A Novel Approach to Enhance the End-to-End Quality of Service for Avionic Wireless Sensor Networks

YEVHENII SHUDRENKO, DANIEL PLÖGER, KOOJANA KULADINITHI, and ANDREAS TIMM-GIEL, Institute of Communication Networks, Hamburg University of Technology

Going wireless is one of the key industrial trends, which assists the emergence of new manufacturing and maintenance processes by reducing the complexity and cost of physical equipment. However, the adoption of Wireless Sensor Networks (WSNs) in production environments is limited due to the strict Quality of Service (QoS) requirements of industrial applications. In particular, Wireless Avionics Intra-Communication (WAIC) systems operating in 4.3 GHz band are designed for intra-aircraft use cases with considerable restrictions on the transmission power of sensors, which results in multi-hop topologies, complicating a guaranteed QoS. The Internet Engineering Task Force (IETF) has developed the protocol stack *IPv6 over the Time Slotted Channel Hopping (TSCH) mode of IEEE 802.15.4* (6TiSCH) based on the IEEE 802.15.4 Standard for Low-Rate Wireless Networks, which combines the TSCH reliability with ubiquitous IPv6 connectivity and with the robust Routing Protocol for Low-Power and Lossy Networks (RPL). The Scheduling Function (SF) is a core IPv6 over the TSCH mode of IEEE 802.15.4 (6TiSCH) component, but the specification of the SF is an open research topic: numerous scientific articles investigated how QoS for a wide range of applications can be met by developing specialized SFs. However, no full-scale information exchange between the layers of the 6TiSCH stack was considered to optimize the SFs and to improve the network performance. In this work, we propose a novel solution named 6TiSCH-CLX to satisfy demanding QoS requirements using cross-layer communication. It is an extension of the 6TiSCH framework at the network and Medium Access Control (MAC) layers, addressing latency and reliability challenges agnostic of the physical layer. 6TiSCH-CLX is evaluated both analytically and in simulations for several safety-critical avionic intra-communication use cases in WAIC. Preliminary results indicate considerable improvements to latency, while maintaining almost 100% Packet Delivery Ratio (PDR) without retransmissions and they highlight the capability of the cross-layer approach compared to existing solutions.

CCS Concepts: • **Computer systems organization → Sensor networks**; **Reliability**; *Redundancy*; • **Networks → Network performance evaluation**; **Cross-layer protocols;**

Additional Key Words and Phrases: WAIC, 6TiSCH, MSF, WSN, IEEE 802.15.4, TSCH, IPv6, RPL, scheduling function, end-to-end QoS, avionics

This publication is funded by the BMWi as part of the *Retrofitbare Sensorsystem-Architektur für prädiktive Instandhaltung* (ReSA) project, for ultra-resilient, retrofittable wireless sensor networks for use in commercial aircraft, funding identifier 20X1721C.

Authors' address: Y. Shudrenko, D. Plöger, K. Kuladinithi, and A. Timm-Giel, Institute of Communication Networks, Hamburg University of Technology, Am Schwarzenberg-Campus 3, Hamburg 21073, Germany; emails: {yevhenii.shudrenko, daniel.ploeger, koojana.kuladinithi, timm-giel}@tuhh.de.

1533-5399/2022/11-ART95
https://doi.org/10.1145/3520441

ACM Reference format:
Yevhenii Shudrenko, Daniel Plöger, Koojana Kuladinithi, and Andreas Timm-Giel. 2022. A Novel Approach to Enhance the End-to-End Quality of Service for Avionic Wireless Sensor Networks. *ACM Trans. Internet Technol.* 22, 4, Article 95 (November 2022), 29 pages.
https://doi.org/10.1145/3520441

1 INTRODUCTION

Future **Internet of Things** (**IoT**) applications, such as connected Industry 4.0 applications, become more challenging with strict **Quality of Service** (**QoS**) requirements, including guaranteed reliability and delay. Several mechanisms in the communication stack to match the expected Quality of Service (QoS) are already discussed and specified at different layers, with the goal to make the communication more reliable. In this article, we mainly focus on guaranteeing the strict QoS requirements for wireless avionic applications.

