28 Aug 2020

Analysis of Distributed and Autonomous Scheduling Functions for 6tisch Networks

Francesca Righetti

Carlo Vallati

Sajal K. Das
Missouri University of Science and Technology, sdas@mst.edu

Giuseppe Anastasi

Recommended Citation

F. Righetti et al., "Analysis of Distributed and Autonomous Scheduling Functions for 6tisch Networks," *IEEE Access*, vol. 8, pp. 158243-158262, Institute of Electrical and Electronics Engineers (IEEE), Aug 2020.
The definitive version is available at https://doi.org/10.1109/ACCESS.2020.3020153

Received August 6, 2020, accepted August 18, 2020, date of publication August 28, 2020, date of current version September 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3020153

Analysis of Distributed and Autonomous Scheduling Functions for 6TiSCH Networks

FRANCESCA RIGHETTI[1], CARLO VALLATI[1], SAJAL K. DAS[2], (Fellow, IEEE), AND GIUSEPPE ANASTASI[1]
[1]Department of Information Engineering, University of Pisa, 56100 Pisa, Italy
[2]Department of Computer Science, Missouri University of Science and Technology, Rolla, MO 65409, USA

Corresponding author: Francesca Righetti (francesca.righetti@ing.unipi.it)

This work was supported in part by the Italian Ministry of Education and Research (MIUR) through the framework of the CrossLab Project (Departments of Excellence).

ABSTRACT The 6TiSCH architecture is expected to play a significant role to enable the Internet of Things paradigm also in industrial environments, where reliability and timeliness are of paramount importance to support critical applications. Many research activities have focused on the Scheduling Function (SF) used for managing the allocation of communication resources in order to guarantee the application requirements. Two different approaches have mainly attracted the interest of researchers, namely *distributed* and *autonomous scheduling*. Although many different (both distributed and autonomous) SFs have been proposed and analyzed, a direct comparison of these two approaches is still missing. In this work, we compare some different SFs, using different behaviors in allocating resources, and investigate the pros and cons of using distributed or autonomous scheduling in four different scenarios, by means of both simulations and measurements in a real testbed. Based on our results, we also provide a number of guidelines to select the most appropriate SF, and its configuration parameters, depending on the specific use case.

INDEX TERMS Industrial Internet of Things, 6TiSCH architecture, scheduling function, simulation and measurements.

I. INTRODUCTION

The Industrial Internet of Things (IIoT) is expected to revolutionize the way industrial systems are designed. The interconnection of physical systems seamlessly integrated into information systems will enable new functions, such as real-time remote monitoring and control, predictive maintenance, big data analytics, etc. [1]. In this context, the adoption of wireless communication technologies is crucial to ensure a rapid and cheap deployment of IIoT systems.

Recently, the 6TiSCH (IPv6 over the TSCH mode of IEEE 802.15.4e) Working Group (WG) has been established by the Internet Engineering Task Force (IETF) in order to standardize an architecture for the IIoT. The WG aims at defining a protocol stack to integrate IoT devices into existing IPv6 networks, still ensuring reliable and timely communication, which are critical requirements in industrial applications. To this aim, 6TiSCH relies on the Time Slotted Channel Hopping (TSCH) mode of operation of the IEEE 802.15.4 standard [2]. TSCH provides time-bounded, guaranteed-bandwidth, and energy-efficient communication through *time-slotted access*, high network capacity through *multi-channel communication*, and robustness against interference and fading through *frequency hopping*.

In 6TiSCH, a Scheduling Function (SF) is used to allocate communication resources (i.e., TSCH cells) in order to guarantee the application requirements. Different SFs [3]–[15] have been proposed to cope with the needs of different use cases. They can be broadly classified according to the approach they take to allocate TSCH cells to nodes, namely, *centralized*, *distributed*, *autonomous*, and *hop-by-hop*. Among all, however, distributed and autonomous approaches have mostly attracted the interest of researchers. There are also *hybrid* solutions that combine the previous basic approaches (e.g., autonomous and distributed scheduling).

In distributed scheduling, TSCH cells are negotiated by neighboring nodes, using the 6top protocol (6P) [16], that allows to allocate and deallocate cells dynamically,

The associate editor coordinating the review of this manuscript and approving it for publication was Giovanni Pau.

depending on the time-varying traffic and network conditions. Obviously, 6P transactions introduce an overhead, in terms of control traffic injected into the network and additional latency experienced by data packets. Instead, in autonomous scheduling, TSCH cells are allocated autonomously by nodes, by using a hash function applied to node addresses, without any packet exchange between neighboring nodes and, consequently, with no negotiation overhead.

Many autonomous and distributed SFs have been proposed and evaluated in the literature. However, in almost all previous papers, SFs taking a similar approach are compared (e.g., distribute [17], or autonomous [13], [14], [18]). A thorough performance evaluation that compare SFs taking different approaches (e.g., autonomous, distributed and, possibly, hybrid), and investigate the pros and cons of the considered approaches in different application scenarios is still missing.

In this article, we consider three SFs for 6TiSCH that follow a different behavior for cell allocation. Specifically, we consider E-OTF [7], a completely distributed SF, ALICE [13] that leverages a link-based autonomous scheduling approach, and MSF [6] that combines both autonomous and distributed scheduling (MSF is currently under standardization by the 6TiSCH WG as the reference SF for IIoT applications). Both E-OTF and MSF are adaptive SFs, as they are able to adjust the number of allocated cells to time-varying traffic conditions, while ALICE leverages a static allocation policy and is, thus, unable to adapt to traffic changes. Therefore, in order to make the comparison fair, in our study we also considered the *Frame Pending* (FP) mechanism provided by the underling TSCH access protocol. This option, when enabled, allows the sender node to signal the receiver that it has more data to send by setting a bit in the TSCH frame.

The FP mechanism makes ALICE adaptive, as it can now react to possible changes in traffic by increasing the number of allocated cells (per slotframe), if necessary. Unlike other adaptive autonomous SFs proposed in the literature [14], which require the exchange of control messages (the cost to pay for adaptation), the FP bit does not introduce any additional control overhead, as it is piggybacked in the underlying TSCH frame. To the best of our knowledge, this is the first article considering the TSCH FP option for (autonomous) scheduling adaptation.

The goal of our performance comparison is to investigate the suitability of autonomous and distributed scheduling to different use cases. To this end, we consider four different scenarios, corresponding to representative real-world use cases, and evaluate the performance of the considered SFs, in terms of reliability, timeliness, and duty cycle. In our analysis, we use both simulations and experimental measurements in a real testbed.

Our results show that there is no SF that outperforms the other ones in all the considered scenarios. Instead, different SFs exhibit pros and cons under different conditions. Hence, we provide below a set of guidelines that can help in selecting the most appropriate SFs, depending on the specific use case and operating conditions.

In a nutshell, the main contributions provided by this article can be summarized as follows:

- A comprehensive evaluation of three popular SFs for IIoT applications in a number of representative scenarios;
- An evaluation of the influence of the TSCH FP option on the performance of the considered SF;
- The presentation of ALICE with FP option as an adaptive autonomous SF with no control overhead;
- A set of guidelines to select the most appropriate SF, depending on the specific use case and operating conditions.

To the best of our knowledge, this is the first work analyzing the performance of SFs with different general approaches, i.e. distributed, autonomous and hybrid, and also to assess the influence of the TSCH FP option in their performance. Also, this is one of the few works that consider different traffic models in addition to the popular upstream traffic, considered in the majority of the works available in the literature.

The reminder of this article is organized as follows. In Section II, we describe the 6TiSCH architecture. In Section III, we provide a general overview of SFs for 6TiSCH and the related work, with special focus on the three SFs considered in our analysis. In Section IV, we present our simulation methodology, while in Section V we discuss the simulation results. In Section VI we investigate the influence of the FP mechanism. In Section VII, we present our experimental measurements. In Section VIII, we summarize the lessons learned from our study and, finally, in Section IX we conclude the paper.

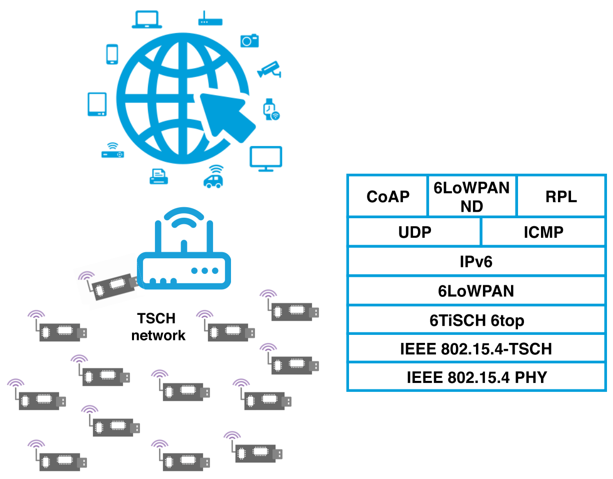

FIGURE 1. The 6TiSCH Architecture.

II. 6TiSCH ARCHITECTURE

The 6TiSCH architecture [19] aims at integrating wireless networks based on the 802.15.4 TSCH standard [2] into existing IPv6 infrastructures. The reference architecture and the complete protocol stack are shown in Figure 1.

At the MAC layer, the TSCH protocol allows wireless communication with bounded delay, high reliability, and low energy consumption. To this end, TSCH relies on *time-slotted channel access*, *multi-channel communication*, and *frequency hopping*. Time is divided into time intervals of fixed duration (timeslots), each of which allows the transmission of a packet and the corresponding acknowledgment. A number of consecutive timeslots form a *slotframe*, which repeats periodically over time. To increase the network capacity, different nodes are allowed to transmit simultaneously on the same timeslot, using a different channel (multi-channel communication). Specifically, 16 different channels are available, identified by a *channel offset* (an integer value in the range 0-15) and, hence, each cell in this two-dimensional slotframe is identified through a couple of information, namely *timeslot*, and *channel offset*. Finally, to mitigate the negative effects of multi-path fading and interferences, TSCH leverages frequency hopping. A predefined frequency-hopping sequence is shared among all the nodes in the network, so that they can select a different operating frequency at each timeslot.

The TSCH protocol provides mechanisms to allocate and deallocate cells to network nodes, according to a communication schedule. Among these mechanisms, the TSCH MAC frame includes the *Frame Pending* (FP) flag, a bit within the Frame Control Field, that can be used by the sending device to signal that it has more data for the recipient. When set to one, the FP bit indicates that the recipient should remain active in the next timeslot and on the same channel, unless the next cell is already allocated for other purposes.

While TSCH provides mechanism to allocate/deallocate cells, it does not specify how cells are allocated to nodes for communication. To this purpose, the 6TiSCH architecture includes the 6TiSCH Operation (6top) sublayer that provides the abstraction of IP link over TSCH, by managing the allocation of cells to nodes in such a way to meet the application requirements.

Above the 6top layer, the 6LoWPAN adaptation protocol is responsible for encapsulating IPv6 datagrams into TSCH frames, while the IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) ensures multi-hop transfer of IPv6 datagrams. RPL organizes the network nodes in a *Destination Oriented Directed Acyclic Graph (DODAG)*, where every node selects a neighbor, called *preferred parent*, as the candidate neighbor for upstream data delivery. The DODAG is rooted at a single node, the root node, that is typically the collector of the network to which upstream data is directed. Although RPL is optimized for upstream data delivery, downstream data delivery from the root to the others nodes is also supported. Finally, the end-to-end delivery of data packets originated by the application is managed by the UDP protocol.

The 6top sublayer is a crucial part of the 6TiSCH architecture, as it determines the schedule used by nodes for communication. It consists of two main components, namely a Scheduling Function (SF) to calculate the number of cells to allocate, depending on the current conditions, and the 6top (6P) protocol to negotiate the required cells with neighbor nodes. The main SFs for 6TiSCH will be discussed in the next Section. We provide below a brief description of the 6P negotiation protocol.

6P [16] defines the operations and messages to implement a complete negotiation between two nodes (6P transaction). A 6P transaction consists of a *Request* followed by a *Response*. The Request message includes a code to specify the requested action, namely, ADD (to request a new allocation), DELETE (to cancel an existing allocation), or CLEAR (to reset the current negotiation). Similarly, the Response message includes a SUCCESS or ERROR code, to notify a successful or failed transaction, respectively.

Typically, a requesting node (node A) sends an ADD request to the corresponding node (node B), specifying the number of cells to allocate and a list of free cells, and node B replies with a SUCCESS response containing the list of allocated cells. However, a transaction may be unsuccessful, e.g., when node B cannot fulfill the request. If a Request/Response message gets lost or corrupted, the Request message is retransmitted after a predefined timeout (6P Timeout). Finally, when a mismatch is detected by node B (e.g., due to a lost Response message), it replies with an ERROR message that forces node A to send a CLEAR message and reset the schedule (both nodes cancels all the allocated cells). Then, node A has to start over the allocation process.

III. SCHEDULING FUNCTIONS

In this section, we first provide a classification of the main SFs proposed in the literature and discuss the related work. Then, we will focus on the three SFs considered in our study.

