Ultra-Low Power and Green TSCH-based WSNs with Proactive Reduction of Idle Listening

Stefano Scanzio, *Senior Member, IEEE*, Federico Quarta, Giacomo Paolini, *Member, IEEE*, Gabriele Formis, *Student Member, IEEE*, and Gianluca Cena, *Senior Member, IEEE*

***Abstract*—Wireless sensor networks are characterized by low power consumption because motes are typically battery-powered. Time slotted channel hopping (TSCH) relies on a fixed transmission schedule, which enables the receiver module of wireless motes to be switched off every time it is not needed. Unfortunately, in many practical contexts most of the reserved slots remain unused, which leads to appreciable energy waste. For periodic traffic, proactive reduction of idle listening (PRIL) techniques have been proven able to mitigate this problem.**

In this paper, PRIL multi-hop (PRIL-M) is introduced with the aim to improve existing PRIL techniques, by lowering energy waste further in large real-world mesh networks. PRIL-M is advantageous in all those contexts where ultra-low power consumption is more important than end-to-end latency. Applications that can benefit from PRIL-M include, e.g., environmental monitoring, where sensors are deployed over the target area and must operate for years without maintenance. A thorough simulation campaign showed that, in these scenarios, energy consumption of PRIL-M is 75% less than standard TSCH, while the average latency is about 20 times larger.

***Index Terms*—IEEE 802.15.4, time slotted channel hopping (TSCH), wireless sensor networks (WSN), Internet of Things (IoT), Ultra-low power, green networking, proactive reduction of idle listening (PRIL), PRIL-F, PRIL-M**

I. Introduction

Wireless communications, and especially wireless sensor (and actuator) networks (WSN/WSAN), are a key enabling technology for the Internet of Things (IoT). The contexts where they can be applied range from unmanned surveillance [1], natural disaster management [2], industrial networks [3], and smart and precision agriculture [4], to arrive to the large and constantly growing home automation sector [5].

Many wireless solutions are exploited nowadays in the IoT, the most notable being IEEE 802.11 (Wi-Fi), 5G, IEEE 802.15.4 (WSAN), long range wide area network (LoRaWAN), and Bluetooth. Every technology has its own characteristics and peculiarities. Those based on the IEEE 802.15.4 standard [6], which include the time slotted channel hopping (TSCH) and deterministic and synchronous multi-channel extension (DSME) MAC-layer profiles, were mainly conceived for minimizing energy consumption, so that the time between battery replacements may increase to several years. As a consequence, they are valuable in all those situations where maintenance costs must be as low as possible.

Several solutions exist that rely on TSCH concepts, the most popular being WirelessHART, ISA 100.11a, and 6TiSCH. DSME is instead a beacon-enabled protocol (that shares some similarities with WIA-PA) for which open-source implementations exist like openDSME. This work focuses on TSCH because of its wider availability in commercial devices. Compared to DSME [7], the main difference is the absence of a contention access period, which makes TSCH less suitable for handling sporadic communications. However, its characteristics make it very suitable for periodic traffic.

The solution we propose targets all those real application contexts where decreasing power consumption is by far more important than ensuring low end-to-end delays. Examples include *environmental monitoring*, which in industrial plants can be used to support product quality tracking in the brownfield (that is, without dedicated communication/power-supply infrastructures), e.g., by logging temperature and humidity of several shop-floor and warehouse zones. In these cases there are typically no specific constraints about latency, but sensors must be able to operate for years with no maintenance (*deploy-and-forget*). In particular, the following three aspects must be borne in mind: a) battery replacement may be problematic/uneconomical when devices are located in difficult-to-reach places; b) sometimes, the whole device need to be replaced when batteries are exhausted, e.g., radiator heat meters; and, c) disposing large amounts of batteries is likely to increase pollution, with a negative environmental impact.

The basic mechanism TSCH exploits to enable low-power communication is time slotting, which is a simplified version of time-division multiple access (TDMA). Time is divided into *slots* of fixed duration T_{slot} and all network traffic (globally defined in terms of a number of *links*, every one connecting one or more senders to one or more receivers) is scheduled to fit within these intervals. In this way, every node can switch into a deep-sleep state in all those slots in which no frames exchanges are scheduled for it, hence saving energy. The slot schedule repeats periodically over time (the period is in the order of one or two seconds), and every link is assigned a specific slot (plus a specific channel offset, but this is inessential for our work). Every time a slot is scheduled to a given link but no transmissions takes place, the recipient node experiences *idle listening*, i.e., it turns its receiving interface on unnecessarily, thus wasting energy since there is no frame

This work was partially supported by the European Union under the Italian National Recovery and Resilience Plan (NRRP) of NextGenerationEU, partnership on "Telecommunications of the Future" (PE00000001 - program "RESTART"). *(Corresponding author: Stefano Scanzio.)*

S. Scanzio and G. Cena are with the National Research Council of Italy (CNR–IEIIT), 20133 Milano, Italy (e-mail:stefano.scanzio@cnr.it).
F. Quarta is with the Politecnico di Torino, 10129 Torino, Italy.
G. Paolini is with University of Bologna, 40136 Bologna, Italy.
G. Formis is with the Politecnico di Torino, 10129 Torino, Italy, and also with the National Research Council of Italy (CNR–IEIIT), 20133 Milano, Italy.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3406646

to be read.

In some cases, the amount of energy spent for idle listening could be non-negligible. For instance, the charge drawn to receive a maximal-size 127 B frame (and send back the related acknowledgment (ACK) frame) in OpenMoteSTM devices is 651.0 μJ, while the charge wasted for idle listening is 303.3 μJ [8], which is about one half of what is needed for data reception. Since WSNs are often used for cyclically sensing physical quantities characterized by slow dynamics (like temperatures), sampling periods can be quite long (depending on the application, they can range from less than one minute to a few hours). Consequently, the amount of energy wasted by idle listening may be quite high.

Whenever the effective usage of scheduled cells can be predicted with good accuracy, proactive reduction of idle listing (PRIL) can be employed to switch off receivers in those slots where no frames are expected to be exchanged, hence enabling ultra-low power communication. If traffic is periodic with a known period, the PRIL-F technique has been defined [9] that only considers the first hop in the path between the source node of a flow and its destination. It can be used to save a sizable amount of energy in single-hop networks (star topology), as well as in those setups where part of the intermediate relays are mains-powered. The major limitation of PRIL-F is that only a subset of the nodes benefits from a reduced energy consumption. As a matter of fact, real WSNs are typically multi-hop, and because of their tree topology (customarily enforced by routing protocols), nodes closer to the root consume more energy. Unfortunately, PRIL-F tends to have little effect on them.

In this paper a new technique is proposed, we named PRIL-M, which specifically targets multi-hop networks. It has been conceived to work together with PRIL-F, overcoming its limitations. The two techniques have been compared and their joint use (PRIL-F is applied to the first hop and PRIL-M to the following hops) analyzed by means of an extensive simulation campaign, to thoroughly investigate the benefits they can provide in realistic environmental monitoring applications. Results show that the combined use of PRIL techniques yields significant energy savings at the expense of latency, which suits the requirements of the considered scenario.

The next section briefly introduces TSCH and the PRIL-F technique, while the new PRIL-M technique is presented in Section III. The experimental setup is described in Section IV, followed by the presentation of results in Section V. Conclusions are finally drawn in Section VI.

II. Proactive Reduction of Idle Listening

A survey is provided below about the operating principles of TSCH and PRIL-F, on which PRIL-M grounds.

A. TSCH basics

A scheduled TSCH slot (i.e., assigned to a link) is named *cell*. *Shared* cells can be used by a plurality of transmitting nodes, whereas every *dedicated* cell is allocated to a single transmitter (there is no contention). The latter are by far the most common option in real setups, and are typically used

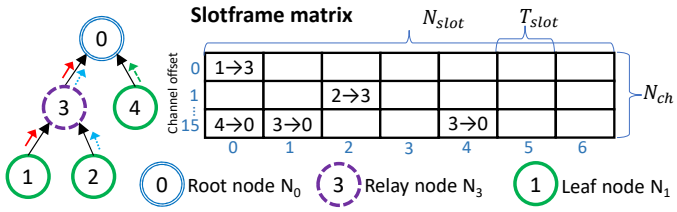

Fig. 1. Example of TSCH operation and slotframe matrix.

for unicast frame transmission (in a given direction) between a specific pair of nodes, e.g., for propagating data sensed from the real world toward the intended sink (root node). The TSCH schedule is defined by a *slotframe* matrix with size $N_{\text{ch}} \times N_{\text{slot}}$, an example of which is depicted in Fig. 1. The schedule spans in two orthogonal directions: time (*time slotting*) and frequency (*channel hopping*).

Regarding time, the slotframe matrix repeats identically with period $T_{\text{sfr}} = N_{\text{slot}} \cdot T_{\text{slot}}$, and defines which communications are allowed in any specific slot (and between which nodes). The notion of time in TSCH is uniquely identified by a counter shared among all the nodes of the network, known as the absolute slot number (ASN), which is initialized to zero and is incremented by one on every slot. Nodes are time synchronized by a specific clock synchronization protocol [10], precise enough to ensure that boundaries between slots are adequately obeyed by every node. Typical configuration parameters of the slotframe matrix in real implementations are $N_{\text{slot}} = 101$ and $T_{\text{slot}} = 20\,\text{ms}$, which implies $T_{\text{sfr}} = 2.02\,\text{s}$.

