Computer Communications 88 (2016) 1–24

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Review

IEEE 802.15.4e: A survey

Domenico De Guglielmo[a], Simone Brienza[a,*], Giuseppe Anastasi[a,b]

[a] Dept. of Information Engineering, University of Pisa, Italy
[b] Smart Cities National Lab., CINI, Italy

ARTICLE INFO

Article history:
Received 25 January 2016
Revised 3 May 2016
Accepted 9 May 2016
Available online 13 May 2016

Keywords:
IEEE 802.15.4e
TSCH
DSME
LLDN
WSANs

ABSTRACT

Several studies have highlighted that the IEEE 802.15.4 standard presents a number of limitations such as low reliability, unbounded packet delays and no protection against interference/fading, that prevent its adoption in applications with stringent requirements in terms of reliability and latency. Recently, the IEEE has released the 802.15.4e amendment that introduces a number of enhancements/modifications to the MAC layer of the original standard in order to overcome such limitations. In this paper we provide a clear and structured overview of all the new 802.15.4e mechanisms. After a general introduction to the 802.15.4e standard, we describe the details of the main 802.15.4e MAC behavior modes, namely *Time Slotted Channel Hopping* (TSCH), *Deterministic and Synchronous Multi-channel Extension* (DSME), and *Low Latency Deterministic Network* (LLDN). For each of them, we provide a detailed description and highlight the main features and possible application domains. Also, we survey the current literature and summarize open research issues.

1. Introduction

Wireless sensor and actuator networks (WSANs) will play a key role in the realization of the future *Internet of Things (IoT)* [1–3] since they represent the main way through which any computational system can interact with the physical world [4–5]. In a WSAN many sensor and actuator devices are placed in the same physical environment for monitoring and control operations. Physical quantities, such as temperature, pressure and light intensity are continuously measured by sensor devices. Then, the acquired data are sent to a central controller using wireless links. The controller analyzes the received information and, if needed, changes the behavior of the physical environment through actuator devices. WSANs are already used in many application domains, ranging from traditional environmental monitoring and location/tracking applications to critical applications such as those in the industrial, smart grid and healthcare domain [6]. In the industrial field, WSAN applications include real-time monitoring of machinery health, factory automation, distributed and process control, detection of liquid/gas leakage, radiation check and so on. In the healthcare domain, WSANs are used for the monitoring of physiological data in chronic patients and transparent interaction with the healthcare system. In smart grid, WSANs have been recognized as a promising technology to achieve seamless, energy efficient, reliable, and low-cost remote monitoring and control of the electric power system. The real-time information gathered from WSANs can be analyzed to diagnose problems early and serve as a basis for taking remedial actions [7,8].

Energy efficiency is usually the main concern in the design of a WSAN. This is because sensor/actuator devices are typically powered by batteries with a limited energy budget and their replacement can be very expensive or even impossible [9]. However, in many application domains, additional requirements such as *reliability, timeliness* and *scalability* need to be considered as well [9–11]. *Reliability* and *timeliness* are very critical issues for industrial and healthcare applications. If data packets are not delivered to the final destination, correctly and within a pre-defined deadline, the correct behavior of the system (e.g., the timely detection of a critical event) may be compromised. The maximum allowed latency depends on the specific application and ranges from tens of milliseconds (e.g., for discrete manufacturing and factory automation), to seconds (e.g., for process control), and even minutes (e.g., for asset monitoring). Finally, *scalability* is fundamental as WSANs can be composed of hundreds to thousands of nodes.

In recent years many standards have been issued by international bodies to support the development of WSANs in different application domains. They include *IEEE 802.15.4* [12], *ZigBee* [13], *Bluetooth* [14], *WirelessHART* [15] and *ISA-100.11a* [16]. At the same time, the *Internet Engineering Task Force (IETF)* has defined a number of protocols to integrate smart objects (i.e., sensor/actuator devices) into the Internet [17]. The most important of them are the *IPv6 over Low power WPAN* (6LoWPAN) [18] adaptation layer

* Corresponding author.
E-mail addresses: domenico.deguglielmo@unipi.it (D. De Guglielmo), simone.brienza@for.unipi.it (S. Brienza), giuseppe.anastasi@unipi.it (G. Anastasi).

http://dx.doi.org/10.1016/j.comcom.2016.05.004
0140-3664/

protocol, the *Routing Protocol for Low power and Lossy networks* (RPL) [19,20], and the *Constrained Application Protocol* (CoAP) [21] that enables web applications on smart objects.

The IEEE 802.15.4 standard [12] defines the physical and MAC (*Medium Access Control*) layers of the protocol stack and is considered the reference standard for commercial WSNs. In fact, many products compliant to this standard are available today. Many studies have investigated the IEEE 802.15.4 performance in WSANs [22–31]. These works highlighted that IEEE 802.15.4 has a number of limitations (such as low communication reliability and no protection against interferences/fading) that make it unsuitable for applications having stringent requirements in terms of latency, reliability, scalability or operating in harsh environments [32]. In order to overcome such limitations, in 2008 the IEEE set up a Working Group (named *802.15 Task Group 4e*) with the aim of enhancing and adding functionality to the 802.15.4 MAC, so as to address the emerging needs of embedded applications. The final result was the release of the 802.15.4e standard in 2012 [32]. The 802.15.4e improves the old standard by introducing mechanisms such as *time slotted access, multichannel communication* and *channel hopping*. Specifically, it defines five new MAC protocols (called *MAC behavior modes*) to support specific application domains and some *general functional enhancements* that are not designed for specific applications. In this regard, the 802.15.4e standard document assumes that readers are quite familiar with the original 802.15.4 technology and presents a significant amount of references to the original standard. Hence, it is absolutely not easy to read for an inexpert reader. The main goal of this paper is to provide a clear and structured overview of all the new 802.15.4e mechanisms. After a general introduction to the 802.15.4e standard, we devote special attention to describe the details of the main 802.15.4e MAC behavior modes, namely *Time Slotted Channel Hopping (TSCH), Deterministic and Synchronous Multi-channel Extension (DSME),* and *Low Latency Deterministic Network (LLDN)*. For each of them, we provide an in-depth description and highlight the main features as well as possible application domains. In addition, we survey the main research works present in the literature and summarize open research issues.

The rest of the paper is structured as follows. In Section 2, we briefly describe the original 802.15.4 standard and highlight the main limitations that motivated the development of the new standard. In Section 3, we present the new 802.15.4e standard and we give a general overview of the introduced enhancements. Then, we focus on the new 802.15.4e MAC behavior modes. Specifically, in Section 4, we describe the TSCH MAC behavior mode. In Section 5, we focus on DSME, whereas Section 6 is devoted to describe LLDN. Finally, Section 7 concludes the paper.

2. IEEE 802.15.4 standard

IEEE 802.15.4 [12] is a standard for low-rate, low-power, and low-cost Personal Area Networks (PANs). A PAN is formed by one PAN coordinator which is in charge of managing the whole network, and, optionally, by one or more coordinators that are responsible for a subset of network nodes. Regular nodes must associate with a (PAN) coordinator in order to communicate. The supported network topologies are *star* (single-hop), *cluster-tree* and *mesh* (multi-hop).

The standard defines two different channel access methods: a *beacon enabled (BE)* mode and a *non-beacon enabled (NBE)* mode. The beacon enabled mode provides a power management mechanism based on a duty cycle. It uses a superframe structure (see Fig. 1) which is bounded by *beacons,* i.e., special synchronization frames generated periodically by the coordinator node(s). The time between two consecutive beacons is called *Beacon Interval* (BI), and is defined through the *Beacon Order* (BO) parameter ($BI = 15.36 \cdot 2^{BO}$ ms, with $0 \leq BO \leq 14$). Each superframe consists of an active period and an inactive period. In the active period nodes communicate with their coordinator, while during the inactive period they enter a low power state to save energy. The active period is denoted as *Superframe Duration* (SD) and its size is defined by the *Superframe Order* (SO) parameter ($SD = 15.36 \cdot 2^{SO}$ ms, with $0 \leq SO \leq BO \leq 14$). It can be further divided into a *Contention Access Period* (CAP) and a *Contention Free Period* (CFP). During the CAP, a slotted CSMA-CA algorithm is used for channel access, while in the CFP communication occurs in a *TDMA (Time Division Multiple Access)* style by using a number of *Guaranteed Time Slots* (GTSs), pre-assigned to individual nodes. In the non-beacon enabled mode there is no superframe, nodes are always active (energy conservation is delegated to the layers above the MAC protocol) and use an unslotted CSMA-CA algorithm for channel access.

2.1. Limitations of 802.15.4

The performance of the 802.15.4 MAC protocol, both in BE mode and NBE mode, have been thoroughly investigated in the past [22–31]. As a result, a number of limitations and deficiencies have been identified:

- *Unbounded Delay*. Since the 802.15.4 MAC protocol, both in BE mode and NBE mode, relies on a CSMA-CA algorithm, it cannot provide any bound on the maximum delay experienced by data to reach the final destination.
- *Limited communication reliability*. The 802.15.4 MAC in BE mode provides a very low delivery ratio, even when the number of nodes is not very high. This is mainly due to the inefficiency of the slotted CSMA-CA algorithm used for channel access. A similar behavior can occur also in the NBE mode when a large number of nodes start transmitting simultaneously (e.g., in event-driven applications).
- *No protection against interferences/fading.* Interference and multi-path fading are very common phenomena in wireless networks. Unlike other wireless network technologies such as Bluetooth [14], ISA 100.11a [16] and WirelessHART [15], the 802.15.4 MAC uses a single-channel and has no built-in frequency hopping mechanism to mitigate the negative effect of interferences and multi-path fading. Hence, the network is subject to frequent instabilities and may also collapse.
- *Powered relay nodes.* The 802.15.4 standard supports both single-hop (star) and multi-hop (peer-to-peer) topologies. In principle, the BE mode could be used to form multi-hop PAN with a tree topology where intermediate nodes do not need to stay active all the time. However, setting multi-hop topologies in 802.15.4 BE mode requires complex mechanisms of synchronization and beacon scheduling that are not specified by the standard [33,34]. To overcome these limitations, in many applications, the intermediate relay nodes in 802.15.4 multi-hop networks keep their radio on all the time, causing a large energy consumption.

For these reasons, 802.15.4 is unsuitable for many critical scenarios, where applications have stringent requirements in terms of timeliness and/or reliability.

3. IEEE 802.15.4e

In 2008, the IEEE created the *802.15 Task Group 4e* with the aim to redesign the existing 802.15.4 MAC protocol so as to overcome its limitations. The goal was to define a low-power multi-hop MAC protocol, capable of addressing the emerging needs of embedded (industrial) applications. The final result was the *IEEE 802.15.4e MAC Enhancement Standard* document [32], approved in

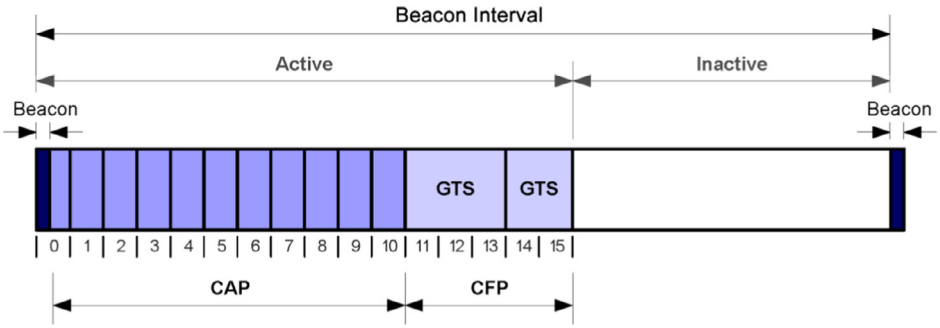

Fig. 1. IEEE 802.15.4 Superframe Structure.

2012. 802.15.4e borrows many ideas from existing standards for industrial applications (i.e., WirelessHART [15] and ISA 100.11.a [16]), including slotted access, shared and dedicated slots, multi-channel communication, and frequency hopping. Specifically, 802.15.4e extends the previous 802.15.4 standard by introducing *MAC behavior modes*, i.e., new MAC protocols designed to support specific application domains, and *general functional enhancements*, i.e., MAC modifications/mechanisms that are not tied to any specific application domain.

3.1. General functional enhancements

IEEE 802.15.4e introduces the following *general functional enhancements*

- *Low Energy (LE).* This mechanism is intended for applications that can trade latency for energy efficiency. It allows a node to operate with a very low duty cycle (e.g., 1% or below), while appearing to be *always on* to the upper layers. This mechanism is important for enabling the Internet of Things paradigm as Internet protocols have been designed assuming that hosts are always on.
- *Information Elements (IE).* It is an extensible mechanism to exchange information at the MAC sublayer.
- *Enhanced Beacons (EB).* Enhanced Beacons are an extension of the 802.15.4 beacon frames and provide a greater flexibility. They allow to create application-specific frames, by including relevant IEs.
- *Multipurpose Frame.* This mechanism provides a flexible frame format that can address a number of MAC operations. It is based on IEs.
- *MAC Performance Metric.* It is a mechanism to provide appropriate feedback on the channel quality to the networking and upper layers, so that appropriate decision can be taken. For instance, the IP protocol may implement dynamic fragmentation of datagrams depending on the channel conditions.
- *Fast Association (FastA).* The 802.15.4 association procedure introduces a significant delay in order to save energy. For time-critical application latency has priority over energy efficiency. Therefore, the FastA mechanism allows a node to associate in a reduced amount of time.

3.2. MAC behavior modes

IEEE 802.15.4e defines five new *MAC behavior modes*, that are listed below,

- *Time Slotted Channel Hopping (TSCH).* It targets application domains such as industrial automation and process control, providing support for multi-hop and multi-channel communications, through a TDMA approach. For further details, refer to Section 4.
- *Deterministic and Synchronous Multi-channel Extension (DSME).* It is aimed to support both industrial and commercial applications with stringent requirements in terms of timeliness and reliability. To this end, it combines contention-based and time-division medium access, and offers two different channel diversity modes. It is specifically designed for multi-hop and mesh networks. For further details, refer to Section 5.
- *Low Latency Deterministic Network (LLDN).* Designed for single-hop and single-channel networks, it is intended for factory automation, where applications require very low latency. For further details, refer to Section 6.
- *Asynchronous multi-channel adaptation (AMCA).* It is targeted to application domains where large deployments are required, such as smart utility networks, infrastructure monitoring networks, and process control networks. In such networks using a single, common, channel for communication may not allow to connect all the nodes in the same PAN. In addition, the variance of channel quality is typically large, and link asymmetry may occur between two neighboring nodes (i.e., a node may be able to transmit to a neighbor but unable to receive from it). The AMCA mode relies on asynchronous multi-channel adaptation and can be used only in non Beacon-Enabled PANs. Basically, in an AMCA network, each device selects the channel with the best local link quality as its *designated listening channel* and starts listening on such a frequency. As soon as two nodes have to exchange packets, the sender device switches to the designated listening channel of the receiver device, in a fully asynchronous way. After transmitting the data packet, the sender switches back to its own designated listening channel and keeps listening. Nodes can exchange information about their designated listening channels by requiring beacon transmissions to coordinators or by sending special *Hello* packets.
- *Radio Frequency Identification Blink (BLINK).* It is intended for application domains such as item/people identification, location and tracking. Specifically, it allows a node to communicate its ID to other nodes without prior association and without acknowledgement. BLINK packets are generally sent by 'transmit only' devices through the Aloha protocol.

The 802.15.4e standard provides only a brief description about AMCA and BLINK. In addition, to the best of our knowledge, no works have been presented in the literature regarding these MAC behavior modes. In this perspective, TSCH, DSME and LLDN are much more interesting. Hence, in the rest of the paper we will focus on these three MAC behavior modes. Specifically, in Table 1 we list their main features, whereas in the following sections we provide an in-depth description, we survey the main research works present in the literature, and summarize open research issues.

Table 1
TSCH, DSME and LLDN's main characteristics.

	TSCH	DSME	LLDN
Beacons	Yes (Enhanced Beacons)	Yes (Enhanced Beacons)	Yes
Time Organization	Periodic Slotframe: - Arbitrary number of timeslots - Dedicated and shared timeslots	Periodic Multisuperframe: - Rigid structure - Recurring CAPs and CFPs	Periodic Superframe: - 3 transmission states - management, uplink, bidirectional timeslots - intended for short timeslots ($<$ 1 ms)
Channel Access	- Time Slotted (dedicated timeslots) - TSCH CSMA-CA (shared timeslots)	- Contention-based (during CAPs) - Time Slotted (during CFPs)	- Time Slotted (dedicated timeslots) - LLDN CSMA-CA (shared timeslots)
Topologies	Star, tree, mesh	Star, tree, mesh	Only star
Multichannel mechanisms	Channel hopping	- Channel hopping - Channel adaptation	No
Timeslot Scheduling Mechanism	Not specified	Distributed GTS Allocation	Centralized
Group ACKs	No	Yes	Yes
Network Synchronization	Frame/Ack-based Synchronization	On Enhanced Beacon Reception	On Beacon Reception

4. TSCH (time slotted channel hopping)

The *Time Slotted Channel Hopping (TSCH)* mode is mainly intended for the support of process automation applications with a particular focus on equipment and process monitoring. Typical segments of the TSCH application domain include oil and gas industry, food and beverage products, chemical products, pharmaceutical products, water/waste water treatments, green energy production, climate control [32].

TSCH combines *time slotted access* with *multi-channel* and *channel hopping* capabilities. Time slotted access increases the potential throughput that can be achieved, by eliminating collision among competing nodes, and provides deterministic latency to applications. Multi-channel allows more nodes to exchange their frames at the same time (i.e., in the same timeslot), by using different channel offsets. Hence, it increases the network capacity. In addition, channel hopping mitigates the effects of interference and multipath fading, thus improving the communication reliability. Hence, TSCH provides increased network capacity, high reliability and predictable latency, while maintaining very low duty cycles (i.e., energy efficiency) thanks to the time slotted access mode. TSCH is also topology independent as it can be used to form any network topology (e.g., star, tree, partial or full mesh). It is particularly well-suited for multi-hop networks where frequency hopping allows for efficient use of the available resources.

4.1. Description

4.1.1. Slotframe structure and synchronization

In the *TSCH* mode nodes synchronize on a periodic slotframe consisting of a number of timeslots. Each node obtains synchronization, channel hopping, timeslot and slotframe information from *Enhanced Beacons (EBs)* frames that are periodically sent by other nodes in order to advertise the network. Basically, upon receiving a valid EB, the node synchronizes to the network, initializes the slotframe and can start sending its own beacons. From this point onwards, the slotframe automatically repeats based on nodes' shared notion of time, and does not require beacons to initiate communications. Fig. 2 left shows a slotframe with 4 timeslots. Each timeslot allows a node to send a maximum-size data frame and receive the related acknowledgement (Fig. 2 right). If the acknowledgement is not received within a predefined timeout, the retransmission of the data frame is deferred to the next time slot assigned to the same (sender-destination) couple of nodes.

