IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received April 14, 2021, accepted April 23, 2021, date of publication April 30, 2021, date of current version May 14, 2021.

*Digital Object Identifier 10.1109/ACCESS.2021.3076861*

# Optimal Initial Synchronization Time in the Minimal 6TiSCH Configuration

**APOSTOLOS KARALIS[1], DIMITRIOS ZORBAS[2], (Member, IEEE), AND CHRISTOS DOULIGERIS[1], (Senior Member, IEEE)**

[1]Department of Informatics, University of Piraeus, 18534 Piraeus, Greece
[2]Department of Computer Science, School of Engineering and Digital Sciences, Nazarbayev University, 010000 Nur-Sultan, Kazakhstan

Corresponding author: Apostolos Karalis (akaralis@unipi.gr)

This research has been financially supported by General Secretariat for Research and Technology (GSRT) and the Hellenic Foundation for Research and Innovation (HFRI) (Scholarship Code:2160). This work has also been partially supported by UPRC (University of Piraeus Research Center) and by COSMOTE through a PEDION24 grant.

**ABSTRACT** 6TiSCH is an emerging networking technology proposed by IETF for the Industrial Internet of Things. As a result of its standardization effort, IETF has proposed the so-called *minimal 6TiSCH configuration*, which sets the minimal requirements for building functional 6TiSCH networks. However, this minimal configuration provably leads to large synchronization times and, consequently, to a waste of energy every time a node desires to join the network. In this paper, in order to optimize this *initial synchronization* process, we study the effect of the channel *scan period* on the initial synchronization time and on the energy consumption. The scan period is the time a node spends on listening for network advertisements on a specific channel as it scans the available channels in the network. Our study includes the mathematical modeling and analysis of the initial synchronization process, its algorithmic representation as well as experiments in a 12-node testbed. The theoretical results demonstrate that by optimally setting the scan period, we can achieve an up to 48.37% and 47.10% reduction in the average initial synchronization time compared to the default scan periods of Contiki-NG and OpenWSN respectively. Both the algorithmic approach and the experiments exhibit almost identical results, thus, confirming the performance improvement. Furthermore, our experiments demonstrate an almost linear relation between the average initial synchronization time and the average energy consumption.

**INDEX TERMS** 6TiSCH, TSCH, IEEE802.15.4, minimal 6TiSCH configuration, Industrial Internet of Things (IIoT), initial synchronization, scan period.

## I. INTRODUCTION

At first, the need of high reliability and low-power consumption in industrial applications of the Internet of Things (IoT) led to the design of mechanisms such as the ''Time Slotted Channel Hopping'' (TSCH) for the IEEE802.15.4 physical layer [1]. Later, the need for an open-standard, IPv6-enabled solution that realizes the Industrial Internet of Things led the Internet Engineering Task Force (IETF) into developing the ''IPv6 over the TSCH mode of IEEE 802.15.4'' (6TiSCH) technology, which bridges the gap between IEEE802.15.4-TSCH and IPv6 [2]. During the standardization process of 6TiSCH, the so-called minimal 6TiSCH configuration [3] has been defined, which specifies the minimal requirements for building functional 6TiSCH networks. The TSCH version of IEEE802.15.4 is part of embedded operating systems [4], [5] and has thoroughly been tested in various deployment scenarios [6]–[8].

In 6TiSCH, a node that intends to join a network must first synchronize with this network. This initial synchronization is performed through the network advertisement frames, called Enhanced Beacons (EBs). To receive an EB, the node scans the available channels (e.g., 16 channels when all the channels of the 2.4GHz ISM band are used). Because of the multiple orthogonal channels and the channel hopping mechanism of IEEE802.15.4-TSCH, finding an EB may take a considerable amount of time, which in turn leads to an increased energy consumption since the node needs to constantly have its radio on during the scan. A shorter synchronization time can be achieved using a higher EB transmission

The associate editor coordinating the review of this manuscript and approving it for publication was Maurice J. Khabbaz.

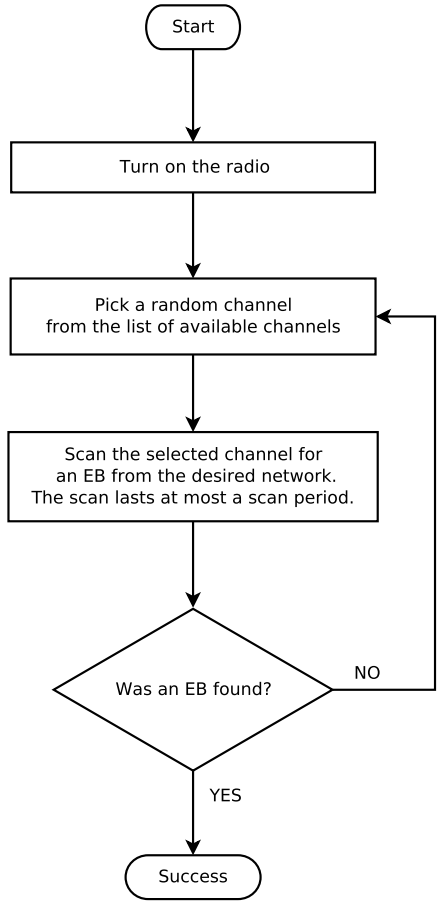


**FIGURE 1.** The flow chart of the initial synchronization procedure.

rate, but at the cost of a higher energy consumption [3]. Additionally, any increase in the EB rate should be made taking into account the possibility of an increase in the collisions of the neighboring EB transmissions, which will definitely affect the performance of the initial synchronization procedure [9], [10]. It is worth noting that the transmission of EBs on a predefined static channel is not an acceptable solution since it comes in contrast to the channel-hopping mechanism of IEEE802.15.4-TSCH, and in general, to the standard operation of the protocol. Moreover, it may cause a node to fail to join the network when, for instance, this channel suffers from heavy interference. For the same reason, a node should avoid to constantly listen for EBs on a randomly selected channel.

To make the description of the initial synchronization procedure more intelligible to the reader, we present in Fig. 1 its flow chart. As we can see in this figure, the following take place after the initial synchronization procedure is triggered by the TSCH link-layer protocol on the side of a node that wishes to join a 6TiSCH network. Initially, the node turns on its radio and selects a random channel to scan for EBs. The scan of the channel lasts at most a scan period. If an EB is successfully received within this period then the procedure completes, otherwise a new channel is randomly selected for scanning.

The use of the minimal 6TiSCH configuration worsens the problem of large synchronization times, since it provides limited resources for network advertising. In the direction of mitigating the problem of large initial synchronization times in the context of the minimal 6TiSCH configuration, we examine the impact of the scan period on the initial synchronization time and on the related energy consumption. It is worth noting that the value of the scan period is defined neither by the TSCH standard nor by the minimal 6TiSCH configuration. As a consequence, previous studies and current implementations consider an arbitrary scan period value. To the best of our knowledge, this is the first paper that studies the impact of the scan period, not only in the context of the minimal 6TiSCH configuration but generally in the field of IEEE802.15.4-TSCH networks. Herein, we focus on the minimal 6TiSCH configuration in order to support the 6TiSCH standardization efforts. We approach the problem of the optimal scan period assignment through an extensive mathematical and algorithmic analysis as well as through experiments in a testbed. We note that although the analysis provided in the current paper is limited to the minimal 6TiSCH configuration which uses a single slot for advertisements, it can be used as a theoretical basis for the analysis of more complex multi-slot approaches that have been proposed in the literature, such as in [11], [12].

In summary, the contributions of the paper are the following:

1) We compute the optimal scan period in the minimal 6TiSCH configuration. More precisely, we demonstrate that the optimal scan period is determined by the product of the number of available channels and the slotframe duration.
2) We demonstrate that using the optimal scan period, we can achieve an up to 48.37% and 47.10% lower average synchronization time compared to the default scan periods of Contiki-NG and OpenWSN respectively. We can also achieve an almost proportional reduction in the energy consumed during the initial synchronization process. It is noted that the minimization of the energy consumed in the initial synchronization is crucial for the feasibility of the initial synchronization when the nodes are powered by capacitors whose energy is limited and regularly replenished by a renewable energy resource. In this case, there is a hard limit on the amount of energy they can spend before joining a network.
3) By providing the optimal scan period, we advance the 6TiSCH standardization efforts in the direction of optimizing the node joining process, and, consequently, the network formation process.
4) In the context of 6TiSCH applications with a predefined number of channels and slotframe duration, we enable the manufacturers or the administrators to optimize the initial synchronization process using the provided optimal scan period. In addition, through our model, they can also calculate the average expected initial synchronization time.

5) In 6TiSCH applications where the nodes do not know the number of channels and the slotframe duration apriori, they can calculate the optimal scan period after the initial synchronization. Consequently, in this case, the nodes can not exploit the optimal scan period in their first synchronization attempt. However, they can exploit it during a rejoin attempt, which for example can take place in the case of a mobile node that voluntarily or involuntarily connects and disconnects from the network as it moves away or approaches the network respectively. Therefore, we enable a node to speed up its rejoin attempts even if it does not know the number of channels and the slotframe duration at boot time.

The remainder of the paper is organized as follows. Section II provides an introduction to the minimal 6TiSCH configuration and the IEEE802.15.4-TSCH networks, while Section III highlights the recent related work. In Section IV, we present an extensive mathematical and algorithmic analysis of the problem of the optimal scan period assignment and we discuss the theoretical results. Section V presents the experimental results. Finally, Section VI concludes the paper and presents ideas for future work.

## II. MINIMAL 6TiSCH CONFIGURATION

### A. MINIMAL SCHEDULE

Since 6TiSCH is based on the IEEE802.15.4-TSCH link layer protocol, the communication between the nodes takes place according to a schedule built on a slotframe; that is, a group of slots repeating over time. Each slot is uniquely identified by its *slot offset*, which is the position of the slot in the slotframe, counting from zero. The schedule does not explicitly define the radio channel where a node transmits or listens in a slot, but it defines a *channel offset*. The channel offset is used for the calculation of the radio channel at each repetition of the slot, a process that is described in the next subsection. This schedule can be represented as a matrix where the columns are the slots and the rows are the channel offsets, while each *cell* of the matrix is an atomic unit of communication resource.

The minimal 6TiSCH configuration uses a single slotframe, the length of which is selected based on the particular network requirements. The recommended timeslot template of the minimal 6TiSCH configuration is the default template of the IEEE802.15.4-TSCH standard for the 2.4GHz ISM band, which means that the length of a slot is 10ms. Regarding the resource allocation, the minimal 6TiSCH configuration allocates only a single cell, which is called a *minimal cell*. The minimal cell can be any pair of a slot offset and a channel offset; nevertheless, the pair $(0, 0)$ is commonly used. The minimal cell, which can be used for any type of the link layer traffic, is used as a reference point where the nodes can exchange the information required for building a functional 6TiSCH network (i.e., network advertising and routing). An example of the minimal schedule proposed by the minimal 6TiSCH configuration is presented in Fig. 2. The unused

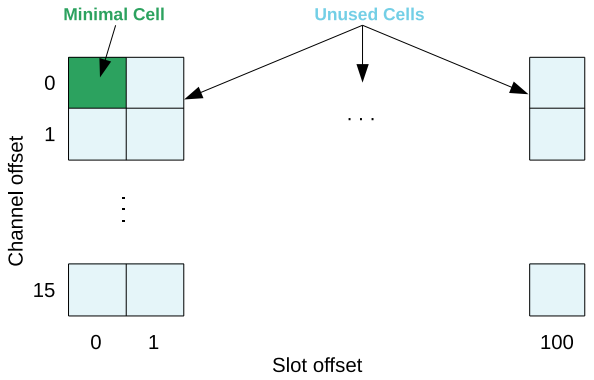


**FIGURE 2.** The schedule of the minimal 6TiSCH configuration when using a slotframe of 101 slots and 16 channels.

cells are typically allocated to data transmissions by a data scheduling mechanism.

