IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received September 25, 2018, accepted October 8, 2018, date of publication November 9, 2018, date of current version December 3, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2876438

An Efficient Contention Resolution Scheme for Massive IoT Devices in Random Access to LTE-A Networks

KYUNGJUN LEE AND JU WOOK JANG
Department of Electronic Engineering, Sogang University, Seoul 04107, South Korea
Corresponding author: Kyungjun Lee (scomchj@sogang.ac.kr)

This work was supported by the Special Research Grant of Sogang University.

ABSTRACT Long-Term Evolution-Advanced (LTE-A) networks have been regarded as having great potential to support the Internet of Things (IoT). However, the high probability of preamble collisions when massive numbers of IoT devices try to access the network within a short period has become a significant problem. In this paper, we propose a new scheme to efficiently handle the initial access contention for massive numbers of IoT devices in LTE-A networks. The proposed scheme divides an entire preamble set into k subgroups; then, contention resolution is performed in each subgroup independently in a parallel and pipelined manner. Simulation results show that our scheme leads to considerable improvement in terms of average access delay and standard deviation of access delays without increasing blocking probability. Moreover, utilization of preamble resources is greatly improved by reducing the number of random access slots, where only a few devices participate in preamble selection.

INDEX TERMS Internet of Things, contention resolution, LTE-advanced, machine-to-machine (M2M) communications, MTC, random access procedure.

I. INTRODUCTION

With the benefit of widely deployed infrastructure and the support of long-range high-mobility communications [4]–[6], the Third Generation Partnership Project (3GPP) Long-Term Evolution Advanced (LTE-A) is regarded as having great potential to support the Internet of Things (IoT) [7]. However, IoT traffic may burden traditional cellular networks, which are supposed to carry human-type communication (HTC), because machine-type communication (MTC) typically features a large number of IoT devices trying to access the network within a very short period [8]. These access attempts cause a shortage of random access (RA) resources and lead to a high probability of RA preamble collisions. Therefore, efficient management of massive RAs is one of the most critical challenges in LTE-A networks when a large number of IoT devices attempt RA with limited RA resources.

In LTE-A, devices perform a contention-based RA procedure to access the network for the first time, as shown in Fig. 1. Each device randomly selects one among 64 or fewer preambles when attempting to access the network [9], [10]. The devices then transmit a randomly chosen preamble to the evolved Node B (eNB) in an RA slot. It is possible for two or more devices to transmit the same RA preamble simultaneously. In this case, acknowledgement by the eNB of having received the RA preamble is not enough. The eNB should further perform contention resolution, through which the eNB should indicate which device's

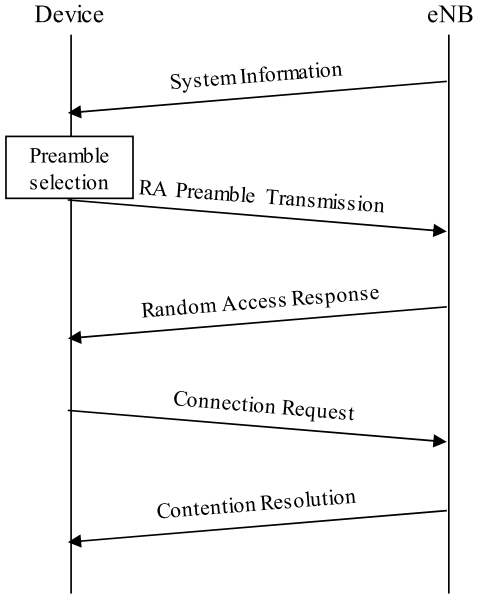

FIGURE 1. Contention-based RA procedure in LTE-A.

TABLE 1. Comparison of contention resolution schemes.

Scheme	Contention resolution	Description
Laya *et al.* [1], [2]	Sequential	Each collided preamble is resolved one by one in each RA slot. For example, collisions in three preambles are resolved sequentially in three RA slots.
Yoon [3]	Parallel	k or fewer collided preambles from current preamble selection are grouped together and resolved one at a time in an RA slot.
Proposed	Parallel + Pipelined	Preambles are divided into subgroups, and contention resolution in individual subgroups is performed in parallel. If some subgroups have no collisions, then newly arriving devices may perform preamble selection in those subgroups, thus reducing overall access latency due to pipelining effects.

transmission has actually been received. Thus, to resolve contention, when a device transmits a connection request message, the device includes its identification in the message. The eNB transmits a contention resolution message. The device that receives this message compares the identity in the message with the previously transmitted identity. A device detecting a match between these identities may regard its RA procedure as having been successful. However, if a device fails to detect its matching identifier, then the device will attempt a new RA procedure [23]. Therefore, if a massive number of devices try to access the network simultaneously, the probability of preamble collision will drastically increase, hence leading to a low RA success rate and high network congestion.

To address this concern, access class barring (ACB) is proposed as an additional access control mechanism before performing RA. ACB regulates the opportunity for devices to participate in an RA attempt with a configured probability provided by the system. A device draws a random number between zero and one, and compares it with the barring factor provided by the system. If the value of the random number is less than the factor, then the device performs the RA procedure. Otherwise, the device retries the RA after a delay lasting the configured barring time.

Extended access barring (EAB) was introduced in 3GPP LTE Release 11 [10], [11] to control for potential surges in access requests. The basic idea of EAB is that the devices are divided into 10 access classes, and the devices belonging to certain access classes are not allowed to perform RA. However, EAB does not specify how to determine the activation time or which parameter values should be used.

In this study, we propose a new contention resolution scheme to efficiently handle the initial access for massive numbers of IoT devices in LTE-A networks. The proposed scheme divides an entire preamble set into k, and contention resolution in individual subgroups is performed in parallel. If preamble collision occurs in a certain subgroup, only that subgroup will be exclusively reserved to resolve the preamble collision. If some subgroups have no collision, then newly arriving devices may perform preamble selection in those subgroups. In order to do so, we extend random access response (RAR) to include which subgroups are available for newly arriving devices in the next RA slot. They receive the RAR to determine whether they can participate in preamble selection and which subgroups they can use in the next RA slot. The proposed scheme has advantages for parallel processing, in that each k preambles are resolved in an RA slot, and pipelined processing in the newly arriving devices can participate in preamble selection in a free subgroup while collisions are resolved in the other subgroups. Consequently, our scheme can lead to a remarkable improvement in terms of average access delay, the standard deviation of the access delays, and utilization of preamble resources, unlike in previous works [1]–[3].

This paper is organized as follows. Section II presents related work and we provide a definition of the problem in section III. Laya *et al.*'s scheme [1], [2], and Yoon's scheme [3] are reviewed in section IV. In section V, we explain our scheme for incorporating DQ mechanism and describe a new feedback design for implementing the scheme. The performance is evaluated in section VI, and section VII concludes this paper.

II. RELATED WORK

There are several papers in the literature that investigate RA mechanisms involving radio access network (RAN) congestion control in LTE-A networks.

Over the last few years, a number of schemes to enhance access class barring (ACB) mechanisms have been proposed [7], [12]–[14]. In [12] and [13], the access-barring parameters are determined by the expected radio access network (RAN) load. In [7], Lin *et al.* pre-allocated RA resources for different MTC classes to prevent a large number of devices from accessing the network at the same time. In [14], Wang and Wong used ACB and timing advance information to relieve the RA load. However, these RAN congestion control schemes may not work under considerable network congestion [24].

The separation of RA resources for human-to-human (H2H) and machine-to-machine (M2M) traffic was introduced to help reduce the negative impact on non-M2M devices [11], [15]. Nevertheless, these solutions provide limited benefits and show worse performance under high M2M traffic loads. The backoff adjustment was suggested to improve RA performance [16]. However, these schemes cannot provide a feasible solution with an appropriate balance among access delay, access success probability, and energy consumption.

Distributed queuing (DQ)-based schemes [1]–[3] have attracted our attention, since researchers claim their schemes can support a huge number of IoT devices in RA. The DQ mechanism [17]–[19] is based on an m-ary tree-splitting algorithm with a simple set of rules to organize devices. When preamble collisions occur, colliding devices are split into groups for subsequent attempts. The set of devices in each group performs RA one at a time, sequentially. This reduces the collision probability by decreasing the number of simultaneous RA attempts.