Inside a timeslot, data packets are transmitted exactly after *TsTxOffset* μs from the beginning of the timeslot itself. However, to allow for slight desynchronization, the receiver node starts listening the channel *GuardTime* μs before. In addition, if the reception of the packet does not begin within *GuardTime* μs after *TsTxOffset*, the node turns off its radio to save energy. This mechanism requires nodes to never be desynchronized for more than *GuardTime* μs, so as to be able to communicate. Anyway, due to differences in manufacturing, temperature and supply voltage, clocks of different nodes typically pulse at a slightly different frequency, resulting in "clock drift". Hence, nodes need to periodically re-synchronize. To this end, each node is associated to a time-source neighbor, to which it must remain synchronized over time (although the 802.15.4e standard does not detail how such a neighbor must be selected). There are two ways for a node to re-synchronize, namely *Frame-based synchronization* and *ACK-based synchronization*. In *Frame-based synchronization,* every time a node receives a data packet from its time source neighbor, it takes note of the instant the reception started. Then, since *TsTxOffset* is known, it shifts its slot boundaries to match those of its time source. Similarly, in *ACK-based synchronization,* every time a node sends a packet to its time source neighbor, the latter takes note of the instant it started receiving the packet, and inserts the obtained timestamp in a field of the acknowledgment. Once again, the sending node uses this value to realign its clock.

4.1.2. Channel hopping

One of the main characteristics of TSCH is multi-channel communication, based on channel hopping. Initially, 16 different channels are available for communication. Each channel is identified by a *channelOffset*, i.e., an integer value in the range [0, 15]. However, some of these frequencies could be blacklisted (because of low quality communication) and, hence, the total number of channels $N_{channels}$ available for channel hopping may be lower than 16. In TSCH a link is defined as the pairwise assignment of a directed communication between nodes in a given timeslot on a given channel offset [32]. Hence, a link between communicating nodes can be represented by a pair specifying the timeslot in the slotframe and the channel offset used by the nodes in that timeslot. Let [*n, channelOffset*] denote a link between two nodes. Then, the frequency f to be used for communication in timeslot n of the slotframe is derived as follows

$$f = F[(ASN + channelOffset)\%N_{channels}] \quad (1)$$

where *ASN* is the *Absolute Slot Number*, defined as the total number of timeslots elapsed since the start of the network (or an arbitrary start time determined by the PAN coordinator) and "%" is the modulo operator. The ASN increments globally in the network, at every timeslot, and is thus used by nodes as timeslot counter. Function F can be implemented as a lookup table. Thanks to the multi-channel mechanism, several simultaneous communications can take place in the same timeslot, provided that they use different channel offsets. Also, Eq. (1) implements the channel hopping mechanism by returning a different frequency for the same link

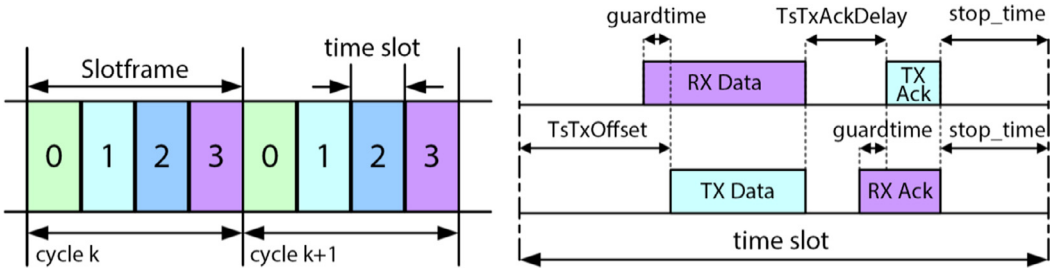

Fig. 2. TSCH Slotframe (left) and Timeslot (right).

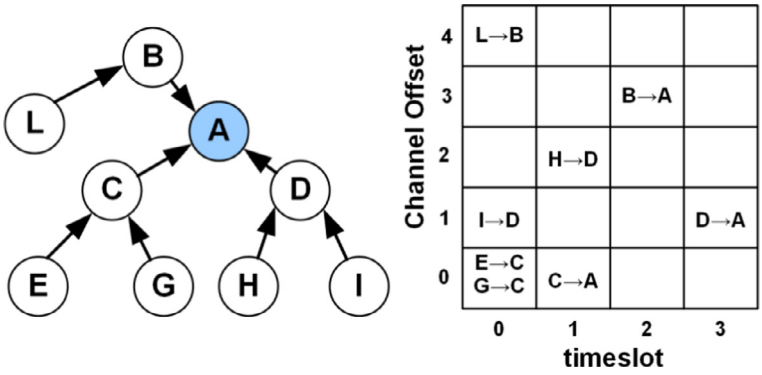

Fig. 3. A sensor network with a tree-topology with a possible link schedule for data-collection.

at different timeslots. This assures that over time all the available channels are used for communications in a link and, hence, allows to mitigate the negative effect of external interference.

Fig. 3 shows a possible link schedule for data collection in a simple network with a tree topology. We have assumed that the slotframe consists of 4 timeslots and there are only 5 channel offsets available. We can see that, thanks to the multi-channel approach used by TSCH, 8 transmissions have been accommodated in a time interval corresponding to 4 timeslots. In the allocation shown in Fig. 3 all links but one are *dedicated* links, i.e., allocated to a single node for transmission. TSCH also allows *shared* links, i.e., links intentionally allocated to more than one node for transmission. This is the case of the link [0,0] allocated to both nodes E and G. Eq. (1) is used to determine the communication frequency for both shared and dedicated links.

4.1.3. TSCH CSMA-CA algorithm

Since shared links can be accessed simultaneously by more than one transmitter, collisions may occur that result in a transmission failure. To reduce the probability of repeated collisions, the standard defines a CSMA-CA retransmission algorithm.

Upon receiving a data frame destined to node r, a sender node s waits for the arrival of the first (dedicated or shared) link assigned to (s, r), and, then, transmits its data frame. If a shared link was used and the transmission was unsuccessful (i.e. the acknowledgment was not received), very likely a collision occurred. Hence, the CSMA-CA algorithm is executed by node s to avoid repeated collisions. Specifically, the following steps are performed by node s.

1. A set of state variables is initialized, namely the number of retransmissions carried out for the on-going frame ($NB = 0$) and the backoff exponent ($BE = macMinBE$).
2. A random number $w \in [0,\ 2^{BE} - 1]$ is generated.
3. The frame retransmission is deferred for w shared links with destination r, or until a dedicated link with destination r is encountered.
4. If the retransmission occurs in a shared link and it is successful (i.e. the acknowledgement is received), the backoff exponent BE is reset to $macMinBE$ and the algorithm terminates. Instead, if the transmission is unsuccessful, state variables are updated as follows: $NB = NB + 1$, $BE = \min(BE + 1,\ macMaxBE)$. Finally, if the number of retransmissions for the current frame has exceeded the maximum allowed value (i.e. $NB > macMaxFrameRetries$) the frame is dropped; otherwise the algorithm falls back to step 2.

If the frame retransmission is carried out in a dedicated link, and it is successful, BE is reset to $macMinBE$, unless there are other frames, destined to the same receiver, ready for transmission. In the latter case the value of BE is left unchanged.

Here we emphasize the differences between the original 802.15.4 CSMA-CA algorithm and the new TSCH CSMA-CA algorithm.

- **Backoff mechanism.** In the original 802.15.4 CSMA-CA each node with a packet ready for transmission waits for a random backoff time before trying to transmit it. The goal is to avoid collisions among nodes starting the execution of the CSMA-CA algorithm at the same time. Conversely, in TSCH CSMA-CA the backoff mechanism is activated only after the node has experienced a collision, i.e., it is used to avoid repeated collisions.
- **Backoff unit duration.** Both the 802.15.4 CSMA-CA and the TSCH CSMA-CA define a backoff unit. In both algorithms a node waits for a random number of backoff units before trying to retransmit a packet. However, while in the original 802.15.4 CSMA-CA the backoff unit is equal to $320\mu s$, in TSCH the backoff unit corresponds to a shared slot. Using a slot as backoff unit assures that a node can experience a collision in a shared slot only if other nodes access the same slot. This is not true in the original 802.15.4 CSMA-CA where, in general, a packet can collide also with packets transmitted at a later time.
- **Clear Channel Assessment (CCA).** In the 802.15.4 CSMA-CA each node performs a CCA, to check the channel state, before performing a packet transmission. This is to avoid a collision with an ongoing transmission. In TSCH, CCAs are not used to prevent collisions among nodes, since all nodes are

synchronized and no transmissions can be ongoing when a CCAs is performed. Conversely, the goal is to avoid transmitting a packet if a strong external interference is detected. In addition, in TSCH CSMA-CA, CCAs are optional.

- **Packet dropping.** In the original 802.15.4 CSMA-CA a packet is dropped by the sender if it has found the channel busy for *macMaxCSMABackoffs* consecutive times. This parameter is not used by TSCH. In TSCH CSMA-CA a packet is dropped only if it reaches the maximum number of retransmissions (specified by the *macMaxFrameRetries* parameter).

4.2. Literature review

In this section we survey the most relevant works regarding TSCH. First we provide a summary of works evaluating its performance. Then, we concentrate on studies that enhance its functionalities or improve its performance. First, we describe works focusing on network synchronization, network formation and node mobility support. Then, we report studies on the impact of adaptive channel hopping in TSCH. Furthermore, we survey both centralized and distributed scheduling algorithms for TSCH networks and works evaluating the cost of network monitoring. Finally, we describe the ongoing work within the 6TiSCH IETF working group and conclude the section by highlighting open research problems and issues.

4.2.1. Performance analysis

In [35] the authors provide a detailed energy-consumption model of TSCH networks. They focus on a single node in the network and perform a fine-grained analysis of the energy spent when it uses different types of slots. Specifically, the authors derive the energy spent during a slot in which the node transmits a packet and receives an acknowledgment as well as the energy consumed during an idle slot. The model has been validated through measurements performed on real hardware. Then, it has been used by the authors to calculate the energy-cost of different synchronization policies as well as the impact of overprovisioning, i.e. having redundant links in the communication schedule.

An advanced model for TSCH networks, that can be used to estimate latency, power consumption and throughput achieved by nodes assuming that the topology, the quality of wireless links and the traffic demands of nodes are known has been presented in [36]. The model has been validated through experimental measurements (performed using a commercial TSCH product called *SmartMesh IP* [37]) and is implemented in a free tool called *SmartMesh Power and Performance Estimator*.

In [38] the scalability of a TSCH network is investigated. The authors consider a network composed of 1 million nodes deployed in an area of 10km^2. This situation is typical in an oil refinery where miles of piping are equipped with hundreds to thousands of temperature, pressure, level and corrosion sensors, which are deployed in a relatively small geographical area. The study proves that such a network can be deployed, provided that 5000 Access Points (i.e. special nodes that collect data generated by nodes) are used. The authors show that the network can achieve a delivery ratio above 99.9%, an end-to-end latency of 2.25 s and a network lifetime of 8.4 years, if 2200mAh AA batteries are used.

4.2.2. Network synchronization

The synchronization mechanism adopted by TSCH must fulfill several requirements, in order to work with constrained devices in critical scenarios. Essentially, time synchronization must be maintained over the whole network with the minimum energy consumption, without requiring the transmission of dedicated synch packets nor introducing delays in the network operation. As explained in Section 4.1.1, according to the TSCH synchronization technique, each node resynchronizes on reception of a data frame or an ACK from its time-source neighbor. Hence, nor special packets are necessary, nor delays are introduced.

In order to avoid desynchronization, resynchs must be performed within a maximum period τ, depending on both the clock drift and the *GuardTime*, as follows

$$\tau = T_g/\Delta_v$$

where T_g indicates the value of *GuardTime* and Δ_v the drift rate. It follows that the larger the guard time or the smaller the drift rate, the less frequently nodes need to re-synchronize. However, typically, the *worst-case drift rate* is used in this calculation. This often results in *"over-synchronization"* and, hence, in a waste of energy, due to the packets exchanged to resynchronize. Recently, some proposals to optimize the network synchronization process in TSCH networks have appeared in the literature [39,40]. They are based on the concept of *adaptive synchronization*. Instead of always re-synchronizing at a worst-case rate, adaptive synchronization allows nodes to resynchronize only when needed. Basically, the actual drift rate between nodes is estimated through real measurements and the effective drift rate is reduced through software corrections that do not require packet transmissions. In detail, each time two nodes exchange a packet to resynchronize, they calculate the offset ε between their clocks. Then, the effective clock drift rate r_{exp} can be derived as $r_{exp} = \varepsilon/\Delta_t$, where Δ_t is the time passed since the last resynchronization. If a node discovers to be faster than the other node, it periodically adjusts its own clock in order to slow down. This mechanism allows each node to track the clock of its neighbor and reduces the effective drift between nodes. However, since the corrections are not perfect and the drift rate can change over time (e.g., with temperature) resynchronization is still needed, but it can be less frequent.

The idea of adaptive synchronization in TSCH networks has been first presented in [39] where the authors have shown, through measurements in a real testbed, that it can reduce the synchronization period by a factor of 10. Then, in [40], mechanisms to use adaptive synchronization in a multi-hop network have been proposed. In this solution, a single node in the network plays the role of time master, whereas all the others synchronize to it. To this end, a routing-tree topology is built using RPL [19] as the routing protocol, and each node uses its parent in the routing tree as time source.

Despite the presence of a routing topology, multi-hop synchronization presents some challenges. Specifically, if nodes at different depths resynchronize at different instants, nodes deeper in the network topology can possibly desynchronize with respect to the rest of the network. For instance, let us consider a linear network of 4 nodes A, B, C and D, where A is the time source of B, B is the time source of C and so on. Also, let us assume that the value of *GuardTime* is 1 ms and that, at some point in time, each node is desynchronized by 400 μs with respect to its time source (see Fig. 4).

Let us assume that, starting from this situation, node B resynchronizes to node A (i.e. they exchange a packet) and node C resynchronizes to node B just after. At this point, node C and D are desynchronized by 1200 μs. Since this is more than *GuardTime*, node D loses synchronization. To avoid this phenomenon, the authors propose to append one additional field to all the ACK packets indicating the resynchronization period of the transmitter. This way, each time a node resynchronizes, it also learns the synchronization period of its time source and, hence, can resynchronize just after it. The authors investigated the performance of their proposal both through simulation and experiments in a real testbed. Their results show that in a 3-hop network, nodes can experience a maximum desynchronization of 76 μs and can reduce the average

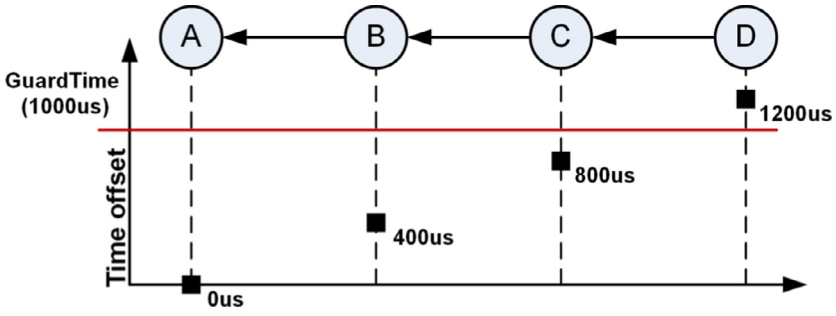

Fig. 4. Multi-hop synchronization.

number of resynchronization packets that are transmitted by 83%, compared to a network not using adaptive synchronization.

4.2.3. Network formation

In TSCH, the network formation process starts when the coordinator begins advertising the network by sending EBs. Hence, if a node wants to join the network, it must turn on its radio and start scanning for possible EB messages. Once received a valid EB, the node can start sending EBs on its turn to announce the network presence. However, the 802.15.4e standard does not define the EB advertising policy. Specifically, it does not indicate which links in the slotframe to use for sending EBs. Also, it does not define the rate at which EBs must be sent. Optimizing the network formation process of TSCH networks is very important. In fact, joining nodes usually keep their radio on during all the time they wait for the EB and, hence, consume a significant amount of energy.

In [41–43], solutions to schedule EB transmissions in TSCH networks with the goal to minimize the average node joining time have been proposed.

In [41], the authors presented a *Random-based Advertisement Algorithm*, where each node in the network is assigned (by the central coordinator) a link in the slotframe to transmit EBs. Each node transmits an EB in the scheduled link with a probability equal to p_{EB}, where p_{EB} is calculated in such a way to minimize the probability of collision between different EBs. The authors used simulations to investigate the impact of different factors on the performance of the algorithm. They found that using more channel offsets for advertisement can significantly reduce the joining time – and, thus, the energy consumption of joining nodes – but only when the network density is quite high.

In [42], two algorithms are proposed, namely *Random Vertical filling (RV) and Random Horizontal filling (RH)*. In these solutions, multiple consecutive slotframes are grouped together to form a *multi-slotframe*. Each node is allowed to send EBs during the first timeslot (namely the *advertisement* slot) of just one slotframe in the multi-slotframe. In both the solutions, the coordinator always transmits its EBs in the first advertisement slot of the multi-slotframe, using channel offset 0. In the RV filling, the other nodes have to transmit their EBs in the same advertisement slot but with a randomly chosen channel offset. Conversely, in the RH filling, they all use channel offset 0, but they randomly choose the advertisement slot in the multi-slotframe. The authors studied the performance of the two EB scheduling solutions through analysis, simulation and real experiments. They found that the RV filling and the RH filling have very close performance.

In [43], the authors model the network formation process by means of a *Discrete Time Markov Chain* (DTMC), and derive an analytical expression of the average joining time. Then, they derive an optimization problem to calculate the optimal EB schedule that minimizes the average joining time. In addition, a *Model-based Beacon Scheduling (MBS)* algorithm is defined, in order to approximate the optimal EB schedule in real scenarios. The authors have performed an extensive simulation study to evaluate the performance of the proposed algorithm in different operating scenarios and compare it with the above-mentioned solutions in the literature. They have observed that MBS outperforms all the other previous algorithms as it reduces the average joining time, and, hence, the energy consumed during the joining phase.

4.2.4. Support to node mobility

Several industrial applications require network nodes to be mobile such as sensors attached to workers or industrial goods. Hence, mechanisms to properly handle node mobility in TSCH networks are needed.

In [44], the authors highlight that node mobility can degrade the network performance. In fact, whenever a node leaves/enters the network, it has to scan the available channels waiting for an EB, in order to join the network and become fully operative again. These scanning times can be very long, given the high number of frequency channels, and, obviously, this affects the latency of mobile nodes' transmissions. This issue is exacerbated by the fact that the beacon and timeslot scheduling mechanisms are not specified by the standard. Hence, the actual joining time of a node depends on the particular beacon/timeslot schedules adopted. In this perspective, the same authors proposed MTSCH [45], a mobility-aware framework, based on the concept of passive beacons. Essentially, instead of using EBs to advertise the network, nodes in MTSCH exploit ACK messages, used to acknowledge packet reception. Moreover, ACK messages are transmitted on a fixed frequency channel, rather than over all the possible frequencies as defined in TSCH. This allows mobile joining nodes to receive synchronization messages (ACKs) more quickly and, hence, to save their energy. In addition, group-ACKs are used. In detail, nodes do not have to send an ACK for every received message, but use a single ACK, at the end of each slotframe, to acknowledge the transmissions of all their neighbors. Thanks to this modification, nodes can save energy due the transmission of individual ACK messages. MTSCH leads to a significant reduction of the duty cycle of mobile nodes, ranging from 7% to 50%, with respect to a standard TSCH network. Also, MTSCH improves the joining time of mobile nodes by a ratio that ranges from 3% to 50%.