### B. CHANNEL HOPPING

To minimize the negative effects of multipath fading and interference, IEEE802.15.4-TSCH uses a slow channel hopping mechanism that maps a cell to different physical channels as the slotframe repeats. More specifically, the channel $CH$ to which a cell is mapped is recalculated at each repetition of the slot containing the cell, through the following formula [1]:

$$CH = HS[(ASN + channelOffset) \bmod HS_{len}], \qquad (1)$$

where $HS$ is the (channel) Hopping Sequence, that is, a list containing the available channels in a defined order; $ASN$ is the Absolute Slot Number, which indicates the number of slots elapsed since the start of the network; $channelOffset$ is the channel offset of the cell, and $HS_{len}$ is the length of the channel hopping sequence. It should be noted that the numbering of the elements in the channel hopping sequence begins from 0. Furthermore, there are as many channel offsets as the available channels, a condition that according to the standard [1] results in a unique channel for each of the cells of a slot. However, this condition is sufficient only when the channel hopping sequence contains each of the available channels once, while the standard allows the use of a channel hopping sequence that contains a channel multiple times. In the latter case, the condition is met only if the distance between the positions of two consecutive appearances of a channel in the channel hopping sequence is equal to the number of the available channels. However, if we call $C$ the number of available channels, then in such a sequence we have a cyclical repetition of the first $C$ elements, which from the perspective of the channel hopping leads to the same results as the sub-sequence of the first $C$ elements, and, thus, there is no reason to use such a channel hopping sequence. A later redefinition of Eq. (1) [13] solves the previous issue by strictly defining the use of channel hopping sequences that contain each of the available channels once, which also means that the length of the channel hopping sequence is always equal to the number of the available channels. Such a channel

hopping sequence is the default channel hopping sequence defined by the standard and it depends on the number of the available channels. The default channel hopping sequence is a pseudo-randomly shuffled set of the available channels.

Another issue that must be taken into account regarding the channel hopping mechanism is the slotframe length. To assure that a cell cyclically hops over the available channels as the slotframe repeats, the slotframe size and the number of available channels must be co-primes [14]. Assuming that this condition is met, the cell completes a hopping cycle on the available channels $C$, in $C$ consecutive slotframes. We call this time the *channel hopping period*. The minimal 6TiSCH configuration recommends the use of all the available channels of the 2.4GHz ISM band (i.e., all the 16 channels), and the corresponding default channel hopping sequence. Therefore, the minimal cell completes a hopping cycle on the available channels in 16 slotframes; that is, the channel hopping period is 16 slotframes.

It is useful to note that, due to the channel hopping mechanism, a cell hops cyclically over the available channels in a defined order; This order (or sequence of channels) can be expressed by the list containing in chronological order the channels mapped to the cell at its first $C$ appearances, that is, in the first channel hopping period. Using the slot offset and the channel offset of the cell, and assuming that the number of slots in the slotframe is $S$, we can calculate the ASN of the $i$-th appearance of the cell through the mathematical expression $(i-1)S + slotOffset$. Consequently, the channel at the $i$-th appearance can be calculated through the expression:

$$HS[((i-1)S + slotOffset + channelOffset) \bmod C].$$

Therefore, the list $W$ according to which the cell hops cyclically over the available channels is given by Eq. (2). Assuming that the first element of $W$ is assigned the index 0, it is implied that the channel of the cell at its $i$-th appearance is equal to $W[(i-1) \bmod C]$.

$$\begin{aligned} W = [ & \\ & HS[(slotOffset + channelOffset) \bmod C], \\ & HS[(S + slotOffset + channelOffset) \bmod C], \\ & \ldots \\ & HS[((C-1)S + slotOffset + channelOffset) \bmod C] \\ & ]. \end{aligned} \tag{2}$$

### C. ENHANCED BEACON TRANSMISSIONS

Initially, a network that is based on IEEE802.15.4-TSCH, such as a 6TiSCH network, contains only the Personal Area Network (PAN) coordinator. To advertise the presence of the network, in order to allow new nodes to join, the PAN coordinator sends EBs. Depending on the network design, a node that joins the network may also send EBs. For convenience, we will use the term *advertisers* to refer to the nodes that send EBs. The standard does not define the rate at which an advertiser sends EBs, but it states that the EB rate is configured by a higher layer as appropriate taking into account the density of the nodes, the desired time for network formation, and the energy devoted to network formation.

Similarly, the minimal 6TiSCH configuration recommends that an advertiser should send EBs at a rate of 1 EB per *Enhanced Beacon period* (*EB period*), where the value of the EB period is set depending on the network requirements. There are two ways for a node to autonomously achieve an EB rate of 1 EB per EB period: (a) a static EB scheduling where the node schedules each next EB transmission in such a way that the distance from the last EB is fixed to be equal to the EB period, and (b) a dynamic random (or probabilistic) based scheduling with an expected distance between two consecutive EB transmissions equal to the EB period. The static scheduling has a severe drawback. Since multiple nodes may start transmitting EBs at the same slot, two or more neighbors of a node may continuously transmit EBs in the same slots; that is, some or all the neighboring EB transmissions of a node may continuously collide resulting in a high synchronization time or, in the worst case scenario, in an inability to join the network. In the context of this paper, we assume that the dynamic random based scheduling of EBs is used.

Table 1 summarizes the recommended TSCH settings for the minimal 6TiSCH configuration.

**TABLE 1. The recommended TSCH settings for the minimal 6TiSCH configuration.**

| Parameter | Recommended Value |
|---|---|
| Slotframe Length | Defined based on the requirements on bandwidth and energy consumption. |
| Number of used cells | 1 (commonly the cell (0,0)). |
| Number of unused cells | All the remaining cells. |
| Max MAC retransmissions | 3 (4 transmission attempts). |
| Timeslot template | IEEE Std 802.15.4 default. |
| Enhanced Beacon Period | Defined based on the requirements on the initial synchronization time and the energy consumption. |
| Number of used channels (2.4 GHz O-QPSK PHY) | IEEE Std 802.15.4 default (16). |
| Channel Hopping sequence (2.4 GHz O-QPSK PHY) | IEEE Std 802.15.4 default ([16, 17, 23, 18, 26, 15, 25, 22, 19, 11, 12, 13, 24, 14, 20, 21]). |

## III. RELATED WORK

The problem of the initial synchronization has been studied in the last few years from different perspectives [15]–[17]. However, none of these studies takes into account the impact of the scan period on the joining time and on the energy consumption. In this paper, we carry out a complete analysis of the impact of the scan period in the context of the minimal 6TiSCH configuration. At this point, to avoid confusion, we must note that in many cases in the literature, the initial synchronization procedure is considered identical to the joining procedure. However, for the secure joining of a node, a security protocol is required to run after the initial synchronization, to meet the needs of authentication,

authorization and security parameter distribution. In this paper, we focus only on the synchronization part of the joining procedure.

Assuming a random transmission policy where the EB transmissions are randomly allocated to a subset of the channel offsets of a single slot, a number of observations has been made. Specifically, it has been observed that the use of a higher number of channel offsets for EBs decreases the joining time, since it decreases the number of EB collisions, but it decreases the resources available for data transmissions [15]. This consequence has also been validated by other authors [17]. The same studies also observe that negotiating the EB channels, instead of using random ones, leads to increased overhead and, thus, it should be avoided. Nevertheless, the advertisers can first listen to the medium and choose non-congested channels in order to improve the joining times [16]. However, the last study does not include results to show the impact of the channel sensing activity on the energy consumption of the advertisers.

More complex EB transmission policies have also been studied in the literature. Guglielmo *et al.* [18] formulate an optimization problem to calculate the optimal cells where an advertiser should transmit EBs with an ultimate goal to minimize the average joining time. However, because their approach may lead to a large number of collisions and may require an advertiser to simultaneously transmit EBs on multiple channels, they propose an alternative approach, where the optimal EB cells are used as a pool from which each advertiser randomly selects a single cell for its EBs. Some other approaches propose collision-free algorithms [9], [12], [19] by allocating extra EB cells. Apparently, by increasing the number of EB cells, the probability a joining node to successfully receive an EB also increases. However, the reservation of extra EB cells leads to a reduction in the number of cells available for data transmissions. Moreover, assuming the minimal configuration of 6TiSCH, the problem of EB scheduling is translated to a slotted-ALOHA allocation problem [10]. It has been observed that because of the high collision probability, the minimal 6TiSCH configuration may lead to very large joining times and to scalability problems [10], [20].

Scheduling methods that adapt to the network size but still reserve more than one advertisement slot have also been explored [11], [20]. In [11], a slotframe structure where an advertiser can dynamically decide the number of beacons to send is proposed. The decision is based on a fuzzy logic mechanism and on the neighbors' status. In [20], extra shared slots are dynamically allocated based on the neighbors' activity. A different approach is followed by Duquennoy *et al.* [21] where the slot allocation and scheduling is decided based on the nodes' unique identifiers (MAC addresses) to avoid a burst of collisions. However, the final schedule for advertisement purposes still uses shared slots (for TSCH EBs and routing control messages).

Finally, the joining time of a node can also be decreased by properly adjusting the EB transmission rate [10], [11], [22]–[25]. In these studies, the structure of the slotframe and the position of the EB cells are fixed but the EB transmission rate is adapted based on the estimated channel congestion status. These solutions can apparently improve the joining times and the energy consumption, however, they are still sub-optimal.

**TABLE 2. A summary of the basic notations and their meaning.**

| Notation | Meaning |
|---|---|
| $P_{eb}$ | Probability of an EB transmission in a minimal cell. |
| $P_{sr}(x)$ | Probability that an EB is transmitted without errors in channel $x$. |
| $\beta(x)$ | The product of $P_{eb}$ and $P_{sr}(x)$. |
| $\overline{\beta}$ | The average value of a series of $\beta$ values. |
| $T_{scan}$ | The scan period. |
| $T_{sf}$ | The duration of the slotframe. |
| $n$ | The ratio of $T_{scan}$ to $T_{sf}$. |
| $S$ | The number of slots in the slotframe. This is also related to $T_{sf}$. |
| $C$ | The number of available radio channels. |
| $T_{eb}$ | The transmission time of an EB. |
| $T_{slot}$ | The duration of a slot, which, following the recommended settings of minimal 6TiSCH configuration is 10ms. |
| $T_{txOffset}$ | The predefined distance between the start of a slot and the time where the transmissions of the slot start, which, following the recommended settings of minimal 6TiSCH configuration is $2120\mu s$. |

## IV. OPTIMAL SCAN PERIOD ASSIGNMENT

In this section, we analyze the problem of the optimal scan period assignment in the Minimal 6TiSCH configuration assuming a random based scheduling of EBs. We make an extensive analysis that includes two different approaches: (a) a mathematical model and (b) an algorithmic approach. Table 2 will help the reader follow the notations used in the following paragraphs.

### A. MATHEMATICAL MODEL

Let us suppose that a node wishes to join a 6TiSCH network and, therefore, it has to initially synchronize with this network. For this reason, it starts scanning the channels in order to find an EB. In the context of the minimal 6TiSCH configuration, the node can find an EB only in the minimal cell. This means that, within one slotframe, the node has exactly one chance to find an EB. We note that, according to the standard of IEEE802.15.4-TSCH, the transmissions in a slot, or equivalently, in a cell, typically start at a predefined time point called *txOffset*. Herein, for presentation purposes, we define the term *EB transmission point* (or, simply, *EB point*) for the txOffset of the minimal cell.

Without loss of generality, we assume that the scan process is divided into steps of one slotframe length each. As the synchronization attempt can start at any time point in the slotframe, the starting point of a step may be (a) before the EB point, (b) at the EB point, or (c) after the EB point. In the

first two cases, the step covers the EB point of the slotframe where it starts, while in the last case it covers the EB point of the next slotframe. Fig. 3 presents an example for each case. Regardless of the start point of a step, we consider that the step always covers the transmissions of a minimal cell since it surely covers the start point of the transmissions of a minimal cell. For convenience, we will use the phrase "the EB channel of the step" to refer to the channel of the minimal cell containing the EB point covered by the step.

To find an EB during a step, the following conditions must be simultaneously satisfied:

i) An EB must be transmitted. The probability that an EB is transmitted within a step is equal to the probability of an EB transmission in a minimal cell. We call this probability $P_{eb}$. It should be noted that the factors that affect this probability are the EB scheduling, and, more specifically, the EB rate, the frame buffering at the senders, and the number of neighboring advertisers (i.e., the node density).

ii) The joining node must listen to the channel where the EB is transmitted; that is, to the channel where the minimal cell has been mapped according to the channel hopping mechanism. For this to happen, it is required that (a) the node has selected the channel where the transmission takes place, and (b) the channel change has been completed before the start of the transmission. The channel that a node will listen to is randomly selected from the list of available channels.

iii) The reception of an EB must be completed without errors. The probability of this happening depends on the external interference in the reception channel and on the likely collisions between the neighboring transmissions on the reception channel. To refer to this probability in a channel $x$ of the network, we define the function $P_{sr}(x)$.

Let $T_{scan}$ be the scan period, $T_{switch}$ the channel switch delay, that is, the time consumed when changing to a new channel, and $T_{channel}$ the total time spent on a selected channel during the scan. When the selected channel is the same as the previous selection, then no channel switch occurs and, thus, $T_{channel} = T_{scan}$, otherwise $T_{channel} = T_{switch} + T_{scan}$. However, since the channel switch delay is practically a negligible quantity, usually a few microseconds [26], we will consider that $T_{channel} = T_{scan}$ in both cases.