1.1 Radio Systems for Avionic In-Cabin Communication

In modern passenger aircraft, a wide range of sensors, actuators, and switches are placed inside and outside the cabin for numerous system applications. Most of these devices, in particular the safety-critical ones, still rely on wired communication to the central onboard processing unit [23]. However, recently new application use cases emerge in monitoring, sensing and maintenance with high demands in configurability, flexibility, and safety of the network. One example is the German government-funded project ***Retrofitbare Sensorsystem-Architektur für prädiktive Instandhaltung*** (**ReSA**)which targets ultra-resilient, retrofittable wireless sensor networks for predictive maintenance in a commercial aircraft [27]. Such use cases involve significantly denser device network and thus a more complex, heavier and costlier electrical wiring than before, which draws more focus on connecting the devices by wireless communication.

On the other hand, wireless industrial sensor networks have seen serious progress over the recent years. While they originally did not facilitate safety-critical applications, modern use cases impose high demands in terms of tolerable latency and packet losses. This change is also reflected in the development of recent industrial standards. These include ISA100.11a [17], WirelessHART [6], and IPv6 over the **Time Slotted Channel Hopping** (**TSCH**) mode of IEEE 802.15.4 (6TiSCH), which is specified by the **Internet Engineering Task Force** (**IETF**) as an extension of IEEE 802.15.4 with increased robustness against radio interference [31]. They all utilize the physical layer of the IEEE Standard for Low-Rate Wireless Networks IEEE 802.15.4 and implement new methods of channel access for improved reliability. While ISA100.11a and WirelessHART mainly target industrial process automation, IPv6 over the Time Slotted Channel Hopping (TSCH) mode of IEEE 802.15.4 (6TiSCH) is a more general extension of IEEE 802.15.4 for deterministic low power wireless applications. All three industrial standards use dedicated time-slotted **Time-Division Multiple Access** (**TDMA**) with flexible time slot configuration and usage of multiple frequency channels.

In aviation, wireless sensor networks are currently being standardized as **Wireless Avionics Intra-Communication** (**WAIC**). Based on the IEEE 802.15.4 physical layer combined with Time-Division Multiple Access (TDMA) and channel hopping, these networks are supposed to complement and replace wired communication in aircraft to improve flexibility and reduce weight [18]. In contrast to the **Industrial, Scientific and Medical** (**ISM**) band used in IEEE 802.15.4, the **International Telecommunication Union** (**ITU**) assigned the frequency band between 4.2 and 4.4 GHz to Wireless Avionics Intra-Communication (WAIC) in 2015, so no interference from WLAN or Bluetooth is expected [12]. Instead, a part of the frequency band is shared with radio altimeters,

which are used for altitude measurements in an aircraft [14]. Additionally, the coexistence with similarly equipped aircraft in particular when they are in close proximity on the ground has to be considered [22].

1.2 6TiSCH with Cross-layer Information Exchange (6TiSCH-CLX)

This article presents an avionic sensor network operating in the WAIC frequency band using the 6TiSCH architecture [31], where 6TiSCH is extended to meet the strict requirements of safety-critical avionic applications such as smoke detection. The contribution is twofold: firstly, the cross layer framework utilizes the route knowledge of the **Routing Protocol for Low-Power and Lossy Networks** (**RPL**) at the link layer scheduling, ensuring end-to-end QoS-aware communication by exchanging information across different layers, in particular between application, network, and link layer. The proposed scheme, named 6TiSCH stack with cross-layer information exchange (6TiSCH-CLX), is implemented and evaluated in the event-based OMNeT++ simulator [25] with a full 6TiSCH stack including Routing Protocol for Low-Power and Lossy Networks (RPL) [1], TSCH and 6TiSCH Operation Sublayer (6top) Protocol (6P) [33]. The physical layer used is not IEEE802.15.4 but WAIC PHY, since this work focuses on evaluating avionic use cases specifically. However, 6TiSCH-CLX is agnostic of the underlying physical layer, since only information exchange between the **Medium Access Control** (**MAC**) layer (i.e., TSCH), the network layer (RPL) and the **Scheduling Function** (**SF**) is required.