In [46], the authors propose a novel solution to optimize routing in a TSCH networks where mobile nodes may be present. They consider a network composed of many static nodes, called *anchor* nodes, with a well-known position, and mobile nodes, whose positions are unknown. All nodes in the network need to transmit their data to the central network coordinator. Hence, the best path connecting them with the coordinator must be selected. Links between anchor nodes are constructed using RPL [19]. In this case, RPL selects links basing on the ETX metric, that indicates the expected number of times a message must be transmitted to be correctly received by its final destination. However, when considering links between mobile nodes and anchor nodes, the actual position of nodes is also taken into account. Specifically, each node

estimates its distance from each anchor node and defines a set of candidate anchors, i.e., a set of nodes that can be selected as parent nodes. Then, a blacklisting process identifies the nodes that should not be selected due to either their unreliability or excessive distance from the mobile node. At this point, the mobile node chooses, as its parent, the node in the set of candidate anchors that minimizes the ETX metric towards the sink. The authors compared their solution with geographical routing [47], where each mobile node forwards its packet to the closest node to the destination, and a solution using the *Packet Reception Rate (PRR)* $\times$ *distance criterion* [48] to select the best anchor node. The results show that the solution in [46] offers the best end-to-end link reliability. Also, the blacklisting process allows to alleviate the negative impact of position errors.

4.2.5. Adaptive channel hopping

Channel hopping is a well-known technique used to mitigate the negative effect of both external interference and multi-path fading. TSCH networks use a simple *blind* channel hopping approach. In practice, nodes use all the possible available channels over time and, on average, for the same duration. Blacklisting [49] is a mechanism that has been shown to further improve the performance of channel hopping. It allows nodes to blacklist channels with a *bad* quality and use channel hopping over only a limited set of *good* channels. In [50], an adaptive channel hopping technique for TSCH networks has been presented. The proposed solution relies on noise floor measurements to decide which channels to blacklist. Specifically, communication between nodes is periodically suspended to acquire noise floor readings. Then, every 512 slotframes, the channels with the highest noise floor levels are blacklisted. Each node inserts the list of its blacklisted channels in a specific field of the EBs it periodically sends, so as to make its neighbors aware of its decisions. The authors evaluated the performance of their solution through experiments performed in a laboratory where 10 WLANs and several Bluetooth nodes were in operation. The obtained results show that the proposed approach can significantly increase the network reliability as well as reduce packet delays.

4.2.6. Link scheduling

A key element in TSCH is the link schedule, i.e., the assignment of links to nodes for data transmissions. Of course, neighboring nodes may interfere and, hence, they should not be allowed to transmit in the same timeslot and with the same channel offset. The multi-channel mechanism makes the link scheduling problem easier with respect to the traditional scenario where a single channel is used. However, finding out an optimal schedule may not be a trivial task, especially in large networks with multi-hop topology. The problem is even more challenging in dynamic networks where the topology changes over time (e.g., due to mobile nodes). Anyway, the IEEE 802.15.4e standard does not specify how to derive an appropriate link schedule.

Transmission scheduling in TDMA-based networks has been a hot research topic in the past years. The vast majority of TDMA scheduling algorithms consider single-channel networks, whereas, only recently, multi-channel TDMA scheduling solutions have been proposed in the literature. Nevertheless, most existing multi-channel scheduling schemes are not suitable for TSCH networks due to one or more of the following reasons: **i**) they cannot be used in TSCH, e.g., they do not allow per-packet channel hopping; **ii**) they have not been designed for resource-constrained nodes and, hence, are not memory efficient; **iii**) they are not efficient in terms of channel utilization, e.g., they do not consider the spatial reuse of channels. Given the limitations of existing solutions, new scheduling algorithms, specifically designed for TSCH networks, have recently appeared in the literature. They can be broadly classified as *centralized* and *distributed*. In centralized solutions, a specific node in the network (usually the network coordinator) creates, distributes and updates the link schedule, on the basis of information received by all the nodes of the network (about network topology and generated traffic). However, the link schedule has to be re-computed and re-distributed every time a change in the operating conditions occurs. Hence, the centralized approach is not very appealing for dynamic networks (e.g., networks with mobile nodes) and large-scale networks, where a distributed approach is typically the best choice. In distributed link scheduling algorithms, the link schedule is computed autonomously by each node, based on local, partial information exchanged with its neighbors. While the overall schedule provided by distributed algorithms is usually not the optimal one, they are more affordable for energy-constrained nodes since their overhead is quite limited. In the following sections we survey the most important scheduling solutions, both centralized and distributed, that have been proposed for TSCH networks. A taxonomy of the surveyed schemes is provided in Fig. 5.

4.2.6.1. Centralized scheduling algorithms. The *Traffic Aware Scheduling Algorithm* (TASA) [51,52] and the *Multichannel Optimized Delay time Slot Assignment* (MODESA) [53] algorithms are the most important centralized scheduling solutions proposed for TSCH networks. Both TASA and MODESA consider a tree network topology, and focus on a convergecast scenario, where collected data must be transmitted to the central coordinator (root). Specifically, TASA assumes that the coordinator has one single radio interface (i.e., the coordinator can receive at most one packet per timeslot), whereas MODESA considers also the case of multiple radio interfaces. Furthermore, TASA assumes heterogeneous traffic conditions (i.e., nodes can generate different amount of traffic), whereas in MODESA all the nodes generate the same number of packets (an extended version of MODESA considering heterogeneous traffic and multiple coordinators has been proposed in [54]).

Both TASA and MODESA aim at finding a communication schedule of minimal length, in order to minimize the number of slots needed to report all data to the coordinator. The generated schedules must be *conflict-free*, in the sense they must avoid error situations such as *duplex-conflicts* and *interference-conflicts*. Duplex-conflicts happen when multiple nodes try to transmit simultaneously to a same receiver. For instance, a parent cannot receive data from more than one of its children at a time. Similarly, in a schedule, it is not possible for a node to transmit and receive during the same slot. Therefore, when a node is transmitting, none of its children can send it data. Conversely, an interference-conflict occurs when two distinct pair of nodes communicate on the same slot and on the same channel, and, hence, one transmission interferes with the other (preventing the correct reception of packets). Naturally, only duplex-conflict free links can be scheduled during the same slot, whereas interference-conflict links can be scheduled during the same slot if different channel offsets are used.

In light of these considerations, TASA constructs a conflict-free schedule where all the packets generated by sensor nodes are delivered to the network coordinator. TASA uses an iterative procedure to build the TSCH schedule. During each iteration, TASA selects a certain number of links and accommodates their transmissions in the same timeslot (using multiple channel offsets if needed). This process is repeated several times until all the transmissions required by each link of the network have been accommodated. In detail, at each iteration, TASA uses a matching algorithm to select a set of duplex-conflict free links (only links that still have packets to transmit are considered). Then, a vertex coloring algorithm assigns different channel offsets to interference-conflict links. TASA uses vertex-coloring to color only a small set

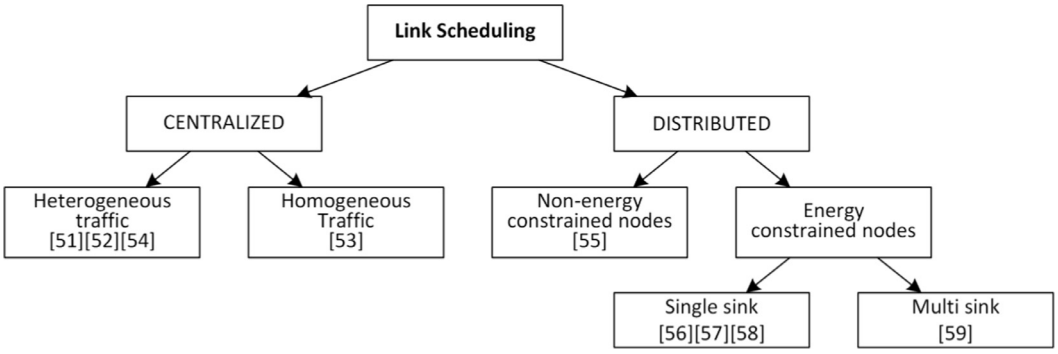

Fig. 5. Taxonomy of link scheduling algorithms.

of links (therefore, it is an affordable operation), i.e. the links that at each step are selected by the matching algorithm. The authors have investigated the impact of the number of channels used for communication on the performance of TASA. They found that using more channels allows to construct a schedule that decreases the delay of packets and increases the throughput of the network. However, they discovered that using more than 3 channels does not provide any substantial advantage. Also, they compared the performance of TASA with the standard 802.15.4 in terms of energy consumption. The results show that TASA significantly decreases the energy consumption of the network and provides a relative energy gain up to 80%.

Unlike TASA, MODESA selects one node at each iteration, and chooses a link to accommodate one of its required transmissions. MODESA terminates its execution, when the transmissions of all nodes in the network have been accommodated. In detail, each node in MODESA has a dynamic priority calculated on the basis of the number of packets it still has to transmit (nodes with an higher number of packets to transmit have higher priority). At the beginning, MODESA picks the node with the highest priority and schedules its first transmission on the first timeslot, on the first channel offset. Then, another node is selected. If the node is in conflict with the previously scheduled node its transmission is allocated on a different channel offset. Otherwise, the same channel offset is used. This process continues until the transmissions of all nodes in the network have been scheduled. MODESA has been shown to be optimal in the case of linear and balanced-tree topologies. In addition, authors have shown that the performance of MODESA is very close to that of an optimal algorithm for random network topologies as well. Furthermore, the authors have shown that using more channels for communication, or more interfaces at the coordinator, can drastically reduce the schedule length.

4.2.6.2. Distributed scheduling algorithms. Since centralized scheduling is not suitable for mobile and/or large-scale networks, several distributed solutions for TSCH networks have appeared in the literature [55–59]. Some of them do not consider energy efficiency as a primary goal [55], while the others specifically focus on networks composed of energy-constrained nodes [56–59]. The proposed solutions consider both single-sink [56–58] and multi-sink [59] networks.

In [55] the authors present two algorithms, namely *Aloha-based scheduling* and *Reservation-based scheduling*, for use in the *Floating Sensor Network* (FSN) project at UC Berkeley. A Floating Sensor Network is composed of a number of mobile, autonomous and motorized sensor packages for deployments in rivers and estuaries. Given the high mobility of nodes, many topological changes occur over time and, hence, a distributed solution is the best choice. Both Aloha-based scheduling and Reservation-based scheduling have the goal to (**i**) establish a bidirectional link between two nodes of the network as soon as they enter the communication range of each other and (**ii**) delete the communication link from the schedule as soon as the nodes are no longer able to communicate. Both Aloha-based scheduling and Reservation-based scheduling use two types of packets, namely *Advertisement* packets and *Connection Request* packets, to establish a link between two nodes. Advertisement packets are used by nodes to announce on which slots and channels they can be reached to establish a link. Specifically, an advertisement packet contains a list of links (slot-channel offset) that are not currently used by the advertising node and, hence, are eligible to accommodate new communications. Connection request packets are sent in response to Advertisements and are transmitted during one of the slots contained in the Advertisement packet, picked at random. If the Connection request packet is acknowledged by the advertising node (i.e., no collisions between nodes occur), the slot used to send the Connection request is marked as a data slot and will be used to transmit data packets. Nodes send and listen for advertisements during idle slots (i.e., not yet allocated slots). In addition to this, in the Reservation-based scheduling, each node includes – inside its Advertisement packets – the list of its neighbors, together with the channels they are listening to.

Both Aloha-based scheduling and Reservation-based scheduling are not suitable for energy-constrained networks, since nodes must always keep their radio on to listen to or send advertisements. In this perspective, distributed scheduling solutions specifically designed for energy-constrained networks have been proposed in [56–59].

In [56] the authors propose to use the Generalized Multi Protocol Label Switching (GMPLS) and the Resource Reservation Protocol - Traffic Engineering (RSVP-TE) in order to construct a TSCH schedule matching the requirements of nodes in terms of bandwidth, latency and energy consumption, in a single-sink topology. Basically, when a new node joins the network, it configures its RSVP layer by indicating its required bandwidth towards the sink. The RSVP layer is in charge of establishing a path towards the sink with the desired bandwidth. To this end, it prepares a PATH message describing the requirements of the path and passes it to the GMPLS layer that forwards the message upstream towards the sink. At this point, the RSVP layer of the sink creates a RESV message that is sent downstream to the node originating the request. The information contained in the RESV message is used by the GMPLS layer of all the nodes between the originating node and the sink to configure the TSCH schedule. At each node, the schedule is built using the Completely Fair Distributed Scheduler (CFDS). Essentially, CFDS provides a timeslot selection mechanism and a channel offset selection strategy designed to alternate transmission and reception slots at each node (to avoid buffer overflows) and minimize the probability of collisions. The authors compared the performance of their proposal with that of MC-LMAC [60], an

algorithm for multi-channel time-slotted sensor networks, in terms of average delay and throughput. They found that their GMPLS-RVSP-TE proposal provides a lower average delay (almost the half) and a higher throughput.

Another recent distributed scheduling algorithm for single-sink TSCH networks is Wave [57]. Wave has the goal to compute a TSCH schedule, minimizing the number of necessary slots. To this end, each node needs to know its parent, the set of its conflicting nodes as well as their traffic demands. Wave builds the network schedule by constructing a series of waves, i.e., matrixes composed of a certain number of cells (slot, channel offset) to accommodate packet transmissions. Basically, at the network startup, the coordinator sends a START message to trigger the computation of the first wave. When a node receives the START message and has the highest priority among its conflicting nodes (since each node has an assigned priority depending on the number of packets it still has to transmit), it assigns to itself a cell in the wave and notifies its conflicting nodes by sending an ASSIGN message. Then, the process repeats until all the nodes have selected a cell in the wave to transmit their first packet. When the computation of the first wave is completed, the coordinator sends a REPEAT message to trigger the construction of the second wave. The second wave (as well as all the successive ones) is a copy of the first wave, where the slots that are no more needed by any node are removed. In total, Wave constructs a number of consecutive waves equal to the maximum number of packets a single node in the network has to transmit. The authors compared the performance of Wave with that of an optimal algorithm, in a number of different scenarios. The results show that Wave provides performance similar to the optimal algorithm. An extension of Wave, where subsequent waves overlap in time has been presented in [58].

Differently from the previous solutions, DeTAS [59] specifically addresses multi-sink networks. Essentially, DeTAS is the distributed version of TASA [51] and aims at building a conflict-free TSCH schedule for RPL-organized networks having multiple sinks. The authors consider the case where every node in the network needs to send its data to one specific sink. Hence, multiple Routing Graphs exist in the network, each of which is rooted at a different sink node. For each Routing Graph, DeTAS constructs a micro-schedule accommodating the transmissions of the nodes belonging to it. Then, the different micro-schedules are composed to form a global macro-schedule. To avoid interference between nodes belonging to different Routing graphs, each micro-schedule is built using a dedicated set of 3 channels. It follows that at most 5 micro-schedules can be scheduled in parallel. In addition, micro-schedules are built in such a way that reception and transmission slots alternate at each node. The authors compared the performance of DeTAS and that of TASA, in terms of maximum queue occupancy. The results show that DeTAS provides a lower queue occupancy with respect to TASA. In addition, the queue sizes result to be practically deterministic. The same authors in [61] present an implementation of DeTAS in the OpenWSN protocol stack. They analyze its performance in terms of end-to-end latency, reliability and duty-cycle by means of experimental measurements, showing its effectiveness even in real environments.

4.2.7. Impact of network monitoring

Both centralized and distributed scheduling solutions for TSCH networks require up-to-date information about the status of communication links in order to construct a high-quality schedule. In [62] the authors focus on evaluating the impact of network monitoring in TSCH networks. They propose to piggy-back network health status information in regular data frames, using a specific Information Element. Then, the cost of network monitoring has been evaluated both in centralized and distributed scheduling scenarios. The results show that the overhead introduced by network monitoring is significantly higher in centralized solutions than in distributed ones. This is because in centralized solutions status information of each link need to be reported to the central scheduler requiring, on average, many transmissions. Conversely, in distributed solutions less transmissions are performed since link status information needs to be known by the link local neighborhood only. Finally, the authors highlight that mechanisms for in-network processing are needed in order to reduce the monitoring overhead.

4.2.8. Autonomous scheduling

In many scenarios network traffic is usually unpredictable. Also, nodes join and leave the network frequently and typically generate different amount of data. Many centralized and distributed scheduling solutions for TSCH networks assume static network topologies and require that the traffic pattern of each node is known a priori. Thus, they are not suitable for dynamic networks. Motivated by these considerations, in [63] the authors propose *Orchestra*, a solution for autonomous scheduling of TSCH transmissions in RPL-based networks. In Orchestra, nodes compute their own transmission schedule, without relying on any central or distributed scheduling entity. In addition, Orchestra does not require any negotiation, signaling or multi-hop path reservation between nodes. Basically, Orchestra introduces an abstraction layer over TSCH. It is based on a virtual multi-slotframe structure that can accommodate different traffic types, such as TSCH beacons, RPL traffic or application data. Virtual slotframes are composed of a number of virtual Orchestra slots. However, an Orchestra slot does not correspond to a single TSCH timeslot. In fact, according to the obtained RPL information, an Orchestra slot can be mapped into none, one or more TSCH slots. This way, for instance, a single virtual slot can be used by a node to communicate with all its children or its preferred parent. In order to determine which TSCH slots associate to a particular virtual slot, Orchestra considers an hash function calculated on the MAC addresses of the node and its neighbors. This allows to reduce contention or even eliminate it in certain cases. The authors implemented Orchestra in the Contiki OS and tested its performance in two different testbeds. Their results show that Orchestra achieves an end-to-end delivery ratio higher than 99.99% and, compared to state-of-the-art asynchronous low-power MAC protocols, Orchestra improves reliability by two orders of magnitude while obtaining a similar latency-energy balance.