A. CLASSIFICATION AND RELATED WORK

A significant number of SFs for 6TiSCH networks have been proposed to cope with the requirements of different use cases. They can be broadly classified according to the paradigms considered by the 6TiSCH WG [19], namely, *centralized*, *distributed*, *autonomous* (or static), and *hop-by-hop* scheduling. In addition, there are also *hybrid* solutions that combine some of the previous approaches.

Centralized scheduling leverages a central entity, referred to as *Point Coordination Element* (PCE), that collects information about the network topology and traffic patterns and, based on those, computes the communication schedule [3]–[5]. Centralized SFs provide optimal schedule but require a high communication overhead. Moreover, they are unable to adapt to time-varying traffic and network conditions and, hence, unsuitable to large or dynamic networks.

In *distributed scheduling* [7]–[10], [20], nodes negotiate the allocation of TSCH cells with their neighbors, using the 6P protocol, and adapt the number of allocated cells, depending on traffic and network conditions. The communication schedule is typically non-optimal, however this approach is very suitable to large-scale and/or dynamic networks. Among distributed SFs, OTF [11] was originally considered by the 6TiSCH WG for standardization as the reference SF. However, OTF has a number of limitations in estimating the number of cells required by the application, as well as in managing congestion (e.g., originated by changes in the preferred parent). These limitations are overcome by E-OTF [7], [17], an enhanced version of OTF.

The main drawback of distributed scheduling is the overhead due to 6P transactions. To avoid this overhead, *autonomous scheduling* [18] was introduced, where nodes allocate cells autonomously (i.e., without negotiation), using a hash function applied to nodes' addresses. More specifically, Orchestra [12] leverages a node-based approach that can follow two different allocation styles, namely, *receiver-based* and *sender-based*. In receiver-based scheduling, each node allocates one cell per slotframe to *receive* packets from all its neighbors, while in sender-based scheduling each node has only one cell per slotframe to *send* packets to all its neighbors. Orchestra is currently the default SF used in the Contiki-NG operating system. ALICE (Autonomous Link-based Cell Scheduling) [13] replaces the node-based allocation scheme used in Orchestra with a link-based scheme, where each node allocates a cell per slotframe for each unidirectional link. By allocating more cells per slotframe than Orchestra, ALICE exhibits better performance, in terms of packet delivery ratio and latency in almost all scenarios [18].

Autonomous scheduling is basically static and, hence, unable to manage unpredictable changes in traffic conditions. To overcome this limitation, in TESLA [14] each node monitors its traffic load and changes the size of the slotframe according to the traffic rate. The cost to pay for adaptation consists in the overhead due to control information to be exchanged (i.e., the new slotframe size in TESLA). Basically, adaptive autonomous SFs take a *hybrid* approach that combines autonomous and distributed scheduling.

Similarly, the Minimal Scheduling Function (MSF) [6], currently under standardization by the 6TiSCH WG as the reference SF, combines autonomous and distributed scheduling. More specifically, nodes use both *autonomous* and *negotiated* cells. Autonomous cells provide a basic amount of bandwidth for control messages and are allocated using a hash function on node addresses (i.e., without negotiation). Instead, negotiated cells are allocated and deallocated dynamically, based on the level of utilization, through the 6P protocol.

Distributed and autonomous approaches have most attracted the attention of the research community, in the last years. Basically, autonomous scheduling reduces the control overhead, while distributed scheduling allows adaptation to traffic changes. Finally, hybrid approaches try to get the best from both. Many performance evaluations have been carried out so far; however, these studies typically compare similar solutions. A direct comparison of distributed and autonomous approaches that emphasizes pros and cons of each of them is still missing. In this article, we compare three different SFs that take different approaches. Specifically, we consider ALICE [13] as autonomous SF, E-OTF [7] as fully distributed SF, and MSF [6] as hybrid SF, and analyze their performance in four different scenarios. Since ALICE is static (while E-OTF and MSF are adaptive to traffic changes), we also consider ALICE with the Frame Pending option enabled, which makes it adaptive to traffic changes. In the following, we present a brief presentation of the three considered SFs.

B. ALICE

The key feature of autonomous scheduling is the lack of control overhead in computing the communication schedule, since cells are allocated autonomously by nodes. Indeed, nodes only exploit information made available by the routing layer and a hash function to allocate cells, without any explicit neighbor-to-neighbor negotiation. By exploiting routing information, autonomous SFs guarantee that the schedule remains consistent with the network topology, despite of topology changes.

ALICE [13] was proposed to enhance Orchestra [12], especially in large and/or dense networks. ALICE replaces the node-based allocation scheme used in Orchestra with a link-based scheme, where each node allocates a single cell per slotframe for every directional link it is involved in. Specifically, a generic node allocates one receiving cell for each of its child nodes and one additional cell for transmissions towards its parent node. By allocating more cells per slotframe than Orchestra, ALICE provides better performance, in terms of end-to-end reliability and latency, especially when the traffic rate increases over a certain threshold.

In ALICE, each node allocates cells autonomously, by computing the corresponding timeslot and channel offset in the two-dimensional slotframe, using a hash function with the following parameters:

- node ID of itself and of its parent/child nodes;
- direction of the communication link (i.e., upstream or downstream)
- slotframe length.

Since the hash-based allocation may result in collisions (i.e., the same cell used by different nodes), the allocation is recomputed by each node at every slotframe. This reduces the negative effects of collisions. Indeed, it was shown in [18] that the cell reallocation improves the performance with respect to the worst-case behavior, while its impact on the average performance is negligible.

From the above description, it clearly emerges that in ALICE the number of allocated cells only depends on the current number of links a node is involved in, while it does not take into consideration the real amount of traffic managed by that node. Hence, ALICE is unable to adapt to unpredictable changes in the traffic pattern (while E-OTF and MSF are adaptive). To allow ALICE to adapt to traffic changes, we exploited the FP mechanism provided by the underlying TSCH protocol [2] that allows the sending node to signal that it has more data to send (see Section II). Using this very simple mechanism, a node can transmit more data packets than the number of scheduled cells, thus facing temporary traffic peaks. More important, these additional cells are obtained without negotiation and, hence, with no overhead (the FP bit is piggybacked in the TSCH frame).

C. E-OTF

Unlike ALICE, E-OTF [7] takes a fully distributed approach in allocating cells, and relies on 6P for neighbor-to-neighbor negotiation. Basically, for each couple of nodes involved in a parent-child relationship, E-OTF monitors the (locally generated and forwarding) traffic towards the parent node and estimates the number of cells required to manage it. Consequently, it determines the number of cells to be added or removed to the current allocation, and triggers the 6P protocol to allocate/deallocate cells, accordingly. In order to reduce the continuous allocation and deallocation of cells, the algorithm includes a hysteresis mechanism that usually results in overprovisioning.

However, estimating the number of required cells only basing on the observed traffic - without considering other factors, such as link quality fluctuations, failures of 6P transactions, temporary disconnections, changes in the preferred parent - may result in significant underestimation of the number of required cells and, consequently, in performance degradation. To prevent bandwidth underestimation, E-OTF explicitly considers the *Expected Transmission Count* (ETX) [21] of the link when computing the required number of cells. In addition, E-OTF introduces a congestion mechanism that is enabled when the local queue size increases over a predefined threshold β. This allows the node to avoid local congestion, or recover from it in a short time. Specifically, a congestion bonus B is used to overprovision the congested node and allows it to transmit the buffered packets more rapidly. Finally, E-OTF monitors the cell utilization U, so as to release the extra-cells cells, acquired during a congestion phase, if they are not necessary anymore.

The detailed E-OTF scheduling algorithm is illustrated in Algorithm 1. The number of cells to be negotiated is determined using the following information: S_c (number of currently scheduled cells), R_c (number of required cells, estimated on the packet generation period (P) and ETX), and T (hysteresis quantum). Whenever the local buffer size exceeds the congestion threshold β, the congested mechanism is enabled, which implies the allocation of the congestion bonus B to the node at each iteration (line 2). By comparing R_c and S_c, the algorithm computes the number of cells to be added or removed, also considering the hysteresis quantum T (lines 4 and 7). When cells need to be removed (line 5), if the congestion mechanism B is enabled and the Utilization U is higher than α, only the B cells acquired as bonus are removed (line 6).

E-OTF can operate in different application domains. However, it is mainly targeted to data collection applications, where most of the traffic flows from networks nodes to the root of the DODAG, while downward traffic is assumed to be sporadic. Hence, this SF is optimized for upward traffic.

Algorithm 1: E-OTF Algorithm

```
Input:
    $P$ = Packet generation period
    $S_c$ = Number of scheduled cells
    $R_c$ = Number of required cells
    $T$ = Hysteresis Quantum
    $B$ = Congestion Bonus (CB)
    $Q$ = Average Queue Occupancy
    $U$ = Average Cell Utilization
    $ETX$ = Estimated Transmission Count
Output:
    $\Delta S$ = Number of cells to add/delete

1 $R_c = R_c(P, ETX)$
2 if $Q > \beta$ then $\Delta S = B$                  $\Rightarrow$ ADD CB
3 if $R_c > S_c$ then
4     $\Delta S += R_c - S_c + \lceil T/2 \rceil$     $\Rightarrow$ ADD CELLS
5 else if $R_c < S_c - T$ then
6     if $U > \alpha$ then $\Delta S = B$             $\Rightarrow$ REMOVE CB
7     else $\Delta S = S_c - R_c - \lfloor T/2 \rfloor$ $\Rightarrow$ DEL CELLS
8 else $\Delta S = 0$                                 $\Rightarrow$ DO NOTHING
```

Downward traffic is managed by allocating a number of shared cells per slotframe that can be used by all the nodes, to send data to their children, on a contention basis.

D. MSF

The 6TiSCH Minimal Scheduling Function (MSF) [6] takes a hybrid approach, as it combines autonomous and distributed scheduling and allocates both autonomous and negotiated cells. Autonomous cells are allocated autonomously (i.e., without neighbor-to-neighbor negotiation), using a hash function to compute their timeslot and channel offset, and are used for control information. Instead, negotiated cells are allocated (and deallocated) through the 6P protocol and allow to adapt to time-varying traffic conditions.

MSF allocates two autonomous cells at each node, namely, an *Autonomous Rx Cell* (AutoRxCell) and an *Autonomous Tx Cell* (AutoTxCell). The AutoRxCell is used for receiving control messages and is permanently allocated at joining time, while the AutoTxCell is allocated only after the node has selected its parent node and changes when a new parent node is chosen. In addition, it is shared with all the other children of the same parent. Finally, it is not permanently scheduled, but added when there is a control message to send, and deleted just after.

Negotiated cells are allocated and deallocated dynamically, based on traffic requirements, following a utilization-based approach. Initially, each node negotiates a single cell with its parent node. Then, MSF periodically (every NUM_MAX_CELLS) checks the utilization of the node, defined as the percentage of used cells with respect to scheduled cells. The algorithm considers two thresholds to decide when to add, or remove, a new cell. As shown in Algorithm 2,

Algorithm 2: MSF Algorithm

```
Input:
    $NCE$ = Number of elapsed negotiated cells
    $MAX\_NUMCELLS$ = Max number of elapsed negotiated cells
    $NCU$ = Number of negotiated cells used for transmission
    $LIM\_NUMCELLSUSED\_HIGH$ = Threshold to add negotiated cell
    $LIM\_NUMCELLSUSED\_LOW$ = Threshold to delete negotiated cell
Output:
    ADD/DEL one negotiated cell

1 if $NCE > MAX\_NUMCELLS$ then
2     if $NCU > LIM\_NUMCELLSUSED\_HIGH$ then
3         trigger 6P to ADD one negotiated cell
4     if $NCU < LIM\_NUMCELLSUSED\_LOW$ then
5         trigger 6P to DEL one negotiated cell
```

if the cell utilization is higher than the upper threshold, one more cell is negotiated with the parent node. Instead, if the utilization falls below the lower threshold, one cell is deleted from the current schedule.

MSF is designed to operate in a wide range of application domains. However, like E-OTF, it is optimized for applications with regular upstream traffic from the nodes to the root [6]. Downward traffic is assumed to be sporadic and its management is not specified. In the implementation used for our experiments, downward traffic is managed through shared cells, as in E-OTF.

IV. EVALUATION METHODOLOGY

In this section we present the methodology used in our evaluation. To compare the performance of the selected SFs in a large variety of scenarios, we used a mixed approach based on both simulations and experimental measurements. Simulation allows us to analyze the performance of the considered SFs in a large number of scenarios and investigate the impact on performance of different factors, such as slotframe length, network size, traffic pattern, traffic rate, and so on. This would be very difficult, if not unfeasible, using a testbed. However, since simulation is not able to capture all the aspects of a real deployment, we also repeated a limited set of experiments on a testbed. The purpose of this experimental analysis is twofold. We want to validate our simulation results and, in addition, to investigate how the considered SFs perform in a real environment.

