Firing Offset Adjustment of Bio-Inspired DESYNC-TDMA to Improve Slot Utilization Performances in Wireless Sensor Networks

Kwangsoo Kim[1], Seung-hun Shin[2], and Byeong-hee Roh[1]
[1]Department of Computer Engineering, Ajou University
[2]Dasan University College, Ajou University
Suwon, Gyeonggi-do, 16499, Republic of Korea
[e-mail: {zubilan, sihnsh, bhroh}@ajou.ac.kr]
*Corresponding author: Byeong-hee Roh

Received October 19, 2016; revised December 28, 2016; accepted January 22, 2017; published March 31, 2017

Abstract

The wireless sensor network (WSN) is a key technology to support the Internet of things (IoT) paradigm. The efficiency of the MAC protocol in WSN is very important to take scalability with restricted wireless resources. The DESYNC-TDMA has an advantage of simple distributed slot allocation inspired by nature, but there is a critical disadvantage of split slots by firing message. The basic split slot model has less efficiency for continuous packet transmitting because of wasting of the slots less than the packet size. In this paper, we propose a firing offset adjustment scheme to improve the efficiency of slot utilizations, which can manage the slot assigned to each node as a single large block, called the single slot model. The performance analysis models for both the existing and the proposed schemes are also derived. Experimental results show that the proposed method provide better efficiency of slot utilization than the existing schemes without any loss of the nature of the desynchronization.

Keywords: Wireless Sensor Networks, Desynchronization, self-organizing, MAC, DEYNC-TDMA, slot efficiency

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea Government (MSIP) (no. NRF-2015R1A2A2A01005577).

https://doi.org/10.3837/tiis.2017.03.014 **ISSN : 1976-7277**

1. Introduction

Recently, supported by advances in the hardware of small devices, Internet of Things (IoT) paradigm that all of the things are connected to the Internet, has attracted a lot of attention. Wireless Sensor Network (WSN) is widely used as a based network technology to collect information and control the actuators. WSN is composed of a large number of low cost devices that have a battery and low computation power. Thus the protocol efficiency of medium access control (MAC) layer in WSN is very important to make the network more scalable that can accommodate a large number of sensor nodes with restricted wireless resources [1].

The MAC protocol for WSN can be roughly categorized into two major categories: contention-based carrier sense multiple access (CSMA) protocols and schedule-based time division multiple access (TDMA) protocols. The collision avoidance (CA) mechanism is mostly used for avoiding collisions caused by hidden nodes, but it causes too much overhead to the protocol. The fixed TDMA is not efficient due to the vacant slot. To overcome this inefficiency problem, the dynamic TDMA (D-TDMA) schemes have been proposed not only to solve the problem of CSMA/CA, but also to meet the quality of service (QoS) requirements of users [2], [3], [4], and [5]. For the operation of D-TDMA, even scheduling the slot via contention is preferred, but it is still challenging to find the optimal slot scheduling scheme. Moreover, communication nodes should synchronize the beginning time of epoch with others in the network, called global synchronization, by using Average Time Synch [6], for example. However, the synchronization has been known as one of the most challengeable and difficult tasks to be operated in practical wireless environments [7].

The self-organizing feature in wireless networks becomes more important as the number of devices increases. The similarity between the synecology and the network system, the bio-inspired networking paradigm has been studied. It is common in the synecology that numerous members perform a mission by communicating with each other and collaborating organically. It can be summarized as an autonomous judgment by the individual member, called stigmergy as a group behavior. [8]

To solve the synchronization problem in TDMA-based protocols, firefly synchronization (F-SYNC) [9] is a good solution. The fireflies make herd and emit the light at the end of its tail periodically. The emission of light is called *firing*. Interestingly, the emission time of firing becomes synchronized as the time goes by because the period of firing is affected by other fireflies. Thus the firings are finally flickering with the same period. The firing is modeled to the epoch start time of TDMA networks in the F-SYNC algorithm to synchronize the beginning of epoch by distributed and decentralized nodes.

The DESYNC-TDMA [7], which is an inverse concept of the F-SYNC algorithm [9], has been proposed. While the F-SYNC algorithm focuses the firing into at a moment, the DESYNC-TDMA algorithm distributes the firings on the period. DESYNC-TDMA adapts a simple reservation based MAC protocol, in which each node determines its TDMA *slot*, which is the time duration that it can transmit data, in a distributed manner by sending control packets (called *firing*) to the network and hearing them from other nodes. Each firing packet from a node is periodically sent at the middle of its slot. In DESYNC-TDMA, nodes can join and leave the network freely with a distributed manner. It is noted that the slots of nodes may change according to the number of nodes in the network. With the self-configuring, distributed and Pulse-Coupled Oscillator (PCO) [10] natures of DESYNC-TDMA, there have been

numerous works to improve the protocol efficiency in wireless sensor networks [11], [12], and [13] and to apply to the mobile ad hoc networks (MANETs) environment [14], [15], and [16].

In the DESYNC-TDMA, the slot is allocated around its firing message, and the firing message split the slot into two half slots. Nodes in DESYNC-TDMA allocate its slot in inverse proportion to the number of neighbor nodes. With this dynamicity and distributed nature, it is a good candidate that can replace the CSMA/CA in the large scale network, but there are two performance degradation points: the short length of allocated slot in crowded network and the splitting slot problem.

In this paper, we focus on the split slot model which is split by the firing message in the middle of the slot. The maximum size of data packet that can be transmitted in a slot is limited to half of the slot in the split slot model. It causes the performance degradation and gives the low capacity to the network by splitting the slot into insufficient size to send a packet. We figure out this insufficient slot as the wasted slot and there are two wasted slots in the split slot model.

To overcome this problem, the single slot model on which we focus as an ideal slot model is suitable. To the best of our knowledge, only one most similar architecture is shown in [17]. Mühlberger [17] suggested a slot model which the firing message is placed at the start of the slot. However, any detail mechanism and effectiveness of the method to realize the single slot model were not addressed in [17] and it was only for the convenience of the analysis. Since nodes in a DESYNC-TDMA network share the channel evenly with others, the suggested method of [17] has also a problem on channel waste, which will be discussed in Section 2.

In this paper, we propose a firing offset adjustment scheme to improve the efficiency of slot utilizations, which can also improve the transmission throughput. The proposed scheme can manage the slot assigned to each node as a single large block, which makes it possible for the node to transmit a data packet of larger size without fragmenting it. It shows the better efficiency of slot utilization than the existing schemes without any loss of the nature of the desynchronization.

The rest of the paper is organized as follows. Section 2 describes the background and defines the channel waste problem of DESYNC-TDMA. Then, the proposed method and its analysis model are addressed in Section 3. Section 4 gives the experimental results of the proposed scheme. Finally, section 5 concludes the paper.

2. Backgrounds

2.1 DESYNC-TDMA

DESYNC-TDMA is a distributed dynamic-TDMA protocol inspired by the firefly's habits. It uses the *firing message* modeled by light emissions of fireflies. In DESYNC-TDMA, firing messages are utilized as anchor times for slot assignments to the participating nodes. The DESYNC-TDMA consists of two processes: the *desynchronization* and the *slot assignment*. With the desynchronization process, firing times of all nodes are evenly distributed in the given cyclic period. Then, the slot for each node to use for its data transmission is allocated through the slot assignment process.