4.3. 6TiSCH IETF working group

In the perspective of the future Internet of Things (IoT) it is important to integrate TSCH within the IoT protocol stack. To this end, the 6TiSCH working group has been created by the IETF with the goal to enable IPv6 over TSCH [64–69]. Specifically, the target of 6TiSCH is to provide mechanisms for combining the high reliability and low energy consumption of TSCH with the ease of interoperability and integration offered by the IP protocol. In 6TiSCH, the TSCH MAC mode is placed under an IPv6-enabled protocol stack, running *IPv6 over Low-Power Wireless Personal Area Network* (6LoWPAN), the *IPv6 Routing Protocol for Low-Power and Lossy Networks* (RPL), and the *Constrained Application Protocol* (CoAP). To properly integrate TSCH with upper layer protocols, *6TiSCH* is defining a new functional entity in charge of scheduling TSCH time slots for frames to be sent on the network. In fact, while IEEE.802.15.4e standard defines the mechanisms for a TSCH node to communicate, it does not define the policies to build and maintain the communication schedule, match that schedule to the multi-hop paths maintained by RPL, adapt the resources allocated between neighboring nodes to the data traffic flows, enforce a differentiated treatment for data generated at the application layer and signaling messages needed by 6LoWPAN and RPL to discover

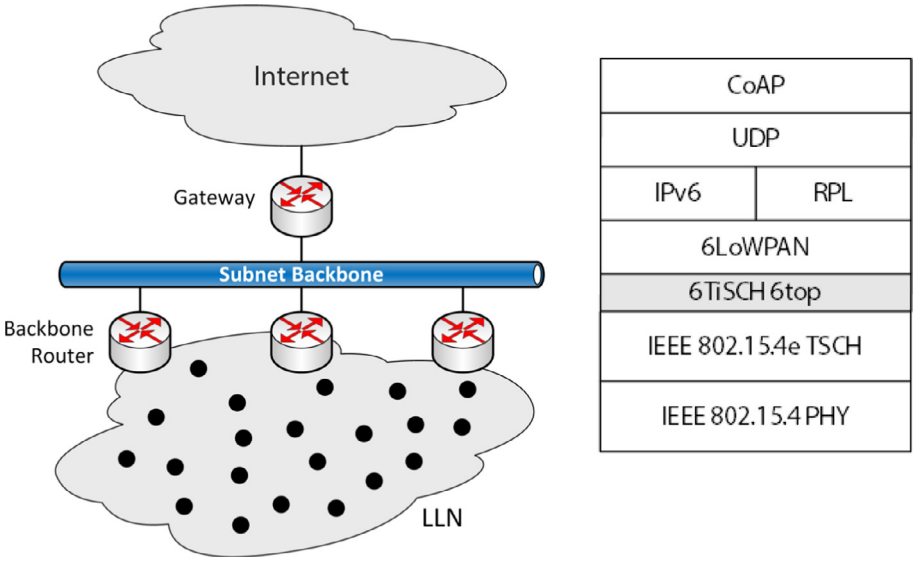

Fig. 6. 6TiSCH Architecture (left) and 6TiSCH Protocol Stack (right).

neighbors and react to topology changes. In the following, we describe the details of the 6TiSCH architecture and protocol stack as well as the scheduling and routing mechanisms that are currently under definition by part of the 6TiSCH working group.

4.3.1. 6TiSCH architecture

The 6TiSCH working group considers a *Low Power Lossy Network (LLN)* composed of hundreds to thousands of nodes deployed in a certain physical environment and using TSCH at the MAC layer. All the nodes in the network belong to the same IPv6 subnet, communicate over IPv6 and use the 6LoWPAN Header Compression (6LoWPAN HC) to transmit packets. To allow the network to scale up to the thousands of nodes and be seen as a single IPv6 subnet, the presence of a high-speed backbone (e.g., a wireless mesh network using 802.11) spanning the entire physical environment is assumed. The backbone provides a fast infrastructure to interconnect and synchronize all the nodes. Constrained nodes are attached to the backbone through one or multiple *Backbone Routers (BBRs)* while the entire backbone is connected to the Internet through a *Gateway*. Fig. 6 left shows the overall architecture.

4.3.2. 6TiSCH protocol stack

6TiSCH aims to combine the IEEE 802.15.4 PHY and IEEE 802.15.4e TSCH MAC layers with higher IETF layers (i.e., 6LoWPAN, RPL, CoAP, etc.) so as to create an open-standard based protocol stack for deterministic IPv6-enabled wireless mesh networks. The 6TiSCH protocol stack is depicted in Fig. 6 right. CoAP enables RESTful interactions with nodes of the network, RPL constructs and maintains a routing topology while 6LoWPAN compacts IPv6 headers to reduce the size of packets to transmit over the wireless medium. Finally, TSCH provides high and deterministic performance.

To allow IETF protocols to operate on top of TSCH in an optimal way, some missing gaps need to be filled. Specifically, a new sublayer, called *6top,* is under definition by the 6TiSCH WG. The 6top sublayer works on top of TSCH and allows a *Management Entity* (ME) to control the TSCH schedule, e.g., to add or remove links (namely *cells*, according to the 6TiSCH terminology). In addition, 6top collects connectivity information that can be useful for upper layers (e.g. RPL) and monitors the performance of cells so as to allow to reschedule them if they are not behaving as expected. 6top has been designed to be used both with centralized and distributed scheduling approaches. To this end, it classifies each cell as either a *hard* cell or a *soft* cell. A hard cell cannot be dynamically reallocated by 6top since it is typically installed and removed by the central scheduler entity. Conversely, soft cells can be dynamically rearranged by 6top if they have bad performance. Soft cells are usually installed by distributed algorithms working over 6top. However, scheduling algorithms only indicate to 6top how many soft cells need to be scheduled towards a certain neighbor. Then, it is the responsibility of 6top to map each cell to a certain *(slot, channel offset)* in the TSCH schedule (6top will use the soft cell negotiation procedure described in [70] to perform this task). In addition, since a 6TiSCH network can transport different types of traffic (possibly with different QoS requirements), 6top can mark cells with different labels so as to identify different traffic flows, thus allowing for flow isolation. In detail, when a packet enters the 6TiSCH network, the 6top layer of the ingress node identifies the service class the packet belongs to, and marks it accordingly. Then, basing on the assigned label, each node inside the 6TiSCH network can decide the cell during which to transmit the packet.

4.3.3. Scheduling and routing

6TiSCH considers three different modes for building and maintaining the TSCH schedule, namely *minimal* scheduling, *centralized* scheduling and *distributed* scheduling. In *minimal scheduling*, the TSCH schedule is static and either preconfigured or learnt by a node at joining time. The minimal schedule can be used during network bootstrap or when a better schedule is not available. The 6TiSCH minimal configuration draft [71] reports a description of the minimal schedule to use in 6TiSCH networks. In *centralized scheduling* [72], a specific entity in the network called *Path Computation Element (PCE)*, collects network state information and traffic requirements of all the nodes. Then, it builds the schedule, making sure that the QoS requirements of all the network flows are met. Finally, it installs the schedule into the network. 6TiSCH will define the protocols to be used to exchange messages between the PCE and the nodes in the network and the format of the control messages to be exchanged. In *distributed scheduling* [61], nodes agree on a common schedule by using distributed multi-hop scheduling protocols and neighbor-to-neighbor scheduling negotiation. In detail, a reservation protocol will be used to transport QoS requirements along a certain path. Then, the 6top sublayer at each hop of the path will be responsible to start a negotiation with the next hop node to decide which and how many cells allocate to satisfy the QoS requirements of the path. The 6TiSCH WG is

Table 2
TSCH: main research fields and contributions.

Performance analysis	[35,36,38]
Network synchronization	[39,40]
Network formation	[41–43]
Support to node mobility	[44–46]
Adaptive Chanel hopping	[49,50]
Link scheduling	CENTRALIZED [51–54] DISTRIBUTED [55–59,61]
Impact of Network monitoring	[62]
Autonomous scheduling	[63]
6TiSCH	[61,64–73]

currently identifying which protocols could be used to transport QoS requirements. Also, strategies for cells allocation are under definition. In this context, in [73], an *On-the-Fly (OTF) bandwidth reservation* mechanism is presented. Basically, OTF is an algorithm that aims at adapting the TSCH schedule of a node to its actual transmission requirements. It is run locally by each node and leverages the 6top sublayer. In detail, OTF constantly monitors the amount of data being sent towards each of the node's neighbors, and if this amount becomes too large (small) compared to the number of cells scheduled to that neighbor, OTF asks 6top to add (delete) cells. OTF has been implemented in OpenWSN and showed to be able to adapt the TSCH schedule to time-varying traffic loads, providing low end-to-end latency and high communication reliability.

Regarding the routing layer, once again the 6TiSCH architecture aims at enabling both distributed and centralized routing solutions over a dynamic communication schedule. Pros and cons of the two approaches are similar to the ones already presented for centralized and distributed TSCH scheduling solutions, in Section 4.2.6.

4.4. Summary & open issues

TSCH is a new MAC protocol that combines time-slotted access with multichannel and channel hopping capabilities. These advanced features, as well as its ability to easily support mesh networks, make TSCH one of the most promising technologies to enable the future Internet of Things. TSCH has received a strong attention from the research community, as it can be observed from Table 2 that summarizes the main research contributions regarding TSCH.

Also, some commercial products using TSCH technology are already available for end-users (e.g., *SmartMesh IP* [37]). Despite this, many open issues and problems still remain to be addressed. First, the majority of works on TSCH networks focus on a convergecast scenario where a number of nodes need to deliver their data to a central network coordinator. While this is a relevant scenario, with the advent of IoT, node-to-node communication will become a common traffic pattern (if not the most common one). To properly support such kind of traffic, a strong research effort is needed. In particular, lightweight scheduling and routing solutions to quickly establish node-to-node paths in TSCH networks should be designed. Another aspect of TSCH that needs to be optimized/improved is the network formation process. In fact, in the near future, the majority of nodes will be mobile (think for instance to sensors attached to human-body or smart objects moved from one room to the other) and, hence, frequent joins and leaves of nodes will occur. Current solutions for network formation assume that joining nodes have their radio always on while joining. Conversely, it is likely that many TSCH nodes will use a duty-cycling mechanism while joining, in order to save their energy. Hence, state-of-the-art solutions could result inappropriate or inefficient for future real network scenarios. Finally, security in TSCH networks merits special attention. Specifically, despite time-synchronization allows to save a significant amount of energy by allowing sender-receiver couples to synchronize their wake-up times, it also makes the network more prone to jamming attacks. This is because an attacker can easily identify possible transmission points (i.e., the start-time of transmission slots). In particular, it can activate its radio for a reduced amount of time and save its energy. Thus, solutions to make TSCH communications unpredictable by part of an attacker should be defined.

5. DSME (deterministic and synchronous multi-channel extension)

The Deterministic and Synchronous Multi-channel Extension (DSME) has been designed for all those critical applications that require *deterministic delay* and *high reliability*, in addition to *flexibility* and *adaptability* to time-varying traffic and operating conditions. In this perspective, DSME is particularly suitable for many industrial, commercial and healthcare applications, such as factory automation, home automation, smart metering, smart buildings and patient monitoring.

DSME derives from the Beacon Enabled mode defined in the former IEEE 802.15.4 standard [12], but introduces many remarkable enhancements. Like in the IEEE 802.15.4 Beacon Enabled mode, time is divided into *Contention Access Periods* (CAPs) and *Collision Free Periods* (CFPs, see Fig. 1). As mentioned in Section 2, during CAPs, nodes use a slotted CSMA-CA (or ALOHA) algorithm for channel access, while during CFPs they use *Guaranteed Time Slots* (GTSs), in a TDMA style. Compared to the IEEE 802.15.4 standard, DSME extends the number of GTS timeslots and increases the number of frequency channels used (previously limited to only one). By adopting a versatile multi-superframe structure, DSME ensures the necessary *flexibility* to accommodate both periodic and aperiodic (or event-driven) traffic, even in large multi-hop networks. Besides, thanks to two channel diversity strategies, DSME can select dynamically the best communication channels, so as to guarantee *robustness* and *high reliability* even in time-varying channel conditions.

In DSME mode, neighboring nodes can communicate in a point-to-point fashion. Specifically, DSME allows to establish dedicated links between any two nodes of the network, resulting in an ideal solution for *covering multi-hop mesh* networks with *deterministic latency*. DSME is *scalable* and does not suffer from a single point of failure because beacon scheduling and slot allocation are performed in a distributed manner, without relying on any central entity. Moreover, since each pair of nodes can autonomously allocate or deallocate GTS slots according to their needs, DSME is able to quickly *adapt to time-varying traffic and changes in the network topology*, without requiring a slot scheduling computation. This is a major difference with TSCH. In fact, DSME can be profitably used in all those cases in which the number of nodes in the network or the generated traffic change over time, e.g., as an answer to an external event. For instance, we may consider a surveillance system that increases the bitrate of the video stream when a movement is detected, or a pollution monitoring system that decreases the sample time when guard levels are exceeded. In a TSCH network, instead, every variation in the topology or in the exchanged traffic would require to compute again the timeslot schedule, an operation that is not always possible and can require the execution of complex algorithms and a significant packet exchange among nodes. Finally, DSME supports a group acknowledgement option that allows to aggregate the acknowledgements of multiple data frames into a single ACK frame, thus improving the *energy efficiency*.

Given the large variety of options and features, DSME turns out to be one of the most complex modes defined in the IEEE 802.15.4e standard. In the following its main characteristics are described.

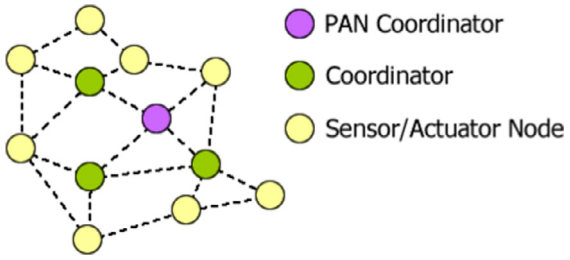

Fig. 7. Example of DSME (mesh) network.

5.1. Description

In this section we describe the main components of DSME. First, we present the DSME multi-superframe structure and describe how DSME uses EBs. Then, we introduce both the DSME channel diversity modes and the DSME Group ACK mechanism. Finally, we describe how GTSs are managed.

5.1.1. Multi-superframe structure

In a DSME network (see Fig. 7), some nodes – referred as *coordinators* – periodically transmit an EB, used to keep all the nodes synchronized and allow new nodes to join the network. The time between two subsequent EBs sent by the same coordinator is called *Beacon Interval* (BI). The latter is composed of several *superframes*. Differently from the IEEE 802.15.4 standard, in DSME there are no inactive periods. Therefore, superframes follow one another seamlessly. Within a Beacon Interval, it is possible to define cycles of repeated superframes, called *multi-superframes*, as shown in Fig. 8.

Like in the IEEE 802.15.4 Beacon Enabled mode, each superframe is divided in 16 equally spaced slots (numbered from 0 to 15) and is composed of three parts namely an *Enhanced Beacon* slot, a *Contention Access Period* (CAP) and a *Collision Free Period* (CFP).

Slot 0 is used to transmit EBs. In particular, each coordinator transmits its EBs only during a superframe that is assigned to it through a distributed scheduling mechanism (see next section).

The CAP starts immediately after the EB and ends before slot 9. It is typically used to transmit control messages and urgent or aperiodic data. During the CAP, nodes use a slotted CSMA-CA (or ALOHA) algorithm for medium access. Both the EB and all the frames sent during the CAP are transmitted using the same channel (the one chosen by the PAN coordinator to form the network and used by nodes to associate). Differently from IEEE 802.15.4, any pair of nodes located within transmission distance can communicate using CAP. Therefore, nodes are required to be always on, for the entire duration of the CAP.

The remaining 7 slots compose the CFP, that is located at the end of the superframe (slots 9-15). Each slot inside the CFP represents a *DSME Guaranteed Time Slot* (DSME-GTS) and is exclusively dedicated to transmissions from a specific source node to a specific destination node. Since there is no contention for accessing the channel, CFP is mainly used to transmit periodic traffic and data frames whose latency must be predictable. Multiple transmissions can be accommodated during the same DSME-GTS by using different channels, thus significantly increasing the capacity of the network. In order to identify a particular DSME-GTS, each superframe inside a multi-superframe has an associated ID. In addition, DSME-GTSs are numbered (i.e., have a Slot ID), based on their position inside the CFP. Therefore, a DSME-GTS can be referred inside a multi-superframe through the pair (Superframe ID, Slot ID).

As mentioned above, nodes need to remain active during all the CAP duration. In order to save energy, DSME provides the *CAP Reduction* mechanism. When CAP reduction is enabled only the first superframe of each multi-superframe presents the CAP, whereas in the other superframes the CAP is omitted and the CFP consists of 15 DSME-GTSs (see Fig. 9).

The time structure described above is regulated by some parameters – namely *macSuperframeOrder* (SO), *macMultisuperframeOrder* (MO) and *macBeaconOrder* (BO) – with $0 \leq SO \leq MO \leq BO \leq 14$ – that determine the duration of superframes, multi-superframes and Beacon Intervals, respectively. Specifically, the length of a superframe is equal to $aBaseSuperframeDuration \times 2^{SO}$ symbols, the length of a multi-superframe is equal to $aBaseSuperframeDuration \times 2^{MO}$ symbols and the length of a beacon interval is equal to $aBaseSuperframeDuration \times 2^{BO}$ symbols, where *aBaseSuperframeDuration* is a constant equal to 960. It follows that a Beacon Interval consists of 2^{BO-SO} superframes, whereas a multi-superframe includes 2^{MO-SO} superframes. For instance, in the multi-superframe structures shown in Fig. 8 and Fig. 9, SO = 5, MO = 7 and BO = 8.

The values of SO, MO and BO are chosen by the network designer and included in the *DSME PAN Descriptor Information Element* of all the EBs that are sent in the network. Obviously, these parameter values must be decided carefully, according to the requirements of applications and the network configuration. For instance, the value of SO influences both the duration of CAP and DSME-GTSs. Therefore, it should be chosen in such a way to obtain a sufficiently large CAP to allow proper contention resolution, and sufficiently large DSME-GTSs to accommodate data frames. Similarly, MO determines the number of DSME-GTSs available in each multi-superframe, but also influences the latency of transmissions.

5.1.2. Enhanced beacons

EBs are sent by each coordinator in the network at regular intervals. Generally, they are directly transmitted over the wireless medium at the beginning of the superframe (during the dedicated EB slot), without CCA or Backoff. However, EBs may experience a collision due to interference with other devices outside the DSME network. In this case, a coordinator can use the *Deferred Beacon* option to reduce the collision probability and improve the reliability of the beacon transmission. Basically, when this option is enabled, the coordinator performs a CCA before sending the beacon. The latter is sent only if the CCA confirms the channel is clear.

Each EB sent in a DSME network includes a special field, namely *DSME PAN Descriptor Information Element*, that contains all the information regarding the superframe structure (i.e., SO, MO and BO) and the enabled options (e.g., Channel diversity mode, CAP Reduction, Deferred beacon, Group ACK, etc.). This way, by receiving an EB, each node can extract the necessary parameter values for correctly operate in the network. An EB also contains the *SDIndex* of the current superframe, i.e., a number used to identify the superframe inside the Beacon Interval. This index starts from 0, that represents the superframe used by the PAN coordinator to transmit its EBs.

In addition, EBs are used to maintain global time synchronization in the PAN. To this end, each node in the network must associate with a coordinator and track its EBs. In detail, when a new node joins the network, it starts a *passive channel scan* over a given list of channels. Specifically, it listens for EBs sent by the network coordinators for a period equal to the maximum possible Beacon Interval period (BO = 14). If no EBs are received in this time, the node switches to the next channel in the list and waits again. Upon discovering its neighbor coordinators, the node associates with one of them. The latter becomes its *time synchronization parent*.