In Laya *et al.* [1], [2], each collided preamble is resolved one by one in an RA slot. For example, when collisions occur in three preambles, each of the preambles is resolved sequentially in the next three RA slots. Therefore, this scheme can guarantee a low collision probability even for massive numbers of access attempts. According to [1], it greatly reduces the access delay compared with the 3GPP standard when the total number of preambles is three and six. However, when the number of preambles is large, such as 56, it shows an even longer access delay than the 3GPP standard, based on the simulation results in [2]. As the total number of preambles increases from three to 56, the preamble collision probability decreases, but the number of colliding devices in each preamble also decreases. Thus, fewer devices are resolved in an RA slot, which increases the total number of RA slots needed to resolve all the collisions. For example, when 112 devices select 56 preambles and each two devices select the same preamble, at least 56 RA slots are needed to resolve all the collisions. In each RA slot, two devices participate in preamble selection with 56 preambles. This increases the average access delay and decreases preamble resource utilization, in that all of the preambles are reserved to deal with a few devices.

Yoon [3] proposed another DQ algorithm with preamble grouping to reduce access delay. The main mechanism of the scheme is that it resolves k of the preambles in which collisions occurred simultaneously in an RA slot. This has clear advantages over Laya *et al.*'s algorithm because up to k preambles are resolved in an RA slot. However, the expected degree of parallelism from preamble grouping is not achieved because fewer than k preambles are resolved at a time when the number of devices that participate in preamble selection simultaneously is small.

III. PROBLEM DEFINITION

Let us assume that n IoT devices are within a cell managed by an eNB and the arrival of devices follows a beta distribution with $\alpha = 3$, $\beta = 4$ [4], [11], [20], [21]. When two or more devices transmit the same preamble, collision will occur in that preamble. In this case, we assume that eNB will not transmit the random access response (RAR) as an acknowledgement of the preamble.

The following is a brief description of the preamble selection:

For a time interval during which n devices arrive and complete preamble selection (whether successful or not),

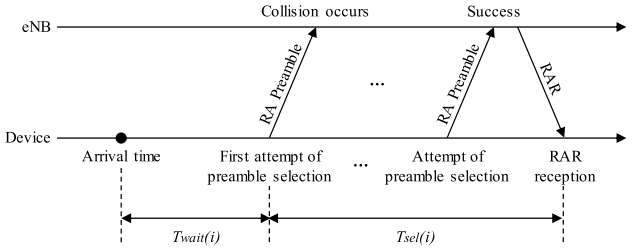

FIGURE 2. The access delay.

the following is performed for each arriving device i ($i = 0, 1, \ldots, n-1$)

for $m = 0$ to *MAX_PREAMBLE_SELECTION_ ATTEMPT*–1
  randomly choose a preamble among the available preambles
  If RAR is received, // no preamble collision
    return success
  else // preamble collision occurred
    continue
  end for
  return failure

In the above procedure, the access delay for device i can be divided into waiting delay $T_{wait}(i)$ and selection delay $T_{sel}(i)$, as shown in Fig. 2. $T_{wait}(i)$ denotes the time interval for device i from the arrival time to the first preamble selection attempt, and $T_{sel}(i)$ denotes the time interval for device i between the first preamble selection attempt and the reception of RAR, including the device's own RA preamble ID in the RAPID subheader. Equations (1), (2), (3), and (4) represent their relationship. $T_{arrival}(i)$ denotes the time when the device arrives, $T_{first\_attempt}(i)$ denotes the time of the first preamble selection attempt, and $T_{success}(i)$ denotes the time of reception of the RAR, including the device's own RA preamble ID in the RAPID subheader.

$$T_{access}(i) = T_{wait}(i) + T_{sel}(i) \tag{1}$$

$$T_{wait}(i) = T_{first\_attempt}(i) - T_{arrival}(i) \tag{2}$$

$$T_{sel}(i) = T_{success}(i) - T_{first\_attempt}(i) \tag{3}$$

$$T_{access}(i) = T_{success}(i) - T_{arrival}(i) \tag{4}$$

We consider only successful accesses for evaluating the average and standard deviation of access delays experienced by n devices, as in Equations (5) and (6). $n_{failure}$ denotes the number of devices that fail in preamble selection even after reaching the maximum number of attempts for preamble selection, MAX_PREAMBLE_SELECTION_ATTEMPT. A large standard deviation implies that certain devices experience much longer access delays than others.

$$E[T_{access}] = \frac{\sum_{i=0}^{n-1} T_{access}(i)}{n - n_{failure}} \tag{5}$$

$$\sigma[T_{access}] = \sqrt{\frac{\sum_{i=0}^{n-1} \{T_{access}(i) - E[T_{access}]\}^2}{n - n_{failure}}} \tag{6}$$

The blocking probability is defined as the ratio of devices that failed in the preamble selection to the total number of devices that tried to obtain access, as shown in Equation (7):

$$P_{block} = \frac{n_{failure}}{n} \tag{7}$$

Our objective is to minimize the average access delay, $E[T_{access}]$, while minimizing the standard deviation of access delays, $\sigma[T_{access}]$, and the blocking probability, P_{block}.

IV. PREVIOUS SCHEMES BASED ON DQ

We discuss in details the existing DQ-based schemes [1]–[3] which have attracted our attention in that they can support a large number of IoT devices in RA for LTE-A networks.

In Laya *et al.*'s scheme [1], [2], when preamble collisions occur, the next attempts by devices are dispersed along the time domain to reduce the preamble collision probability in the next attempts. The scheme resolves each collided preamble one by one in each RA slot (time slot). For example, when collisions occur in j preambles (i.e. $P_0, P_1, \ldots, P_{j-1}$) in RA slot t, P_0 is resolved in RA slot $t+1$, P_1 is resolved in RA slot $t+2, \ldots$, and P_{j-1} is resolved in RA slot $t+j$, in a serial manner.

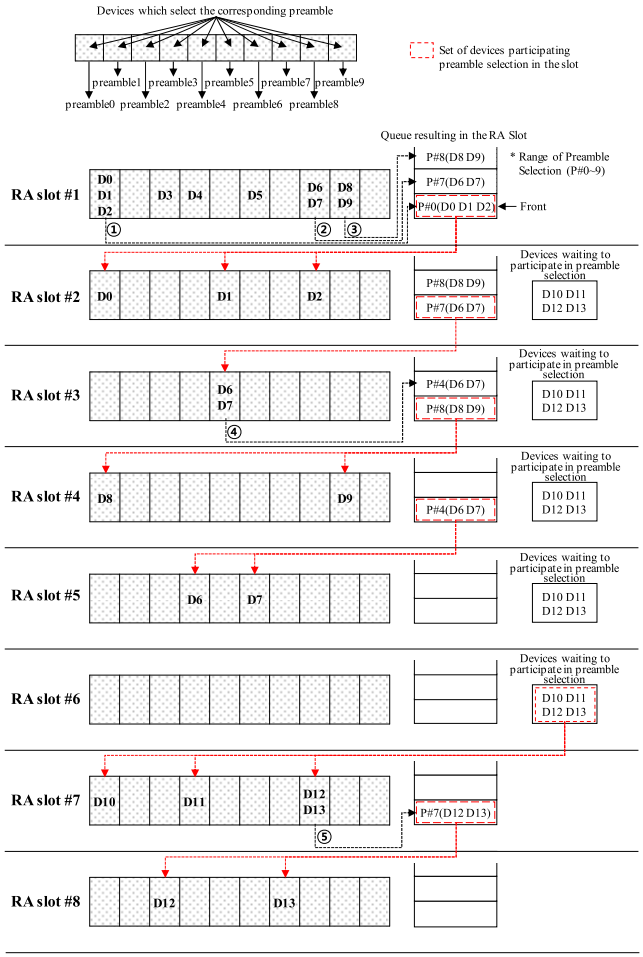

FIGURE 3. Detailed descriptions of Laya *et al.*'s scheme [1], [2].

Fig. 3 illustrates Laya *et al.*'s scheme along the time domain (in each RA slot). We assume that the total number of preambles is 10, the interval between RA slots is 10 ms, and the average time from the attempt to select a preamble to the corresponding RAR reception is 6 ms [22]. Here, the 10 concatenated boxes in each RA slot denote the 10 preambles, and D0–D13 in the boxes denote the devices that have selected the corresponding preamble. We also assume that devices D0–D9 arrive in RA slot #1 and devices D10–D13 arrive in RA slot #2.

In RA slot #1, 10 devices participate in preamble selection. The range for selecting preambles is P#0–P#9. D0, D1, and D2 choose the same preamble P#0; D6 and D7 choose P#7; and D8 and D9 choose P#8, i.e., collisions occur in these three preambles. The eNB informs the devices of the only preambles in which collision occurred. These preambles are put into a first-in, first-out (FIFO) queue for sequential resolution in ascending order by preamble number. In RA slot #1 of Fig. 3, collision occurred in P#0, P#7, and P#8, and the FIFO queue holds the preamble numbers in that order, with P#0 at the front of the queue. The devices that selected P#0, P#7, and P#8 in RA slot #1 will be allowed to select preambles in RA slots #2, #3, and #4, respectively. Since D3, D4, and D5 succeeded in preamble selection in RA slot #1 (no collision), the access delay for each device is estimated as 6 ms ($T_{wait}(i) = 0$, $T_{sel}(i) = 6\ ms$, for $i = 3, 4$, and 5).