2.1.1 Desynchronization Process

Fireflies have the herd instinct in nature, called synecology. Each firefly emits lights at the end of its tail, periodically. The bioluminescence of firefly has the tendency to move to the

most crowded time. At the end of the moving the emission period, almost of fireflies emit (hereafter fire) the light (hereafter firing message) at the same time. It can be called a synchronization and it is similar to the synchronization process of the start time of epoch in TDMA. Such a synchronization scheme inspired by firefly's instinct called Firefly synchronization [9]. In contrast to the Firefly synchronization which focuses on firings at a moment, the desynchronization process distributes the firings evenly in a given cycle *T*. It means that in desynchronization, the time is split into a constant cyclic period of *T*.

Let *N* and $\phi_i(t)$ be the number of nodes and the time that *i*-th node firings in *t*-th *T* interval, respectively, where *i*=1, 2, …, *N* and *t*=0, 1, 2, …. Let $\phi_{i,-}(t)$ and $\phi_{i,+}(t)$ be the firing times of other nodes, which are just before and after *i*-th node's firing time in *t*-th *T* interval, respectively. Based on the Pulse Coupled Oscillator (PCO) [10] mechanism, the desired firing time $\phi_i(t+1)$ in the next (*t*+1)-th *T* interval is written by

$$\phi_i(t+1) = T + (1-\alpha) \times \phi_i(t) + \alpha \times \{\phi_{i,-}(t) + \phi_{i,+}(t)\}/2 , \quad (1)$$

where $\alpha \in [0,1]$ is the jumping factor that determines how far the node moves from its current phase $\phi_i(t)$ towards the desired midpoint $\phi_i(t+1)$.

Fig. 1 shows an example of the desynchronization process with 6 nodes. The time interval *T* is represented by a large circle. Black circles denote firing times of nodes during the current time interval *t*. Grey circles represent firing times of corresponding nodes during the previous time interval *t*-1. Let consider firing times of nodes are given arbitrarily at *t*=0 (an initial stage) as shown in **Fig. 1 (a)**. From **Fig. 1 (b)** and **Fig. 1 (c)**, we can see that each node moves its firing time to the midpoint of those just before and after its firing time in the previous time interval *t*-1 according to Eq. (1). As the time interval *t* are going on, finally, all firing times of nodes are evenly distributed in the time interval, as shown in **Fig. 1 (d)**, which is called that the network becomes desynchronized.

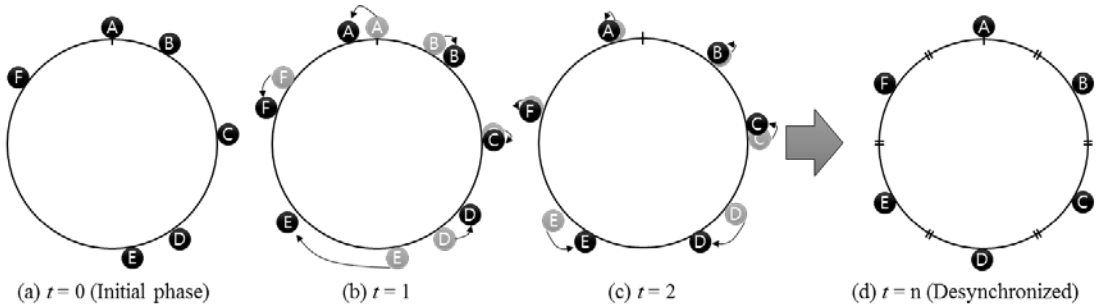

Fig. 1. An example of desynchronization process.

2.1.2 TDMA Slot Assignment Process

The *slot* is defined as the time period that a node can transmit its data during the given time interval *T*. With the information of firing times obtained from the desynchronization process, each node can determine its slot in a distributed manner as follows: Let define $S_{i,start}(t)$ and $S_{i,end}(t)$ as the start and end times of *i*-th node during *t*-th time interval, respectively. Then the slot of *i*-th node during (*t*+1)-th time interval is calculated by

$$S_{i,start}(t+1) = T + \{\phi_{i,-}(t) + \phi_i(t)\}/2 \quad (2)$$

$$S_{i,end}(t+1) = T + \{\phi_i(t) + \phi_{i,+}(t)\}/2 . \quad (3)$$

As we can see from Eqs. (2) and (3), the slot length varies depends on firing times, $\phi_{i,-}(t)$ and $\phi_{i,+}(t)$ as well as $\phi_i(t)$. It means that the slot length decreases as the number of neighbor nodes increases.

Fig. 2 shows an example of the slot assignment process for three neighbor nodes, A, B and C. In **Fig. 2**, $\phi_A(t)$, $\phi_B(t)$, and $\phi_C(t)$ denote firing times of node A, B and C, respectively, during current time interval *t*, while $\phi_A(t+1)$, $\phi_B(t+1)$, and $\phi_C(t+1)$ denote ones during next time interval *t*+1. To explain the process of calculating the firing time and slot, the firing time of node A and B are fixed in this example. Regarding node B, $\phi_{B,-}(t) = \phi_A(t)$ and $\phi_{B,+}(t) = \phi_C(t)$. According to Eqs. (2) and (3), for example, the slot start and end times of node B during next time interval *t*+1, $S_{B,start}(t+1)$ and $S_{B,end}(t+1)$, are determined as $\text{mid}(\phi_A(t), \phi_B(t))$ and $\text{mid}(\phi_B(t), \phi_C(t))$, respectively, where $\text{mid}(X, Y)$ means a function whose result is the average of *X* and *Y*. Also, the firing time of node B in next time interval *t*+1, $\phi_B(t+1)$, is located within the slot, as shown in **Fig. 2**. With the slot assignment process, each node can determine its slot without overlapping with others. It is noted that all the slot lengths of nodes in one hop range are same since all firing times of all nodes are evenly distributed in the time interval as a result of the desynchronization process as mentioned previously.