Each coordinator transmits its EBs during a specific superframe in the Beacon Interval, determined through a distributed scheduling algorithm. Basically, a coordinator that has already joined the network shares its beacon schedule information through its EBs. In

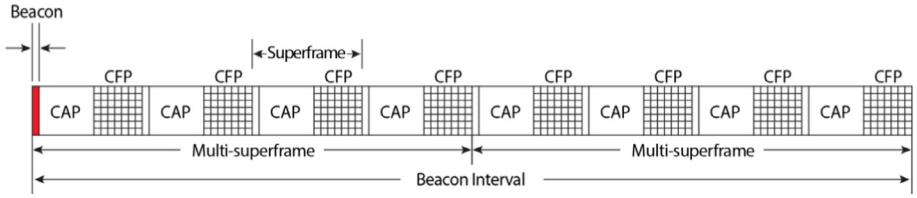

Fig. 8. DSME multi-superframe structure.

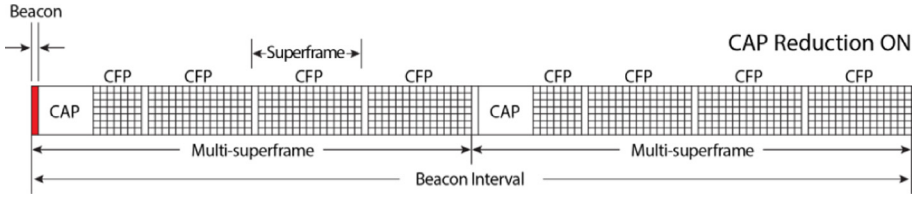

Fig. 9. DSME multi-superframe structure with CAP Reduction enabled.

practice, it sends a bitmap sequence (with a length equal to the number of superframes inside the BI), indicating the usage of EB slots by its one-hop neighbors. A bit in the bitmap is set to one if the corresponding EB slot is already used for beacon transmission. A prospective coordinator searches for an EB slot marked as 0 in all the bitmaps it receives (in fact, it can receive more EBs from different coordinators during a BI). Once a vacant beacon slot has been found, the new coordinator starts using it as its own EB slot. In addition, it transmits a *DSME-Beacon Allocation Notification* command during the CAP, in order to inform its neighbors about the slot allocation. Of course, a collision occurs if two or more nodes try to compete for the same EB slot. For instance, this can happen if two nodes are hidden to each other and cannot listen to each other's transmissions. In this situation, known as *hidden node problem*, the common neighbors of the two nodes will receive the DSME-Beacon Allocation Notification commands of both nodes. To avoid that the same EB slot is assigned to more than one node, a *DSME-Beacon Collision Notification* is sent to the node that has notified later, so as to make it choose a different EB slot.

5.1.3. Channel diversity

In order to provide multi-channel communication during DSME-GTSs, DSME offers two channel diversity methods, namely *Channel Adaptation* and *Channel Hopping*. A DSME network must decide which one of the two methods to use, and announce the choice through a specific flag in EBs.

When the *Channel Adaption* mode is used, two neighboring nodes can decide to communicate with each other using *any* of the free available frequency channels. The channel to be used is decided during the DSME-GTS allocation phase (see the following section), taking into account the channel quality estimated by the two nodes. Basically, each time the allocated DSME-GTS starts (even in subsequent multi-superframes), the two nodes have to switch to the channel negotiated during the allocation. The channel used in a DSME-GTS does not change over time, unless its quality degrades. In this case, it is recommended that the DSME-GTS is deallocated and replaced with a new DSME-GTS with better link quality. An example of the use of Channel Adaptation is illustrated in Fig. 10. In this example, during the second superframe, nodes 1 and 4 use physical channel 12 in slots 0 and 1 and, then, they switch to physical channel 11 on slot 4. Please note that the schedule repeats among subsequent multi-superframes.

In *Channel Hopping* mode, nodes change the communication channel at each DSME-GTS, following a predefined sequence called *Hopping Sequence*. Although the Hopping Sequence is the same for all the nodes in the network (it is decided by the PAN coordinator), different nodes start hopping from different positions in the sequence. The starting position of a node depends on the *Channel Offset*, i.e., an integer chosen for each node during its association to the network (avoiding that the same channel offset is assigned to neighboring nodes). When two nodes want to communicate, the transmitting node has to switch to the channel used by the receiver. This represents the main difference between the channel hopping mechanisms of DSME and TSCH: in TSCH the channel offset is chosen on a per-link basis and is agreed by all the nodes that use the link, conversely, in DSME, each node has its own fixed channel offset and the sender must use the channel offset of the receiver. Also, the formula to derive the channel frequency to be used for communication is different. Specifically, the frequency channel C to be used during a DSME-GTS can be determined as follows:

$$C = macHoppingSequenceList[(i + j \times l + macChannelOffset + macPANCoordinatorBSN) \ \% \ macHoppingSequenceLength]$$

where: *macHoppingSequenceList* represents the channel hopping sequence. Basically, the position to be considered in the sequence depends on the following parameters:

(i) i: Slot ID of the considered DSME-GTS;
(ii) j: SDIndex of the superframe of the considered DSME-GTS;
(iii) l: is equal to 15 if CAP Reduction is enabled and j is not zero, or 7 otherwise;
(iv) *macChanneloffset*: channel offset of the receiver node;
(v) *macPANCoordinatorBSN*: sequence number of the EB sent by the PAN Coordinator;
(vi) *macHoppingSequenceLengt*: length of the hopping sequence.

An example of the schedule of channels and DSME-GTSs with Channel Hopping mode is illustrated in Fig. 11. In this example, the Hopping Sequence is {1, 2, 3, 4, 5, 6} and two nodes, namely Node 1 and Node 2, use Channel Offset values 0 and 2, respectively. We can observe that the two nodes use all the channels of the Hopping Sequence over time.

5.1.4. Group ACK

DSME provides a group Acknowledgement option that can be profitably used when nodes in the network need to send periodic traffic towards their coordinators. Basically, when group ACK is enabled, the coordinator uses a single DSME-GTS to aggregate, in just one frame, all the acknowledgments for data frames received in the previous DSME-GTSs. Also, the group ACK option allows coordinators to specify a DSME-GTS (inside the multi-superframe) in

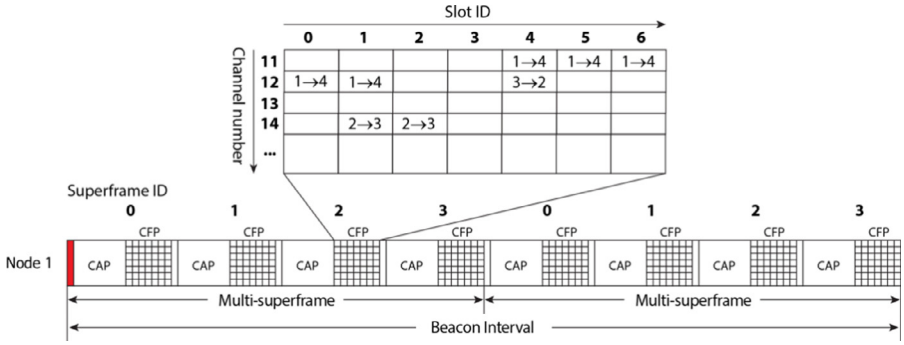

Fig. 10. Channel usage of DSME-GTSs in Channel Adaptation.

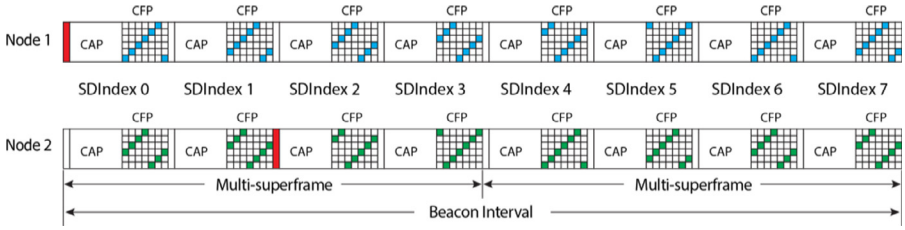

Fig. 11. Channel usage of DSME-GTSs in Channel Hopping.

which those frames that have not been correctly received by the coordinator can be retransmitted.

This way, group ACK achieves a twofold benefit. First, it improves the energy efficiency, since the coordinator does not have to acknowledge each received frame. Second, by providing a retransmission opportunity within the same multi-superframe, it allows to reduce the latency. This can be a crucial requirement for many applications.

Group ACK works as follows. When the application running on a coordinator enables the group ACK option, the coordinator allocates two DSME-GTSs, namely GACK1 and GACK2, that will be used to send acknowledgment frames to the associated nodes. In particular, GACK1 is used to acknowledge the data frames received between slot 0 and GACK1, while GACK2 is used to acknowledge data frames received between GACK1 and GACK2. From this moment, the beacons sent by the coordinator will indicate that the group ACK option is active (through a specific flag) and will contain the superframe and slot IDs of both GACK1 and GACK2.

To take advantage of the GACK feature, a node must allocate two DSME-GTSs towards the coordinator: one before GACK1 and one between GACK1 and GACK2. The former is used to transmit a frame, whereas the latter, called DSME-GTSR (i.e., GTS for Retransmission), is used for retransmission in case of failure on the first attempt. When the node transmits the frame to the coordinator, it knows that it will not receive an immediate acknowledgment, but it has to wait for the GACK1 slot. During GACK1, the coordinator sends a bitmap containing an entry for each DSME-GTS from the beginning of the multi-superframe. If a bit in the bitmap is equal to 1, it means that a data frame was correctly received by the coordinator during the corresponding DSME-GTS. Therefore, by checking the bitmap, the node can understand if its transmission failed. In this case, it can retransmit its data frame using the DSME-GTSR. Once again, the node will find out the outcome of the retransmission by parsing the bitmap sent by the coordinator during GACK2.

5.1.5. DSME-GTS management

The DSME-GTS functionality allows a pair of neighboring nodes to operate on a channel within a reserved portion of the superframe, being sure that no other node of the network will interfere with the communication. A DSME-GTS must be *allocated* before use. Specifically, it is the destination node that decides whether to allocate a DSME-GTS based on the requirements of the *DSME-GTS request* sent by the source node and the current slot availability. In case of a multi-hop flow, DSME-GTSs should be allocated sequentially on each hop to reduce the end-to-end delay. Each DSME-GTS can be *deallocated* at any time by one of the two communicating nodes. In addition, a DSME-GTS can *expire* if (**i**) the transmitter node does no longer use the DSME-GTS, (**ii**) the receiver node does not receive any data frame for *macDSMEGTSExpirationTime* (7 by default) consecutive multi-superframes, (**iii**) the link quality is bad, i.e., no acknowledgment frame is received for *macDSMEGTSExpirationTime* consecutive multi-superframes. In all these cases, the DSME-GTS must be deallocated.

In order to manage the DSME-GTSs, each node stores two data structures, namely *DSME Allocation Counter Table* and *DSME Slot Allocation Bitmap*. In detail, the former is a table containing the following data for each DSME-GTS allocated to the node:

- *Superframe ID*.
- *Slot ID*.
- *Channel ID*. In channel adaptation, this field contains the Channel number (i.e. the physical channel) of the DSME-GTS. In channel hopping, it contains the Channel Offset of the receiving node.
- *Direction*, i.e., Transmission or Reception.
- *Type*, i.e., regular, DSME-GTSR, GACK1 or GACK1 (see previous section).
- *Priority level*, i.e., High or Low.
- *Address* of the node at the other end of the communication.
- *Idle counter*, i.e., the number of idle multisuperframes since the last usage of the DSME-GTS.
- *Link Quality*.

Conversely, the *DSME Slot Allocation Bitmap* is a bitmap used to store which DSME-GTSs in the multi-superframe have been allocated to the node and its one-hop neighbors. In channel adaptation mode, this bitmap contains a bit for each possible pair (*ch, ts*), indicating whether the physical channel *ch*, during GTS *ts*, is used (1) or not (0). For instance, for a superframe containing 7 GTSs, the bitmap contains $7 \times 16 = 112$ bits. Instead, in channel hopping mode, since only the channel specified by the Hopping Sequence

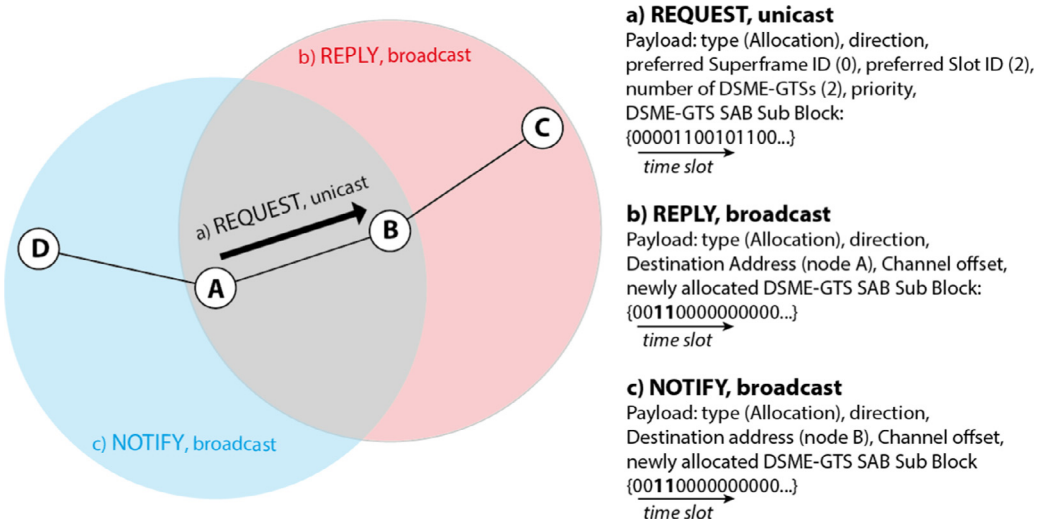

Fig. 12. Example of a handshake for DSME-GTS allocation.

can be used during a GTS, the bitmap is smaller and simply indicates if each GTS in the multi-superframe is used or free.

The scheduling of DSME-GTSs is performed in a fully distributed way. Specifically, the allocation of a DSME-GTS consists in a three-step handshake. As normal for command frames, messages are exchanged during the CAP. Let us consider the situation illustrated in Fig. 12, where node A needs to allocate a DSME-GTS to communicate with node B (in channel hopping mode).

(a) Step 1, Request. Initially, node A transmits a DSME-GTS REQUEST command frame to node B. In its request, node A must specify (**i**) the request type (in this case Allocation), (**ii**) the direction of the communication (i.e., transmission or reception), (**iii**) the preferred Superframe ID and (**iv**) the preferred Slot ID for DSME-GTS allocation, (**v**) the number of requested DSME-GTSs, (**vi**) the priority of the DSME-GTS, and (**vii**) a *DSME-GTS SAB Sub Block*. Specifically, this last field contains a subset of the bitmap described above (i.e., the Slot Allocation Bitmap), and is used to inform node B about the slots that can be used for allocation. Two further fields in the request frame specify the length and starting point of the DSME-GTS SAB Sub Block. After transmission, node A waits for a reply command frame from node B for at most *macMaxFrameTotalWaitTime* symbols. If the reply frame is not received within this time frame, the allocation is assumed to be failed.

(b) Step 2, Reply. Upon receiving a request, node B answers with a REPLY command frame. This frame is broadcast, even though its payload contains the address of node A. At this point, node B must decide which GTS to allocate to node A. If channel adaptation mode is enabled, it must select also the channel to use. It is recommended that the preferred Superframe ID and the preferred Slot ID indicated by the requesting node are taken into consideration for slot allocation. If the preferred slot is not available, the next one is considered, and so on. Specifically, node B compares the received bitmap with its own, in order to find a number of common free slots (i.e., slots marked as 0 in both) equal to the requested number of DSME-GTSs. If DSME-GTSs compatible with the slot schedule of both nodes are found, node B updates its two data structures in order to include the just allocated DSME-GTSs, and a status flag is set to SUCCESS in the reply frame. In addition, the reply frame will contain the indication about which slots have been newly allocated. In channel hopping mode, it will also include the channel offset of node B.

(c) Step 3, Notify. When node A receives the reply command frame, if the status value is SUCCESS, it adds the newly allocated DSME-GTSs (indicated in the received frame) to its data structures. Then it broadcasts a NOTIFY command frame. The notify frame payload contains the address of node B and, once again, indicates the slots that have just been allocated.

If a node other than A or B (e.g., nodes C and D in Fig. 12) receives a successful REPLY or NOTIFY frame, its address will differ from the one contained in the frame payload. In this case, the node must check if the newly allocated slots – specified in the received command frame – are conflicting with the slots that the node has allocated for itself. If so, the node sends a request command frame of type *Duplicated Allocation Notification* to the source of the received command. This command is used to inform the source of the reply or notify frame that the allocation performed is not valid (since the slot was already used) and must be undone. Otherwise, if there is no conflict, the node just updates its bitmap. This way, all the neighbors of both node A and node B can take note of the allocated DSME-GTSs.

Deallocation and duplicated allocation notification operations follow a similar three-message exchange.

5.2. Literature review

5.2.1. Performance comparisons with other MAC protocols

The first studies about DSME consisted in performance comparisons with the original 802.15.4 MAC protocol. These studies clearly show that DSME overcomes the well-known limitations of the former standard version, while providing significant performance improvement. For instance, in [74], the authors evaluate and compare the performance of DSME with that of the IEEE 802.15.4 slotted CSMA-CA MAC protocol, both in single-hop and multi-hop scenarios. They consider a monitoring application with periodic data traffic, and, in order to evaluate DSME, assume that data frames are transmitted in scheduled DSME-GTSs and not during the CAP. Simulation results confirm the 802.15.4 CSMA-CA unreliability problem already known in the literature (e.g., [31]), showing that the provided throughput quickly tends to 0 as the number of connected nodes increases. They also show that the throughput provided by DSME is limited only by the number of available DSME-GTSs in the multi-superframe. Therefore, DSME can always

assure a minimum throughput, that can be raised by enabling the CAP Reduction option (i.e., increasing the number of DSME-GTSs in the multi-superframe). This difference is much more relevant in a multi-hop scenario. In this case, DSME can exploit its multi-channel capabilities to allow simultaneous transmissions, whereas 802.15.4 CSMA-CA can rely just on a single channel, thus providing a lower throughput (up to 12 times if DSME CAP Reduction is enabled). Even in terms of energy efficiency, CSMA-CA exhibits an increase in the energy consumption when the number of network nodes increases, since carrier sensing operations and collisions occur more frequently. On the other hand, DSME shows a significant lower consumption, as, in reserved DSME-GTSs, collisions cannot happen and carrier sensing is not needed. However, the analysis is somehow incomplete, since energy consumption during CAP is not taken into account in the evaluation of DSME.