To carry out our analysis, we implemented the three selected SFs in the Contiki-NG Operating System (OS),[1] a popular OS for sensor networks that run on a wide range of sensor platforms. Contiki-NG already supports the 6TiSCH basic operations, which simplifies the implementation of the SFs. Specifically, for our simulations we exploited Cooja [22], a simulator for sensor networks that is part of the Contiki-NG suite. It supports the emulation of hardware components of sensor nodes on which the same code written for real devices can be run. Instead, for real experiments, we leveraged the PINT testbed, an IoT testbed deployed in a two-floor building of our Department, at the University of Pisa. It is composed of 22 sensor nodes, each of which equipped with a Zolertia REMote,[2] characterized by ARM Cortex-M3 micro-controller at 32 Mhz, 32KB of RAM, and CC2538 wireless transceiver that implements the IEEE 802.15.4 standard at 2.4 GHz. In all the (simulation and real) experiments, we used the RPL routing protocol with default parameters to allow multi-hop communication.

In our analysis, we mainly considered the following metrics.

- **Packet Delivery Ratio (PDR)**, defined as the ratio between the overall number of data packets received by the final destination and the number of packets sent by all the nodes. It measures the end-to-end reliability provided by the SF.
- **End-to-end Latency**, defined as the time interval between the generation of a data packet at the source node and its correct reception at the final destination. This metric measures the timeliness of the SF in managing data collection.
- **Duty Cycle**, defined as the ratio between the number of cells in which a node remains awake to receive/transmit a data packet and the slotframe length. This metric provides an indirect measurement of the energy consumption of the node. It also takes into consideration the negotiation overhead required by a specific SF, as 6P messages require additional cells, with respect to data packets, during which nodes must remain active.

In some experiments we also measured the queue size, i.e., the number of data packets stored in the local buffer of the node, waiting for transmission. This gives an indication of possible congestion experienced by the node.

In our analysis, we considered four different scenarios, characterized by different communication paradigms (many-to-one, one-to-many, many-to-many) and traffic patterns (periodic or bursty), so as to investigate different use cases. Specifically, in the first two scenarios we considered the case where all the nodes send their data packets to the root (*many-to-one communication*). They both refer to data collection applications (e.g., monitoring applications), but differ in the traffic generation pattern. In the first scenario nodes generate and report data *periodically*, as in environmental monitoring applications. In the second scenario, the traffic pattern is *bursty*, i.e., nodes occasionally send a burst of data packets, while for the rest of the time they remain idle. This scenario is representative of event-driven applications (e.g., an alert

[1]https://github.com/contiki-ng/contiki-ng.Accessed:08-04-2020

[2]https://github.com/Zolertia/Resources/wiki/RE-Mote.Accessed:08-04-2020

system), in which the detection of a certain event triggers the generation of a number of data packets (e.g., an image taken from a camera). In the third scenario we considered the case where the root sends packets to all (or many) nodes in the network (*one-to-many communication*). This happens, for instance, when the root needs to send commands/updates to sensor or actuator nodes. Finally, in the last scenario we considered the case of *one-to-one communication*, where a device (e.g., a sensor) sends a flow of data packets to another device (e.g., an actuator). This occurs in all those applications (e.g. industrial instrumentation) that require a direct communication between devices, i.e. Machine-to-machine (M2M) communication.

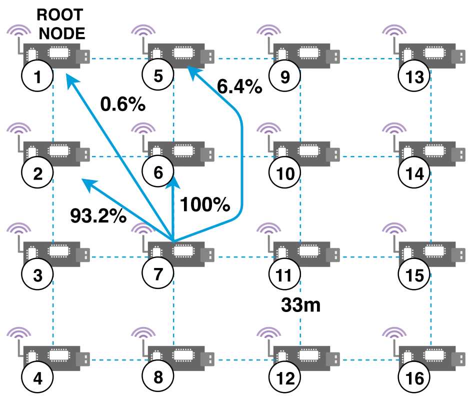

FIGURE 2. Example of a 4 × 4 grid topology with average Packet Delivery Probabilities.

In our study, we investigated the scalability of the considered SFs, with respect to different factors, such as number of nodes and traffic rate. We also studied the impact of some TSCH parameters. Specifically, we considered the effect of varying the slotframe length S. In addition, as anticipated, we investigated the impact of the TSCH Frame Pending bit, especially on the performance of ALICE.

Both simulation and real experiments were run for a fixed period of time, namely, 1 hour for simulations and 45 minutes for experiments on the testbed. Nodes are programmed to start the generation of traffic after 4 minutes from the beginning of the experiment. This is to allow the network formation phase (DODAG formation) that may take up to some minutes. In order to obtain statistically sound results, we performed 10 independent replicas for each simulation experiment and 5 different replicas for each experiment on the testbed. For each metric, the average value, computed over all the replicas, is reported in the plots below. When significant, the 95% confidence level is also shown.

V. SIMULATION RESULTS

In this section, we present the results obtained with our simulation experiments. To this end, we refer to the network depicted in Figure 2, namely a grid topology with a number of nodes (N) distant 33m from each other. To make the communication channel realistic, each link is modeled

TABLE 1. Simulation Setting and Parameters.

Simulation setting and parameters	
MACMinBE	**1**
MACMaxBE	**5**
MACMaxFrameRetries	**7**
Queue Buffer	**16**
RPL Objective Function	**MRHOF - ETX**
RPL Mode	**Storing**
RPL DIO interval doublings	**8**
RPL redundancy threshold	**10**
TSCH slotframe number	**3** TSCH EB slotframe (s0) RPL-6P slotframe (s1) unicast slotframe (s2)
TSCH s0 slotframe length	**101**
TSCH s1 slotframe length	**31**
TSCH s2 slotframe length	**17-29-47**
TSCH timeslot duration	**10ms**
TSCH number of channels for s0	**4**
TSCH number of channels for s1 & s2	**16**
6top Timeout	**32s**
TSCH Control Timeslots (s0)	**1**
6top Timeslots (s1)	**2**
RPL Control Timeslots (s1)	**1**

through the *Multi-path Ray-tracer Medium* (MRM) model [22]. MRM implements ray-tracing techniques with various propagation effects (e.g., multi-path, refraction, diffraction, etc.), and associates a *Packet Delivery Probability* (PDP) to each link, which changes over time due to propagation effects and concurrent transmissions. The average PDP for different kind of links is shown in Figure 2. Each node is configured to send UDP data packets with a certain generation period (P). All the parameter settings used in the simulation experiments are shown in Table 1.

In the following, we first discuss the simulation results obtained in the scenarios characterized by many-to-one communication with both periodic traffic (Section V-A) and bursty traffic (Section V-B). Then, we consider the scenarios with one-to-many communication (Section V-C) and one-to-one communication (Section V-D).

A. MANY-TO-ONE COMMUNICATION WITH PERIODIC TRAFFIC

In this scenario, each node in the network generates UDP packets of fixed size (60 bytes), destined to the root node, with a constant period P. For brevity, this traffic pattern will be referred to as CBR (*Constant Bit Rate*).

We start our performance comparison by analyzing the scalability of the SFs with the network size. To this end, we vary the number of nodes N in the network from 16 to 64. In this first set of experiments, the packet generation period (P) is constant and equal to 5s. In a second set of experiments, described later, we will investigate the scalability with the

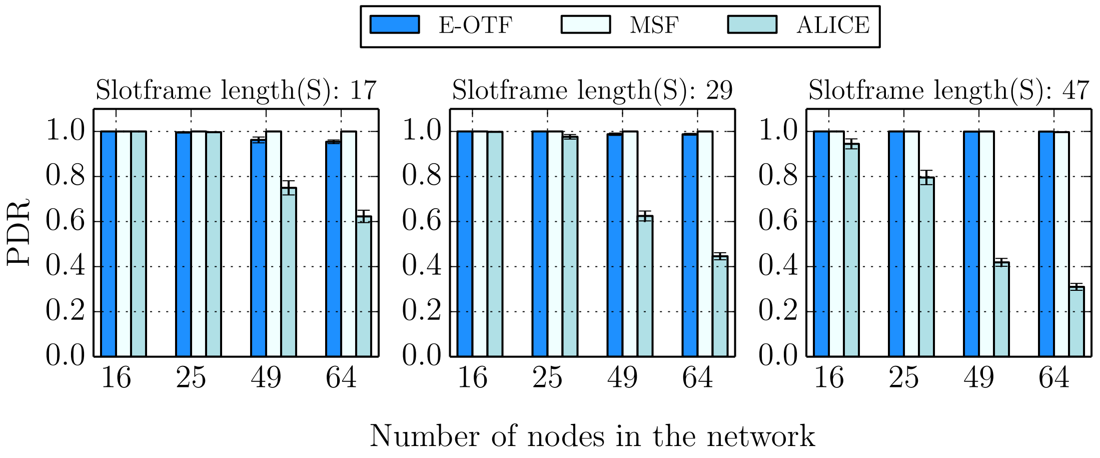

FIGURE 3. PDR for E-OTF, MSF and ALICE. Many-to-One Communication, CBR Traffic. Packet generation period (*P*): 5s.

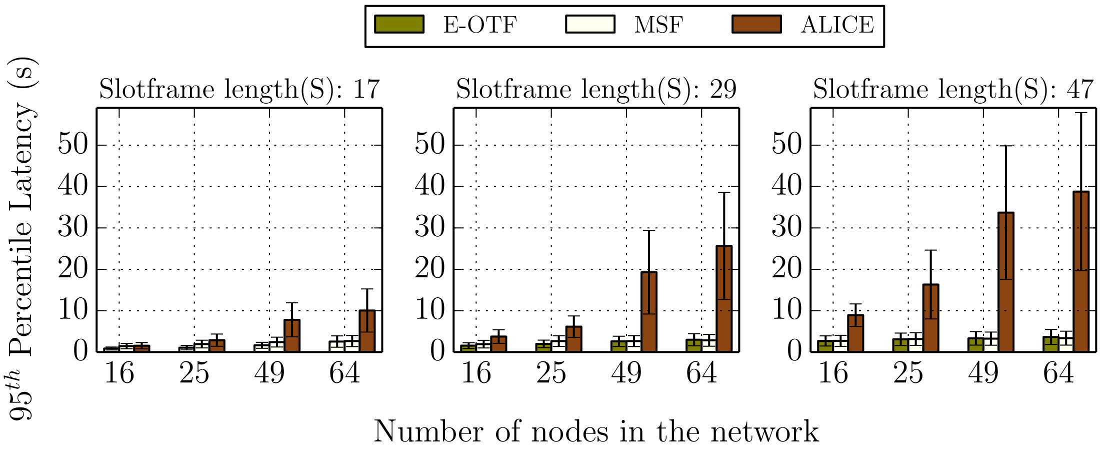

FIGURE 4. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Many-to-One Communication, CBR Traffic. Packet generation period (*P*): 5s.

traffic rate by changing the P value, while keeping the number of nodes constant.

Figure 3 shows the Packet Delivery Ratio (PDR) provided by the three considered SFs, as a function of the network size, for different lengths of the slotframe (namely, $S = 17, 29, 47$). The general trend is the same for all the considered S values. As expected, when the number of nodes grows up, the PDR tends to decrease. However, this decrease is very remarkable with ALICE, while it is only light, or even negligible, for MSF and E-OTF. This is because in ALICE, the number of cells (per slotframe) allocated to a node is constant, and only depends on the number of links managed by the node. If the number of (source) nodes in the network increases, the overall traffic to manage increases accordingly, and ALICE (unlike MSF and E-OTF) is unable to adapt. Figure 3 also shows that, for a given network size, the PDR provided by ALICE is strongly influenced by the slotframe length, while both MSF and E-OTF are not very sensitive to this parameter. The reason is the same as above. In ALICE, the number of scheduled cells per slotframe is constant. Hence, if the slotframe length increases, the allocated bandwidth (measured in cells per second) decreases accordingly. With 64 nodes, the PDR provided by ALICE reduces from about 62% to 31%, when S passes from 17 to 47.

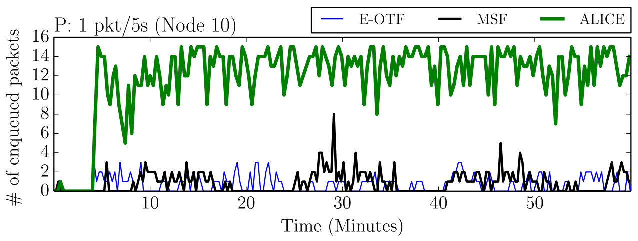

FIGURE 5. Queue occupancy of node 10 in the 8 × 8 grid topology for E-OTF, MSF and ALICE. Many-to-One Communication, CBR traffic. Packet generation period (*P*): 5s. Slotframe length (*S*): 29.

Figure 4 reports the 95th percentile of the end-to-end latency, as a function of the network size, for the three considered S values. The trend is the same as above: the latency increases with the network size, as expected; however, the increase is dramatic with ALICE (up to 40 seconds), while it is limited for MSF and E-OTF (less than 5 seconds).