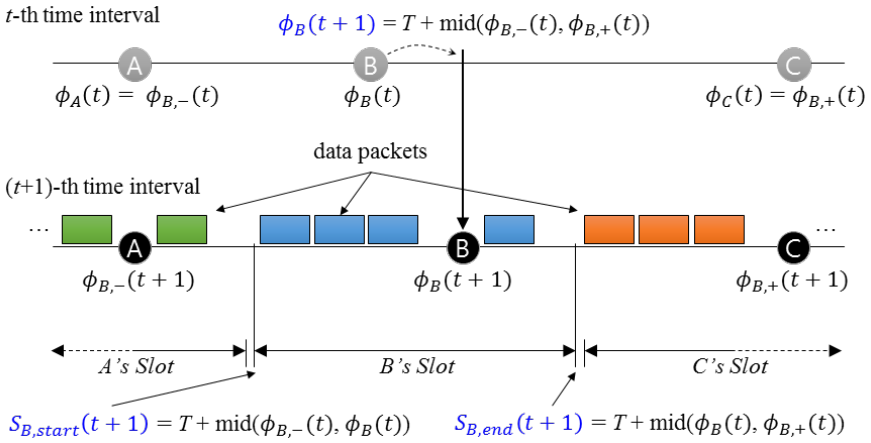

Fig. 2. DESYNC-TDMA slots

2.2 Problem Definition: The Slot Waste Problem

DESYNC-TDMA is a D-TDMA MAC protocol and works like previous proposals [2][3][4][5][18]. Most of D-TDMA protocols have an overhead period for the slot assignment or negotiation. The period *T* is split into the contention period for request and confirmation the TDMA slot, and the data period for transmitting the data using the allocated slots. A node knows its slot offset which can be used dedicatedly. Similarly, DESYNC-TDMA uses firing message as a slot request overhead but there is no negotiation process. The slot in DESYNC-TDMA is allocated around the firing message.

Firing times of each node are always located within its allocated slot periods, as shown in **Fig. 2**. In addition, each node's firing message should be sent at its firing time determined by Eq. (1). Accordingly, if the transmission time of a frame from a node is overlapped with its firing message transmission time, then the frame should be fragmented or the start time of the frame transmission should be delayed after the firing.

Since the desynchronization process makes all nodes' firing times to be evenly distributed in a time interval, slot lengths of all nodes converge a same value, $L_s \approx T / N$, by Eqs. (2) and (3).

Since the firing messages are sent at the middle of allocated slots, if the transmission time of a frame is overlapped with that of the firing message as shown in **Fig. 3 (a)**, the frame transmission should be deferred after the completion of the firing message's transmission. In addition, if the transmission completion time of a frame exceeds its slot boundary, the transmission should be also delayed until its next slot. It causes *the slot waste problem*, in which a part of the slot can not be utilized for the frame delivery due to the firing at the middle of the slot as shown in **Fig. 3 (a)**. It is noted that the slot waste problem of **Fig. 3 (a)** is caused by the firing message sent at the middle of the slot, and we call it as *the split slot model*.

Let assume a saturation traffic condition where all nodes always have frames to transmit. As the number of nodes increases, the slot length for each node decrease, and the number of frames delayed and buffered during the slot also increases. Then, it may lead to the saturation traffic condition. Under the saturation traffic condition, the maximum sizes of frames to be transmitted consecutively before and after the firing time, $L_{\max}^{(1)}$ and $L_{\max}^{(2)}$, respectively, are limited to as following

$$L_{\max}^{(1)} \le L_s/2 \quad \text{and} \tag{4}$$

$$L_{\max}^{(2)} \le L_s/2 - L_f, \tag{5}$$

where L_f denotes the transmission time of the firing message, and the inter-frame gap (IFG) between continuous and consecutive transmitted frames is assumed to be ignored.

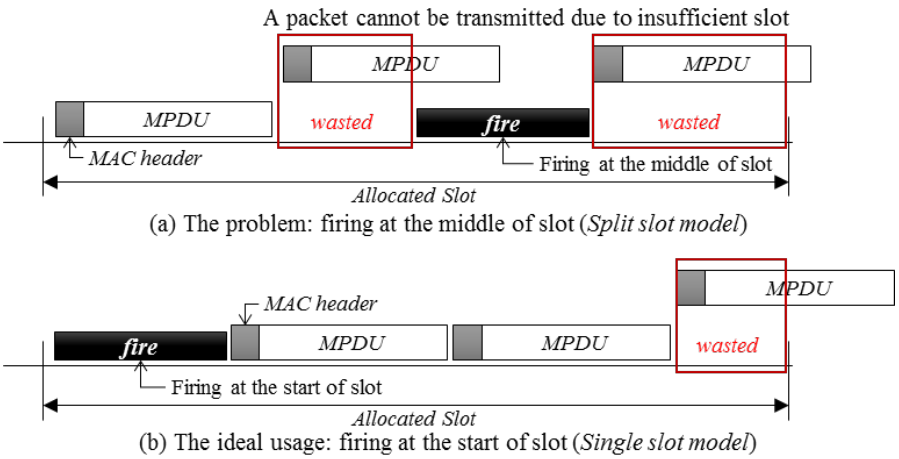

Fig. 3. The examples of slot wasted by split slot model and single slot model

To overcome the slot waste problem, the time of the firing message can be moved to the start of the slot as shown in **Fig. 3 (b)**. We call it as *the single slot model*. Then, the maximum size of frames to be transmitted consecutively during the slot ($L_{\max}^{\text{single}}$) is given by $L_{\max}^{\text{single}} \le L_s - L_f$, which is longer than that in the split slot model and the slot can be utilized more effectively for the frame transmission. A similar structure of the single slot model has been assumed in [17], in which it was only an assumption that the first packet during the slot may be utilized as the firing message. However, the author of [17] did not provide any practical mechanism and effectiveness of the method.

3. Firing Offset Adjustment of DESYNC-TDMA

We can consider a slot assignment scheme to implement the *single slot model* while complying with original DESYNC-TDMA as follows. **Fig. 4** shows a comparison of two slot assignment schemes: In the slot assignment process of original DESYNC-TDMA, the slot is allocated according to Eqs. (2) and (3) and the firing occurs at the middle of the slot, as shown in **Fig. 4 (a)**. To implement the *single slot model*, the slot of a node can be reallocated from the start time of its firing to the time just before the firing of its next neighbor (i.e. *firing* to *firing*), as shown in **Fig. 4 (b)**. However, it is not feasible due to the slot violation problem by firing messages, as described in **Fig. 5**. For example, given the slot allocation situation at *t*-th time interval shown in **Fig. 4 (b)**, node B's next firing time $\phi_B(t+1)$ is determined as the $\text{mid}(\phi_A(t), \phi_C(t))$. When $\phi_B(t+1) \subset [S'_{A,start}(t+1), S'_{A,end}(t+1)]$, $\phi_B(t+1)$ collides to the slot of node A. It means that the node B's next firing message may be collided with node A's data packet when it is sent when the network is not desynchronized. The loss of firing messages due to such collisions causes the failure of desynchronization. Thus to use the *firing* to *firing* slot assignment scheme, the desynchronization process needs to be modified not to collide the firing message and the slots of other nodes. However, it breaks the rule of desynchronization.

