

Received January 12, 2021, accepted February 15, 2021, date of publication February 16, 2021, date of current version February 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3059950

Scalable Wi-Fi Backscatter Uplink Multiple Access for Battery-Free Internet of Things

JUNG-HYOK KWON¹, (Member, IEEE), XUE ZHANG²,
AND EUI-JIK KIM¹, (Senior Member, IEEE)

¹School of Software, Hallym University, Chuncheon 24252, South Korea

²College of Ocean Science and Engineering, Shandong University of Science and Technology, Qingdao 266590, China

Corresponding author: Eui-Jik Kim (ejkim32@hallym.ac.kr)


This research was supported by Hallym University Research Fund, 2020, (HRF-202004-018).

ABSTRACT This paper presents a scalable uplink multiple access (SUMA) protocol for bistatic Wi-Fi backscatter systems, composed of a Wi-Fi reader, Wi-Fi helper, and multiple Wi-Fi backscatter tags. SUMA uses a Wi-Fi reader-initiated dynamic framed slotted ALOHA (DFSA)-based multiple access protocol to minimize collisions caused by simultaneous Wi-Fi backscatter uplink traffic from multiple Wi-Fi backscatter tags. In SUMA, the Wi-Fi helper first estimates the number of tags at the start of network operation and derives an appropriate slot-count parameter (i.e., Q), based on which the frame size is specified. Then, the Wi-Fi helper adaptively adjusts the value of Q to maximize network performance while continuously monitoring the number of remaining Wi-Fi backscatter tags to detect information. An experimental simulation was performed to verify the superiority of SUMA. The results demonstrated that SUMA obtained higher performance in terms of the number of collided and empty slots, delay, and throughput compared with the legacy DFSA approach adopted in the EPCglobal Class 1 Gen 2 standard.

INDEX TERMS Anti-collision algorithm, battery-free Internet of Things, DFSA protocol, multiple access, bistatic Wi-Fi backscatter.

I. INTRODUCTION

Backscatter communications harvests energy from ambient radio frequency (RF) sources, such as TV towers, frequency modulation (FM) radio towers, cellular base stations, and Wi-Fi access points (APs), enabling ultralow-power or even battery-free operation of Internet of Things (IoT) devices and significantly mitigating the deployment hurdles of many IoT applications [1]–[3]. Wi-Fi backscatter has recently been considered a promising solution for achieving a battery-free IoT paradigm because commodity Wi-Fi APs and Wi-Fi devices are immediately deployable as the transmitter and receiver of the backscatter communications architecture, significantly reducing its deployment costs [4]–[7]. The Wi-Fi backscatter has a bistatic backscatter system architecture composed of a Wi-Fi helper (as an RF source), Wi-Fi reader (as a receiver), and Wi-Fi backscatter tag (as a transmitter), where the Wi-Fi backscatter tag reflects or absorbs the packets sent from the Wi-Fi helper to enable the Wi-Fi reader to detect the tag information [8]–[10]. In such bistatic systems,

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

it is common to have multiple Wi-Fi backscatter tags. Consequently, the uplink communications from the Wi-Fi backscatter tag to the Wi-Fi reader is highly likely to suffer from frequent collisions, wasting bandwidth and energy and increasing the transmission delay. Therefore, it is indispensable to support multiple access for efficiently reducing collisions between multiple Wi-Fi backscatter tags from a link-layer perspective, which enables the large-scale deployment of Wi-Fi backscatter systems.

In the literature, many prior studies have suggested the use of the dynamic framed slotted ALOHA (DFSA) approach adopted in RF identification (RFID) backscatter systems, which is a reader-driven anti-collision protocol, because of its architectural similarity to the Wi-Fi backscatter [10]–[12]. The EPCglobal Class 1 Gen 2 standard uses a DFSA protocol based on the Q -algorithm, where time is divided into frames consisting of multiple slots [13]–[15]. Moreover, the frame size is dynamically adjusted according to the number of successful slots (i.e., slots used by one RFID tag), collided slots (i.e., slots used by multiple RFID tags simultaneously), and empty slots (i.e., unused slots) in the previous frame. However, if the DFSA approach of EPCglobal Class 1 Gen 2

standard is used as it is in a bistatic Wi-Fi backscatter system, it is challenging to provide scalability with respect to the number of Wi-Fi backscatter tags because its Q-algorithm uses a fixed value of Q for the first frame, and then adjusts the value of Q for subsequent frames without considering the number of Wi-Fi backscatter tags. Consequently, the use of the existing DFSA approach can lead to frame sizes that are too large or too small compared with the number of Wi-Fi backscatter tags in the system, thus creating a large number of empty slots or collided slots within the frame, causing long access delays and frequent collisions.

Many studies have been conducted to efficiently support multiple access for backscatter systems using multiple backscatter tags from a network perspective. Ma *et al.* proposed a carrier sense multiple access with collision avoidance (CSMA/CA)-based solution to reduce collisions in the backscatter system, where the backscatter tags compete with each other for channel access using a binary exponential backoff (BEB) algorithm [16]. However, the carrier sensing process in CSMA/CA may require more energy than the energy harvested from ambient signals; in this case, the backscatter tag cannot operate properly due to lack of energy. Yang *et al.* proposed a time division multiple access (TDMA)-based solution that enables a central controller to allocate unique slots to backscatter tags, thereby achieving collision-free backscatter communications [17]. However, this solution often suffers from high computational and operational complexity, resulting in wasted bandwidth and energy, especially in low contention situations. Thus, it may not be suitable for an energy-limited backscatter system. Khandelwal *et al.* proposed a DFSA-based solution to reduce collision probability by adaptively adjusting the frame size based on the estimated number of backscatter tags [18]. However, it suffers from high computational overhead and performance degradation due to estimation errors. In [18], the reader should estimate the number of backscatter tags at the beginning of every frame. In addition, it uses only the number of empty slots in the previous frame as input to estimate the number of backscatter tags. Then, the frame size is simply determined in proportion to the number of backscatter tags. This causes estimation errors and may degrade network performance. Furthermore, the estimator cannot be used when the number of empty slots is zero due to the mathematical errors of the estimator.

In this paper, we propose a scalable uplink multiple access (SUMA) protocol for bistatic Wi-Fi backscatter systems. SUMA aims to provide a multiple access solution with scalability under environments where multiple Wi-Fi backscatter tags exist, thereby minimizing collisions caused by simultaneous backscatter uplink traffic from multiple Wi-Fi backscatter tags and enabling the practical deployment of battery-free IoT devices. Accordingly, SUMA enhances a Q-algorithm of the DFSA protocol adopted in the EPCglobal Class 1 Gen 2 standard so that the value of Q for the frame can be adaptively changed according to the number of Wi-Fi backscatter tags. In SUMA, the Wi-Fi helper

estimates the number of Wi-Fi backscatter tags by repeatedly counting the number of successful, collided, empty slots during a specific period from the start of network operation and derives an appropriate value of Q based on the estimated number of Wi-Fi backscatter tags. The Wi-Fi helper monitors the number of remaining Wi-Fi backscatter tags to detect the information, based on which the value of Q is adaptively adjusted every frame to maximize network throughput. An experimental simulation was conducted to verify the superiority of SUMA. The results demonstrated that SUMA outperformed the DFSA protocol of the EPCglobal Class 1 Gen 2 standard in terms of the number of collided and empty slots, delay, and throughput compared.

The rest of this paper is organized as follows. Section II describes the related works. In Section III, we present the design of SUMA in detail. Section IV details its simulation configuration and results. Finally, Section V concludes the paper.

II. RELATED WORKS

Efficient multiple access—scalable to the network size—must be supported to enable practical adoption of Wi-Fi backscatter communication. However, to the best of our knowledge, there is still no multiple access protocol for dedicated use in Wi-Fi backscatter communications to address the collision problem from a link layer perspective. Note that the multiple access solutions proposed in [16]–[18] aims to reduce collisions between multiple backscatter tags in conventional or monostatic backscatter system (e.g., RFID system). Nevertheless, they are difficult to be a dedicated approach for bistatic Wi-Fi backscatter communication because of the difference in their system model. Moreover, existing studies [8]–[10] on bistatic Wi-Fi backscatter communication have focused on establishing a communication link with a single Wi-Fi backscatter tag, and thus it is necessary to discuss the existing multiple access protocols to solve the collision problem under environments where multiple Wi-Fi backscatter tags exist. Therefore, in this section, representative multiple access solutions designed to address collisions in wireless communications are investigated to discuss their applicability to Wi-Fi backscatter communications.

A. CSMA/CA-BASED MULTIPLE ACCESS

Many wireless network standards, such as IEEE 802.11x and IEEE 802.15.x, specify medium access control (MAC) protocols that use a CSMA/CA-based approach to reduce collisions between devices [19], [20]. CSMA/CA enables all devices in the same network to have a random chance of accessing the wireless medium through the BEB algorithm [21]–[23]. Specifically, in CSMA/CA, each device randomly selects the number of slots between zero and the minimum contention window minus one ($CW_{\min}-1$). It then waits until the number of slots reaches zero and attempts the packet transmission to the intended receiver. If a collision occurs, the device doubles CW_{\min} and selects the number of slots again using the increased contention window (CW),

which can only be increased up to the maximum contention window (CW_{max}). Naderi *et al.* [24], Kim *et al.* [25], Ha *et al.* [26] proposed CSMA/CA-based MAC protocols for wireless powered sensor networks (WPSNs) in which the energy harvesting sensor devices adjust contention parameters such as slot time, inter-frame space, and CW size to coordinate both energy harvesting and data transmission among one another. In [27], [28], the authors proposed extended CSMA/CA protocols for energy harvesting-aided wireless local area networks (WLANs), which adaptively manage the backoff process of devices according to the residual energy level and the amount of energy consumed to transmit data.

This CSMA/CA-based approach can significantly reduce collisions between Wi-Fi backscatter tags. However, Wi-Fi backscatter tags using the CSMA/CA-based approach require much more energy than they can harvest to repeatedly perform a clear channel assessment (CCA) carrier sensing operation, used to verify whether the channel is busy. A Wi-Fi backscatter system cannot afford to use CSMA/CA in terms of energy, and thus it is not suitable for providing scalable multiple access to multiple Wi-Fi backscatter tags.

Kellogg *et al.* [10], Bharadia *et al.* [8], and Ji *et al.* [29] proposed link layer protocols for bistatic Wi-Fi backscatter, which commonly uses a special packet, *Clear to send (CTS)-to-Self*, to prevent interference from neighboring legacy Wi-Fi devices. When the neighboring Wi-Fi devices receive *CTS-to-Self* from a Wi-Fi helper, they defer their transmissions by setting a network allocation vector (NAV). However, the use of *CTS-to-Self* is not sufficient to support successive transmissions of the Wi-Fi backscatter tag because it can cause the legacy Wi-Fi device to defer its transmission by only up to 32 ms. Moreover, this interference prevention using *CTS-to-Self* cannot be applied to the operation of multiple Wi-Fi backscatter tags that repeatedly reflect or absorb the packets transmitted from a Wi-Fi helper rather than the legacy Wi-Fi devices. Kwon *et al.* [9] proposed a priority-based channel access protocol for Wi-Fi backscatter uplink communications, enabling the Wi-Fi helper to occupy the channel dominantly by tuning several CSMA/CA parameters, such as the distributed coordination function inter-frame space (DIFS), CW_{min} , and CW_{max} . However, in [9], the authors considered a bistatic Wi-Fi backscatter system with a single Wi-Fi backscatter tag; consequently, when multiple Wi-Fi backscatter tags are used, it suffers from an unpredictable delay due to frequent collisions.

B. DFSA-BASED MULTIPLE ACCESS

From the link layer perspective, contributions for the multiple access technique of backscatter communications have occurred primarily in the study of the RFID backscatter system, which has a monostatic architecture composed of passive RFID tags and a reader [30]–[33]. The EPCglobal Class 1 Gen 2 standard was developed for RFID communications in which the battery-free RFID tags modulate and reflect the RF signals sent from a reader to response their data to the reader [13]. The EPCglobal Class 1 Gen 2 standard

minimizes collisions between multiple tags by adopting a DFSA-based anti-collision protocol using the Q-algorithm. In the DFSA-based anti-collision protocol of EPCglobal Class 1 Gen 2 standard, time is divided into multiple frames consisting of multiple slots, and the reader sends a *Query* or *QueryAdjust* command to inform the RFID tags of the frame size (i.e., the number of slots in a frame) at the beginning of each frame. The *Query* and *QueryAdjust* commands are used to inform the initial and adjusted frame sizes, respectively. Upon receiving one of these commands, the RFID tag sets its slot counter to a random value between $[0, \text{frame size}-1]$ and decrements it by one whenever receiving the *QueryRep* command. Then, the RFID tag responses its data to the reader when the slot counter reaches zero. The *QueryRep* command is repeatedly sent at the beginning of each slot except the first slot of the frame. The reader reduces collisions between multiple RFID tags by adjusting the frame size using the Q-algorithm, in which the frame size is 2^Q . The reader dynamically determines the slot-count parameter, Q , based on the number of empty slots and collided slots.