Subsequently, we define $T_{sf}$ as the duration of the slotframe, which is equal to the duration of a step. Moreover, we define that the ratio of $T_{scan}$ to $T_{sf}$ is equal to $n$, that is:

$$n = \frac{T_{scan}}{T_{sf}}, \quad n \in \mathbb{R}_{>0}. \tag{3}$$

Assuming that the number of slots in the slotframe is $S$ and the duration of a slot is $T_{slot}$, then $T_{sf}$ is equal to the product of $S$ and $T_{slot}$. If we call $C$ the number of available channels in the network, then as we mentioned in Section II-B, $S$ and $C$ must be co-primes. This restriction is applied in our analysis.

Furthermore, as we also mentioned in Section II-B, a cell hops cyclically over the available channels following a specific sequence described by the list $W$ of Eq. (2). Since the initial synchronization process may start at any time during the lifetime of the network, the EB channel of the first step may be any of the available channels, and hence, it may be located at any position in the list $W$. If we assume that the EB channel of the first step is at the position with index $y$ in the list $W$, where $y \in \{0, 1, \ldots, C-1\}$, then the EB channel of the $k$-th step, where $k \in \mathbb{N}_{\neq 0}$, is given by the following function:

$$X(k, y) = W[(y + k - 1) \bmod C]. \tag{4}$$

Because the EB channel of a step affects the probability $Psr$, we must take into account all the possible cases about the position of the EB channel of the first step in the list $W$ that is, for each $y \in \{0, 1, \ldots, C-1\}$. For presentation reasons, we will use the phrase "index of a channel" to refer to the index of the position of the channel in the list $W$.

Herein, we define the objective function $\beta$ that maps a channel $x$ to the product of $P_{eb}$ and the $P_{sr}$ value of $x$, that is:

$$\beta(x) = P_{eb} P_{sr}(x). \tag{5}$$

It is obvious that the higher the $P_{eb}$ and the higher the $P_{sr}$ values the higher the values of the $\beta$ function. However, it should be mentioned that a higher $P_{eb}$ may increase the collisions in the minimal cell and, thus, it may reduce the $P_{sr}$ values.

Regarding the channel selections, since the channels are selected randomly from the list of available channels, each selection is independent of the previous ones. Obviously, during a step, the probability of selecting the channel where the EB is transmitted is $1/C$. Consequently, if we assume that we are in the $k$-th step of the synchronization process, the conditional probability of finding an EB, given that the EB channel of the first step has index $y$ is:

$$P_{step}(k, y) = \frac{1}{C} \beta\big(X(k, y)\big). \tag{6}$$

It is implied that the probability of finding an EB at a random step is not affected by the previous steps. That is, the steps are independent. The probability that the synchronization is achieved at step $k$ is equal to the probability that an EB is not found at the first $k-1$ steps but it is found at step $k$. Next, in order to create the mathematical model of the initial synchronization, we divide the calculations into three cases:

(a) $n \in (0, 1)$, that is, the scan period is shorter than the duration of a step,

(b) $n \in \mathbb{N}^*$, i.e. the scan period is an integer multiple of the step duration, and,

(c) $n \in \mathbb{R}_{>1} - \mathbb{N}$, that is, the scan period is longer than the step duration, but it is not an integer multiple of the step duration.

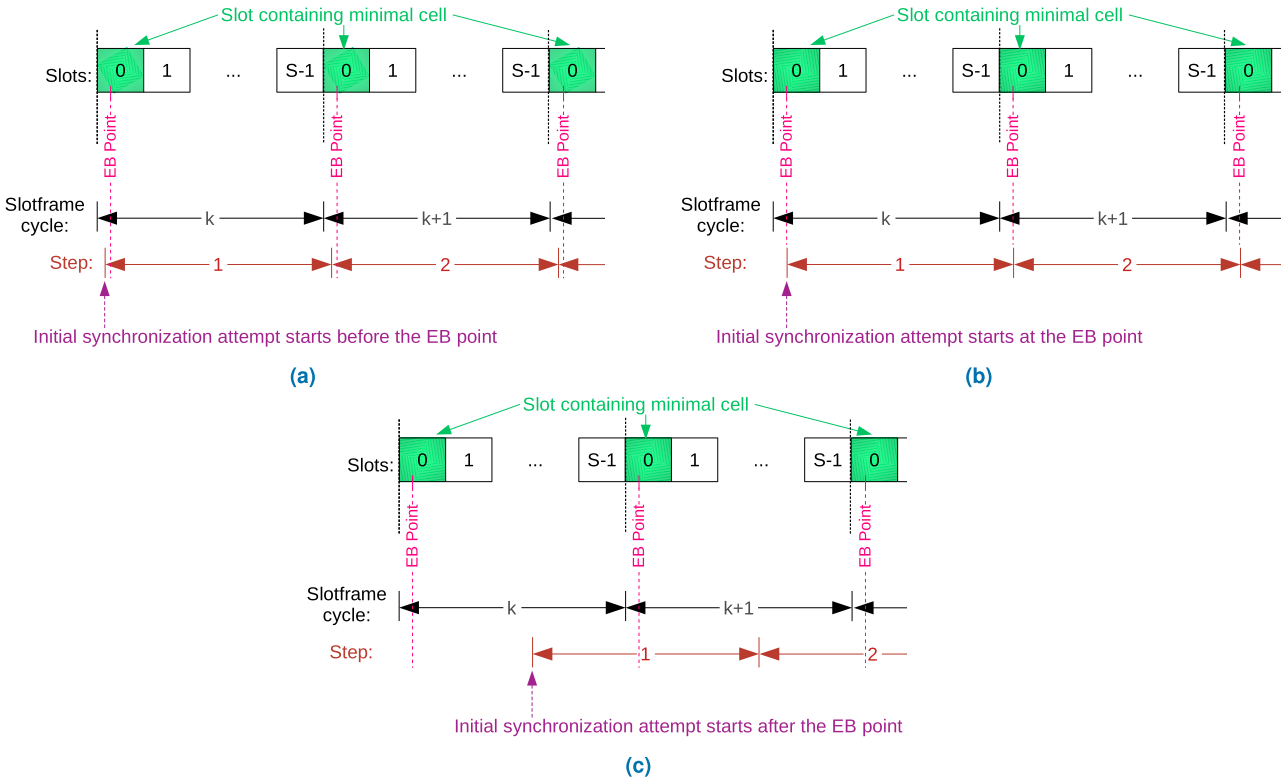


**FIGURE 3.** An example about the starting point of an initial synchronization attempt that randomly begins at the $k$-th cycle of a slotframe containing $S$ slots when the starting point of the attempt is (a) before the EB point of this slotframe cycle, (b) at the EB point of this slotframe cycle, and (c) after the EB point of this slotframe cycle.

#### 1) CASE 1: $n \in (0, 1)$

In this case, since the scan period is shorter than the duration of the slotframe, and, consequently, shorter than the time between two consecutive EB points, the EB point of each step is covered by a different scan period, and, thus, by a different channel selection. An example of this case is presented in Fig. 4.

Taking into account the independence of the steps, it is obvious that the conditional probability of synchronization in the $k$-th step, given that the EB channel of the first step has index $y$ is given by:

$$P_{sync}^{cond}(k, y) = \Big( \prod_{i=1}^{k-1} \big(1 - P_{step}(i, y)\big) \Big) P_{step}(k, y). \quad (7)$$

Finding the absolute probability of synchronization at step $k$ requires to take into account all the possible cases about $y$. Since these cases are mutually exclusive and the probability of each of them (i.e., the probability that the EB channel of the first step has a specific index) is $1/C$, the (absolute) probability of synchronization at step $k$ is given by:

$$P_{sync}(k) = \sum_{y=0}^{C-1} \frac{1}{C} P_{sync}^{cond}(k, y). \quad (8)$$

To calculate the average synchronization time we must take into account that the scanning process can start at any time, and thus, the EB point can be in any position of a step. Therefore, the average expected transmission start time of the EB at the last step is $T_{sf}/2$ after the beginning of the step and $(k-1)T_{sf} + T_{sf}/2$ from the beginning of the scan. As a result, the average synchronization time is given by the following formula:

$$T_{sync}^{avg} = \sum_{k=1}^{\infty} P_{sync}(k) \left[ (k-1)T_{sf} + \frac{T_{sf}}{2} + T_{eb} \right], \quad (9)$$

where $T_{eb}$ is the time required for the transmission of an EB.

#### 2) CASE 2: $n \in \mathbb{N}^*$

In this case, the scan period exactly covers $n$ steps, and, thus, $n$ EB points, which means that the same channel selection is used for $n$ consecutive EB points. An example of this case is presented in Fig. 5. It is worth noting that, when $n = C$ – that is, when the scan period is equal to the channel hopping period – then an EB point will appear exactly once in the selected channel. However, it is not certain that an EB will be found because this also depends on the $\beta$ values.

The first step of the scan period, that is $k_f$, that contains the $k$-th step of the synchronization process is given by:

$$k_f = \left\lfloor \frac{k-1}{n} \right\rfloor n + 1. \quad (10)$$

Consequently, from the beginning of the scan period till the beginning of the $k$-th step, the number of the channel hopping

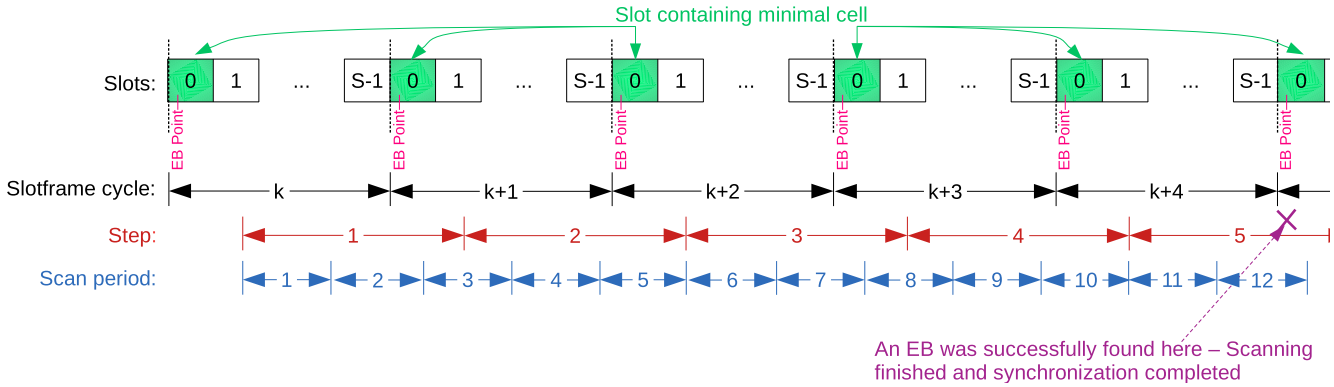


**FIGURE 4.** An example of the initial synchronization process in the case where $n = 0.4$, that is, $n \in (0, 1)$. The synchronization process randomly starts at the $k$-th cycle of a slotframe containing $S$ slots and successfully finishes at the fifth scanning step.

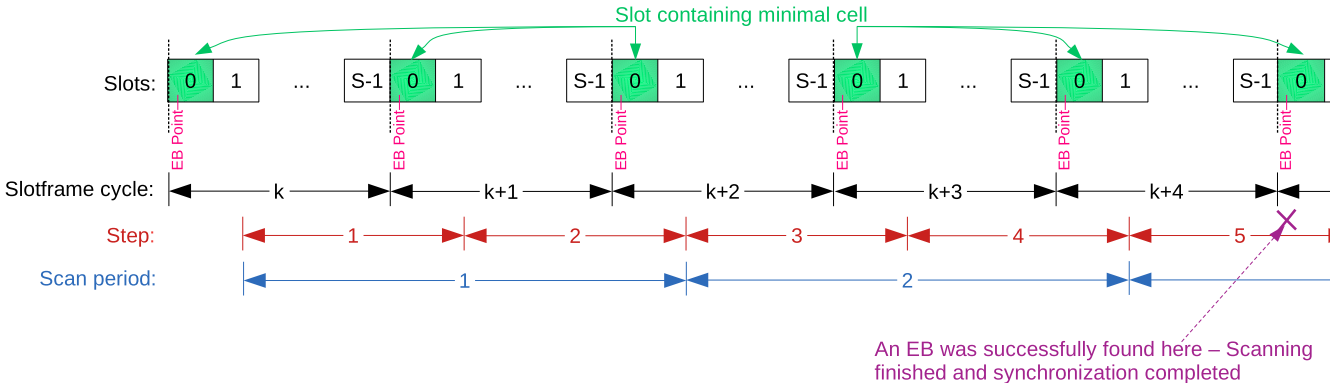


**FIGURE 5.** An example of the initial synchronization process in the case where $n = 2$, that is, $n \in \mathbb{N}^*$. The synchronization process randomly starts at the $k$-th cycle of a slotframe containing $S$ slots and successfully finishes at the fifth scanning step.

periods (i.e., $N_{chp}$) that appear is given by the following formula:

$$N_{chp} = \left\lfloor \frac{k - k_f}{C} \right\rfloor. \tag{11}$$

Therefore, before the $k$-th step appears, its EB channel has already appeared $N_{chp}$ times (i.e., in $N_{chp}$ steps) in the scan period. As a consequence, in the context of the scan period, the conditional probability $P^{sp}_{step}(k, y)$, that an EB is not found in the previous appearances of the EB channel (or, more generally, within the previous steps of the scan period), but it is found in the $k$-th step, given the fact that the EB channel of the first step of the synchronization process has index $y$, is given by:

$$P^{sp}_{step}(k, y) = \Big(1 - P_{eb}P_{sr}\big(X(k, y)\big)\Big)^{N_{chp}} P_{step}(k, y). \tag{12}$$

Since the event of achieving synchronization in a specific step of a scan period and the event of achieving synchronization in another step of the scan period are mutually exclusive, it follows that in the context of the $i$-th scan period, which starts at step $(i-1)n+1$ and ends at step $i \cdot n$, the probability of failure to synchronize, given that the EB channel of the first step of the synchronization process has index $y$, is:

$$Q_{sp}(i, y) = 1 - \sum_{k=(i-1)n+1}^{i \cdot n} P^{sp}_{step}(k, y). \tag{13}$$

Consequently, the conditional probability of synchronization in the $k$-th step, given that the EB channel of the first step has index y, is given by:

$$P^{cond}_{sync}(k, y) = \Big(\prod_{i=1}^{\lfloor \frac{k-1}{n} \rfloor} Q_{sp}(i, y)\Big) P^{sp}_{step}(k, y). \tag{14}$$

Finally, we can calculate the absolute probability of synchronization in the $k$-th step using Eq. (8), and, then, the average synchronization time using Eq. (9).