In RA slot #2, the devices that selected P#0 in RA slot #1 participate in preamble selection. P#0 at the front of the FIFO queue is removed. Again, the range for selecting preambles is P#0–P#9. D0, D1, and D2 select preambles P#0, P#4, and P#7, respectively, with no collision. The access delay for each of devices D0, D1, and D2 is thus estimated as 16 ms ($T_{wait}(i) = 10$, $T_{sel}(i) = 6\ ms$, for $i = 0$, 1, and 2). Note that devices D10, D11, D12, and D13 arrive in RA slot #2, but they are not allowed to participate in preamble selection until all of the collisions in the FIFO queue are resolved.

In RA slot #3, devices D6 and D7 participate in the preamble selection. In this example, they both select the same preamble P#4, again resulting in collision. Then, P#4 (D6, D7) is added to the end of the FIFO queue for another resolution, as shown in the right of RA slot #3.

In RA slot #4, devices D8 and D9 which collided in P#8 in RA slot #1 participate in preamble selection. D8 and D9 select P#0 and P#8, respectively, for successful resolution. The access delay for D8 or D9 is estimated as 36 ms ($T_{wait}(i) = 30$, $T_{sel}(i) = 6\ ms$, for $i = 8, 9$).

In RA slot #5, the devices that selected P#4 in RA slot #3 participate in the preamble selection. D6 and D7 select P#3 and P#5, respectively, with no collision. The access delay for D6 or D7 is estimated as 46 ms ($T_{wait}(i) = 40$, $T_{sel}(i) = 6\ ms$, for $i = 6$, 7). In RA slot #6, no devices participate in the preamble selection. This is a free RA slot, during which new devices detect that the FIFO queue is empty and initiate preamble selection in the next RA slot, i.e., RA slot #7.

In RA slot #7, devices D10, D11, D12, and D13 participate in preamble selection after they recognize a free RA slot in slot #6. D10 selects P#0, and D11 selects P#3; thus, no collision occurs. D12 and D13 collide in P#7. Therefore, P#7 is

added to the FIFO queue. The access delay for D10 and D11 is 56 ms (T_{wait} $(i) = 50$, T_{sel} $(i) = 6$ ms, for $i = 10, 11$).

In RA slot #8, D12 selects P#2 and D13 selects P#6, with no collision. The access delay for D12 and D13 is 66 ms (T_{wait} $(i) = 60$, T_{sel} $(i) = 6ms$, for $i = 12, 13$).

In the above illustrative example of Laya's scheme, eight RA slots (RA slots #1 through #8) are used for 14 devices (D#0 through D#13) to succeed in preamble selection. The average access delay for the 14 devices is 33.9 ms, and the standard deviation of the access delays is 21.8 ms.

While we may call the above Laya *et al.*'s scheme "sequential," in that it resolves the collided preambles one by one per RA slot, Yoon's scheme [3] may be called "parallel," in that it resolves multiple collided preambles simultaneously in an RA slot. More specifically, when collisions occur in j preambles in RA slot t, k preambles out of the j preambles are grouped together to be resolved in each RA slot numbering from $t + 1$ to $t + \left\lceil \frac{j}{k} \right\rceil$.

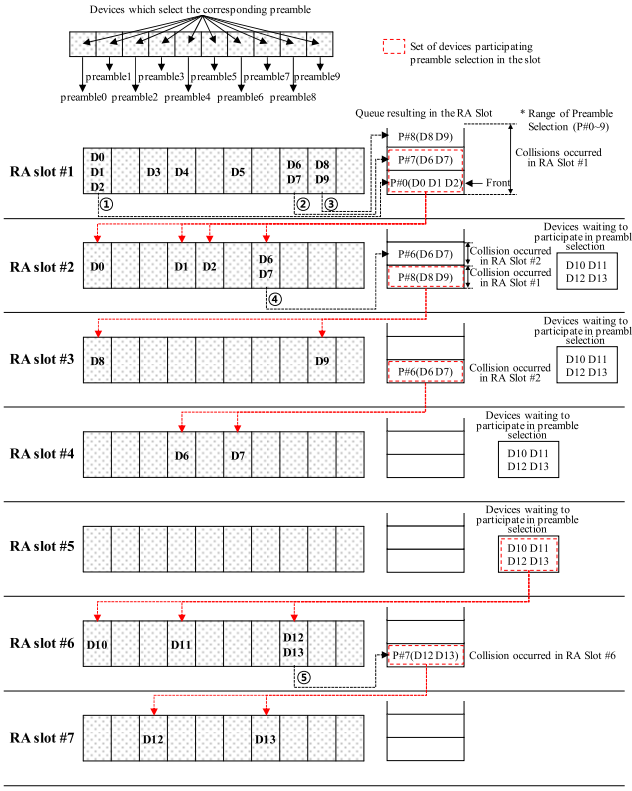

FIGURE 4. Detailed descriptions of Yoon's scheme [3] (grouping size: 2).

Fig. 4 illustrates Yoon's scheme with $k = 2$ under similar assumptions as in Fig. 3.

In RA slot #1, 10 devices (D0–D9) arrive to participate in preamble selection. The available preambles are P#0–P#9. D0, D1, and D2 choose P#0; D6 and D7 choose P#7; and D8 and D9 choose P#8. Thus, collisions occur in three preambles: P#0, P#7, and P#8. The three collided preambles are added to the FIFO queue for future resolution, with P#0 at the front. Since D3, D4, and D5 have succeeded in preamble selection in RA slot #1 (no collision), the access delay for each of these devices is estimated as 6 ms (T_{wait} $(i) = 0$, T_{sel} $(i) = 6ms$, for $i = 3, 4$, and 5).

Devices D0, D1, D2, D6, and D7, which collided in P#0 or P#7 in RA slot #1, are now allowed to select preambles in RA slot #2. Collisions in the two preambles are resolved together, resulting in a parallelism of two. (Note that in Fig. 3, only the devices that selected P#0 in RA slot #1 are allowed to select preambles in RA slots #2). Two preambles, P#0 and P#7, are removed from the front of the FIFO queue. Again, the range for selecting preambles is P#0–P#9. D0, D1, and D2 select preambles P#0, P#3, and P#4, respectively, with no collision, and both D6 and D7 select P#6, again colliding. Then, P#6 (D6, D7) is added to the end of the FIFO queue for another resolution, as shown to the right of RA slot #2. The access delay for devices D0, D1, and D2 is thus estimated as 16 ms each (T_{wait} $(i) = 10$, T_{sel} $(i) = 6$ ms, for $i = 0$, 1, and 2). Note that devices D10, D11, D12, and D13 arrive in RA slot #2, but they are not allowed to participate in preamble selection until all collisions in the FIFO queue have been resolved.

Now, devices D8 and D9—which collided in P#8 in RA slot #1—will be allowed to select preambles in RA slot #3. P#8 at the front of the FIFO queue is removed. D8 selects P#0, and D9 selects P#8, for successful resolution. The access delay for D8 or D9 is estimated as 26 ms (T_{wait} $(i) = 20$, T_{sel} $(i) = 6$ ms, for $i = 8, 9$).

In RA slot #4, P#6 (D6, D7) is removed from the front of the FIFO queue for resolution. D6 selects P#3, and D7 selects P#5, with no collision. The access delay for D6 and D7 is estimated as 36 ms (T_{wait} $(i) = 30$, T_{sel} $(i) = 6$ ms, for $i = 6, 7$). Note that all of the collisions in the FIFO queue are now resolved. In RA slot #5, no device participates in preamble selection, since devices initiate preamble selection only after they detect that the FIFO queue is empty. This is a free RA slot, during which new devices detect the empty FIFO queue and initiate preamble selection in the next RA slot, i.e., RA slot #6.

In RA slot #6, devices D10, D11, D12, and D13 participate in preamble selection. D10 and D11 select P#0 and P#3, respectively. D12 and D13 collide in P#7. Therefore, P#7 is added to the FIFO queue. The access delay for D10 and D11 is estimated as 46 ms (T_{wait} $(i) = 40$, T_{sel} $(i) = 6$ ms, for $i = 10, 11$).

In RA slot #7, devices D12 and D13 select P#2 and P#6, respectively, to succeed in resolution. The access delay for D12 and D13 is 56 ms (T_{wait} $(i) = 50$, T_{sel} $(i) = 6$ ms, for $i = 12, 13$).

