value of CC_i aims to estimate the mean number of collided devices in each preamble. Basically, the value of CC_i for a given RAO i is computed as follows:

$$CC_i = \frac{NumberOfCollidedDevices}{NumberOfCollidedPreambles}. \quad (2)$$

This article seeks to make the number of branches b in a given RAO equal to the mean number of collided devices in each preamble. Therefore, the number of preambles b in each group will be approximately equal to the number of collided devices in that group, which helps to increase the access success probability and the throughput of preambles. It has been shown in [23] that maximum success probability and throughput is achieved when the number of contending devices in a given RAO is equal to the number of preambles available in that RAO. For this reason, this study aims to split the total number of preambles in a given RAO into G groups of contention frames, where each contention frame contains a number of preambles that is almost equal to the number of collided devices that will contend in this frame for the next access attempt. Fig. 2 shows an example of DTS for

a given RAO with $M = 6$ preambles and $N = 15$ contending devices. The first preamble transmission in the first level of the tree ($k = 1$) occurs randomly based on the standard random access. The numbers written in the slots indicate how many devices have chosen the preamble. It is shown that there are four collided preambles in the initial RAO. Each of the four collided preambles initiates three branches for the next access attempt. The number of branches is determined based on the collision coefficient of the initial RAO CC_1, which is equal to $\lfloor 14/4 \rfloor = 3$. Note that the number of contention frames/groups on the second level of the tree cover two RAOs since each RAO contains only six preambles, while the second level requires 4×3 preambles. All the collided preambles on the second level will continue with the same process for the third transmission. However, for the third transmission, the number of branches is reduced to be equal to two, which is equal to CC of the previous level ($CC_2 = \lfloor 8/4 \rfloor = 2$). Since each level of the tree represents one transmission attempt of the collided devices, a maximum of $k_{\max}$ levels are allowed for each tree. The process of DTS is repeated until either all collisions are resolved or $k = k_{\max}$, and therefore, the unsolved requests are dropped.

Furthermore, we note that the limited capacity of the physical downlink control channel (PDCCH) affects the performance of the RACH. Messages Msg. 2, Msg. 4, and Msg. 4b are sent through PDCCH, which can reply only to a maximum of 15 devices per RAO. However, to surpass this challenge, we adopted the bundling transmission scheme proposed in [24]. In this scheme, several control messages for several devices are multiplexed into one packet data unit and masked with reserved radio network temporal identifiers. This packet data unit is then sent through the physical downlink shared channel (PDSCH), which allows for the virtual increase of the PDCCH capacity. According to [24], the bundling transmission scheme is able to overcome the limited resources of the PDCCH by using only one more transport block on the PDSCH, which is considered to be sufficient in most communication scenarios. Therefore, we assume that the eNodeB can reply to all requests in each RAO by using the bundling transmission. All the successful requests that have succeeded in Msg. 1 and Msg. 3 will receive Msg. 4 through the PDSCH resources. Moreover, the collided devices will receive feedback message Msg. 4b through the same channel as well.

B. DTS Analysis

We consider a total number of N devices activated according to the beta distribution within a short time period $[0, T_A]$, which is assumed to be known by the base station. For a given RAO i, there are n_i arriving requests that immediately proceed to perform the first access attempt. The number of new arrivals n_i that are activated in a given RAO i is determined as follows [25]:

$$n_i = N \int_{t_{i-1}}^{t_i} \frac{t^{(\alpha-1)}(T_A - t)^{(\beta-1)}}{T_A^{(\alpha+\beta-1)} B(\alpha, \beta)} dt, \qquad 0 \le t \le T_A \tag{3}$$

where $t_i = 0, 1, 2, \ldots, T_A$. The first access attempt is done according to standard random access, where each of the n_i devices uniformly randomly selects one of the M preambles that are available in the ith RAO. The success probability of the first transmission ($k = 1$) for a given RAO i where $i \le T_A$ can be written as follows:

$$Ps_{i,1} = \frac{n_i}{M} \left(1 - \frac{1}{M}\right)^{(n_i - 1)}. \tag{4}$$

The idle probability is given by

$$Pd_{i,1} = \left(1 - \frac{1}{M}\right)^{n_i}. \tag{5}$$

The collision probability can be computed as follows:

$$Pc_{i,1} = 1 - Ps_{i,1} - Pd_{i,1}. \tag{6}$$

We can easily compute the number of successful requests $ns_{i,1}$ for the first access attempt, which is equal to the number of successful preambles in the ith RAO as follows:

$$ns_{i,1} = M \times Ps_{i,1}. \tag{7}$$

The number of collided preambles $mc_{i,1}$ can be computed as

$$mc_{i,1} = M \times Pc_{i,1}. \tag{8}$$

Since the number of arrivals n_i is known, the number of collided requests $nc_{i,1}$ can be computed as follows:

$$nc_{i,1} = n_i - ns_{i,1}. \tag{9}$$

After the first access attempt, each RAO that contains collided preambles must launch a subtree to resolve these collisions. Assuming that there is a single RAO in each time slot at the activation time T_A, the maximum number of subtrees is equal to T_A and the maximum number of levels of a subtree is $k_{\max}$. For the rest of this article, the ith subtree refers to the DTS launched by the ith RAO where the collision was initiated. The first level of a subtree always refers to the initial RAO where the first transmission was performed. All the collided devices must undergo DTS for 2, 3,..., $k_{\max}$ access attempts. The devices that are collided in a given preamble are directed to the next available group of preambles in an upcoming RAO for their next access attempt. The beginning of the DTS RAOs start after the activation time T_A to avoid collision with new burst arrivals. Therefore, all the RAOs after the activation time T_A are reserved only for the collision resolution of the devices that have collided in their first access attempt during the activation time T_A. The time and the group of preambles for the next access attempt is assigned serially by the DTS for each collided preamble, i.e., the eNodeB searches for the nearest group of preambles in the nearest RAO that is not yet reserved and assigns it to the current group of collided devices. The number of preambles or branches in each group is determined according to the value of $CC_{i,k}$, which estimates the mean number of collisions per preamble for the ith RAO and the kth transmission where the collision was initiated. The value of $CC_{i,k}$ for the kth transmission of the ith RAO is computed as follows:

$$CC_{i,k} = \begin{cases} \frac{nc_{i,k}}{mc_{i,k}}, & \text{if } mc_{i,k} \ge 1 \\ 0, & \text{otherwise} \end{cases} \tag{10}$$

where $1 \le k \le k_{\max}$. The number of branches (preambles) $b_{i,k}$ that will be assigned to each group to be used in the next access

attempt is determined according to the following equation:

$$b_{i,k} = \begin{cases} \lfloor CC_{i,k} \rfloor , & \text{if } M \bmod CC_{i,k} = 0 \\ \lfloor CC_{i,k} \rfloor - 1 , & \text{if } M \bmod CC_{i,k} > 0 \\ 0, & \text{otherwise} \end{cases} \quad (11)$$

where $\lfloor . \rfloor$ is the floor function that is used to ensure that the value of $b_{i,k}$ is an integer less than or equal to the mean number of collisions per preamble. To ensure full utilization of the preambles, all preambles in a given RAO must be assigned to a group, therefore, M mod $CC_{i,k}$ must be equal to zero. The number of preamble groups $G_{i,k}$ for the kth transmission of the ith subtree can be computed as follows:

$$G_{i,k} = \frac{M}{b_{i,k-1}}. \quad (12)$$

The DTS for the collided devices starts immediately after the end of the burst arrivals (activation time) to avoid collisions with new arrivals. All the collided devices $nc_{i,1}$ results from the first transmission of the ith RAO start the second access attempt after $i + T_A$ time slots and compete within the predetermined number of preambles $b_{i,k}$. The number of preambles $M_{i,k}$ for the kth level of the ith subtree can be computed based on the number of collided preambles of the previous level as follows:

$$M_{i,k} = mc_{i,k-1} \times b_{i,k-1}, \quad 1 < k \leq k_{\max}. \quad (13)$$

Note that $M_{i,k}$ can be greater than M, which means that the kth transmission can extend over more than one RAO. Let $l_{i,k}$ denote the length of the level k, which is defined as the number of RAOs that are covered by the kth level, then

$$l_{i,k} = \frac{M_{i,k}}{M}. \quad (14)$$

The access success probability for the 2, 3,..., $k_{\max}$ transmission can be written as follows:

$$Ps_{i,k} = \frac{nc_{(i,k-1)}}{M_{i,k}} \left(1 - \frac{1}{M_{i,k}}\right)^{(nc_{(i,k-1)}-1)}, 1 < k \leq k_{\max}. \quad (15)$$

The idle probability for the kth level is

$$Pd_{i,k} = \left(1 - \frac{1}{M_{i,k}}\right)^{nc_{(i,k-1)}}, \quad 1 < k \leq k_{\max}. \quad (16)$$

The collision probability of the kth level can be computed as

$$Pc_{i,k} = 1 - Ps_{i,k} - Pd_{i,k}, \quad 1 < k \leq k_{\max}. \quad (17)$$

The number of successful devices for the kth level can be computed based on the number of preambles $M_{i,k}$ as follows:

$$ns_{i,k} = M_{i,k} \times Ps_{i,k}, \quad 1 < k \leq k_{\max}. \quad (18)$$

The number of the collided preambles for the kth level is

$$mc_{i,k} = M_{i,k} \times Pc_{i,k}, \quad 1 < k \leq k_{\max}. \quad (19)$$

The number of collided devices for the kth transmission can be written as

$$nc_{i,k} = nc_{i,k-1} - ns_{i,k}, \quad 1 < k \leq k_{\max}. \quad (20)$$

Algorithm 1: DTS Algorithm.

Input: T_A is the activation time period; $k_{\max}$ is the maximum number of transmissions; n_i is the number of arrival requests in the ith RAO; $mc_{i,k}$ is the number of collided preambles for the kth transmission of the ith RAO; $nc_{i,k}$ is the number of collided requests for the kth transmission of the ith RAO.

Output: $T(m_{i,k})$ is the time of the next transmission attempt; $g(m_{i,k})$ is the preamble group number of the next access attempt.

```
for i = 1 to MaxRAO do
  for k = 1 to k_max do
    if (mc_{i,k} = 0) then
      break;
    else
      Compute: CC_{i,k} based on (10);
      Compute: b_{i,k} based on (11);
      for m = 1 to mc_{i,k} do
        if (i <= T_A) then
          Set: T(m_{i,k}) = i + T_A; {Set a time for the
          next access attempt}
        else
          Set: T(m_{i,k}) = i + 1;
        end if
        for r = 1 to G_{i,k} do
          if (g(r) available) then
            Set: g(m_{i,k}) = g(r); {Assign the first
            available preamble group g(r) to the
            devices that collided in preamble m}
          end if
        end for
      end for
    end if
  end for
end for
```

After each RAO, the DTS will assign a time T and a group g to each collided preamble m to be used by its collided devices in the next access attempt. This information will be broadcasted by the eNodeB to all collided devices through Msg. 4b. The process of DTS is repeated until either $mc_{i,k} = 0$ or $k = k_{\max}$. The detailed steps of the DTS are shown in Algorithm 1.