Secondly, a mathematical model for a static multi-hop topology with the fixed scheduling is developed to analyze the target **Key Performance Indicators** (**KPIs**), such as the end-to-end delay and the **Packet Reception Probability** (**PRP**). Among existing solutions, the **Minimal Scheduling Function** (**MSF**) is chosen for benchmarking, as it is a relatively simple but robust scheduling approach close to being standardized [4]. The proposed analytical model allows to conduct different evaluations computationally fast, by varying parameters such as the number of hops and the link collision probability. The model provides bounds of the expected performance, which are valid universally, in contrast to simulation samples of limited runtime.

Three prominent scenarios, which may utilize the WAIC standard, are evaluated with a different number of nodes and application requirements to verify the proposal. The first scenario uses wireless sensors for seat belt status monitoring, which aperiodically sends high priority data. The second one is a smoke/fire alarm scenario with burst traffic of highest priority. And the third scenario is a cabin humidity monitoring system with periodic, low-priority data. Results show that low latency requirements can be met in safety-critical scenarios with 30 sensor nodes, while keeping a high **Packet Delivery Ratio** (**PDR**). In the large-scale case with up to 200 nodes, noticeable improvements are achieved with a low overhead cost for a number of metrics, including end-to-end delay, throughput and jitter.

2 BACKGROUND AND RELATED WORK

Over the years, an IPv6-enabled protocol stack named 6TiSCH has been standardized, incorporating TSCH, a Scheduling Function (SF) as well as RPL to manage communication in **Wireless Sensor Networks** (**WSNs**). Many works have investigated the scheduling problem and proposed a decent amount of both, centralized and decentralized SFs, focusing on link-layer improvements. This work aims at enhancing the QoS for Wireless Sensor Networks (WSNs) by utilizing communication between the layers of the 6TiSCH stack. This provides a reliable, latency-optimizing schedule to support time- and reliability-critical avionic applications.

An overview of the 6TiSCH stack is given first, focusing on the TSCH medium access control and the RPL protocol, both being essential components of the proposed cross-layer framework. We also highlight all related works focused on improving the performance of 6TiSCH protocol stack.

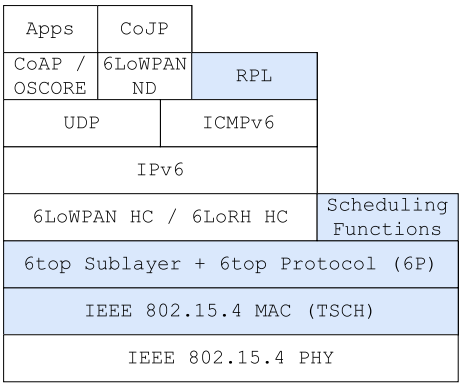


Fig. 1. IETF 6TiSCH Protocol Stack [31].

2.1 The 6TiSCH Protocol Stack

The primary goal of **IPv6 over the TSCH mode of IEEE 802.15.4** (**6TiSCH**) is to enable IPv6 in constrained, low-data rate WSNs, by further extending the TSCH mode of IEEE 802.15.4 with management capabilities, in order to satisfy the stringent QoS requirements of industrial applications. 6TiSCH defines the network architecture shown in Figure 1, which consists of the following layers:

(1) TSCH with the corresponding IEEE 802.15.4 physical layer.
(2) The 6P, an interface between TSCH and the SF. It defines the ADD, DELETE and RELOCATE commands which can be exchanged by nodes in 2- or 3-step negotiated transactions in order to setup and modify node scheduling.
(3) **IPv6 over Low-Power Wireless Personal Area Networks** (**6LoWPAN**) and **User Datagram Protocol** (**UDP**) header compression.
(4) IPv6 on the network layer.
(5) RPL and SF for routing and resource allocation.
(6) User Datagram Protocol (UDP) as a transport layer protocol.
(7) **Constrained Join Protocol** (**CoJP**) as a basis for the application layer and for synchronization with the 6TiSCH network.