In the above illustrative example of Yoon's scheme, seven RA slots (RA slots #1 through #7) are used until all 14 devices (D#0–D#13) succeed in preamble selection. The average access delay for the 14 devices is 28.1 ms, and the standard deviation of the access delays is 17.4 ms. Compared with Laya *et al.*'s scheme illustrated in Fig. 3, the average access delay is reduced by 5.8 ms (17%), and the standard deviation is reduced by 4.4 (20%).

V. THE PROPOSED SCHEME

A. DESCRIPTIONS OF THE PROPOSED SCHEME

We may call our scheme "parallel and pipelined" to simplify its differences from Laya *et al.*'s "sequential" scheme and Yoon's "parallel" scheme. The proposed scheme divides the available preambles into k subgroups, and preamble selection in the k subgroups is performed simultaneously in parallel ("parallel"). For the initial selection, devices participate in preamble selection with all of the available preambles. If there is any collision in a subgroup, resolution is performed within the subgroup, using only the preambles belonging to it. Subgroups with no collision may be used in preamble selection for newly arriving devices, thus reducing overall access latency due to pipelining effects ("pipelined").

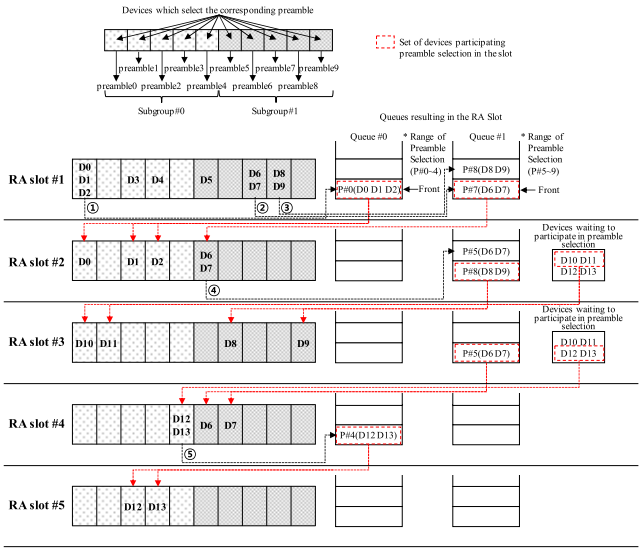

FIGURE 5. Detailed descriptions of the proposed scheme (No. subgroups: 2).

Fig. 5 describes an illustrative example of the proposed scheme with $k = 2$, under similar assumptions as in Figs. 3 and 4. In the figure, 10 preambles (P#0–P#9) are divided into two subgroups: P#0–P#4 for subgroup #0 and P#5–P#9 for subgroup #1. Queues #0 and #1 represent the FIFO queues for preamble subgroups #0 and #1, respectively.

In RA slot #1, 10 devices (D0–D9) participate in preamble selection. For this initial selection, all 10 preambles (P#0–P#9) are available for the 10 devices. D0, D1, and D2 choose preamble P#0; D6 and D7 choose P#7; and D8 and D9 choose P#8, resulting in collisions in three preambles. The eNB informs the devices of whether collision pending resolution exists for each subgroup and the preambles in which collision occurred. Each collided preamble is added to the FIFO queue for the subgroup to which it belongs, with P#0 to Queue #0 and P#7 and P#8 to Queue #1, with P#7 at the front. Since D3, D4, and D5 succeeded in preamble selection in RA slot #1 (no collision), the access delay for each of these devices is estimated as 6 ms ($T_{wait}\ (i) = 0$, $T_{sel}\ (i) = 6ms$, for $i = 3$, 4, and 5).

In RA slot #2, P#0 is removed from Queue #0, and P#7 is removed from Queue #1. The devices (D0, D1, D2) that selected P#0 in RA slot #1 are allowed to select among the five preambles (P#0, P#1, P#2, P#3, and P#4) that belong to subgroup #0.

In the same RA slot #2, the devices (D6, D7) that selected P#7 in RA slot #1 are allowed to select among the five preambles (P#5, P#6, P#7, P#8, and P#9) that belong to subgroup #1. D0, D1, and D2 select preambles P#0, P#2, and P#3, respectively, with no collision, and D6 and D7 both select P#5, again resulting in collision. P#5 is added to Queue #1.

We assume D10, D11, D12, and D13 arrive in RA slot #2. The eNB informs the devices that Queue #0 is empty. Therefore, new devices recognize that they can participate in preamble selection within subgroup #0 in the next RA slot. With the given probability of 0.5, the newly arriving devices are allowed to participate in preamble selection among the five preambles (P#0–P#4) in the next RA slot (i.e., RA slot #5). In general, the probability is set to m/k, if we have k subgroups and m ($\leq k$) subgroups are available for newly arriving devices.

In RA slot #3, P#7 is removed from Queue #1, and the devices (D8, D9) that selected P#8 in RA slot #1 will be allowed to select preambles among the five preambles (P#5, P#6, P#7, P#8, and P#9) that belong to subgroup #1. D8 and D9 select P#6 and P#9, respectively, with no collision. In the same RA slot #3, newly arrived devices (D10, D11, D12, and D13) attempt to select a preamble belonging to subgroup #0 with a probability of 1/2. We further assume that D10 and D11 actually attempted to select, while D12 and D13 did not (i.e., half of the arrived devices attempted to). D10 and D11 select P#, and P#1, respectively, with no collision. It shows pipelining advantages in that newly arriving devices participate in preamble selection within subgroup #0 while resolution is still being performed in subgroup #1.

The access delay for D8 and D9 is estimated as 26 ms ($T_{wait}\ (i) = 20$, $T_{sel}\ (i) = 6\ ms$, for $i = 8$, 9); for D10 and D11, it is estimated as 16 ms ($T_{wait}\ (i) = 10$, $T_{sel}\ (i) = 6\ ms$, for $i = 10$, 11). Now, Queue #0 is empty and Queue #1 holds P#5 (D6, D7).

In RA slot #4, devices D12 and D13 select preambles among P#0–P#4, which belong to subgroup #0. D12 and D13 collide in P#4. P#4 is added to Queue #0. P#5 (D6, D7) is removed from Queue #5, and D6 and D7 select P#5 and P#6, respectively, for successful resolution. The access delay for D6 or D7 is 36 ms ($T_{wait}\ (i) = 30$, $T_{sel}\ (i) = 6\ ms$, for $i = 6$, 7). The eNB informs the devices that Queue #1 is empty.

In RA slot #5, P#4 (D12, D13) is removed from Queue #0. D12 and D13 select P#2 and P#3, respectively, for successful resolution. The access delay for D12 and D13 is 36 ms ($T_{wait}\ (i) = 30$, $T_{sel}\ (i) = 6\ ms$, for $i = 12$, 13). Now, all 14 devices will have finished preamble selection.

In the above illustrative example of the proposed scheme, five RA slots (RA slots #1 through #5) are used for all

Algorithm 1 Basic Proposed Algorithm

$Q[s,t]$.**add**(element): add 'element' to the rear of queue for subgroup s in RA slot t
$Q[s,t]$.**remove**(): return the 'element' from the front of queue for subgroup s in RA slot t
$Q[s,t]$.**isEmpty**(): return Boolean for whether the queue for subgroup s is empty or not in RA slot t
1: $Q[s,t]$ <- new Queue() // initiate queues for each subgroup
2: **while** (until preamble selection is completed for all of the devices)
3: $D_t[p,t]$ <- **PREAMBLE_SELECTION**($Q[s,t]$)
4: $Q[s,t]$ <- **COLLISION_DETECTION**($D_t[p,t]$)
5: $t = t+1$
6: **end while**

Input: Queue for subgroups$s$in RA slot t, $Q[s,t]$
Output: A device or a set of devices that transmit preamble p in RA slot t, $D_t[p,t]$
PREAMBLE_SELECTION ($Q[s,t]$)
1: $D_w[t]$ <- $D_w[t-1] + D_n[t]$
2: **for** $s = 0$ to N_g-1, **do in parallel** ▷ Parallel execution at each subgroup
3: **if** ($Q[s,t]$.**isEmpty**() == 1) ▷ Check if subgroup s is free
4: **for** all devices $D_w[t]$, **do**
5: **if** (**randi**$(1, N_g)$ <= 1) ▷ The probability (1/no. Of subgroups)
6: select a preamble within subgroup s, **randi**$(s*N_p/N_g, (s+1)*N_p/N_g-1)$ ▷ Preamble selection within subgroup s
7: $D_t[p,t]$) <- add device that selects a preamble p
8: remove device that selects a preamble from $D_w[t]$ ▷ Exclude from the waiting device list
9: **end if**
10: **end for**
11: **else**
12: $D_q[s,t]$ <- $Q[s,t]$.**remove**() ▷ Remove devices from the queue for subgroup s
13: **for** all devices $D_q[s,t]$, **do**
14: preamble selection within subgroup s, **randi**$(s*N_p/N_g, (s+1)*N_p/N_g-1)$ ▷ Preamble selection within subgroup s
15: **end for**
16: **end if**
17: **end for**
18: **return** $D_t[p,t]$