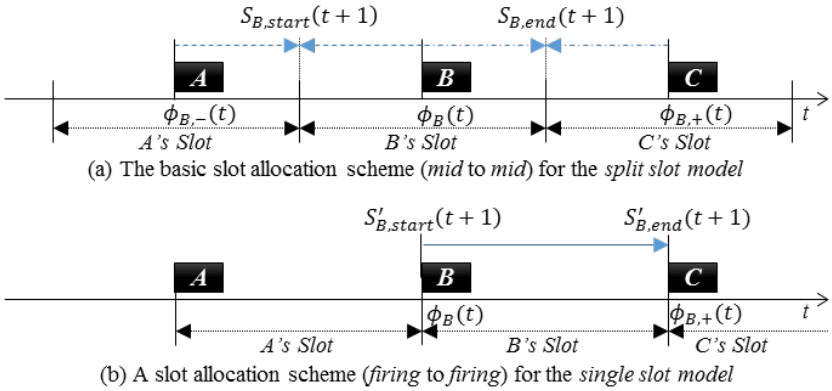

Fig. 4. A comparison of slot assignment schemes

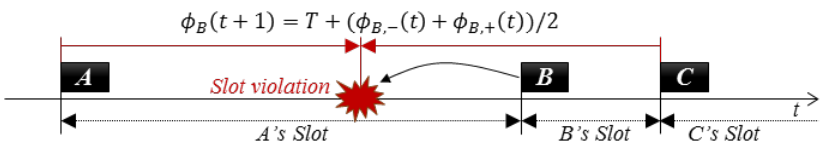

Fig. 5. An example of slot violation when using the *firing-to-firing* slot assignment

3.1 Firing Offset Adjustment Scheme

As discussed before, the implementation of the single slot model is not simple work. The critical problem of the single slot model in **Fig. 4 (b)** is the slot violation, in which the firing message is sent out of the allocated slot and then it causes the failure of the desynchronization process due to the collision of firing messages. Basically, to maintain the desynchronization process in the single slot model, each node should know the original firing time of each neighbor node, $\phi(t)$ [7].

In this section, we propose a firing offset adjustment scheme to effectively implement the single slot model in DESYNC-TDMA, which does not break the rules of the desynchronization process. It is noted that the exact scheduling of nodes' firing times is the key point to maintain the desynchronization process in DESYNC-TDMA. The proposed method consists of three parts: transfer of the time offset by firing message, estimation of the original firing time, and calculation of the desired firing time in the next slot.

1) Transfer of the time offset by firing message

In DESYNC-TDMA, nodes utilize the reception time of firing messages from neighbors to calculate its firing time in next time interval using Eq. (1). When it is applied to the single slot model shown in **Fig. 4 (b)**, it may cause the slot violation problem as **Fig. 5**, for example. To overcome the slot violation problem in the desynchronization process, each node should know the original firing times of neighbors calculated by Eq. (1), instead of the reception times of firing messages.

Let $T_{j,offset}(t)$ be the time offset between the original firing time of node j calculated by Eq. (1) and the actual firing time sent at the start of the assigned slot in t-th time interval. Then, we have

$$T_{j,offset}(t) = \phi_j(t) - S_{j,start}(t). \tag{6}$$

Fig. 6 shows the proposed format of the firing message, which is extended from the message described in [7]. The firing message format defined in the original DESYNC-TDMA of [7] consists of `<Type>` and `<ID>`, which mean the field to distinguish whether it is firing or data packet and the node's identifier (e.g., node's MAC address), respectively. Also, the new added field of `<Time_Offset>` denotes the time offset obtained by Eq. (6).

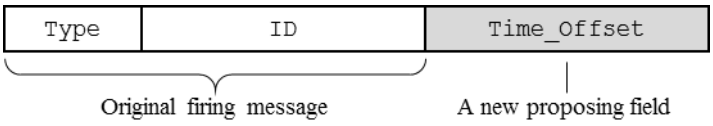

Fig. 6. The format of firing message

2) Estimation of the original firing time

Let $\hat{\phi}_{i,j}(t)$ be the time that node i receives a firing message from node j in t-th time interval. In the firing message, the time offset of node j, $T_{j,offset}(t)$, is included. Then, the node i can obtain the original firing time of node j as

$$\phi_{i,j}(t) = \hat{\phi}_{i,j}(t) + T_{j,offset}(t). \quad (7)$$

Among $\phi_{i,j}(t)$ s, node *i* can determine its $\phi_{i,-}(t)$ and $\phi_{i,+}(t)$ as the estimated original firing times just before and after its firing time, $\phi_i(t)$, respectively.

3) Calculation of the desired firing time in the next slot

With estimated $\phi_{i,-}(t)$ and $\phi_{i,+}(t)$, the desired firing time of node *i* in the next (*t*+1)-th time interval, $\phi_i(t+1)$, can be calculated by using Eq. (1). The slot duration $[S_{i,start}(t+1), S_{i,end}(t+1))$ in (*t*+1)-th time interval is also calculated by using Eqs. (2) and (3). Then, node *i* sends its firing message at $S_{i,start}(t+1)$ in (*t*+1)-th time interval, which includes time offset value of $T_{i,offset}(t+1) = \phi_i(t+1) - S_{i,start}(t+1)$.

3.2 Numerical Analysis

To analyze the performance of the proposed single slot model, it is assumed a saturation traffic condition, in which each node always have the data packet to send. The slot utilization is considered as a major performance measure, and the following terms are utilized for the analysis as shown in **Table 1**.

Table 1. Notations

Symbol	Description
N	number of nodes in the network
T	TDMA frame time interval (in second), a cycle length
L_s	Length of a slot (in second), assigned to a node
L_f	Length of a firing message (in second)
P	MPDU length (in second)
H	Header length (in second), i.e., packet preamble + MAC header
R, R_1, R_2	Reminder of slot (in second) in a slot, 1^{st} half, and 2^{nd} half, respectively
N_{pkts} ,$N_{pkts,1}$, $N_{pkts,2}$	Number of packets in a slot, 1^{st} half, and 2^{nd} half, respectively
P_{add} ,$P_{add,1}$,$P_{add,2}$	Additionally transmitted packet length in a slot, 1^{st} half, and 2^{nd} half, respectively

As mentioned before, as the result of the desynchronization process, each node's slot is evenly distributed in a time interval *T*. That is, the slot length of each node converges the same value of $L_s \approx T/N$, by Eqs. (2) and (3). The slot structures for analysis of the split and the single slot models are shown in **Fig. 7 (a)** and **Fig. 7 (b)**, respectively, which are based on that illustrated in [17]. When the remainder part of a slot is not enough to send a whole data packet,

it can be wasted not sending the data packet or used to send the data packet by fragmenting it according to the slot utilization scenarios, which will be described later. Since the firing message is sent at the middle of the slot in the split slot model, each slot is split into two parts such as 1st half and 2nd half, before and after the firing message, respectively, as shown in **Fig. 7 (a)**. On the other hand, in the single slot model, each slot starts with a firing message, which is followed by data packets, as shown in **Fig. 7 (b)**. A data packet includes a header (*H*) and an MPDU (*P*). In both models, when the remainder part of a slot is not long enough to send a whole data packet, it can be wasted not sending the packet or used to send the packet by fragmenting it according to the slot utilization scenarios, which will be described later.