Fig. 1 depicts the operation of the Q-algorithm adopted in the EPCglobal Class 1 Gen 2 standard [13]. Q_{fp} is the floating-point representation of Q . In the Q-algorithm, the reader maintains Q_{fp} and updates it in each slot. The initial value of Q_{fp} is fixed to 4.0 and varies between the minimum value (i.e., zero) and the maximum value (i.e., 15). At the beginning of the frame, the reader rounds Q_{fp} to generate Q expressed as an integer value and sends a *Query* or *QueryAdjust* command including Q to inform the RFID tags of the frame size. Then, the reader updates Q_{fp} based on the number of responses from RFID tags received in each slot. If the number of responses is zero (i.e., empty slot), Q_{fp} is decremented by a constant value, Δ , determined in the range of $[0.1, 0.5]$. In contrast, if there are two or more responses (i.e., collided slot), Q_{fp} is incremented by the same Δ . If there is only one response in a slot, Q_{fp} remains the same as the previous slot. However, the Q-algorithm of the EPCglobal Class 1 Gen 2 standard initializes the frame size using a fixed value of Q_{fp} and then adjusts the frame size without considering the number of RFID tags in the system. Thus, it increases the number of collided and empty slots in the frame, causing a delay and degradation in throughput performance.

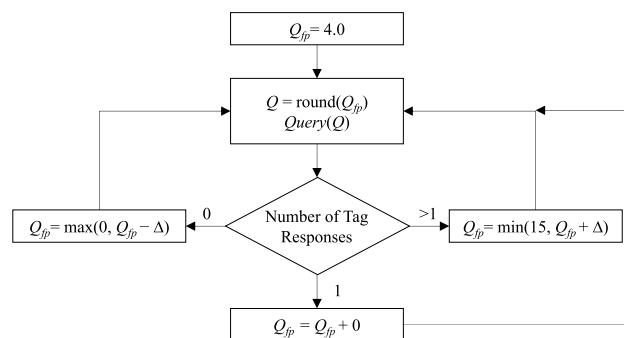


FIGURE 1. Q-algorithm in EPCglobal Class 1 Gen 2 standard.

Furthermore, it is difficult to apply it as it is to a bistatic Wi-Fi backscatter system composed of the Wi-Fi reader, Wi-Fi helper, and Wi-Fi backscatter tags because of the monostatic architecture of the RFID backscatter system.

III. DESIGN OF SUMA

SUMA is designed to minimize collisions caused by simultaneous Wi-Fi backscatter uplink traffic from multiple tags. Accordingly, it provides a Wi-Fi reader-initiated DFSA-based multiple access protocol for bistatic Wi-Fi backscatter systems. Fig. 2 illustrates the system model of SUMA, which is composed of a Wi-Fi helper (act as an RF source), a Wi-Fi reader (act as a receiver), and multiple Wi-Fi backscatter tags (act as transmitters). Wi-Fi backscatter communication is initiated or terminated through the message exchange between the Wi-Fi reader and the Wi-Fi helper. Once the Wi-Fi backscatter communication starts, the Wi-Fi helper first broadcasts a command to enable random access of multiple Wi-Fi backscatter tags, and then sends several packets consecutively. The Wi-Fi backscatter tags accessing the channel reflect or absorb the packets sent from the Wi-Fi helper so that the Wi-Fi reader can detect the tag information. Then, the Wi-Fi reader notifies the detection result for the tag information by sending a message to the Wi-Fi helper and the Wi-Fi backscatter tags. This operation is repeated until the Wi-Fi reader receives the tag information for all Wi-Fi backscatter tags.

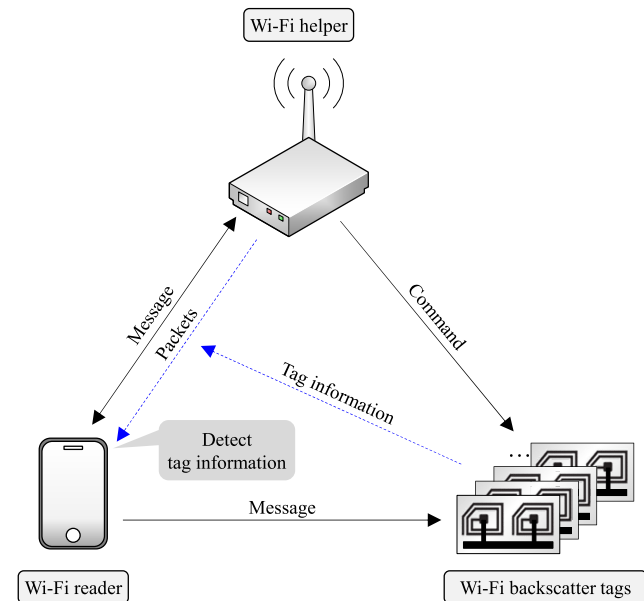


FIGURE 2. System model of SUMA.

The system model of SUMA inherently has a bistatic backscatter system architecture because a Wi-Fi AP and a Wi-Fi device act as a Wi-Fi helper and a Wi-Fi reader, respectively. Additionally, unlike the existing bistatic Wi-Fi backscatter, it includes multiple backscatter tags. Note that a conventional or monostatic backscatter system (e.g., RFID

system) consists of two main components: a reader and a tag, and the reader contains both an RF source and a receiver in the same device. On the other hand, in a bistatic backscatter system (e.g., Wi-Fi backscatter), an RF source and a receiver are separated into different devices. In SUMA, the Wi-Fi helper estimates the number of Wi-Fi backscatter tags at the start of network operation and calculates an appropriate slot-count parameter, Q , by which to specify the frame size. Then, the Wi-Fi helper adaptively adjusts the value of Q to maximize network performance by considering the number of remaining tags to detect information. In this section, we describe the design of SUMA in detail.

A. OVERALL OPERATION

Fig. 3 illustrates the superframe structure of SUMA, which includes four operational periods: 1) Request Period (RP), 2) Tag Estimation Period (TEP), 3) Communications Period (CP), and 4) Termination Period (TP). The RP and TP are periods to initiate and terminate Wi-Fi backscatter communications, and the TEP and CP are periods to estimate the number of tags and detect the tag information, respectively. The TEP and CP are composed of multiple frames, each of which is divided into multiple slots, and in these periods, the Wi-Fi backscatter system operates in a DFSA manner as in the EPCglobal Class 1 Gen 2 standard.

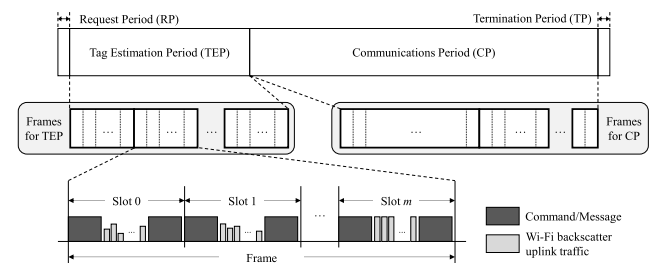


FIGURE 3. Superframe structure of SUMA.

Fig. 4 illustrates the overall operation of SUMA. In RP, the reader sends a *REQ* message to the helper to initiate Wi-Fi backscatter communications. In TEP and CP, the helper broadcasts *Query*, *QueryAdjust*, and *QueryRep* commands at the beginning of each slot, which commonly announces the start of a new frame and includes a slot-count parameter, Q , to specify the frame size. Note that, in TEP, a fixed value of Q is used for all frames, while in CP, it can be adjusted to a different value for each frame depending on the number of remaining tags to detect information. *QueryAdjust* indicates that the value of Q used in the previous frame has changed. *QueryRep* command is transmitted in the slots after the first slot, and its value of Q is the same as that of the *Query* or *QueryAdjust* in the first slot. Upon receiving the *Query* or *QueryAdjust* command, each tag initializes its slot counter by randomly selecting the number in the range of $[0, 2^Q - 1]$. On the other hand, the tag decrements its slot counter by one whenever receiving the *QueryRep* command. Then, it checks the value of the slot counter to determine whether to start the

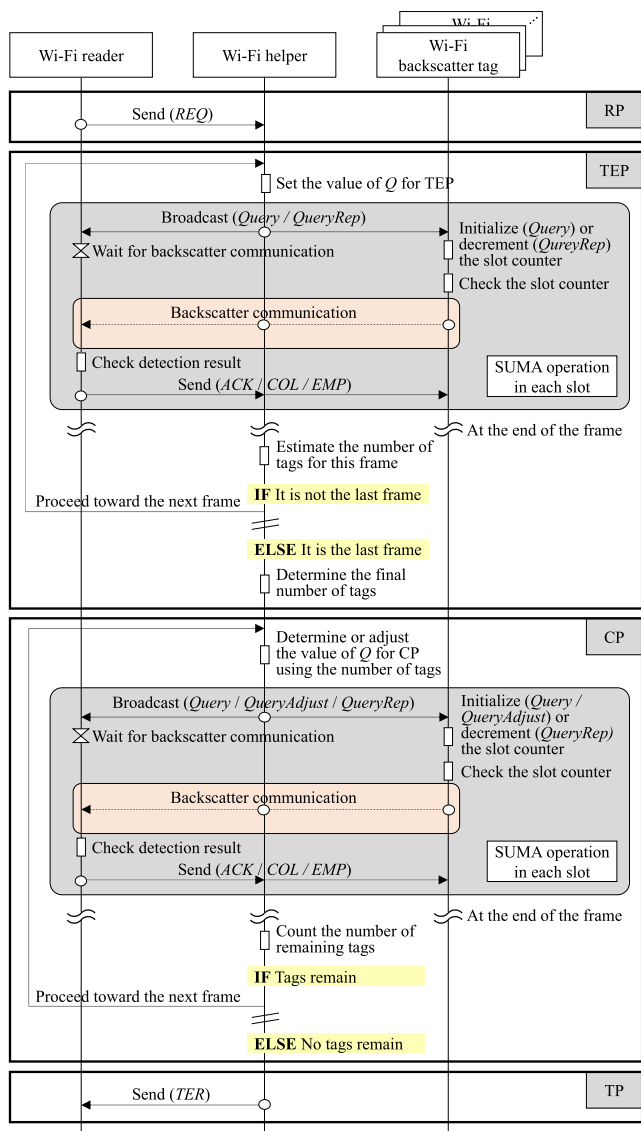


FIGURE 4. Overall operation of SUMA.

backscatter communication. When the slot counter reaches zero, the tag reflects or absorbs the packets from the helper in order for the reader to detect the tag information. In each slot, the reader selectively sends *ACK*, *COL*, and *EMP* messages according to the detection result of the tag information. If the reader successfully detects the tag information of a particular tag, it sends *ACK*. In contrast, *COL* is sent when the multiple tags simultaneously reflect the helper’s packets, and *EMP* is sent if no tag information is detected during a slot. With these messages, the helper can identify each slot as successful, collided, or empty. Note that, in SUMA, we consider the use of a backscatter tag capable of both uplink and downlink communications so that it can receive messages (i.e., *ACK*, *COL*, and *EMP*) transmitted by the reader. The reader sends directly *ACK*, *COL*, and *EMP* messages with different lengths to the tag. The tag is powered through an RF signal (message transmission) transmitted from the reader, and

identifies each message using the difference in time when energy is detected.

If the helper has received a *REQ* message from the reader in RP, and the *REQ* message does not include the number of tags, it tries to estimate the number of tags in the TEP. The TEP consists of multiple frames with the same size. At the end of each frame, the helper estimates the number of tags by counting the number of successful, collided, and empty slots. Then, the helper calculates the average of the estimated number of tags in each frame and determines the value as the final number of tags. Then, it ends the TEP and proceeds to the CP. The *REQ* message may include the number of tags through user input. In this case, the helper proceeds toward CP without operating in TEP. The estimation procedure for the number of tags is described in Section III.B, in detail. The CP is the period in which tag information is obtained from multiple tags. After the number of tags is obtained, the helper initiates CP. The CP consists of multiple frames having different numbers of slots. In CP, the value of Q in the first frame is determined based on the number of tags in the bistatic Wi-Fi backscatter system. It is continuously adjusted based on the number of remaining tags to detect tag information as the frames progress. When no tags remain to detect the tag information, the CP ends.