Input: A device or a set of devices that transmit preamble p in RA slot t, $D_t[p,t]$
Output: Queue for subgroups$s$in RA slot t, $Q[s,t]$
COLLISION_DETECTION ($D_t[p,t]$)
1: **for** $s = 0$ to N_g-1,**do in parallel** ▷ Parallel execution at each subgroup
2: **for** $p = s*N_p/N_g$ to $(s+1)*N_p/N_g+1$, **do** ▷ For preambles within subgroup s
3: **if** eNB cannot decode preamble p ▷ Collision is detected in preamble p
4: $S_p[p,t] = 1$; ▷ update state of preamble p
5: $Q[s,t]$.**add**($D_t[p,t]$) ▷ add colliding devices to the queue for subgroup s
6: **else**
7: $S_p[p,t] = 0$; ▷ update state of preamble p
8: **end if**
9: **end for**
10: **end for**
11: **return** $Q[s,t]$

14 devices (D#0 through D#13) to finish preamble selection. The average access delay for the 14 devices is 21 ms, and the standard deviation of the access delays is 11.2 ms. Compared with Laya *et al.*'s scheme and Yoon's scheme for the same illustrative example as in Figs. 3 and 4, the average access delay for the proposed scheme is reduced by 12.9 ms (38%) and 7.1 ms (25%), respectively. The standard deviation is reduced by 10.6 (49%) and 6.2 (36%), respectively. Note that the improvement is limited because the examples are simplified with 10 preambles; the previous schemes may cause performance degradation under a large number of preambles.

We think the performance improvement is the result of our "parallel and pipelined" approach against Laya *et al.*'s "sequential" scheme and Yoon's "parallel" scheme. Our "pipelined" approach allows us to serve newly arriving devices in empty subgroups while resolving previous collisions in the FIFO queue using corresponding subgroups.

B. A PSEUDOCODE FOR THE PROPOSED SCHEME

A pseudocode for our scheme is provided in Algorithm 1. Refer to Table 2 for the notations used in the pseudocode. The pseudocode can be divided into two parts: preamble selection and detection of preamble collision.

TABLE 2. Notations for the pseudocode in algorithm 1.

Notation /Variables	Descriptions
p	Preamble ID (p = 0,1,2,…,N_p-1)
s	Preamble subgroup number (s = 0,1,2,…, N_g-1)
t	RA slot (t = 1,2,3,…)
N_g	Number of preamble subgroups
N_p	Number of non-dedicated preambles (N_p is divisible by N_g)
$D_t[p, t]$	A device or a set of devices that transmit preamble p in RA slot t
$D_q[s, t]$	A device or a set of devices that exist at the front of the queue for subgroup s in RA slot t
$D_n[t]$	A device or a set of devices newly participating in preamble selection in RA slot t
$D_w[t]$	Newly arriving device(s) waiting to participate in preamble selection in RA slot t
$Q[s, t]$	Queue for preamble subgroup s in RA slot t
$S_p[p, t]$	State of collision for preamble p in RA slot t 0: No collision in preamble p 1: Existing collision in preamble p

1) PREAMBLE SELECTION

We first check for any collisions pending resolution for each subgroup. If any collisions are pending resolution for subgroups, the devices at the front of the queue for the subgroup select preambles between $s \times \frac{N_p}{N_g}$ and $(s+1) \times \frac{N_p}{N_g} - 1$. If a free subgroup s exists, newly arriving devices waiting to participate in preamble selection can select a certain preamble within subgroup s with a probability $\frac{1}{N_g}$, and other devices that cannot participate in the preamble selection wait for the next RA slot, $t+1$. This procedure is executed in parallel for each subgroup.

2) DETECTION OF PREAMBLE COLLISION

Next, we identify preamble collisions in each subgroup. If eNB cannot decode preamble p in RA slot t, a collision occurs. Then, devices are added to the queue for subgroup $\left\lfloor \frac{p \times N_g}{N_p} \right\rfloor$. This procedure is executed in parallel for each subgroup.

3) PROPOSED EXTENSION OF THE FEEDBACK INFORMATION IN RAR

We extend RAR, which is used to acknowledge the RA preamble, to convey additional feedback information to incorporate the proposed scheme into the conventional RA procedure in LTE-A [9]. The RAR is a MAC protocol data unit (PDU) consisting of a MAC header and zero or more MAC RAR.

The MAC header is of variable size and consists of one or more MAC subheaders. MAC subheaders come in two types: a random access preamble identifier (RAPID) subheader and a backoff indicator (BI) subheader. The RAPID subheader includes the preambles that the eNB succeeds in receiving, and the BI subheader contains backoff time information for failed devices.

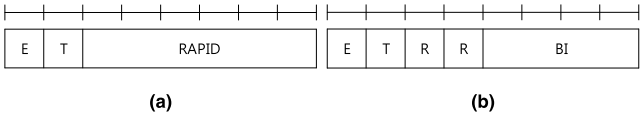

FIGURE 6. MAC subheaders. (a) RAPID MAC subheader (b) BI MAC subheader.

The following fields are shown in Fig. 6:

- E: Extension field that indicates if more subheaders are present or not
- T: Type field that indicates if the subheader is RAPID ("1") or BI ("0")
- RAPID: Random access preamble identifier field
- R: Reserved bit, set to "0"
- BI: Backoff value

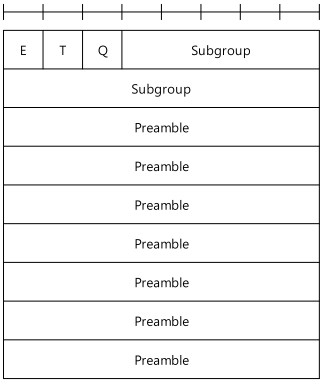

FIGURE 7. Preamble contention resolution queue (PCRQ) subheader.

To provide additional feedback information for the proposed scheme, we propose a new type of MAC subheader called the preamble contention resolution queue (PCRQ) subheader, as shown in Fig. 7. In the PCRQ subheader, we set the T field to "0" and the following third bit called the Q field to "1," to distinguish it from the two other RAPID and BI subheaders.

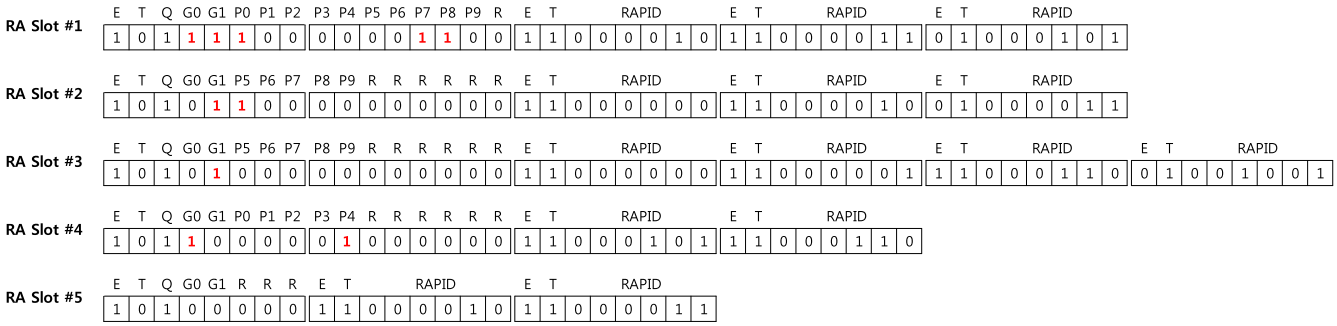

FIGURE 8. Example of the content of the PCRQ subheaders and the RAPID subheaders for the RAR, following the behavior in Fig. 5 for RA slots #1–5. MAC subheaders.

The PCRQ subheader consists of a subgroup field and a preamble field. The subgroup field is a bitmap indicating whether the subgroup has any collisions pending resolution. The first bit corresponds to subgroup #0. The value 0 in the bitmap indicates that the corresponding subgroup is free, while the value 1 in the bitmap indicates that the corresponding subgroup has collisions pending resolution. Based on this field, newly arriving devices recognize whether they can participate in preamble selection and which preambles they can use in the next RA slot.

