DualBlock: Adaptive Intra-Slot CSMA/CA for TSCH

JEONGJUN PARK\textsuperscript{1}, (Student Member, IEEE), HONGCHAN KIM\textsuperscript{1}, (Student Member, IEEE), HYUNG-SIN KIM\textsuperscript{2}, (Member, IEEE), SAEWOONG BAHK\textsuperscript{1}, (Senior Member, IEEE),
\textsuperscript{1}Department of ECE and INMC, Seoul National University, Seoul, Republic of Korea
\textsuperscript{2}Graduate School of Data Science, Seoul National University, Seoul, Republic of Korea

Corresponding authors: Hyung-Sin Kim (hyungkim@snu.ac.kr) and Saewoong Bahk (sbahk@snu.ac.kr).

This research was supported by a grant to Bio-Mimetic Robot Research Center Funded by Defense Acquisition Program Administration, and by Agency for Defense Development (UD190018ID).

ABSTRACT IEEE 802.15.4 Time-Slotted Channel Hopping (TSCH) has drawn significant attention as a low-power network solution for the Internet of Things (IoT). To make the TSCH scalable and robust, slot scheduling is an important issue that needs to be addressed. The research community has made significant strides with various autonomous or decentralized technologies in recent years. While these techniques try to provide non-overlapped slots for each link, they take collision for granted when multiple links utilize the same slot. In this paper, we challenge the perspective, investigate what happens within a TSCH slot, and find room for performance improvement even when multiple links unfortunately share a slot. To this end, we propose DualBlock that provides another chance for collision avoidance when multiple links try to utilize a slot at the same time by enabling clear channel assessment (CCA) and random backoff within the congested slot. In addition, given that the intra-slot backoff consumes more energy, a control mechanism is added that adjusts maximum backoff by monitoring network congestion level. DualBlock operates in a distributed manner for scalability. Extensive experiments demonstrate that TSCH networks achieve significant performance improvement in many aspects when DualBlock is combined with a scheduler (i.e., Orchestra), up to 3.6 times higher packet delivery ratio and 75% less radio duty cycle.

INDEX TERMS IEEE 802.15.4, time slotted channel hopping (TSCH), collision mitigation

I. INTRODUCTION

For two decades, efficient MAC operation has been a long-loved research topic in the low-power and lossy network (LLN) community since it is a non-trivial challenge to enable a large number of nodes to send their packets while minimizing collision, coordination, and radio duty-cycle. Time-Slotted Channel Hopping (TSCH), standardized in IEEE 802.15.4 in 2015, was a major breakthrough in that low-power devices do not have to suffer idle-listening and wireless interference (e.g., Wi-Fi). On the flip side, it opened up timeslot scheduling as an interesting research issue: how to allocate independent (collision-free) slots for each directional link with modest coordination overhead? After an autonomous scheduler Orchestra was first introduced in 2015 with its open implementation [1], the community has made significant progress in this regime, e.g., transition from node scheduling to link scheduling [2] and from fixed scheduling to traffic-aware adaptive scheduling [3], [4], [5].

Challenges. Although various techniques try to provide collision-free slots for each directional link, they operate in a best-effort way due to the strict resource constraint in LLNs. In other words, despite the use of state-of-the-art schedulers, there are always unfortunate cases when multiple links happen to transmit their packets in a timeslot. Then the multiple senders cause nothing but collision because they are time-synchronized and send packets at the start of the same slot simultaneously, not able to sense each other’s transmission attempts in advance (e.g., clear channel assessment (CCA)). Furthermore, undesirable slot overlap is not trivial since traffic demand in LLNs is growing as IoT applications are diversified and combined with the global megatrend of artificial intelligence (AI) [6], [7], [8].

To the best of our knowledge, this problem has relatively been overlooked in TSCH research and even the state-of-the-art TSCH implementation in Contiki takes for granted this kind of collision. Nodes that experience a packet collision in a slot hastily retry in the same slot of the next slotframe together, resulting in another collision. Given recent achieve-
ments in slot scheduling research that significantly distributes transmissions over different slots, we believe it is time to pay more attention to what happens within a slot, stepping forward to support more diverse IoT and AI applications using low-power networks.

**Approach.** In this paper, we propose **DualBlock**, an adaptive intra-slot CSMA/CA mechanism that mitigates packet collision in TSCH networks when multiple senders try to use the same slot. To this end, we carefully consider time operation of embedded devices, including CCA, software execution, and radio mode transition. Then we devise a backoff period that is just enough for CCA to detect on-going transmissions (~700 us). In addition, we apply a linear (non-exponential) backoff mechanism to minimize timeslot expansion due to the use of intra-slot backoff, resulting in 2.1 ms longer slot period compared to Contiki implementation.

Although employing the intra-slot CSMA/CA alleviates congestion, it also has a downside: a receiver should turn on its radio for a longer period since it cannot confirm whether a timeslot is idle or has a packet to receive until the end of its last intra-slot backoff period. This leads to inefficient energy consumption especially when collisions rarely occur. To improve energy efficiency, we add another mechanism at receivers that measures network congestion level and adaptively controls the maximum number of intra-slot backoffs accordingly. A transmitter’s intra-slot backoff is bounded by the maximum backoff value that its corresponding receiver determines.

**Contributions.** We summarize the contributions of this work as follows.

- We propose a novel method **DualBlock**, the first attempt for adaptive intra-slot CSMA/CA in TSCH networks. Importantly, **DualBlock** is orthogonal to slot scheduling techniques, creating synergy with any existing TSCH scheduler. We implement **DualBlock** in Contiki OS [9] and open the source code.

- To minimize energy consumption at receivers by adapting maximum intra-slot backoff, we propose two metrics that measure collision level: cyclic redundancy check (CRC)-based and retransmission count-based ones.

- We extensively evaluate **DualBlock** on a topology of 32 nodes in FIT/IoT-LAB [10], a public testbed for LLNs. When combined with a slot scheduler (i.e., Orchestra [1]), **DualBlock** shows significant performance improvement, 3.6 times higher packet delivery ratio and 75% less radio duty cycle, despite a slightly longer slot period.

**II. BACKGROUND: IEEE 802.15.4 TSCH**

TSCH, a link-layer protocol standardized in IEEE 802.15.4-2015 standard [11], is one of the representative time-synchronized medium access control (MAC) protocols for LLNs. Its time synchronization contributes to reducing redundant transmissions for rendezvous of Tx and Rx nodes compared to asynchronous MACs [12], [13], [14], resulting in reduced energy consumption. Furthermore, TSCH is robust to external interference and multi-path fading because it uses a channel hopping mechanism.