C. Delay and Throughput Analysis

The delay experienced during the DTS is analyzed as the number of RAO slots required to serve all the N requests. It is shown in (14) that the number of RAOs required for the kth level is $l_{i,k}$. Moreover, the collision resolution of the ith subtree starts after T_A slots to avoid collision with new arrivals. Therefore, the number of RAOs required for the ith subtree to complete the DTS is

$$d_i = T_A + \sum_{k=1}^{k_{\max}} \frac{M_{i,k}}{M}. \quad (21)$$

The total delay D to complete the DTS for all the subtrees, which was initiated from each RAO of the activation time T_A can be computed as follows:

$$D = \sum_{i=1}^{T_A} d_i. \tag{22}$$

Since there are T_A subtrees, the mean delay $\overline{D}$ experienced by a subtree is

$$\overline{D} = \frac{D}{T_A}. \tag{23}$$

The throughput of the DTS for the ith RAO is analyzed as the number of successful requests per preamble and can be written as follows:

$$Tp_i = \frac{ns_i}{M}. \tag{24}$$

The mean throughput can be computed by dividing the total number of successful devices N_s by the total number of preambles required to complete the DTS. The total number of successful devices is

$$N_s = \sum_{i=1}^{T_A} \sum_{k=1}^{k_{\max}} \lceil ns_{i,k} \rceil . \tag{25}$$

Since there are D RAOs required to complete the DTS and each RAO contains M preambles, the mean throughput of the DTS is

$$\overline{Tp} = \frac{N_s}{M \times D}. \tag{26}$$

The mean throughput of the DTS is plotted in Fig. 9 as a function of the total number of devices N. The plot shows that the throughput is increased as the number of devices is increased. The maximum throughput of the DTS is achieved when $N = 50000$ and it reaches 0.3896 requests per preamble, which indicates that the DTS is efficient for massive arrivals of the M2M traffic.

VI. Performance Evaluation

Besides the mathematical analysis, the performance of the proposed DTS algorithm is evaluated through extensive simulations. The following subsections shows the details of the simulation.

A. Simulation Settings

The simulation of the DTS was conducted through a custom MATLAB simulator. We considered a single eNodeB serving N machine devices that are activated based on the beta distribution within T_A time slots. The total number of devices N ranged from 5000 to 50 000 UE requests. A total of 1000 independent simulations were performed, 100 replications for each number of devices, resulting in 96% confidence intervals. A maximum of eight transmission attempts for each UE were considered. Additional details of the simulation settings are shown in Table II. All the simulation settings follow the standard settings suggested by the 3GPP for M2M random access simulations [10].

TABLE II
Simulation Settings

Parameter	Setting
Total number of UE requests N	5,000 to 50,000
Total number of preambles M per initial RAO	54
Maximum number of transmission attempts k_{max}	8
Activation time period T_A	50 slots.
RACH configuration index	6
Subframe length	1 ms.
Periodicity of RAOs.	5 ms.
Backoff indicator for SRA and DAB	20 ms.
RAR window size	5 ms.

B. Benchmarks

The performance of the DTS was compared to several benchmarks. The first one was the LTE standard random access SRA with a standard backoff for collision resolution [10]. The second benchmark was the q-ary tree-splitting algorithm, which has been proposed to resolve RACH collisions for delay-sensitive M2M services [18]. The third benchmark was the hybrid random access (hybrid RA), for which the binary tree collision resolution was combined with dynamic access barring, which was used as a congestion control protocol to presmooth burst arrivals before they joined the tree collision resolution [20]. Moreover, we compared our proposed DTS algorithm to two state-of-the-art innovations: priority-based access class barring (PACB) [16] and the delay-aware double deep q-learning (DDQL) mechanism [17].

C. Performance Metrics

The performance of the DTS algorithm is evaluated based on the following four metrics.

1) The mean access delay, defined as the mean time required to serve all N requests from the starting instant of the burst arrivals until all requests are either successful or dropped.
2) The mean throughput, defined as the number of successful requests N_s divided by the mean time period in preambles from the beginning of activation time until all N requests are either successful or dropped.
3) The success rate defined as the ratio of successful requests to the total number of devices N, which can be easily computed as

$$s_r = \frac{N_s}{N}. \tag{27}$$

4) The mean number of transmissions, defined as the mean number of transmission attempts performed by each device until it either successfully accesses the network or is dropped.

VII. Evaluation Results and Discussion

This section presents an evaluation of the proposed DTS algorithm based on the analysis and simulation results. It further provides a comparison between the DTS algorithm and the previous benchmarks.

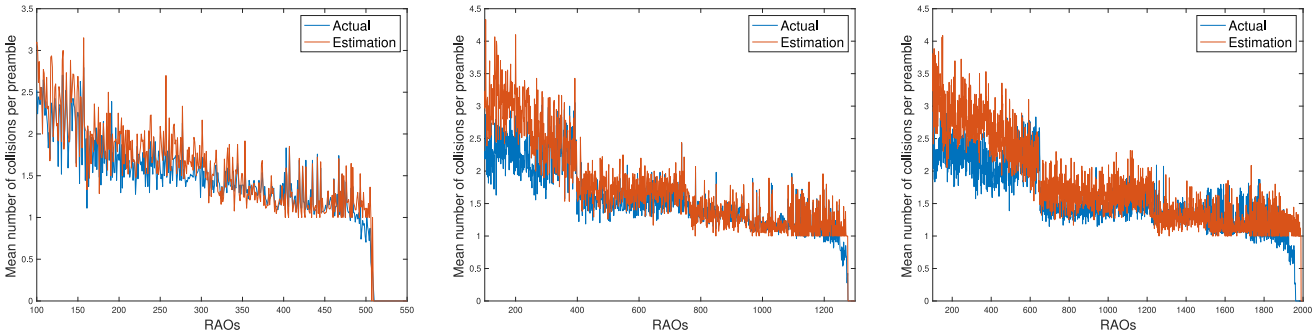


Fig. 3. Mean number of collisions per preamble for DTS with $M = 54$ preambles, $k_{\max} = 8$, and a different number of devices N. (a) N = 10,000. (b) N = 30,000. (c) N = 50,000.