2.1.1 IEEE 802.15.4 TSCH. IEEE 802.15.4 is widely used in WSNs mainly due to the low price of suitable transceivers, their low power consumption and their small size. It operates in the 2.4 GHz Industrial, Scientific and Medical (ISM) frequency band with 16 channels and 5 MHz spacing between each channel. The ISM band is also used by other devices, e.g., WLAN and Bluetooth. TSCH was introduced as an amendment to the IEEE 802.15.4 standard, featuring a novel medium access scheme. TSCH utilizes both, time and frequency division for reliable and energy-efficient communication in a scheduled manner. A TSCH network reduces the probability of packet errors caused by external interference compared to contention-based protocols. In order to reduce the collision probability, TSCH chooses a different frequency channel for each transmission by using a predetermined sequence. This process is called *channel hopping*. Furthermore, the channel access of transmissions is strictly deterministic because of the TDMA mechanism.

Figure 2 shows an example of the TSCH schedule. This *slotframe* is defined as a fixed number of timeslots, which repeats periodically during the network lifetime. The number of timeslots in a slotframe is called **Slotframe Length** (**SFL**). The timeslot duration in TSCH includes the

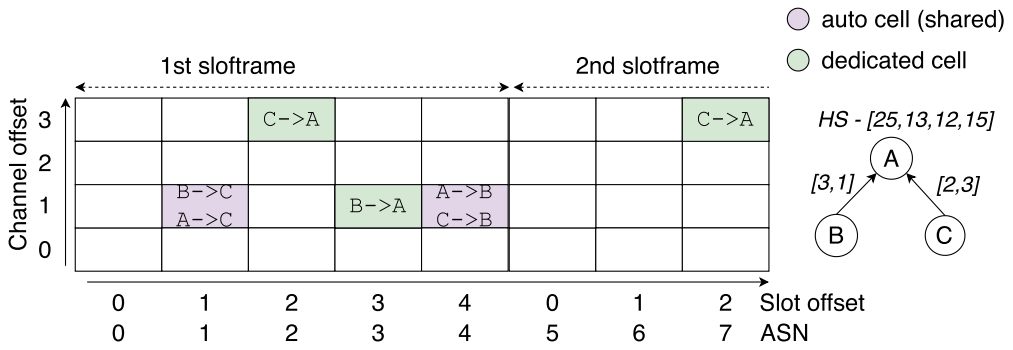


Fig. 2. An example of a TSCH schedule for an $SFL = 5$ and $channelOffset = 4$.

transmission of a single frame and the reception of the corresponding acknowledgment. The recommended value from the standard is 10 ms.

The communication between nodes in a network with TSCH occurs within a *cell*, which is defined by the tuple [*slotOffset*, *channelOffset*]. The first element of the tuple indicates the position within the slotframe and the second constitutes a deviation from the standard hopping sequence. While conventional TDMA systems assign a timeslot for the communication of one node pair in the network at a time, in a TSCH network distinct node pairs can communicate on different channels at the same time by using differing channel offsets.

When a node joins a TSCH network, it synchronizes with the network by the reception of an **Enhanced Beacon** (**EB**) message, which carries information necessary to operate in this network:

— Timeslot duration.
— SFL.
— Minimal shared cell(s) for control traffic.
— The current **Absolute Slot Number** (**ASN**) and the **Hopping Sequence** (**HS**).