If the number of tags is estimated through TEP, the estimated number of tags may differ from the number of tags actually deployed in the bistatic Wi-Fi backscatter system. Therefore, at the end of each frame, the helper decides whether to proceed toward the next frame or end the CP. Specifically, if the number of empty slots in the current frame is equal to the current frame size, the helper decides that no tags remain to detect tag information and ends the CP; otherwise, it proceeds to the next frame in the CP. The adjustment procedure for the value of Q is described in Section III.C, in detail. Then, in TP, the helper sends a *TER* message to the reader to terminate the current Wi-Fi backscatter communications.

B. TAG ESTIMATION

In SUMA, the reader initiates Wi-Fi backscatter communications by sending a *REQ* message to the helper in RP. If the received *REQ* message includes the number of tags, the helper can proceed directly toward CP without operating in TEP. Otherwise, the helper must estimate the number of tags in TEP before the operation in CP. Algorithm 1 presents the procedure for estimating the number of tags in the bistatic Wi-Fi backscatter system. In the algorithm, the helper initializes the variables (i.e., i_{tep} , n_{frame} , $n_{tag,i_{tep}}$, n_{avg_tag} , and n_{tag}), where i_{tep} denotes the frame index for frames in TEP, n_{frame} is the number of frames in which the number of tags is estimated, $n_{tag,i_{tep}}$ is the number of tags temporarily estimated in the frame i_{tep} , n_{avg_tag} is the average number of tags estimated in each frame, and n_{tag} is the number of tags finally determined by the algorithm. Moreover, the helper maintains a predefined threshold of n_{frame} , $n_{frame_threshold}$,

which is used as a reference value to determine the number of tags and determined empirically in advance.

In the first slot of the frame i_tep , the helper broadcasts a *Query* command containing Q_{tep} , which is an arbitrary value of Q used to estimate the number of tags in TEP; thus, the size of all frames in TEP is equal to is $2Q_{tep}$. For estimating the number of tags under the same condition for each frame, a fixed value of Q_{tep} is used for all frames within TEP, and this value is empirically determined in advance. Upon receiving the *Query*, the tag sets its slot counter to a randomly selected value in the range of $[0, 2Q_{tep} - 1]$. Then, the helper starts sending a series of packets to enable the reader to obtain the tag information from multiple tags. Only the tags with a slot counter of zero reflect or absorb the packets to deliver their tag information. When the reader successfully detects the tag information by measuring the received signal strength indicator (RSSI), it responds with an *ACK* message to the helper and the corresponding tag; the tag then ends its operation. In contrast, if the reader detects a collision or detects nothing, it broadcasts a *COL* or *EMP* message, respectively. The *COL* message makes the tags wait until the next frame, and if the tag receives an *EMP* message, it does nothing. When *ACK*, *COL*, and *EMP* messages are received, the helper updates the number of successful slots (n_{ACK}), the number of collided slots (n_{COL}), and the number of empty slots (n_{EMP}), respectively. Note that n_{ACK} , n_{COL} , and n_{EMP} are initialized to zero at the beginning of each frame and represented by n_{ACK,i_tep} , n_{COL,i_tep} , and n_{EMP,i_tep} in the frame i_tep . From the second slot, the helper sends a *QueryRep* command, followed by a series of packets. The tag decrements its slot counter by one when receiving the *QueryRep* command and reflects or absorbs the packets from the helper when the slot counter reaches zero. This operation is repeated until the last slot of the frame (i.e., $2Q_{tep}$ -th slot).

At the end of every frame, the helper estimates the number of tags in the bistatic Wi-Fi backscatter system based on the expected and measured number of successful, collided, and empty slots. For estimating the number of tags in the frame i_tep (i.e., n_{tag,i_tep}), the helper calculates the expected number of empty, successful, and collided slots for the frame i_tep . None (i.e., empty slot), one tag (i.e., successful slot), or multiple tags (i.e., collided slot) can occupy each slot; thus, the probability that the slot is occupied by k tags among n tags can be expressed by Eq. (1) when the frame size is L ($L = 2Q_{tep}$) [34].

$$f(k, n, p) = \binom{n}{k} p^k (1-p)^{n-k}, \quad 0 \leq k \leq n \quad (1)$$

where p is the probability that one tag occupies a random slot within a frame ($p = 1/L = 2^{-Q_{tep}}$). The expected number of empty, successful, and collided slots in the frame i_tep can be calculated from Eq. (1) when k is zero, k is one, and k is two or more, respectively. Therefore, $E[n_{EMP,i_tep}]$, $E[n_{ACK,i_tep}]$, and $E[n_{COL,i_tep}]$ can be calculated by Eqs. (2)–(4).

$$E[n_{EMP,i_tep}] = L(1-p)^n = 2^{Q_{tep}} \left(1 - 2^{-Q_{tep}}\right)^n \quad (2)$$

TABLE 1. Notation list for algorithm 1.

Notation	Description
Q	Slot-count parameter
i_tep	Frame index in TEP
n_{frame}	Number of frames in TEP
n_{tag,i_tep}	Number of tags estimated in frame i_tep
n_{avg_tag}	Average number of tags estimated in each frame of TEP
n_{tag}	Number of tags finally determined in TEP
$n_{frame_threshold}$	Threshold of n_{frame}
Q_{tep}	Arbitrary value of Q to estimate the number of tags in TEP
n_{ACK}	Number of successful slots
n_{COL}	Number of collided slots
n_{EMP}	Number of empty slots
n_{ACK,i_tep}	Number of successful slots in frame i_tep
n_{COL,i_tep}	Number of collided slots in frame i_tep
n_{EMP,i_tep}	Number of empty slots in frame i_tep
L	Frame size
p	Probability that one tag occupies a slot within a frame
$\mathbf{E}_{i_tep}(n)$	Set of the expected number of empty, successful, and collided slots in frame i_tep
\mathbf{O}_{i_tep}	Set of the observed number of empty, successful, and collided slots in frame i_tep

$$E[n_{ACK,i_tep}] = n(1-p)^{n-1} = n \left(1 - 2^{-Q_{tep}}\right)^{n-1} \quad (3)$$

$$E[n_{COL,i_tep}] = L - E[n_{EMP,i_tep}] - E[n_{ACK,i_tep}] \quad (4)$$

Then, the helper calculates n_{tag,i_tep} using Eq. (5) [35].

$$n_{tag,i_tep} = \min_n \|\mathbf{E}_{i_tep}(n) - \mathbf{O}_{i_tep}\|^2 \quad (5)$$

where $\mathbf{E}_{i_tep}(n)$ is a set of the expected number of slots represented by $[E[n_{EMP,i_tep}], E[n_{ACK,i_tep}], E[n_{COL,i_tep}]]^T$, \mathbf{O}_{i_tep} is a set of the observed number of slots represented by $[n_{EMP,i_tep}, n_{ACK,i_tep}, n_{COL,i_tep}]^T$, and $\|\cdot\|$ is the Euclidean norm. After calculating n_{tag,i_tep} , the helper averages the estimated number of tags for each frame to obtain n_{avg_tag} and increments n_{frame} by one. If n_{frame} is smaller than $n_{frame_threshold}$, the helper proceeds to the next frame by sending a *Query* command and repeats the above operation. Otherwise, it finally determines the number of tags of the bistatic Wi-Fi backscatter system (i.e., n_{tag}) as n_{avg_tag} .

C. Q ADJUSTMENT

In SUMA, the value of Q specifies the number of slots in a frame (i.e., frame size), which should be fitted to the number of tags in the network (i.e., network size) to maximize throughput. After the number of tags in the bistatic Wi-Fi backscatter system, n_{tag} is obtained through a *REQ* message, including the number of tags or the tag estimation procedure in TEP, the helper starts its operation in CP by broadcasting

Algorithm 1 Estimation Procedure for the Number of Tags

```

1: INITIALIZE  $i\_tep$ ,  $n_{frame}$ ,  $n_{tag,i\_tep}$ ,  $n_{avg\_tag}$ , and  $n_{tag}$  to 0
2: /* Repeat the estimation procedure. */
   WHILE  $n_{frame} < n_{frame\_threshold}$ 
3:    $n_{ACK,i\_tep} \leftarrow 0$ 
      $n_{COL,i\_tep} \leftarrow 0$ 
      $n_{EMP,i\_tep} \leftarrow 0$ 
4:   /* Repeat for all slots. */
     FOR each slot,  $j$ ,  $j \in [0, 2^{Q_{tep}} - 1]$ 
5:     Notify the value of  $Q$  and wait for the response from the reader
     // Send Query or QueryRep command.
6:     Update  $n_{ACK,i\_tep}$ ,  $n_{COL,i\_tep}$ , or  $n_{EMP,i\_tep}$ 
     // Accumulate the number of ACK, COL, and EMP messages.
7:   ENDFOR
8:   Calculate  $n_{tag,i\_tep}$  // Estimate the number of tags for the frame  $i\_tep$ .
9:    $n_{avg\_tag} \leftarrow (n_{tag} + n_{tag,i\_tep}) / (i\_tep + 1)$ 
10:   $n_{frame} \leftarrow n_{frame} + 1$ 
11:   $i\_tep \leftarrow i\_tep + 1$  // Increment the frame index.
12: ENDWHILE
13:   $n_{tag} \leftarrow n_{avg\_tag}$ 
14: RETURN  $n_{tag}$ 

```

a *Query* command. In CP, the helper determines the value of Q of the first frame based on the obtained number of tags and then adjusts it continuously based on the number of remaining tags to detect the tag information as the frames progress, in order to maximize throughput. Specifically, the helper counts the received *ACK* messages so that the latest number of remaining tags to detect tag information is maintained. The number of remaining tags in the frame i_cp (n_{i_cp}) can be given by Eq. (6).

$$n_{i_cp} = \begin{cases} n_{tag}, & i_cp = 0 \\ n_{tag} - \sum_{j=0}^{i_cp-1} n_{ACK,j}, & i_cp \geq 1 \end{cases} \quad (6)$$

where i_cp denotes the frame index of frames in CP and n_{ACK,i_cp} is the number of successful slots in the frame i_cp .

Algorithm 2 presents a procedure for adjusting the value of Q . In the algorithm, the helper initializes the variables (i.e., Q_{i_cp} , Q_{temp} , and Th_{max,i_cp}), where Q_{i_cp} is the value of Q for the frame i_cp , Q_{temp} is a temporary value of Q to search the values of Q , and Th_{max,i_cp} is the maximum throughput for the frame i_cp , respectively. Furthermore, n_{ext_frame} is the number of extra frames used when the estimated number of tags (i.e., n_{tag}) is less than the number of tags actually deployed in the Wi-Fi backscatter system (i.e., n_{act_tag}). In the opposite case, the extra frame is not used because the CP ends before the number of remaining tags becomes zero. Moreover, Q_{max} is the largest value of Q that can be selected and $Th_{i_cp}(Q_{temp})$ is a function that outputs the throughput

TABLE 2. Notation list for algorithm 2.