The preamble field is a bitmap identifying whether a collision exists in each preamble in the previous RA slot. A value of 0 in the bitmap indicates that there is no collision or no detection in the corresponding preamble, while a value of 1 in the bitmap indicates that a collision exists in the corresponding preamble in the previous RA slot. Unlike the subgroup field, the preamble field only consists of preambles belonging to the subgroup in which the corresponding bit value is 1. If the bit value in the subgroup field is "0," then preambles in the corresponding subgroup will not be presented in the PCRQ subheader because no collision occurred in the preambles in the previous RA slot. Therefore, the length of the preamble field can vary from 0 to the total number of preambles, depending on the states of the subgroups.

Unlike in the LTE-A RA procedure, not only the devices that transmitted a preamble in the previous RA slot but also the devices that are willing to newly attempt RA in the next RA slot should receive the RAR. According to the subgroup and preamble field information in the PCRQ subheader, devices that transmitted a preamble in the previous RA slot can identify whether their transmissions succeeded, and newly arriving devices can find a free subgroup to attempt RA in the next RA slot. If they cannot find a free subgroup, they will continue to receive RARs and check the subgroup field in the RAR until they find one.

Fig. 8 shows example content of the PCRQ subheaders and the RAPID subheaders following the behaviors described in Fig. 5 for the five RA slots. Here, G0 and G1 denote subgroup #0 and #1, and P0–P9 denote preambles #0 to #9, respectively. For RA slot #1, the feedback for subgroups #0 and #1 on the PCRQ subheader is G0 = G1 = 1, which indicates a contention resolution state for both subgroups because preamble P#0 in subgroup #0 and preambles P#7 and P#8 in subgroup #1 have collided. The feedback for P#0, P#7, and P#8 is P0 = P7 = P8 = 1, which indicates a preamble collision in this RA slot, and the associated RAPID subheaders indicate a success for preambles P#2, P#3, and P#5. For RA slot #2, the feedback for subgroup #0 on the PCRQ subheader is G0 = 0, which indicates a free subgroup #0, as all of the devices in the queue of subgroup #0 succeed, and the queue becomes empty. The feedback for subgroup #1 is G1 = 1, which indicates a contention resolution state for that subgroup because devices remain in its queue. Moreover, the feedback for preamble P#5 is P5 = 1, which indicates a collision, and the associated RAPID subheaders indicate success for P#0, P#2, and P#3. Here, the feedback for the preambles (P0–P4) in subgroup #0 is not presented in the PCRQ subheader because the state of subgroup #0 becomes free after all of the devices that selected P#0 succeed with different preambles and leave the subgroup's queue. For RA slot #3, the feedback for subgroup #1 is G1 = 1, which indicates a contention resolution state for that subgroup because devices remain in its queue. For RA slot #4, the feedback for subgroup #0 is G0 = 0, which indicates a contention resolution state for that subgroup because of a collision in P#4. The feedback for subgroup #1 is G1 = 0, which indicates that the subgroup is free because all of the devices in its queue have succeeded and the queue has become empty. For RA slot #5, the feedback for subgroups #0 and #1 on the PCRQ subheader is G0 = G1 = 0, which indicates that the subgroups are free. The feedback for all of the preambles (P0–P9) in subgroups #0 and #1 is not presented in the PCRQ subheader because the state of subgroups #0 and #1 has become free.

VI. PERFORMANCE EVALUATION

A. SYSTEM MODEL

We considered an LTE-A network operating in frequency division duplex (FDD) mode, in which a single eNB serves a large number of IoT devices. We assumed that all of the devices had already received all configuration parameters related to the contention-based RA procedure in the system information block (SIB) from the eNB. We assigned

56 non-dedicated RA preambles and 8 dedicated RA preambles out of 64 RA preambles in an LTE-A network [5]. We assumed that if two or more devices transmitted the same preamble, a collision would occur and the eNB would not transmit the RAR for the preamble. During our simulation, the number of IoT devices was increased from 1,000 to 10,000 in steps of 1,000. The arrival of devices followed a beta distribution with $\alpha = 3$, $\beta = 4$ [4], [11], [20], [21]. The parameters used in our simulations are summarized in Table 3 [1], [2], [4], [22].

TABLE 3. Simulation parameters.

Parameter	Value
PRACH configuration index	3
Total number of non-dedicated RA preambles	56
Maximum number of preamble retransmissions	20
RAR window size	5 ms
Backoff indicator	480 ms
Barring factor	80%
Barring time	4 s
Number of IoT devices	1,000–10,000
Arrival distribution	Beta distribution over 1 s ($\alpha = 3$, $\beta = 4$)

The following metrics are used to compare the performance of the proposed scheme against three previously known schemes: the 3GPP standard RA procedure with ACB [10], [11] as a baseline, Laya *et al.*'s scheme [1], [2], and Yoon's scheme [3].

1) **Average access delay:** The average time elapsed between the device's arrival and RAR reception, as defined in Equation (5). The device arrival indicates the time when the device arrives, and the RAR reception indicates the time when the device receives RAR, including the device's own RA preamble ID in the RAPID subheader. Only successful accesses are considered for the average calculation.
2) **Standard deviation of access delays:** The standard deviation of access delays among the devices that succeed in preamble selection, as defined in Equation (6).
3) **Blocking probability:** The probability that each device will reach the maximum number of attempts without successful access, as defined in Equation (7).
4) **Underutilization rate of preamble resources:** The proportion of RA slots in which the number of devices participating in preamble selection is lower than the specific values, which are below the total number of RA preambles among the total RA slots. It seems like a waste of preamble resources that only a few devices participate in preamble selection in an RA slot when the number of preambles is large.
5) **Number of devices that succeed in preamble selection:** The number of devices that succeed in preamble selection in each RA slot.

B. SIMULATION RESULTS

The simulation was performed in MATLAB to compare the proposed scheme with three previous schemes: the 3GPP standard RA procedure with ACB [10], [11] serving as a baseline, Laya *et al.*'s scheme [1], [2], and Yoon's scheme [3]. The results were obtained by averaging the 100 simulations.

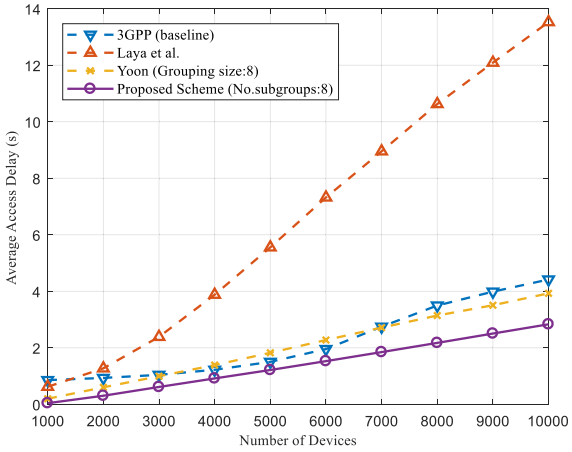

FIGURE 9. Average access delay vs. number of devices.

Fig. 9 compares the average access delay of the proposed scheme against those of the three previous schemes as the number of devices increased from 1,000 to 10,000. Laya *et al.*'s scheme showed the steepest increase in the average access delay. Since it resolves collisions one by one sequentially, the access delay may increase greatly, as the scheme is slow to respond to increased collisions due to more arriving devices. For example, when collisions occur in k out of 56 preambles in RA slot t—say, P_0, P_1, ..., and P_{k-1}—P_0 is resolved in RA slot $t + 1$, P_1 is resolved in RA slot $t + 2$, ..., and P_{k-1} is resolved in RA slot $t + k$, in a serial manner. It may take a long time to resolve all of the collisions when the number of devices increases. On the other hand, Yoon's scheme and the proposed scheme greatly reduce the average access delay compared with Laya *et al.*'s scheme because they resolve the preambles in a parallel manner. However, the proposed scheme exceeds Yoon's scheme by up to 27.89% because the proposed scheme has pipelining advantages, in that newly arriving devices are allowed to start preamble selection in any free subgroup.