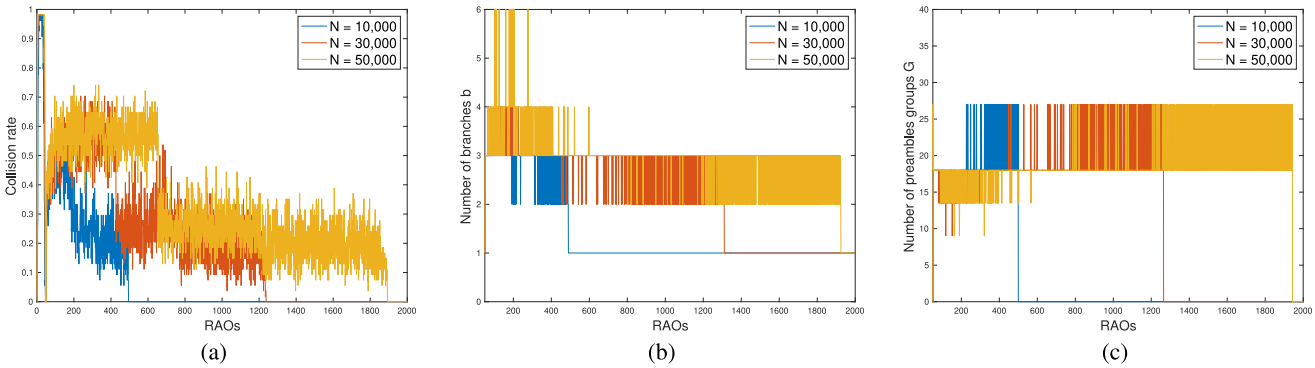


Fig. 4. Relation between collision rate, number of branches, and number of groups per RAO for DTS with $M = 54$ preambles, $k_{\max} = 8$, and a different number of devices N. (a) Collision rate. (b) Number of branches. (c) Number of groups.

A. DTS Evaluation

In this section, we present numerical results to demonstrate the performance of DTS under various loads and to clarify the relation between DTS parameters. First, we plot the mean number of collisions per preamble for the DTS algorithm in Fig. 3. We compare our estimation of $CC_{i,k}$, which has been produced based on (10) for each RAO, to the actual value of the mean number of collisions, which was extracted from a controlled simulation scenario in which the number of collided devices per preamble is known. The results in Fig. 3 show that our estimation is accurate even for a large number of devices, with a maximum 0.3 estimation error for the early RAOs. However, this error starts to reduce with time until it reaches less than 0.1 for the late RAOs. It is also seen in Fig. 3 that DTS helps to reduce the number of collisions per preamble for the upcoming RAOs until all collisions are resolved. Moreover, it is seen in Fig. 3 that the number of RAOs required to solve all collisions is increased as the total number of devices is increased. This increment results from the higher collision rate caused by the large number of devices. However, DTS aims to minimize the required number of RAOs by optimizing the number of branches in each RAO. Fig. 4(a) shows the collision rate for different values of N based on (17). It is clearly seen that the collision rate is very high in the first 50 RAOs (activation time). However, after the start of DTS (RAOs $>$ 50), the collision rate starts to drop until it reaches 0 (all collisions are resolved). Moreover, Fig. 4(b) shows the number of branches based on (11) for each RAO, which is determined based on the collision rate of the respective RAO. It can be observed from Fig. 4(a) and (b) that the number of branches per RAO is increased as the collision rate is increased in the respective RAO. The number of branches is also increased as the total number of devices N is increased. For example, when $N = 50\,000$, the number of branches ranges from 2 to 6, while it ranges from 2 to 3 when $N = 10\,000$ devices. However, the number of contention groups per RAO is reduced as the collision rate is increased. Fig. 4(c) shows that the number of groups per RAO is fewer for the early RAOs and increases for the subsequent RAOs, which have a lower collision rate. In fact, this is logical because the relation between the number of branches and the number of groups is an inverse relationship, as seen in (12). The main advantage of this dynamic splitting is to increase preamble utilization, and therefore, reduce access delay. The preamble utilization is reflected in the throughput of the RAO, which was defined in the analysis as the number of successful requests per preamble. The throughput of the DTS is shown in Fig. 5, based on (24) for a different number of devices. It is clearly seen that the throughput for the early RAOs is very low. However, after the implementation of DTS (RAOs $>$ 50), the

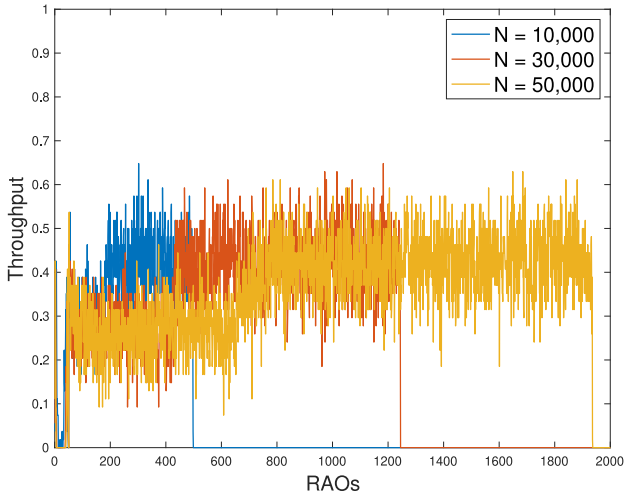


Fig. 5. RACH throughput for DTS with $M = 54$ preambles, $k_{\max} = 8$, and a different number of devices N.

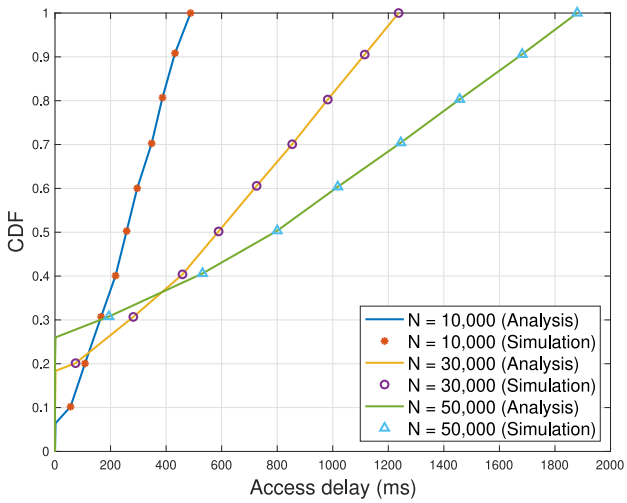


Fig. 6. CDF of access delay for DTS with $M = 54$ preambles, $k_{\max} = 8$, and a different number of devices N.