**A. TIME SYNCHRONIZATION (TS)**

A timeslot is the basic time unit for communication in TSCH, long enough to exchange a single frame and enhanced acknowledgement (EACK). A TSCH network assigns an absolute slot number (ASN) to each timeslot to represent elapsed time. A larger time unit called **slotframe** consists of $S_{sf}$ consecutive timeslots. A timeslot’s relative position within the slotframe, called time offset $T_{offset}$, is calculated as

$$T_{offset} = ASN \mod S_{sf}. \quad (1)$$

For timeslot-based communication, each node should have a time source node and synchronize with the node’s time information that is included in its enhanced beacons (EBs) and EACKs.

**B. CHANNEL HOPPING (CH)**

The channel offset $C_{offset}$ represents an offset for channel hopping. The channel allocated for packet exchange is determined by the timeslot’s ASN and channel offset, as

$$Channel = \text{List}_c[(ASN + C_{offset}) \mod N_{List_c}] \quad (2)$$

where $\text{List}_c$ is a list of channel candidates for channel hopping and $N_{List_c}$ is the number of channel candidates. Fig. 1 shows an example in which nodes $B$, $C$, and $D$ send packets to node $A$. As time goes by (i.e., ASN increases), each node transmits data over a different channel at each slotframe. Lastly, a **cell** stands for a tuple of specific timeslot and channel, which is the basic scheduling unit.

---

1After acceptance.

---

**FIGURE 1.** An example of a tree topology for data collection and TSCH scheduling.
C. INTRA-SLOT OPERATION.

TSCH scheduling assigns an activity of either ‘transmit’, ‘receive’, or ‘sleep’ to each cell. Fig. 1 shows an example of a two-dimensional TSCH slotframe where four nodes’ activities are scheduled without collision. When a cell is scheduled for a packet transmission (e.g., cell C→A), the sender (node C) first wakes up and performs clear channel assessment (CCA) to make sure if there is no interference. If the CCA reports “idle,” the sender sends a packet, receives an EACK, and sleeps again. Given that the sender needs some preparation time before actual packet transmission, the receiver waits for additional time called Rx offset before receiving a packet. Otherwise (i.e., CCA reports “busy”), the sender does not send the packet and sleeps to avoid the collision.

D. SLOT COLLISION.

Given that IEEE 802.15.4-2015 does not specify TSCH scheduling mechanism, a number of schedulers have been developed in research community. These schedulers try to prevent multiple links from sharing the same cell for packet transmission, called slot collision. For example, Orchestra [1] allocates a unique cell in a slotframe for each node by hashing node ID, and utilizes parent-child relationship given by the routing layer for matching a sender-receiver pair. As a node ID-based approach, however, Orchestra (in case of receiver-based scheduling option) allows multiple senders (nodes B, C, and D) to transmit packets to the same receiver (node A) in a single cell, as shown in Fig. 2. Once slot collision occurs, the senders experience transmission failures and need to retransmit the same packet. Other state-of-the-art schedulers [2, 3, 4, 5] cannot completely block the slot collision either, given that they operate in a best-effort way due to the strict complexity constraint in LLNs.

III. MOTIVATION AND PRELIMINARY

As introduced in Section II, slot collision is inevitable in TSCH networks, no matter what scheduler is used. In this section, we quantify the impact of collision on TSCH network performance and identify why a slot collision causes a transmission failure even with the use of CCA.
defined as the ratio of the number of successfully avoided collisions via CCA to the number of collisions occurred at the link layer. Note that TSCH utilizes CCA-based collision avoidance, as shown in Fig. 1.

Fig. 3(d) shows that the CAR is extremely low in all IPI settings. With the low CAR, a TSCH network experiences performance degradation whenever multiple senders happen to use the same timeslot for packet transmission, even when traffic is not too heavy. The results represent that CCA does not work properly for collision avoidance in the TSCH network.

B. WHAT’S WRONG WITH CCA?

Why is TSCH so vulnerable to collisions even though CCA is performed before each packet transmission? Fig. 4 gives the answer. First, Fig. 4(a) shows a typical case when CCA successfully detects an on-going transmission. Before the advent of TSCH, asynchronous MAC protocols were widely used for LLNs [12], [13], [14]. Since each sender operates independently in these asynchronous networks, CCA works in many cases unless it is unfortunately performed during other senders’ radio turnaround time or ACK waiting time.

In TSCH, however, every node is synchronized and operates based on timeslot. Fig. 4(b) shows a downside of the synchronized operation. When a sender is performing CCA, all other senders are also performing CCA. Given that none of them detect each other during the CCA period, collision is unavoidable. CCA is still useful for detecting external interference but not for avoiding internal collisions.

The preliminary study motivates us to look into intra-slot operation and devise DualBlock, an intra-slot CSMA/CA mechanism that can be combined with any slot scheduler synergistically.

IV. DESIGN REQUIREMENTS

Before describing DualBlock, we provide its design requirements to mitigate collisions in TSCH networks.

- **DualBlock should make transmissions in a timeslot temporarily asynchronous.** The collision problem in TSCH network arises because all senders in a timeslot start to transmit simultaneously. To make CCA work, a sender should be able to defer its transmission randomly within a timeslot: intra-slot random backoff.

  - **DualBlock should maintain time synchronization despite temporarily asynchronous transmissions.** With intra-slot random backoff, a receiver mis-recognizes the intra-slot backoff as part of clock drift. To mitigate the confusion, DualBlock should explicitly consider each intra-slot random backoff when compensating for clock drift.

  - **DualBlock should not sacrifice energy efficiency since it runs on resource-constrained devices.** With intra-slot random backoff, a receiver should turn on its radio for a longer period since it does not know when to receive a packet. The larger maximum intra-slot backoff at a sender, the longer radio active period at a receiver. To mitigate the additional energy consumption, maximum intra-slot backoff should be adaptively controlled, just enough to handle the current level of collision.

V. DUALBLOCK DESIGN

In this section, we describe DualBlock that aims to mitigate collisions in a TSCH timeslot. As an intra-slot mechanism, DualBlock can be combined with any timeslot scheduler.

A. OVERVIEW

As shown in Fig. 5, DualBlock consists of functional parts of Tx and Rx nodes. Before transmission, each Tx node randomly selects an intra-slot random backoff (IBO) value to spread transmissions within the timeslot, which allows it to sense other nodes’ transmissions by CCA. Then, it sends a packet after the IBO delay and receives the clock drift information from the receiver via an EACK. By using both the timing information and IBO delay, each node fixes its clock drift.

However, the IBO operation causes the Rx node to turn on the radio longer, resulting in more energy consumption. Therefore we add receiver-triggered adaptation of IBO window size according to the collision level. To estimate the collision level, we consider two measurement methods: cyclic redundancy check (CRC)-based measurement and collision-free probability (CFP)-based measurement, and name these as DualBlock-CRC and DualBlock-CFP, respectively. Here
CFP indicates a probability that the receiver receives packets without collision.