### 3) CASE 3: $n \in \mathbb{R}_{>1} - \mathbb{N}$

In this case, the scan period does not cover an exact (integer) number of steps, but some steps are covered by two consecutive scan periods; one that finishes and one that starts. For convenience, we call these steps *switch steps*. In addition, we divide a switch step into two parts, the left part, which is covered by the scan period that finishes, and the right part, which is covered by the scan period that starts. An example

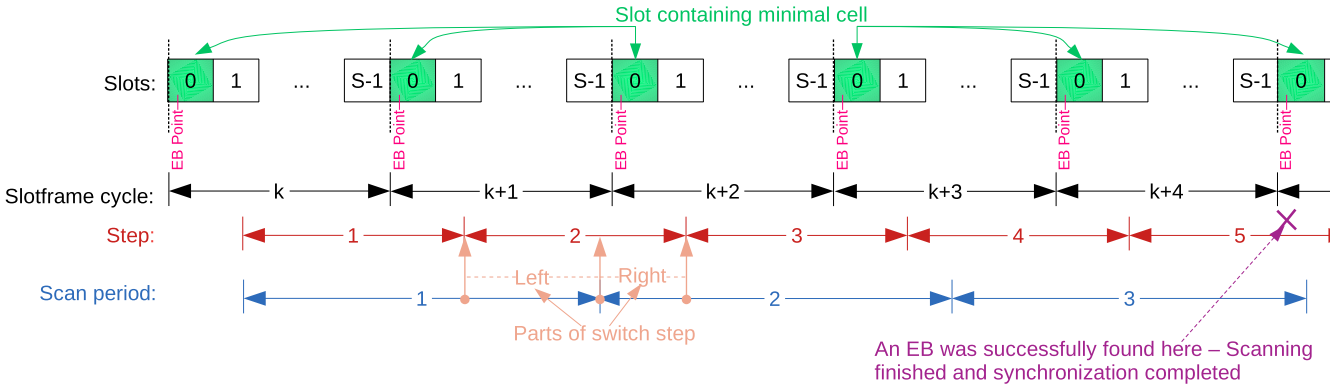


**FIGURE 6.** An example of the initial synchronization process in the case where $n = 1.6$, that is, $n \in \mathbb{R}_{>1} - \mathbb{N}$. The synchronization process randomly starts at the $k$-th cycle of a slotframe containing $S$ slots and successfully finishes at the fifth scanning step.

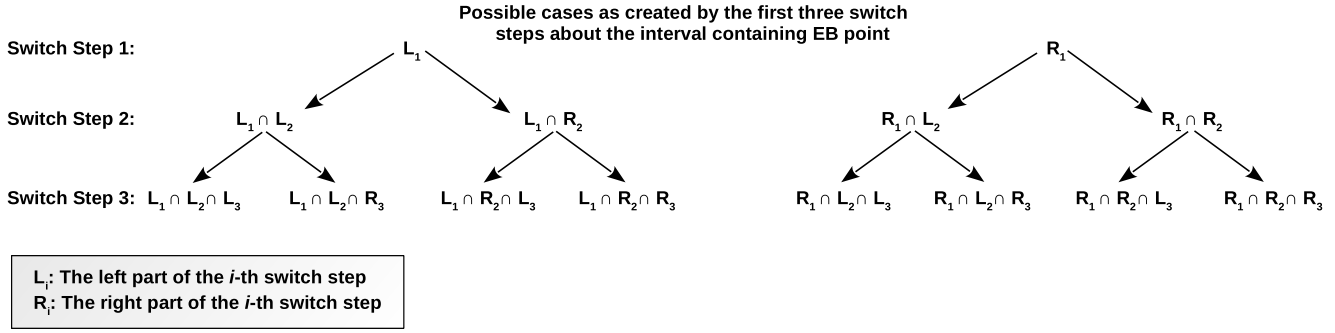


**FIGURE 7.** The possible cases that can be generated by the first three switch steps in relation to the position of the EB point.

of this case is presented in Fig. 6. Obviously, the position of the EB point in a switch step determines in which scan period it may be detected. Therefore, the position of the EB point affects the probability of synchronization in this step. Moreover, it is obvious that the appearance of the EB point in the left part increases the probability of synchronization in the scan period that ends in this step. At the same time, it negatively affects the probability of synchronization in the next scan period, a fact that shows that the probability of synchronization depends on the position of the EB point. Thus, in order to calculate the average synchronization time, we must take into account all the possible cases that are created due to the position of the EB point in each switch step. Specifically, in the first switch step of the scan process there are two distinct and independent cases: (a) the EB point belongs to the left part, and (b) the EB point belongs to the right part. In the second switch step, we must also take into account the two cases of the left and the right part. Generally, in a switch step two cases exist; one for the left part and one for the right part, for each of the possible cases of the previous switch step. In Fig. 7 we indicatively show the cases generated by the first three switch steps. Each of the cases of a switch step is characterized by the interval $I$ to which the EB point belongs. For example, one of the cases arising from the second switch step is for $I = L_1 \cap R_2$, where $L_1$ is the left part of the first switch step and $R_2$ is the right part of the second switch step. It is implied that a possible case is generated only if $I \neq \emptyset$. Furthermore, when $I \neq \emptyset$, it is obvious that $I$ is a subset of either the left or the right part of the switch step, which generates the case with this $I$. Finally, we note that the first switch step of the scan process is always the step where the first scan period finishes.

Let us now assume the case where we are in the $i$-th scan period in one of the possible cases of the last switch step of the first $i-1$ scan periods, when the first step of the synchronization process has index $y$. Additionally, let us assume that, in this case, the probability that the synchronization is not achieved in the first $i-1$ scan periods is $q$. We also define the symbols $I_s$ and $I_{len}$ to refer to the start point of $I$ and to its length respectively. To avoid possible confusion we note here that for $i = 1$, that is, for the first scan period we obviously have $q = 1$ and $I = [0, T_{sf}]$. It is clear that the first step of the scan period, that is, the step where the scan period starts, is a switch step only if the product $(i-1)n$ is not an integer number. Therefore, if we consider that $B$ is a Boolean variable that takes the value 1 when the first step of the $i$-th scan period is a switch step and 0 otherwise, it follows that:

$$B(i) = \begin{cases} 1 & \text{if } \lfloor (i-1)n \rfloor \neq (i-1)n, \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

Consequently, counting from the beginning of the synchronization process, the first step of the scan period is given by the following formula:

$$k_f = \begin{cases} \lceil (i-1)n \rceil & \text{if } B(i) = 1, \\ (i-1)n + 1 & \text{otherwise.} \end{cases} \tag{16}$$

The last step of the scan period, i.e., the step where the scan period finishes, is given by the following formula:

$$k_l = \lceil i \cdot n \rceil . \tag{17}$$

For a step $k$ of the scan period we have $k_f \leq k \leq k_l$. The contribution of the step in the average synchronization time depends on its position within the scan period.

For $k = k_f$, that is, for the first step of the scan period, we must examine whether it is a switch step. When the first step is a switch step, the synchronization can be achieved in the part that covers the current scan period, that is, in the right part of the step, only if the EB point of the step is in this part. Assuming that $R_f$ is the right part of the step, it is obvious that $R_f = [((i-1)n - \lfloor (i-1)n \rfloor)T_{sf}, T_{sf}]$, and the EB point is in the right part only if $I \subseteq R_f$. When the first step is not a switch step, the scan period covers the step entirely, and, thus, it covers the EB point of the step. In conclusion, in the context of the scan period, the probability of finding an EB in the first step of the scan period is given by the following formula:

$$P_{step}^{first} = \begin{cases} P_{step}(k_f, y) & \text{if } B(i) = 0 \text{ or } I \subseteq R_f, \\ 0 & \text{otherwise.} \end{cases} \tag{18}$$

Therefore, the probability that the synchronization is achieved in the first step of the scan period is:

$$P_{sync}^{first} = qP_{step}^{first}. \tag{19}$$

Moreover, since the EB point can be anywhere within $I$, the related contribution of this step in the average synchronization time is:

$$E_{first} = P_{sync}^{first}\left[(k_f - 1)T_{sf} + I_s + \frac{I_{len}}{2} + T_{eb}\right]. \tag{20}$$

At this point, it is useful to mention that, for each step of the scan period, except from the last step, that is, for $k_f \leq k < k_l$, we can calculate the number of EB points that have appeared from the start of the scan period until the end of the step, by using the following equation:

$$M(k) = \begin{cases} k - k_f + 1 & \text{if } B(i) = 0 \text{ or } I \subseteq R_f, \\ k - k_f & \text{otherwise.} \end{cases} \tag{21}$$

For $k_f < k < k_l$, that is, for each of the intermediate steps between the first and the last, if such steps exist, the following apply. In the context of the scan period, the probability that an EB is not found in the previous steps of the scan period and it is found in the intermediate step is given by the following formula:

$$P_{step}^{inter} = \left(1 - P_{eb}P_{sr}\left(X(k, y)\right)\right)^{\lfloor \frac{M(k)-1}{C} \rfloor} P_{step}(k, y), \tag{22}$$

and, consequently, the probability of synchronization in this step is:

$$P_{sync}^{inter} = qP_{step}^{inter}. \tag{23}$$

The related contribution of an intermediate step in the average synchronization time is given by the following formula:

$$E_{inter} = P_{sync}^{inter}\left[(k-1)T_{sf} + I_s + \frac{I_{len}}{2} + T_{eb}\right]. \tag{24}$$

For $k = k_l$, that is for the last step, we must examine whether it is a switch step. It is obvious that the last step of the $i$-th scan period is a switch step if and only if the first step of the next scan period is a switch step; that is, if $B(i+1) = 1$. In this case, the left part of the switch step is $L_l = [0, (i \cdot n - \lfloor i \cdot n \rfloor)\, T_{sf}]$, while its right part is $R_l = [(i \cdot n - \lfloor i \cdot n \rfloor)T_{sf}, T_{sf}]$. If we call $P_{lsc}$ the probability that the EB point of the last step is covered by the scan period, then:

$$P_{lsc} = \begin{cases} \dfrac{|I \cap L_l|}{|I|} & \text{if } B(i+1) = 1, \\ 1 & \text{otherwise.} \end{cases} \tag{25}$$

In the context of the scan period, the probability that an EB is not found in the previous steps of the scan period but it is found in the last step, given that the EB point of the last step is covered by the scan period is given by:

$$P_{step}^{last} = \left(1 - P_{eb}P_{sr}\left(X(k_l, y)\right)\right)^{\lfloor \frac{M(k_l-1)}{C} \rfloor} P_{step}(k_l, y), \tag{26}$$

and, thus, the probability of synchronization in the last step of the scan period is given by:

$$P_{sync}^{last} = qP_{lsc}P_{step}^{last}. \tag{27}$$

The related contribution of the last step in the average synchronization time is:

$$E_{last} = \begin{cases} 0 & \text{if } Z = \emptyset, \\ P_{sync}^{last}\left[(k_l - 1)T_{sf} + Z_s + \dfrac{Z_{len}}{2} + T_{eb}\right] & \text{otherwise,} \end{cases} \tag{28}$$

where $Z_s$ and $Z_{len}$ are respectively the start point and the length of the interval $Z$ that contains the EB point, which is equal to $|I \cap L_l|$ when $B(i+1) = 1$ and $I$ otherwise.