Regarding frequency, up to N_{ch} transmissions can be carried out concurrently (between different pairs of nodes) in every slot (identified by the ASN). In particular, for operations in the 2.4 GHz band with offset quadrature phase shift keying (O-QPSK) PHY modulation, the IEEE 802.15.4 standard defines 16 channels ($N_{\text{ch}} = 16$). Let c be the logical channel (or channel offset) of a specific scheduled cell in the slotframe matrix, which coincides with its row number. The physical channel ch on which transmission is performed is derived from c through the formula $\text{ch} = H[(\text{ASN} + c) \mod N_{\text{ch}}]$, where $H[\,]$ is the hopping sequence (typically implemented as a vector of N_{ch} elements) that performs such mapping. If N_{slot} and N_{ch} are selected as coprime numbers, in case of errors subsequent retransmissions of the same packet will be performed on different channels, which makes link reliability less susceptible to the behavior of single channels [11].

As thoroughly analyzed in [12], proper selection of network parameters (e.g., N_{slot}, T_{slot}, and retry limit) permits to set a trade-off among latency, reliability, and power consumption. PRIL techniques constitute an additional method, orthogonal to configuration, to decrease energy consumption.

B. Conserving energy in WSNs

Consuming as little energy as possible is one of the main goals of WSNs, and TSCH is no exception. Whenever no scheduled receptions or transmissions are foreseen, a TSCH node can be safely switched into a deep power-saving state,

in which energy consumption is negligible. This aspect makes this solution very attractive.

A number of research activities are available in the literature that specifically focus on power consumption. Some of them act on specific aspects of the protocol. The active-scan procedure proposed in [13] permits to decrease the network formation time, while in [14] the same goal is obtained using Q-learning to reduce the congestion of messages used by the network formation algorithm. Balancing nodes' load is another way to reduce battery discharge [15]. In [12] it is shown that reducing energy consumption is possible by tuning communication parameters. Other contributions focus on improving synchronization accuracy among TSCH nodes: in [16] a guard beacon strategy is proposed to decrease the guard time, in [17] the guard time is selected by each node based on its distance from the root, and in [18] a better synchronization accuracy is obtained by modifying the sending period of the enhanced beacon (which, besides other purposes, delivers information about time).

Other techniques optimize instead the routing protocol for low-power and lossy networks (RPL), which is typically used to generate the network topology (routes) in TSCH-based networks. In particular, in [19] RMA-RP is introduced to make RPL more energy efficient in case of node mobility, in [20] mobility is managed by improving the way RPL selects the new parent (e.g., by using more effective metrics), and in [21] nodes adjust the sending frequency of control messages based on the mobility level of neighbor nodes.

Generally speaking, all the approaches aimed at increasing communication quality (latency and reliability) by decreasing the average number of retransmissions have a positive impact on power consumption as well. This is the case of black and white listing techniques, which are meant to dynamically select the best channel on which transmissions are performed: in [22] bad channels are blacklisted based on the packet delivery ratio, [23] introduces the concept of probabilistic blacklisting, where the probability that a channel is not skipped depends on its perceived communication quality, in [24] cells are scheduled based on network metrics including the packet delivery ratio and end-to-end latency, and finally in [25] blacklisting is managed as a dynamic multi-armed Bernoulli bandit process to adapt selected channels to time-varying interference.

Historically, a large part of the techniques aimed at reducing power consumption in this kind of WSNs relies on scheduling algorithms [26], by optimally assigning transmission frequencies and slots of the slotframe matrix according to some specific objective function that minimizes, for instance, power consumption. In particular, in [27] the VAM-HSA algorithm was proposed to solve an energy-efficient maximization problem by reducing its complexity, in [28] the PRCOS algorithm was proposed to obtain a scheduling aimed at maximizing system lifetime by taking into account power consumption, in [29] a MAC layer scheduling algorithm named TREE was proposed to dynamically optimize allocated cells, instead in [30] the scheduling algorithm optimizes the use of shared cells. A suitable scheduling not only permits to decrease power consumption of the network, but also to improve its real-time capabilities: in [31] this was achieved with the combination of spectral clustering unsupervised learning and earliest deadline first (EDF) scheduling algorithm, while in [32] similar goals were obtained with an orchestrator (SDN-TSCH) that acts in a centralized fashion.

Concerning techniques based on proactive reduction of idle listening, PRIL-F, was preliminary introduced in [33] and completely defined in [9], whereas PRIL-M is first presented in this work. Both PRIL-F and PRIL-M are completely independent from above solutions, hence their joint adoption is possible to reduce energy waste further.

C. PRIL over the first hop (PRIL-F)

The basic idea behind PRIL to prevent idle listening (or reduce it as much as possible) is to switch off receivers also in their scheduled cells when no frames are expected. Several options exist to determine the occurrence of such conditions. A quite sophisticated approach is that the nodes analyze traffic at runtime and automatically determine the actual usage pattern of their scheduled cells, e.g., by using machine learning algorithms. This information is then exploited to disable those cells that, with high likelihood, would remain unused, hence incurring in idle listening. Simpler techniques exist as well that foresee that the transmitter on a link directly controls the receiver based on locally available information, by turning it off for a given number of subsequent scheduled cells so that energy can be saved on the receiving side.

In this work, we analyze only techniques of the second type, where specific *sleep* commands are added to the frames exchanged on a link to explicitly instruct recipients to go to sleep. We mostly focused on those IoT applications in which sensors belonging to the perception layer (either leaf or intermediate relay nodes in the tree network topology) generate cyclic data that are sent in the upward direction to a sink (root node). This is the usual case in almost all real WSNs, therefore it does not constitute a real limitation. However, the techniques we introduce also work in the reverse (downward) direction. An efficient way to deliver sleep commands to the receiver is by means of *information elements* (IE), which are specific user-defined elements that can be included in the header of IEEE 802.15.4 frames. Let T^i_{app} be the generation period of node N_i. Intuitively, the longer the generation period, the more energy can be saved by PRIL techniques, because the receiving side of the link can be turned off for longer periods of time.

The first proposal for reducing idle listening is PRIL-F, which was conceived to lower the energy waste of receiving nodes located one hop away from the traffic source. For example, by looking at the topology of Fig. 1, when the source node is N_1 it can disable N_3 temporarily so as to make it conserve energy. In the case of traffic flowing in the upward direction from N_1 to N_0 passing through relay N_3 ($\mathrm{N}_1 \rightarrow \mathrm{N}_3 \rightarrow \mathrm{N}_0$), PRIL-F is only effective on N_3 and can do nothing for N_0. In fact, N_3 has no clue in advance of when packets will arrive from N_1 that it must forward to N_0, and hence it is seemingly unable to generate effective sleep commands for the latter.

Let notation $\mathrm{N}_i \xrightarrow{x} \mathrm{N}_j$ represent the scheduled cell x of the link $\mathrm{N}_i \rightarrow \mathrm{N}_j$ when N_j leaves its receiving interface on (this

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3406646

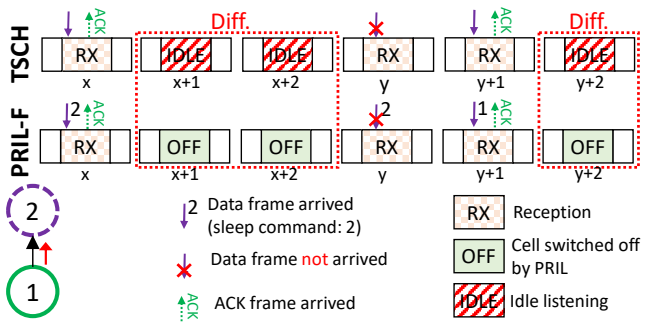

Fig. 2. Example of PRIL-F operation compared with default TSCH.

corresponds to the default TSCH behavior). When a single cell of the matrix is allocated to the link, the x-th cell is found in slotframe x. If a frame is transmitted in the cell, it is received by N_j (possibly corrupted by errors); otherwise, if no transmissions took place, N_j experiences idle listening. The case when reception in cell x has been (temporarily) disabled on N_j by PRIL is instead represented by the notation $\mathrm{N}_i \xrightarrow{\overline{x}} \mathrm{N}_j$.

An example that compares TSCH and PRIL-F operations is shown in the timing diagram of Fig. 2, where a packet is enqueued in the sender node N_1 every three slotframes. Let us assume that the initial transmission attempt is performed for a certain frame $\mathcal{F}$ in cell $\mathrm{N}_1 \xrightarrow{x} \mathrm{N}_2$ (this event is denoted $\mathcal{F}_x$). Following the successful exchange, the link state of standard TSCH in the immediate future (before the next packet is generated) can be described as $\mathrm{N}_1 \xrightarrow{x+1,x+2} \mathrm{N}_2$, and hence cells $x+1$ and $x+2$ suffer from idle listening.