The Hopping Sequence (HS) defines a pattern of channels to be used in the network and the Absolute Slot Number (ASN) indicates the total number of timeslots elapsed since the network was deployed. By taking into account the ASN, the HS, and the corresponding channel offset, each node computes an actual frequency channel f to be used for transmission in the current cell at [*slotOffset*, *channelOffset*] as follows:

$$f = (ASN + channelOffset) \bmod hoppingSequenceLength. \tag{1}$$

Cells can be assigned to a communication between multiple nodes (*auto* or *shared*) or a communication between two specific nodes (*dedicated*). Auto cells are calculated using a hash function of a node Medium Access Control (MAC) address, which allows neighbors to exchange messages without any prior scheduling. Depending on their purpose, cells can also be referred to as *TX* or *RX* for data transmission/reception, respectively. Figure 2 shows a sample network of three nodes, where *Node B* can communicate with *Node A* in cell [3, 1] and *Node C* can communicate with *Node A* in cell [2, 3]. This network uses a hopping sequence of $HS = [25, 13, 12, 15]$ and a slotframe length of $SFL = 5$. We assume that *Node C* joins when $ASN = 3$ and *Node B* joins afterwards at $ASN = 4$. *Node C* computes the channel f in its cell [2, 3] for the first slotframe according to Equation (1). *Node C* will transmit to *Node A* in the frequency channel $ch = 12$, because the frequency channel is computed as $f = (3 + 3) \bmod 4 = 2$ and with a zero-based indexing of the HS, 2 corresponds to the third HS entry, i.e., $ch = 12$. If the length of the HS is not a common multiple of the slotframe length, a cell with a fixed channel offset will use a different actual channel for transmission

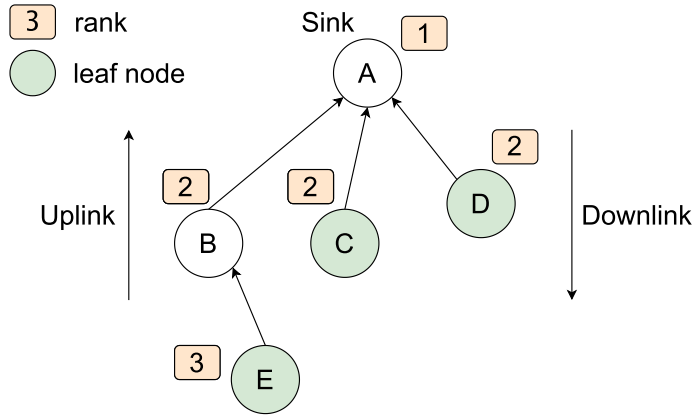


Fig. 3. RPL Destination-Oriented Directed Acyclic Graph (DODAG) example (the hop count is used as an Objective Function metric).

in each slotframe. This mitigates the impact of external interference on a single frequency band across multiple transmissions. Following this example, *Node C* will then use channel $ch = 15$ for transmission in the second slotframe.

2.1.2 RPL. An RPL is the proposed standard for low-power, constrained networks operating under IPv6. It is a proactive, distance-vector routing protocol [1, p. 19], organizing the network topology in a hierarchical tree-like structure called **Destination-Oriented Directed Acyclic Graph** (**DODAG**), as shown in Figure 3. All links are directed towards the sink, so that sensor nodes forward their data *upwards*, while control messages can also travel from the sink *downwards* the Destination-Oriented Directed Acyclic Graph (DODAG). RPL control messages include:

- The ***DODAG Information Object (DIO)*** is a broadcast message used to disseminate DODAG control information across the network. Each RPL-enabled node, when first receiving a DODAG Information Object (DIO), may decide to join an advertised DODAG instance, which is identified by a *DODAG-ID*. The nodes calculate their *rank* during the joining process and choose a *preferred parent* by using the **Objective Function** (**OF**). The rank represents an abstract distance to the sink in terms of the chosen evaluation metrics, such as **Expected Transmission Count** (**ETX**), latency, throughput, etc. The preferred parent is a neighbor with the lower rank, i.e., closer to the sink, chosen as the next hop for uplink data forwarding.
- The ***Destination Advertisement Object (DAO)*** is required to enable route discovery and maintenance. Each node advertises itself as a possible destination to the sink, which then uses this data to gain a full network topology overview.
- The ***DODAG Information Solicitation (DIS)*** is used by a joining node as the means to manually request a DIO transmission containing relevant DODAG control information.