In [75], the same authors extend the comparison, by considering the case of external WiFi interference. They consider a single interfering WiFi source and study the error frame rate introduced in the DSME network, for different traffic loads and power levels of the WiFi transmission. Simulation results show that, in DSME, the frame error rate cannot exceed 25%. This is due to the fact that a WiFi channel (as defined in the IEEE 802.11b standard) can overlap to at most four of the channels used in IEEE 802.15.4e (out of 16). Hence, using the diversity modes provided by DSME, it is possible to exploit the channels that are not affected by WiFi transmissions to communicate without interferences. Conversely, once again, the performance of 802.15.4 CSMA-CA depends on the number of nodes in the network and quickly degrades when the WiFi traffic or the WiFi transmission power increases. This is due to the fact that the WiFi interference affects the channel occupation, exacerbating the contention of nodes for accessing the medium. In conclusion, in all the considered scenarios, DSME has proven to be much more resistant to interference than 802.15.4.

When referring to IEEE 802.15.4, the most common MAC protocol used during the CAP is the CSMA-CA. However, the standard also allows to use Aloha. Hence, [76] presents a performance comparison, based on both analysis and simulation, between 802.15.4 Slotted Aloha and DSME. The authors consider a biomedical scenario, where some patients wear a number of biomedical sensors that periodically perform measurements and transmit the acquired data to a coordinator node. They use an impulse radio ultra-wideband (IR-UWB) physical layer, because of its low power consumption and low level of interference with other hospital equipment and existing WLAN devices. Also in this scenario, DSME performs much better than 802.15.4 slotted Aloha. As expected, the results show that DSME allows to achieve very reliable communications. Conversely, when using 802.15.4 S-Aloha the delivery ratio quickly drops as the number of nodes increases (it is about 65% for 15 transmitting nodes). In addition, the authors highlight that the transmission delay for each packet is upper-bounded when using DSME, whereas it is unpredictable for S-Aloha.

In addition to comparing DSME with the original 802.15.4 standard, it is useful to perform comparisons with the other MAC behavior modes defined by 802.15.4e, so as to investigate which scenarios and operating conditions make a mode preferable over the others. In this perspective, in [77] the authors compare DSME and TSCH. The evaluation considers a process automation scenario and assumes a cluster-tree network topology. Simulation results show that both protocols are robust towards channel noise, assuring highly reliable communication. In general, TSCH offers better end-to-end delays than DSME when the number of nodes is limited (i.e., below 30 nodes in the considered experiments). Indeed, the rigid structure of DSME multi-superframes does not adapt to the size of the network. When few nodes are connected, the multi-superframe contains more DSME-GTSs than those actually needed. Therefore, nodes experience long delays, waiting for the DSME-GTSs allocated for transmissions. On the contrary, TSCH has a more flexible slotframe structure than DSME, since a single timeslot can be inserted in or removed from the slotframe. However, timeslots have to be larger than in DSME, in order to accommodate ACKs. In DSME, the Group ACK option allows to aggregate all the acknowledgments from a coordinator into a single frame, thus permitting shorter timeslots (without the ACK). It also allows nodes to retransmit a packet after a transmission failure, before the end of the multi-superframe. For these reasons, when the number of nodes grows, more DSME-GTSs in the multi-superframe are used – thus improving the bandwidth efficiency – and the delay values become lower than in TSCH.

5.2.2. General enhancements

The literature on DSME does not include only performance evaluations of DSME and comparisons with other MAC protocols for WSNs. A number of works propose solutions to improve its performance. This is the case of [78], where the authors present a number of enhancements to make DSME more energy efficient in cluster-tree networks. Through simulations, the authors show that the energy consumption of DSME in leaf nodes (typically, energy constrained devices) can be dramatically reduced, by limiting the time spent by their radio in RX state (e.g., during the CAP). To this end, the authors present *ELPIDA* (*Enhancements for Low-Power Instrumentation DSME Applications*), i.e., a proposal that introduces three main changes to DSME: (**i**) *CAP Wake-up*, (**ii**) *DSME-GTS Wake-up*, and (**iii**) *Beacon Look-up*. Essentially, in cluster-tree topologies, it is unusual for parents to send frames to children. Therefore, in normal conditions, end devices can sleep during the CAP (after sending their own frames) or during the DSME-GTSs allocated in reception with their coordinators. If a coordinator needs to send a frame to its associated nodes, it can order a CAP or DSME-GTS Wake-up by setting the *Frame Pending* field inside its EB. Simulation results show that ELPIDA can significantly decrease the energy consumption of end devices, up to a factor 7, if compared with standard DSME (i.e., without ELPIDA).

Another proposal improving DSME performance is presented in [79], where the authors address the association phase. They show that, although IEEE 802.15.4e introduces a new fast association procedure (namely *FastA*), it may require many Beacon Intervals for all the nodes to be correctly associated to the DSME network. For instance, this can happen when many devices perform a passive scan at the same time, and send association requests during the CAP, using CSMA-CA, after receiving a beacon frame from a coordinator. In this case, many collisions occur, causing the repetition of the whole procedure for many consecutive times. To overcome this problem, the authors propose an *Enhanced Fast Association* mechanism (namely *EFastA*). With EFastA, after the reception of the beacon from a coordinator, a node does not sent its association request immediately, but waits for a superframe randomly selected inside the multi-superframe. This way, it is possible to drastically reduce the contention among nodes due to the association handshakes. Through simulation, the authors show that – by properly choosing the Superframe and Multi-superframe duration – EFastA can reduce the time needed for associating all the nodes to the network up to about 90%, with respect to FastA, even for large scale scenarios with hundreds of devices.

5.2.3. Beacon scheduling

The main challenges for constructing scalable multi-hop 802.15.4 networks based on Beacon-Enabled mode concern synchronization and beacon collision avoidance. In order to address these problems, the 802.15.4e standard defines a distributed beacon scheduling mechanism. However, the authors of [80] have shown that the proposed algorithm presents a number of limitations such as (**i**) long waiting time for network construction, (**ii**)

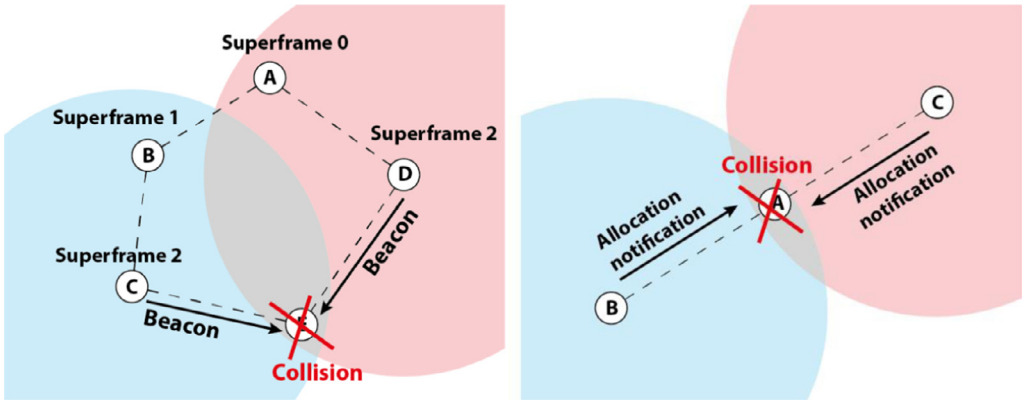

Fig. 13. Scheduling problems: Beacon collision (left) and Allocation notification collision (right).

memory wastage, (**iii**) variable and growing length of EB frame size, (**iv**) potential beacon collision issues. Although the scheduling algorithm of DSME can solve a variety of beacon collision problems, in some situations the distributed algorithm fails. In particular, beacon collisions can happen due to the hidden node problem, since in this case nodes cannot identify the presence of other devices. Fig. 13 shows two critical situations (we assumed that nodes' interference and transmission ranges are comparable). Fig. 13 left represents a *"loop with a hole"* scenario, in which nodes C and D choose the same superframe to send their EBs. In this case, the EBs collide at node E, that cannot associate. In Fig. 13 right, instead, nodes B and C want to associate with A, but their DSME-Beacon Allocation notification messages are transmitted at the same time and collide at the coordinator. Hence, they start using the same beacon slot.

To overcome these drawbacks, in [80], the authors present a new beacon scheduling algorithm, called *DFBS (Distributed Fast Beacon Scheduling)*. They point out that many of the observed problems are caused by the use of a bitmap to represent slot allocation information. Therefore, they replace the bitmap with a single indicator, namely *RINSD (Representative Indicator of Neighbor Superframe Duration)*, that is essentially an integer. Each node maintains its own RINSD and, initially, the PAN Coordinator has a RINSD equal to 0. Differently from DSME, DFBS allows active associations, through a three-step process, during which nodes exchange their RINSDs and – basing on the gathered values – the prospective node chooses its beacon slot. In detail, DFBS works as follows

1. A prospective node that wants to join the network broadcasts (through CSMA-CA) an *Active Association Request* message and waits for a reply.
2. Upon receiving this message, all its one-hop neighbors reply by sending their RINSDs increased by 1.
3. At this point, the prospective node selects the highest among all the received integers and updates the value of its RINSD. The obtained value represents the index of the superframe during which the node will send its own EBs. To notify its decision, the node broadcasts the just determined RINSD through an *Allocated superframe Advertisement*.

To avoid beacon collisions, when receiving an Allocated superframe Advertisement, each node compares the received number with its RINSD. If the received value is higher than the local value, the node updates its RINSD. Otherwise, it notifies the prospective node, that must change beacon slot and update its RINSD number. By means of DFBS, each node can manage slot allocation information by using just a fixed-length variable of 1 or 2 bytes. In addition, simulation results (conducted over many multi-hop network topologies) show that the proposed approach allows to solve the highlighted beacon collision problems. Furthermore, the authors show that by relying on an active association (rather than the passive scan of DSME) it is possible to drastically reduce the network construction time (up to 75% with 25 nodes).

Beacon scheduling issues are addressed also in [81] and [82], in which the authors propose some improvements to the original DSME algorithm and present an enhanced version of the protocol, namely *E-DSME*. Preliminarily, they focus on the criteria to be considered for choosing a beacon slot among the vacant positions in the bitmap (an aspect that the standard does not deal with). Specifically, they define a method, called *most-available-bit* (MAB), that consists in scanning the bitmap from the beginning, looking for the last slot containing a '1'. The searched slot is the next one. Through simulations, the authors show that this approach can minimize the number of nodes that choose the same beacon slot, thus decreasing collisions and speeding up the network formation process. However, this mechanism is not enough to solve all the above-mentioned collision problems. Therefore, the authors present a new distributed permission notification mechanism. Essentially, whereas DSME uses a negative approach – by sending a collision notification only when a node tries to allocate a not available slot – E-DSME relies on a positive one. This means that a node that has sent an allocation notification message must wait for an explicit *Permission Notification* in order to complete the allocation. Only the coordinator that has transmitted the latest beacon can send permission notifications. To make this mechanism effective, the authors propose to divide the CAP into *Allocation Contention Periods* (*ACP*) and *Permission Notification Periods* (*PNP*) that alternate seamlessly. During ACPs prospective nodes compete to transmit allocation notifications towards coordinators, whereas during PNPs only the entitled coordinator can transmit a permission notification. Simulation results show that the enhancements introduced by E-DSME allow to achieve very high success ratio (near to 100%) and to avoid beacon collisions even in complex networks.

5.3. Summary & open issues

DSME is a new MAC protocol that has a great potential, given its flexibility and suitability for many critical application domains. In fact, thanks to its multi-superframe structure, DSME can be used to reliably transmit both periodic and aperiodic traffic. Besides, the distributed beacon and DSME-GTS scheduling algorithms allow to quickly react to time-varying traffic and changes in the network topology.

However, till now, it has not received the same attention as other 802.15.4e behavior modes, e.g., TSCH. In fact, few works are present in the literature about DSME, as it emerges from Table 3, that summarizes the main research contributions regarding DSME.

Table 3
DSME: main research fields and contributions.

Performance comparisons	[74–77]
Energy Efficiency	[78]
Fast Association	[79]
Beacon Scheduling	[80–82]

Indeed, DSME lacks a complete implementation, thus limiting its application in real environments. At present, only partial implementations have been realized in order to evaluate some specific aspects, as reported in the previous section. All the current works are limited to single-hop or cluster-tree networks, and do not investigate the potentialities of mesh topologies, for which DSME can be considered an enabling technology. Hence, complete performance evaluations addressing different aspects such as network formation, beacon scheduling, and DSME-GTS allocation/deallocation, even in mesh networks, are still missing.

In addition, the standard is often unclear when describing DSME, leaving many aspects to the interpretation of implementers. For instance, the standard does not explain what are the criteria to be followed in order to choose the DSME-GTSs to be allocated, to accommodate traffic in GTSs, or to select the communication channel when using channel adaptation. Often, the standard introduces abstract concepts without any concrete outline and implementation details, as in the case of the beacon scheduling algorithm, that presents many evident limitations. Therefore, DSME needs to be enhanced in order to be applied to various topology models and environments. In this perspective, more attention should be devoted to the energy topic. In fact, during the CAP, all nodes (typically battery-powered) are supposed to be in reception mode, that is one of the most energy consuming radio states. Hence, further investigations should be conducted to find solutions that combine energy efficiency, reliability, and reactivity to changing operating conditions. Also, DSME conceals a number of security issues, that should be analyzed and solved. For instance, the distributed DSME-GTS allocation mechanism assumes that all the nodes are trustworthy, which may be a very strong assumption, especially for critical industrial or healthcare applications.

Finally, the feasibility of the integration of DSME with the upper layers of the network stack should be investigated, in particular with reference to the Internet of Things protocols, such as *6LoWPAN, RPL*, and *CoAP*.

6. LLDN (low latency deterministic network)

The LLDN mode specifically addresses the industrial automation application domain, where a large number of devices observe and control the factory production. In this context, wireless communication represents a valid alternative to the cabling of industrial sensors (typically expensive, time-consuming and cumbersome) and also provides advantages in case of mobility and retrofit situations. As an example, LLDN devices can be located on robots, cranes, and portable tools in the automotive industry. They can collect data on machine tools, such as milling machines and lathes, and control revolving robots. Further application areas include control of conveyor belts in cargo and logistics scenarios.

Common requirements for all the above-mentioned applications (and other similar ones) are low latency and high cyclic determinism. As a design target, LLDN shall allow to transmit data from 20 different sensor nodes every 10 ms. To this end, since it has been widely proven that IEEE 802.15.4 PANs cannot fulfill such a constraint (neither with CSMA-CA nor with CFP in the Beacon Enabled mode), the new LLDN mode defines a fine granular deterministic TDMA access.

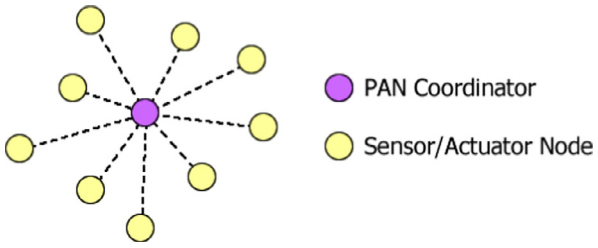

Fig. 14. Topology of an LLDN network.

6.1. Description

6.1.1. Characteristics

Differently from TSCH and DSME, LLDN has been designed only for star topologies, where a number of nodes need to periodically send data to a central sink (i.e., the PAN coordinator), as shown in Fig. 14. In addition, in a LLDN network, all the nodes communicate exploiting just a single frequency channel, i.e., the one chosen by the PAN coordinator during the network formation.

In LLDN, time is divided into superframes that follow one another seamlessly. A node can access the wireless medium during a dedicated portion of the superframe, according to a time division (TDMA) approach. Otherwise, shared group timeslots can be configured, in order to allow multiple access for a group of nodes (using a CSMA protocol). The exclusive channel access, together with the *DSSS (Direct Sequence Spread Spectrum)* coding ensures a highly reliable communication. Obviously, the number of timeslots in a superframe determines how many nodes can access the channel. If many nodes need to send their data, the standard suggests to equip the PAN coordinator with multiple transceivers, so as to allow simultaneous communications on different channels. Compared with TSCH, LLDN provides also a group acknowledgment feature. Besides, LLDN nodes do not have to wait after the beginning of the timeslot in order to start transmitting. Hence, timeslots can be much shorter than in TSCH, since it is not necessary to accommodate waiting times and Acknowledgment frames. In addition, in LLDN, short MAC frames with just a 1-octet MAC header are used, so as to accelerate frame processing and reduce transmission time. In fact, when transmitting during a dedicated timeslot, a node can omit the address fields in the LLDN header, since all frames are destined to the PAN coordinator and the latter knows to which node each timeslot has been allocated. By keeping packets and timeslots short, it is possible to limit the duration of superframes, so as to satisfy the application requirements in terms of latency. In fact, short superframes allow frequent transmissions from each node, with low and deterministic latencies.

6.1.2. Transmission states

An LLDN network passes through three different states before becoming fully operative. The current state is announced by the PAN coordinator in a specific field of the beacon. The three states are discussed below.

1. Discovery. It is the first step during network setup or for addition of new nodes to an existing network. A device that wants to join the network scans the different channels until it detects a beacon from a PAN coordinator, indicating the Discovery state. At this point, the device informs the PAN coordinator, by sending a *Discovery Response* frame. After a predefined number of seconds (256 by default) without receiving any Discovery Response frame, the PAN coordinator switches the network state to the Configuration state.
2. Configuration. It is the second step during network setup. It is also used for network reconfiguration. In this state, each node that receives the beacon sends a *Configuration Status* frame to

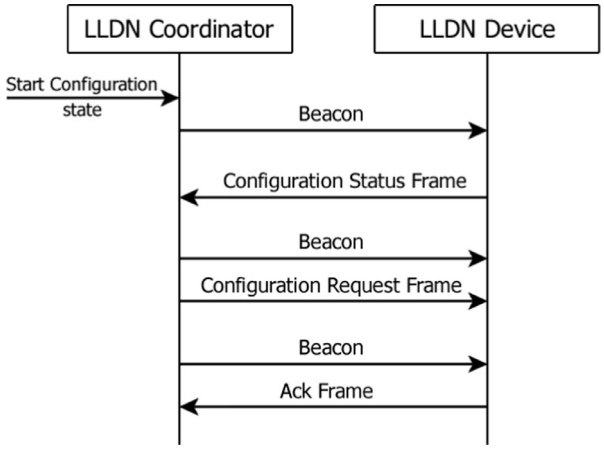

Fig. 15. LLDN Configuration state.

the PAN coordinator. This frame specifies the configuration parameter values currently configured or required by the node, such as the node's short and full MAC addresses, the timeslot duration required by the application (e.g., the size of payload data), the communication direction (i.e., uplink or bidirectional), and the already assigned timeslots. The PAN coordinator shall respond with a *Configuration Request* frame, containing the new configuration parameter values that the node must use during the Online state. Essentially, the Configuration Request frame indicates the timeslots that have been assigned to the node and the timeslot duration. Fig. 15 illustrates the message exchange during configuration state.

3. Online. After the successful completion of the configuration phase, the network can go into Online state. Data and readings from nodes can only be transmitted during the Online state.