If, as depicted in the leftmost part of the figure, the frame reaches the destination node N_2 immediately (without any retry), as notified by the reception of the related ACK frame, the following two scheduled cells for the link (i.e., $x+1$ and $x+2$) could be safely disabled to save energy on it. As said before, PRIL-F does so by making the sender include an IE in the frame that tells the receiver the number of scheduled cells for which it must sleep. This can be specified either in absolute terms (ASN of the scheduled cell when the link must be re-enabled) or in relative terms (number of scheduled cells for which the receiver node must stay disabled). These two solutions are equivalent from the functional point of view; however, we opted for the latter because it simplifies descriptions and the information to be stored in the IE is smaller. Referring again to Fig. 2, if N_1 knows that the packet generation of the application executing on it is periodic with period $T_{\mathrm{app}} = 3$ slotframes, it can add a specific *sleep* command with value 2. Frame transmission in this case is symbolically represented by notation $\mathcal{F}_x^{.s=2}$, where "$.s = 2$" denotes the sleep IE. If correctly received by N_2, the following two scheduled cells of the link will be disabled, which can be described as $\mathrm{N}_1 \xrightarrow{\overline{x+1,x+2}} \mathrm{N}_2$.

The rightmost part of the figure describes instead what happens when transmission errors occur that corrupt the frame. In particular, frame $\mathcal{F}_y^{.s=2}$ is sent in slot y but fails to reach the destination. Thus, it is retransmitted in slot $y+1$, after decreasing the value in the sleep command by one, since this is a retry (this action is synthetically described as $\mathcal{F}_{y+1}^{.s=1}$). In this case, only cell $\mathrm{N}_1 \xrightarrow{\overline{y+2}} \mathrm{N}_2$ will be disabled, and the amount of energy saved on N_2 is smaller.

If a frame with a sleep IE correctly arrives to destination but the ACK is lost, the receiver turns the relevant cells off but the sender is unaware of this. Hence, all further retries of that frame will no longer reach the destination, until the sleep command ends and the link is re-enabled on N_2.

III. PRIL over multiple hops (PRIL-M)

The PRIL multi-hop (PRIL-M) technique proposed in this work is meant to overcome the main limitation of PRIL-F, i.e., that it can save energy only on nodes located one hop away from the traffic source. The technique we present is characterized by modest complexity, which enables direct and effective implementations on real WSN devices (typically characterized by low computational power).

In Fig. 3 a sample network setup is shown where three periodic traffic flows from leaf nodes to the root are defined, $\tau_1 = \langle \mathrm{N}_1 \rightarrow \mathrm{N}_4 \rightarrow \mathrm{N}_0, T_1 \rangle$, $\tau_2 = \langle \mathrm{N}_2 \rightarrow \mathrm{N}_4 \rightarrow \mathrm{N}_0, T_2 \rangle$, and $\tau_3 = \langle \mathrm{N}_3 \rightarrow \mathrm{N}_4 \rightarrow \mathrm{N}_0, T_3 \rangle$, where T_1, T_2, T_3 represent the respective generation periods. When applied to this example, PRIL-F only affects N_4, whereas PRIL-M impacts on all the following nodes in the path to the destination, i.e., those with a distance from the source greater than one hop. Consequently, the joint use of PRIL-F and PRIL-M mitigates the idle listening problem for all the nodes in the WSN that have receiving cells scheduled for unicast transmissions.

Referring to the above example, PRIL-M counteracts idle listening in N_0 by disabling a subset $\mathcal{S}$ of the cells scheduled to the link $\mathrm{N}_4 \rightarrow \mathrm{N}_0$ (this action is denoted $\mathrm{N}_4 \xrightarrow{\overline{\mathcal{S}}} \mathrm{N}_0$). The way subset $\mathcal{S}$ is determined is more complex than PRIL-F: in fact, a link controlled by PRIL-M is typically traversed by many packet flows, including those not directly generated by the node that handles transmissions on the link (for them, it just acts as a relay) but by nodes located at lower levels in the tree topology (or upper, for downward flows). In other words, not all frames sent on $\mathrm{N}_4 \rightarrow \mathrm{N}_0$ are generated by N_4. In these conditions, correctly predicting whether or not a specific scheduled cell will be left unused is hardly possible for relay nodes like N_4. The main problem is due to retransmissions of frames (when they are not acknowledged), which generate unpredictability in the arrival time (and order) of packets

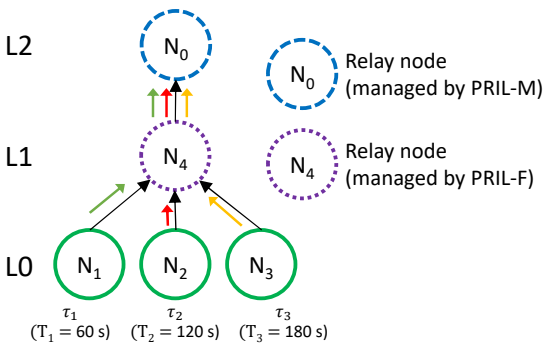

Fig. 3. Simple network topology to explain PRIL-M operations.

belonging to distinct flows. In large mesh networks, where paths from sources to sinks may consist of many hops, traffic flows get mixed in increasingly random way the closer links are to the root node.

The main idea behind PRIL-M is to have the transmitting side of a link turning off the receiving side (by using sleep commands) for a time equal to the minimum among the periods of the flows traversing the link itself. In fact, the flow with the smallest period is typically the one with the tightest constraints in terms of delivery latency (soft deadlines, in our case). To propagate the knowledge about the period of any flow τ_ℓ to all the following nodes in the related path, the originating node (source) attaches a *timing* IE to its frames that specifies its generation period. Unlike sleep IEs, which act on a single link, timing IEs attached to the frame are forwarded unchanged by relay nodes between subsequent hops. For instance, in the example of Fig. 3, the frames sent by N_1 (hence belonging to flow τ_1) are encoded as $\mathcal{F}^{.T=T_1}$, where the timing IE denoted "$.T = T_1$" is added by N_1 and propagates period T_1 along the entire path $\mathrm{N}_1 \rightarrow \mathrm{N}_4 \rightarrow \mathrm{N}_0$.

To operate correctly, every relay that executes PRIL-M (e.g., N_4) must know the minimum period among all the flows crossing its outgoing link. This information is typically acquired in the *learning* phase, and enables PRIL-M to save energy when the node enters the subsequent *runtime* phase. A separate instance of PRIL-M is executed on every relay node for every outgoing link.

A. Learning phase

At network startup (or upon network topology changes), when a relay receives a frame $\mathcal{F}^{.T=T_\ell}$ that contains a timing IE with the period T_ℓ of the flow τ_ℓ it belongs to, it starts a learning phase whose duration equals T_ℓ (or a small multiple of it). In the learning phase the node collects the periods of all the frames it forwards (grouped by outgoing link) and computes the minimum among such values, denoted $T_{\min}$. The node that generates the related "fastest" flow is identified as $\mathrm{N}_{\mathrm{ref}}$. If several flows exist for a given outgoing link whose periods coincide with $T_{\min}$, which are generated by distinct upstream nodes, one of them is selected as $\mathrm{N}_{\mathrm{ref}}$.

After the learning phase is finished, every time a frame $\mathcal{F}^{.T=T_\ell}$ is received for which $T_\ell < T_{\min}$, both $T_{\min}$ and $\mathrm{N}_{\mathrm{ref}}$ are updated consequently. Instead, if the relay does not receive any frames belonging to the flow that originates from the current $\mathrm{N}_{\mathrm{ref}}$ for a given time period $T_{\max}$ (the timeout can be set, e.g., to $T_{\max} = 10 \cdot T_{\min}$), it switches back to the learning phase, looking for another flow with minimum period. In case of topology changes, in TSCH-based networks the new routing is typically decided by the RPL protocol. This operation can take several minutes [34], during which the node is set to behave according to legacy TSCH. When routing is re-established, the node enters the learning phase again, with the aim of re-activating PRIL-M as soon as possible. Optimizations can be brought to the above procedure, to lower both the duration of the learning phase and its energy consumption, but they have not been reported here since the benefits we obtained were mostly inessential. During the learning phase, the node obeys standard TSCH rules.

B. Runtime phase

In this phase both $T_{\min}$ and $\mathrm{N}_{\mathrm{ref}}$ are correctly initialized. Let $\mathrm{N}_i \rightarrow \mathrm{N}_j$ be a link operated according to PRIL-M that is traversed by frames of flow τ_ℓ to reach their destination. With reference to Fig. 3, N_i corresponds to N_4 and N_j corresponds to N_0. Node N_i understands that PRIL-M must be used instead of PRIL-F because it is not the originator of the flow, i.e., it is not the first node of the related path. As in PRIL-F, a sleep command $\mathcal{F}^{.s=T_{\mathrm{end}}}$ permits to deactivate the link for a given amount of time, where T_{end} represents the number of cells related to the link $\mathrm{N}_i \rightarrow \mathrm{N}_j$ where reception will be disabled. Again, we assume that a relative value is included in the sleep command, but the same reasoning applies to absolute values.