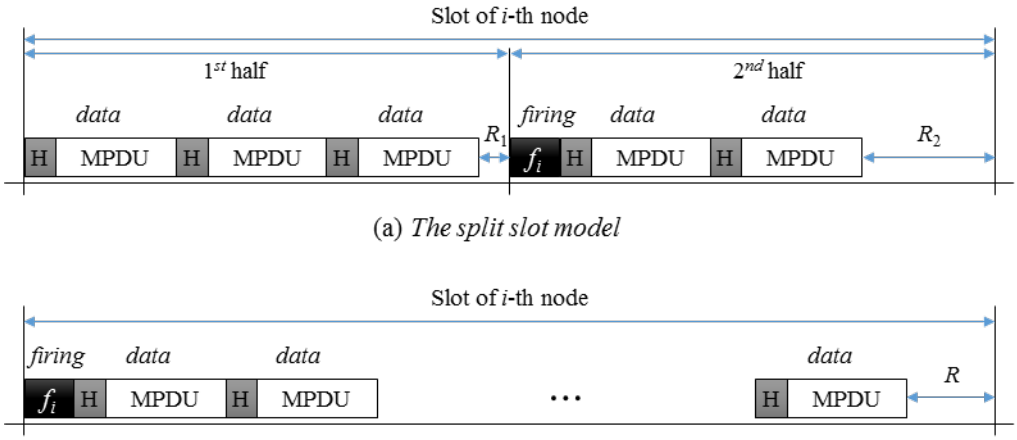

Fig. 7. Analysis model of two slot models

For the analysis, the following two scenarios with respect to the fragmentation are considered for the proposed single slot model (SNGL) and the existing split slot model (SPLT) presented in [7]: (a) without fragmentation (NOFRAG) and (b) with fragmentation (FRAG).

3.2.1 NOFRAG Scenario: without fragmentation

In NOFRAG scenario, nodes cannot transmit a data packet larger than the reminder of a slot, R_1 or R_2 in SPLT, or R in SNGL.

1) NOFRAG-SPLT (NOFRAG Scenario in the split slot model)

In the split slot model, a firing message splits its slot into two parts: the former part from the start of the slot to the start of the firing message (1st half), and the latter part of the firing message to the end of the slot (2nd half). It is noted that since the firing message is included in the 2nd half, thus the 2nd half has less usable than the 1st half. We have the maximum number of data packets that can be transmitted in each half as following

$$N_{pkts,1} = \left\lfloor \frac{L_s / 2}{H + P} \right\rfloor \tag{8}$$

$$N_{pkts,2} = \left\lfloor \frac{L_s/2 - L_f}{H+P} \right\rfloor. \tag{9}$$

Then, the total number of data packets in the whole slot allocated is written as

$$N_{pkts}^{\text{NOFRAG-SPLT}} = N_{pkts,1} + N_{pkts,2}. \tag{10}$$

2) NOFRAG-SNGL (NOFRAG Scenario in the single slot model)

As we can see from **Fig. 7 (b)**, the firing message sends at the beginning of the slot, the rest of the slot can be fully utilized for the delivery of data packets. So, the maximum number of data packets in the slot will be calculated by:

$$N_{pkts}^{NOFRAG-SNGL} = \left\lfloor \frac{L_s - L_f}{H+P} \right\rfloor \tag{11}$$

3.2.2 FRAG Scenario: with fragmentation

In some wireless communication environments, fragmentation schemes are used to improve reliability by increasing the probability of successful transmission in poor wireless channel characteristics [19]. We use the similar fragmentation scheme to model the fragmenting scenario. MPDU will be divided into two packets and the latter fragment needs additional header *H*. In FRAG scenario, when a data packet is larger than the reminder, it is fragmented into two parts. The first fragmented packet is made to be suitable to be sent in the remainder. Then, an additional header is appended to the second fragmented packet and it is sent at the next half or next slot.

1) FRAG-SPLT (FRAG Scenario in the split slot model)

In FRAG-SPLT scenario, there could be three types of fragmented data packets as shown in **Fig. 8**, where D means a whole data packet, while $D_{\bullet,1}^{f}$ and $D_{\bullet,2}^{f}$ denote the first and the second fragmented packets of $D_{\bullet}$. In **Fig. 8**, $D_{1,2}^{f}$ is the second fragmented packet not sending in the previous slot. $D_{2,1}^{f}$ and $D_{2,2}^{f}$ are the two fragmented packets split in a slot, respectively. They can be sent in a slot. And, $D_{3,1}^{f}$ is the first part of a packet fragmented and in the 2^{nd} half. The second fragmented packet $D_{3,2}^{f}$ will be sent at the start of the next slot.

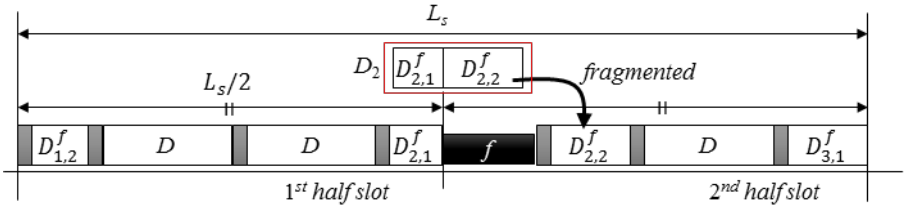

Fig. 8. Rearrangement of the fragmented packet

For the sake of the analysis, since the packet fragmentation is done in the remainder in each half, we can merge the durations of fragmented packets in each half into one remainder, R_1 and R_2 in 1^{st} and in 2^{nd} halves, respectively, as shown in the bottom of **Fig. 8**. Then, with Eqs. (8) and (9), the lengths of the merged reminder R1 and R2 can be obtained by

$$R_1 = \frac{L_s}{2} - N_{pkts,1} \times (H + P), \tag{12}$$

$$R_2 = \frac{L_s}{2} - L_f - N_{pkts,2} \times (H + P). \tag{13}$$

Now we can get the time which additional ratio of a packet can be transmitted in each half, written by:

$$P_{add,1} = \begin{cases} R_1 - 2H & if\ R_1 > 2H, \\ R_1 - H & if\ 2H \geq R_1 > H, \\ 0 & otherwise. \end{cases} \tag{14}$$

$$P_{add,2} = \begin{cases} R_2 - 2H & if\ R_2 > 2H, \\ R_2 - H & if\ 2H \geq R_2 > H, \\ 0 & otherwise. \end{cases} \tag{15}$$

Then, the total maximum number of data packets that can be sent in a slot is obtained by

$$N_{pkts}^{FRAG-SPLT} = N_{pkts,1} + N_{pkts,2} + \frac{(P_{add,1} + P_{add,2})}{P}. \tag{16}$$

2) FRAG-SNGL (FRAG Scenario in the single slot model)

Similarly, for the single slot model, the length of reminder R is calculated by the Eq. (14), and given by:

$$R = L_s - L_f - N_{pkts} \times (H + P). \tag{17}$$

Then we can have the time of packet transmitted in the reminder R, can be written by:

$$P_{add} = \begin{cases} R - 2H & \text{if } R > 2H, \\ R - H & \text{if } 2\text{H} \geq R > H, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Finally, the total number of packets in the slot for the single slot model is obtained by:

$$N_{pkts}^{FRAG-SNGL} = N_{pkts} + \frac{P_{add}}{P}. \tag{19}$$

6. Performance Evaluation

In this section, we show the performance enhancement of our firing offset adjustment scheme, called single slot model, comparing to the split slot model of [7]. We use the length parameters for a firing message (L_f) and a header (H) including packet preamble, as follows:

$$L_f = \begin{cases} L_{type} + L_{ID} & \text{for split slot model} \\ L_{type} + L_{ID} + L_{offset} & \text{for single slot model (proposed)} \end{cases} \tag{20}$$

$$H = \begin{cases} L_{preamble} + L_{type} + 2 \times L_{ID} & \text{for scenario } (a) \\ L_{preamble} + L_{type} + 2 \times L_{ID} + L_{frag} & \text{for scenario } (b) \end{cases}. \tag{21}$$

where L_{type}, L_{ID}, L_{offset}, and L_{frag} are the length variables for the field of type, identifier of a node, T_{offset}, and fragment, respectively. The $L_{preamble}$ stands for the preamble length. The header needs two IDs of source and destination. **Table 2** lists the parameters for performance evaluation of both analysis and simulation. The $L_{preamble}$ and L_{ID} are quoted from IEEE 802.11b [19].