B. INTRA-SLOT RANDOM BACKOFF (IBO)

To enable intra-slot CSMA/CA operation, the basic unit of intra-slot backoff (IBO) should be set properly: time duration that one IBO requires. While the IBO unit should be as short as possible to minimize time wastage, it should also be long enough to perform CCA during an actual packet transmission. Fig. 4(b) shows an extreme example where IBO unit is nearly zero. Although sender B starts CCA slightly after sender A, the timing is weigh before sender A’s packet comes out to wireless channel, resulting in a collision avoidance failure.

Considering the trade-off, we design the IBO unit as shown in Fig. 6. When a sender wakes up at the start of a timeslot for a packet transmission, it goes through four types of time delay before sending the packet out of the radio: CCA offset, CCA period, radio turnaround time for changing radio mode from Rx (reception) to Tx (transmission), and miscellaneous delay for software execution. Considering the delay, we define the IBO unit, $\Delta t$, as 700 us.4

With the IBO unit, DualBlock operation is similar to CSMA/CA. Assuming that the maximum IBO value is $M$, a sender randomly selects its IBO value $k$ ($0 \leq k \leq M$) before each packet transmission. It performs CCA after waiting for the selected IBO period $k \times \Delta t$ and sends its packet only when the CCA returns “idle.” Otherwise, the sender gives up sending its packet in the current timeslot and waits for the next timeslot that is scheduled for its transmission, which depends on a scheduler. Given that IBO is merely for enabling CCA, a sender detecting “busy” channel does not increase maximum IBO, like exponential random backoff, but always chooses an IBO value in the same region: from 0 to $M$. How to distribute transmissions over multiple timeslots is a slot scheduling issue, which is orthogonal to the problem that we aim to tackle in this work.

Figure 7 shows an example of IBO-based CSMA/CA within a timeslot. Although three senders try to send their packet in the same timeslot, all of them select different IBO values. As a result, only sender B that chooses the smallest value (zero) sends a packet while other two senders give up using the timeslot since they detect sender B’s transmission.

Given that the default timeslot length (i.e., 10 ms) is just enough to contain a packet transmission without any backoff, slot length should be extended to utilize IBO. In our implementation, timeslot length is set to 12.1 ms, 2.1 ms longer than the default length, that allows up to four IBO units ($M = 3$). The slot length extension is a downside of adopting DualBlock since a longer timeslot causes fewer transmission opportunities on average. It should be evaluated if DualBlock’s positive effect (collision avoidance) overcomes its weakness (less transmission opportunities).

C. CLOCK DRIFT COMPENSATION

In LLNs, each resource-constrained node has a low-cost clock that does not provide a strictly consistent tick period. Although nodes are synchronized once, the random clock drift on each node is gradually accumulated as time goes by, hindering timeslot operation. To maintain synchronization, TSCH compensates for clock drifts as frequently as possible, not only through periodic enhanced beacons (EBs) but also EACKs. Given that DualBlock utilizes intra-slot backoff (IBO) for each transmission, the IBO period should be additionally considered in EACK-based clock drift compensation.

Fig. 8(a) depicts an example of EACK-based clock drift compensation in TSCH between a pair of receiver and transmitter. The receiver expects to start receiving a packet at time $t_0$ but the transmitter sends its packet at its own time, at $t_1$. Then the receiver calculates difference between the expected and actual reception time, $t_0 - t_1$, and includes the information when sending an EACK for the received packet. Upon EACK reception, the transmitter corrects its clock drift accordingly. Thus, TSCH compensates for clock drift between nodes on each packet transmission.

In DualBlock, however, the clock jitter ($t_0 - t_1$) measured...
by the receiver contains not only the clock drift but also the IBO delay of the transmitter, as shown in Fig. 8(b).

\[ t_0 - t_1 = \text{clock drift} - \text{IBO delay} \]

Given that the IBO delay is not a random drift but an intended delay, it should be removed from the EACK-based clock drift compensation. Since the receiver does not know how the IBO value that the transmitter used, it still includes \((t_0 - t_1)\) in the EACK. The transmitter, however, stores the IBO value it selected for the packet transmission and utilizes both \((t_0 - t_1)\) in the received EACK and the stored IBO information to calculate actual clock drift as

\[ \text{clock drift} = (t_0 - t_1) + \text{IBO delay} \]

D. TRADE-OFF: RECEIVER ENERGY CONSUMPTION

Although intra-slot CSMA/CA and clock drift compensation in \textit{DualBlock} alleviate the collision problem in a timeslot, it causes more energy consumption at receivers. As shown in Fig. 9, a \textit{DualBlock} receiver should turn on the radio for up to \(M \times \Delta t\) more than standard TSCH where \(M\) is the maximum IBO value. When there is a packet transmission or even contention in a timeslot, the additional energy consumption is worthwhile. When a sender picks up an IBO value smaller than \(M\), the receiver’s waiting time also becomes smaller than \(M \times \Delta t\). However, when there is no packet transmission in a timeslot, the receiver does nothing but waste more energy for \(M \times \Delta t\).

To maximize efficiency of \textit{DualBlock}, it needs to adjust how aggressively it utilizes intra-slot CSMA/CA, according to collision level. Although timeslot length is long enough for \(M\) IBO value, it is better not utilize IBO at all at a very light traffic load; a sender might choose its IBO value among a smaller window, between 0 and \(m\) where \(m \leq M\). As traffic load and collision level go up, IBO window size \((m)\) needs to be linearly increased to distribute transmission time and enable CCA-based collision avoidance.

E. RECEIVER-TRIGGERED, COLLISION-AWARE IBO WINDOW SIZE ADAPTATION

To adapt IBO window size \(m \leq M\) for a pair of sender and receiver, either of them should measure collision level, adjust IBO window size, and notify the other node of the updated value. If the sender takes the lead, a number of packets can be lost until the receiver gets the new IBO window information from the sender. In addition, given that many senders may access the same timeslot, if a sender self-controls its IBO window, the receiver needs to timely collect many different IBO window sizes from the senders and operate based on the worst value. Therefore the receiver is in a better position to control the IBO window and share it with all the senders.

When the receiver is aware of a non-trivial collision, it increases its IBO window size \(m\) by one (possible to increase until \(M\)) to reduce collisions and notifies the updated window size to transmitters. Inversely, when there is no collision, the receiver decreases its IBO window size by one and lets the transmitters know this, resulting in reduced radio duty cycle. The adaptive control of IBO window helps \textit{DualBlock} achieve a balance between transmission reliability and energy consumption.