6.1.3. Superframe structure

Due to the stringent requirements of low-latency applications, the LLDN mode relies on a minimal superframe structure, represented in Fig. 16. In LLDN, superframes have a fixed duration and are divided into timeslots. Specifically, there are several types of timeslot, according to their function:

1. *Beacon timeslots*. It is always present at the beginning of a superframe and is reserved to the PAN coordinator in order to transmit beacon frames. In LLDN, beacons are used to indicate the start of a superframe and maintain nodes' synchronization. They also contain important information needed for the correct operation of the network.
2. *Management timeslots*. Two management timeslots – one uplink and one downlink – allow nodes to receive and transmit management commands from and towards the PAN coordinator, respectively. Management timeslots are implemented as shared group access timeslots and their size is specified in the beacons sent by the PAN coordinator.
3. *Uplink timeslots*. They are reserved for unidirectional communication of data towards the PAN coordinator. Typically, each node is assigned to a particular timeslot (i.e., a dedicated timeslot) during the Configuration phase. However, more than one device can be assigned to a timeslot, resulting in a shared group timeslot.
4. *Bidirectional timeslots*. They are placed at the end of the superframe, and are used for transmission of data to the PAN coordinator as well as from the PAN coordinator to the node. The direction of the communication for all the bidirectional timeslots is signaled through the *Transmission Direction* bit in the beacon.

The superframe structure changes according to the transmission state. In Discovery and Configuration states, no uplink or bidirectional timeslots are allowed, and, thus, the superframe contains only the beacon slot and the two management timeslots (that occupy the whole superframe). Conversely, in the Online state, management slots are optional. Their presence and length are indicated both in beacons and in Configuration Request frames. In the Online state, the superframe contains *macLLDNnumUploadTS* uplink timeslots and *macLLDNnumBidirectionalTS* bidirectional timeslots (20 and 0 by default, respectively). Usage and order of slots in a superframe during the Online state are shown in Fig. 16.

Each node can send only one data frame during a timeslot. Therefore, the timeslot length (that is the same for all the uplink and bidirectional timeslots) is calculated based on the maximum expected frame size. The obtained length mainly depends on the size of the application data payload, but also considers the overhead introduced by both the physical and the MAC layers, and the Interframe Space (IFS) required to handle incoming frames. Obviously, increasing the timeslot size leads to an increase in the superframe duration (and, hence, in the experienced latency), or to a decrease in the number of timeslots accommodated in the superframe (and, hence, in the number of nodes that can transmit). An analysis of the relationship among payload size, timeslot duration, number of nodes in the network and delay is provided in [83].

In order to acknowledge transmissions towards the PAN coordinator (i.e., uplink), LLDN provides a *Group Acknowledgement* feature. Basically, every beacon contains a bitmap that, for each uplink timeslot of the previous superframe, indicates if the transmission succeeded or failed. For failed packets, it is possible to reserve the initial uplink timeslots specifically for their retransmission (as in Fig. 16a). Alternatively, the Group ACK bitmap can be sent in a separate frame, during a dedicated uplink timeslot (as in Fig. 16b). In this case, the bitmap aggregates the acknowledgements for the previous timeslots of the superframe, whereas the timeslots after the Group ACK slot are reserved for retransmissions. This way, LLDN offers nodes an opportunity to retransmit failed packets inside the same superframe.

The Group ACK mechanism is also applied to bidirectional timeslots, if transmission direction is uplink. Conversely, in case of downlink transmission, each packet is acknowledged by the receiving node with an ACK frame in the following superframe. This assumes that after sending data through bidirectional slots, the PAN coordinator sets the Transmission Direction bit to uplink for the following superframe, so as to receive ACKs.

6.1.4. Channel access within timeslots

In order to regulate the access to the wireless medium, each timeslot can be divided in the following three parts, (as exemplified in Fig. 17).

- *Exclusive (from t_0 to t_1)*. As mentioned above, during the Configuration phase, a timeslot can be assigned to a node. Therefore, when the timeslot starts, its owner has exclusive access from instant t_0 to t_1, and can transmit directly. Obviously, once the transmission has begun, the node can use the entire slot.
- *Contention (from t_1 to t_2)*. If the owner does not use the slot, the residual part can be used by other nodes. In particular, from t_1 to t_2, it can be used by any node other than the PAN coordinator. To this end, the PAN coordinator must notify the availability of the timeslot, by broadcasting a *Clear To Send Shared Group* frame. Then, nodes that are interested in using the slot, shall transmit a *Request To Send (RTS)* frame to the coordinator and wait for the corresponding *Clear To Send (CTS)* frame, before starting data transmission. Since more nodes compete to access the channel, all data and command frames sent between t_1 and t_2 must be transmitted using a simplified version of the

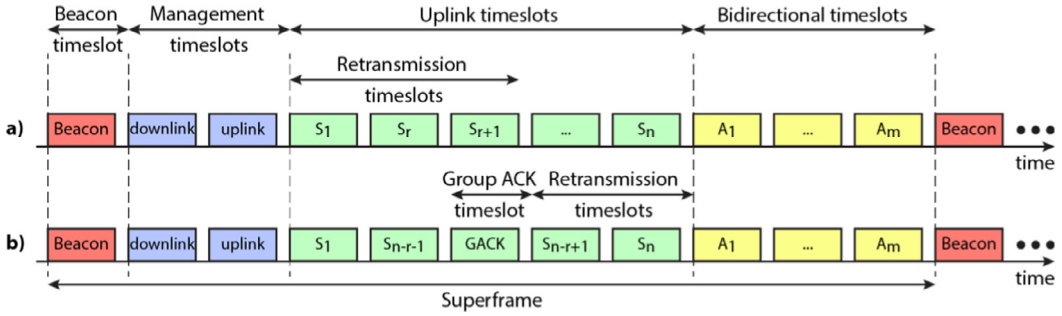

Fig. 16. Superframe in Online state: without (a) and with (b) separate Group ACK timeslot.

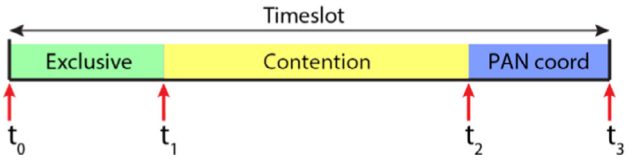

Fig. 17. Timeslot division.

old 802.15.4 CSMA-CA algorithm. By default, this simplified version allows just one backoff stage, meaning that the packet is dropped as soon as a CCA fails.

- *PAN coordinator (from t_2 to t_3).* Finally, from t_2 to t_3, if the timeslot has not been used yet, it can be used by the PAN coordinator.

The size of the three parts can be configured separately for each timeslot, by varying the value of t_0, t_1, t_2 and t_3. For instance, it is possible to reserve the whole slot to a single node (dedicated timeslot) by setting t_1 and t_2 equal to t_3. Similarly, shared group timeslots with contention-based access (e.g., management slots) can be realized by setting t_1 equal to t_0.

6.2. Literature review

Few studies have addressed LLDN so far. The first paper surveyed in this section studies the LLDN protocol from an analytical point of view, aiming at deriving a theoretical model and mathematical expressions of several performance indexes. The other works, instead, propose solutions to improve the protocol performance, i.e., to increase communication reliability, extend network coverage area, or support node mobility. Irrespective of the final objective, these solutions follow a common approach, i.e., they rely on dedicated relay nodes, that act as intermediaries between end devices and the PAN coordinator.

6.2.1. Modeling and performance analysis

The authors of [84] model the channel access in shared slots in LLDN networks, by means of a discrete time Markov chain, considering a single node transmitting its data packet. In their model, they take into account the simplified CSMA-CA protocol specified by the LLDN standard. Specifically, their formulation assumes that the Group ACK option is enabled and dedicated timeslots for retransmission are allocated. It also assumes an ideal channel. Under these conditions, theoretical expressions of reliability, energy consumption, throughput, delay and jitter are derived using the stationary probability distribution of the Markov chain. The authors validate the presented model through Monte Carlo simulations performed with Matlab, considering several scenarios with different number of nodes and packet size. Also, they compare the results obtained with their model with the ones derived from an analytical model of IEEE 802.15.4. Their comparison shows that using the Group ACK option and retransmission timeslots allows to significantly reduce the transmission delay and energy consumption with respect to former 802.15.4 standard.

6.2.2. Improving communication reliability through Relay nodes

In [85], the authors propose a novel Retransmission operating mode for LLDN devices. Basically, the relay node overhears packets sent by the end devices and reads the Group ACK bitmaps sent by the PAN coordinator. If the relay node realizes that a transmission failed, it takes charge of the retransmission. This mechanism allows end devices to save energy (at least 33%), since they do not need to receive acknowledgments for their transmissions, nor are required to perform retransmissions. Moreover, the benefit is greater (up to 50%) when the *Packet Error Rate* (i.e. the fraction of lost packets due to channel corruption) increases. The presence of the relay node also allows to recover a greater amount of packets, since – being deployed between the end devices and the PAN coordinator – it can benefit from a better link quality towards the coordinator.

A key factor to achieve good performance with this approach is the correct deployment of relay nodes. In [85], the authors show that placing the relay node in the middle between a device and the coordinator leads to the best results, with a significant reduction of lost packets. However, such a strategy requires to place a relay node for each LLDN device in the network, which is a very expensive and often unpractical solution. In addition, the energy consumed by the relay node is not negligible, and the total energy consumed (device plus relay) is significantly higher than the energy consumed by a single device. For this reason, in [86], the authors propose a solution that aim at choosing the best deployment for relay nodes in an LLDN network, so as to reduce the packet loss while limiting the overall energy consumption. Their method uses a *Rainbow Product Ranking* algorithm in order to limit the number of required relay nodes and determine their positions. The algorithm takes into account a number of elements, such as the distance with the PAN coordinator and sensor nodes, the accessibility (i.e., presence of walls or other areas where it is infeasible to place nodes), the AC power availability, the presence of objects on the line of sight.

To improve communication reliability, in the same paper [86], the authors also consider Reed Solomon codes as a *Forward Error Correction (FEC)* technique, and propose an Adaptive Retransmission algorithm to switch between the two strategies – i.e., FEC and relay nodes – according to the experienced link quality. The study has been conducted analytically and the proposed solution has been implemented on Matlab. Simulation results show that the Adaptive Retransmission scheme can lead to significant energy savings in relay nodes (up to 85%) compared with the original approach described in [85] (i.e., without the Adaptive Retransmission mechanism).

Always assuming the presence of relay nodes, the authors of [87] propose another method to further increase the communication reliability in LLDN networks. They suggest to apply the

Combinatorial Testing method in order to try to recover a packet, whenever both its transmission and the corresponding retransmission (from the relay node) fail. This method consists in comparing the two erroneous frames at the destination, so as to detect the different (wrong) bits. Afterwards, different combinations of the identified bits are tried, until the CRC test is passed. The so-obtained frame is then assumed to be correct. The authors also propose a similar method to be applied when the two erroneous frames are equal and, thus, the wrong bits have the same position. They have verified their approach on a real testbed and the obtained measurements show that the proposed solutions allow to recover more than 95% of the packets faultily received by the coordinator.

6.2.3. Extending the network coverage through Relay nodes

By using relay nodes, it is possible to overcome the limitations of the star topology by realizing a two-hop network, while maintaining the LLDN superframe structure. In this perspective, the solution proposed in [85] extends the area covered by the LLDN network and allows sensor nodes, which are not in the communication range of the PAN coordinator, to still transmit their packets to the coordinator. The described approach is based on an opportunistic coding technique that enables forwarding of multiple packets in a single transmission. Basically, the relay node receives both the beacon from the coordinator and the data packet from the end device (that is too far to communicate directly with the PAN coordinator). At this point, the relay node encodes the two packets (i.e., it simply applies the XOR operator), waits for the timeslot assigned to it, and forwards the encoded frame. Both the PAN coordinator and the end device can decode the packet by XOR-ing it with their respective original information. This way, with just one timeslot, the relay node is able to serve both the ends of the communication. Although working, this approach increases the packet latency, and, above all, increases the probability of losing data packets, since two hops must be crossed instead of only one.

Another way to realize a two-hop network using relay nodes is described in [88]. The proposed solution, namely *MultiChannel-LLDN (MC-LLDN)*, divides the network in different sub-networks that operate at the same time on different channels. Basically, in MC-LLDN, each neighbor of the PAN coordinator can act as a sub-coordinator for another group of nodes. In practice, a sub-coordinator receives the beacon from the PAN coordinator, then switches to the channel associated to its sub-network and forwards the beacon. Since different channels are exploited, end devices of different sub-networks can transmit their data packets to their sub-coordinators during the same timeslots. In order to make data reach the PAN coordinator, sub-coordinators have to aggregate the payloads received from nodes and produce a unique packet to be sent to the PAN coordinator. The number of data that must be concatenated determine the size of the packet sent by sub-coordinators, and, hence, the timeslot and the superframe duration. Therefore, it is necessary to determine the optimal number of sub-coordinators in the network to achieve the shortest cycle time. To this end, the authors define an algorithm that calculates the optimal number given the number of nodes in the network. Simulation results show that the proposed solution not only allows to extend the network coverage, but also to use shorter superframe with respect to the star topology, thus permitting more frequent transmissions and shorter data generation periods. In fact, considering a large number of nodes (i.e., more than 20 in the presented experiments), the aggregation of data performed by sub-coordinators allows to avoid the PHY and MAC overheads and the IFSs that are present when each node sends its own packet directly to the PAN coordinator.

Table 4
LLDN: main research fields and contributions.

Analytical Model	[84]
Improving Reliability	[85–87]
Extending Network size	[85,88]
Support to node mobility	[44,89]

6.2.4. Support to node mobility

The authors of [44] investigate the impact of mobility on the performance of a LLDN network. In fact, when a node enters and exits the network, the frequent dissociations degrade the node connectivity and cause several disruptions to the network functionality. In this perspective, the authors have determined that many problems are due to the presence of three transmission states (Discovery, Configuration, Online). In fact, during the Online state, nodes can only send data and, hence, new nodes cannot associate. On the other hand, during Discovery and Configuration states, sensor readings cannot be transmitted, increasing data latency. The problem is much more severe since there is not an automatic way to switch from a state to another, but the network administrator is in charge of manually setting the duration of each phase. Finally, the adoption of a star topology makes the association phase much slower, since only the PAN coordinator can accept association requests. Under these considerations, the same authors, in [89], propose a *Mobile-Aware LLDN* scheme (MA-LLDN) that introduces the concept of proxy coordinator and some changes to the LLDN superframe/states. According to the MA-LLDN scheme, the superframe must contain some bidirectional timeslots. Proxy coordinators are nodes that use passive beacons (i.e., simply they insert some additional headers into their data packets) to advertise such timeslots. A joining node reads passive beacons to determine when bidirectional timeslots start, and use them to send its association request. The proxy coordinator will receive the request and forward it to the PAN coordinator. Similarly, the proxy will forward the PAN coordinator's answer to the joining node. This mechanism assures shorter association times, since a joining node has to wait for a passive beacon from any of the proxy coordinators. To further speed up the association process, the authors propose to remove the Discovery and Configuration states, by inserting two management slots (optional by default) in each Online superframe.

6.3. Summary & open issues

The IEEE 802.15.4e LLDN MAC protocol exploits a TDMA approach to provide communication reliability and low deterministic latency. To this end, it relies on a very simple superframe structure, regulated by beacon frames periodically transmitted by the PAN coordinator. Each LLDN device can obtain the exclusive access for a timeslot in the superframe, so as to send data to the PAN coordinator. This way, LLDN is particularly suitable for the factory automation context, where a (high) number of nodes communicate to a central sink. To guarantee low latency and ease of configuration, LLDN has been designed to work in star network topologies, using just one channel frequency. This aspect represents one major limitation of LLDN, since the extension of a LLDN network is forcedly limited, as well as the tolerance to bad channel conditions. For this reason, several solutions have been proposed in the literature, in order to improve coverage and reliability. Table 4 summarizes the main research contributions regarding LLDN.

However, LLDN presents a number of other limitations that have not been addressed yet. For instance, LLDN is not able to react to time-varying traffic or changes in the topology. Indeed, number of nodes and packet size must be known in advance, before the Online state. In addition, the standard does not explain how and when the network can come back to the Discovery or

the Configuration states in order to allow nodes to join/leave the network or reconfigure their slot allocation.

7. Conclusions

The IEEE has recently released the 802.15.4e amendment that introduces a number of enhancements/modifications to the MAC layer of the original 802.15.4 standard to overcome its limitations, i.e., low reliability, unbounded packet delays and no protection against interference/fading. The 802.15.4e standard document assumes that readers are quite familiar with the 802.15.4 technology and presents many references to the original standard resulting a not easy-to-follow document for an inexpert audience. In this paper, we provide a clear and structured overview of all the new 802.15.4e mechanisms. In particular, we describe the details of the main 802.15.4e MAC behavior modes (i.e., TSCH, DSME, and LLDN), highlighting their main features as well as possible application domains. Also, we provide a detailed survey of the current literature.

For each of these MAC protocols, we have pointed out that a number of research issues still exist. In general, since the standard is relatively recent, many works analyze their performance considering the same applications scenarios as the original 802.15.4. This means, for instance, that mesh networks have not been studied enough yet, even though TSCH and DSME have been specifically designed for such topologies. Also, although 802.15.4e is considered the base of the IoT stack, its integration with the upper layers (such as 6LowPAN, RPL and CoAP) has not been fully investigated and numerous issues are still unsolved. In addition, the 802.15.4e standard does not specify how some of the presented mechanisms must be implemented, leaving many aspects to designers, e.g., TSCH link scheduling and DSME-GTS allocation. Hence, much work must be done in order to have a complete implementation of all the 802.15.4e MAC modes. Finally, the three surveyed MAC protocols present severe security issues that must be overcome in order to use 802.15.4e networks in critical scenarios.

Acknowledgment

This work has been partially supported by the University of Pisa, in the framework of the PRA 2015 program.