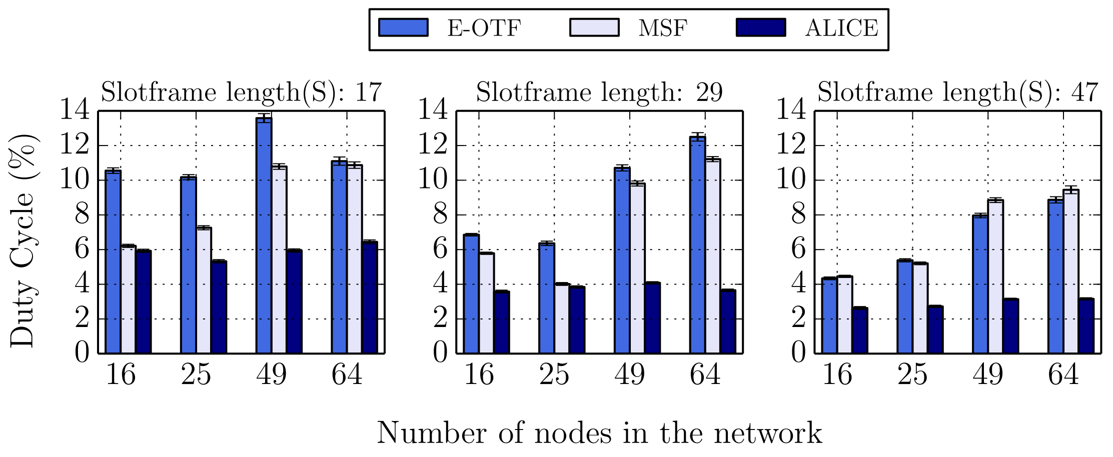

FIGURE 6. Duty cycle of the most loaded node for E-OTF, MSF and ALICE (Node 6 for *N*=16, node 7 for *N*=25, node 9 for for *N*=49 and node 10 for *N*=64). Many-to-One Communication, CBR Traffic. Packet generation period (*P*): 5s.

Again, the performance of ALICE is highly influenced by the slotframe length, for the same reasons exposed above.

To better understand the different behavior of ALICE, with respect to E-OTF and MSF, we looked at the local queue at nodes. Figure 5 shows the queue length, as a function of time, for the most loaded node in the network, namely, node 10 in the 8×8 grid topology. We considered this node because it is located at one hop from the root node and in a central position with respect to the other nodes. Hence, it has to forward the data packets originated by all the other nodes in its subtree, which is the biggest in comparison with the other subtrees rooted at its neighbors. The data reported in Figure 5 comes from a single simulation run, with packet generation period of 5 seconds and slotframe length set to 29. We observe that, after the nodes start sending data (i.e., at around minute 4), with ALICE the queue occupancy tends to grow up and, then, stabilizes in the range of 10-15 packets. This results in increased delay experienced by packets and, possibly, packet dropping. Hence, the low reliability and high end-to-end latency observed in Figure 3 and Figure 4, respectively. On the other hand, with E-OTF and MSF, the queue size oscillates over time but remains under control and never reaches very high values, due to their capability to adapt to traffic conditions. It may be worthwhile observing here that E-OTF experiences a lower average queue than MSF, thanks to its congestion bonus.

The cost to pay for the better performance of E-OTF and MSF, with respect to ALICE, is the higher duty cycle, as shown in Figure 6. This is because ALICE allocates a fixed number of cells (per slotframe), while MSF and E-OTF adjust the number of cells to traffic conditions. This also explains why the duty cycle of ALICE does not depend on the number of network nodes. In addition, ALICE does not introduce any negotiation overhead. Instead, the duty cycle of MSF and E-OTF also includes the contribution of 6P messages for cell negotiation. E-OTF typically has a larger duty-cycle than MSF because it allocates more cells (due to the congestion bonus).

So far, we have compared the performance of the considered SFs with increasing network sizes. In order to assess the scalability with respect to the traffic rate, we carried out an additional set of simulation experiments, where we varied the packet generation period P, while keeping constant the number of nodes ($N = 25$) and the slotframe length ($S = 29$). The considered network size is the largest one for which ALICE exhibits acceptable reliability in Figure 3.

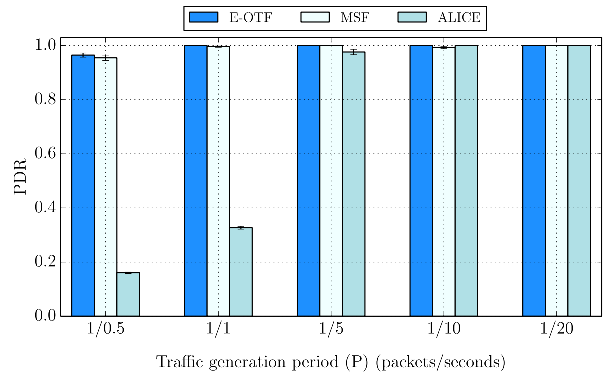

FIGURE 7. PDR for E-OTF, MSF and ALICE. Many-to-One Communication, CBR Traffic. Slotframe length (*S*): 29, *N* = 25.

Figure 7 and Figure 8 show the delivery ratio and 95th percentile of end-to-end latency, respectively. ALICE exhibits a performance similar to MSF and E-OTF as long as the traffic rate remains below a certain threshold (i.e., 1 packet every 5 seconds, in our experiments). When the traffic rate is high, the performance of ALICE drops dramatically, because of its static allocation, while E-OTF and MSF do not experience significant variations, thanks to their ability to adapt to traffic conditions. In terms of duty cycle (Figure 9), ALICE has the lowest value, which is not influenced by the traffic rate, as expected. MSF and OTF experience a higher duty cycle, especially under high traffic conditions, as they allocate more cells and due to the negotiation overhead (6P messages). E-OTF is the SFs with the highest duty cycle.

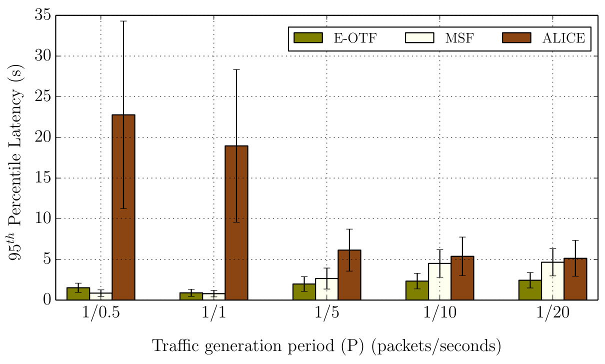

FIGURE 8. 95^{th} **Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Many-to-One Communication, CBR Traffic. Slotframe length (S): 29, $N = 25$.**

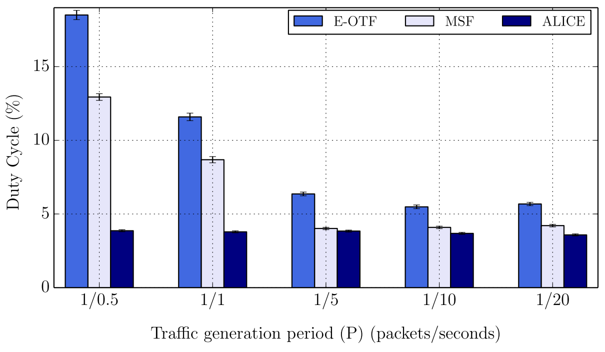

FIGURE 9. **Duty cycle of the most loaded node (node 7) for E-OTF, MSF and ALICE. Many-to-One Communication, CBR Traffic. Slotframe length (S): 29, $N = 25$.**

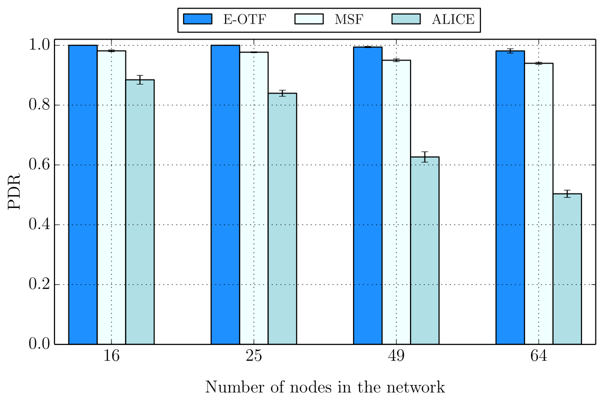

FIGURE 10. **PDR for E-OTF, MSF, and ALICE. Many-to-One Communication, Bursty traffic. Slotframe length (S): 29.**

B. MANY-TO-ONE COMMUNICATION WITH BURSTY TRAFFIC

In this Section, we present the results obtained in the scenario with many-to-one communication and bursty traffic, which is representative of event-driven monitoring applications. The goal is to assess the performance of the considered SFs in a case when nodes alternate between idle periods and periods with high traffic intensity. More specifically, each node in the network alternate between OFF periods, when no packet is generated, and ON periods, during which 100 UDP packets, of fixed size (60 bytes), are generated and sent to the root node with a period of 0.5 seconds. After sending the last UDP packet the node stops, until the beginning of the next ON period. The duration of OFF periods is a random variable with uniform distribution between 1 and 20 minutes. In all the experiments, we considered a network with increasing size (from 16 to 64 nodes), but a fixed sloftrame length ($S = 29$).

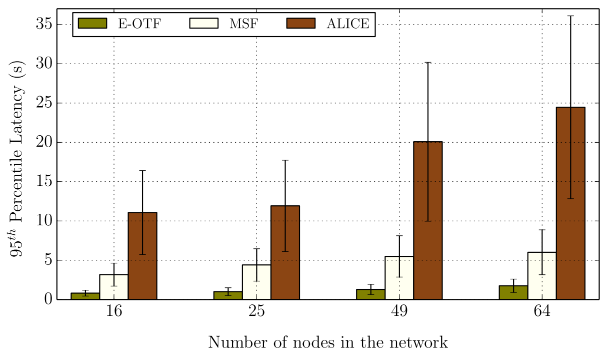

FIGURE 11. 95^{th} **Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Many-to-One Communication, Bursty traffic. Slotframe length (S): 29.**

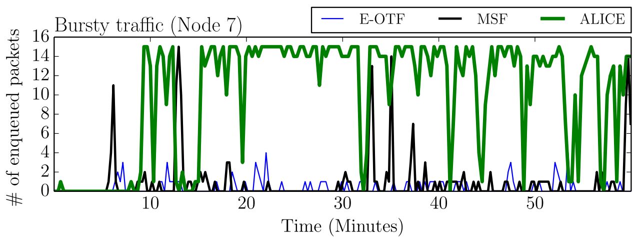

FIGURE 12. **Queue occupancy of node 10 in the 8 × 8 grid topology for E-OTF, MSF and ALICE. Many-to-One Communication, Bursty traffic. Slotframe length (S): 29.**

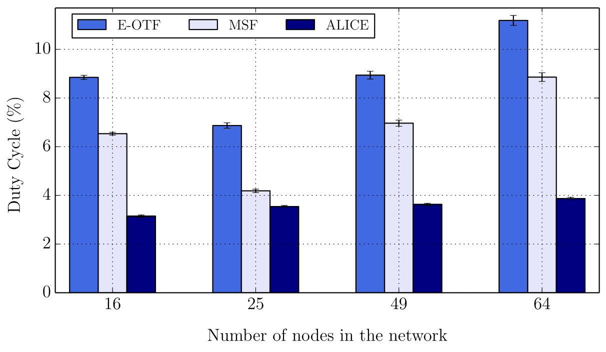

FIGURE 13. **Duty cycle of the most loaded node for E-OTF, MSF and ALICE (Node 6 for $N = 16$, node 7 for $N = 25$, node 9 for for $N = 49$ and node 10 for $N = 64$). Many-to-One Communication, Bursty traffic. Slotframe length: 29.**

Figure 10 shows the packet delivery ratio. The general trend is the same as the one observed in the scenario with CBR traffic (see Figure 3), i.e., the performance of ALICE decreases when the network size increases, while MSF and E-OTF are not influenced significantly by this parameter.

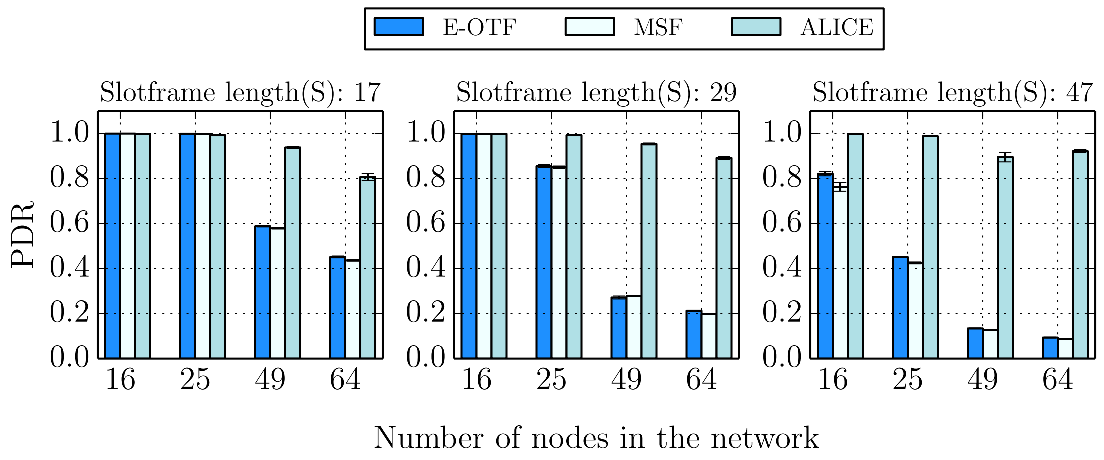

FIGURE 14. PDR for E-OTF, MSF and ALICE. One-to-Many Communication, Downward traffic. Packet generation period (*P*): 30s.