Table 2. Parameters for performance evaluation

Parameter	Value
T	0.1 *sec*
N	From 1 to 150 (*x-axis*)
$L_{preamble}$	192 *us*
datarate	1,000,000 *bps*
L_{type}	4 *us* (4 *bits*)
L_{ID}	48 *us* (48 *bits*)
L_{frag}	4 *us* (4 *bits*)
L_{offset}	32 *us* (32 *bits*)

The simulation was performed by MATLAB and it conducted by modeling the DES simulation in a node. We collect the actual total amount of packet transmitted by a node during simulation time 1,000 *sec*. A node sends a packet and increases its simulation time as packet length (i.e. transmission delay). Then we compare the results of slot to MPDU efficiency from analysis and simulation, calculated by

$$S = \frac{N_{pkts}^{SCHME} \times P}{L_s}. \tag{22}$$

where *SCHME* is one of *NOFRAG-SPLT*, *FRAG-SPLT*, *NOFRAG-SNGL*, or *FRAG-SNGL*. It is noted that *NOFRAG-SPLT* and *FRAG-SPLT* denote the schemes for the original DESYNC-TDMA, while *NOFRAG-SNGL* and *FRAG-SNGL* denote the proposed ones.

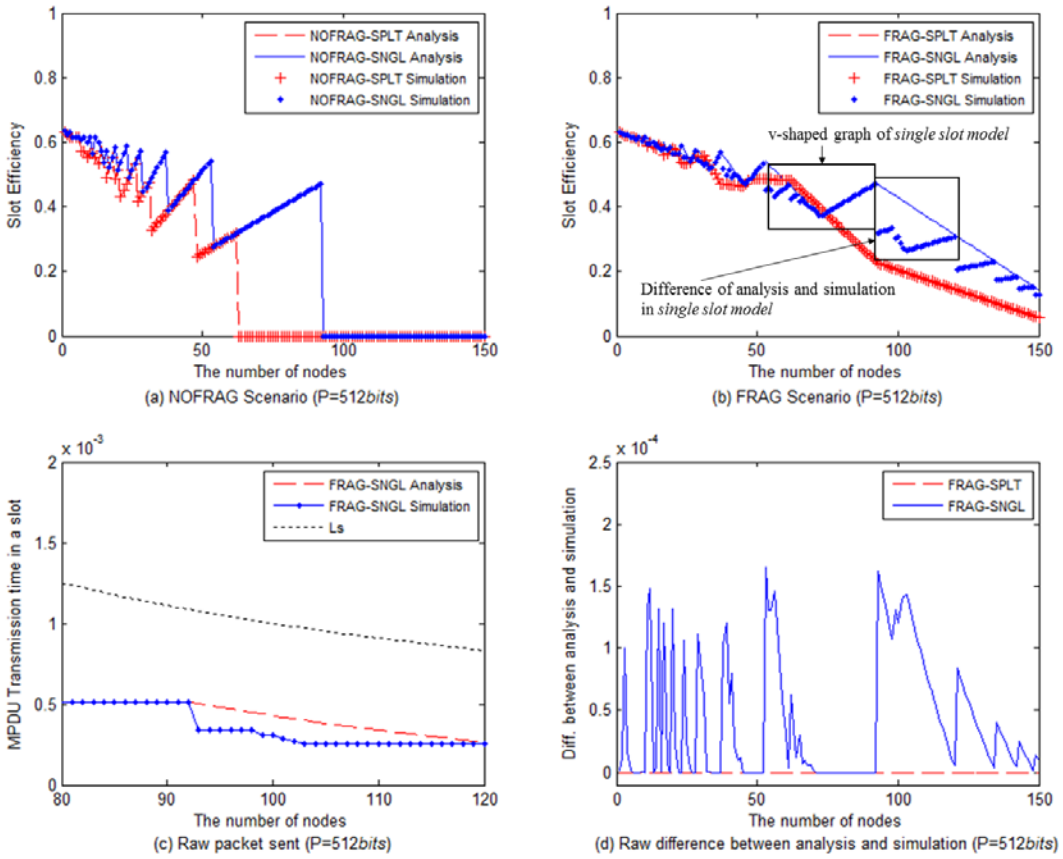

Fig. 9. Performance enhancement by adjusting firing offset

Fig. 9 shows the results of analysis and simulation when P=512 *bits*. **Fig. 9 (a)** and **(b)** are the results of NOFRAG scenario and FRAG scenario, respectively. The graph denoted by SPLT and SNGL in **Fig. 9** represent the split slot model and our proposed single slot model, respectively.

Fig. 9 (a) shows the single slot model (SNGL) enhances the capacity in the NOFRAG scenario. There is no capacity after 63 nodes in the split slot model, whereas the slot to MPDU efficiency is increased and packets are able to be transmitted until 92 nodes in our single slot model. The analysis and simulation show the same results. **Fig. 9 (b)** depicts the result FRAG scenario. The single slot model (our proposed) shows more efficient than the split slot model (the existing). There are two remarkable points: v-shaped graph and difference of analysis and simulation of FRAG-SNGL. First, the v-shaped graph shown in the section 53-92 nodes has the decreasing phase and the increasing phase of the efficiency. The decreasing phase is caused by the decrease of amount of the traffic sent more than the decrease of the slot length, and the increasing phase is caused by the fixed amount of traffic sent but the decreasing slot length.

Another remarkable point is the difference of analysis and simulation in the FRAG-SNGL scenario. In contrast in the almost of existing case, the results of analysis and simulation are the same, there are some differences in the FRAG-SNGL scenario. It is caused by a packet fragmentation due to the lack of enough slot length, in the section of 93-120 nodes in the **Fig. 9 (c)**, representatively. For example, slot length at n=100 nodes is 1 *ms* and the MPDU can be

transmitted only 428 *us* since the L_f and H need 276 *us* and 296 *us*, respectively. Thus the MPDU (512 *bits* in this case) is fragmented 428 *bits* and 84 *bits*, and the second fragment will be sent at the next slot. Totally 1536 bits can be sent during 5 slots in the simulation. We get the 0.428 for analysis and 0.3072 for simulation, at the 100 nodes. Therefore the difference is caused by the additional header. The result of analysis stands for the maximum efficiency and that of simulation is the actual transmitted throughput. **Fig. 9 (d)** shows the gap of analysis and simulation result of FRAG-SPLT and FRAG-SNGL. It shows the differences with maximal 165.4 *us* at 53 nodes.