References

[1] E. Borgia, The internet of things vision: key features, applications and open issues, Comput. Commun. 54 (1) (2014) 1–31.
[2] L. Atzori, A. Iera, G. Morabito, The internet of things: a survey, Comput. Netw. 54 (15) (2010) 2787–2805.
[3] L.A. Grieco, A. Rizzo, S. Colucci, S. Sicari, G. Piro, D. Di Paola, G. Boggia, IoT-aided robotics applications: technological implications, target domains and open issues, Comput. Commun. 54 (1) (2014) 32–47.
[4] M. Conti, S.K. Das, C. Bisdikian, M. Kumar, L.M. Ni, A. Passarella, G. Roussos, G. Tröster, G. Tsudik, F. Zambonelli, Looking ahead in pervasive computing: challenges and opportunities in the era of cyber–physical convergence, Pervasive Mobile Comput. 8 (1) (2012) 2–21.
[5] M. Conti, Computer communications: present status and future challenges, Comput. Commun. 37 (1) (2014) 1–4.
[6] A. Wichmann, B.D. Okkalioglu, T. Korkmaz, The integration of mobile (tele) robotics and wireless sensor networks: a survey, Comput. Commun. 51 (15) (2014) 21–35.
[7] M. Erol-Kantarci, H.T. Mouftah, Wireless multimedia sensor and actor networks for the next generation power grid, Ad Hoc Netw. 9 (4) (2011) 542–551.
[8] E. Fadel, V.C. Gungor, L. Nassef, N. Akkari, M.G. Abbas Malik, S. Almasri, I.F. Akyildiz, A survey on wireless sensor networks for smart grid, Comput. Commun. 71 (1) (2015) 22–33.
[9] G. Anastasi, M. Conti, M. Di Francesco, A. Passarella, Energy conservation in wireless sensor networks: a survey, Ad Hoc Netw. 7 (3) (2009) 537–568.
[10] R. Zurawski, Networked Embedded Systems, CRC press, Boca Raton, FL, USA, 2009.
[11] S. Chouikhi, I. El Korbi, Y. Ghamri-Doudane, L.A. Saidane, A survey on fault tolerance in small and large scale wireless sensor networks, Comput. Commun. 69 (15) (2015) 22–37.
[12] IEEE Computer Society, IEEE standard for information technology, Part 15.4, Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LRWPANs), 2006.
[13] ZigBee Alliance, The ZigBee Specification version 1.0, 2007.
[14] Wireless Medium Access Control (MAC) and Physical Layer (PHY), Specifications for personal area networks (WPANs), IEEE Stand. (2005).
[15] HART Communication Foundation Std, HART Field Communication Protocol Specification, 2007.
[16] ISA, International Society of Automation, Wireless Systems for Industrial Automation: Process Control and Related Applications Standard ISA-100.11a, 2009.
[17] M. Conti, C. Boldrini, S.S. Kanhere, E. Mingozzi, E. Pagani, P.M. Ruiz, M. Younis, From MANET to people-centric networking: milestones and open research challenges, Comput. Commun. 71 (1) (2015) 1–21.
[18] RFC 4944: Transmission of IPv6 Packets over IEEE 802.15.4 Networks [Online]. Available: http://tools.ietf.org/html/rfc4944.
[19] N. Accettura, L.A. Grieco, G. Boggia, P. Camarda, Performance analysis of the RPL routing protocol, in: Proceeding of IEEE International Conference on Mechatronics (ICM), 2011, pp. 767–772.
[20] E. Ancillotti, R. Bruno, M. Conti, The role of the RPL routing protocol for smart grid communications, IEEE Commun. Mag. 51 (1) (2013) 75–83.
[21] Z. Shelby, K. Hartke, C. Bormann, B. Frank. Constrained application protocol (CoAP), draft-ietf -core-coap-13. Available online: https://datatracker.ietf.org/doc/draft-ietf-core-coap/.
[22] R. Daidone, G. Dini, G. Anastasi, On evaluating the performance impact of the IEEE 802.15.4 security sub-layer, Comput. Commun. 47 (1) (2014) 65–76.
[23] P. Di Marco, C. Fischione, F. Santucci, K.H. Johansson, Modeling IEEE 802.15.4 networks over fading channels, IEEE Trans. Wireless Commun. 13 (10) (2014) 5366–5381.
[24] D. De Guglielmo, F. Restuccia, G. Anastasi, M. Conti, S. Das, Accurate and efficient modeling of 802.15.4 unslotted CSMA-CA through event chains computation, IEEE Trans. Mobile Comput. (2016).
[25] E.T. Yazdi, A. Willig, K. Pawlikowski, Frequency adaptation for interference mitigation in IEEE 802.15.4-based mobile body sensor networks, Comput. Commun. 53 (1) (2014) 102–119.
[26] S. Brienza, M. Roveri, D. De Guglielmo, G. Anastasi, Just-in-time adaptive algorithm for optimal parameter setting in 802.15.4 WSNs, ACM Trans. Auton. Adapt. Syst. 10 (4) (2016) 26 Article 27.
[27] S. Brienza, D. De Guglielmo, G. Anastasi, M. Conti, V. Neri, Strategies for optimal MAC parameter setting in IEEE 802.15.4 wireless sensor networks: A performance comparison, in: Proceedings of the 2013 IEEE Symposium on Computers and Communications (ISCC), Split, 2013.
[28] K. Yedavalli, B. Krishnamachari, Enhancement of the IEEE 802.15.4 MAC protocol for scalable data collection in dense sensor networks, in: Proceedings of the 6th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks and Workshops (WiOPT 2008), 2008, pp. 152–161.
[29] C. Kishore Singh, A. Kumar, P.M. Ameer, Performance evaluation of an IEEE 802.15.4 sensor network with a Star Topology, Wireless Netw. 14 (4) (2008) 543–568.
[30] S. Pollin, M. Ergen, S. Ergen, B. Bougard, L. Van der Perre, I. Moerman, A. Bahai, P. Varaiya, F. Catthoor, Performance analysis of slotted carrier sense IEEE 802.15.4 medium access layer, IEEE Trans. Wireless Commun. 7 (9) (2008) 3359–3371.
[31] G. Anastasi, M. Conti, M. Di Francesco, A comprehensive analysis of the MAC unreliability problem in IEEE 802.15.4 wireless sensor networks, IEEE Trans. Indus. Inf. 7 (1) (2011) 52–65.
[32] IEEE std. 802.15.4e, Part. 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendament 1: MAC sublayer, IEEE Computer Society, 2012.
[33] L.-W. Yeh, M.-S. Pan, Beacon scheduling for broadcast and convergecast in ZigBee wireless sensor networks, Comput. Commun. 38 (1) (2014) 1–12.
[34] M.-S. Pan, Y.-C. Tseng, Quick convergecast in ZigBee beacon-enabled tree-based wireless sensor networks, Comput. Commun. 31 (5) (25 2008) 999–1011.
[35] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, C. Tengfei, K. Pister, A realistic energy consumption model for TSCH networks, IEEE *Sensors J.* 14 (2) (2014) 482–489, doi:10.1109/JSEN.2013.2285411.
[36] T. Watteyne, J. Weiss, L. Doherty, J. Simon, Industrial IEEE802.15.4e networks: performance and trade-offs, in: Proceedings of *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 8–12, doi:10.1109/ICC.2015.7248388.
[37] T. Watteyne, L. Doherty, J. Simon, K. Pister, Technical overview of SmartMesh IP, in: *International Workshop on Extending Seamlessly to the Internet of Things (esIoT)*, Taiwan, 2013, pp. 3–5.
[38] S. Zats, R. Su, T. Watteyne, K. Pister, Scalability of time synchronized wireless sensor networking, in: Proceedings of *37th Annual Conference on IEEE Industrial Electronics Society (IECON 2011)*, 2011, pp. 7–10, doi:10.1109/IECON.2011.6119789.
[39] D. Stanislowski, X. Vilajosana, Q. Wang, T. Watteyne, K. Pister, Adaptive synchronization in IEEE802.15.4e networks, IEEE Trans. *Indus. Inf.* (2014), doi:10.1109/TII.2013.2255062.
[40] T. Chang, T. Watteyne, K. Pister, Q. Wang, Adaptive synchronization in multi-hop TSCH networks, Comput. Netw. (2015), doi:10.1016/j.comnet.2014.11.003.
[41] D. De Guglielmo, A. Seghetti, G. Anastasi, M. Conti, A performance analysis of the network formation process in IEEE 802.15.4e TSCH wireless sensor/actuator networks, in: Proceeding of *2014 IEEE Symposium on Computers and Communication (ISCC)*, 2014, pp. 23–26, doi:10.1109/ISCC.2014.6912607.
[42] E. Vogli, G. Ribezzo, L.A. Grieco, G. Boggia, Fast join and synchronization schema in the IEEE 802.15.4e MAC, in: Proceedings of *2015 IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*, 2015, pp. 9–12, doi:10.1109/WCNCW.2015.7122534.

[43] D. De Guglielmo, S. Brienza, G. Anastasi, A Model-based Beacon Scheduling algorithm for IEEE 802.15.4e TSCH networks, in: IEEE 17th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM), 2016, pp. 21–24.

[44] Y. Al-Nidawi, H. Yahya, A.H. Kemp, Impact of mobility on the IoT MAC infrastructure: IEEE 802.15.4e TSCH and LLDN platform, in: Proceedings of the 2nd IEEE World Forum on Internet of Things (WF-IoT), Milan, 2015, pp. 478–483.

[45] Y. Al-Nidawi, A.H. Kemp, Mobility aware framework for timeslotted channel hopping IEEE 802.15.4e sensor networks, IEEE *Sensors J.* 15 (12) (2015) 7112–7125, doi:10.1109/JSEN.2015.2472276.

[46] M. Barcelo, A. Correa, X. Vilajosana, J.L. Vicario, A. Morell, Novel routing approach for the TSCH mode of IEEE 802.15.14e in wireless sensor networks with mobile nodes, in: Proceedings of *2014 IEEE 80th Vehicular Technology Conference (VTC Fall)*, 2014, pp. 14–17, doi:10.1109/VTCFall.2014.6966074.

[47] B. Peng, A.H. Kemp, Energy-efficient geographic routing in the presence of localization errors, Comput. Netw (2010), doi:10.1016/j.comnet.2010.10.020.

[48] M. Zúñiga Zamalloa, K. Sead, B. Krishnamachari, A. Helmy, Efficient geographic routing over lossy links in wireless sensor networks, ACM Trans. Sensor Netw. (2008), doi:10.1145/1362542.1362543.

[49] T. Watteyne, A. Mehta, K. Pister, Reliability through frequency diversity: why channel hopping makes sense, in: *Proceedings of the 6th ACM symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks* (PE-WASUN '09), ACM, New York, NY, USA, 2009, pp. 116–123, doi:10.1145/1641876.1641898.

[50] D. Peng, G. Roussos, Adaptive channel hopping for wireless sensor networks, Proceeding of *2011 International Conference on Selected Topics in Mobile and Wireless Networking (iCOST)*, 2011, doi:10.1109/iCOST.2011.6085828.

[51] M.R. Palattella, N. Accettura, M. Dohler, L.A. Grieco, G. Boggia, Traffic Aware Scheduling Algorithm for reliable low-power multi-hop IEEE 802.15.4e networks, in: Proceedings of *IEEE 23rd International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, 2012, doi:10.1109/PIMRC.2012.6362805.

[52] M.R. Palattella, N. Accettura, L.A. Grieco, G. Boggia, M. Dohler, T. Engel, On optimal scheduling in duty-cycled industrial IoT applications using IEEE802.15.4e TSCH, IEEE *Sensors J.* 13 (10) (2013) 3655–3666, doi:10.1109/JSEN.2013.2266417.

[53] R. Soua, P. Minet, E. Livolant, MODESA: an optimized multichannel slot assignment for raw data convergecast in wireless sensor networks, Proceeding of *2012 IEEE 31st International Performance Computing and Communications Conference (IPCCC)*, 2012, doi:10.1109/PCCC.2012.6407742.

[54] R. Soua, E. Livolant, P. Minet, MUSIKA: a multichannel multi-sink data gathering algorithm in wireless sensor networks, Proceeding of *2013 9th International Wireless Communications and Mobile Computing Conference (IWCMC)*, 2013, doi:10.1109/IWCMC.2013.6583756.

[55] A. Tinka, T. Watteyne, Kris Pister, A decentralized scheduling algorithm for time synchronized channel hopping, Ad Hoc Networks, Springer, Berlin Heidelberg, 2010.

[56] A. Morell, X. Vilajosana, J.L. Vicario, T. Watteyne, Label switching over IEEE802.15.4e networks, Trans. Emerg. Telecommun. Technol. 24 (5) (2013) 458–475.

[57] R. Soua, P. Minet, E. Livolant, Wave: a distributed scheduling algorithm for convergecast in IEEE 802.15.4e TSCH networks, Trans. Emerg. Telecommun. Technol (2015), doi:10.1002/ett.2991.

[58] R. Soua, P. Minet, E. Livolant, DiSCA: a distributed scheduling for convergecast in multichannel wireless sensor networks, Proceeding of *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, 2015, doi:10.1109/INM.2015.7140288.

[59] N. Accettura, M.R. Palattella, G. Boggia, L.A. Grieco, M. Dohler, Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things, Proceeding of *2013 IEEE 14th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2013, doi:10.1109/WoWMoM.2013.6583485.

[60] O.D. Incel, L. van Hoesel, P. Jansen, P. Havinga, MC-LMAC: a multi-channel MAC protocol for wireless sensor networks, Ad Hoc Netw. (2011), doi:10.1016/j.adhoc.2010.05.003.

[61] N. Accettura, E. Vogli, M.R. Palattella, L.A. Grieco, G. Boggia, M. Dohler, Decentralized traffic aware scheduling in 6tisch networks: design and experimental evaluation, Internet. Things J. (2015) 455–470.

[62] D. Fanucchi, R. Knorr, B. Staehle, Impact of network monitoring in IEEE 802.15.4e-based wireless sensor networks, in: Proceedings of *2015 IEEE 16th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2015.

[63] S. Duquennoy, B. Al Nahas, O. Landsiedel, T. Watteyne, Orchestra: robust mesh networks through autonomously scheduled TSCH, in: Proceedings of the International Conference on Embedded Networked Sensor Systems (ACM SenSys 2015), Seoul, South Korea, 2015.

[64] N. Accettura, G. Piro, Optimal and secure protocols in the IETF 6TiSCH communication stack, in: Proceedings of *2014 IEEE 23rd International Symposium on Industrial Electronics (ISIE)*, 2014, doi:10.1109/ISIE.2014.6864831.

[65] P. Thubert, T. Watteyne, M.R. Palattella, X. Vilajosana, Q. Wang, IETF 6TSCH: combining IPv6 connectivity with industrial performance, in: Proceedings of *2013 Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS)*, 2013, doi:10.1109/IMIS.2013.96.

[66] M.R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L.A. Grieco, G. Boggia, M. Dohler, Standardized protocol stack for the internet of (important) things, IEEE *Commun. Surv. Tut.* 15 (3) (2013) 1389–1406, doi:10.1109/SURV.2012.111412.00158.

[67] D. Dujovne, T. Watteyne, X. Vilajosana, P. Thubert, 6TiSCH: deterministic IP-enabled industrial internet (of things), IEEE *Commun. Mag.* 52 (12) (2014) 36–41, doi:10.1109/MCOM.2014.6979984.

[68] M.R. Palattella, et al., 6TiSCH wireless industrial networks: determinism meets IPv6, Internet of Things, Springer International Publishing, 2014 2014.

[69] P. Thubert. An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4. Working Draft, IETF Secretariat, Internet-Draft draft-ietf-6tisch-architecture-09, 2015.

[70] Q. Wang, X. Vilajosana, and T. Watteyne, 6TiSCH Operation Sublayer (6top), IETF Std. draft-wang-6tisch 6top-sublayer-01 [work-in-progress], 2014.

[71] X. Vilajosana, K. Pister, Minimal 6TiSCH Configuration-draft-i-etf-6tisch-minimal-00, IETF: Fremont, CA, USA, 2013.

[72] P. Thubert, M.R. Palattella, T. Engel, 6TiSCH centralized scheduling: when SDN Meet IoT, in: Proceedings of IEEE Conference on Standards for Communications & Networking (CSCN'15), 2015.

[73] M.R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L.A. Grieco, T. Engel, On-the-fly bandwidth reservation for 6tisch wireless industrial networks, IEEE Sensors J. 16 (2) (2016) 550–560.

[74] W.C. Jeong, J. Lee, Performance evaluation of IEEE 802.15. 4e DSME MAC protocol for wireless sensor networks, in: Proceedings of the first IEEE Workshop on Enabling Technologies for Smartphone and Internet of Things (ETSIoT), Seoul, South Korea, 2012.

[75] J. Lee, W.C. Jeong, Performance analysis of IEEE 802.15.4e DSME MAC protocol under WLAN interference, in: Proceedings of the IEEE International Conference on ICT Convergence (ICTC), Jeju Island, South Korea, 2012.

[76] T. Paso, J. Haapola, J. Iinatti, Feasibility study of IEEE 802.15.4e DSME utilizing IR-UWB and S-Aloha, in: Proceedings of the 24th IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC), London, United Kingdom, 2013.

[77] G. Alderisi, G. Patti, O. Mirabella, L. Lo Bello, Simulative assessments of the IEEE 802.15.4e DSME and TSCH in realistic process automation scenarios, in: Proceedings of the 13th IEEE International Conference on Industrial Informatics (INDIN), Cambridge, United Kin g dom, 2015.

[78] S. Capone, R. Brama, F. Ricciato, G. Boggia, A. Malvasi, Modeling and simulation of energy efficient enhancements for IEEE 802.15.4e DSME, in: Proceedings of Wireless Telecommunications Symposium (WTS), Washington, DC, USA, 2014.

[79] X. Liu, X. Li, S. Su, Z. Fan, G. Wang, Enhanced fast association for 802.15.4e-2012 DSME MAC Protocol, in: Proceedings of the 2nd International Conference on Computer Science and Electronics Engineering (ICCSEE), Hangzhou, China, 2013.

[80] W. Lee, K. Hwang, Y.A. Jeon, and S.S. Choi. Distributed fast beacon scheduling for mesh networks. In Proceedings of 8th IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), Valencia, Spain, 2011.

[81] K. Hwang, S. Nam, Analysis and enhancement of IEEE 802.15.4e DSME Beacon Scheduling Model, J. Appl. Math. (2014), doi:10.1155/2014/934610.

[82] S. Nam, K. Hwang, in: Enhanced Beacon Scheduling of IEEE802.15.4e DSME (chapter 60 of Frontier and Innovation in Future Computing and Communications), Springer, Netherlands, 2014, pp. 495–503.

[83] M. Anwar, X. Yuanqing, IEEE 802.15.4e LLDN: superframe configuration for networked control systems, in: Proceedings of 33rd Chinese Control Conference (CCC), Nanjing, China, 2014.

[84] C. Ouanteur, D. Aïssani, L. Bouallouche-Medjkoune, M. Yazid, H. Castel-Taleb, Modeling and performance evaluation of the IEEE 802.15. 4e LLDN mechanism designed for industrial applications in WSNs, Wireless Netw. (2016) 1–16.

[85] A. Berger, M. Pichler, A. Springer, W. Haslmayr, Energy efficient and reliable wireless sensor networks – an extension to IEEE 802.15.4e, EURASIP J. Wireless Commun. Netw. (2014).

[86] H. Kapil, C.S.R. Murthy, Rainbow product ranking based relay placement and adaptive retransmission scheme for a reliable 802.15.4e LLDN, in: Proceedings of IEEE International Conference on Industrial Technology (ICIT), Seville, Spain, 2015.

[87] A. Berger, A. Entinger, A. Potsch, A. Springer, Improving IEEE 802.15.4e LLDN performance by relaying and extension of combinatorial testing, in: Proceedings of IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 2014.

[88] G. Patti, G. Alderisi, L. Lo Bello, Introducing multi-level communication in the IEEE 802.15.4e protocol: the MultiChannel-LLDN, in: Proceedings of IEEE Emerging Technology and Factory Automation (ETFA), Barcelona, Spain, 2014.

[89] Y. Al-Nidawi, H. Yahya, A.H. Kemp, Tackling mobility in low latency deterministic multihop IEEE 802.15.4e sensor network, IEEE Sensors J. 16 (5) (2016) 1412–1427.