1) Collision measurements

For collision estimation, we consider two approaches: cyclic redundancy check (CRC)-based and collision-free probability (CFP)-based measurements.

CRC-based measurement. The receiver gets the CRC information based on the received signal on packet reception. A broken CRC implies either a packet collision or link loss. Since link loss is rare compared to packet collisions in TSCH networks with the help of channel hopping and routing, we consider a CRC error as a collision error.

CFP-based measurement. This approach is inspired by CFP computation in [3]. The receiver calculates the CFP by using the traffic load of transmitters every time interval \(T_{CFP}\). Each transmitter sends the information of its backlogged queue size and the number of transmission trials to the receiver by piggybacking this in the IEEE 802.15.4 MAC
header. Specifically, each transmitter \( i \) calculates the traffic load as
\[
L(i, A) = Q(i, A) + T(i, A)
\]
and informs the receiver \( A \) of this. \( Q(i, A) \) is the currently backlogged queue size from node \( i \) to \( A \) which represents the current level of congestion from node \( i \) to \( A \), and \( T(i, A) \) denotes the number of transmission trials from node \( i \) to \( A \). Then the receiver \( A \) calculates the CFP for each node \( i \) as
\[
CFP(i, A) = \prod_{k \in N(A) \setminus \{i\}} \left(1 - \frac{L(k, A)}{W}\right)
\]
where \( N(A) \) is the set of neighbors of the receiver node \( A \), and \( W \) is the number of Rx slots of node \( A \) during \( T_{CFP} \). \( \frac{L(i, A)}{W} \) implies the probability\(^6 \) that node \( i \) accesses an Rx slot of node \( A \). Thus, the probability that the other nodes except \( i \) do not access node \( A \) becomes the probability that node \( i \) accesses \( A \) without collisions.

2) Collision-aware IBO window size adaptation
We introduce receiver-triggered IBO window size control for DualBlock. According to the collision measurement method, we control the IBO window size a bit differently.

**DualBlock-CRC.** In DualBlock-CRC, the receiver increases or decreases the IBO window size \( m \) according to Algorithm 1. When the receiver is aware of a broken CRC, it immediately decides to increase the window size to counteract the collision. Before increasing, it compares its current window \( m \) with \( M \) and increases \( m \) by one only when \( m < M \). The receiver piggybacks its updated window \( m \) on the IEEE 802.15.4 frame header and broadcasts it towards the transmitters through broadcasting message from link layer or upper layer. For guaranteeing reception of this broadcasting message, the receiver piggybacks this on the EACK too when exchanging unicast frames. After receiving the updated \( m \), each transmitter applies it to its IBO operation to reduce collisions. The window \( m \) is maintained until the receiver updates it further.

In contrast, decreasing \( m \) should be done with caution. This is because a smaller \( m \) can increase collisions and reduce reliability. To this end, the receiver decreases \( m \) only when it does not observe any collision during \( T_{CRC} \). The receiver notifies the decreased \( m \) to transmitters via broadcasting packet, and the transmitters confirm reception through unicast EACK. The receiver applies the decreased \( m \) after waiting for \( T_{guarantee} \) to ensure that the transmitters update the new \( m \) value.

**DualBlock-CFP.** The receiver in DualBlock-CFP controls the IBO window size according to Algorithm 2 every \( T_{CFP} \). The minimum CFP \( \minCFP \) is the CFP of a node with the lowest CFP value among neighbors \( (N(A)) \). The receiver calculates \( \minCFP \) every \( T_{CFP} \) and identifies severe collisions when \( \minCFP \leq T_{inc} \), where \( T_{inc} \) is the threshold for increasing IBO window. Then the receiver increases the IBO window size by one to counter the collision problem. On the other hand, when \( \minCFP \geq T_{dec} \), where \( T_{dec} \) is the threshold for decreasing IBO window, the receiver decreases the window size \( m \). Otherwise, when \( T_{inc} < \minCFP < T_{dec} \), \( m \) is maintained. We use double thresholds for stability. DualBlock-CFP uses the same way as DualBlock-CRC to exchange updated IBO window size between the receiver and transmitters.

### Algorithm 1: Adaptive IBO window size control in DualBlock-CRC (on the receiver)

**Initialize:**
- \( m \leftarrow 0 \);

**Procedure:**
- if \( \text{CRCbroken} \land m < M \) then
  - reset timer for \( T_{CRC} \);
  - \( m \leftarrow m + 1 \);
- else if Packet type is equal to broadcast then
  - piggyback \( m \) on frame header;
  - send packet;
- end if
- else if Packet type is equal to unicast then
  - piggyback \( m \) on EACK;
  - receive packet;
  - send EACK;
- end if
- else if timer expired for \( T_{CRC} \) \&\& !CRCbroken \&\& \( m > 0 \) then
  - if Packet type is equal to broadcast then
    - piggyback \((m - 1)\) on frame header;
    - send packet;
  - end if
  - else if Packet type is equal to unicast then
    - piggyback \((m - 1)\) on EACK;
    - receive packet;
    - send EACK;
  - end if
  - wait for \( T_{guarantee} \);
  - \( m \leftarrow m - 1 \);
- end if

### VI. PERFORMANCE EVALUATION

In this section, we evaluate DualBlock on a real-world testbed, and compare its performance with that of standard TSCH. We evaluate three types of DualBlock, DualBlock-Basic that uses the fixed IBO window size of \( M \) and DualBlock-CRC and DualBlock-CFP that adapt IBO window size with CRC-based and CFP-based collision measurements, respectively. We first examine how well DualBlock
Algorithm 2: Adaptive IBO window size control in DualBlock-CFP (on the receiver)

\[Initialize:\]
\[m \leftarrow 0;\]

\[Procedure:\]
\[minCFP \leftarrow \text{calculate\_minCFP}();\]
if \(minCFP \leq Th_{\text{inc}} \&\& m < M\) then
\[m \leftarrow m + 1;\]
if Packet type is equal to broadcast then
piggyback \(m\) on frame header;
send packet;
end if
else if Packet type is equal to unicast then
piggyback \(m\) on EACK;
receive packet;
send EACK;
end if
end if
else if \(minCFP \geq Th_{\text{dec}} \&\& m > 0\) then
if Packet type is equal to broadcast then
piggyback \((m - 1)\) on frame header;
send packet;
end if
else if Packet type is equal to unicast then
piggyback \((m - 1)\) on EACK;
receive packet;
send EACK;
end if
wait for \(T_{\text{guarantee}}\);
\[m \leftarrow m - 1;\]
end if

adapts to dynamic traffic environments and discuss its impact on performance. Then, we present the impact of parameter settings in DualBlock.