RPL supports two modes of operation: *storing* and *non-storing*. In the former, each node keeps routing information about its relative sub-DODAG and thus it is capable of performing the shortest path selection on its own. In the non-storing mode, routing in the network is only possible by using a source-routing header which is constructed by the sink. DIOs are broadcasted periodically and the inter-transmission time is regulated by the *Trickle Timer* mechanism [21]. This ensures that less control packets are sent in a stable network, while enough packets are exchanged during the network bootstrap or repair operations to secure a fast topology convergence.

2.1.3 SF. The 6TiSCH SF is a flexible submodule, which encapsulates an algorithm for dynamic schedule construction and management according to a specific set of goals. The SF itself may be run locally by each node (distributed) or remotely at the sink (centralized) using the collected information. An appropriate SF is required per use case to achieve an optimal network performance according to the target metrics, e.g., low latency and low energy consumption. A number of SFs have been proposed by both the IETF and the scientific community in general, trying to find a balance between complexity, robustness and speed. A more detailed review of existing SFs of centralized as well as of distributed scheduling is presented in Table 1.

2.1.4 6top Sublayer. The 6top sublayer [33] was introduced as schedule management interface for distributed scheduling in TSCH networks. This sublayer is regulated by the 6P protocol in a request-response manner to negotiate which cells to use on a communication link between two neighbors. The SF triggers the addition and deletion of cells via the 6P protocol, e.g., depending on the offered traffic on a link or whether new neighbors join the network.

2.2 Related Work — The 6TiSCH Protocol Stack

This section highlights how QoS has been achieved by previous works in the context of 6TiSCH [31]. Some focus on enhancing the SF using centralized or distributed approaches, while others target RPL-specific improvements.

Centralized vs Distributed SF: The centralized approaches [11, 19, 24, 26] allow close to optimal schedules which satisfy the most strict QoS requirements, but such schedules get outdated relatively fast, triggering an expensive recalculation routine. While distributed solutions [3, 5, 8–10, 13, 15, 29] may provide no strict performance guarantees, they are far more robust to dynamic network changes and achieve satisfying practical performance in most cases. One of the concerns for decentralized approaches is their implementation overhead with respect to the default 6TiSCH framework. While some can be considered more lightweight, as they reuse existing mechanisms and scheduling functions such as [8, 15, 20], the others require sophisticated configuration and testing like [9, 29] due to an extended set of control messages, slotframe managing tools, and so on.

SF Enhancements from RPL: Only a few publications use the capabilities provided by RPL for TSCH schedule construction [3, 5, 15, 20]. In particular, [15] tries to split the network into blocks with disjoint portions of the slotframe assigned to them, but does not propose a solution to manage these sub-schedules, while LLSF [5] focuses solely on end-to-end delay by employing a greedy daisy-chaining approach without extensive evaluations. DeTAS [3] utilizes RPL topological information to construct a collision-less schedule, but assumes that the RPL nodes operate in the storing mode by default and are, therefore, able to perform micro-scheduling on their own. In [20], an interesting combined approach is introduced that actively uses the RPL topology to schedule a dedicated, guaranteed cell for each node to communicate with its preferred parent. Moreover, this allows satisfying end-to-end delay requirements by adjusting how often a node is allowed to access the slotframe. However, the scheme doesn't scale well for networks consisting of several large subtrees rooted at the sink. These sub-trees would apply the same algorithm completely in parallel, agnostic of each others' schedules, which could substantially increase the collision probability.