The basic idea is to deactivate the link for a time period equal to $T_{\min}$, so as to prioritize periodic traffic with the fastest generation rate. When this approach is implemented in practice, various aspects must be taken into account, related to the fact that frames can be lost and retransmitted up to a maximum number of times. In particular, the link state seen by N_i and N_j may differ due to losses. For this reason, a reference implementation for PRIL-M is provided by means of two state machines, we denote RX and TX, reported in Fig. 4, along with detailed explanations. Every arc is labeled with a conventional event-action pair. The upper part (event) reports the conditions that must be verified to enable the transition to the new state. The leftmost condition (in **bold**) consists in one of four possible events: **sleep** (reception of a sleep command), **ack** (correct reception of an ACK frame), **no_ack** (missing or corrupted ACK frame), and **slot_end** (end of a slot related to a cell scheduled to link $\mathrm{N}_i \rightarrow \mathrm{N}_j$). Instead, the lower part (action) lists the sequence of operations that are executed on state change, which are mostly aimed at updating the value of the variables pertaining to the given state machine instance.

1) Receiving side: The RX state machine of receiver N_j is depicted in Fig. 4.a. Node N_j defines a separate instance of this state machine for every link, i.e., for every sender that communicates with it. In the ON state the link is active and obeys standard TSCH rules. When a frame $\mathcal{F}^{.s=T_{\mathrm{end}}}$ with a sleep command is received (**sleep** event), the local counter `sleep_end` is initialized to T_{end} and the state machine switches to the OFF state, where the link is deactivated so that N_j starts saving energy. In this state, counter `sleep_end` is decreased by one on every new slot scheduled to the link $\mathrm{N}_i \rightarrow \mathrm{N}_j$. At the end of every such slots (**slot_end** event) the counter `sleep_end` is checked. When it reaches zero the RX state machine switches back to the ON state, and the link returns to standard TSCH operation.

2) Transmitting side: The TX state machine for the sender N_i is shown in Fig. 4.b. As can be seen, it is more complex than RX. When a frame $\mathcal{F}^{.T=T_{\min}}$ sent by $\mathrm{N}_{\mathrm{ref}}$ is received by N_i, counter `sleep_end` is initialized with the number of cells scheduled to the link $\mathrm{N}_i \rightarrow \mathrm{N}_j$ that fit in an interval of duration $T_{\min}$. For example, if $T_{\min} = 2\,\mathrm{min}$, $T_{\mathrm{sfr}} = 2.02\,\mathrm{s}$, and one cell is reserved per slotframe, `sleep_end` is set to 59. The value of `sleep_end` is then decremented by one on every slot scheduled to the link. It is worth pointing out that, at the arrival of $\mathcal{F}^{.T=T_{\min}}$, the transmission buffer of N_i may

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3406646

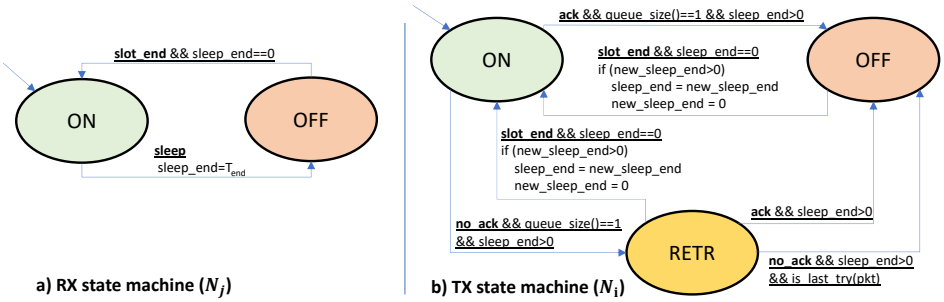

Fig. 4. State machine of the receiver side (N_j, on the left) and sender side (N_i, on the right) of an intermediate link in a path.

already contain queued packets. Moreover, additional packets may arrive after $\mathcal{F}^{\cdot T=T_{\min}}$ is relayed, which will be enqueued. According to TSCH rules, all packets in the outgoing queue of link $\mathrm{N}_i \rightarrow \mathrm{N}_j$ (on N_i) will be orderly sent (one attempt per scheduled cell). When transmitting the last of them (that is, when `queue_size()==1`), N_i adds a sleep command to the frame, that is, it sends $\mathcal{F}^{\cdot s=T_{\text{end}}}$ with T_{end} equal to the residual value of `sleep_end` at that time.

If transmission of $\mathcal{F}^{\cdot s=T_{\text{end}}}$ succeeds at the first attempt, the TX state machine switches directly to the OFF state; otherwise, it moves to the RETR state, whose purpose is managing retransmissions of frames that embed a sleep command. In the RETR state, N_i can not determine if N_j is in the ON or OFF state, because it does not know whether the data frame or the ACK frame went lost. Consequently, it keeps on retrying. If transmission eventually succeeds, i.e., an ACK frame is received by N_i (**ack** event), N_i switches to the OFF state and is aware that also N_j is in the OFF state. Instead, if the retry limit (as specified by N_{tries}) is exceeded (**no_ack** event and `is_last_try(pkt)` returns true), N_i still moves to the OFF state, but it does not actually know whether N_j is in the ON or OFF state. Doing so prevents the loss of subsequent frames from N_i when N_j managed to switch to the OFF state.

When N_i is in the OFF state, it waits until `sleep_end` reaches 0 to return to the ON state. Less likely, but possible, is the situation where N_i returns to the ON state from the RETR state. This happens when, during the transmission of the last packet in the queue, the value of `sleep_end` is close to 0 and, due to retransmissions, it reaches 0 before N_i exceeds the retry limit. The actions performed for the two arcs from OFF to ON and from RETR to ON make use of the variable `new_sleep_end` to deal with the arrival of a new packet $\mathcal{F}^{\cdot T=T_{\min}}$ at N_i when it is still in the OFF or RETR states. This packet may arrive earlier than expected because, for instance, the previous packet $\mathcal{F}^{\cdot T=T_{\min}}$ was late due to retransmissions or excessive queuing delays in the preceding hops of the path (upstream nodes). Since in the OFF and RETR states the counter `sleep_end` is already in use by the above algorithm (and hence, it cannot be overwritten), a secondary counter `new_sleep_end` is needed to keep track of the future reactivation time of the link itself. Both counters are contextually decreased by one on every scheduled cell of the link $\mathrm{N}_i \rightarrow \mathrm{N}_j$. When the node returns to the ON state, if `new_sleep_end` is greater than zero (i.e., it is active), it is copied in `sleep_end`.

3) Implementation complexity: PRIL-M requires the addition of two IEs to the frame, namely the sleep command $\mathcal{F}^{\cdot s}$ and the flow generation period $\mathcal{F}^{\cdot T}$. In addition, the RX and TX state machines of Fig. 4 have to be implemented: they are quite simple, and every instance includes just 9 variables, counting both those used to store the link state and those for managing the learning phase. A separate instance of PRIL-M must be created for every outgoing link. Summing up, the variables to store the current state of the RX and TX state machines require 1 bit and 2 bits, respectively. If 2 bytes are allocated for each one of the remaining seven variables (encoded as unsigned short integers), every instance takes less than 15 bytes, which is a modest amount of memory also for motes. It is worth pointing out that only one copy of the executable code implementing PRIL-M must be stored in memory, regardless of the number of instances.

Concerning memory footprint it must be noted that, because of queuing, larger transmission buffers are seemingly needed. However, since transmissions are regulated by the flow with the shortest period, no more than one packet belonging to the other flows could be in theory enqueued at any given time, which is the same as TSCH (in other words, buffer size is the same but its effective usage in PRIL-M is noticeably higher).

Regarding the size of the code segment, it is indicatively less than 2 kbytes, but the exact amount depends on many aspects, including the CPU type (x86 or ARM) and the operating system in use. It is worth pointing out that popular devices implementing TSCH, such as OpenMote B, are characterized by 256 kbytes of flash memory and 32 kbytes of RAM, which is more than enough for implementing PRIL-M. Finally, computational complexity for relay operations on a link is $\mathcal{O}(1)$, and only marginally affects CPU resource usage.

C. Example

In Fig. 5, three examples are provided to make the details of PRIL-M operations clearer. In all cases, two frames, denoted ${}^1\mathcal{F}$ and ${}^2\mathcal{F}$, are found in the queue of the sender node (TX) immediately before scheduled cell x. We set $N_{\text{tries}} = 2$ (only one retry is allowed). Moreover, ${}^1\mathcal{F}$ belongs to the flow with the shortest generation period ($T_{\min} = 5$ slotframes). In all the cases reported in the figure, the state of the TX node is described in the upper diagram, while the lower diagram refers to the RX node. Moreover, let us assume that the first frame (${}^1\mathcal{F}$) is successfully delivered at time x.

In *CASE a*, frame ${}^{2}\mathcal{F}_{x+1}^{s=3}$ (which instructs the receiver to sleep for 3 scheduled cells) fails to reach the destination in scheduled cell $x+1$ and is retransmitted in slot $x+2$ with a sleep command decreased to 2, i.e., ${}^{2}\mathcal{F}_{x+2}^{s=2}$, which correctly deactivates the RX node preventing idle listening in two slots. Conversely, in *CASE b* also the second transmission of frame ${}^{2}\mathcal{F}_{x+2}^{s=2}$ in slot $x+2$ fails, and since the maximum number of allowed transmissions attempts is reached ($N_{\text{tries}}=2$) the frame is discarded and removed from the queue. This means that the RX node does not reach the OFF state, consequently behavior remains the same as standard TSCH and idle listening is experienced. Finally, in *CASE c*, frame ${}^{2}\mathcal{F}_{x+1}^{s=2}$ sent in slot $x+1$ correctly arrives to destination but the corresponding ACK is lost. The RX node enters the OFF state and starts saving energy, but the TX node, which is not aware that the frame has arrived, continues retransmissions until the retry limit is reached, hence wasting some energy.