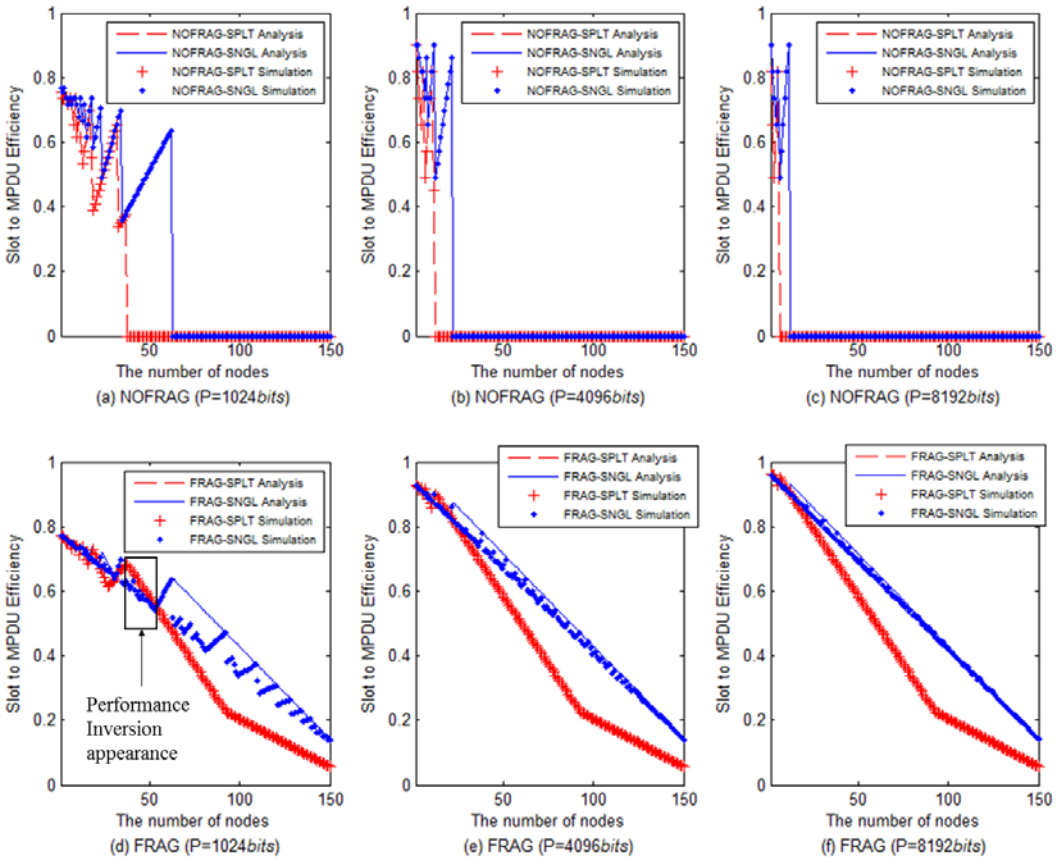

Fig. 10. Performance aspects of varying the packet size

We have conducted several results of varying the packet size shown in **Fig. 10**. The graphs of **Fig. 10 (a)**, **(b)**, and **(c)** show the results of the NOFRAG scenario. As the packet size increases, the available capacity of the number of nodes is decreased due to the lack of slot reminder. Our single slot model has better capacity than the existing split slot model. Then, the graphs of **Fig. 10 (d)**, **(e)**, and **(f)** depict the performance enhancement of FRAG scenario. Our model shows generally better performance than the existing model, and the gap of analysis and simulation of FRAG-SNGL is decreased because the ratio of slot reminder is also decreased by increasing MPDU size. Moreover, there are some inversed sections, which show FRAG-SPLT has better efficiency than that of FRAG-SNGL, in **Fig. 10 (d)** such as the section of 35-53 nodes. It is caused by increasing the overhead of T_{offset} field in the firing message.

The original DESYNC-TDMA splits the allocated slot into two half slots by the firing message which should be transmitted in the middle of the slot. The proposed scheme in this paper which is called firing offset adjustment scheme has an advantage of increasing the number of pacekts transmnitted in the allocated slot. Thus the capacity of the network is also increased. The proposed scheme has only a small amount of overhead to forward the offset how far the firing was moved from the original firing phase. However, the amount of overhead is much smaller than the amount of slot that can be secured by the proposed scheme, thus the impact of the weakness is negligible.

Moreover, the remarkable contribution of our offset adjustment scheme of firing message is that the firing message can be located any time in the slot using the `time_offset`. It provides the single slot model as well as the protocol flexibility of the firing message. For example, the firing messages of each node can be placed in the specific section in the time interval T such as control period if needed.

5. Conclusion

The DESYNC-TDMA is a bio-inspired MAC protocol suitable for WSN environments with its lower overhead than CSMA/CA-based MAC protocols. DESYNC-TDMA has a good property for self-organization in WSNs, which is PCO-based desynchronization. All nodes in a DESYNC-TDMA network can allocate their slots in a distributed manner without any central controller. However, as mentioned earlier, the split slot model (existing scheme) adapted in conventional DESYNC-TDMA-based schemes has the limitation that firing messages should be sent at the middle of slots to maintain the desynchronization process. Due to the firing scheme, each allocated slot is split into two half slots. It causes waste of slot utilization as discussed earlier. In this paper, an effective firing offset adjustment scheme has been proposed not only to overcome the slot waste problem but also to improve slot utilization performances. The proposed method can provide larger slot length than the original DESYNC-TDMA without breaking the rules of DESYNC-TDMA. To show the effectiveness of the proposed method, we derived the performance models of the existing and the proposed methods for two scenarios whether the fragmentation is applied or not. Thorugh the analysis and the simulation experiments, peformance comparisions between the existing and the proposed schemes have been carried out. The results showed that the proposed scheme can minimize the wasted slots, and hence proivde much better efficiency of slot utilization to deliver packets than the existing schemes.

As everything including sensors are being connected to the Internet through wireless networks, the distributed and autonomous networking operations of sensors in WSNs should be needed much more than before. It is expected the DESYNC-TDMA can be one of the promising solutions to achieve the requirements. And, it is also expected that the proposed method can be utilized effectively in such a future WSN environments.

Most of existing works including the paper have considered the single hop environments. In practical WSN environments, multihop packet delivery may be required. The extended DESYNC-TDMA for such multihop topologies has been proposed [20]. However, since the method is also based on the original DESYNC-TDMA, which apapts the split slot model, it still has the slot waste problem as discussed earlier. The proposed scheme is expected to be a good solution to overcome the problem that the extended DESYNC-TDMA has. The extension of the proposed method to multihop packet delivery environements requires for further study.