The probability that the EB point of the last step is covered by the current scan period and the synchronization is not achieved in the first $i$ scan periods is:

$$Q_C = qP_{lsc}\left(1 - \left(P_{step}^{first} + P_{step}^{last} + \sum_{k=k_f+1}^{k_l-1} P_{step}^{inter}\right)\right). \tag{29}$$

In order to avoid confusion, we note that, when there are no intermediate steps, the sum $\sum_{k=k_f+1}^{k_l-1} P_{step}^{inter}$ is an empty sum, and, therefore, it is not taken into account.

The probability that the EB point of the last step is covered by the next scan period and the synchronization is not achieved in the first $i$ scan periods is:

$$Q_{NC} = q\frac{|I \cap R_l|}{|I|}\left(1 - \left(P_{step}^{first} + \sum_{k=k_f+1}^{k_l-1} P_{step}^{inter}\right)\right). \quad (30)$$

In the context of the examined case, which is described by $y$, $q$ and $I$, the contribution of the $i$-th scan period in the average synchronization time is equal to the sum of the contributions of the first step, of the intermediate steps and of the last step. Since the possible cases about the position of the EB point in the last switch step of the first $i-1$ scan periods take place in different non-overlapping $I$ intervals, they are different to each other, and, consequently, for a given $y$, the contribution of the $i$-th scan period in the average synchronization time is equal to the sum of the contributions of the scan period in each of these cases. Finally, the overall contribution of the $i$-th scan period is equal to the sum of the contributions of the scan period in each of the possible values of $y$, since the cases that arise from these values are mutually exclusive.

For the needs of the simulation and in order to calculate the contribution of the $i$-th and the later scan periods in the average synchronization time in the examined case, we develop the function *RecursiveCalc* that is presented in Algorithm 1. By using Eqs. (15) – (30), this function initially calculates the contribution of the $i$-th scan period and, then, it recursively calculates the contribution of the next scan periods in each of the cases arising from the position of the EB point. It should be noted that the recursive call of the function stops when the parameter $q$ tends to 0, which means that the related contribution is essentially 0. It is also worth noting that in the implementation of the algorithm we consider that $q$ tends to zero when $q$ is less than $10^{-9}$. Starting with $i = 1$, $q = 1$, $I = [0, T_{sf}]$, and, a specific value of $y$, this function examines all the possible cases arising from the position of the minimal cell in each switch step, and, finally, calculates the contribution of the case of the given value of $y$ in the average synchronization time. Therefore, by summing the results coming from the different values of $y$, we get the average synchronization time.

Table 3 presents a synopsis of the formulas provided by the model for the average synchronization time in each of the three cases of $n$.

**Algorithm 1** Recursive Model-Based Calculation of the Average Synchronization Time When $n \in \mathbb{R}_{>1} - \mathbb{N}$

```
T_avgSync ← Σ_{y=0}^{C-1} (1/C) RecursiveCalc(1, 1, [0, T_sf], y)
function RecursiveCalc(q, i, I, y)
    if I = ∅ or q → 0 then return 0 end if
    res ← E_first                ▷ the variable res holds the result
    k ← k_f + 1
    while k ≤ k_l − 1 do
        res ← res + E_inter
        k ← k + 1
    end while
    res ← res + E_last
    res ← res + RecursiveCalc((Q_C, i + 1, Z, y))
    if B(i + 1) = 1 then
        res ← res + RecursiveCalc(Q_NC, i + 1, I ∩ R_l, y)
    end if
    return res
end function
```

### B. ALGORITHMIC APPROACH

An alternative way of approaching the problem of the optimal scan period assignment is through the algorithmic representation of the initial synchronization process, which simulates the process. We provide such a representation in Algorithm 2. In addition to the multifaceted approach of the problem, the goal of the presented algorithmic approach is to work as a tool to check the validity of our model calculations. Algorithm 2 takes as parameters the parameters presented in Table 2 as well as the channel hopping sequence, and produces as a result the time that a random execution of the initial synchronization process lasted. Initially, in line 1 of Algorithm 2, we set the start time of the process. Since a node may start the initial synchronization process at any time point throughout the network life, we set the start time to a random time point. To avoid confusion, we note that, without loss of generality, we measure the time from the start of the network operation. For example, if the process starts at the thirtieth second after the network start time, then the process start time is 30s. In line 2, we calculate the ASN of the slot where the process begins. Taking in mind that the process may start before, after or at the EB point, in lines 3-8 we define a variable called *asn* and set this to point to the ASN of the first minimal cell after the process start time that its EB point has not elapsed; this is the first minimal cell that an EB can be received. For instance, if the process start time is 30s and the slotframe has 101 slots of 10ms, then the process starts at the slot with ASN 3000, which is not a minimal cell, and, therefore we set the variable *asn* to point to the slot of the next minimal cell, which has an ASN of 3030. In lines 9-11, we define three variables named *lastSelectedChannel*, *lastSelectionTime*, and *nextSelectionTime*, which respectively hold the last channel selected for scanning, the time of the selection, and the time that the next selection will happen. Obviously, the first selection happens at the beginning of the first scan period, that is, at the beginning of the synchronization process. Furthermore, the next channel selection happens at the end of the current scan period. In lines 12-33, we execute a loop that starts from the first minimal cell that we can receive EBs and hops to the next minimal cell at each repetition. The loop stops when an EB is successfully received. More specifically, at each repetition of the loop, we first calculate the time that the transmissions of the current minimal cell take place (i.e., the time of the EB point appearance), as shown

**TABLE 3.** A synopsis of the model's formulas for the average synchronization depending on $n$ (i.e., the scan period expressed in slotframes).

| Case | Average Sync Time Formula | Notes |
|---|---|---|
| $n \in (0,1)$ | $T_{sync}^{avg} = \sum_{k=1}^{\infty}\sum_{y=0}^{C-1} \frac{1}{C} P_{sync}^{cond}(k,y)\left[(k-1)T_{sf} + \frac{T_{sf}}{2} + T_{eb}\right]$ | $P_{sync}^{cond}(k,y)$ is given by Eq. (7) |
| $n \in \mathbb{N}^*$ | | $P_{sync}^{cond}(k,y)$ is given by Eq. (14) |
| $n \in \mathbb{R}_{>1} - \mathbb{N}$ | $T_{avgSync} = \sum_{y=0}^{C-1} \frac{1}{C}\textsc{RecursiveCalc}(1,1,[0,T_{sf}],y)$ | see Algorithm 1 |

in line 13. Afterwards, in line 14, we calculate the channel that is mapped to the current minimal cell according to the channel hopping sequence. In line 15, we check if the last scan period, where the last channel selection happens, covers the EB point of the current minimal cell. If not, we repeat in lines 16–22 the channel selection procedure for each of the next scan periods until we reach the scan period that covers the EB point. Otherwise, as shown in line 24, the last selected channel is the channel of the current scan period. For example, assuming again the case where the scan starts at the thirtieth second and the slotframe has 101 slots of 10ms, and if we additionally assume that $T_{scan} = 10ms$, then the first EB point is covered by the fourth scan period, and, consequently, the lines 16–22 run in order to skip the first three scan periods. Finally, in lines 26-31, we check if an EB has successfully received, which means that the synchronization completed, and if this is true we give as a result the synchronization time. By repeating the algorithm multiple times for the same parameters, such as the scan period, we can estimate the average synchronization time in relation to these parameters. Consequently, by sampling for different scan periods, keeping the rest of the parameters the same, we can evaluate the impact of different scan periods, and, thus, we can examine the existence of an optimal scan period.

### C. THEORETICAL RESULTS

To demonstrate that the model and the algorithm provide similar results, we compare their results for 100K random instances for each of the three scan period cases of the model as described in subsection IV-A. For the needs of the comparison, we implemented the algorithm in C++,[1] in order to calculate the average synchronization time for each random instance. Using the algorithm, we collected the synchronization time from 1M synchronization attempts. Then, we recorded the differences between the model and the algorithm in the average synchronization time.

In Fig. 8, we present the results of the mathematical model as well as of the algorithm for different average values of $\beta$ ($\overline{\beta}$), scan periods, and number of available channels. We note that, since $\beta$ values are real numbers, there are infinite cases of $\beta$ values providing the same $\overline{\beta}$ which, however, can be distinguished by their standard deviation. For example, assuming 4 channels called $c_1$, $c_2$, $c_3$ and $c_4$, two cases providing $\overline{\beta} = 0.8$ are (a) $\beta(c_1) = 0.6$, $\beta(c_2) = 0.9$, $\beta(c_3) = 0.8$, $\beta(c_4) = 0.9$, and (b) $\beta(c_1) = 0.8$, $\beta(c_2) = 0.75$, $\beta(c_3) = 0.75$, $\beta(c_4) = 0.9$. Herein, for presentation reasons, we present the results for two different standard deviations (SD) of $\beta$ values: (a) the zero standard deviation representing the case with the minimum variation of $\beta$ values,

[1]The code is publicly available at https://github.com/akaralis/M6SS

**Algorithm 2** Algorithmic Representation of the Initial Synchronization Process in the Minimal 6TiSCH Configuration

**Input:** $C, S, P_{eb}, P_{sr}, T_{scan}, T_{eb}, T_{slot}, T_{txOffset}, CHS$
**Output:** $T_{sync}$

1: $startTime \leftarrow$ a random time point
2: $startASN \leftarrow \lfloor \frac{startTime}{T_{slot}} \rfloor$
3: **if** $startASN \mod S = 0$ **and**
4: $startTime \leq startASN\ T_{slot} + T_{txOffset}$ **then**
5: $asn \leftarrow startASN$
6: **else**
7: $asn \leftarrow startASN + S - (startASN \mod S)$
8: **end if**
9: $lastSelectedChannel \leftarrow$ a random channel
10: $lastSelectionTime \leftarrow startTime$
11: $nextSelectionTime \leftarrow lastSelectionTime + T_{scan}$
12: **while** true **do**
13: $txTime \leftarrow asnT_{slot} + T_{txOffset}$
14: $minimalCellChanel \leftarrow CHS[asn \mod C]$
15: **if** $txTime \geq nextSelectionTime$ **then**
16: **repeat**
17: $scannedChannel \leftarrow$ a random channel
18: $lastSelectedChannel \leftarrow scannedChannel$
19: $lastSelectionTime \leftarrow nextSelectionTime$
20: $nextSelectionTime \leftarrow$
21: $nextSelectionTime + T_{scan}$
22: **until** $nextSelectionTime > txTime$
23: **else**
24: $scannedChannel \leftarrow lastSelectedChannel$
25: **end if**
26: $p \leftarrow$ a random real number between 0 and 1
27: **if** $minimalCellChannel = scannedChannel$
28: **and** $p < P_{eb}\ P_{sr}[minimalCellChannel]$ **then**
29: $T_{sync} \leftarrow txTime - scanStartTime + T_{eb}$
30: **break** ▷ Synchronization Completed
31: **end if**
32: $asn \leftarrow asn + S$ ▷ Go to the next minimal cell
33: **end while**