Having the state on two sides of the link always coherent is impossible, since both data and ACK frames may be lost. Nonetheless, it must be remarked that the state machines were conceived to never allow the transmitter node to be in the ON state when the receiver node is OFF. This intended behavior ensures that frames cannot be lost due to PRIL-M.

D. Overall behavior

Above descriptions refer to the link between a pair of nodes, managed on its two sides by the TX and RX state machines. To understand the overall network behavior it must be remembered that the path followed by the packets of every flow is made up of the ordered sequence of traversed links, where the state machine in every intermediate relay independently selects the upstream node N_{ref} with the fastest generation rate (whose period is denoted $T_{\min}$). On the whole, this procedure can be seen as a network-wide single-elimination tournament, where the fastest flows autonomously selected by sibling relays (local minima) compete in their common ancestor, in a process that stretches from every leaf up to the root (this holds for both the upward and downward directions).

This has two consequences: first, the values T_{end} selected by the different relays (to be included in their respective sleep commands) are usually not the same. Second, only one flow

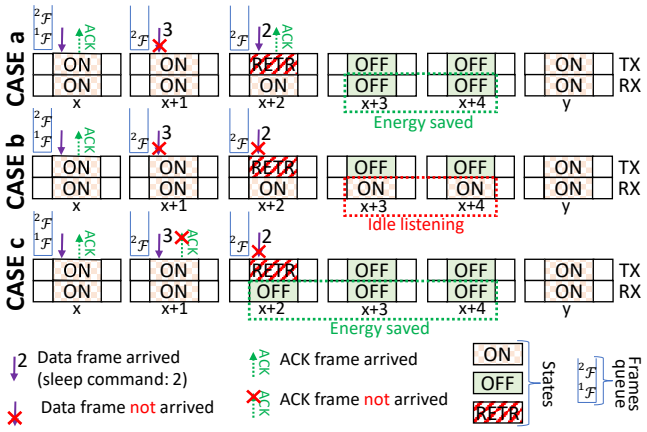

Fig. 5. Example of PRIL-M operation in three different cases.

(the one for which T_{app} is the global minimum) is granted that its packets never suffer from additional delays due to the outgoing links found disabled. For all the other flows, packets receive a privileged treatment only in the lower part of their path (close to the leaves), while they may suffer from additional delays in the upper part (near to the root). In particular, when arriving to a relay where a different N_{ref} has been selected, forwarding pace is dictated by the packets of that flow, causing random delays uniformly distributed in the range $[0, T_{\min}]$. In this way, packet forwarding occurs in bursts, which inherit the treatment of the winner flow. This means that no further delays will be experienced by them until a relay is encountered with a different N_{ref}. This situation has been explicitly shown for the deep topology on the right side of Fig. 6, by highlighting in green the portion of the paths where the winner remains the same (and no relevant queuing delay is experienced due to PRIL-M), whereas discontinuities (that unavoidably lead to queuing delays) are colored in red.

IV. Experimental Setup

Both PRIL-F and PRIL-M were compared with standard TSCH using a discrete event simulator we purposely developed, named TSCH-predictor. It differs from other publicly-available tools like 6TiSCH simulator [35] and TSCH-Sim [36] in terms of the drastically lower implementation complexity. This simulation framework permits new TSCH-based techniques, aimed at improving network behavior, to be easily and quickly implemented, assessed, and fully understood. In fact, only those features strictly needed to model basic operations of the specific techniques and to obtain raw indications about their performance have to be actually implemented. Results produced by our simulator were checked against measurements obtained from a real network setup based on OpenMote B devices running OpenWSN, and the absolute error concerning the average latency was found to be negligible (less than 12 ms), this denoting accurate and faithful modeling. This simulator has already been satisfactorily employed in former scientific works [9].

Network parameters used for simulation were taken from the default configuration of real devices, for example, slot duration ($T_{\text{slot}} = 20\,\text{ms}$), number of slots in a slotframe ($N_{\text{slot}} = 101$), maximum allowed number of transmission attempts ($N_{\text{tries}} = 16$). Instead, the failure probability for attempts related to data frames ($\epsilon_{\text{data}} = 12.6\%$) and ACK frames ($\epsilon_{\text{ACK}} = 8.0\%$) was derived from experimental data obtained from a real setup[1].

Regarding the power consumption model, the energy required to transmit one data frame including 127 B (the data size used in this work) and to receive the related ACK was set to $485.7\,\mu$J, the energy spent to receive every one of such frames and to return the related ACK was set to $651.0\,\mu$J, while the energy wasted for idle listening in a single slot was set to $303.3\,\mu$J. These values were obtained from [8], and refer to an STM32F103RB 32-bit microcontroller and an Atmel

[1]Experimental data used to compute the loss probability values are included in the file `default-101-16-15days.dat`, whose details can be found in https://dx.doi.org/10.21227/fg62-bp39, while the method for computing the two probabilities is described in [9].

AT86RF231 radiochip mounted on an Open-MoteSTM device. Other energy models are available in the scientific literature, which were obtained for different devices, but their adoption does not substantially change the obtained results in terms of the reduction of power consumption. The duration of simulated experiments was set to one year for every campaign.

V. Results

Experimental campaigns were performed for the three network configurations depicted in Fig 6.

A. Simple network configuration

The *simple* topology (Fig 6.a), already analyzed in Section III, is used to compare PRIL-F alone and PRIL-M (always used in conjunction with PRIL-F), highlighting some peculiarities. This configuration is characterized by three flows directed from leaves to the root N_0, whose generation periods were set equal to 3001, 6003, and 9005 slots, corresponding to about 60 s, 120 s, and 180 s, respectively. We selected coprime numbers to models quartz tolerance in real devices. By doing so, offsets between the generation times of frames belonging to different traffic flows keep drifting slowly over the entire duration of experiments (one year), making simulation results more reliable.

Table I reports the power consumption of every single node, as well as of all nodes in the network (row labeled "All"). In particular, P stands for the total power consumption (transmissions and receptions) while P_{listen} only considers contributions of receive modules (RX). As expected, PRIL-F practically zeroes the P_{listen} component of the intermediate node N_4, but it has no effect on N_0 since is located two hops away from the source. The reason why P_{listen} is not exactly zero depends on the learning phase, during which PRIL does not provide any benefits. When PRIL-M is additionally activated, energy is saved also on N_0. The total power consumption P (all nodes included) is $663.90\,\mu$W for standard TSCH, $239.22\,\mu$W with PRIL-F (36.0% of TSCH), and $108.46\,\mu$W with PRIL-M (16.3% of TSCH). Hence, the latter approach doubled the energy saving capabilities of PRIL-F.

The same configuration is then analyzed in terms of the end-to-end latency, by taking into account the following performance indices: mean value (μ_d), standard deviation (σ_d), 99− to 99.99−percentiles (d_{p99}, $d_{p99.9}$, and $d_{p99.99}$), and maximum ($d_{\max}$). Results, reported in Table II, highlight one of the most severe drawbacks of PRIL-M: unlike PRIL-F, which always brings beneficial effects (power consumption improves while latency remains unchanged), PRIL-M significantly increases delays, and consequently it should not be used in time-sensitive applications. However, it is important to remark that, in many application contexts, energy consumption is the first and foremost performance index for WSNs.

Latency increase is more evident for flows other than the one with the minimum generation period $T_{\min}$, as their packets may remain stuck for a while in intermediate nodes when the related outgoing links are temporarily disabled. As can be seen, the mean latency μ_d experienced by τ_2 and τ_3 is about 30 s. In fact, since the link returns to the ON state every ~ 60 s (i.e., $T_{\min}$) due to packets from N_1, and the generation times on N_2 and N_3 are not synchronized with N_1, the time they have to wait in the transmission queue before the link switches to ON is, on average, $\frac{T_{\min}}{2}$.

TABLE I
Comparison among TSCH and the proposed PRIL strategies (simple topology): power consumption.

a) Simple topology (2 layers)

Node	Hops	TSCH		PRIL-F		PRIL-M	
		P_{listen}	P	P_{listen}	P	P_{listen}	P
		[μW]		[μW]		[μW]	
N_0	2	138.64	163.34	138.62	163.36	0.19	23.83
N_4	1	438.92	482.09	.0017	41.20	.0017	50.11
N_3	0	0	3.36	0	6.34	0	6.25
N_2	0	0	5.04	0	9.46	0	9.42
N_1	0	0	10.07	0	18.85	0	18.87
All		**577.56**	**663.90**	**138.63**	**239.22**	**0.20**	**108.46**

TABLE II
Comparison among TSCH and the proposed PRIL strategies (simple topology): end-to-end latency.

a) Simple topology (2 layers)

	Flow (Source)	μ_d	σ_d	d_{p99}	$d_{p99.9}$	$d_{p99.99}$	$d_{\max}$
		[s]					
TSCH	All	1.720	1.389	6.220	9.400	12.000	17.260
PRIL-F	All	1.722	1.383	6.240	9.400	12.120	17.580
PRIL-M	$\tau_3(\mathrm{N}_3)$	30.229	16.879	60.340	63.820	67.200	69.940
	$\tau_2(\mathrm{N}_2)$	30.446	16.930	60.200	62.640	66.080	68.600
	$\tau_1(\mathrm{N}_1)$	4.282	2.337	11.160	14.620	17.860	21.660
	All	16.134	17.365	59.000	62.280	65.280	69.940

Overall power consumption (for the whole network) and mean latency (experienced by packets of source nodes N_1, N_2, and N_3) are graphically sketched in the two diagrams of Fig. 7, which permit to quickly compare the performance of TSCH, PRIL-F, and PRIL-M.