However, if we compare the results in Figure 10 with those in Figure 3, we can see that, with bursty traffic, ALICE provides a lower delivery ratio. This is because, when the traffic pattern is bursty, it may happen that multiple nodes are simultaneously in their ON period and, consequently, a high traffic intensity is generated. Since ALICE relies on a static allocation, it is unable to cope efficiently with such a situation. Instead, both E-OTF and MSF can temporarily increase the number of allocated cells, thus reducing the likelihood of buffer overflow to occur.

This conclusion is also confirmed by the queue level, over time, shown in Figure 12 (for node 10 in the 8×8 grid topology). With ALICE, the number of packets buffered in the local queue is very often close to the maximum value. Instead, with both E-OTF and MSF the queue is always under control. We also observe that E-OTF has, on average, a lower queue occupancy than MSF. This justifies the better performance exhibited by E-OTF, with respect to MSF, in the scenario with bursty traffic, in terms of both delivery ratio (Figure 10) and, above all, end-to-end latency (Figure 11).

More specifically, Figure 11 shows that, even with a small network size (16 nodes), the 95^{th} percentile of the end-to-end latency introduced by ALICE is beyond 10 second. Moreover, it increases very sharply with the network size. Instead, the delay introduced by E-OTF is always in the order of 1-3 seconds.

In terms of duty cycle (Figure 13), we can see the same trend already observed in the previous scenario with CBR traffic, with E-OTF exhibiting the highest value. However, with bursty traffic, the gap between E-OTF and MSF is reduced and the higher duty cycle experienced by E-OTF is justified by its better performance, in terms of delivery ratio and, above all, end-to-end latency.

The conclusion we can draw from the previous results is that ALICE is unable to manage (many-to-one) scenarios with bursty traffic conditions. Among adaptive SFs, E-OTF is more suitable than MSF, as it is able to react more quickly to changing traffic conditions.

C. ONE-TO-MANY COMMUNICATION

In the previous sections we have analyzed the performance of the considered SFs when the traffic flows *upward*, i.e., from nodes to the root. In this section we want to investigate the case when the traffic flows *downward*, i.e., from the root to nodes. This is not so frequent in IoT environments. However, for sure it happens when the root needs to send a command to sensors (e.g., updated parameter values, configuration changes, etc.) or actuators (e.g., actions to be performed). In our experiments, we assumed that the root sends UDP packets of fixed size (60 bytes) to nodes in the network, with a certain period P. As above, we considered a network with increasing size (from 16 to 64 nodes) and three different lengths of the slotframe ($S = 17, 29, 47$).

In the first set of experiments we assumed that the packet generation period is large (e.g., 1 minute) and each packet is directed to a single destination, randomly selected among all nodes, so as to simulate the case of *sporadic* downward traffic. Under such conditions, we observed that all the considered SFs perform well, irrespective of the network size and slotframe length. For instance, in the worst case (i.e., with $N = 64$ and $S = 47$) we obtained a PDR of approximately 95% for all the considered SF, and an end-to-end latency of approximately 35s for MSF and E-OTF, and about 3s for ALICE.

To analyze the impact of the downward traffic on the performance of the SFs, we increased the amount of traffic injected into the network, by decreasing the packet generation period (P) at the root and/or increasing the number of destinations the packet is directed to. Figure 14 and Figure 15 show the PDR and the 95^{th} percentile of end-to-end latency, respectively, when P is 30s and different copies of the packet are destined to *all* the nodes in the network. This corresponds to a very high aggregate downward traffic and allows to understand the behavior of each SF in extreme conditions.

We can observe that, for all the considered SFs, the performance degrades as the network size and/or slotframe length increases, as expected. However, unlike the previous

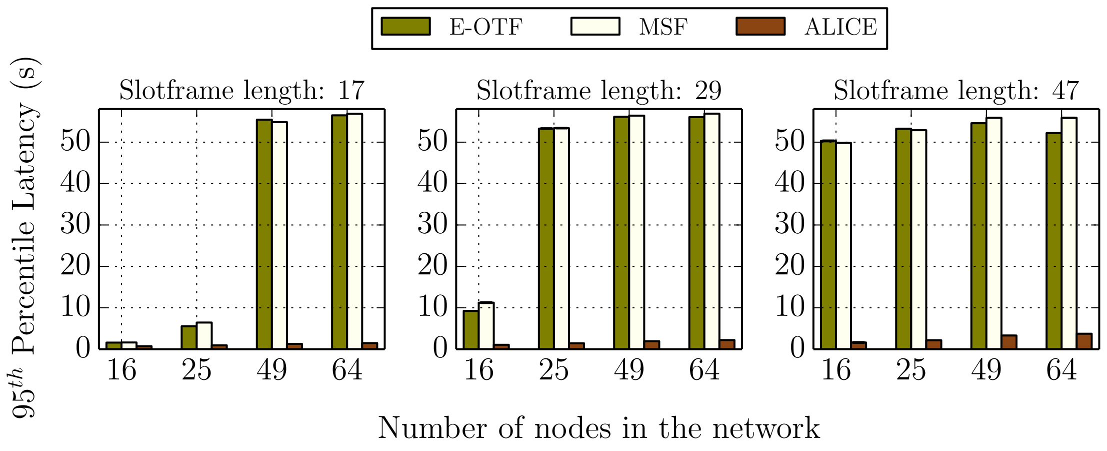

FIGURE 15. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. One-to-Many Communication, Downward traffic. Packet generation period (*P*): 30s.

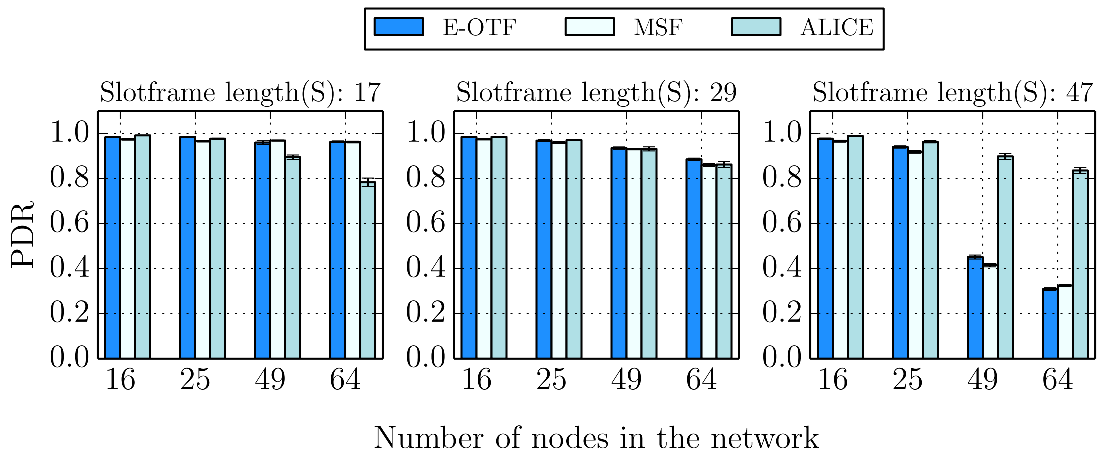

FIGURE 16. PDR for E-OTF, MSF and ALICE. One-to-One Communication, device-to-device traffic. Packet generation period (*P*): 30s.

scenarios with upward traffic, now the performance degradation is moderate for ALICE and very remarkable for MSF and E-OTF. Specifically, MSF and E-OTF exhibit a very low delivery ratio when the aggregate traffic load is high, i.e., for large network size and/or slotframe length. This is because both MSF and E-OTF are optimized for upward traffic and, in our implementation, they manage downward traffic by allocating a number of shared cells per slotframe that can be used by all the nodes, on a contention basis. For high traffic loads, this results in a very large number of collisions and, hence, high latency and packet dropping, as reflected in Figure 15 and Figure 14, respectively. Instead, with ALICE each node allocates one cell for each link it has with a neighbor and, hence, it leverages dedicated cells (instead of shared cells) also for managing downward traffic.

The conclusion we can draw from the previous results is that, even though downward traffic is less frequent in IoT environments, ALICE is able to manage this kind of traffic much more efficiently than MSF and E-OTF.

D. ONE-TO-ONE COMMUNICATION

In this final scenario we consider the case of device-to-device communication. This is a very common scenario in IIoT environments, where the source device may be a sensor sending a data packet to an actuator. In this scenario, the path followed by a packet includes both an upward component and a downward component. A packet sent by a source device is forwarded upward, along the RPL DODAG, until it reaches the root of the subtree including the destination node. Then, it is forwarded downward to the destination node, again following the DODAG.

In our experiments, we assumed from four up to eight device-to-device simultaneous communications (depending on the network size considered), with source and destination nodes located at opposite locations in the grid, so that packets have to travel upward from the source to the root and, then, downward from the root to the destination. For instance, with reference to the 4×4 grid shown in Figure 2, nodes 1, 2, 3, and 4 are the source nodes, while nodes 13, 14, 15 and 16 are the corresponding destinations. Each source node generates a

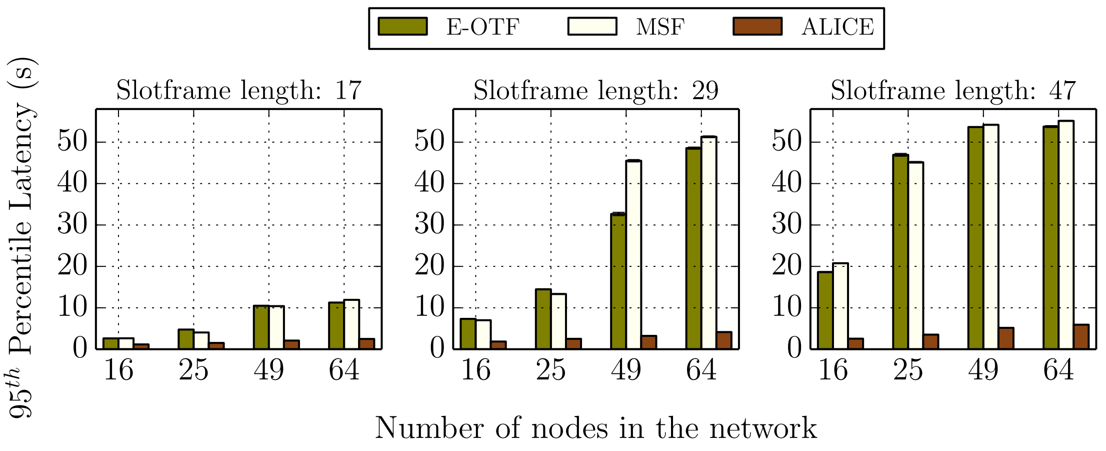

FIGURE 17. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. One-to-One Communication, device-to-device traffic. Packet generation period (*P*): 30s.

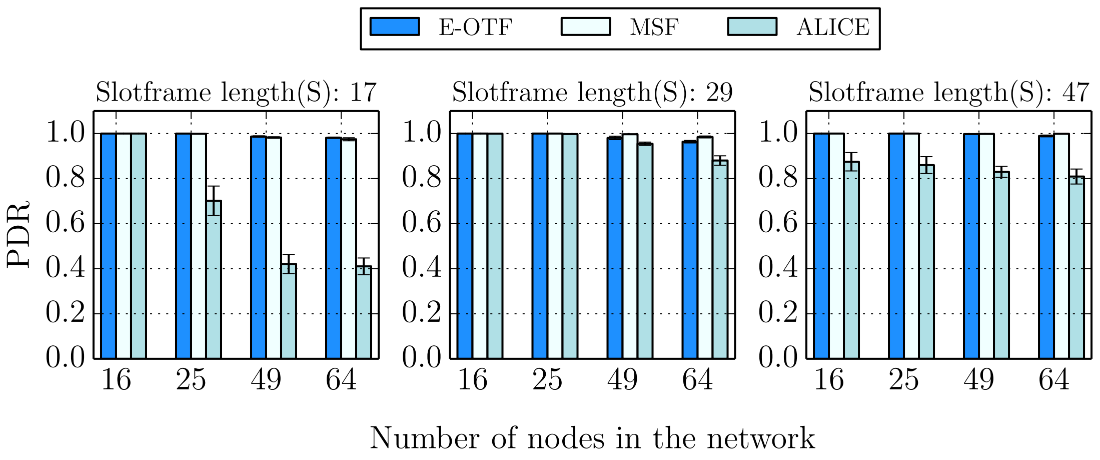

FIGURE 18. PDR for E-OTF, MSF and ALICE. Many-to-One Communication, CBR traffic. Frame Pending enabled. Packet generation period (*P*): 5s.