Fig. 10 compares the average access delays between the proposed scheme and Yoon's scheme. We varied the number of subgroups in the proposed scheme and the grouping size in Yoon's scheme as 4, 8, and 14. The solid lines indicate the results of the proposed scheme, and the broken lines indicate the results of Yoon's scheme. The proposed scheme exceeds Yoon's scheme for all numbers of subgroups. For example, in the case of 10,000 devices, the proposed scheme exhibits

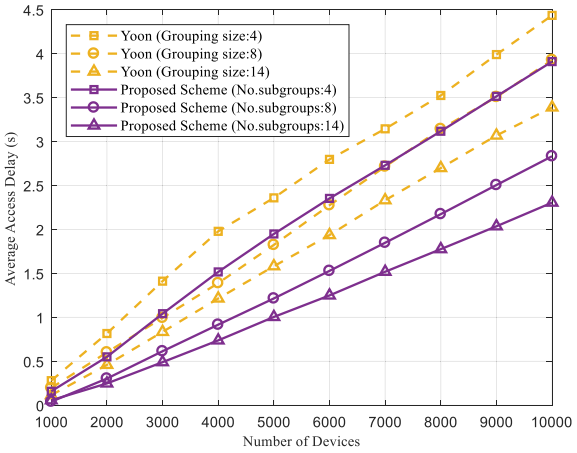

FIGURE 10. Average access delay comparison by grouping size in Yoon's scheme and the number of subgroups in the proposed scheme.

an average access delay that is 31.96% lower than Yoon's scheme when the number of subgroups and the grouping size are 14. Both Yoon's scheme and the proposed scheme have advantages for parallel processing in that each k preambles can be resolved in an RA slot. However, in Yoon's scheme, there are some cases in which fewer than k preambles are resolved when the number of devices that participate in preamble selection simultaneously is small. Therefore, this illustrates how the proposed scheme has advantages due to pipelined processing over Yoon's scheme, in that newly arriving devices can participate in preamble selection in any free subgroup while collisions for previously arrived devices are being resolved in the other subgroups.

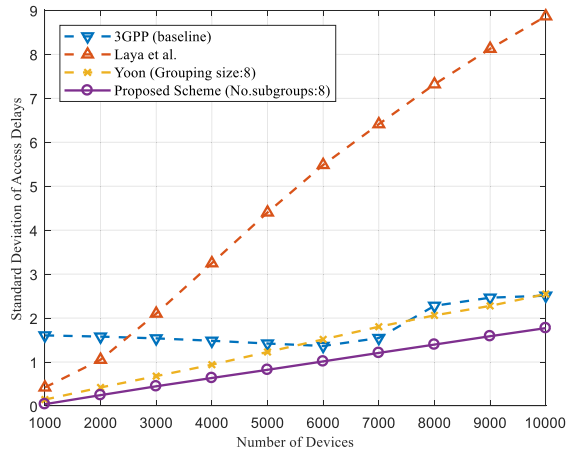

FIGURE 11. Standard deviation of access delays.

Fig. 11 compares the standard deviation of access delays for the proposed scheme against those of the previous schemes. A large standard deviation means that some devices experience much longer access delays than others. In the case of Laya *et al.*'s scheme, the standard deviation of access delays increases sharply as the number of devices increases. The proposed scheme shows lower standard deviation compared to the three previous schemes. For example, in the case of 10,000 devices, the proposed scheme reduces the standard deviation of access delays by 29.40%, 80.02%, and 30.44% compared to 3GPP standard access with ACB, Laya *et al.*'s scheme, and Yoon's scheme, respectively.

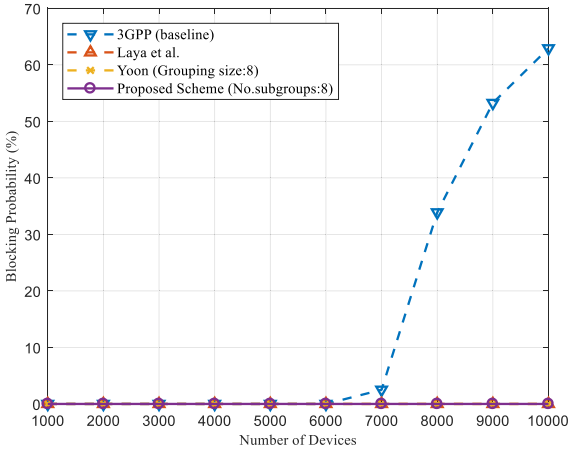

FIGURE 12. Blocking probability (probability of a device reaching the maximum number of attempts for preamble selection and being unable to succeed in the preamble selection).

Fig. 12 compares the blocking probability of the proposed scheme against those of the previous schemes as the number of devices increases from 1,000 to 10,000. The blocking probability of the 3GPP standard increases sharply after the number of devices increases past 6,000. This could be mitigated by increasing the backoff time, increasing the barring time, or decreasing the barring rate. However, these changes may result in even higher access delays.

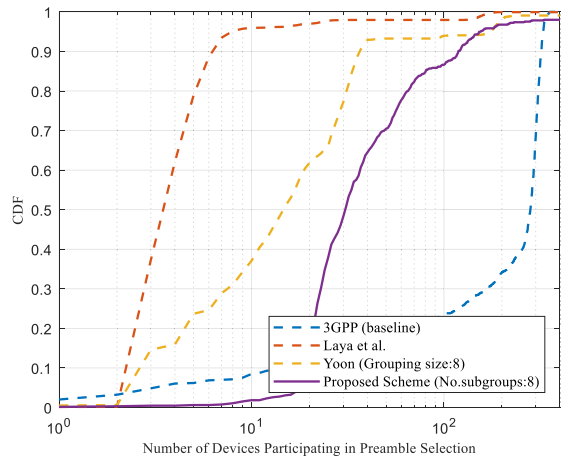

FIGURE 13. CDF of the number of devices participating in preamble selection in each RA slot.

Fig. 13 and Table 4 compare the proposed scheme against Laya *et al.*'s scheme and Yoon's scheme in terms of the number of devices participating in preamble selection in each RA slot in the case of 10,000 devices. These show how evenly the devices participate in preamble selection among

TABLE 4. Comparison of the underutilization rate of preamble resources (%) among the three schemes.

	Laya *et al.* [1] [2]	Yoon [3] (grouping size = 8)	Proposed scheme (No. subgroups = 8)
N_d^* < 56 (N_p^{**})	2947/3005 (98.07%)	906/1016 (89.17%)	486/650 (74.77%)
N_d^* < 28 (N_p^{**}/2)	2890/3005 (96.17%)	850/1016 (83.66%)	287/650 (44.15%)
N_d^* < 14 (N_p^{**}/4)	2889/3005 (96.14%)	535/1016 (52.66%)	18/650 (2.77%)
N_d^* < 7 (N_p^{**}/8)	2849/3005 (94.81%)	323/1016 (31.79%)	5/650 (0.77%)
N_d^* < 4 (N_p^{**}/14)	2164/3005 (72.01%)	165/1016 (16.24%)	3/650 (0.46%)

* N_d: Number of devices participating in preamble selection in an RA slot
** N_p: Total number of RA preambles = 56

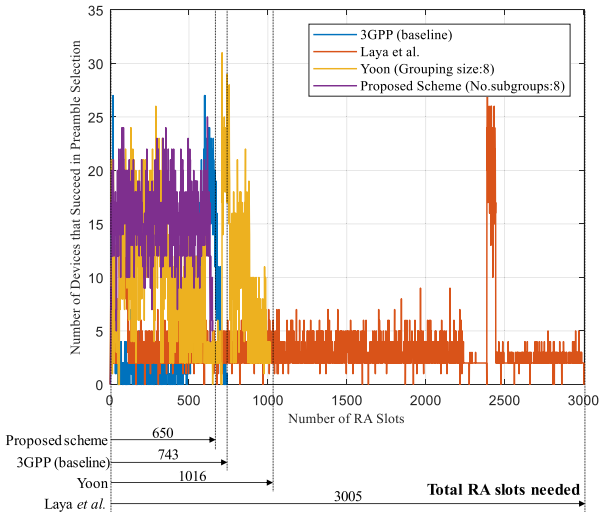

FIGURE 14. Number of devices that succeed in preamble selection in each RA slot.

RA slots. An even distribution implies higher utilization of RA slots. A lower number of devices in some RA slots may waste resources and require more RA slots. Fig. 13 shows the cumulative distribution function (CDF) of the number of devices participating in preamble selection in each RA slot. In Laya *et al.*'s scheme, the case in which 2–7 devices participate in the preamble selection in an RA slot accounts for more than 90% of the total RA slots. Hence, the total number of RA slots needed to resolve all of the collisions increases because only a few (2–7) devices participate in preamble selection in most of the RA slots. On the other hand, the proposed scheme greatly reduces the number of RA slots in comparison with the two other schemes (Laya *et al.*'s scheme and Yoon's scheme). For example, in the proposed scheme, the RA slots in which fewer than 10 devices participate in preamble selection account for less than 3% of the total RA slots. The RA slots in which 10–50 devices participate in preamble selection account for about 70% of the total RA slots. In Laya *et al.*'s scheme, preamble collision rarely occurs, but it takes a longer time for all of the devices to complete preamble selection than in the proposed scheme because only a few devices participate in preamble selection in many RA slots, wasting resources.