To set a trade-off between power consumption and latency in PRIL-M, an upper limit for the time a cell can remain disabled could be possibly enforced. Because this aspect is outside the scope of this paper, it is left as future work.

B. Realistic network configurations

Two larger and more realistic network configurations were additionally analyzed, we term *star* (Fig. 6.b) and *deep* (Fig. 6.c) topologies, whose complexity resembles real WSNs. Again, flows are directed from leaves to the root node N_0.

The star topology consists of 3 layers (root excluded) and interconnects 16 leaves (sensors), 6 intermediate nodes (relay only) at a distance of one hop from leaves and 3 nodes at two hops, plus the root N_0 (sink), for a total of 26 nodes. Two different periods were defined for flows, namely, ~ 10 min and ~ 30 min. Again, actual periods were defined in slot times as coprime numbers. Mapping between leaves and periods is shown in the figure. This topology represents a typical example of a WSN in which nodes are deployed in such a way to provide complete coverage of a certain area and the sink (gateway) is positioned near the center of this area.

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3406646

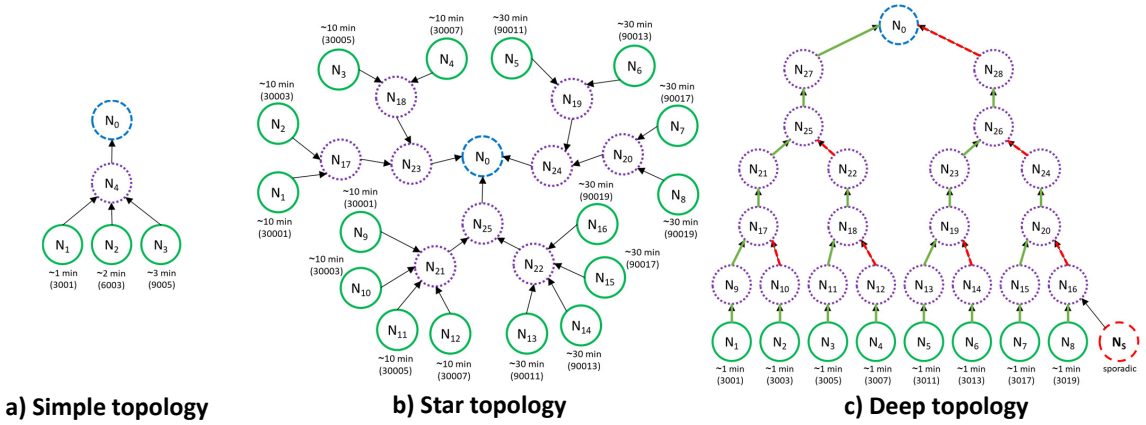

Fig. 6. Simple, star (3 layers), and deep (6 layers) network configurations used for simulations: the generation period (in slot times) is reported between parentheses.

The deep topology of Fig. 6.c is also quite common in the real world, especially when the area to be covered has an elongated or irregular shape. In this case, the maximum number of hops from leaves to the sink may be sensibly higher than the star topology, if networks with the same number of nodes are considered. In the example we considered, which consists of 6 layers, the maximum number of children for every node is two and the total number of nodes is 29. The generation period was set equal to about 1 min for all the 8 leaf nodes (actual periods of flows are similar but coprime).

Only for the experiment described in Section V-C, an additional node $\mathrm{N_S}$ was also inserted in the deep network topology to analyze the behavior of PRIL-M in the presence of sporadic traffic flows.

Table III about power consumption shows both aggregate results (distinguishing among leaves at the lowest network layer and the two layers of relays located one and two hops away from leaves, respectively) and results related to some specific individual nodes belonging to the different layers ($\mathrm{N_0}$, $\mathrm{N_{23}}$, $\mathrm{N_{21}}$, $\mathrm{N_{17}}$, and $\mathrm{N_1}$). Regarding the entire network (row in the table labeled "All"), the effect of PRIL-M is clear: compared to PRIL-F the total power consumption decreases to 46.4% (from 2140.2 μW to 993.71 μW) for the star topology and to 34.3% (from 3941.5 μW to 1350.2 μW) for the deep topology. When using PRIL-M, the contribution P_{listen} to power consumption (which is related to the idle listening phenomenon) is practically canceled. This is not true for PRIL-F: in this case, when the distance of the considered node from the source of the packet flow (number of hops) is greater than one, P_{listen} is the same as standard TSCH.

TABLE III
COMPARISON AMONG TSCH AND THE PROPOSED PRIL STRATEGIES (STAR AND DEEP TOPOLOGIES): POWER CONSUMPTION.

b) Star topology (3 layers)

Node	Hops	TSCH P_{listen} [μW]	TSCH P [μW]	PRIL-F P_{listen} [μW]	PRIL-F P [μW]	PRIL-M P_{listen} [μW]	PRIL-M P [μW]
$\mathrm{N_0}$	3	375.15	536.77	375.15	536.76	.13	159.07
$\mathrm{N_{23}}$	2	275.18	369.32	275.18	369.28	.13	101.01
$\mathrm{N_{21}}$	1	575.48	669.60	.0012	89.83	.0012	98.60
$\mathrm{N_{17}}$	1	287.73	334.82	.00058	44.93	.00058	53.66
$\mathrm{N_1}$	0	0	10.07	0	18.90	0	18.84
Relays	2	825.57	1107.8	825.57	1107.8	.19	299.71
Relays	1	2327.1	2609.3	.0069	269.36	.0069	308.79
Leaves	0	0	120.63	0	226.2	0	226.14
All		**3527.8**	**4374.6**	**1200.7**	**2140.2**	**0.33**	**993.71**

c) Deep topology (6 layers)

Node	Hops	TSCH P_{listen} [μW]	TSCH P [μW]	PRIL-F P_{listen} [μW]	PRIL-F P [μW]	PRIL-M P_{listen} [μW]	PRIL-M P [μW]
$\mathrm{N_0}$	6	250.17	357.77	250.16	357.78	.53	105.97
$\mathrm{N_{27}}$	5	125.04	219.16	125.05	219.15	.064	101.87
$\mathrm{N_{25}}$	4	275.19	369.30	275.18	369.29	0.78	101.52
$\mathrm{N_{21}}$	3	137.59	184.66	137.59	184.66	.12	55.04
$\mathrm{N_{17}}$	2	287.73	334.82	287.73	334.82	.98	54.73
$\mathrm{N_9}$	1	143.87	167.42	.00029	22.48	.00029	31.18
$\mathrm{N_1}$	0	0	10.06	0	18.87	0	18.79
All		**3903.3**	**5030.7**	**2752.3**	**3941.5**	**7.11**	**1350.2**

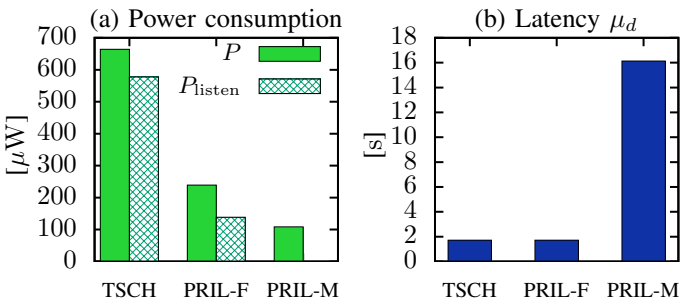

Fig. 7. Comparison among the proposed PRIL strategies (simple topology).

Regarding latency, Table IV shows that PRIL-M adoption makes performance worse than conventional TSCH. As expected, there is a consistent increase of the average latency μ_d and, more in general, of all the statistical indicators related to latency. Besides statistics evaluated over the entire set of flows, the table also reports results related to the two flows characterized by the minimum and maximum average latency, labeled "Min" and "Max", respectively. As can be seen, performance indicators about latency experienced large variations. In Fig. 8 the very same results are depicted graphically as diagrams.