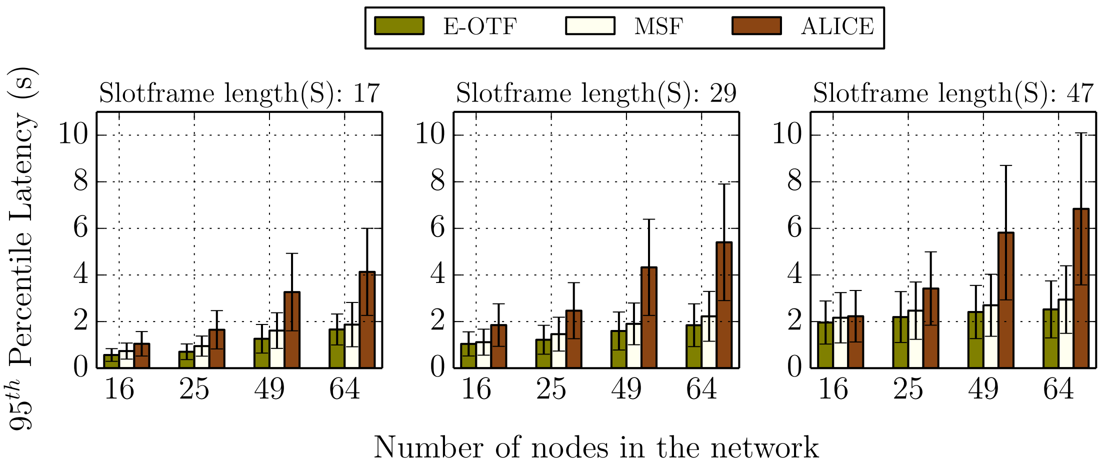

FIGURE 19. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Many-to-One Communication, CBR traffic. Frame Pending enabled. Packet generation period (*P*): 5s.

periodic flow of UDP packets, of fixed size (60 bytes), with a period P of 30s. As in the previous scenarios, we consider increasing network sizes (from 16 to 64) and different slotframe lengths (17, 29, and 47).

Figure 16 and Figure 17 show the delivery ratio and the 95th percentile of end-to-end latency, respectively. In this scenario, all three considered SFs exhibit similar performance, both in terms delivery ratio and end-to-end latency, except when the number of nodes in the network is high (e.g., 64). In such a case, ALICE outperforms significantly both MSF and E-OTF. The delivery ratio is also influenced by the slotframe length, as a larger slotframe implies a lower number of cells available for transmissions, per time unit. In terms of end-to-end latency, the gap between ALICE and the other two SFs is even more apparent. These results can be easily interpreted, on the basis of the results presented in Section V-A and V-C, if we observe that, in this scenario, there is a mix of (periodic) upward and downward traffic. The low performance exhibited by MSF and E-OTF when the aggregate offered load increases, is due to the their inefficient management of downward traffic.

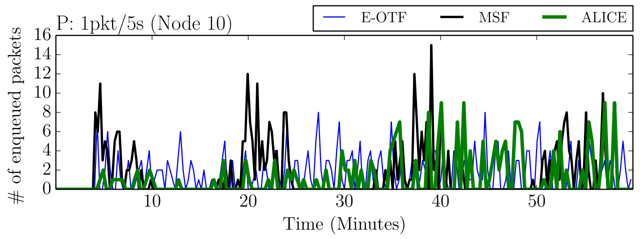

FIGURE 20. Queue occupancy of node 10 in the 8 × 8 grid topology. Many-to-One Communication, CBR traffic. Frame Pending enabled. Packet generation period (*P*): 5s. Slotframe length(*S*):29.

VI. INFLUENCE OF THE FRAME PENDING OPTION

In this Section, we evaluate the impact of using the *Frame Pending* (FP) option made available by the underlying TSCH protocol. This option is mainly beneficial to ALICE, as it provides a simple mechanism to adapt to traffic conditions without any negotiation overhead, which is in accordance with the philosophy of autonomous scheduling. However, since the FP option is a MAC layer mechanism, in the following experiments we will investigate its impact also on E-OTF and MSF. As above, we consider first scenarios with Many-to-One (i.e., upward) communication (with both CBR and bursty traffic), and then scenarios with One-to-Many (i.e., downward) and One-to-One (i.e., upward and downward) communication.

In the first scenario (**upward CBR traffic**), we consider the same parameter values used in Section V-A, in order to compare the results obtained with FP option enabled and disabled. Hence, we assume that the packet generation period is equal to 5 seconds, and $S = 17, 29,$ and 47. Figure 18 shows the packet delivery ratio for the three SFs, as a function of the network size, when the FP mechanism is enabled. Through a direct comparison with Figure 3, we can observe that the impact of FP on E-OTF and MSF is negligible and, in some cases, slightly negative (e.g., when $S = 29$, with a large number of nodes). Instead, the behavior of ALICE is a little bit more complex: the delivery ratio increases significantly when the slotframe is long enough ($S = 29$ and 47), while it tends to decrease when the slotframe is short ($S = 17$). In the former case, the improvement is due to the extra cells made available by the FP mechanism. This allows a node to achieve more bandwidth every time a backlog of packets occurs. This behavior is also confirmed by the analysis of the queue size over time. By comparing Figure 20 with Figure 5, we can see that the queue size with ALICE is now very low. Instead, both MSF and E-OTF exhibit a larger average queue, when using the FP option. This is because the FP mechanism, by allowing the transmission of packets on non-scheduled cells, may interfere with the basic adaptation mechanism (e.g., reducing the utilization and triggering the deallocation of some cells). Hence, the slight decrease in the delivery ratio of both MSF and E-OTF observed in some cases.

The different behavior of ALICE when $S = 17$ (with respect to $S = 29$ and 47), is due to the increased number of collisions caused by the additional transmissions made possible by the FP mechanism. Since the additional transmission of a node is performed in the next timeslot (using the same channel offset), if the corresponding cell is used simultaneously by a neighbor node, a collision occurs. Of course, the shorter the slotframe length, the higher the collision probability.

In terms of end-to-end latency, the benefit of using the FP option for ALICE is even much more apparent. From Figure 19, it emerges that ALICE still exhibits the largest delay, compared with E-OTF and MSF. However, if we compare these results with those in Figure 4 (with FP disabled), we can observe a dramatic decrease in the end-to-end latency of ALICE (from about 40 seconds to approximately 7 seconds, when $N = 64$ and $S = 47$). As above, MSF and E-OTF are not influenced in a significant way.

Finally, in terms of duty cycle (Figure 21), ALICE still exhibits the lowest value. However, as expected, the FP option increases the duty cycle and reduces the gap with respect to E-OTF and MSF.

We now analyze the scenario with **upward bursty traffic**. Figure 22 and Figure 23, show the delivery ratio and the 95th percentile of the end-to-end latency, respectively, in this scenario. A comparison with Figure 10 and Figure 11, allows us to observe that the FP option is now beneficial to all the considered SFs, as it helps in managing the sudden change in the traffic pattern of nodes. As above, ALICE is the one the benefits the most. For instance, when the network size is 64, the delivery ratio of ALICE increases from about 50% to more than 85%, while the end-to-end latency drops from around 25 seconds to less than 7 seconds. Despite that, ALICE has the worst performance also in this scenario.

The results related to the third scenario (i.e., **downward traffic**) are shown in Figure 24 (delivery ratio) and Figure 25 (end-to-end latency). They must be compared with the results in Figure 14 and Figure 15, respectively, obtained, under the same conditions, with the FP disabled. As above, using the FP option is beneficial to all the considered SFs, as they improve their performance. However, the overall situation remains unchanged, with ALICE outperforming both MSF

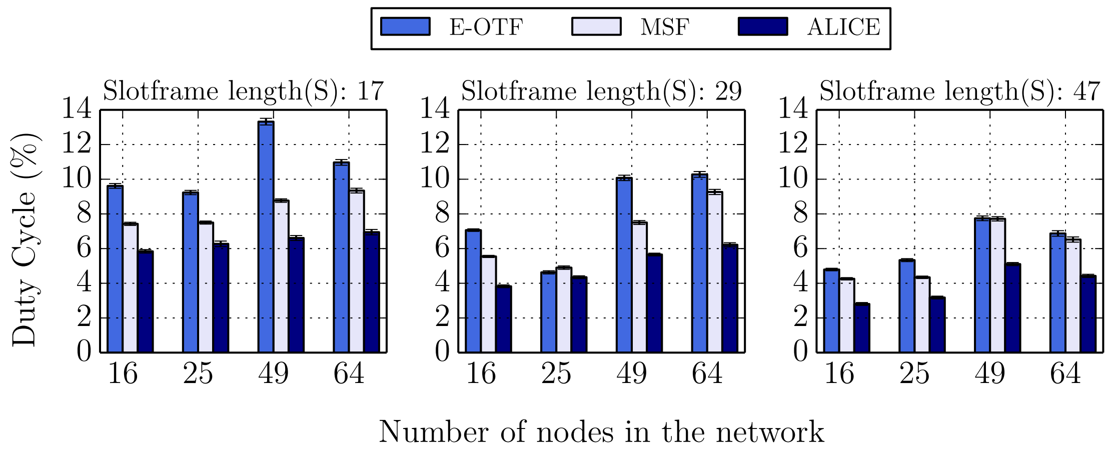

FIGURE 21. Duty cycle of the most loaded node for E-OTF, MSF and ALICE (Node 6 for $N = 16$, node 7 for $N = 25$, node 9 for for $N = 49$ and node 10 for $N = 64$). Many-to-One Communication, CBR traffic. Frame Pending enabled. Packet generation period (P): 5s.

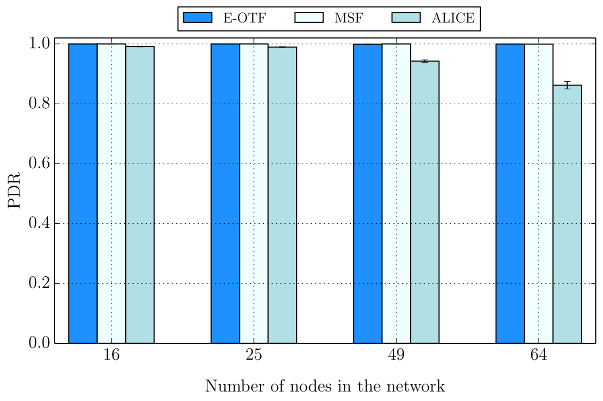

FIGURE 22. PDR for E-OTF, MSF and ALICE. Many-to-One Communication, Bursty traffic. Frame Pending enabled. Slotframe length(S): 29.

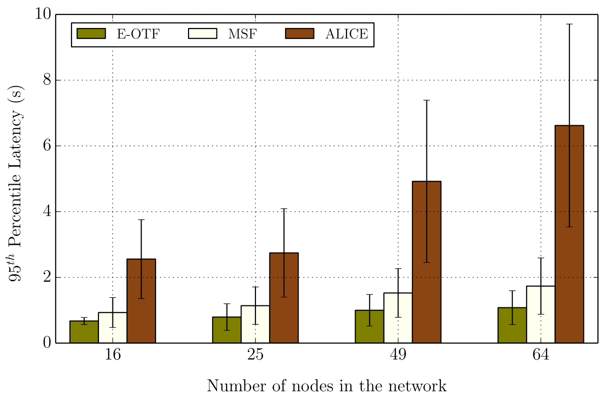

FIGURE 23. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Many-to-One Communication, Bursty traffic. Frame Pending enabled. Slotframe length (S): 29.

and E-OTF. The delivery ratio provide by ALICE is now close to 100% in all the considered configurations.

Finally, the results obtained in the scenario with device-to-device communication (**upward+downward traffic**) are shown in Figure 26 and Figure 27. A rapid comparison with the corresponding results in Figure 16 and Figure 17, respectively, allows us to confirm the same trend observed in the previous scenarios. The FP option improves the performance of all the three SFs (more or less), however, the overall situation remains unchanged, with ALICE outperforming both MSF and E-OTF, especially when the network size and, hence, the overall traffic increases.

In conclusion, the FP option is very beneficial to ALICE, as it provides a simple adaptation mechanism, without introducing additional negotiation overhead, and extends the number of scenarios where ALICE provides acceptable performance. Instead, the same mechanism has only slight (in some cases, even negative) effects on the performance of MSF and E-OTF.

VII. EXPERIMENTAL RESULTS

As anticipated, we also performed a set of experimental results using the PINT testbed available at our department (the map of node locations is shown in Figure 28), in order to validate the previous simulation results and analyze the performance of the considered SFs in a real environment. Since experiments in a real testbed are more difficult to carry out, our experimental analysis only includes a subset of the experiments analyzed through simulation. Specifically, we considered only the scenario with upward (periodic) traffic as it is the one for which the 6TiSCH architecture and the SFs are optimized. The slotframe length is equal to 29, and the Frame Pending option is enabled. The packet generation period ranges from 0.5 to 10 seconds.

The results obtained are summarized in Figure 29 (Packet Delivery Ratio), Figure 30 (95th percentile of the end-to-end latency), and Figure 31 (duty cycle). They confirm the previous simulation results under the same conditions (see Figures 18, 19, 21 for comparison).