Notation	Description
i_cp	Frame index in CP
n_{i_cp}	Number of remaining tags in frame i_cp
n_{ACK,i_cp}	Number of successful slots in frame i_cp
Q_{i_cp}	Value of Q in frame i_cp
Q_{temp}	Temporary value of Q
Th_{max,i_cp}	Maximum throughput for frame i_cp
n_{ext_frame}	Number of extra frames in CP
n_{act_tag}	Number of actually deployed tags
Q_{max}	Largest value of Q
$Th_{i_cp}(Q_{temp})$	Throughput function for frame i_cp
P_{EMP}	Probability of empty slot in a particular frame
P_{ACK}	Probability of successful slot in a particular frame
P_{COL}	Probability of collided slot in a particular frame
P_{EMP,i_cp}	Probability of empty slot in frame i_cp
P_{ACK,i_cp}	Probability of successful slot in frame i_cp
P_{COL,i_cp}	Probability of collided slot in frame i_cp
T_{Slot}	Slot duration
T_{tag}	Duration for Wi-Fi backscatter uplink traffic
T_{CS}	Duration for <i>CTS-to-Self</i> transmission
n_{pkt}	Number of packets in a slot
T_{pkt}	Duration of a packet transmission
T_{Query}	Duration for <i>Query</i>
$T_{QueryAdjust}$	Duration for <i>QueryAdjust</i>
$T_{QueryRep}$	Duration for <i>QueryRep</i>
T_{ACK}	Duration for <i>ACK</i>
T_{EMP}	Duration for <i>EMP</i>
T_{COL}	Duration for <i>COL</i>
Th_{i_cp}	Throughput for frame i_cp
T_{ACK,i_cp}	Successful slot duration in frame i_cp
T_{EMP,i_cp}	Empty slot duration in frame i_cp
T_{COL,i_cp}	Collided slot duration in frame i_cp
T_{Slot,i_cp}	Slot duration in frame i_cp

for the frame i_cp using Q_{temp} as an input. The function $Th_{i_cp}(Q_{temp})$ is valid only when n_{i_cp} is greater than zero. However, n_{i_cp} can be zero or negative. If n_{tag} is less than or equal to n_{act_tag} . Furthermore, when $n_{tag} \leq n_{act_tag}$, if n_{i_cp} is one, the helper cannot receive any *ACK* message due to repeated collisions. Specifically, when n_{i_cp} is one, the value of Q that maximizes the throughput of the function $Th_{i_cp}(Q_{temp})$ is zero, which causes all tags to set the slot counter to zero. Therefore, in the algorithm, the helper first verifies n_{i_cp} before adjusting the value of Q . If n_{i_cp} is greater than one, the helper iteratively calculates the throughput using

Algorithm 2 Q Adjustment Procedure

```

1: INITIALIZE  $Q_{i\_cp}$ ,  $Q_{temp}$ , and  $Th_{max,i\_cp}$  to 0
2: IF  $n_{i\_cp} > 1$  // If the number of remaining tag in the
   frame  $i\_cp$  is greater than one.
3:    $n_{ext\_frame} \leftarrow -1$  // Initialize the number of extra
   frames.
4:   /* Find the value of  $Q$  that maximizes throughput. */
   FOR each temporary value of  $Q$ ,  $Q_{temp}$ ,  $Q_{temp} \in$ 
    $[0, Q_{max}]$ 
5:     IF  $Q_{temp} == 0$ 
6:        $Th_{max,i\_cp} \leftarrow Th_{i\_cp}(Q_{temp})$  // Calculate the
   throughput using the
   first value of  $Q$ .
7:      $Q_{i\_cp} \leftarrow Q_{temp}$  // Update  $Q_{i\_cp}$  to  $Q_{temp}$ .
8:     ELSE
9:        $Th_{max,i\_cp} \leftarrow \max[Th_{max,i\_cp}, Th_{i\_cp}(Q_{temp})]$ 
   // Find the maximum throughput.
10:      IF  $Th_{max,i\_cp} > Th_{i\_cp}(Q_{temp})$ 
11:         $Q_{i\_cp} \leftarrow Q_{i\_cp}$  // Maintain the existing
   value of  $Q$  if the maximum
   throughput does not change.
12:      ELSE
13:         $Q_{i\_cp} \leftarrow Q_{temp}$  // Update  $Q_{i\_cp}$  to  $Q_{temp}$  if
   the maximum throughput
   changes.
14:      ENDIF
15:    ENDIF
16:  ENDFOR
17: ELSE // If the number of remaining tags in the frame
    $i\_cp$  is less than two.
18:    $n_{ext\_frame} \leftarrow n_{ext\_frame} + 1$  // Increment the number
   of extra frames by
   one when  $n_{i\_cp} \leq 1$ .
19:    $Q_{i\_cp} \leftarrow n_{ext\_frame}$  // Determine the value of  $Q$  for
   extra frame.
20: ENDIF
21: RETURN  $Q_{i\_cp}$  // Determine the value of  $Q$  for the
   frame  $i\_cp$ .

```

all values of Q_{temp} in the range of $[0, Q_{max}]$ and then adjusts Q_{i_cp} to Q_{temp} to maximize the throughput for the frame i_cp . Otherwise, the helper sets Q_{i_cp} to n_{ext_frame} .

In the algorithm, the throughput for a particular frame is a key criterion for adjusting the value of Q , for which the helper first calculates the probability of an empty slot (P_{EMP}), successful slot (P_{ACK}), and collided slot (P_{COL}) in the particular frame. From Eq. (1), P_{EMP} , P_{ACK} , and P_{COL} can be given as Eqs. (7)–(9) when k is zero, k is one, and k is two or more, respectively.

$$P_{EMP} = (1 - p)^n \tag{7}$$

$$P_{ACK} = np(1 - p)^{n-1} \tag{8}$$

$$P_{COL} = 1 - P_{EMP} - P_{ACK} \tag{9}$$

In CP, each frame has different P_{EMP} , P_{ACK} , and P_{COL} based on the number of remaining tags. Thus, the probabilities for the frame i_cp (i.e., P_{EMP,i_cp} , P_{ACK,i_cp} , and P_{COL,i_cp}) are given by Eqs. (10)–(12), as shown at the bottom of the next page.

Fig. 5 illustrates the structure of a successful slot in which the duration for Wi-Fi backscatter uplink traffic is given by Eq. (13).

$$T_{tag} = T_{CS} + n_{pkt} (SIFS + T_{pkt}) \tag{13}$$

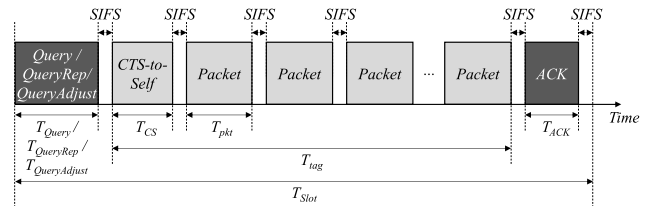


FIGURE 5. Structure of successful slot.

where T_{CS} is the duration for *CTS-to-Self* transmission, n_{pkt} is the number of packets in a slot, *SIFS* is the short inter-frame space, and T_{pkt} is the duration of a packet's transmission. In SUMA, we consider using *SIFS* between burst packets to guarantee reliable uplink traffic within a slot, which is proposed in [9]; thus, the helper sends each packet at *SIFS* interval. *CTS-to-Self* is a control packet that forces the neighboring legacy Wi-Fi devices to defer their transmissions for a specific period (by up to 32 ms). In SUMA, the *QueryAdjust* command is sent only when the value of Q used in the previous frame has changed. However, for simplicity of expression of the expected duration for each slot, we assume that the *Query* command is sent only in the first frame and that the *QueryAdjust* command is sent in other frames. Therefore, the expected duration of the successful slot for the frame i_cp can be obtained by Eq. (14), as shown at the bottom of the next page, where T_{Query} , $T_{QueryAdjust}$, $T_{QueryRep}$ and T_{ACK} are the duration for the *Query*, *QueryAdjust*, *QueryRep* commands and the *ACK* message, respectively. Similarly, the expected duration of the empty and collided slots for the frame i_cp can be given by Eqs. (15) and (16), as shown at the bottom of the next page, where T_{EMP} and T_{COL} are the duration for *EMP* and *COL* messages, respectively. In the bistatic Wi-Fi backscatter system, a single packet is decoded into one bit. Therefore, the size of tag information is equal to n_{pkt} . However, the tag information includes an 8-bit preamble; thus, the payload size of the tag information becomes $n_{pkt} - 8$. The throughput for the frame i_cp (Th_{i_cp}) can be expressed by Eq. (17), as shown at the bottom of the next page.

Fig. 6 illustrates an example of CP operation. In the example, it is assumed that the total number of tags is three. At Slot 0 in the first frame, the helper broadcasts the *Query* command containing $Q = 2$ derived through Algorithm 2. Upon receiving the command, the tags randomly set their slot counters in the range $[0, 3]$. In Slot 0, all the tags do nothing because their slot counters are greater than zero, and thus the reader sends

an *EMP* message. The helper broadcasts a *QueryRep* command at the beginning of Slot 1. In this slot, the slot counter of each tag is decremented by one; thus, the tag 1's slot counter becomes zero, reflecting and absorbing the packets sent from the helper. In this case, the reader sends the *ACK* message to notify the successful detection of tag information. Upon receiving the *ACK* message, tag 1 ends its communications. The packet transmitted from the helper is used to decode one bit in tag information. In each frame, the helper consecutively sends multiple packets at *SIFS* intervals so that the reader can detect the tag information composed of several bits. Through the *SIFS* interval, the reader and tags distinguish individual packets within a frame. In Slot 3, the collision occurs because the slot counters of the multiple tags become zero simultaneously. Therefore, the reader sends the *COL* message, and the corresponding tags wait for the next frame. When the second frame starts, the helper adjusts the value of Q to one derived through Algorithm 2 and broadcasts the *QueryAdjust* command. Tags 2 and 3 reset their slot counters in the range of $[0, 1]$. Then, the operation repeats until the number of remaining tags reaches zero.

IV. PERFORMANCE EVALUATION

We evaluated the performance of SUMA through experimental simulations using the MATLAB simulator. The simulation results of SUMA were compared with those of the DFSA-based anti-collision protocol adopted in the EPC-global Class 1 Gen 2 standard. In the following subsections, we describe the simulation setting and configuration and discuss the simulation results in detail.

A. SIMULATION SETTING AND CONFIGURATION

We considered a bistatic Wi-Fi backscatter system consisting of a Wi-Fi reader, a Wi-Fi helper, and several Wi-Fi backscatter tags. We also assumed that each device is deployed within each other's communications range. SUMA focuses on supporting the scalability of multiple access to the network size (i.e., the number of tags). Thus, in the simulation, the number of tags varies from 10 to 80 to investigate the performance of SUMA as the number of tags changes. We further assumed that the *REQ* message received from the reader does not contain information on the number of tags; thus, the helper performs the tag estimation procedure

$$P_{EMP,i_{cp}} = (1 - p_{i_{cp}})^{n_{i_{cp}}} = \begin{cases} (1 - 2^{-Q_{i_{cp}}})^{n_{tag}}, & i_{cp} = 0 \\ (1 - 2^{-Q_{i_{cp}}})^{n_{tag} - \sum_{j=0}^{i_{cp}-1} n_{ACK,j}}, & i_{cp} \geq 1 \end{cases} \quad (10)$$

$$P_{ACK,i_{cp}} = n_{i_{cp}} p_{i_{cp}} (1 - p_{i_{cp}})^{n_{i_{cp}}-1} = \begin{cases} n_{tag} (2^{-Q_{i_{cp}}}) (1 - 2^{-Q_{i_{cp}}})^{n_{tag}-1}, & i_{cp} = 0 \\ \left(n_{tag} - \sum_{j=0}^{i_{cp}-1} n_{ACK,j} \right) (2^{-Q_{i_{cp}}}) (1 - 2^{-Q_{i_{cp}}})^{n_{tag} - \sum_{j=0}^{i_{cp}-1} n_{ACK,j}-1}, & i_{cp} \geq 1 \end{cases} \quad (11)$$

$$P_{COL,i_{cp}} = 1 - P_{EMP,i_{cp}} - P_{ACK,i_{cp}} \quad (12)$$

$$E[T_{ACK,i_{cp}}] = \begin{cases} 2^{-Q_{i_{cp}}} T_{Query} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{ACK} + 3SIFS, & i_{cp} = 0 \\ 2^{-Q_{i_{cp}}} T_{QueryAdjust} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{ACK} + 3SIFS, & i_{cp} \geq 1 \end{cases} \quad (14)$$

$$E[T_{EMP,i_{cp}}] = \begin{cases} 2^{-Q_{i_{cp}}} T_{Query} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{EMP} + 3SIFS, & i_{cp} = 0 \\ 2^{-Q_{i_{cp}}} T_{QueryAdjust} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{EMP} + 3SIFS, & i_{cp} \geq 1 \end{cases} \quad (15)$$

$$E[T_{COL,i_{cp}}] = \begin{cases} 2^{-Q_{i_{cp}}} T_{Query} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{COL} + 3SIFS, & i_{cp} = 0 \\ 2^{-Q_{i_{cp}}} T_{QueryAdjust} + (1 - 2^{-Q_{i_{cp}}}) T_{QueryRep} + T_{tag} + T_{COL} + 3SIFS, & i_{cp} \geq 1 \end{cases} \quad (16)$$

$$\begin{aligned} Th_{i_{cp}} &= \frac{E[\text{Payload size of tag information in frame } i_{cp}]}{E[\text{Duration of frame } i_{cp}]} \\ &= \frac{(n_{pkt} - 8) 2^{Q_{i_{cp}}} P_{ACK,i_{cp}}}{2^{Q_{i_{cp}}} E[T_{Slot,i_{cp}}]} \\ &= \frac{(n_{pkt} - 8) P_{ACK,i_{cp}}}{P_{EMP,i_{cp}} E[T_{EMP,i_{cp}}] + P_{ACK,i_{cp}} E[T_{ACK,i_{cp}}] + P_{COL,i_{cp}} E[T_{COL,i_{cp}}]} \\ &= \frac{(n_{pkt} - 8) P_{ACK,i_{cp}}}{1 + P_{EMP,i_{cp}} (T_{EMP} - T_{COL}) + P_{ACK,i_{cp}} (T_{ACK} - T_{COL})} \end{aligned} \quad (17)$$