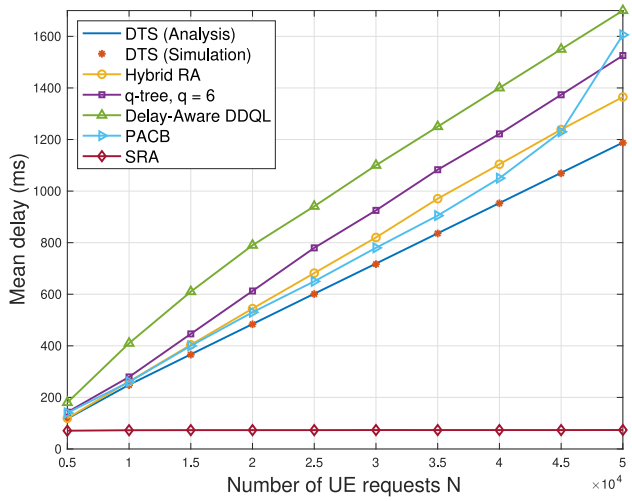


Fig. 7. Mean access delay.

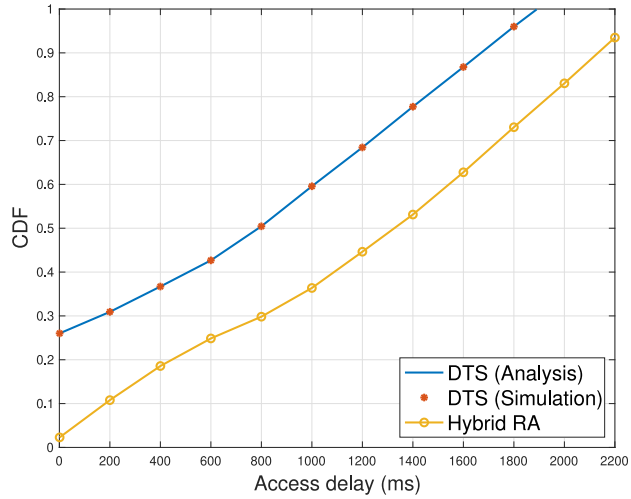


Fig. 8. CDF of access delay for DTS and hybrid RA with $M = 54$ preambles, $k_{\max} = 8$, and $N = 50\,000$.

throughput starts to increase until it reaches its maximum value before all collisions are resolved. It is also seen in Figs. 4(a) and 5 that the higher collision rate causes the low throughput for the respective RAOs. However, the DTS helps to reduce the collision rate for the upcoming RAOs, which in turn helps to increase the throughput of the RAOs.

The access delay of the DTS algorithm is shown in Fig. 6. Since we are targeting delay-sensitive devices, we plotted the cumulative distribution function (CDF) of the access delay to show the percent of devices that successfully accessed the network within a specific time guarantee. The presented results were obtained from the analysis and simulations of different loads. We define the access delay for a given UE as the time period from the arrival of the UE until it successfully completes the random access procedure. Fig. 6 shows that 100% of successful devices completed the random access procedure in less than 2 s when $N = 50\,000$, which is considered a short time in light of the huge number of devices activated within a very short activation time. Furthermore, if we consider a stricter deadline for the access delay (e.g., 1 s), we can see that 100% of successful devices meet the deadline when $N = 10\,000$. However, this percent starts to decrease for the same deadline when the total number of devices N is increased, i.e., when $N = 30\,000$, only 80% of successful devices meet the deadline and 60% when $N = 50\,000$. It is logically expected that the access delay is increased as the total number of activated devices is increased within the same activation time because the arrivals rate will be much higher.

B. Comparison to Previous Solutions

The performance of the DTS and the considered benchmarks is plotted in Figs. 7–11 as a function of the total number of devices N. The mean access delay of the DTS is shown in Fig. 7. It is clear that the DTS has reduced the access delay approximately 12.5% compared to the hybrid RA and around 29% compared to the delay-aware DDQL algorithm. The hybrid

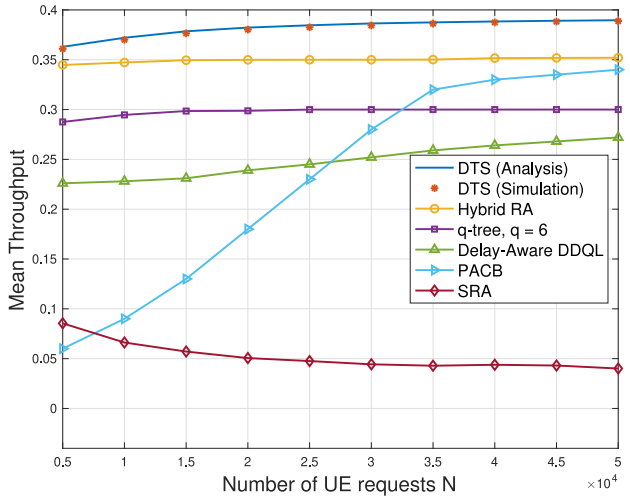


Fig. 9. Mean throughput.

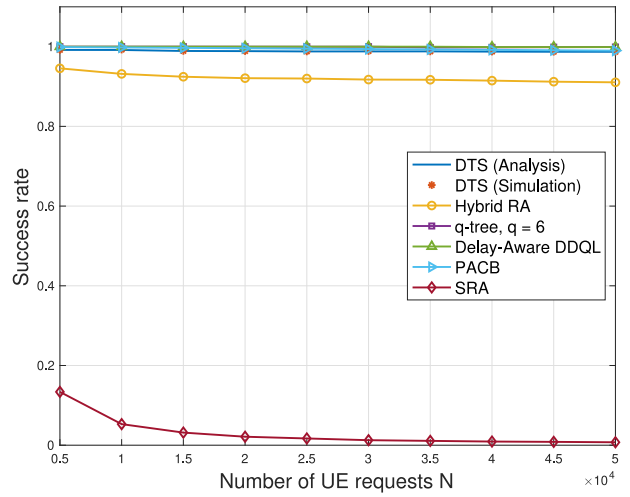


Fig. 10. Success rate.