A. METHODOLOGY AND EXPERIMENT SETUP

We implement DualBlock on Contiki OS (version NG) and use a flexible multi-hop tree topology of 32 nodes on the FIT/IoT-LAB [10] in Strasbourg, France. The nodes are deployed according to the topology in Fig. 10. The nodes feature a 32-bit ARM Cortex-M3 microcontroller (STM32F103REY) and an AT86RF231 IEEE 802.15.4 radio chip. We set the size of the Tx queue to 8 as the default implementation in Contiki-NG. Each node communicates using four channels (number 15, 20, 25, and 26) with a Tx power of -17 dBm. We conduct each experiment for 30 minutes by changing the traffic offered. The experimental results are plotted with error bars having 95% confidence interval.

For DualBlock, we set \(\Delta t\) and the maximum IBO value to 700 \(\mu s\) (considering the software delay) and 3, respectively.

Therefore, the timeslot length should be 12,100 \(\mu s\) (about 12 \(ms\)) to accommodate Tx frames with maximum length. We use Contiki-RPL as the routing layer above TSCH layer. Lengths of enhanced beacon (EB), RPL shared slotframe, and unicast slotframe are set to 397, 31, and 17, respectively.

B. VERIFICATION OF COLLISION AVOIDANCE VIA CCA

We first evaluate DualBlock’s collision avoidance via CCA. We generate upward traffic with various IPIs and measure the collision avoidance ratio (CAR), defined as the ratio of the number of successful packet detections using CCA to the number of collisions.

Fig. 11 plots the CAR in TSCH and DualBlock-Basic/CRC/CFP. As already shown in Fig. 3(d), TSCH cannot successfully perform collision avoidance due to its synchronized operation. Although CAR of standard TSCH slowly increases with traffic load, it does not mean that TSCH works better when traffic load is high. With high traffic load, a TSCH node experiences many transmission failures and tries to change its parent node. With many backlogged packets, however, it hastily sends packets to the new parent node before being synchronized, resulting in more transmission failures and parent changes. Without receiving any packet from its parent nodes (time sources) for a while, the TSCH node
is finally desynchronized from the network due to severely accumulated clock drift. Therefore the slightly higher CAR simply shows TSCH’s disastrous operation under high traffic load.

In contrast, the CAR in DualBlock-Basic/CRC/CFP are greater than that in standard TSCH owing to the use of IBO operation. Since the CAR becomes higher when the IBO window size is larger, DualBlock-CRC/CFP show slightly lower CAR than DualBlock-Basic in all traffic loads.

C. IMPACT OF TRAFFIC LOAD

We investigate the impact of traffic load with varying average IPI. To verify the performance gain of DualBlock over TSCH, we evaluate several metrics.

1) End-to-end reliability

Fig. 12 represents the average end-to-end PDR. In low traffic, all the schemes have a similar average end-to-end PDR. Since collisions are rare in low traffic, standard TSCH and DualBlock-Basic/CRC/CFP can overcome the collision problem by using only retransmissions. While DualBlock-Basic always uses the maximum IBO value of $M (=3)$, most of nodes in DualBlock-CRC/CFP set their IBO values to zero due to low traffic, de-activating intra-slot CSMA/CA.

In high traffic, the PDR in standard TSCH drops to 25% due to severe collisions. However, DualBlock-Basic/CRC/CFP alleviate the collision problem by exploiting IBO operation. For the IPI of 5, the highest traffic load, DualBlock-CFP achieves $>3.6$ higher end-to-end PDR ($>55\%$) than standard TSCH.

![End-to-end packet delivery ratio](image1)

**FIGURE 12.** End-to-end packet delivery ratio.

2) Radio duty cycle

Fig. 13(a) shows the radio duty cycle for four schemes. Fig. 13(b) and 13(c) present the radio duty cycle for Tx and Rx of nodes, respectively. When a node transmits or receives a packet, it consumes 7.5 and 12.3 mA [15], respectively.

We first discuss Tx radio duty cycle in Fig. 13(b). The Tx duty cycle of each scheme increases with the traffic load. Standard TSCH cannot avoid other nodes’ transmissions through CCA, so it keeps transmitting without entering sleep mode even when a collision occurs. It shows the highest Tx duty cycle and a low CAR (<4.5% at IPI of 5) for high traffic. DualBlock-CFP increases IBO window size slower than DualBlock-Basic/CRC, resulting in slightly higher Tx duty cycle than DualBlock-Basic/CRC in heavy traffic. However, it is important to note that Rx duty cycle is generally much higher than Tx duty cycle, affecting the total duty cycle much more.

As shown in Fig. 13(c), the Rx duty cycle in standard TSCH sharply increases as the average IPI decreases (up to 15.6% Rx duty cycle). Standard TSCH suffers from severe collisions due to high traffic. If nodes cannot exchange data and EACK for several times due to collisions, lots of them may frequently experience parent changes by the routing layer. This disturbs synchronizing with the parent node and

![Radio duty cycle](image2)

**FIGURE 13.** Radio duty cycle.
causes some nodes to leave the TSCH network. The nodes which leave the network turn on the Rx radio and enter always-on mode to listen to the enhanced beacon for fast rejoining. Therefore, the Rx duty cycle in standard TSCH is overwhelmingly higher than that in the others.

DualBlock-Basic using the maximum IBO window size of $M$ has a larger Rx duty cycle than standard TSCH in low traffic. On the other hand, DualBlock-CFP/CRC adaptively controlling the IBO window size have an IBO window size nearly zero in low traffic, but they have the IBO window size similar to DualBlock-Basic in high traffic. In low traffic, DualBlock-CFP/CRC have a Rx duty cycle lower than standard TSCH since they have a timeslot longer than standard TSCH to accommodate packets with maximum IBO value. In high traffic, DualBlock-CFP/CRC have a duty cycle lower than DualBlock-Basic. This is because all the nodes in DualBlock-Basic always use the maximum IBO window size, whereas leaf nodes in DualBlock-CFP/CRC use small IBO values.

Furthermore, the Rx duty cycle in DualBlock-CFP is slightly smaller than that in DualBlock-CRC since it increases the IBO window size by referring to the CFP instead of responding to collisions directly. DualBlock-CFP always shows the lowest duty cycle in all traffic scenarios and reduces the duty cycle by about 75% compared to standard TSCH.

3) Link layer reliability

Fig. 14 shows the average PRR in the link layer. As the traffic increases, standard TSCH shows dramatically decreased PRR since it only relies on retransmissions to handle collisions. DualBlock-Basic/CRC/CFP outperform standard TSCH in all traffic scenarios owing to the use of intra-slot CSMA/CA.