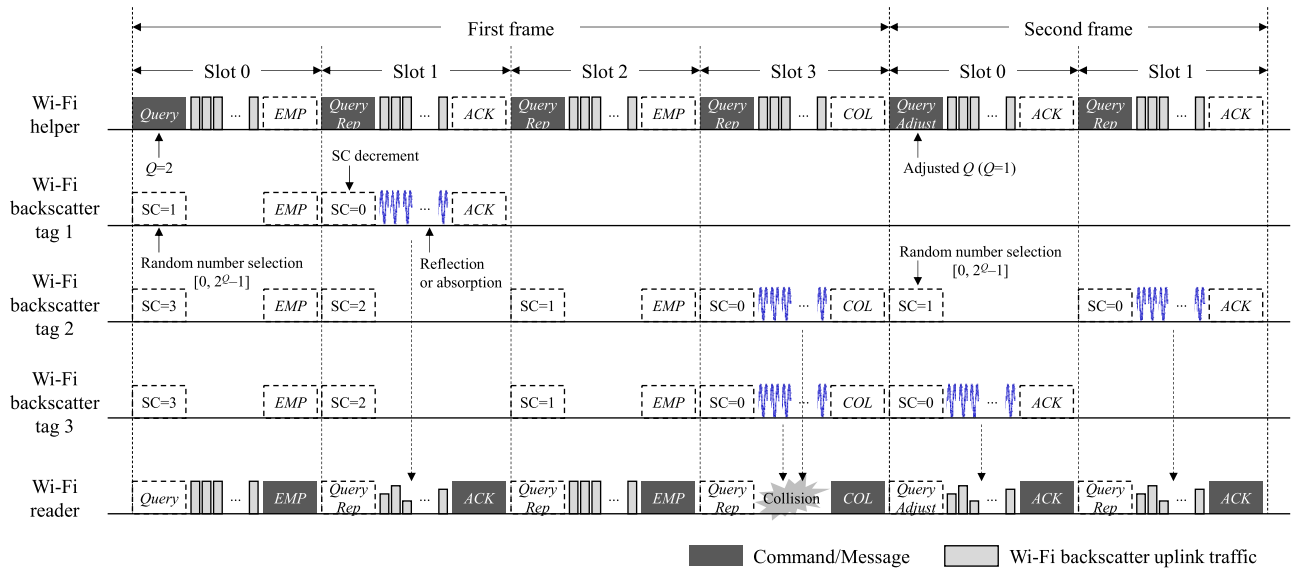


FIGURE 6. Example of CP operation.

during the TEP before proceeding with the CP. Two parameters, Q_{tep} and $n_{frame_threshold}$, used in Algorithm 1, should be obtained empirically. Accordingly, we iteratively estimated the number of tags using various parameters for Q_{tep} and $n_{frame_threshold}$. Then, the parameter values that minimize the sum of the lengths for the TEP and CP periods were derived, respectively. Two different sizes of payload, 16-bit and 32-bit, were considered when investigating the impact of payload size in tag information on delay and throughput performance.

For a comparative study, we compared the performance of SUMA with that of the DFSA-based anti-collision protocol of the EPCglobal Class 1 Gen 2 standard (i.e., the legacy DFSA). In SUMA, the helper adjusts the value of Q of each frame based on the number of tags and sends a *Query* or *QueryAdjust* command with the adjusted value of Q . In contrast, for the legacy DFSA, the helper sets the initial value of Q to 4.0 and changes the value of Q based on the number of collided and empty slots in the previous frame. Conventionally, the helper increases the value of Q by Δ when the collision occurs and decreases it by Δ when there is no response from the tag. Δ was set in the range of [0.1, 0.5]. In the simulation, *ACK*, *COL*, and *EMP* messages were set to have different lengths so that the tags can identify each message. Moreover, considering the *Query*, *QueryRep*, and *QueryAdjust* commands that contain different information, the lengths of the commands were also set to 22, 4, and 9 bytes, respectively. The simulation was iterated 200 times. The detailed simulation parameters are listed in Table 3.

B. SIMULATION RESULTS

In this paper, the operation of SUMA includes the TEP period to estimate the number of tags in the Wi-Fi backscatter system. If the helper receives a *REQ* message that does not contain the number of tags in the RP period, it attempts to

TABLE 3. Simulation parameters.

Parameter	Value
PHY/MAC	IEEE 802.11n
Data rate	300 Mbps
<i>SIFS</i>	10 μ s
<i>CTS-to-Self</i>	14 bytes
Packet	1,024 bytes
Q	0–15
Number of Wi-Fi backscatter tags	10–80
Preamble of tag information	8 bits
Payload of tag information	16, 32 bits
<i>REQ</i>	20 bytes
<i>TER</i>	10 bytes
<i>Query</i>	22 bytes
<i>QueryRep</i>	4 bytes
<i>QueryAdjust</i>	9 bytes
<i>ACK</i>	14 bytes
<i>COL</i>	10 bytes
<i>EMP</i>	8 bytes
Δ	0.1–0.5

estimate the number of tags using the estimation procedure of Algorithm 1. This estimation procedure is affected by two parameters, $n_{frame_threshold}$ and Q_{tep} , determined empirically in advance. Therefore, we first experimentally analyzed the impact of these two parameters on the TEP and CP periods and derived the values that minimize the sum of the TEP

and CP period lengths. In the experiment, the values of $n_{frame_threshold}$ and Q_{tep} were set to vary in the ranges [2, 5] and [0, 10], respectively, and the tag information was set to a 32-bit payload.

Figs. 7(a) and 7(b) illustrate the variations in the length of the TEP period as the value of $n_{frame_threshold}$ changes for fixed values of Q_{tep} and those in the opposite case, respectively. The length of the TEP period increases as the values of $n_{frame_threshold}$ and Q_{tep} increase because the number of frames in the TEP period is determined by $n_{frame_threshold}$, and the number of slots in each frame is determined by Q_{tep} . The number of slots in one frame is $2^{Q_{tep}}$, and thus the length of TEP period increases rapidly when Q_{tep} increases. Moreover, the TEP period ends when the number of frames reaches $n_{frame_threshold}$. Therefore, the number of tags actually deployed in the Wi-Fi backscatter system (i.e., n_{act_tag}) does not affect the length of the TEP period. Even if the value of n_{act_tag} changes, the length of the TEP period does not change when the values of $n_{frame_threshold}$ and Q_{tep} are fixed. In the experimental results, the length of the TEP period changed from 3.4 to 5,600 ms as the values of the two parameters changed. Furthermore, the length of the TEP period was less

than 10 ms when the value of Q_{tep} was set to less than five for all the values of $n_{frame_threshold}$.

The estimated number of tags (i.e., n_{tag}) may differ from the number of tags actually deployed in the Wi-Fi backscatter system (i.e., n_{act_tag}). The greater the difference between them, the more empty or collided slots, resulting in a longer CP period. The graphs in Fig. 8 illustrate the variations in the estimation error according to the change of the values of Q_{tep} and n_{act_tag} when the value of $n_{frame_threshold}$ is fixed. The estimation error denotes the difference between n_{tag} and n_{act_tag} . In each graph, the curve shows that the estimation error varies depending on n_{act_tag} when the value of Q_{tep} is fixed. If the number of slots in a frame determined by Q_{tep} (i.e., $2^{Q_{tep}}$) is too large or too small compared to n_{act_tag} , the difference between the expected number of slots and the observed number of slots increases, thereby increasing the estimation error. In the figure, the color represents the level of the average estimation error for all values of n_{act_tag} when both $n_{frame_threshold}$ and Q_{tep} are fixed. The darker the color, the smaller the average estimation error. All graphs commonly show that the average estimation error tends to decrease as the value of Q_{tep} approaches four. In the estimation procedure for the number of tags, the helper repeatedly estimates the number of tags $n_{frame_threshold}$ times and averages the results to improve estimation accuracy. Therefore, the estimation error tends to decrease as the value of $n_{frame_threshold}$ increases. On average, for each $n_{frame_threshold}$ (i.e., 2, 3, 4, and 5), the estimation errors are 17.30, 17.28, 17.26, and 17.21, respectively. Furthermore, when Q_{tep} is four and $n_{frame_threshold}$ varies from two to five, the smallest average estimation errors are 2.125, 1.63, 1.63, and 1.88, respectively.

As depicted in Figs. 7 and 8, the value of Q_{tep} should be determined to be less than five. Therefore, in the subsequent experiments, we limited the value of Q_{tep} to the range of [0, 5]. The length of the CP period changes as the value of n_{act_tag} varies. In this context, we clarify the relationship between the two parameters (i.e., Q_{tep} and $n_{frame_threshold}$) and the length of the CP period by averaging the lengths of CP period for each value of n_{act_tag} varying from ten to eighty.

Fig. 9 illustrates the variations in the length of the CP period as the values of Q_{tep} and $n_{frame_threshold}$ change. The length of the CP period tends to be shorter due to the decrease of the estimation error when the values of Q_{tep} and $n_{frame_threshold}$ are close to four and five, respectively. In the experiment, the shortest length for the CP period was approximately 150 ms, on average. Moreover, the length of the CP period increases sharply when the value of Q_{tep} is zero because collisions frequently occur due to large estimation error. In other words, in this case, the frame size in the CP period becomes too small compared with the number of tags, and as a result, the number of collided slots increases significantly. Accordingly, the number of frames required to detect all tag information increases, resulting in a longer CP period.

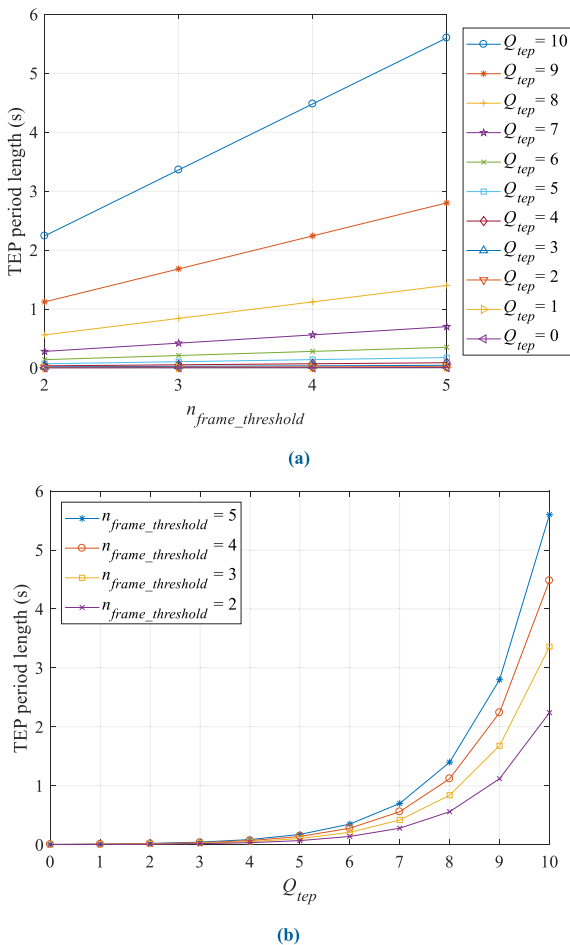


FIGURE 7. TEP period length (a) for varying values of $n_{frame_threshold}$ and (b) for varying values of Q_{tep} .