One of the problems of PRIL-M is that the packet with the minimum period originated by $\mathrm{N_{ref}}$ deactivates the receiving

This article has been accepted for publication in IEEE Internet of Things Journal. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2024.3406646

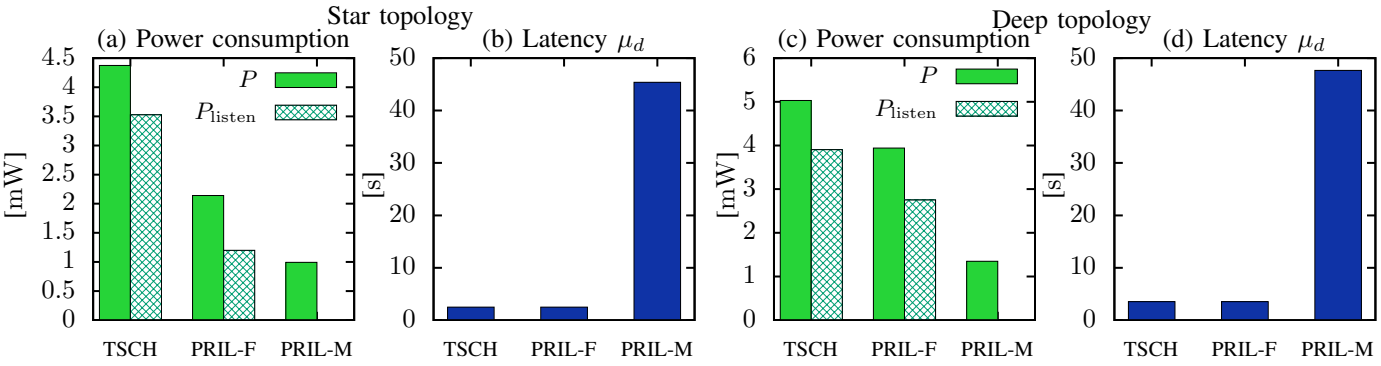

Fig. 8. Comparison among the proposed PRIL strategies (star and deep topologies): responsiveness is traded for power saving in PRIL-M.

TABLE IV
COMPARISON AMONG TSCH AND THE PROPOSED PRIL STRATEGIES (STAR AND DEEP TOPOLOGIES): END-TO-END LATENCY.

b) Star topology (3 layers)

	Flow	μ_d	σ_d	d_{p99}	$d_{p99.9}$	$d_{p99.99}$	$d_{\max}$
		[s]					
TSCH	All	**2.503**	1.872	8.760	12.660	16.960	32.320
PRIL-F	All	**2.503**	1.872	8.780	12.680	17.120	32.900
PRIL-M	Min	12.493	3.278	21.340	25.380	29.020	41.080
	Max	114.79	50.05	215.88	232.10	240.12	246.50
	All	**45.392**	34.271	158.22	205.80	227.72	246.50

c) Deep topology (6 layers)

	Flow	μ_d	σ_d	d_{p99}	$d_{p99.9}$	$d_{p99.99}$	$d_{\max}$
		[s]					
TSCH	All	**3.543**	2.352	10.800	14.500	18.240	28.140
PRIL-F	All	**3.544**	2.354	10.840	14.540	18.320	27.400
PRIL-M	Min	24.102	4.559	35.740	40.500	45.040	55.020
	Max	70.374	24.551	121.66	130.12	136.10	147.08
	All	**47.658**	23.847	112.96	125.18	132.40	147.08

Sporadic traffic

TSCH	N_S	3.925	2.663	12.340	16.160	20.500	21.820
PRIL-M	N_S	98.992	29.575	164.40	176.86	184.16	184.78

node (OFF state) even if the children nodes of the transmitting node still have packets to be sent. This phenomenon causes an increase in the latency on the other traffic flows, but also on the subsequent transmissions of packets related to the flow associated with N_{ref}. This issue poses new research challenges for improving PRIL-M.

Above set of experiments confirms that PRIL-M lowers energy consumption at the expense of latency and determinism. This implies that it can be profitably applied to real-world applications for which short latency is irrelevant, e.g., for monitoring purposes (periodic logging of physical quantities where timestamping is carried out by sensors). Conversely, PRIL-F is a valid option in all those contexts with real-time requirements, although energy saving is suboptimal.

C. Traffic characteristics

Additional experiments were finally carried out to analyze how the characteristics of the traffic impact on performance. In all cases the deep topology in Fig 6.c was considered.

In the first set of experiments the generation period T_{app} of leaf nodes was varied from 20 s to 5 minutes with a step of 20 s (always using coprime periods). The topmost plot in Fig. 9 shows the power consumption versus T_{app}. As expected, PRIL-M exhibits tangibly lower energy consumption than both PRIL-F and TSCH. When the generation period is shortened, the behavior of the three approaches becomes similar. In fact, the fraction of cells allocated to the link that are actually exploited for frame transmissions progressively increases, consequently decreasing the effects of idle listening. At the limit, when no cells remain unused (which means that the link operates in saturation conditions), idle listening cannot take place, irrespective of PRIL usage. On the other hand, when the period grows larger the power consumption of every approach converges to a specific limit value, and differences among them become more or less constant. Reasoning again at the limit, when T_{app} is enlarged consistently the majority of the allocated cells remain unused, which implies that the most part of energy is wasted by idle listening. Hence, PRIL ability to lessen this phenomenon (especially the multi-hop version) plays a crucial role in determining power consumption.

Concerning the average latency (lower part of Fig. 9), in PRIL-M it increases linearly with the generation period T_{app}. This is an expected behavior: in fact, the time for which the outgoing link in every intermediate relay node remains OFF is equal to the minimum generation period $T_{\min}$ it observes among the incoming flows. In this experiment, all periods T_{app} were set (about) equal, therefore $T_{\min}$ (and the sleep time T_{end}) are approximately the same for the different relays.

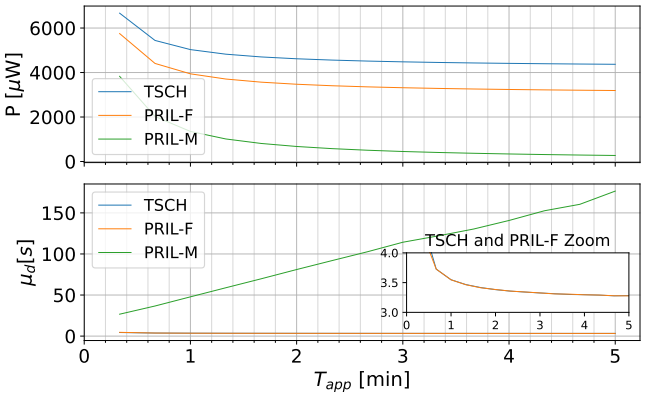

Fig. 9. Effect of the variation of the generation periods (deep topology).

As explained in Section III-D, for any given relay the frames of flows other than the one originating from $\mathrm{N_{ref}}$ may find the link OFF. In this case, they will wait for the arrival of a packet from $\mathrm{N_{ref}}$, after which all the queued frames will be orderly sent as a batch. When frame arrivals are not aligned with forwarding opportunities (this can be determined by looking at the network topology, see red arrows in Fig. 6.c), the forwarding delay introduced by every relay node is, on average, $T_{\min}/2$. As a consequence, the network-wide mean latency is proportional to T_{app}.

As far as PRIL-F and TSCH are concerned (see the zoomed in portion included in Fig. 9), they have the same latency, which decreases as the period increases and tends asymptotically to a limit value because packet queuing phenomena within relays become progressively less likely. Conversely, when the period gets shorter, the probability for a packet to arrive to a relay when other packets are also enqueued grows higher. This causes additional queuing delays, which explains the reason why, when T_{app} is small, mean latency increases up to several times T_{sfr} (remaining nevertheless below PRIL-M).

A second set of experiments was performed to assess the effect of PRIL-M on best effort sporadic traffic, whose results are reported in the lower part of Table III. A new node $\mathrm{N_S}$ (shown with a dashed red circle) was added to this purpose as a leaf to the network in Fig 6.c, and it was programmed to generate sporadic traffic where packet intertimes are selected randomly according to an exponential distribution with mean value $1\,\mathrm{hour}$. As expected, packets generated by $\mathrm{N_S}$ suffer in PRIL-M from the largest latency increase with respect to TSCH, because they are delayed more in every hop of their path (the PRIL-M state machines running in relay nodes privilege the flow with the minimum period, which never coincides with the sporadic flow).

VI. Conclusions

Time-slotted networks like TSCH may benefit from the adoption of the PRIL-F and PRIL-M techniques, which exploit traffic periodicity to reduce power consumption dramatically (periodic traffic patterns are commonplace in WSNs deployed at the perception layer in the IoT pyramid). In particular, in all the network configurations we analyzed in this work, the joint use of PRIL-M and PRIL-F decreased power consumption to less than 25% compared to standard TSCH, and to less than 50% compared to PRIL-F alone.

This remarkable achievement permits PRIL-M to be profitably adopted whenever IoT solutions based on WSNs have to be set up according to the deploy-and-forget paradigm. In these cases, reducing energy consumption to the bare minimum is a prerequisite to reduce maintenance activities involved in battery replacement, hence decreasing the total cost of ownership. Another non-negligible advantage of PRIL-M is that, it reduces the overall amount of exhausted batteries that need to be disposed, which makes these solutions greener.

The only disadvantage of PRIL-M is a non-negligible increase of end-to-end latency, which suggests that it is unsuitable for applications with real-time constraints. In these cases, PRIL-F should be employed alone. For this reason, a consistent part of our future activities on the subject will be devoted to reducing the impact of PRIL-M on communication latency as much as possible, for instance by limiting the maximum time for which scheduled slots can remain in the OFF state. Other potential research activities include the extension of PRIL techniques to similar communication technologies for WSNs, such as DSME.

References

[1] I. Chukwuemeka Chimsom and M. K. Habib, "Design of a Two-Tier WSN-based IoT Surveillance System with Cloud Integration," in International Conference on Research and Education in Mechatronics (REM), May 2019, pp. 1–7.

[2] M. Erdelj, M. Król, and E. Natalizio, "Wireless Sensor Networks and Multi-UAV systems for natural disaster management," Computer Networks, vol. 124, pp. 72–86, 2017.