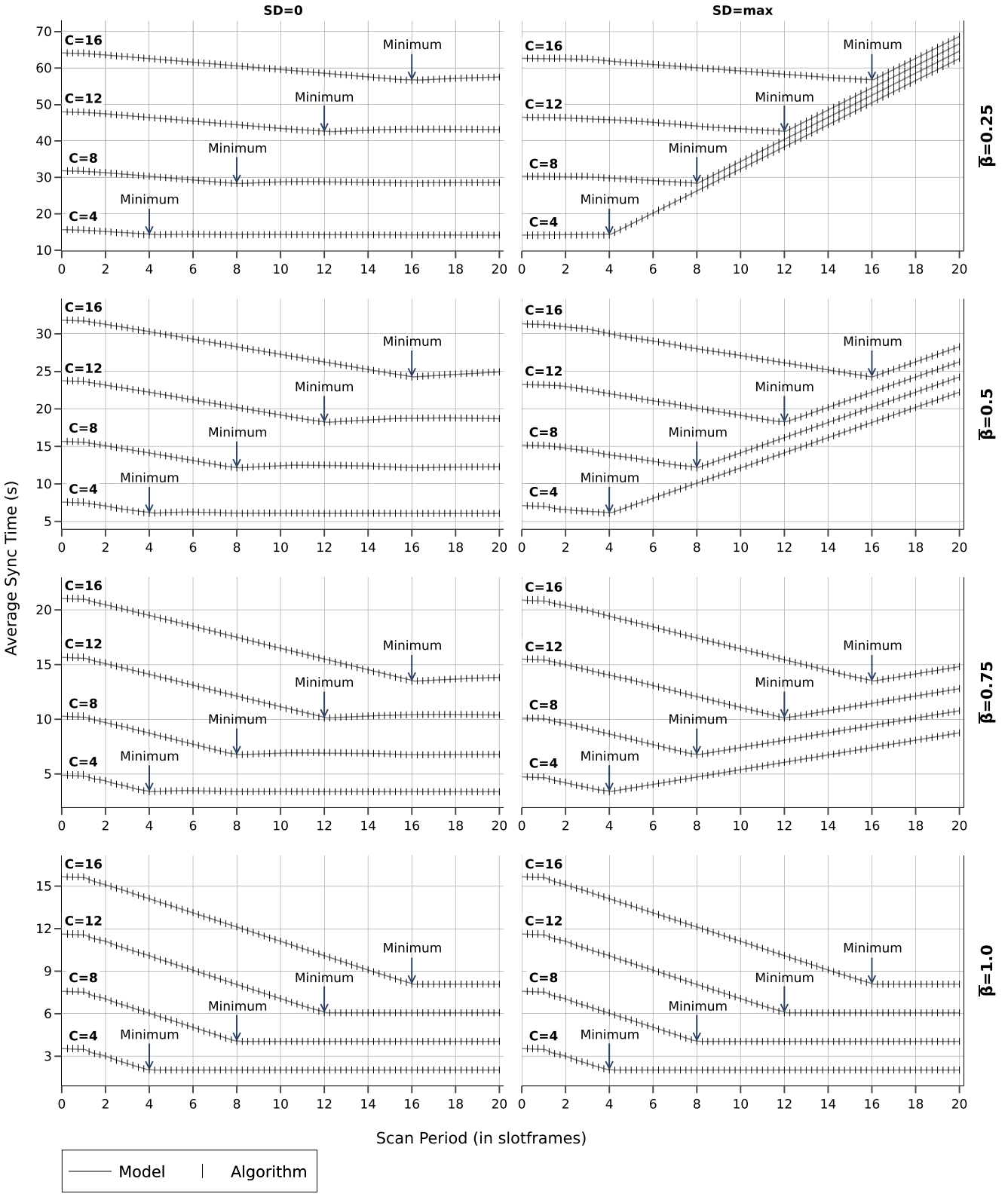


**FIGURE 8.** Average synchronization time calculated by the model and the algorithm for 4, 8, 12 and 16 channels ($C$) when there are 101 slots (of 10ms) in the slotframe, $\beta$ is (a) 0.25, (b) 0.5, (c) 0.75 and (d) 1, and the standard deviation (SD) of $\beta$ values is (a) the minimum (i.e., 0) and (b) the maximum.

and (b) the maximum standard deviation representing the case with the maximum variation of $\beta$ values, that is, the case with the maximum possible number of extreme $\beta$ values (i.e., 0 and 1). It is noted that when $\overline{\beta} = 1$, all the $\beta$ values are obviously always 1. Therefore, when $\overline{\beta} = 1$ the standard deviation is always 0, and, thus, the cases of the zero and of the maximum standard deviation are identical.

Since we follow the recommended settings of the minimal 6TiSCH configuration, we use the default timeslot template (i.e., slots of 10ms) and the default channel hopping sequence. Regarding the slotframe length, we use the commonly used length of 101 slots. The confidence intervals for the algorithmic results are not depicted because they are tiny and not visible to the reader. The results show that the average synchronization time between the model and the algorithm differs by only 0.08% in average, while the maximum difference we recorded is 0.59%. Therefore, the results of the comparison show that the difference between the model and the algorithm is negligible.

Moreover, through the detailed mathematical analysis that we conducted using the model and the algorithm, we found that the best average synchronization time is achieved by setting the scan period equal to $C$ slotframes, where $C$ is the number of available channels. Comparing, for $\overline{\beta} < 1$, the case where the standard deviation of $\beta$ values is minimum (i.e., zero) with the case where the standard deviation is the maximum possible we observe the following. For scan periods less or equal to the optimal (i.e., $C$ slotframes), the average synchronization times are quite similar diverging at most 10%. However, for scan periods larger than the optimal, the average synchronization time is changing in a different way in each case. Specifically, in the case with the zero standard deviation, we observe that the average synchronization time tends to the minimum as the scan period increases. This happens because all the channels have the same $\beta$ values, and, thus, the same quality. On the contrary, when the standard deviation is maximum, the average synchronization time increases as the scan period increases; that is, the average synchronization time tends to infinity as the scan period tends to infinity. Obviously, this happens because of the presence of channels with zero $\beta$ values (i.e., bad/blocked channels) the selection of which leads to a waste of time that increases as the scan period increases. Apparently, in this case, the worst scan period cannot be delimited, and, therefore, we can not set an upper bound to the maximum improvement achieved by using the optimal scan period.

When $SD = 0$, the worst average synchronization time appears when the scan period is less or equal to 1 slotframe, while the benefit of using the optimal setting is as follows. In the case where $\overline{\beta} = 0.25$, the improvement may be up to 9.67% when using 4 channels, 11.11% with 8 channels, 11.58% with 12 channels, and 11.81% with 16 channels. When $\overline{\beta} = 0.5$, the maximum improvement increases to 20.0% for 4 channels, 22.57% for 8 channels, 23.4% for 12 channels, and 23.81% for 16 channels. When $\overline{\beta} = 0.75$, the improvement is up to 31.01%, 34.41%, 35.47% and 36.0% for 4, 8, 12 and 16 channels respectively. Finally, in the case where $\overline{\beta} = 1$ the improvement reaches 42.81% when there are 4 channels, 46.64% with 8 channels, 47.81% with 12 channels, and 48.37% with 16 channels. As we can observe, the benefit of using the optimal scan period significantly increases as $\overline{\beta}$ increases. Moreover, we observe that as the number of channels increases the benefit increases, but the improvement is not considerable.

Using the theoretical results of the model, we compare in Fig. 9 the performance of the optimal scan period with the default scan periods of two famous operating systems for 6TiSCH networks: (a) Contiki-NG,[2] and (b) OpenWSN.[3] The default scan period of Contiki-NG is 1$s$, while the default scan period of OpenWSN is 1600$ms$. In this comparison, we vary the slotframe size, the number of available channels, and the value of $\overline{\beta}$. For presentation results, we present the theoretical results of the case with the zero standard deviation of $\beta$ values; the results with the maximum standard deviation are similar. As we have already mentioned above, it is clear that the benefit of using the optimal scan period is getting higher with a higher value of $\overline{\beta}$ and with a higher number of channels. Moreover, the results reveal considerable performance gains when the slotframe size is longer. This is explained by the fact that the default scan periods of Contiki-NG and OpenWSN are fixed time values; they are translated to a different submultiple or multiple of the slotframe duration for different slotframe durations, which in the language of the model implies a different $n$ for each different slotframe duration. On the one hand, as the length of the slotframe decreases, the default scan periods of Contiki-NG and OpenWSN expressed in slotframes are getting larger. On the other hand, with a longer slotframe, these default scan periods have a lower value expressed in slotframes. Therefore, depending on the slotframe length, these default scan periods may get closer or get further away from the optimal scan period. Our results show that using the optimal scan period instead of the default scan periods of Contiki-NG and OpenWSN we can reduce the average synchronization time up to 48.37% and 47.10% respectively.

## V. TESTBED SETUP, EXPERIMENTS, & RESULTS

We also conducted a set of experiments on a real testbed.[4] The purposes of doing the experiments are the following:

- To validate the theoretical results,
- To present for formal reasons an experimental comparison between the optimal scan period and the default scan periods of the operating systems Contiki-NG and OpenWSN,
- To examine the relation between the synchronization time and the energy consumption, and

[2]https://www.contiki-ng.org

[3]https://openwsn.atlassian.net/wiki

[4]The code of experiments as well as the experiments results are publicly available at https://github.com/akaralis/contiki-ng-minimal_6tisch_sync

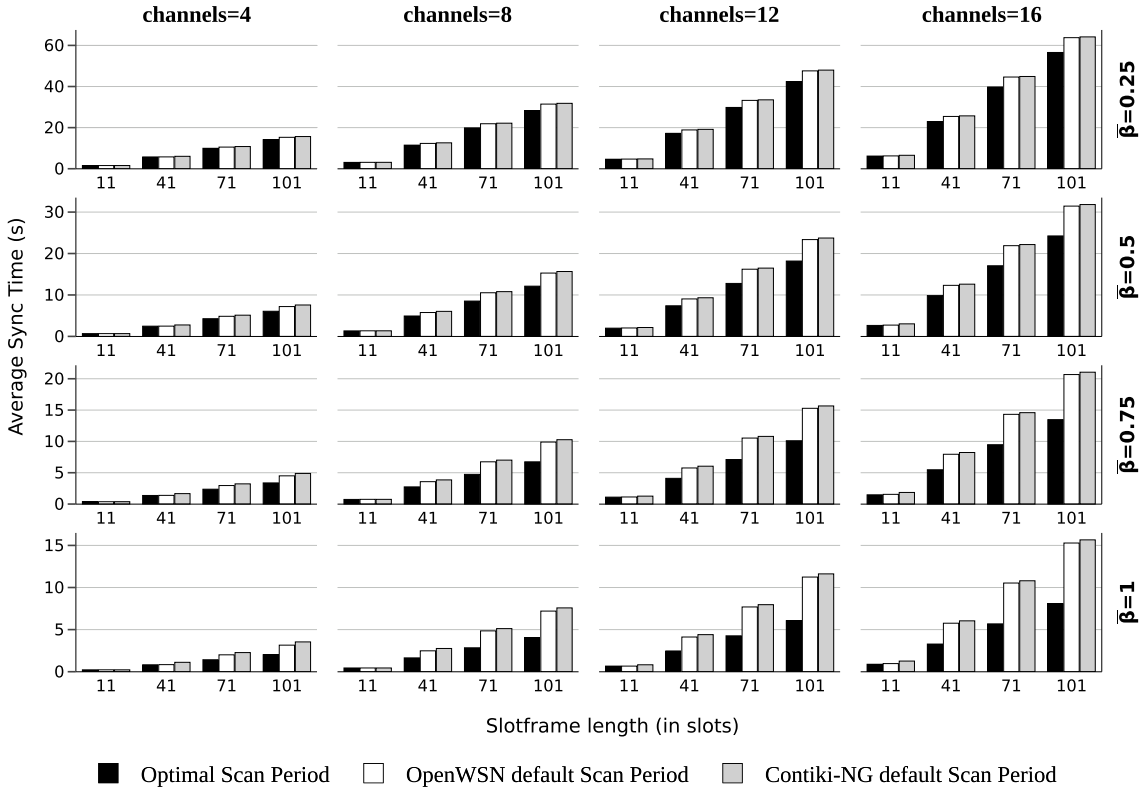


**FIGURE 9.** Comparison between the optimal scan period and the default values of Contiki-NG and OpenWSN for different radio channels and slotframes when the standard deviation of the $\beta$ values is zero.

- To demonstrate the improvement achieved by using the optimal scan period in terms of the energy consumption.

It is worth noting that in the context of a real environment we cannot take very large samples in a reasonable time, since the initial synchronization of the nodes is a very time-consuming process. Additionally, since in a real environment $P_{sr}$ physically fluctuates over time, we cannot use a fixed value for $\overline{\beta}$, thus, the average value over hundreds of experiments is presented. The method of how $P_{eb}$ and $P_{sr}$ were captured is explained later in the text.

### A. EXPERIMENTAL SETUP

To examine the validity of our model, we conduct experiments using 12 Zolertia Re-Mote B devices,[5] which run the Contiki-NG operating system. The nodes are randomly positioned in a $30m^2$ area, co-existing with WiFi networks, while the RPL routing protocol is run to build the topology. We used the recommended settings of the minimal 6TiSCH configuration as they were presented in Table 1. Apart from the recommended number of channels (i.e., 16), we also conducted experiments with 4, 8, and 12 channels. Regarding the slotframe length and the EB period, which are not predefined in the minimal 6TiSCH configuration, we selected the commonly used slotframe of 101 slots (1.01s) and an EB period of 4 slotframes (4.04 s). It should be noted that for the scheduling of EB transmissions, we used a random scheduling policy that statistically achieves an expected EB rate of 1 EB per EB period. More specifically, to achieve an expected rate of 1 EB per EB period, the nodes send an EB in each minimal cell repetition with a probability given by the ratio of the slotframe duration to the EB period, which according to the Binomial Distribution leads to the desired expected EB rate. We must mention that by default Contiki-NG uses a random-based scheduling, but it introduces a limited randomness in the EB scheduling. In the context of the minimal 6TiSCH configuration, the randomness of this solution becomes smaller as the EB period reduces, leading to an almost static EB scheduling. For this reason, we select to implement an alternative fully random EB scheduling that provides the desired average EB rate.