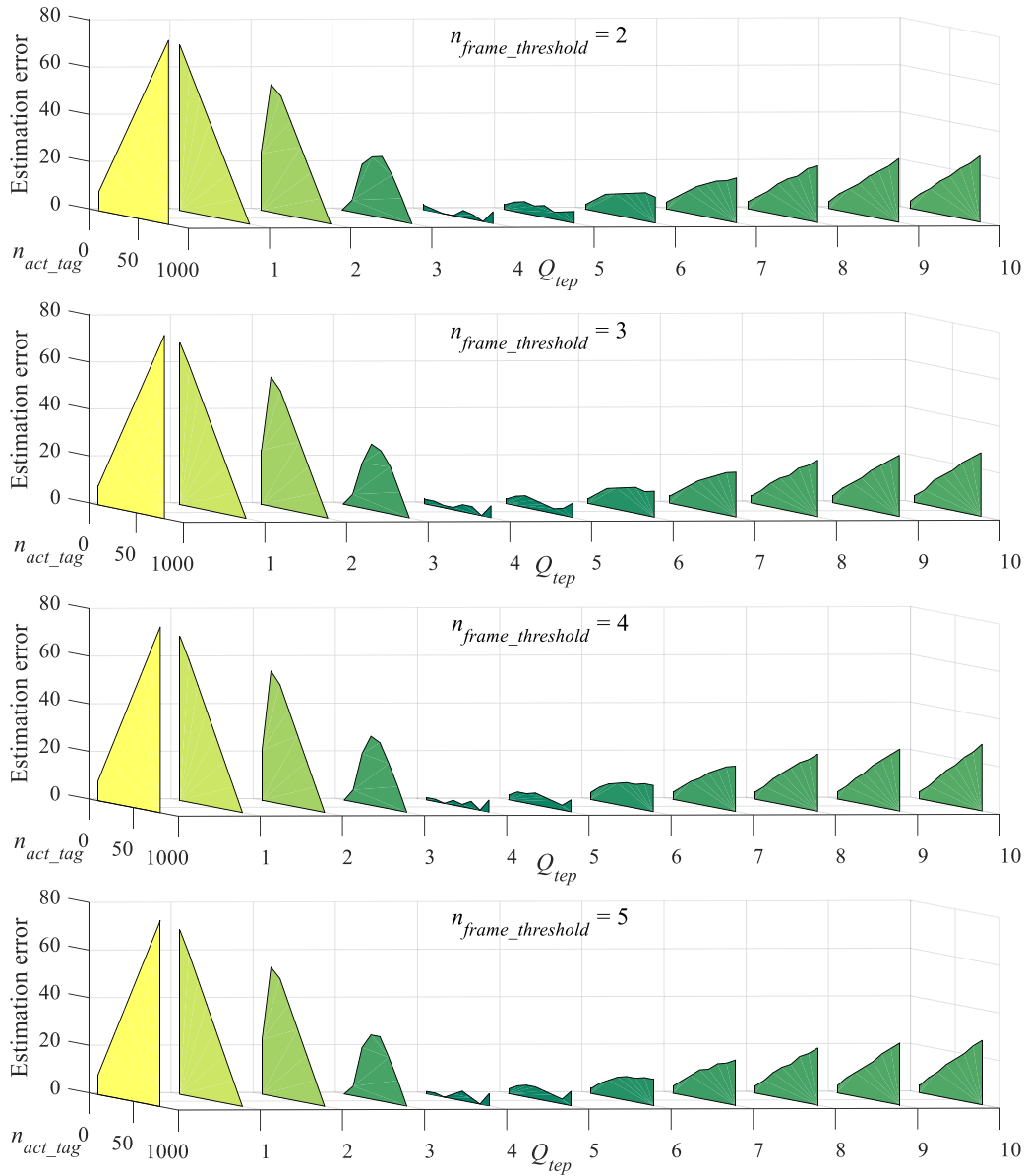


FIGURE 8. Estimation error for the number of tags.

The length of TEP period is proportional to the frame size (i.e., $2Q_{tep}$) and the number of frames in TEP period (i.e., $n_{frame_threshold}$). Meanwhile, the length of CP period is inversely proportional to the estimation error, which decreases as the value of Q_{tep} approaches four and the value of $n_{frame_threshold}$ increases. Therefore, in SUMA, the values of Q_{tep} and $n_{frame_threshold}$ that minimize the sum of the lengths of the TEP and CP periods are selected. Fig. 10 illustrates the sum of the lengths of the TEP and CP periods for various values of Q_{tep} and $n_{frame_threshold}$. In contrast to the experimental results for CP period length (Fig. 9), the sum of the lengths of the TEP and CP periods is minimized when the value of Q_{tep} is four and the value of $n_{frame_threshold}$ is two because the length of the TEP period increases when the values of Q_{tep}

and $n_{frame_threshold}$ increase, whereas that of the CP period increases when those values are too small. Quantitatively, the shortest sum of the lengths of the TEP and CP periods is approximately 200 ms, on average.

Figs. 11–14 illustrate the number of successful slots, the number of collided slots, the number of empty slots, and the total number of slots, respectively, when the Wi-Fi backscatter uplink communications are completed. In the simulation, the payload size of the tag information is set to 16 bits, and each legacy DFSA uses a different value of Δ in the range of [0.1, 0.5]. Both SUMA and the legacy DFSA approaches terminate the communications when the reader receives tag information from all tags deployed in the Wi-Fi backscatter system. Thus, in all cases, the number of successful slots

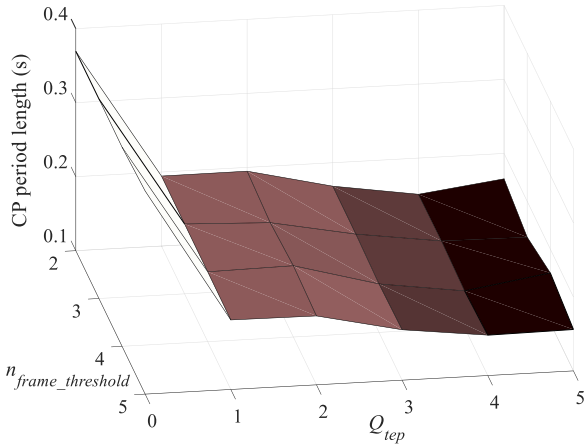


FIGURE 9. CP period length.

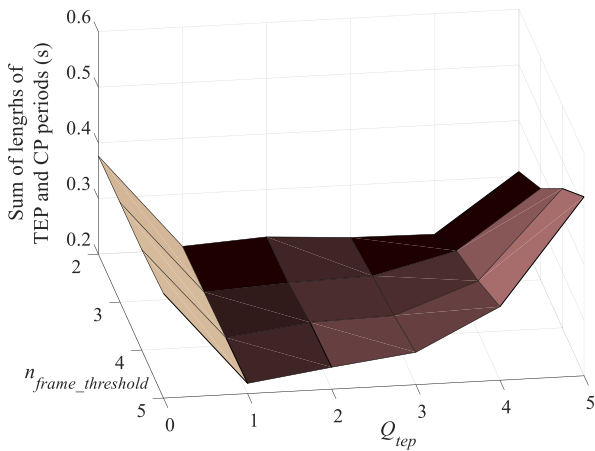


FIGURE 10. Sum of the lengths of TEP and CP periods.

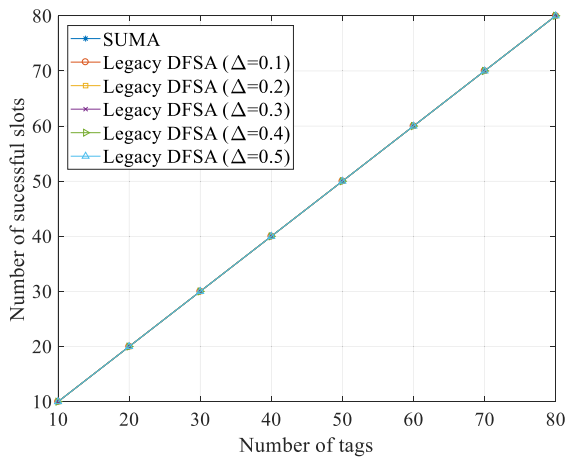


FIGURE 11. Number of successful slots (16-bit payload).

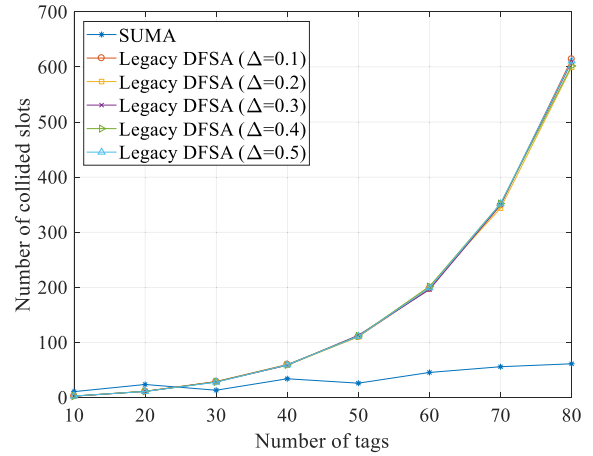


FIGURE 12. Number of collided slots (16-bit payload).

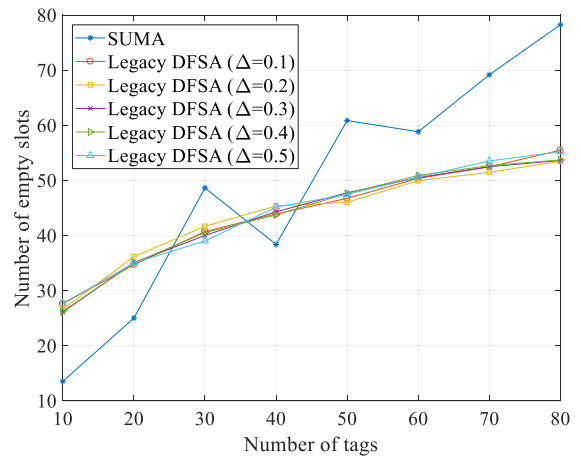


FIGURE 13. Number of empty slots (16-bit payload).

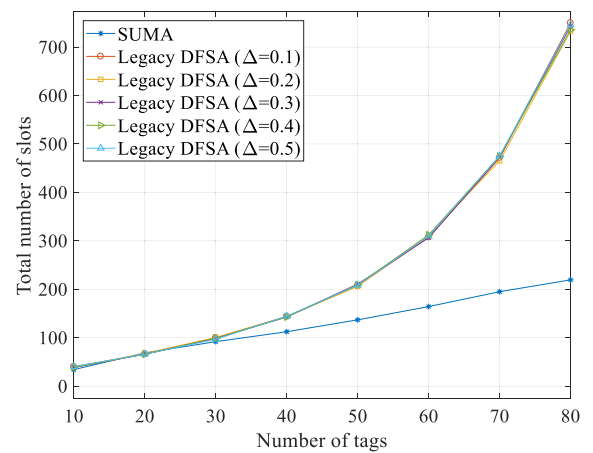


FIGURE 14. Total number of slots (16-bit payload).

equals the number of tags. However, as depicted in Fig. 12, SUMA exhibits a smaller number of collided slots compared with the legacy DFSA approaches because SUMA reduces the collision probability by adjusting the value of Q of each frame based on the number of remaining tags.

SUMA determines the value of Q for the first frame based on the number of tags, whereas the legacy DFSA uses a fixed value of Q (i.e., four) for the first frame. Therefore, if the number of tags is less than 30, the number of collided slots

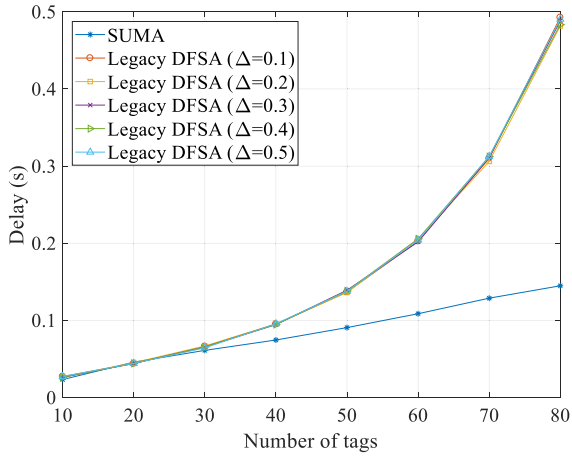


FIGURE 15. Delay (16-bit payload).