RA and the delay-aware DDQL adopted dynamic ACB as a congestion control scheme, which requires the activated requests to wait for some time before the first access attempt. This process is not appropriate for delay-sensitive devices because it increases the access delay. However, the DTS scheme allows the activated requests to immediately proceed to the first access attempt, only after which the collided devices join the DTS. The dynamic allocation of the preambles to each group of collided devices increases the success probability of the next access attempt, which in turn reduces the access delay of the DTS algorithm. Moreover, DTS reduced the access delay around 22% compared to the q-arry algorithm. This is because the q-arry algorithm assigns a static number of preambles (equal to q) to each collided group without considering the number of collided devices, which causes the preambles to be wasted when the number of collided devices is less than q, and therefore, increases the access delay. However, the DTS optimizes the allocation of preambles according to the mean number of collided devices in each group, which in turn reduces the access delay. Furthermore, we plotted the access delay of PACB only for the highest priority class that achieved the lowest delay compared to the other PACB classes. However, the DTS reduced the access delay approximately 25% compared to the PACB algorithm when the total number of devices was equal to 50 000 and around 5% when the total number of devices was equal to 30 000. Note that the results of the PACB represent the access delay for only one class, which contains only 33% of total devices. Furthermore, Fig. 7 shows that the access delay of SRA is very low compared to the other schemes. The reason for this is that most of the access requests in the SRA scheme collided and reached the maximum number of transmission attempts, which caused them to be dropped in a short time period. The access delay of SRA is observed to be static even for a large number of devices because of the limited capacity of the RACH under the SRA scheme and the short static backoff time, which causes a massive arrival to be dropped within a short time even for a large number of devices. This indicates that the standard backoff collision resolution is not efficient in resolving massive collisions because it allows the collided devices to try the next access attempt after a short time (usually equal to 20 ms), which causes the backoff devices to collide again with new massive arrivals. In contrast, DTS defers the second access attempt until the burst arrivals are reduced, which allows the backoff devices to avoid collisions with new burst arrivals.

Furthermore, in Fig. 8, we compare the CDF of the access delay for the DTS and the hybrid RA when $N = 50\,000$. Our comparison is limited to the hybrid RA scheme because it achieved the best mean access delay among all benchmarks. Hence, comparison to the rest of the benchmarks would be meaningless since all of them achieved a higher mean access delay than the hybrid RA. The results show that when implementing the DTS algorithm, 100% of successful devices completed the random access procedure in less than 1.9 s, while only 80% of devices completed the random access within the same access delay when implementing the hybrid RA. This indicates that DTS improved the access delay for around 20% of devices, compared to the hybrid RA.

The mean throughput obtained from the simulation is shown in Fig. 9 as well as the expected throughput from the DTS analysis [see (26)]. Fig. 7 shows that the mean throughput of DTS is increased as the total number of devices increases, which indicates that DTS is efficient in handling large numbers of machine devices. Moreover, the DTS algorithm increased the mean throughput of the RACH around 12% compared to the best benchmark (hybrid RA) and around 26% compared to the q-ary scheme. This indicates that the dynamic splitting of the DTS algorithm is more efficient than the static splitting of the q-ary scheme because the DTS optimizes the usage of preambles according to collision intensity. Moreover, DTS improved the RACH throughput around 11% compared to PACB and around 40% compared to the delay-aware DDQL algorithm. It can also be seen in Fig. 7 that the throughput of the SRA with a standard backoff collision resolution is very low (below 0.1), which implies that SRA is not efficient in resolving RACH collisions for massive arrivals.

The access success rate plotted in Fig. 10 shows that DTS increased the success rate by approximately 10% compared to

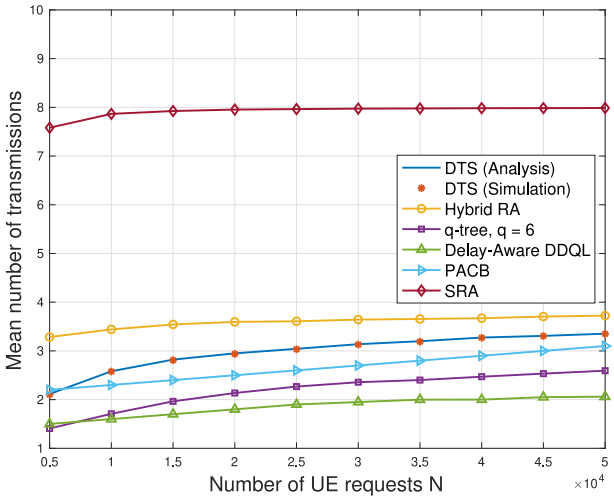


Fig. 11. Mean number of transmission attempts.

TABLE III
IMPACT OF DIFFERENT VALUES OF k_{MAX} ON DIFFERENT PERFORMANCE INDICES

k_{max}	Mean throughput	Mean delay (ms)	Mean number of transmissions	Success rate
8	0.3864	718	3.13	0.99000
10	0.3858	750	3.20	0.99700
12	0.3851	766	3.26	0.99920
14	0.3869	773	3.31	0.99990
16	0.3831	777	3.35	0.99997

the hybrid RA and around 98% compared to the SRA. DTS achieves a very high success rate, almost equal to the success rates of the q-ary, PACB and delay-aware DDQL, which are over 0.98. The high success rate indicates the reliability of the DTS algorithm in massive arrival scenarios, even for a large number of devices. It is also shown that the SRA has a very low success rate, which implies that most of the access requests could not access the network. This supports our analysis regarding the short access delay of the SRA, which results from the high drop rate caused by the short backoff time.