TABLE 5. Average number of devices that succeed in preamble selection in a RA slot

	3GPP (baseline) [4], [5]	Laya *et al.* [1] [2]	Yoon [3] (grouping size = 8)	Proposed scheme (No. subgroups = 8)
Average number of devices that succeed in preamble selection	5.06 devices/RA slot (151.95%)	3.33 devices/RA slot (100%)	9.84 devices/RA slot (295.50%)	15.38 devices/RA slot (461.86%)

Fig. 14 compares the number of devices that succeed in preamble selection in each RA slot, and Table 5 summarizes the average number of devices that succeed in preamble selection in an RA slot. The proposed scheme shows the largest average number of devices that succeed in preamble selection in an RA slot compared with the other schemes (3GPP standard, Laya *et al.*'s scheme, and Yoon's scheme). For example, in the proposed scheme, the average number of devices that succeed in preamble selection in each RA exceeds Laya *et al.*'s scheme by as much as 361.86%. Consequently, the proposed scheme consumes the smallest number of RA slots to complete preamble selection for all of the devices, compared to the other schemes, as shown in Fig. 14.

VII. CONCLUSION

This paper addressed the shortage of RA preamble resources in LTE-A due to the RA attempts of a massive number of IoT devices. To solve the problem, we have proposed a novel pipelined contention resolution scheme based on DQ for a massive number of IoT devices in RA in LTE-A networks. The proposed scheme divides an entire preamble set into k subgroups, and contention resolutions in individual subgroups are performed in parallel. The simulation results demonstrate the advantages of our scheme. It can greatly reduce the average access delay and the standard deviation while keeping the blocking probability to a minimum. Moreover, utilization of preamble resources is greatly improved by reducing the number of random access (RA) slots, where only a few devices participate in preamble selection.

In future work, we plan to extend our scheme with quality of service (QoS) guarantees in RA for IoT devices. We also plan to study dynamic RA resource allocation based on congestion level.

REFERENCES

[1] A. Laya, L. Alonso, and J. Alonso-Zarate, "Efficient contention resolution in highly dense LTE networks for machine type communications," in *Proc. IEEE GLOBECOM*, San Diego, CA, USA, Dec. 2015, pp. 1–7.

[2] A. Laya, L. Alonso, and J. Alonso-Zarate, "Contention resolution queues for massive machine type communications in LTE," in *Proc. IEEE 26th Annual Int. Symp. Pers., Indoor, Mobile Radio Commun. (PIMRC)*, Aug./Sep. 2015, pp. 2314–2318.
[3] C. Yoon, "Distributed queuing with preamble grouping for massive IoT devices in LTE random access," in *Proc. Int. Conf. Inf. Commun. Technol. Converg. (ICTC)*, Oct. 2016, pp. 103–105.
[4] M.-Y. Cheng, G.-Y. Lin, H.-Y. Wei, and A. C.-C. Hsu, "Overload control for machine-type-communications in LTE-advanced system," *IEEE Commun. Mag.*, vol. 50, no. 4, pp. 38–45, Jun. 2012.
[5] A. Ksentini, Y. Hadjadj-Aoul, and T. Taleb, "Cellular-based machine-to-machine: Overload control," *IEEE Netw.*, vol. 26, no. 6, pp. 54–60, Nov./Dec. 2012.
[6] A. G. Gotsis, A. S. Lioumpas, and A. Alexiou, "M2M scheduling over LTE: Challenges and new perspectives," *IEEE Veh. Technol. Mag.*, vol. 7, no. 3, pp. 34–39, Sep. 2012.
[7] T.-M. Lin, C.-H. Lee, J.-P. Cheng, and W.-T. Chen, "PRADA: Prioritized random access with dynamic access barring for MTC in 3GPP LTE-A networks," *IEEE Trans. Veh. Technol.*, vol. 63, no. 5, pp. 2467–2472, Jun. 2014.
[8] C. A. Astudillo, T. P. C. de Andrade, and N. L. S. da Fonseca, "Allocation of control resources with preamble priority awareness for human and machine type communications in LTE-Advanced networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2017, pp. 1–6.
[9] *Evolved Universal Terrestrial Radio Access (E-UTRA); Medium Access Control (MAC) Protocol Specification*, document 3GPP TS 36.321 version 10.5.0, Mar. 2017.
[10] *Evolved Universal Terrestrial Radio Access (E-UTRA); Radio Resource Control (RRC); Protocol Specification*, document 3GPP TS 136 331-V8.6.0, Dec. 2016.
[11] *Study on RAN Improvements for Machine-Type Communications*, document 3GPP TR 37.868, Sep. 2011.
[12] C. M. Chou, C. Y. Huang, and C.-Y. Chiu, "Loading prediction and barring controls for machine type communication," in *Proc. IEEE ICC*, Jun. 2013, pp. 5168–5172.
[13] M. Tavana, V. Shah-Mansouri, and V. W. S. Wong, "Congestion control for bursty M2M traffic in LTE networks," in *Proc. IEEE ICC*, Jun. 2015, pp. 5815–5820.
[14] Z. Wang and V. W. S. 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.
[15] G. Foddis, R. G. Garroppo, S. Giordano, G. Procissi, S. Roma, and S. Topazzi, "On RACH preambles separation between human and machine type communication," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kuala Lumpur, Malaysia, May 2016, pp. 1–6.
[16] X. Yang, A. Fapojuwo, and E. Egbogah, "Performance analysis and parameter optimization of random access backoff algorithm in LTE," in *Proc. IEEE Veh. Technol. Conf. (VTC Fall)*, Sep. 2012, pp. 1–5.
[17] W. Xu and G. Campbell, "A near perfect stable random access protocol for a broadcast channel," in *Proc. IEEE ICC*, vol. 1, Jun. 1992, pp. 370–374.
[18] W. Xu and G. Campbell, "A distributed queueing random access protocol for a broadcast channel," *Comput. Commun. Rev.*, vol. 23, no. 4, pp. 270–278, Oct. 1993.
[19] L. Alonso, R. Agusti, and O. Sallent, "A near-optimum MAC protocol based on the distributed queueing random access protocol (DQRAP) for a CDMA mobile communication system," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 9, pp. 1701–1718, Sep. 2000.
[20] G. C. Madueño, C. Stefanovic, and P. Popovski, "Reengineering GSM/GPRS towards a dedicated network for massive smart metering," in *Proc. IEEE Int. Conf. Smart Grid Commun. (SmartGridComm)*, Nov. 2014, pp. 338–343.
[21] X. Jian, X. Zeng, J. Huang, Y. Jia, and Y. Zhou, "Statistical description and analysis of the concurrent data transmission from massive MTC devices," *Int. J. Smart Home*, vol. 8, no. 4, pp. 139–150, 2014.
[22] *Feasibility Study for Further Advancements for E-UTRA (LTE-Advanced)*, document 3GPP TR 36.912, Dec. 2015.
[23] S. J. Yi, S. D. Chun, Y. D. Lee, S. J. Park, and S. H. Jung, *Radio Protocols for LTE and LTE-Advanced*. Singapore: Wiley, 2012, pp. 143–145.
[24] C.-W. Chang, Y.-H. Lin, Y. Ren, and J.-C. Chen, "Congestion control for machine-type communications in LTE-a networks," in *Proc. IEEE GLOBECOM*, Washington, DC, USA, Dec. 2016, pp. 1–6.

KYUNGJUN LEE received the B.S. and M.S. degrees in electronics engineering from Sogang University, Seoul, South Korea, in 2006 and 2009, respectively, where he is currently pursuing the Ph.D. degree in electronics engineering. From 2009 to 2012, he was a Research Engineer with LG Electronics, Seoul. He is currently a Senior Research Engineer with Korea Telecom, Seoul. He has been participating in the standardization of radio protocols for UMTS, LTE, LTE-Advanced, LTE-Advanced Pro, and 5G NR in 3GPP since 2009. His current research interests include radio interface architecture, radio resource control, mobility management, and network performance optimization in 3GPP LTE-Advanced Pro and 5G NR.

JU WOOK JANG received the B.S. degree in electronics engineering from Seoul National University, Seoul, South Korea, the M.S. degree in electrical engineering from the Korean Advanced Institute of Science and Technology, and the Ph.D. degree in electrical engineering from the University of Southern California, California, CA, USA. From 1985 to 1988 and from 1993 to 1994, he was with Samsung Electronics, Suwon, South Korea, where he was involved in the development of a 1.5-Mb/s video codec and a parallel computer. Since 1995, he has been with Sogang University, Seoul, where he is currently a Professor. He has also built systems for videoconferencing, streaming, home networks, and ad hoc networks using protocols, such as RTP, SIP, Internet of Things, and Blockchain. His current research interests include WiMAX protocols, mobile networks, and next-generation networks. He received the LG Yonam Overseas Research Grant in 2001.