We observe that DualBlock-Basic mostly shows higher PRR than DualBlock-CFP/CRC. However, it consumes much more energy to accept packets. With receiver-triggered IBO window size adaptation, DualBlock-CRC shows marginally higher PRR than DualBlock-CFP in high traffic. This is because DualBlock-CRC as an aggressive approach increases IBO window size whenever a received CRC is broken. On the other hand, DualBlock-CFP uses the CFP instead of responding to sudden collisions, overcoming the collision problem while maintaining energy efficiency.

In addition to PRR, we should consider energy efficiency too, especially in LLNs. DualBlock-Basic/CRC marginally improve the PRR at the expense of more energy consumption compared to DualBlock-CFP as shown in Fig. 12 and 13(a). Thus, we observe that DualBlock-CFP balances reliability and energy efficiency very well.

4) Backlogged queue size

Fig. 15 shows the average Tx queue’s backlog size when each node transmits a packet. The backlogged queue size in each scheme is close to 1 at IPI 10 (low traffic). As the traffic increases, the backlogged queue size in each scheme also grows. DualBlock has a smaller Tx queue’s backlog size than the standard TSCH. DualBlock can dequeue packets from the Tx queue by employing IBO operation eventually when multiple transmitters transmit packets on a TSCH shared cell. However, the standard TSCH has difficulty in dequeuing packets when multiple transmitters contend for a TSCH shared cell. More seriously, due to the desynchronization problem, many packets are dropped at the queue right away without any chance for transmission. In contrast, we observe that DualBlock reduces the Tx queue size compared to the standard TSCH by alleviating the collision problem.

5) Throughput

Fig. 16 shows the results of the average throughput comparison in the application layer. In the standard TSCH, the average throughput increases up to the IPI of 8 and then drops sharply due to severe collisions in high traffic. DualBlock-Basic/CRC/CFP achieve better throughput than the standard TSCH. They show increased throughput up to IPI 7 and maintain high end-to-end PDR of 93%. At IPI 7 and higher, they show slightly decreased throughput due to reduced number of received packets in high traffic. They can avoid severe throughput degradation owing to IBO operation.

Even if the timeslot length in DualBlock becomes longer than that in the standard TSCH, DualBlock improves throughput since its gain due to collision-mitigating transmissions overwhelms the negative impact of resource reduction.
We investigate the impact of DualBlock on network performance in low-power and lossy networks (LLNs). TSCH and RPL [16] are the link layer and the routing layer protocol for LLN, respectively. We verify how DualBlock affects TSCH and RPL. To certify its impact on TSCH, we measure the total number of nodes’ leaving the TSCH network. We also measure the average number of routing path changes per node to confirm the impact on the RPL routing layer. We verify that DualBlock improves the stability of LLN.

In RPL, the most widely used objective function for routing is minimum rank with hysteresis objective function (MRHOF). To calculate the rank, RPL measures the expected transmission count (ETX) for each communication as the link quality metric. The accumulated ETX from a node to the root node is the rank of the node. In accordance with increasing ETX due to the degradation of PRR, the rank of the node increases. If the node’s rank becomes larger than its hysteresis, it changes its routing path to another node with a low rank. If the PRR of the entire network is poor, lots of nodes change their routing path frequently, resulting in large control overhead and latency.

Fig. 17(a) shows the average number of routing path changes. As the traffic increases, the PRR decreases, leading to routing path changes by RPL. Compared to standard TSCH, our schemes, DualBlock-Basic/CRC/CFP, experience fewer routing path changes owing to high PRR achieved by IBO operation.

When a node detects the poor link quality due to the collisions, it tries to change its parent node. After that, the node hastily transmits the packets to the new parent node before it synchronizes with the new parent node (i.e., receiving an EB). This incurs more transmission failures and repetitive parent changes. As a result, the node experiences desynchronization and the network leaving. As shown in Fig. 17(b), in standard TSCH, the number of nodes’ leaving the network increases as IPI decreases. DualBlock maintains network stability better than standard TSCH. This verifies that IBO operation affects both the routing layer and the TSCH network. DualBlock enhances link reliability and keeps nodes synchronized. It also minimizes routing path changes caused by PRR degradation.

E. IMPACT OF DUALBLOCK PARAMETERS

We perform experiments with various parameters on DualBlock-CRC/CFP, which use $T_{CRC}$ and $T_{CFP}$, respectively, to estimate the collision history. DualBlock-CRC decreases IBO window size when a collision does not occur during $T_{CRC}$ duration. DualBlock-CFP increases or decreases IBO window size by calculating the CFP for $T_{CFP}$. Furthermore, DualBlock-CFP uses the calculated CFP to make a comparison with the threshold probabilities $T_{hinc}$ and $T_{hdec}$ for making decisions on increasing or decreasing the maximum IBO value. We measure link reliability and radio duty cycle according to the traffic load. In Figs. 18, 19, and 20, high and low traffic indicate the IPI of 10 and 6, respectively.

1) $T_{CRC}$ in DualBlock-CRC

Fig. 18 shows that the PRR and duty cycle increase in low traffic when $T_{CRC}$ increases. The longer the $T_{CRC}$, the later the nodes decrease IBO window size (i.e., maintaining the current IBO window size longer). In high traffic, the PRR and duty cycle increase with $T_{CRC}$. DualBlock-CRC consumes more energy to achieve higher reliability.
the TSCH network. We set \((T_{\text{inc}}, T_{\text{dec}})\) to (95, 100) for achieving high reliability and low radio duty cycle.

In summary, *DualBlock*-CRC improves reliability at the expense of energy efficiency. *DualBlock*-CFP achieves high link reliability and high energy efficiency with appropriate parameter values of \(T_{\text{CFP}}, T_{\text{inc}}, \) and \(T_{\text{dec}}\). *DualBlock*-CFP outperforms standard TSCH and *DualBlock*-Basic/CRC when link reliability and energy efficiency are considered together.

VII. RELATED WORK

Our work is inspired by many studies that deal with the collision problems in IoT networks [17], [18], [19], [20], especially focusing on the collision problem that occurs when using IEEE 802.15.4 TSCH. Prior work on the collision problem in LLNs can be divided into two types: TSCH scheduling algorithm and IEEE 802.15.4e TSCH CSMA/CA algorithm.

A. TSCH SCHEDULING ALGORITHM

IETF Working Group 6TiSCH [21] published a 6tisch minimal configuration [22], which defines a simple fixed scheduling method for TSCH network. It simply places all the TSCH schedules on the first timeslot of slotframe to run IPv6 traffic on top of low-power TSCH networks. All nodes share one cell for both transmission and reception in the TSCH network. 6tisch minimal configuration is simple but exposed to collisions in a shared cell.

One of the most representative TSCH scheduling algorithms is Orchestra [1], an autonomous TSCH scheduler working with the RPL routing layer [16]. Many state-of-the-art works based on Orchestra have been proposed to improve network performance.