[3] S. Scanzio, L. Wisniewski, and P. Gaj, "Heterogeneous and dependable networks in industry – A survey," Computers in Industry, vol. 125, p. 103388, 2021.

[4] X. Yu, P. Wu, W. Han, and Z. Zhang, "A survey on wireless sensor network infrastructure for agriculture," Computer Standards & Interfaces, vol. 35, no. 1, pp. 59–64, 2013.

[5] J. M. Batalla, G. Mastorakis, C. X. Mavromoustakis, and J. Zurek, "On cohabitating networking technologies with common wireless access for home automation system purposes," IEEE Wireless Communications, vol. 23, no. 5, pp. 76–83, 2016.

[6] IEEE, "IEEE Standard for Low-Rate Wireless Networks," IEEE Std 802.15.4-2015 (Rev. of IEEE Std 802.15.4-2011), pp. 1–709, Apr 2016.

[7] I. A. Mantilla Gonzalez, F. Meyer, and V. Turau, "A Comprehensive Performance Comparison of IEEE 802.15.4 DSME and TSCH in a Realistic IoT Scenario for Industrial Applications," ACM Trans. Internet Things, vol. 4, no. 3, jun 2023. [Online]. Available: https://doi.org/10.1145/3595188

[8] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. J. Pister, "A Realistic Energy Consumption Model for TSCH Networks," IEEE Sensors Journal, vol. 14, no. 2, pp. 482–489, 2014.

[9] S. Scanzio, G. Cena, and A. Valenzano, "Enhanced Energy-Saving Mechanisms in TSCH Networks for the IIoT: The PRIL Approach," IEEE Transactions on Industrial Informatics, pp. 1–11, 2022.

[10] M. Mongelli and S. Scanzio, "A neural approach to synchronization in wireless networks with heterogeneous sources of noise," Ad Hoc Networks, vol. 49, pp. 1–16, 2016.

[11] G. Cena, S. Scanzio, M. G. Vakili, C. G. Demartini, and A. Valenzano, "Assessing the Effectiveness of Channel Hopping in IEEE 802.15.4 TSCH Networks," IEEE Open Journal of the Industrial Electronics Society, vol. 4, pp. 214–229, 2023.

[12] S. Scanzio, M. G. Vakili, G. Cena, C. G. Demartini, B. Montrucchio, A. Valenzano, and C. Zunino, "Wireless Sensor Networks and TSCH: A Compromise Between Reliability, Power Consumption, and Latency," IEEE Access, vol. 8, pp. 167 042–167 058, 2020.

[13] M. Mohamadi, B. Djamaa, and M. R. Senouci, "RAST: Rapid and energy-efficient network formation in TSCH-based Industrial Internet of Things," Computer Communications, vol. 183, pp. 1–18, 2022.

[14] D. Z. Fawwaz and S.-H. Chung, "Adaptive Trickle Timer for Efficient 6TiSCH Network Formation Using Q-Learning," IEEE Access, vol. 11, pp. 37 931–37 943, 2023.

[15] S. Afshari, M. Nassiri, and R. Mohammadi, "SA-RPL: a scheduling-aware forwarding mechanism in RPL/TSCH-operated networks," International Journal of Ad Hoc and Ubiquitous Computing, vol. 34, no. 1, pp. 35–44, 2020.

[16] M. A. Sordi, O. K. Rayel, G. L. Moritz, and J. L. Rebelatto, "Towards Improving TSCH Energy Efficiency: An Analytical Approach to a Practical Implementation," Sensors, vol. 20, no. 21, 2020.

[17] A. Mavromatis, G. Z. Papadopoulos, A. Elsts, N. Montavont, R. Piechocki, T. Tryfonas, G. Oikonomou, and X. Fafoutis, "Adaptive Guard Time for Energy-Efficient IEEE 802.15.4 TSCH Networks," in Wired/Wireless Internet Communications, M. Di Felice, E. Natalizio, R. Bruno, and A. Kassler, Eds. Cham: Springer International Publishing, 2019, pp. 15–26.

[18] J. Vera-Pérez, D. Todolí-Ferrandis, J. Silvestre-Blanes, and V. Sempere-Payá, "Bell-X, An Opportunistic Time Synchronization Mechanism for Scheduled Wireless Sensor Networks," Sensors, vol. 19, no. 19, 2019.

[19] H. Farag, P. Österberg, M. Gidlund, and S. Han, "RMA-RP: A Reliable Mobility-Aware Routing Protocol for Industrial IoT Networks," in IEEE Global Conference on Internet of Things (GCIoT 2019), 2019, pp. 1–6.
[20] S. Murali and A. Jamalipour, "Mobility-Aware Energy-Efficient Parent Selection Algorithm for Low Power and Lossy Networks," IEEE Internet of Things Journal, vol. 6, no. 2, pp. 2593–2601, 2019.
[21] Y. Shin and S. Seol, "Improvement of Power Consumption in RPL-based Networks for Mobility Environment," in International Conference on Electronics, Information, and Communication (ICEIC 2020), 2020, pp. 1–3.
[22] V. Kotsiou, G. Z. Papadopoulos, P. Chatzimisios, and F. Theoleyre, "LA-BeL: Link-Based Adaptive BLacklisting Technique for 6TiSCH Wireless Industrial Networks," in Proceedings of the 20th ACM International Conference on Modelling, Analysis and Simulation of Wireless and Mobile Systems, ser. MSWiM. New York, NY, USA: Association for Computing Machinery, 2017, pp. 25–33.
[23] G. Cena, S. Scanzio, and A. Valenzano, "Ultra-Low Power Wireless Sensor Networks Based on Time Slotted Channel Hopping with Probabilistic Blacklisting," Electronics, vol. 11, no. 3, p. 304, Jan 2022. [Online]. Available: http://dx.doi.org/10.3390/electronics11030304
[24] W. Jerbi, O. Cheickhrouhou, A. Guermazi, and H. Trabelsi, "MSU-TSCH: A Mobile scheduling updated algorithm for TSCH in the internet of things," IEEE Transactions on Industrial Informatics, pp. 1–8, 2022.
[25] N. T. Javan, M. Sabaei, and V. Hakami, "Adaptive Channel Hopping for IEEE 802.15.4 TSCH-Based Networks: A Dynamic Bernoulli Bandit Approach," IEEE Sensors Journal, vol. 21, no. 20, pp. 23 667–23 681, 2021.
[26] N. Accettura, E. Vogli, M. R. Palattella, L. A. Grieco, G. Boggia, and M. Dohler, "Decentralized Traffic Aware Scheduling in 6TiSCH Networks: Design and Experimental Evaluation," IEEE Internet of Things Journal, vol. 2, no. 6, pp. 455–470, 2015.
[27] M. Ojo, S. Giordano, G. Portaluri, D. Adami, and M. Pagano, "An energy efficient centralized scheduling scheme in TSCH networks," in IEEE International Conference on Communications Workshops (ICC), 2017, pp. 570–575.
[28] T. Matsui and H. Nishi, "Time slotted channel hopping scheduling based on the energy consumption of wireless sensor networks," in IEEE 15th International Workshop on Advanced Motion Control (AMC 2018), 2018, pp. 605–610.
[29] T. van der Lee, G. Exarchakos, and S. H. de Groot, "Swarm-Based Energy Efficient Scheduling for Wireless Sensor Networks," in IEEE Conference on Standards for Communications and Networking (CSCN), 2019, pp. 1–6.
[30] B. Özceylan, B. Ünlü, and B. Baykal, "An energy efficient optimum shared cell scheduling for TSCH networks," in IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob 2017), 2017, pp. 1–8.
[31] M. G. Gaitán, D. Dujovne, J. Zuñiga, A. Figueroa, and L. Almeida, "Multi-Gateway Designation for Real-Time TSCH Networks using Spectral Clustering and Centrality," IEEE Embedded Systems Letters, pp. 1–1, 2022.
[32] F. Veisi, J. Montavont, and F. Theoleyre, "Enabling Centralized Scheduling Using Software Defined Networking in Industrial Wireless Sensor Networks," IEEE Internet of Things Journal, pp. 1–1, 2023.
[33] S. Scanzio, G. Cena, A. Valenzano, and C. Zunino, "Energy Saving in TSCH Networks by Means of Proactive Reduction of Idle Listening," in Ad-Hoc, Mobile, and Wireless Networks, L. A. Grieco, G. Boggia, G. Piro, Y. Jararweh, and C. Campolo, Eds. Cham: Springer International Publishing, 2020, pp. 131–144.
[34] C. Vallati, S. Brienza, G. Anastasi, and S. K. Das, "Improving Network Formation in 6TiSCH Networks," IEEE Transactions on Mobile Computing, vol. 18, no. 1, pp. 98–110, 2019.
[35] E. Municio, G. Daneels, M. Vučinić, S. Latré, J. Famaey, Y. Tanaka, K. Brun, K. Muraoka, X. Vilajosana, and T. Watteyne, "Simulating 6TiSCH networks," Transactions on Emerging Telecommunications Technologies, vol. 30, no. 3, p. e3494, 2019.
[36] A. Elsts, "TSCH-Sim: Scaling Up Simulations of TSCH and 6TiSCH Networks," Sensors, vol. 20, no. 19, 2020.