In terms of delivery ratio (Figure 29), the experimental measurements confirm that, while E-OTF and MSF guarantee

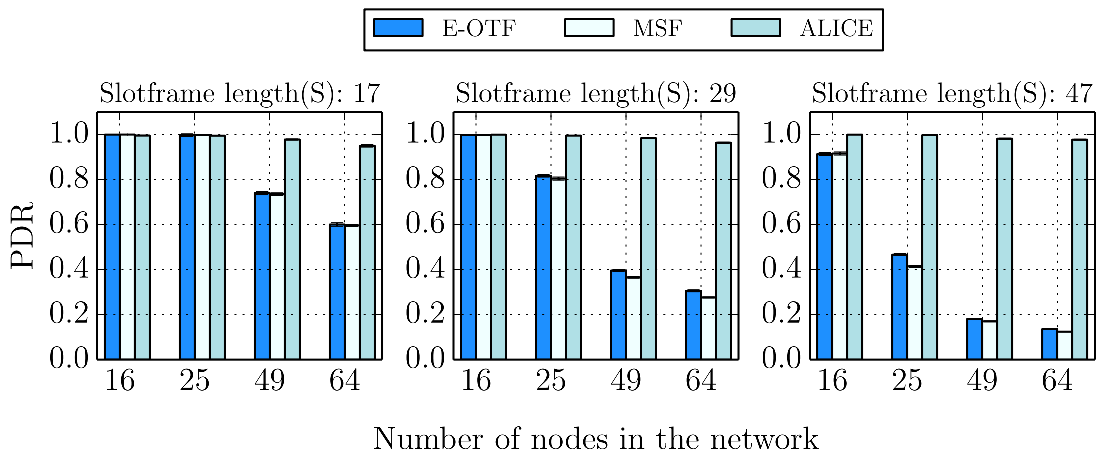

FIGURE 24. PDR for E-OTF, MSF and ALICE. Frame Pending enabled. One-to-Many Communication, Downward traffic. Packet generation period (*P*): 30s.

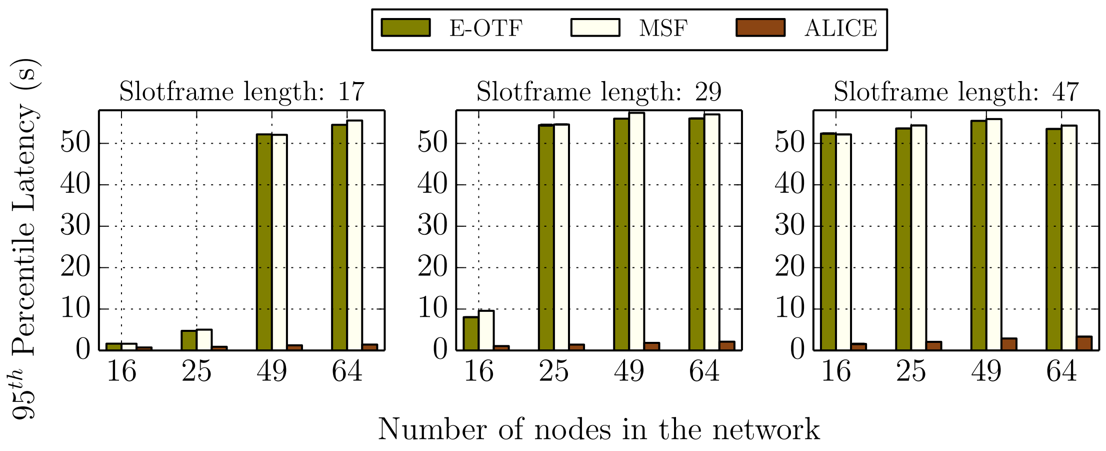

FIGURE 25. 95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Frame Pending enabled. One-to-Many Communication, Downward traffic. Packet generation period (*P*): 30s.

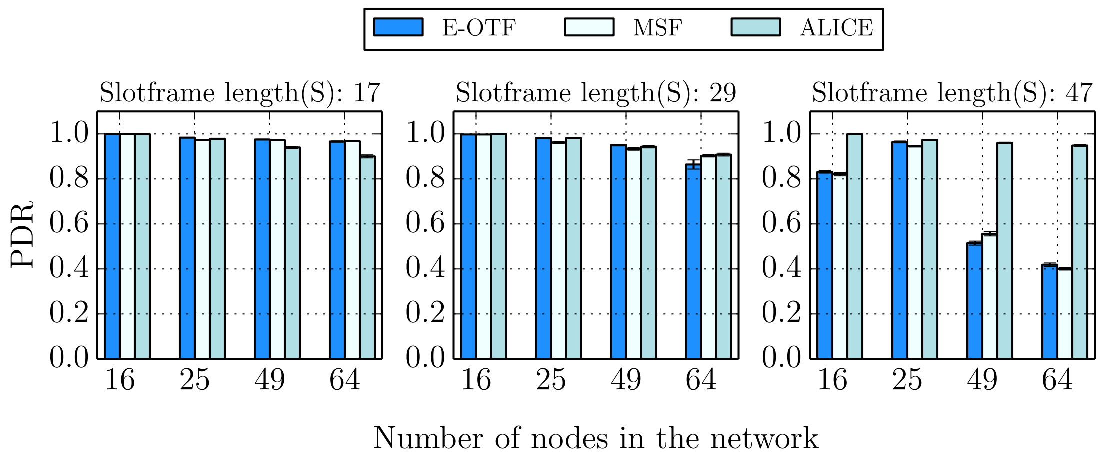

FIGURE 26. PDR for E-OTF, MSF and ALICE. Frame Pending Enabled Device to device traffic. Packet generation period (*P*): 30s.

a very high reliability with all the considered traffic rates, ALICE (with Frame Pending bit enabled) provides a good reliability only when the traffic rate is not so high. Similarly, in terms of end-to-end latency (Figure 30), ALICE exhibits the worst performance. Finally, Figure 31 confirms that ALICE has the lowest duty cycle among the three considered

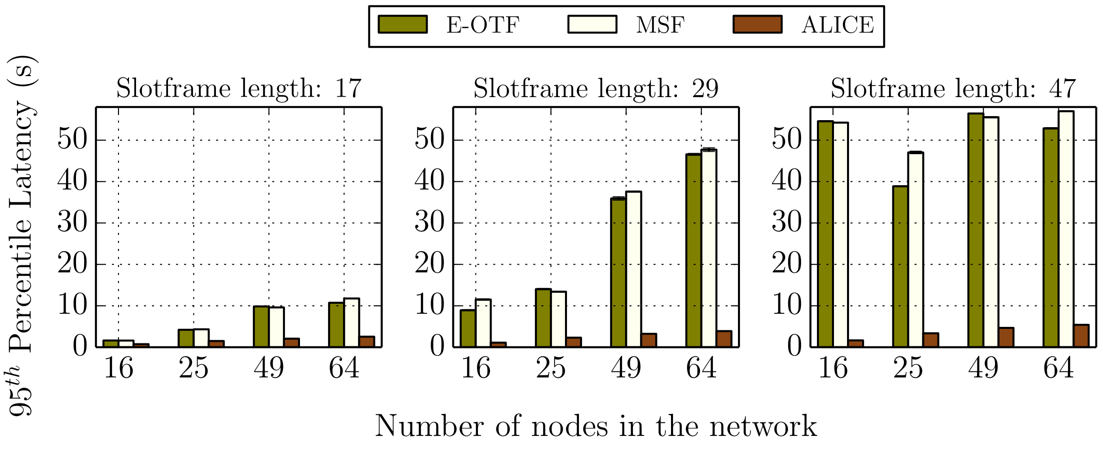

FIGURE 27. **95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. Frame Pending Enabled. Device to Device traffic. Packet generation period (*P*): 30s.**

SFs, as it allocates a lower number of cells. However, at high traffic rates, this results in low reliability and high latency, as shown in Figure 29 and Figure 30.

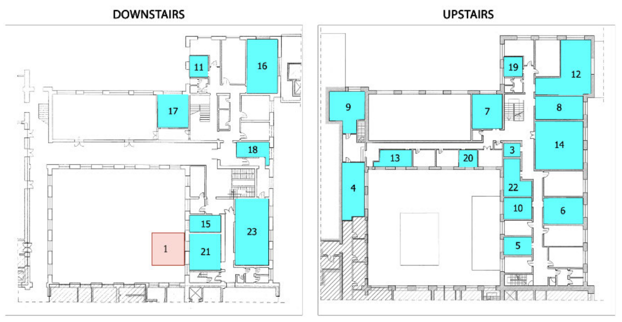

FIGURE 28. **PINT testbed map.**

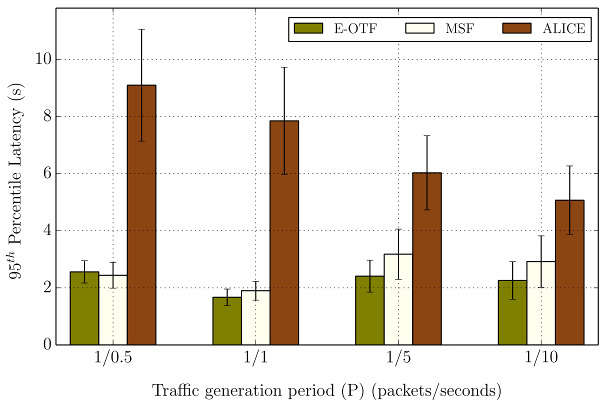

FIGURE 30. **95th Percentile of the end-to-end latency for E-OTF, MSF and ALICE. PINT testbed. Many-to-One Communication, CBR traffic. Slotframe length(*S*): 29.**

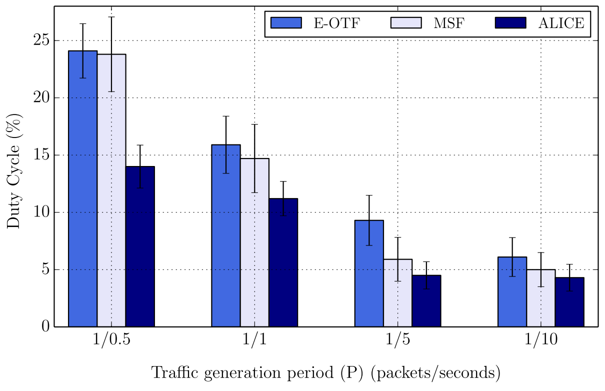

FIGURE 31. **Duty cycle of node 3 for E-OTF, MSF and ALICE. PINT testbed. Many-to-One Communication, CBR traffic. Slotframe length(*S*): 29.**

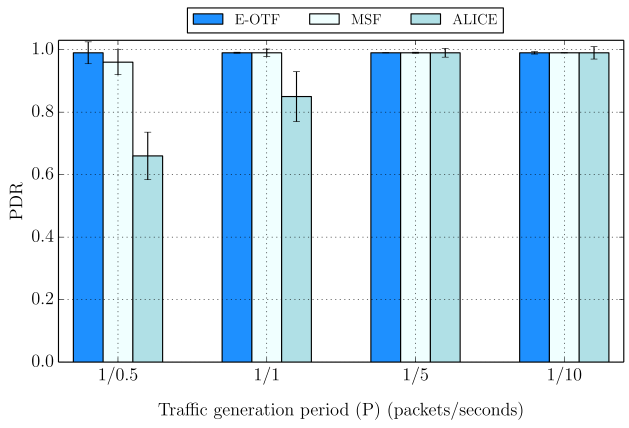

FIGURE 29. **PDR for E-OTF, MSF and ALICE. PINT testbed. Many-to-One Communication, CBR traffic. Slotframe length (*S*): 29.**

VIII. LESSON LEARNED

The results obtained through our performance comparison, allows us to summarize below some lessons we have learned about the behavior of the considered SFs, and draw a set of guidelines for selecting the most appropriate SF and parameter setting, depending on the use case.

In scenarios characterized by many-to-one communication (i.e., **upward traffic**), ALICE is the optimal choice in networks with moderate offered load (i.e., networks with small size and/or limited traffic rate). In these scenarios, it provides acceptable reliability and end-to-end latency, with low duty cycle and without negotiation overhead (which contributes to increase the network lifetime). In addition, if used with the FP option enabled, ALICE can also moderately adapt to unpredictable changes in the traffic pattern. However, its performance is strongly influenced by the slotframe length (S). This parameter must be set appropriately, in order to reach the best compromise between latency and duty cycle. Specifically, a short slotframe reduces the end-to-end latency and increases the delivery ratio, at the cost of an increased duty cycle.

When the aggregate offered load is high (e.g., due to the large number of nodes and/or high traffic rate) E-OTF and MSF are the only available options, as they perform very well, in terms of both reliability and end-to-end latency. This is especially true when the traffic and network conditions change over time, thanks to their ability to adapt the current schedule to the new conditions. This typically comes at the cost of a higher duty cycle. In addition, we need to consider the overhead introduced by the 6P protocol for negotiations.

When the traffic conditions are very dynamic, as in scenarios with **(upward) bursty traffic**, ALICE is not a suitable option, while E-OTF outperforms MSF, in terms of reliability and, above all, end-to-end latency, especially when the offered load is high [17]. This is because E-OTF is more reactive than MSF and can adapt more rapidly to changes in traffic and network conditions (with MSF, cells are allocated and deallocated one at a time). The drawback is an increased duty cycle, mainly due to the larger negotiation overhead, since E-OTF typically issues more 6P transactions than MSF.