References

[1] P. Rawat, K. D. Singh, H. Chaouchi, and J. M. Bonnin, "Wireless sensor networks: A survey on recent developments and potential synergies," *The Journal of Supercomputing*, Vol.68, No.1, pp 1–48, Apr. 2014. Article (CrossRef Link).

[2] V. K. N. Lau and Y.-K. Kwok, "CHARISMA: a novel channel-adaptive TDMA-based multiple access control protocol for integrated wireless voice and data services," in *Proc. of IEEE WCNC*, pp. 507-511, Sep. 2000. Article (CrossRef Link).

[3] C. Zhu and M. S. Corson, "A Five-Phase Reservation Protocol (FPRP) for Mobile Ad Hoc Networks," *Wirel. Netw.* Vol. 7, No. 4, pp. 371-384. Sep. 2001. Article (CrossRef Link).

[4] I. Jawhar and J. Wu, "QoS Support in TDMA-Based Mobile Ad Hoc Networks," *J. Comput. Sci. & Technol*., Vol. 20, No. 6, pp. 797-810. Nov. 2005. Article (CrossRef Link).

[5] S. Park and D. Sy, "Dynamic control slot scheduling algorithms for TDMA based Mobile Ad Hoc Networks," in *Proc. of IEEE MILCOM '08*, Nov. 2008. Article (CrossRef Link).

[6] L. Schenato and F. Fiorentin, "Average TimeSynch: A consensus-based protocol for clock synchronization in wireless sensor networks," *Automatica*, Vol. 47, No. 9, pp. 1878-1886, Sep. 2011. Article (CrossRef Link).

[7] J. Degesys, I. Rose, A. Patel, and R. Nagpal, "DESYNC: self-organizing desynchronization and TDMA on wireless sensor networks," in *Proc. of IEEE IPSN '07*, Apr. 2007. Article (CrossRef Link).

[8] F. Dressler and O. B. Akan, "A Survey on Bio-inspired Networking," *Computer Networks Journal (Elsevier)*, vol. 54, no. 6, pp. 881-900, Apr. 2010. Article (CrossRef Link).

[9] G. Werner-Allen, G. Tewari, A. Patel, R.Nagpal, and M. Welsh, "Firefly-Inspired Sensor Network Synchronicity with Realistic Radio Effects," in *Proc. of ACM SenSys '05*, Nov. 2005. Article (CrossRef Link).

[10] R. Mirollo and S. Strogatz, "Synchronization of pulse-coupled biological oscillators," *SIAM Journal of Applied Math*, Vol. 50, No. 6, pp. 1645-1662, Dec. 1990. Article (CrossRef Link).

[11] C. Lien, S. Chang, C. Chang, and D. Lee, "Anchored desynchronization," in *Proc. of IEEE INFOCOM '12*, Mar. 2012. Article (CrossRef Link).

[12] Y. Taniguchi, "Desynchronization-based Weighted Scheduling Adaptive to Traffic Load for Wireless Networks," in *Proc. of IEEE I4CT '14*, Sept. 2014. Article (CrossRef Link).

[13] Y. Kim, H. Choi, and J. Lee, "A Bioinspired Fair Resource-Allocation Algorithm for TDMA-Based Distributed Sensor Networks for IoT," *Int. J. Distrib. Sens. N.*, Vol. 2016, Apr. 2016. Article (CrossRef Link).

[14] K. Kim, B.-h. Roh, B.-s. Roh, and J. Choi., "Enhanced DESYNC-TDMA algorithm for efficient packet delivery in distributed MANETs," in *Proc. of BICT 2014*. Dec. 2014. Article (CrossRef Link).

[15] B.-s. Roh, D. Kum, T. Kim, J. Choi, K. Kim, and B.-h. Roh, "Efficient and adaptive resource allocation scheme for self-organizing DESYNC TDMA network," in *Proc. of BICT 2014*. Dec. 2014. Article (CrossRef Link).

[16] B.-s. Roh, M. Han, M. Hoh, H. Park, K. Kim, and B.-h. Roh, "Distributed Call Admission Control for DESYNC-TDMA in Mobile Ad Hoc Networks," in *Proc. of BICT 2015*. Dec. 2015. Article (CrossRef Link).

[17] C. Mühlberger, "Energetic and Temporal Analysis of a Desynchronized TDMA Protocol for WSNs," 8. Fachgespräch Sensornetzwerke, FGSN 2009.

[18] D. Young, "USAP: A Unifying Dynamic Distributed Multichannel TDMA Slot Assignment Protocol," in *Proc. of IEEE MILCOM '96*, Oct. 1996. Article (CrossRef Link).

[19] IEEE 802.11 Working Group, "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," in *Proc. of IEEE Std 802.11-2012 (Revision of IEEE Std 802.11-2007)*, pp.1-2793, Mar. 2012. Article (CrossRef Link).

[20] C. Mühlberger, and R. Kolla, "Extended desynchronization for multi-hop topologies," *Institut für Informatik*, Universität Würzburg, Tech. Rep. 460, Jul. 2009. Article (CrossRef Link).

Kwangsoo Kim received a B.S. degree in Information and Computer Engineering from Ajou University, Suwon, Korea, in 2009, and a Ph.D. degree in Computer Engineering from Ajou University, Suwon, Korea in 2017. Since January 2017, he works with LIG Nex1, a leading defense company in Korea, as a researcher. His research interests include modeling and simulation, tactical networks, network security and cyber warfare, MANET, mobile multimedia networking, and network QoS.

Seung-hun Shin received a B.S. degree in Information & Computer Engineering from Ajou University, Suwon, Korea, in 2000, and M.S. and Ph.D. degrees in Information & Communication Engineering from Ajou University, Suwon, Korea, in 2002 and 2011, respectively. From 2011 to 2016, he was a Lecture Professor in the department of Software Convergence Technology, Ajou University. Since March 2016, he has been an Assistant Professor in the Da-san University College, Ajou University. His research interests include software test automation, network intrusion detection, mobile multimedia networking, and IoT (Internet of Things)

Byeong-hee Roh received a B.S. degree in Electronics Engineering from Hanyang University, Seoul, Korea, in 1987, and M.S. and Ph.D. degrees in Electrical Engineering from Korea Advanced Institute of Science and Technology (KAIST), Daejeon, Korea, in 1989 and 1998, respectively. From 1989 to 1994, he was with Telecommunication Networks Laboratory, Korea Telecom, as a researcher. From February 1998 to March 2000, he worked with Samsung Electronics Co., Ltd., Korea, as a Senior Engineer. Since March 2000, he has been with the department of Software and Computer Engineering, and the department of Computer Engineering, Graduate School, Ajou University, Suwon, Korea, where he is currently a professor. During 2005, he was a visiting associate professor at Dept. of Computer Science, State University of New York, at Stony Brook, New York, USA. During 2014, he was an adjunct researcher at ADD (Agency for Defense Development), Korea. His research interests include mobile multimedia networking, network QoS, IoT (Internet of Things) platform and services, network security and cyber warfare, modeling and simulation, and military communications.