Dynamic Adaptation to Traffic Demand: One of the major concerns during the scheduling process is to decide how many cells to be allocated initially per node pair and how the cell allocation reacts to the traffic demand. The decision can be taken depending on the application and the control traffic demands (e.g., DIO/Destination Advertisement Object (DAO) messages). In avionic sensor networks with a diverse set of applications and non-deterministic traffic, we expect highly dynamic resource allocation demands. For this reason, we focus on the reactive allocation

Table 1. Comparison of the Related Work

Ref	Scheduling Methods	Targeted QoS	Evaluation
MSF [4]	*Distributed*, cell monitoring and dynamically addition and deletion of cells using 6P.	Only Packet Delivery Ratio (PDR) as a cell reallocation is triggered after a certain packet is lost at the link layer.	A proposal for the SF for 6TiSCH by the IETF. MSF functionalities are evaluated mostly for delays.
Enhanced MSF [13]	*Distributed*, nodes calculate the average traffic rate (T) at every *X* slotframes and compare it to the number of allocated cells (N) with its preferred parent.	Only PDR. Schedule negotiation overhead is reduced, as it adapts to the queue size quicker than MSF by predicting the amount of traffic for every slotframe and allocating the number of cells accordingly, which results in lower delays for larger scenarios.	Only simulations, 2–100 nodes, Key Performance Indicators (KPIs): 6P negotiation errors, packet overhead, latency and average queue size.
Overhearing of 6P, [10]	*Distributed*, nodes overhear 6P transactions exchanged by their neighbors on *Auto* cells. This approach depends on the number of *Shared* cell allocation to overhear all 6P messages.	Minimizes the packet collisions by guaranteeing the use of non-overlapping cells by neighbors.	Only simulations, 100 nodes with 1 packet per slotframe, KPIs: number of colliding packet and *TX* cells.
DeTAS [3]	*Distributed*, maintains the micro schedule depending on the queue size of 1-hop neighbors.	Potential PDR and latency improvements, but not presented in the results.	Analytical evaluation verified on a Python simulator, Scenarios with 60, 80, and 100 nodes, only the queue size is shown.
WAVE [29]	*Distributed*, manages local schedules by gathering data from neighbors and taking queue sizes into account.	Minimizes collisions, delay optimization at the cost of higher signaling overhead.	Simulations using a general-purpose programming toolbox to estimate slotframe utilization with 10 to 100 nodes.
TASA [26]	*Centralized*, two phases—"Matching and Coloring" to define the schedule centrally based on the heuristic incremental method, ideal for homogeneous traffic.	PDR and the delay may be guaranteed.	Python simulator, Scenarios 20 to 80 nodes, KPIs: throughput, single hop delay and power consumption.
AMUS [19]	*Centralized*, cells are allocated based on the link traffic load, free cells are assigned to facilitate retransmissions, an End-of-Queue control message allows nodes to delete retransmission cells adaptively.	Improved reliability by allocating cells for retransmissions.	Simulations, 20 nodes, KPIs: delay, PDR, cell utilization and % of radio duty cycle.
6TiSCH-CLX	*Hybrid*, MSF features are used (distributed) together with daisy chaining cell allocation over a multi-hop network (centralized).	PDR by maintaining frequency-disjoint RPL branches, end-to-end delay through cell daisy-chaining.	Simulations and analytical, Avionic scenarios (30 to 200 nodes), KPIs: PDR, latency, throughput, jitter.

of cells in this work. In addition to the Minimal Scheduling Function (MSF) [4], a reactive cell allocation that matches traffic requirements is proposed in Orchestra [9]. MSF uses 6P transactions to add/remove cells if the link layer cell utilization surpasses a given threshold. Orchestra distinguishes traffic by its target layer—control (MAC, routing) or application (data)—and maintains distinct slotframes with non-overlapping periods for each traffic type. This improves the deterministic nature of TSCH even further, allowing each slotframe type to be managed separately in terms of its length, periodicity and slot types. However, it is difficult to achieve the optimum cell allocation over a multi-hop network, as each node closer to the sink may have a *Transmit (TX) cell* scheduled to its parent before a *Receive (RX) cell* is scheduled from its child. Orchestra has been evaluated in several publications for different scenarios, mostly using the *Cooja Simulator*.