Fig. 11 shows that DTS has achieved a moderate mean number of transmission attempts, ranging from 2 to 3 attempts per device compared to 3 to 4 attempts for the hybrid RA. This indicates that DTS is efficient in saving the energy of machine devices that are mostly battery operated. The q-ary, PACB, and delay-aware DDQL algorithms achieved a lower number of transmission attempts at the expense of higher access delays. In contrast, most devices in SRA reached the maximum number of allowed transmissions, and therefore, dropped, which explains the low success rate.

Furthermore, the presented results are affected by the settings of the maximum number of transmission attempts $k_{\max}$. To investigate this issue, we present the analysis results for $N = 30000$ with different values of $k_{\max}$ ranging from 8 to 16 in Table III. It can be seen that the success rate increases as the number of maximum transmissions $k_{\max}$ is increased. The larger values of $k_{\max}$ allow the UEs to make more access attempts, which increases the success probability. Moreover, the mean delay is increased as the maximum number of transmissions is increased. This is reasonable since each transmission attempt requires the devices to wait for a new RAO, which in turn increases the access delay. The mean number of transmissions is also increased with larger values of $k_{\max}$ because the devices have the chance to perform a higher number of transmissions before they can be dropped. Furthermore, the mean throughput decreases very slightly with an increased number of maximum transmissions, though the impact should be considered insignificant.

VIII. CONCLUSION AND FUTURE WORK

This article has proposed a DTS algorithm to efficiently resolve the collisions of RACH in massive access scenarios for delay-sensitive services in M2M communications. In this algorithm, the devices that have collided in the same preamble are directed to a specific number of preambles for their next access attempts, which allows the collided devices to be divided into several contending groups. The number of preambles assigned to each group is determined based on the mean number of collisions in the group. This dynamic splitting helps to increase the utilization of the RACH resources, which in turn increases the RACH throughput and reduces the access delay. A mathematical analysis of the proposed algorithm was presented, and the mean throughput and delay were derived. The analysis and simulation results show that the proposed DTS algorithm improved RACH throughput and access delay by approximately 12% compared to the best benchmark, which validates the efficiency of the DTS algorithm. Moreover, DTS achieved high success rates and a moderate number of preamble transmissions, which indicates the reliability of the proposed algorithm.

Future work could include optimizing the random access for heterogeneous M2M services to deliver better service based on different QoS requirements. The collision resolution splitting may be conducted based on the different priorities of devices. Furthermore, the performance of the DTS algorithm could be investigated under different arrival models, considering different traffic types.

ACKNOWLEDGMENT

The authors would like to thank everyone who provided support to improve the content of this article.

REFERENCES

[1] 3GPP, "Medium access control (MAC) protocol specification (Release 8)," ETSI, Sophia Antipolis, France, TS 36.321,V16. 4.0, 2021.
[2] H. Althumali and M. Othman, "A survey of random access control techniques for machine-to-machine communications in LTE/LTE-A networks," *IEEE Access*, vol. 6, pp. 74 961–74983, 2018.
[3] 3GPP, "Technical specification group services and system aspects, service accessibility (Release 9)," ETSI, Sophia Antipolis, France, TS 22.011, V9.4.0, 2010.
[4] 3GPP, "Further performance evaluation of EAB information update mechanisms," ETSI, Sophia Antipolis, France, R 2–120270, 2012.
[5] Z. Wang and V. W. Wong, "Optimal access class barring for stationary machine type communication devices with timing advance information," *IEEE Trans. Wireless Commun.*, vol. 14, no. 10, pp. 5374–5387, Oct. 2015.

[6] S. Duan, V. Shah-Mansouri, Z. Wang, and V. W. Wong, "D-ACB: Adaptive congestion control algorithm for bursty M2M traffic in LTE networks," *IEEE Trans. Veh. Technol.*, vol. 65, no. 12, pp. 9847–9861, Dec. 2016.
[7] M. Alvi, K. M. Abualnaja, W. T. Toor, and M. Saadi, "Performance analysis of access class barring for next generation IoT devices," *Alexandria Eng. J.*, vol. 60, pp. 615–627, 2021. [Online]. Available: https://doi.org/10.1016/j.aej.2020.09.055
[8] L. Tello-Oquendo *et al.*, "Performance analysis and optimal access class barring parameter configuration in LTE-A networks with massive M2M traffic," *IEEE Trans. Veh. Technol.*, vol. 67, no. 4, pp. 3505–3520, Apr. 2018.
[9] L. Tello-Oquendo, D. Pacheco-Paramo, V. Pla, and J. Martinez-Bauset, "Reinforcement learning-based ACB in LTE-A networks for handling massive M2M and H2H communications," in *Proc. IEEE Int. Conf. Commun.*, 2018, pp. 1–7.
[10] 3GPP, "Study on RAN improvements for machine type communications," ETSI, Sophia Antipolis, France, TR 37.868, 2011.
[11] W. Jiang, X. Wang, and T. Deng, "Performance analysis of a pre-backoff based random access scheme for machine-type communications," in *Proc. Int. Conf. Intell. Green Building Smart Grid*, 2014, pp. 1–4.
[12] R. Harwahyu, X. Wang, R. F. Sari, and R.-G. Cheng, "Analysis of group paging with pre-backoff," *EURASIP J. Wireless Commun. Netw.*, vol. 2015, no. 1, 2015, Art. no. 34.
[13] H. Althumali, M. Othman, N. K. Noordin, and Z. M. Hanapi, "Dynamic backoff collision resolution for massive M2M random access in cellular IoT networks," *IEEE Access*, vol. 8, pp. 201 345–201 359, Nov. 2020.
[14] 3GPP, "MTC simulation results with specific solutions," ETSI, Sophia Antipolis, France, TR R 2–104662, 2010.
[15] Y. Chang, C. Zhou, and O. Bulakci, "Coordinated random access management for network overload avoidance in cellular machine-to-machine communications," in *Proc. 20th Eur. Wireless Conf.*, 2014, pp. 1–6.
[16] Y. Sim and D.-H. Cho, "Performance analysis of priority-based access class barring scheme for massive MTC random access," *IEEE Syst. J.*, vol. 14, no. 4, pp. 5245–5252, Dec. 2020.
[17] D. Pacheco-Paramo and L. Tello-Oquendo, "Delay-aware dynamic access control for mMTC in wireless networks using deep reinforcement learning," *Comput. Netw.*, vol. 182, 2020, Art. no. 107493.
[18] G. C. Madueno, Č. Stefanović, and P. Popovski, "Efficient LTE access with collision resolution for massive M2M communications," in *Proc. IEEE Globecom Workshops*, 2014, pp. 1433–1438.
[19] J. Capetanakis, "Tree algorithms for packet broadcast channels," *IEEE Trans. Inf. Theory*, vol. 25, no. 5, pp. 505–515, Sep. 1979.
[20] H. M. Gürsu, M. Vilgelm, W. Kellerer, and M. Reisslein, "Hybrid collision avoidance-tree resolution for M2M random access," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 53, no. 4, pp. 1974–1987, Aug. 2017.
[21] 3GPP, "MTC LTE simulations," ETSI, Sophia Antipolis, France, RAN WG2 71 R2-104663, 2010.
[22] 3GPP, "5G; NR; Medium access control (MAC) protocol specification (Release 15)," ETSI, Sophia Antipolis, France, TS 38.321,V15.6.0, 2019.
[23] M. Koseoglu, "Lower bounds on the LTE-A average random access delay under massive M2M arrivals," *IEEE Trans. Commun.*, vol. 64, no. 5, pp. 2104–2115, May 2016.
[24] B. Yang, G. Zhu, W. Wu, and Y. Gao, "M2M access performance in LTE-A system," *Trans. Emerg. Telecommun. Technol.*, vol. 25, no. 1, pp. 3–10, 2014.
[25] O. Arouk and A. Ksentini, "General model for RACH procedure performance analysis," *IEEE Commun. Lett.*, vol. 20, no. 2, pp. 372–375, Feb. 2015.