In our experiments, we divided the nodes into two categories (a) the advertisers, which are the already joined nodes in the network and which can send EBs, and (b) the samplers, whose role is to collect samples for the synchronization time by making repeated synchronization attempts. Each synchronization attempt starts at a random time and finishes when an EB is found. We used 4 advertisers and 8 samplers. During the experiments, the advertisers were recording their EB transmissions. By having the EB transmissions recorded, and since the nodes of our testbed are physical neighbors, we can

[5]https://github.com/Zolertia/Resources/wiki/RE-Mote

easily calculate $P_{eb}$ for the entire network as well as for any desired time interval of the experiment. The samplers were recording the channels they scanned during each of the synchronization attempts. It is obvious that, by using the recorded information, we can find the number of cases where an EB is transmitted in a channel scanned and – as a consequence – the number of cases where an EB is successfully received. In this way, we can calculate the $P_{sr}$ values of a specific sample.

To compare with the model and to cover all the three cases presented in Section IV-A, we collected samples for a scan period from 0.5 to 20 slotframes with a size step of 1.5 slotframes. Additionally, we took samples for the scan period of $C$ slotframes ($C$ is the number of available channels in each experiment), for the scan period of 1 second (or approximately 0.99 slotframes), and for the scan period of 1600ms (or approximately 1.59 slotframes). We remind that the scan period of $C$ slotframes is the optimal scan period according to the model, while the scan period of 1 second and of 1600ms are the default scan periods of Contiki-NG and OpenWSN respectively. To avoid confusion, we also remind that the results of all the above cases including the default scan period of OpenWSN were taken using the Contiki-NG operating system according to the configuration described at the beginning of the section. In total, we collected 6000 samples for each of the scan periods. All the experiments were carried out within a time period of approximately 2 weeks. The experiment parameters are summarized in Table 4.

To accurately capture the energy consumption in our experiments we used the Energest module[6] of Contiki-NG, which can be used to implement a lightweight, software-based energy estimation approach for resource-constrained IoT devices. Energest allows to record the time that a hardware component is on, off, or in different modes. Therefore, if we additionally know the power consumption of the component in each of its possible modes, which is typically given by the manufacturer, we can estimate its energy consumption.

Using Energest we can easily keep information about the operation time of the base component of a resource-constrained IoT device: (a) the microcontroller (i.e., the central processing unit – CPU), and (b) the radio. Specifically, we can record the time that the microcontroller is on, the time that is in low power mode, and the time that is in deep low power mode. Regarding the radio, we can record the time that the radio is in transmit mode as well as the time that the radio is in receive mode. Assuming the use of only the base components, as in the case of our experiment, we can calculate the total energy consumption by summing the energy consumption of the microcontroller and of the radio.

Based on the manufacturer specifications[7] of the devices, we know that their microcontrollers consume $20mA$ when they are active and $1.3\mu A$ in the low power mode, while, as we also confirmed through our experiments, they never shift to a deep low power mode. Additionally, the radio consumes $20mA$ (peak) at the receive mode. Consequently, the formula that we used to calculate the energy consumed by a sampler during the initial synchronization process is given by:

$$E_{total} = (0.02 \cdot T_{cpu}^{active} + 1.3 \cdot 10^{-6} \cdot T_{cpu}^{lpm} + 0.02 \cdot T_{sync}) \cdot V,$$

where $T_{cpu}^{active}$ is the time in seconds that the microcontroller was active, $T_{cpu}^{lpm}$ is the time in seconds that the microcontroller was in low power mode, $T_{sync}$ is the synchronization time in seconds which is obviously the time the node had its radio in listening (receive) mode, and $V$ is the voltage in volts of the battery feeding the devices, which is equal to 3.7V.

**TABLE 4.** Testbed experiments parameters.

| Parameter | Value |
|---|---|
| Number of nodes | 12 |
| Number of advertisers | 4 |
| Number of gateways (PAN Coordinator) | 1 |
| Deployment area size | 30 $m^2$ |
| Slotframe Length | 101 |
| Number of Channels | [4, 8, 12, 16] |
| Transmit power | 0 dBm |
| Operating System | Contiki-NG |
| EB PERIOD | 4 slotframes (4.04 s) |
| Routing | RPL (OF0) |

### B. EXPERIMENTAL RESULTS

In Table 5, we compare the results of the experiments with those of the model when the recommended number of channels is used. We present the average synchronization time as well as the difference between the testbed and the model values. It is observed that the results of the experiments are very close to the theoretical results. We should note that the similarity between the model and the testbed results have been also verified using 4, 8 and 12 channels. Since the performance trend is identical for all channel numbers, we only present experimental results using 16 channels.

In Fig. 10 we compare the results taken from our experiments (a) by forcing the nodes to use the optimal scan period according to the model (Model Optimal), (b) the default scan period of Contiki-NG (Contiki Default), (c) the default scan period of OpenWSN (OpenWSN Default), and (d) the experiment optimal scan period (Experiment Optimal), that is, the scan period that gives the best average synchronization time among a large set of scan periods (0.5–20 slotframes). For completeness, we also present the corresponding $\overline{\beta}$ and the standard deviation of the $\beta$ values captured in each case. As we can observe, the difference between the Model Optimal and the Experiment Optimal is negligible. A very small difference that is observed in some scenarios (e.g., when $C = 8$) is justified by the slightly different $\overline{\beta}$ value. Compared to the default scan periods of Contiki-NG and OpenWSN, the model optimal scan period gives much better

[6] https://github.com/contiki-ng/contiki-ng/wiki/Documentation:-Energest

[7] https://github.com/Zolertia/Resources/raw/master/RE-Mote/Hardware/Revision%20B/Datasheets/ZOL-RM0x-B%20-%20RE-Mote%20revision%20B%20Datasheet%20v.1.0.0.pdf

**TABLE 5.** Comparison between the theoretical and the experimental results for 16 channels. The scan period ($T_{scan}$) is expressed in slotframes ($sf$).

| $T_{scan}$ ($sf$) | $\beta$ values | | Average Sync Time (s) | | Difference (%) |
|---|---|---|---|---|---|
| | $\overline{\beta}$ | SD | Theoretical | Testbed | |
| 0.5 | 0.581 | 0.020 | 27.331 | 27.382 | 0.188 |
| 0.99 | 0.599 | 0.020 | 26.478 | 26.368 | 0.417 |
| 1.59 | 0.573 | 0.027 | 27.316 | 26.903 | 1.512 |
| 2 | 0.612 | 0.012 | 25.412 | 25.745 | 1.313 |
| 3.5 | 0.617 | 0.024 | 24.394 | 24.456 | 0.255 |
| 5 | 0.598 | 0.022 | 24.504 | 24.593 | 0.364 |
| 6.5 | 0.613 | 0.013 | 23.065 | 23.247 | 0.793 |
| 8 | 0.586 | 0.023 | 23.559 | 23.432 | 0.539 |
| 9.5 | 0.586 | 0.015 | 22.781 | 23.067 | 1.254 |
| 11 | 0.594 | 0.024 | 21.651 | 21.460 | 0.881 |
| 12.5 | 0.598 | 0.013 | 20.683 | 20.425 | 1.247 |
| 14 | 0.602 | 0.018 | 19.779 | 19.485 | 1.488 |
| 15.5 | 0.598 | 0.019 | 19.173 | 19.213 | 0.212 |
| 16 | 0.586 | 0.016 | 19.477 | 19.226 | 1.287 |
| 17 | 0.589 | 0.023 | 19.534 | 19.677 | 0.730 |
| 18.5 | 0.590 | 0.024 | 19.736 | 19.103 | 3.207 |
| 20 | 0.588 | 0.012 | 19.948 | 19.704 | 1.221 |

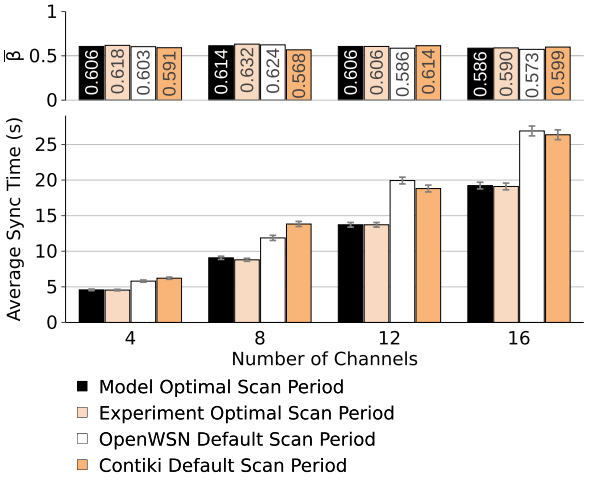


**FIGURE 10.** Comparison of the average synchronization time recorded in the experiments for the Model Optimal scan period, with the Experiment Optimal average synchronization time, and with the average synchronization times taken for the default scan periods of Contiki-NG and OpenWSN. The 95% confidence intervals are presented. The values in the top bar chart is the $\overline{\beta}$ recorded in each case.

average synchronization times, even when the latter has a lower $\overline{\beta}$.

Besides the synchronization time, we also capture the related energy consumption due to the synchronization. To accurately capture the energy consumption we used the Energest module of Contiki-NG. The results are illustrated in Fig. 11. Our results show a practically linear relation between the average synchronization time and the related average energy consumption. This is explained by the fact that in the initial synchronization process, the average energy consumption of the microcontroller was a very small percentage of the total average energy consumption, and more specifically close to 2.5%. Since there exists an almost linear relation between the average synchronization time and

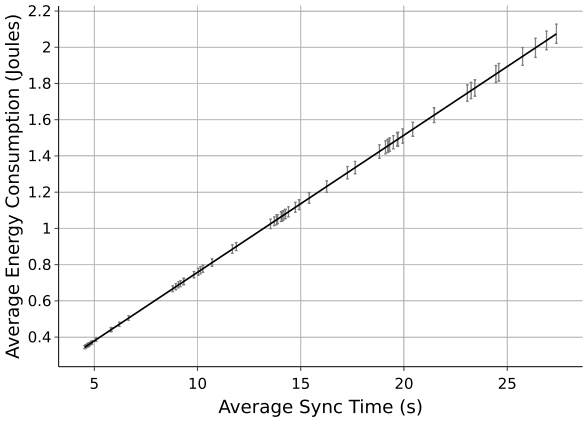


**FIGURE 11.** Correlation of the average synchronization time with the average energy consumption. The 95% confidence intervals are presented.

the related average energy consumption, we can conclude that a reduction in the average synchronization time leads to an almost proportional reduction in the average energy consumption. Consequently, the optimal and the worst scan periods from the perspective of the synchronization time are also the optimal and the worst scan periods in terms of the energy consumption respectively. It is also obvious that by knowing the average synchronization time, we can satisfactorily estimate the related average energy consumption. Furthermore, since the use of the optimal scan period can lead to an up to 48.37% lower average synchronization time, it can also lead to an almost the same maximum percentage reduction in the average energy consumption.

Although our results show that using the optimal scan period we can achieve a significant reduction in the average energy consumed during the initial synchronization, we should examine how important is this reduction compared to the total energy consumed throughout the lifecycle of a node. However, the lifecycle of a node depends on its duty cycle and on the available energy sources (e.g., battery, solar panel). Herein, we make an analysis using as a reference point for the lifecycle of a node the battery provided for our devices by their manufacturer. This battery has a capacity of 600mAh and operates at 3.7V.