The situation is completely different in scenarios characterized by one-to-many communication (i.e., **downward traffic**). In such a case, ALICE (possibly with FP option enabled) is always the best option, as it is able to manage this kind of traffic much more efficiently than MSF and E-OTF. The latter SFs are optimized for upward traffic, as downward traffic is not so frequent in IoT environments.

Also in scenarios characterized by a mix of **upward and downward traffic**, as in device-to-device communications, ALICE (possibly with FP option enabled) is, overall, the best option. This can be easily understood, as this case is combination of the previous ones.

Finally, the **FP option** is very beneficial to ALICE, especially in scenarios with upward traffic, as it provides additional transmission opportunities and allows to manage unpredictable traffic changes, thus making ALICE adaptive. In practice, this simple mechanism extends the set of configurations where the adoption of ALICE is convenient and makes ALICE with FP option the best candidate for many real-world use cases. Instead, the FP mechanism does not provide any significant improvement in the performance of MSF and E-OTF, as they are intrinsically adaptive. In some cases, the FP option may also deteriorate their performance, as it conflicts with the adaptation policy used by the SF.

IX. CONCLUSION

In this article, we have performed a detailed performance comparison, based on both simulation and experiments on a real testbed, of three Scheduling Functions (SFs) for 6TiSCH networks that take a different approach in scheduling cells for communication. Specifically, we have compared E-OTF that exploits a completely distributed approach, ALICE that takes an autonomous approach, and MSF that combines both autonomous and distributed scheduling. We have evaluated the above-mentioned SFs, in four different scenarios that are representative of many real-world use cases. We have also complemented our simulation experiments with a set of experimental measurements in a real testbed.

Our results have shown that there is no SF that outperforms the other ones in all the considered scenarios. Instead, different SFs exhibit pros and cons under different conditions. Hence, in Section VIII, we have provided a set of guidelines to select the best SFs, depending on the specific scenario and operating conditions. Moreover, this is also the policy of the 6TiSCH WG. They are currently standardizing MSF (Minimal Scheduling Function) as a reference SF for IIoT applications. However, multiple SFs are expected to be used in real deployments, in order to accommodate the requirements of different use cases.

In a nutshell, from our analysis it emerges that autonomous scheduling, as implemented by ALICE, is the best option for use cases characterized by downward or mixed (i.e., upward + downward) traffic, as it is able to manage downward traffic more efficiently than MSF and E-OTF. The latter ones are optimized for upward traffic, which is much more frequent in IIoT environments.

Indeed, in scenarios characterized by upward traffic, ALICE is the best option only when the aggregate offered load is low or moderate, as it has no negotiation overhead. If used with the Frame Pending option provided by TSCH, it can also adapt to moderate traffic changes. However, ALICE is unsuitable when the aggregate offered load becomes significant (e.g., high traffic rates and/or large number of source nodes), especially when the traffic and network conditions change over time. In these scenarios, distributed SFs (like MSF and E-OTF) are mandatory, thanks to their ability to adapt. The negotiation overhead is the cost to pay for adaptation. Among the considered adaptive SFs, E-OTF is more reactive than MSF and, hence, it is preferable in very dynamic scenarios (e.g., event-driven applications). The drawback is a slight larger duty cycle, mainly due to a higher negotiation overhead.

Our experiments have also highlighted that the performance of all the considered SFs is influenced by the RPL routing protocol. Specifically, RPL instabilities often result in path changes (i.e., selection of a new parent node) and, consequently, in increased delay and packet dropping. Instead, for Industrial IoT application communication reliability and timeliness are key requirements. In the future, we plan to investigate the interplay between SFs and RPL protocol and propose solutions for making RPL more stable.

ACKNOWLEDGMENT

The authors would like to thank Vincenzo Consales for his invaluable help in the implementation of the experiments. This work was supported in part by the Italian Ministry of Education and Research (MIUR) through the framework of the CrossLab Project (Departments of Excellence).

REFERENCES

[1] E. Sisinni, A. Saifullah, S. Han, U. Jennehag, and M. Gidlund, "Industrial Internet of Things: Challenges, opportunities, and directions," *IEEE Trans. Ind. Informat.*, vol. 14, no. 11, pp. 4724–4734, Nov. 2018.

[2] *IEEE Standard for Low-Rate Wireless Networks*, Standard 802.15.4-2015 Revision IEEE Std 802.15.4-2011, Apr. 2016.

[3] Y. Jin, P. Kulkarni, J. Wilcox, and M. Sooriyabandara, "A centralized scheduling algorithm for IEEE 802.15.4e TSCH based industrial low power wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf.*, Apr. 2016, pp. 1–6.

[4] Y. Jin, U. Raza, A. Aijaz, M. Sooriyabandara, and S. Gormus, "Content centric cross-layer scheduling for industrial IoT applications using 6TiSCH," *IEEE Access*, vol. 6, pp. 234–244, 2018.

[5] P. Thubert, M. R. Palattella, and T. Engel, "6TiSCH centralized scheduling: When SDN meet IoT," in *Proc. IEEE Conf. Standards for Commun. Netw. (CSCN)*, Oct. 2015, pp. 42–47.

[6] T. Chang, X. Vilajosana, S. Duquennoy, and D. Dujovne, "6TiSCH minimal scheduling function (MSF)," in *Proc. Internet-Draft Work Prog.*, Apr. 2020, pp. 1–5.

[7] F. Righetti, C. Vallati, G. Anastasi, and S. K. Das, "Analysis and improvement of the on-the-fly bandwidth reservation algorithm for 6TiSCH," in *Proc. IEEE 19th Int. Symp.*, Jun. 2018, pp. 1–9.

[8] M. Domingo-Prieto, T. Chang, X. Vilajosana, and T. Watteyne, "Distributed PID-based scheduling for 6TiSCH networks," *IEEE Commun. Lett.*, vol. 20, no. 5, pp. 1006–1009, May 2016.

[9] E. Municio, K. Spaey, and S. Latré, "A distributed density optimized scheduling function for IEEE 802.15.4e TSCH networks," *Trans. Emerg. Telecommun. Technol.*, vol. 29, no. 7, p. e3420, Jul. 2018.

[10] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana, "LLSF: Low latency scheduling function for 6TiSCH networks," in *Proc. Int. Conf. Distrib. Comput. Sensor Syst. (DCOSS)*, May 2016, pp. 93–95.

[11] M. R. Palattella, T. Watteyne, Q. Wang, K. Muraoka, N. Accettura, D. Dujovne, L. A. Grieco, and T. Engel, "On-the-Fly bandwidth reservation for 6TiSCH wireless industrial networks," *IEEE Sensors J.*, vol. 16, no. 2, pp. 550–560, Jan. 2016.

[12] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne, "Orchestra: Robust mesh networks through autonomously scheduled TSCH," in *Proc. 13th ACM Conf. Embedded Networked Sensor Syst.*, 2015, pp. 337–350.

[13] S. Kim, H.-S. Kim, and C. Kim, "ALICE: Autonomous link-based cell scheduling for TSCH," in *Proc. 18th Int. Conf. Inf. Process. Sensor Netw.*, Apr. 2019, pp. 121–132.

[14] S. Jeong, J. Paek, H.-S. Kim, and S. Bahk, "TESLA: Traffic-aware elastic slotframe adjustment in TSCH networks," *IEEE Access*, vol. 7, pp. 130468–130483, 2019.

[15] A. Karaagac, I. Moerman, and J. Hoebeke, "Hybrid schedule management in 6TiSCH networks: The coexistence of determinism and flexibility," *IEEE Access*, vol. 6, pp. 33941–33952, 2018.

[16] Q. Wang, X. Vilajosana, and T. Watteyne, *6Tisch Operation Sublayer (6top) Protocol (6p)*, document RFC 8480, 2018.

[17] F. Righetti, C. Vallati, S. K. Das, and G. Anastasi, "An evaluation of the 6TiSCH distributed resource management mode," *ACM Trans. Internet Things*, vol. 1, no. 4, pp. 1–31, Aug. 2020.

[18] A. Elsts, S. Kim, H.-S. Kim, and C. Kim, "An empirical survey of autonomous scheduling methods for TSCH," *IEEE Access*, vol. 8, pp. 67147–67165, 2020.

[19] P. Thubert, "An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4," *Proc. Internet-Draft Work Prog.*, 2019, pp. 2–8.

[20] F. Righetti, C. Vallati, S. K. Das, and G. Anastasi, "An experimental evaluation of the 6top protocol for industrial IoT applications," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, Jun. 2019, pp. 1–6.

[21] D. S. J. D. Couto, D. Aguayo, J. Bicket, and R. Morris, "A high-throughput path metric for multi-hop wireless routing," *Wireless Netw.*, vol. 11, no. 4, pp. 419–434, Jul. 2005.

[22] F. Osterlind, A. Dunkels, J. Eriksson, N. Finne, and T. Voigt, "Cross-level sensor network simulation with COOJA," in *Proc. 31st IEEE Conf. Local Comput. Netw.*, Nov. 2006, pp. 641–648.

FRANCESCA RIGHETTI received the master's degree in computer engineering from the University of Pisa, Italy, in May 2017, where she is currently pursuing the Ph.D. degree in information engineering. Her research interests include wireless sensor networks and their applications, the Internet of Things (IoT), and the Industrial Internet of Things (IIoT). She took part in the Project "ECOAP: Experimental assessment of congestion control strategies for the Constrained Application Protocol." She has served as a member of the Organization Committee for the International Conference IEEE SMARTCOMP 2018 and 2019.

CARLO VALLATI is currently an Assistant Professor (tenured) of computer systems engineering with the University of Pisa, Italy. He has been involved in different national and international projects, including in particular the FP7-ICT Project "BETaaS: Building the Environment for the Things as a Service." He has been the Principal Investigator of the Project "ECOAP: Experimental assessment of congestion control strategies for the Constrained Application Protocol," funded with a grant of 45 000 euro by the European Project WiSHFUL under the Fifth Open Call for experiments (WiSHFUL-OC5). He is also the Coordinator of the Cloud Computing, Big Data and Cybersecurity Crosslab, founded in the framework of the Departments of Excellence funded by the Italian Ministry of Education, University, and Research. He is a coauthor of more than 50 international publications. His research interests include wireless and sensor networks, protocols and platforms for the Internet of Things, algorithms, and architectures for edge/fog computing.

SAJAL K. DAS (Fellow, IEEE) was the Chair of the Computer Science Department, Missouri University of Science and Technology, from 2013 to 2017. He is currently a Professor of computer science and the Daniel St. Clair Endowed Chair with the Missouri University of Science and Technology. His research interests include wireless and sensor networks, mobile and pervasive computing, mobile crowd sensing, cyber-physical systems and the IoT, smart environments, distributed and cloud computing, and cyber security. He has published extensively in these areas with over 700 research articles in high-quality journals and refereed conference proceedings. He holds five U.S. patents and coauthored four books. His H-index is 85 with more than 31 500 citations according to Google Scholar.

Prof. Das was a recipient of ten Best Paper Awards in prestigious conferences like ACM MobiCom and IEEE PerCom, and numerous awards for teaching, mentoring, and research, including the IEEE Computer Society's Technical Achievement Award for pioneering contributions to sensor networks and mobile computing, and the University of Missouri System President's Award for Sustained Career Excellence. He serves as the founding Editor-in-Chief for *Pervasive and Mobile Computing* (Elsevier) journal; and an Associate Editor for several journals, including the *Journal of Parallel and Distributed Computing*, the IEEE Transactions on Dependable and Secure Computing, the IEEE Transactions on Mobile Computing, and the *ACM Transactions on Sensor Networks*.

GIUSEPPE ANASTASI is currently a Professor and the Head of the Department of Information Engineering (DII), University of Pisa, Italy. His scientific research interests include the Internet of Things, wireless sensor networks, smart cities/industries, and cybersecurity. He is a co-editor of two books: *Advanced Lectures in Networking* (LNCS 2497, Springer, 2002) and *Methodologies and Technologies for Networked Enterprises* (LNCS 7200, Springer, 2012). He has published more than 150 research articles in the area of computer networking and distributed systems. His publications have received about 9000 citations, according to Google Scholar (H-index = 40). He is currently leading the "CrossLab" Project, funded by the Italian Ministry of University and Research (MIUR), which aims at structuring six interdisciplinary and integrated laboratories (CrossLabs) for Industry 4.0 at the University of Pisa. He is also the past (and founding) Director of the CINI "Smart Cities" National Laboratory, a distributed laboratory consisting of 29 nodes (local labs) located at different Italian universities.

Dr. Anastasi serves as Steering Committee Member of the IEEE SMARTCOMP conference. He has served as the General Chair of IEEE SMARTCOMP 2018 and IEEE WoWMoM 2005; and the Program Chair of IEEE SMARTCOMP 2016, IEEE MSN 2015, IFIP/IEEE SustainIT 2012, IEEE PerCom 2010, and IEEE WoWMoM 2008. He has served as an Associate Editor for several journals, including *Pervasive and Mobile Computing*, *Sustainable Computing*, and *Computer Communications*.