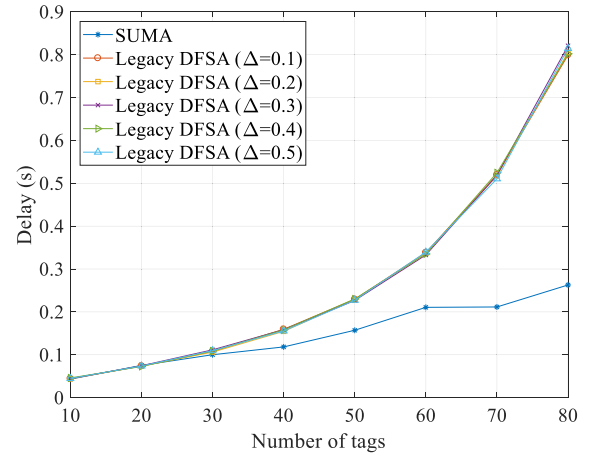


FIGURE 17. Delay (32-bit payload).

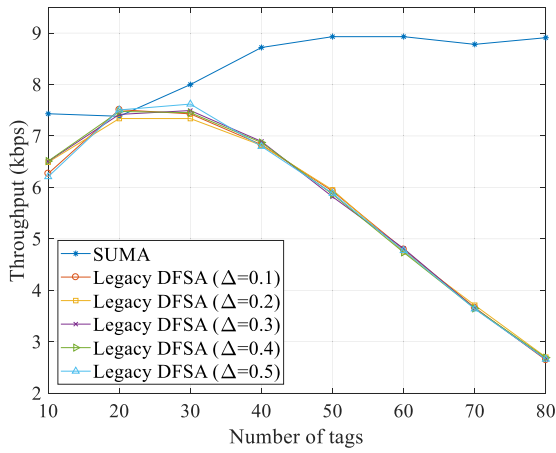


FIGURE 16. Throughput (16-bit payload).

of SUMA is slightly greater than that of the legacy DFSA approaches. The difference between the values of Δ in the legacy DFSA approaches is too small to change the frame size significantly. Thus, the number of collided slots in the legacy DFSA remains almost the same even if the value of Δ varies. In Fig. 13, the number of empty slots in SUMA is greater than that of the legacy DFSA, on average. However, as depicted in Fig. 14, the total number of slots used to detect all tag information of SUMA is less than that of the legacy DFSA because SUMA significantly reduces the number of collided slots by adjusting the value of Q . SUMA exhibits a shorter delay in detecting all tag information compared with the legacy DFSA.

Figs. 15 and 16 illustrate the delay and throughput, respectively, when the Wi-Fi backscatter uplink communications are completed. The payload of the tag information was set to 16 bits. The delay increases as the number of tags increases, due to the increase in the number of empty and collided slots. In SUMA, the helper determines the size of the first frame in CP period appropriate for the network size through tag estimation in TEP period. Nevertheless, due to the random slot

counter selection of the tags, the collision or empty slot may occur in subsequent frames, and thus, it continuously adjusts the value of Q to maximize throughput. Accordingly, the ratio of successful slots in a frame is maximized, and the time required to detect all tag information (i.e., delay) can be minimized. However, in legacy DFSA, regardless of the number of tags, a fixed first value of Q is used, and as a result, more empty or collided slots are highly likely to occur in the first frame compared with SUMA. Furthermore, even in the subsequent frames, the value of Q is adjusted considering only the number of empty and collided slots in the previous frame. Therefore, as the number of tags increases, a large number of frames are required to find an appropriate value of Q , resulting in a longer delay. In Fig. 15, SUMA exhibits a shorter delay than the legacy DFSA because it significantly reduces the number of collided slots. Specifically, the legacy DFSA suffers from a long delay when the number of tags is more than 30 because the difference in the number of collided slots between SUMA and the legacy DFSA increases rapidly as the number of slots increases. In the simulation, SUMA exhibits a 49.25% shorter delay than the legacy DFSA, on average. In terms of the throughput, SUMA outperforms the legacy DFSA because SUMA adjusts the value of Q for each frame to maximize the throughput. Quantitatively, as depicted in Fig. 16, SUMA exhibits 48.23% higher throughput than the legacy DFSA, on average. Note that SUMA maintains a TEP period for estimating the number of tags, and thus involves the initial overhead, which was considered in evaluating the delay and throughput. However, as the number of tags increases, the CP period length increases, and as a result, the impact of TEP period can be diminished when considering a long operation time of Wi-Fi backscatter system.

Figs. 17 and 18 illustrate the variation in the delay and throughput, respectively, when the payload size of tag information changes to 32 bits. The reader decodes one packet transmitted from the helper into one bit of tag information. Thus, if the payload size contained in the tag information increases by one bit, the helper should transmit an additional

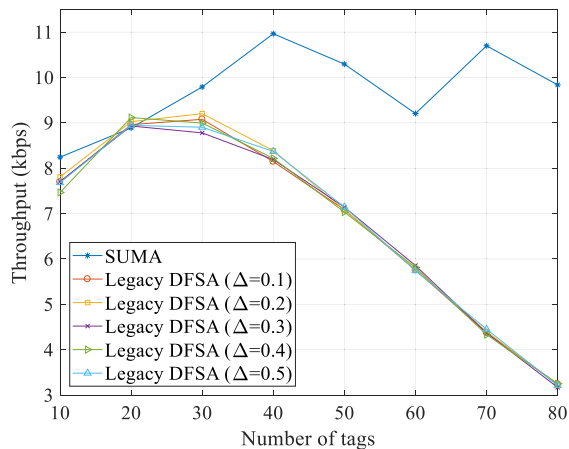


FIGURE 18. Throughput (32-bit payload).

packet—the slot size increases as the payload size increases. Accordingly, the delay in the 32-bit payload case is close to twice that of the 16-bit payload case. The reason why the increase in the delay is less than double is that the number of control packets used in each slot does not change even if the payload size changes. In the simulation, as the payload size increased from 16 to 32 bits, the delay of SUMA increased from 84.72 to 147.36 ms, on average. Moreover, due to the increase in delay, the throughput does not increase significantly even if the payload size increases; the SUMA throughput increased slightly from 8.38 to 9.74 kbps. Compared with the legacy DFSA approaches, SUMA exhibits a 51.89% shorter delay and 43.38% higher throughput, on average.

As depicted in Figs. 16 and 18, the throughput of SUMA is about 7.38–10.96 kbps. Specifically, the minimum and maximum throughputs were attained for ten tags from the result of Fig. 16 and forty tags from Fig. 18, respectively. This is certainly not high, but it is similar level to that of existing works (i.e., 1–10 kbps) for backscatter communication, such as Wi-Fi backscatter [10], ambient backscatter [36], and FM backscatter [11]. SUMA focuses on supporting the scalability of multiple access, which minimizes collisions caused by simultaneous uplink traffic from multiple tags, rather than the transmission rate for tag information detected by the receiver. In this context, in order to improve transmission efficiency, a decoding method that maps multi-bits according to RF signal strength can be considered, but this is out of scope in our work.

Regarding the computational complexity, the SUMA and legacy DFSA commonly require arithmetic operations to adjust the value of Q for each frame. Specifically, in SUMA, one initial tag estimation via Algorithm 1 and the Q adjustment via Algorithm 2 for each frame are performed. Meanwhile, in legacy DFSA, the Q -algorithm depicted in Fig. 1 is performed for each slot. When considering only the Wi-Fi backscatter uplink communications, the amount of resources required to run algorithms for SUMA and legacy DFSA can be similar because of their simple arithmetic operations.

However, SUMA has a higher computational burden due to the additional operations for tag estimation. Nevertheless, the simulation results show that SUMA can provide the scalability of multiple access through tag estimation and Q adjustment procedures, even at the cost of higher computational complexity.

V. CONCLUSION

In this paper, we presented SUMA, a reader-initiated DFSA-based multiple access protocol for bistatic Wi-Fi backscatter systems. SUMA focuses on supporting the scalability of multiple access to the network size (i.e., the number of tags), which is indispensable for enabling practical adoption of Wi-Fi backscatter communication. In this context, the main contributions of this paper can be summarized threefold: First, we devised a tag estimation procedure that improves the estimation accuracy for the number of tags in Wi-Fi backscatter system. Before the Wi-Fi backscatter uplink communications from multiple tags start, the number of tags in the system is repeatedly estimated several times, and the average of the estimated values is used to determine the initial frame size. Second, we devised a Q adjustment procedure to minimize collisions and maximize network performance. Once the Wi-Fi backscatter uplink communications start, the frame size is continuously adjusted based on the changed network size (i.e., the number of remaining tags to detect the tag information). At this time, the value of Q is adjusted in consideration of the throughput rather than the collided and empty slots. Finally, the impact of two key parameters, $n_{frame_threshold}$ and Q_{tep} affecting the performance of SUMA was investigated to achieve both short delay and high estimation accuracy for the number of tags. Through experiments, we determined these parameters to minimize the sum of the time required for tag estimation (i.e., TEP period length) and the time required for Wi-Fi backscatter uplink communications (i.e., CP period length). Furthermore, SUMA considers two auxiliary countermeasures for interference from nearby Wi-Fi devices; the transmission of *CTS-to-Self* in each slot and the use of *SIFS* interval between consecutive packets in a slot. The former forces the neighboring legacy Wi-Fi devices (i.e., Wi-Fi helper or Wi-Fi reader in other system) to defer their transmissions for a specific period, and the latter guarantees reliable uplink traffic within a slot.

We conducted an experimental simulation to evaluate the performance of SUMA. First, we experimentally analyzed the effects of two parameters, $n_{frame_threshold}$ and Q_{tep} , on the TEP and CP periods, and derived them as two and four, respectively. Then, based on these values, the performance of SUMA was evaluated and compared with that of the legacy DFSA approach in terms of the number of successful, empty, and collided slots, throughput, and delay. The results demonstrated that SUMA reduces the number of slots used to detect all tag information by reducing the number of collided slots significantly. Quantitatively, the number of collided slots of SUMA was 48.1% less than that of the legacy DFSA.

For payload sizes of 16 and 32 bits, SUMA obtained, on average, 48.23% and 43.38% higher throughput and 49.25% and 51.89% shorter delays, respectively, than the legacy DFSA approach.

ACRONYMS

AP	Access point
BEB	Binary exponential backoff
CCA	Clear channel assessment
CP	Communications period
CSMA/CA	Carrier sense multiple access with collision avoidance
CTS	Clear to send
CW	Contention window
DFSA	Dynamic framed slotted ALOHA
DIFS	Distributed coordination function inter-frame space
IoT	Internet of Things
MAC	Medium access control
NAV	Network allocation vector
RF	Radio frequency
RFID	Radio frequency identification
RP	Request period
RSSI	Received signal strength indicator
SIFS	Short inter-frame space
SUMA	Scalable uplink multiple access
TDMA	Time division multiple access
TEP	Tag estimation period
TP	Termination period
WLANs	Wireless local area networks
WPSNs	Wireless powered sensor networks