The most recent TSCH scheduling algorithm, SmartTiSCH [23], proposes an interference-aware engine for IEEE 802.15.4e-based networks. It passively observes and infers the internal or external interference by utilizing the existing data exchanges. When the interference is observed in the link, the transmitter and receiver will autonomously enter the cell in the control channel and exchange the interference mitigation strategy by shifting the timing of ACK. Although SmartTiSCH handles the internal interference, it counteracts...
after it experiences the interference and still cannot avoid the internal interference through the CCA operation.

In [2], the authors propose advanced autonomous TSCH scheduling, ALICE, for unicast slotframes. It allocates each directional link (a pair of transmitter and receiver) to a separate timeslot for each slotframe to grant link diversity. ALICE performs time-varying scheduling to prevent consecutive collisions due to a fixed schedule (when different links always overlap each other in the same cell for each slotframe). However, the time-varying scheduling may incur collisions at other links since ALICE allocates each link to each cell according to the result of the hash function.

A\(^3\) [5] is an autonomous and adaptive slot allocation scheme that adjusts the number of slots per slotframe according to the traffic load. It measures the traffic load by using both transmitter-side and receiver-side load estimations. With the high traffic load, packet collisions rise up even if A\(^3\) allocates more slots.

In [4], the authors propose a traffic-aware on-demand TSCH scheduling scheme, OST. It creates a binary resource tree to provide a dedicated link to each node. Nevertheless, if nodes cannot get required resources, they send packets to the shared link during the negotiation process. The nodes transmitting on the shared link may suffer from collisions until they get required resources.

The authors in [3], [24], [25] point out that Orchestra provides a fixed schedule based on a given routing topology. For this reason, some nodes with many child nodes can suffer from many packets. In [3], each node reports its traffic load to neighbor nodes. The receiving node adaptively controls its slotframe size according to the required traffic load of each transmitter. From doing this, nodes with many child nodes can take more resources.

The work in [24] allows nodes to exchange scheduling information by using the reserved fields in RPL protocol messages, and it allocates more cells to nodes that require more resources. Therefore, it can work adaptively with reliability requirements and traffic intensity. In [25], some nodes suffer from a large number of backlogged queue size due to fixed schedules based on the number of routing-layer neighbors. To empty the transmission queue quickly, multiple packets are transmitted. The transmitter transmits its backlogged queue size by injecting it into the TSCH header. Upon receiving packets, the receiver schedules receiving cells immediately. Thus, the transmitter and receiver can exchange multiple packets on multiple slots. The schemes above adaptively give priority or more resources according to the traffic intensity. However, they do not solve the collision problem directly.

TSCH scheduling methods may alleviate collisions through efficient resource allocation. However, they cannot be a fundamental solution to avoid collisions. We design DualBlock to enhance MAC operation. Applying DualBlock to the above TSCH scheduling methods can improve performance.

### B. IEEE 802.15.4E TSCH CSMA/CA ALGORITHM

In recent years, many studies analyze IEEE 802.15.4e TSCH CSMA/CA by considering the collision probability. Bianchi’s model in [26] has inspired most of these analytical studies, and analyzes throughput performance of IEEE 802.11 distributed coordination function (DCF) by using a two-dimensional Markov chain under saturated traffic and ideal channel conditions. This method greatly influenced the performance derivation of IEEE 802.15.4e TSCH CSMA/CA mechanism.

In [27], the authors present an analytical model based on a Markov chain that considers retry limits, transmission, and acknowledgement mechanism in a timeslot, under the saturation condition. In [28], the authors present an analytical model based on a discrete time Markov chain to describe the behavior of IEEE 802.15.4-TSCH network. They consider both bursty traffic and the usage of shared cell under the non saturated condition. In [29], the authors develop a two-dimensional Markov chain model for IEEE 802.15.4e TSCH CSMA/CA mechanism. They take into account the deterministic behavior of this mechanism in a shared link. They study how the number of devices that share the same link affects network performance under saturated and unsaturated conditions. In [30], the authors present an analytical model of TSCH CSMA/CA algorithm based on a Markov chain model by considering the collision probability. They also consider the capture effect that commonly occurs in real wireless networks.

In [31], the authors develop an analytical model based on a discrete time Markov chain for modelling the behaviour of IEEE 802.15.4e TSCH CSMA/CA algorithm by taking into account channel errors in industrial wireless sensor networks. In [32], the authors propose a stochastic model for performance analysis of TSCH networks for shared links with non-ideal wireless link properties. The above works are used to derive network performance using shared links in TSCH networks, in terms of packet delivery ratio, packet latency, and energy consumption. Although the analytic model formulates collision probability, there has been no method for alleviating the collision problem of TSCH intra network.

In summary, all the above studies evaluate and try to understand performance under the limitation that CSMA/CA cannot be performed within a TSCH slot. However, we overcome this limitation with DualBlock which mitigates the intra-slot collision problem while achieving acceptable energy efficiency.

### VIII. CONCLUSIONS

In this paper, we verified that the collision problem in TSCH shared cells degrades performance in TSCH networks. To mitigate this, we proposed DualBlock that uses a back-off operation within TSCH’s timeslots. To consider a trade-off between reliability and energy efficiency, we enhanced DualBlock by adding the receiver-triggered IBO window size adaptation that adaptively controls window size of IBO by estimating the CRC or CFP at a receiver. To verify the
performance of DualBlock compared to that of standard TSCF, we implemented the competitive schemes on low-power embedded devices using Contiki OS, and evaluated them under various traffic load scenarios. The experimental results confirm that DualBlock significantly outperforms standard TSCF; DualBlock-CFP performs best by achieving >3.6x higher end-to-end PDR while reducing the radio duty cycle by ~75% compared to standard TSCF in high traffic.