Huda Althumali (Graduate Student Member, IEEE) received the bachelor's degree in computer science from Taif University, Taif, Saudi Arabia, in 2009, and the master's degree in computer science, with a specialization in distributed computing, in 2017 from Universiti Putra Malaysia, Seri Kembangan, Malaysia, where she is currently working toward the Ph.D. degree in computer networks.

She is currently a Lecturer with Imam Abdulrahman Bin Faisal University, Al Jubail, Saudi Arabia. Her research interests include wireless networks, the Internet of Things, cloud computing, and network security.

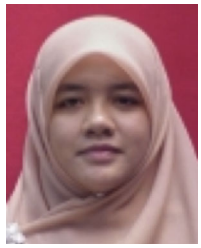
Mohamed Othman (Senior Member, IEEE) received the Ph.D. (Hons.) degree in computer science from the National University of Malaysia, Bangi, Malaysia, in 1999.

He was the Deputy Director with the Information Development and Communication Centre, where he was the in charge of the UMPNet network campus, uSport Wireless Communication Project, and the Universiti Putra Malaysia (UPM) Data Centre, Seri Kembangan, Malaysia. He is currently a Professor in computer science with the Department of Communication Technology and Networks, UPM. He is also working as an Associate Researcher and a Coordinator of high-speed machines with the Laboratory of Computational Science and Informatics, Institute of Mathematical Science, UPM, and a Visiting Professor with South Kazakhstan State University, Shymkent, Kazakhstan, and L. N. Gumilyov Eurasian National University, Astana, Kazakhstan. He has led six Malaysian, one Japanese, one South Korean, and three U.S. patents. He has authored or coauthored more than 300 international journal articles and 330 proceeding articles. His research interests include computer networks, parallel and distributed computing, high-speed interconnection networks, network design and management (network security, and wireless and traffic monitoring), consensus in the Internet of Things, and mathematical models in scientific computing.

Dr. Othman is a Life Member of the Malaysian National Computer Confederation and the Malaysian Mathematical Society. In 2017, he was the recipient of the Honorary Professorship from SILKWAY International University (formerly known as South Kazakhstan Pedagogical University), Shymkent, and the Best Ph.D. thesis by Sime Darby Malaysia and the Malaysian Mathematical Science Society, in 2000.

Nor Kamariah Noordin (Senior Member, IEEE) received the B.Sc. degree in electrical engineering, with a major in telecommunications, from the University of Alabama, Tuscaloosa, AL, USA, in 1987, the master's degree in electrical engineering (telecommunications) from University Teknologi Malaysia, Johor Bahru, Malaysia, in 1990, and the Ph.D. degree in communications and network engineering from University Putra Malaysia (UPM), Seri Kembangan, Malaysia, in 2006.

Since 1988, she has been with UPM. In 2006, she was appointed as an Associate Professor, and in 2012, a Professor. During her tenure, she has been the Head of the Department, the Deputy Dean of academics, and the Director of Corporate Strategy and Communication of the university. She is currently the Dean of the Faculty of Engineering, UPM. She has been with UPM for more than 25 years. She has authored or coauthored more than 200 journal articles and conference papers, and has secured more than 30 research and consultancy projects.



Zurina Mohd Hanapi (Member, IEEE) received the B.Sc. degree in computer and electronic systems from the University of Strathclyde, Glasgow, U.K., in 1999, the M.Sc. degree in computer and communication system engineering from Universiti Putra Malaysia (UPM), Seri Kembangan, Malaysia, in 2004, and the Ph.D. degree in electrical, electronic, and systems engineering from Universiti Kebangsaan Malaysia, Bangi, Malaysia, in 2011.

Since 2004, she has been a Lecturer with UPM. She is currently working as an Associate Professor with the Department of Communication Technology and Networks, Faculty of Computer Science and Information Technology, UPM. She has authored or coauthored more than 70 papers in cited journals and conferences in the area of security and wireless sensor networks. Her current research interests include security, routing, wireless sensor networks, wireless networks, distributed computing, and cyber-physical-systems.

Dr. Hanap is a Member of the Malaysian Security Research Committee.