REFERENCES

- N. Van Huynh, D. T. Hoang, X. Lu, D. Niyato, P. Wang, and D. I. Kim, "Ambient backscatter communications: A contemporary survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2889–2922, 4th Quart., 2018.
- C. Xu, L. Yang, and P. Zhang, "Practical backscatter communication systems for battery-free Internet of Things: A tutorial and survey of recent research," *IEEE Signal Process. Mag.*, vol. 35, no. 5, pp. 16–27, Sep. 2018, doi: [10.1109/MSP.2018.2848361](https://doi.org/10.1109/MSP.2018.2848361).
- W.-S. Lee, C.-H. Kang, Y.-K. Moon, and H.-K. Song, "Determination scheme for detection thresholds using multiple antennas in Wi-Fi backscatter systems," *IEEE Access*, vol. 5, pp. 22159–22165, 2017, doi: [10.1109/ACCESS.2017.2759806](https://doi.org/10.1109/ACCESS.2017.2759806).
- C.-H. Kang, W.-S. Lee, Y.-H. You, and H.-K. Song, "Signal detection scheme in ambient backscatter system with multiple antennas," *IEEE Access*, vol. 5, pp. 14543–14547, 2017, doi: [10.1109/ACCESS.2017.2732948](https://doi.org/10.1109/ACCESS.2017.2732948).
- H. Hwang, J.-H. Lim, J.-H. Yun, and B. Jeong, "Pattern-based decoding for Wi-Fi backscatter communication of passive sensors," *Sensors*, vol. 19, no. 5, p. 1157, Mar. 2019, doi: [10.3390/s19051157](https://doi.org/10.3390/s19051157).
- T. Kim, H. Park, Y. Jung, and S. Lee, "Wi-Fi backscatter system with tag sensors using multi-antennas for increased data rate and reliability," *Sensors*, vol. 20, no. 5, p. 1314, Feb. 2020, doi: [10.3390/s20051314](https://doi.org/10.3390/s20051314).
- H. Hwang and J.-H. Yun, "Adaptive transmission repetition and combining in bistatic Wi-Fi backscatter communications," *IEEE Access*, vol. 8, pp. 55023–55031, 2020, doi: [10.1109/ACCESS.2020.2981868](https://doi.org/10.1109/ACCESS.2020.2981868).
- D. Bharadia, K. R. Joshi, M. Kotaru, and S. Katti, "BackFi: High throughput WiFi backscatter," *Comput. Commun. Rev.*, vol. 45, no. 4, pp. 283–296, Oct. 2015, doi: [10.1145/2829988.2787490](https://doi.org/10.1145/2829988.2787490).
- J.-H. Kwon, H.-H. Lee, Y. Lim, and E.-J. Kim, "Dominant channel occupancy for Wi-Fi backscatter uplink in industrial Internet of Things," *Appl. Sci.*, vol. 6, no. 12, p. 427, Dec. 2016, doi: [10.3390/app6120427](https://doi.org/10.3390/app6120427).
- B. Kellogg, A. Parks, S. Gollakota, J. R. Smith, and D. Wetherall, "Wi-Fi backscatter: Internet connectivity for RF-powered devices," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 607–618, Feb. 2015, doi: [10.1145/2740070.2626319](https://doi.org/10.1145/2740070.2626319).
- P. Zhang, C. Josephson, D. Bharadia, and S. Katti, "FreeRider: Backscatter communication using commodity radios," in *Proc. 13th Int. Conf. Emerg. Netw. EXperiments Technol.*, Nov. 2017, pp. 389–401, doi: [10.1145/3143361.3143374](https://doi.org/10.1145/3143361.3143374).
- Y. Li, L. Fu, Y. Ying, Y. Sun, K. Chi, and Y.-H. Zhu, "Goodput optimization via dynamic frame length and charging time adaptation for backscatter communication," *Peer Peer Netw. Appl.*, vol. 10, no. 3, pp. 440–452, May 2017, doi: [10.1007/s12083-016-0480-1](https://doi.org/10.1007/s12083-016-0480-1).
- EPC Radio-Frequency Identify Protocols Class-1 Generation-2 UHF RFID Protocol for Communications at 860 MHz-960 MHz Ver. 2.0.1*. Brussels, Belgium: GS1 EPCglobal Inc., Apr. 2015.
- W.-T. Chen, "A feasible and easy-to-implement anticollision algorithm for the EPCglobal UHF class-1 generation-2 RFID protocol," *IEEE Trans. Autom. Sci. Eng.*, vol. 11, no. 2, pp. 485–491, Apr. 2014, doi: [10.1109/TASE.2013.2257756](https://doi.org/10.1109/TASE.2013.2257756).
- L. Zhu and T.-S. P. Yum, "The optimal reading strategy for EPC gen-2 RFID anti-collision systems," *IEEE Trans. Commun.*, vol. 58, no. 9, pp. 2725–2733, Sep. 2010, doi: [10.1109/TCOMM.2010.080310.090421](https://doi.org/10.1109/TCOMM.2010.080310.090421).
- Z. Ma, L. Feng, and F. Xu, "Design and analysis of a distributed and demand-based backscatter MAC protocol for Internet of Things networks," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 1246–1256, Feb. 2019, doi: [10.1109/JIOT.2018.2869015](https://doi.org/10.1109/JIOT.2018.2869015).
- G. Yang, D. Yuan, Y.-C. Liang, R. Zhang, and V. C. M. Leung, "Optimal resource allocation in full-duplex ambient backscatter communication networks for wireless-powered IoT," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2612–2625, Apr. 2019, doi: [10.1109/JIOT.2018.2872515](https://doi.org/10.1109/JIOT.2018.2872515).
- G. Khandelwal, K. Lee, A. Yener, and S. Serbetli, "ASAP: A MAC protocol for dense and time-constrained RFID systems," *EURASIP J. Wireless Commun. Netw.*, vol. 2007, no. 1, p. 18730, Dec. 2007, doi: [10.1155/2007/18730](https://doi.org/10.1155/2007/18730).
- IEEE Standard for Information Technology-Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks-Specific Requirements Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2012, (Revision of IEEE Std. 802.11-2007), 2012.
- IEEE Standard for Low-Rate Wireless Personal Area Networks (LR-WPANs)*, IEEE Standard 802.15.4-2015, (Revision of IEEE Std. 802.15.4-2011), 2015.
- E.-J. Kim, S. Youm, and C.-H. Kang, "Power-controlled topology optimization and channel assignment for hybrid MAC in wireless sensor networks," *IEICE Trans. Commun.*, vol. E94-B, no. 9, pp. 2461–2472, 2011, doi: [10.1587/transcom.E94.B.2461](https://doi.org/10.1587/transcom.E94.B.2461).
- J.-H. Kwon, S.-B. Lee, and E.-J. Kim, "Group-based concurrent transmissions for spatial efficiency in IEEE 802.15.7 visible light communications," *Appl. Math. Model.*, vol. 53, pp. 709–721, Jan. 2018, doi: [10.1016/j.apm.2017.08.033](https://doi.org/10.1016/j.apm.2017.08.033).
- J.-H. Kwon and E.-J. Kim, "Adaptive multi-channel allocation for vehicular infrastructure mesh systems," *Multimedia Tools Appl.*, vol. 74, no. 5, pp. 1593–1609, Mar. 2015, doi: [10.1007/s11042-013-1752-x](https://doi.org/10.1007/s11042-013-1752-x).
- M. Y. Naderi, P. Nintanavongsa, and K. R. Chowdhury, "RF-MAC: A medium access control protocol for re-chargeable sensor networks powered by wireless energy harvesting," *IEEE Trans. Wireless Commun.*, vol. 13, no. 7, pp. 3926–3937, Jul. 2014, doi: [10.1109/TWC.2014.2315211](https://doi.org/10.1109/TWC.2014.2315211).
- T. Kim, J. Park, J. Kim, J. Noh, and S. Cho, "REACH: An efficient MAC protocol for RF energy harvesting in wireless sensor network," *Wireless Commun. Mobile Comput.*, vol. 2017, pp. 6438726:1-6438726:8, Sep. 2017, doi: [10.1155/2017/6438726](https://doi.org/10.1155/2017/6438726).
- T. Ha, J. Kim, and J.-M. Chung, "HE-MAC: Harvest-then-transmit based modified EDCF MAC protocol for wireless powered sensor networks," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 3–16, Jan. 2018, doi: [10.1109/TWC.2017.2757024](https://doi.org/10.1109/TWC.2017.2757024).
- Y. Zhao, J. Hu, Y. Diao, Q. Yu, and K. Yang, "Modelling and performance analysis of wireless LAN enabled by RF energy transfer," *IEEE Trans. Commun.*, vol. 66, no. 11, pp. 5756–5772, Nov. 2018, doi: [10.1109/TCOMM.2018.2848974](https://doi.org/10.1109/TCOMM.2018.2848974).

- [28] Y. Zhao, J. Hu, K. Yang, and S. Cui, "Deep reinforcement learning aided intelligent access control in energy harvesting based WLAN," *IEEE Trans. Veh. Technol.*, vol. 69, no. 11, pp. 14078–14082, Nov. 2020, doi: [10.1109/TVT.2020.3019687](https://doi.org/10.1109/TVT.2020.3019687).
- [29] B. Ji, B. Xing, K. Song, C. Li, H. Wen, and L. Yang, "The efficient BackFi transmission design in ambient backscatter communication systems for IoT," *IEEE Access*, vol. 7, pp. 31397–31408, 2019, doi: [10.1109/ACCESS.2019.2899001](https://doi.org/10.1109/ACCESS.2019.2899001).
- [30] I. Filho, I. Silva, and C. Viegas, "An effective extension of anti-collision protocol for RFID in the industrial Internet of Things (IIoT)," *Sensors*, vol. 18, no. 12, p. 4426, Dec. 2018, doi: [10.3390/s18124426](https://doi.org/10.3390/s18124426).
- [31] X. Fan, "Gen2-based tag anti-collision algorithms using Chebyshev's inequality and adjustable frame size," *ETRI J.*, vol. 30, no. 5, pp. 653–662, Oct. 2008, doi: [10.4218/etrij.08.1308.0098](https://doi.org/10.4218/etrij.08.1308.0098).
- [32] P. Šolić, J. Radić, and N. Rožić, "Energy efficient tag estimation method for ALOHA-based RFID systems," *IEEE Sensors J.*, vol. 14, no. 10, pp. 3637–3647, Oct. 2014, doi: [10.1109/JSEN.2014.2330418](https://doi.org/10.1109/JSEN.2014.2330418).
- [33] Y.-C. Lai and L.-Y. Hsiao, "General binary tree protocol for coping with the capture effect in RFID tag identification," *IEEE Commun. Lett.*, vol. 14, no. 3, pp. 208–210, Mar. 2010, doi: [10.1109/LCOMM.2010.03.092208](https://doi.org/10.1109/LCOMM.2010.03.092208).
- [34] J. Vales-Alonso, V. Bueno-Delgado, E. Egea-Lopez, F. J. Gonzalez-Castano, and J. Alcaraz, "Multiframe maximum-likelihood tag estimation for RFID anticollision protocols," *IEEE Trans. Ind. Informat.*, vol. 7, no. 3, pp. 487–496, Aug. 2011, doi: [10.1109/TII.2011.2158831](https://doi.org/10.1109/TII.2011.2158831).
- [35] Y. Wang, H. Wu, and Y. Zeng, "Capture-aware estimation for large-scale RFID tags identification," *IEEE Signal Process. Lett.*, vol. 22, no. 9, pp. 1274–1277, Sep. 2015, doi: [10.1109/LSP.2015.2396911](https://doi.org/10.1109/LSP.2015.2396911).
- [36] V. Liu, A. Parks, V. Talla, S. Gollakota, D. Wetherall, and J. R. Smith, "Ambient backscatter: Wireless communication out of thin air," in *Proc. ACM SIGCOMM Conf. SIGCOMM*, Aug. 2013, pp. 39–50, doi: [10.1145/2486001.2486015](https://doi.org/10.1145/2486001.2486015).



XUE ZHANG received the B.S., M.S., and Ph.D. degrees in electronic engineering from Hallym University, Chuncheon, South Korea, in 2013, 2015, and 2018, respectively. From January 2019 to August 2019, she worked with the College of Mechanical and Electronic Engineering, Shandong University of Science and Technology, Qingdao, China. Since September 2019, she has been a Lecturer with the College of Ocean Science and Engineering, Shandong University of Science and Technology. Her current research interests include ocean sensors and semiconductor devices.



JUNG-HYOK KWON (Member, IEEE) received the B.S. degree in electronic engineering from Soongsil University, Seoul, South Korea, in 2010, the M.S. degree in electronics and computer engineering from Korea University, Seoul, in 2012, and the Ph.D. degree in convergence software from Hallym University, Chuncheon, South Korea, in 2019. From September 2012 to March 2013, he was with the Electrical and Electronic Engineering Research Institute (EERI), Korea University. From April 2013 to June 2015, he was with the Software Research and Development Laboratory, LIG Nex1 Company Ltd., Seongnam, South Korea. Since March 2019, he has been a Research Professor with the Smart Computing Laboratory, Hallym University. His research interests include design and performance analysis of medium access control protocols for next-generation communication systems, vehicular communications (V2X), machine-to-machine communications, machine learning, wireless powered communications, and the Internet of Things. He was selected as a recipient of the Global Ph.D. Fellowship Program sponsored by the National Research Foundation of Korea, in 2016.



EUI-JIK KIM (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in electronics and computer engineering from Korea University, Seoul, South Korea, in 2004, 2006, and 2013, respectively. From September 2005 to December 2005, he was with the Intel Korea Research and Development Center, Intel Corporation, Seoul. From February 2006 to July 2009, he was with the DMC Research and Development Center, Samsung Electronics Company Ltd., Suwon, South Korea. From August 2009 to August 2013, he was with the Advanced Institute of Technology, KT Corporation, Seoul. Since September 2013, he has been an Associate Professor with the School of Software, Hallym University, Chuncheon, South Korea. His current research interests include wireless/mobile networks with an emphasis on QoS guarantee and adaptation, wireless sensor networks, energy harvesting, wireless powered communication networks, and the Internet of Things.

...