REFERENCES
[1] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, “Orchestra: Robust mesh networks through autonomously scheduled tsch,” in Proceedings of the 13th ACM conference on embedded networked sensor systems, 2015, pp. 337–350.
[2] S. Kim, H.-S. Kim, and C. Kim, “Alice: Autonomous link-based cell scheduling for tsch,” in Proceedings of the 18th International Conference on Information Processing in Sensor Networks, 2019, pp. 121–132.
[3] J. Jeong, J. Pae, H.-S. Kim, and S. Bahk, “Tesla: Traffic-aware elastic slotframe adjustment in tsch networks,” IEEE Access, vol. 7, pp. 130468–130483, 2019.
[4] S. Jeong, H.-S. Kim, J. Pae, and S. Bahk, “Ost: On-demand tsch scheduling with traffic-awareness,” in IEEE INFOCOM 2020-IEEE Conference on Computer Communications. IEEE, 2020, pp. 69–78.
[5] S. Kim, H.-S. Kim, and C.-S. Kim, “Adaptive autonomous allocation of tsch slots,” in Proceedings of the 20th International Conference on Information Processing in Sensor Networks (co-located with CPS-IoT Week 2021), 2021, pp. 299–314.
[6] D. Jung, Z. Zhang, and M. Winslett, “Vibration analysis for iot enabled predictive maintenance,” in 2017 IEEE 33rd International Conference on Data Engineering (ICDE). IEEE, 2017, pp. 1271–1282.
[7] S. Kim, M. P. Andersen, H.-S. Kim, and D. E. Culler, “Performant (TCP) for low-power wireless networks,” in 17th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 20), 2020, pp. 911–932.
[8] J. Adkins, B. Ghena, N. Jackson, P. Pannuto, S. Rohrer, B. Campbell, and P. Dutta, “The signpost platform for city-scale sensing,” in 2018 17th {ACM}//IEEE International Conference on Information Processing in Sensor Networks (IPSN). IEEE, 2018, pp. 188–199.
[9] A. Dunkels, B. Grønvall, and T. Voigt, “Contiki-a lightweight and flexible operating system for tiny networked sensors,” in 29th annual IEEE international conference on local computer networks. IEEE, 2004, pp. 455–462.
[10] C. Adji et al., “Fit iot-lab: A large scale open experimental iot testbed,” in 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT). IEEE, 2015, pp. 459–465.
[11] L.-E. C. Infrastructure and M. L. P. Layer, “Ieee standard for low-rate wireless networks,” IEEE Std, pp. 1–708, 2015.
[12] M. Buettner, G. V. Yee, E. Anderson, and R. Han, “X-mac: a short preamble mac protocol for duty-cycled wireless sensor networks,” in Proceedings of the 4th international conference on Embedded networked sensor systems, 2006, pp. 307–320.
[13] Y. Sun, O. Gurewitz, and D. B. Johnson, “Ri-mac: a receiver-initiated asynchronous duty cycle mac protocol for dynamic traffic loads in wireless sensor networks,” in Proceedings of the 6th ACM conference on Embedded network sensor systems, 2008, pp. 1–14.
[14] A. Dunkels, “The contiki mac radio cycling protocol,” 2011.
[15] AT86RF231/232/231/ZU/ZF datasheet., Atmel Corporation. [Online]. Available: http://www.atmel.com/Images/doc8111.pdf
[16] T. Winter, P. Thubert, A. Brandt, J. W. Hui, R. Kelsey, P. Levis, K. Pister, R. Struik, J.-P. Vasseur, R. K. Alexander et al., “Rpl: Ip6v6 routing protocol for low-power and lossy networks.” RFC, vol. 6550, pp. 1–157, 2012.
[17] S. Cui and J. Joe, “Collision prediction for a low power wide area network using deep learning methods,” Journal of Communications and Networks, vol. 22, no. 3, pp. 205–214, 2020.
[18] D. Kim, E. Noel, and K. W. Tang, “Wsn communication topology construction with collision avoidance and energy saving,” in 2014 IEEE 11th Consumer Communications and Networking Conference (CCNC). IEEE, 2014, pp. 398–404.
[19] A. Musaddiq, R. Ali, J.-G. Choi, B.-S. Kim, and S.-W. Kim, “Collision observation-based optimization of low-power and lossy iot network using reinforcement learning,” CMC-ADAPTIVE INTRA-SLOT CSMA/CA for TSCF

This article has been accepted for publication in IEEE Access. This is the author’s version which has not been finally edited and content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3186990

JEONGJUN PARK (Student Member, IEEE) received the B.S. degree in department of information communications engineering from Hankuk University of Foreign Studies (HUFS) and M.S. degree in electrical and computer engineering from Seoul National University (SNU), in 2018 and 2020, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering, Seoul National University (SNU), Seoul, South Korea. His research interest includes Internet of Things and contention resolution for low-power networks.
HONGCHAN KIM (Student Member, IEEE) received the B.S. degree in Electrical and Computer Engineering from Seoul National University in 2015. He is currently pursuing the Ph.D. degree in the Department of Electrical and Computer Engineering, Seoul National University, Seoul, South Korea. His research interests include designing network protocols for low-power networks and constructing practical IoT systems. He received the National Research Foundation (NRF) Global Ph.D. Fellowship, in 2016.

HYUNG-SIN KIM (Member, IEEE) received the B.S. degree in electrical engineering and the M.S. and Ph.D. degrees in electrical engineering and computer science (EECS) from Seoul National University (SNU), Seoul, South Korea, in 2009, 2011, and 2016, respectively, all with outstanding thesis awards. He was a Postdoctoral Scholar at Network Laboratory (NETLAB), SNU, until August 2016 and Real-time, Intelligent, Secure, Explainable systems (RISELab), University of California, Berkeley, until August 2019, and a Software Engineer at Google Nest until February 2020. He received the Qualcomm Fellowship in 2011 and the National Research Foundation (NRF) Global Ph.D. Fellowship and Postdoctoral Fellowship in 2011 and 2016, respectively. He is currently an assistant professor at Graduate School of Data Science, SNU. His research interest includes Internet of Things and ambient artificial intelligence.

SAEWOONG BAHK (Senior Member, IEEE) received the B.S. and M.S. degrees in electrical engineering from Seoul National University (SNU), in 1984 and 1986, respectively, and the Ph.D. degree from the University of Pennsylvania, in 1991. He was with AT&T Bell Laboratories as a Member of Technical Staff, from 1991 to 1994, where he had worked on network management. From 2009 to 2011, he served as the Director of the Institute of New Media and Communications. He is currently a Professor at SNU. He has been leading many industrial projects on 3G/4G/5G and the IoT connectivity supported by Korean industry. He has published more than 300 technical articles and holds more than 100 patents. He is a member of the National Academy of Engineering of Korea (NAEK). He was a recipient of the KICS Haedong Scholar Award, in 2012. He was President of the Korean Institute of Communications and Information Sciences (KICS). He has been serving as Chief Information Officer (CIO) of SNU. He was General Chair of the IEEE WCNC 2020 (Wireless Communication and Networking Conference), IEEE ICCE 2020 (International Conference on Communications and Electronics), and IEEE DySPAN 2018 (Dynamic Spectrum Access and Networks). He was Director of the Asia-Pacific Region of the IEEE ComSoc. He is an Editor of the IEEE Network Magazine and IEEE Transactions on Vehicular Technology. He was TPC Chair of the IEEE VTC-Spring 2014, and General Chair of JCCI 2015, Co-Editor-in-Chief of the Journal of Communications and Networks (JCN), and on the Editorial Board of Computer Networks Journal and the IEEE Tran. on Wireless Communications.