Based on this assumption and assuming, for presentation reasons, a zero standard deviation of $\beta$ values, we compare in Fig. 12 the energy consumption of a single initial synchronization when (a) the optimal, and (b) the worst scan period are used for the cases where $\overline{\beta}$ is equal to 0.25, 0.5, 0.75 and 1, and for 4,8,12, and 16 channels in cases where the slotframe size is 11, 41, 71, 101. The presented energy consumption is an estimation based on the average synchronization time as it is provided by the model and the power consumption of the radio of our devices when they are in listen mode. The results show that the percentage of the average energy consumption of a single synchronization in relation to the total energy may reach up to 0.06% approximately. Additionally, using the optimal scan period we can approximately achieve an up to 0.007% reduction in terms of the total energy consumption

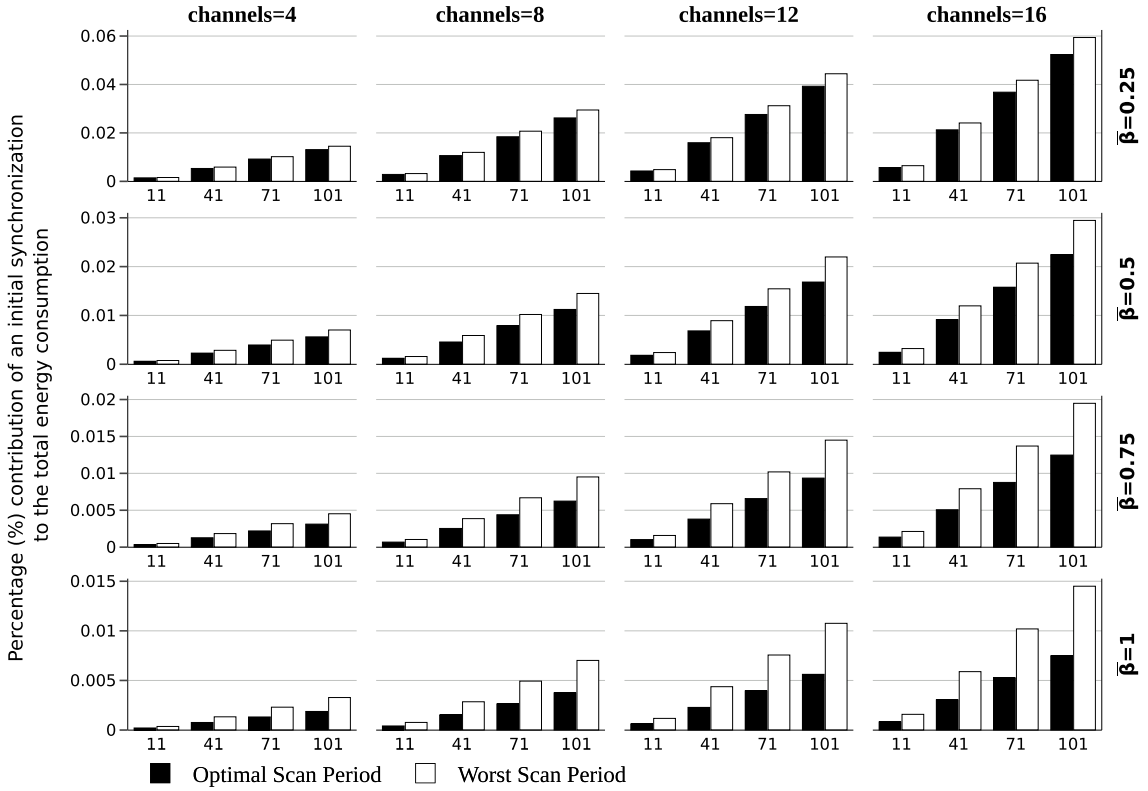


**FIGURE 12.** Comparison between the worst and the optimal scan period in terms of the percentage of the total energy consumed during an initial synchronization for different radio channels, slotframes, and $\bar{\beta}$, when the standard deviation of $\beta$ values is zero.

of a node. Therefore, it is clear that from the perspective of the total energy consumption of a node, the impact of the energy consumption of a single initial synchronization as well as the benefit of using the optimal scan period are negligible. However, we should mention that a node may make multiple (re)join attempts during its life-cycle. For example, this intensively happens in the case of a mobile node that voluntarily or involuntarily connects and disconnects from the network as it moves away or approaches the network respectively. Moreover, the nodes may be powered by capacitors whose energy is limited and regularly replenished by a renewable energy resource. In such cases, the impact of the initial synchronization as well as the benefit of using the optimal scan period in terms of energy may be significant.

## VI. CONCLUSION & FUTURE WORK

In this paper, we dealt with the problem of the initial synchronization of the nodes in the context of the minimal 6TiSCH configuration. More specifically, we examined the impact of the scan period on the synchronization time and on the related energy consumption. As far as we know, this study is the first that examines the impact of the scan period on the initial synchronization procedure of IEEE802.15.4-TSCH networks. We presented a thorough mathematical analysis as well as an algorithmic approach and experimental results and we observed that by setting the scan period to $C$ slotframes, where $C$ is the number of available channels, we get the best average synchronization time. We also achieved an up to 48.37% and 47.1% improvement in the average synchronization time and an almost proportional reduction in the related energy consumption compared to the default scan periods of Contiki-NG and OpenWSN, respectively. It is worth noting that, regarding the 6TiSCH applications where the nodes do not know the number of channels and the slotframe duration apriori, they can calculate the optimal scan period after the initial synchronization, and, therefore, we enable a node to speed up its rejoin attempts even if it does not know the number of channels and the slotframe duration at boot time. As part of our future work, we intend to investigate the optimal scan period in the context of the minimal 6TiSCH configuration when a channel blacklisting mechanism is used. Moreover, we intend to assess the optimal synchronization time in the presence of jamming attacks.

## ACKNOWLEDGMENT

This research has been financially supported by General Secretariat for Research and Technology (GSRT) and the Hellenic Foundation for Research and Innovation (HFRI) (Scholarship Code:2160). This work has also been partially supported by UPRC (University of Piraeus Research Center) and by COSMOTE through a PEDION24 grant.

## REFERENCES

[1] *IEEE Standard for Low-Rate Wireless Networks*, IEEE Standard 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011), Apr. 2016, pp. 1–709.
[2] X. Vilajosana, T. Watteyne, T. Chang, M. Vučinić, S. Duquennoy, and P. Thubert, "IETF 6TiSCH: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 1, pp. 595–615, 1st Quart., 2020.
[3] X. Vilajosana, K. Pister, and T. Watteyne, *Minimal IPv6 Over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration*, document BCP, RFC 8180, May 2017. [Online]. Available: https://rfc-editor.org/rfc/rfc8180.txt
[4] A. Dunkels, B. Gronvall, and T. Voigt, "Contiki—A lightweight and flexible operating system for tiny networked sensors," in *Proc. 29th Annu. IEEE Int. Conf. Local Comput. Netw.*, Nov. 2004, pp. 455–462.
[5] S. Duquennoy, A. Elsts, B. A. Nahas, and G. Oikonomo, "TSCH and 6TiSCH for contiki: Challenges, design and evaluation," in *Proc. 13th Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, Jun. 2017, pp. 11–18.
[6] A. Elsts, X. Fafoutis, G. Oikonomou, R. Piechocki, and I. Craddock, "TSCH networks for health IoT: Design, evaluation, and trials in the wild," *ACM Trans. Internet Things*, vol. 1, no. 2, pp. 1–27, Apr. 2020.
[7] K. Brun-Laguna, A. L. Diedrichs, D. Dujovne, C. Taffernaberry, R. Léone, X. Vilajosana, and T. Watteyne, "Using SmartMesh IP in smart agriculture and smart building applications," *Comput. Commun.*, vol. 121, pp. 83–90, May 2018.
[8] M. Vucinic, T. Chang, B. Škrbić, E. Kočan, M. Pejanovic-Djurisic, and T. Watteyne, "Key performance indicators of the reference 6TiSCH implementation in Internet-of-Things scenarios," *IEEE Access*, vol. 8, pp. 79147–79157, 2020.
[9] A. Karalis, D. Zorbas, and C. Douligeris, "Collision-free advertisement scheduling for IEEE 802.15.4-TSCH networks," *Sensors*, vol. 19, no. 8, p. 1789, Apr. 2019.
[10] M. Vučinić, T. Watteyne, and X. Vilajosana, "Broadcasting strategies in 6TiSCH networks," *Internet Technol. Lett.*, vol. 1, no. 1, p. e15, Jan. 2018.
[11] T. P. Duy, T. Dinh, and Y. Kim, "A rapid joining scheme based on fuzzy logic for highly dynamic IEEE 802.15. 4e time-slotted channel hopping networks," *Int. J. Distrib. Sensor Netw.*, vol. 12, no. 8, 2016, Art. no. 1550147716659424.
[12] E. Vogli, G. Ribezzo, L. A. Grieco, and G. Boggia, "Fast network joining algorithms in industrial IEEE 802.15.4 deployments," *Ad Hoc Netw.*, vol. 69, pp. 65–75, Feb. 2018.
[13] T. Watteyne, M. R. Palattella, and L. A. Grieco, *Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement*, document RFC 7554, May 2015. [Online]. Available: https://rfc-editor.org/rfc/rfc7554.txt
[14] R. Teles Hermeto, A. Gallais, and F. Theoleyre, "Scheduling for IEEE802.15.4-TSCH and slow channel hopping MAC in low power industrial wireless networks: A survey," *Comput. Commun.*, vol. 114, pp. 84–105, Dec. 2017.
[15] D. De Guglielmo, A. Seghetti, G. Anastasi, and M. Conti, "A performance analysis of the network formation process in IEEE 802.15.4e TSCH wireless sensor/actuator networks," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2014, pp. 1–6.
[16] J.-Y. Kim, S.-H. Chung, and Y.-V. Ha, "A fast joining scheme based on channel quality for IEEE802.15.4e TSCH in severe interference environment," in *Proc. 9th Int. Conf. Ubiquitous Future Netw. (ICUFN)*, Jul. 2017, pp. 427–432.
[17] R. T. Hermeto, Q. Bramas, A. Gallais, and F. Théoleyre, "Analysis of the network attachment delay of mobile devices in the industrial Internet of Things," in *Ad-Hoc, Mobile, and Wireless Networks*, M. R. Palattella, S. Scanzio, and S. C. Ergen, Eds. Cham, Switzerland: Springer, 2019, pp. 90–101.
[18] D. De Guglielmo, S. Brienza, and G. Anastasi, "A model-based beacon scheduling algorithm for IEEE 802.15.4e TSCH networks," in *Proc. IEEE 17th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2016, pp. 1–9.
[19] I. Khoufi and P. Minet, "An enhanced deterministic beacon advertising algorithm for building TSCH networks," *Ann. Telecommun.*, vol. 73, nos. 11–12, pp. 745–757, Dec. 2018.
[20] C. Vallati, S. Brienza, G. Anastasi, and S. K. Das, "Improving network formation in 6TiSCH networks," *IEEE Trans. Mobile Comput.*, vol. 18, no. 1, pp. 98–110, Jan. 2019.
[21] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust mesh networks through autonomously scheduled TSCH," in *Proc. 13th ACM Conf. Embedded Netw. Sensor Syst.*, Nov. 2015, pp. 337–350.
[22] J. Vera-Pérez, D. Todolí-Ferrandis, S. Santonja-Climent, J. Silvestre-Blanes, and V. Sempere-Payá, "A joining procedure and synchronization for TSCH-RPL wireless sensor networks," *Sensors*, vol. 18, no. 10, p. 3556, Oct. 2018.
[23] J. Vera-Pérez, D. Todolí-Ferrandis, J. Silvestre-Blanes, and V. Sempere-Payá, "Bell-X, an opportunistic time synchronization mechanism for scheduled wireless sensor networks," *Sensors*, vol. 19, no. 19, p. 4128, Sep. 2019.
[24] A. Karalis, "ATP: A fast joining technique for IEEE802.15. 4-TSCH networks," in *Proc. IEEE 19th Int. Symp. World Wireless, Mobile Multimedia Netw. (WoWMoM)*, Jun. 2018, pp. 588–599.
[25] A. Kalita and M. Khatua, "Channel condition based dynamic beacon interval for faster formation of 6TiSCH network," *IEEE Trans. Mobile Comput.*, early access, Mar. 16, 2020, doi: 10.1109/TMC.2020.2980828.
[26] M. Sha, D. Gunatilaka, C. Wu, and C. Lu, "Empirical study and enhancements of industrial wireless sensor–actuator network protocols," *IEEE Internet Things J.*, vol. 4, no. 3, pp. 696–704, Jun. 2017.

**APOSTOLOS KARALIS** is currently pursuing the Ph.D. degree with the Department of Informatics, University of Piraeus, Greece. His Ph.D. research focuses on the Internet of Things with emphasis on the Industrial Internet of Things. He received the Ph.D. Scholarship from the Greek General Secretariat for Research and Technology, and the Hellenic Foundation for Research and Innovation.

**DIMITRIOS ZORBAS** (Member, IEEE) received the Ph.D. degree in computer science from the University of Piraeus, Greece. He has worked as a Postdoctoral Researcher with the Inria Lille–Nord Europe and the University of La Rochelle, France. He was also a Researcher with the Tyndall National Institute, University College Cork, Ireland. He is currently an Assistant Professor with Nazarbayev University, Kazakhstan. He is the author of more than 50 peer-reviewed publications in the areas of wireless communications and distributed computing. He has worked in several national, and FP7 and H2020 projects. He received the Marie Curie Fellowship.

**CHRISTOS DOULIGERIS** (Senior Member, IEEE) is currently a Professor with the Department of Informatics, University of Piraeus, Greece. He held positions at the Department of Electrical and Computer Engineering, University of Miami. He was an Associate Member of the Hellenic Authority for Information and Communication Assurance and Privacy, and the President and the CEO of the Hellenic Electronic Governance for Social Security SA. He has published extensively in the networking scientific literature and has participated in many research and development projects. His main research interests include computer networking, communications, network security, cyber security, and new technologies in education and emergency response operations.