Summary — Related Work: Table 1 summarizes some of the most prominent works with regards to enhancing the 6TiSCH protocol stack. The column *Scheduling Methods* refers to the schedule type and maintenance routines. The column *Targeted QoS* indicates how the proposed enhancements to the 6TiSCH improve the application performance. In the column *Evaluation* we summarize the **Key Performance Indicators** (**KPIs**) of its evaluation.

2.3 Problem Statement

Table 1 shows that the distributed TSCH scheduling algorithms mostly rely on the basic 6P random cell selection without requiring the complete knowledge about the network topology. We believe that especially in stationary scenarios there is a missed opportunity to enhance end-to-end QoS further by interacting with the upper layers to achieve a sophisticated scheduling, based on the

Table 2. Physical Layer Parameters - 802.15.4 vs WAIC

Parameter	802.15.4	WAIC
Center frequency	2.4 GHz (ISM band)	4.2 GHz to 4.4 GHz (licensed band)
Number of channels	16 (defined as 11 ... 26)	40 (defined as 0 ... 39)
Channel bandwidth	2 MHz	2.6 MHz
Default TX power	0 dBm (mostly used)	−29 dBm/MHz [28]
Communication range (depends on the TX power)	>10 m	3–4 m

logical topology of the network. There are widely used SFs in previous work such as MSF and Orchestra. The MSF is chosen for a comparison in this work as it is the standard scheduling function proposed by the IETF for the 6TiSCH framework. As avionic WSNs should also cater for a higher number of nodes (at least up to 200—300 nodes), the recommended Slotframe Length (SFL) of 101 slots is chosen and has to be utilized carefully to minimize the delay for emergency scenarios like *Smoke Detection*. On the other hand, air quality sensors like *Humidity Monitoring* systems could tolerate higher delays. This shows the need of flexible and adaptive scheduling for different traffic requirements.

As a result, we propose a hybrid approach where the slot offset and the channel offset are determined based on the RPL topology to achieve a close-to-optimal cell allocation, minimizing the delays while keeping PDR high. The cross-layer framework proposed in this work, further referred to as 6TiSCH-CLX, relies on the interaction between core components of the 6TiSCH stack to meet industrial-grade QoS in a lossy, constrained wireless sensor network. The WAIC PHY is used instead of the IEEE 802.15.4 physical layer, with TSCH scheduling at the link layer. Table 2 compares the main differences between WAIC and IEEE 802.15.4 physical layers. Due to the low transmission power of WAIC, a deep multi-hop topology can be expected even with a low number of nodes, which highlights the importance of enhancing multi-hop communication to meet the QoS requirements.

3 CROSS-LAYER FRAMEWORK

In this section, we introduce the details of the design and philosophy of the 6TiSCH framework with cross-layer enhancements, 6TiSCH-CLX, and how it changes the way the 6TiSCH stack operates.

3.1 Challenges

The primary goal of this framework is the optimization of both end-to-end delay and PDR in multi-hop TSCH networks. To better understand one of the main factors impacting delay, let us refer to the visualization in Figure 4, where two possible schedules of the same network are shown. The blue schedule is built by a random cell allocation, which for example is used in MSF. Each pair of nodes selects their communication window randomly and uniformly from the available range. On the contrary, the red schedule focuses on an ordered, *daisy-chained* cell allocation, which has a noticeable positive impact on the end-to-end delay for a data packet sent by node D to the sink.

Without loss of generality, let us assume that the link-collision probability is 0 and there is no need for retransmissions, because the packet is always successfully forwarded to the next hop. In this case, the end-to-end delay of this packet is determined entirely by the timeslot duration and by the queuing time at the intermediate nodes. The latter is heavily influenced by the order of TX cells along the multi-hop transmission path. For example, if nodes closer to the sink transmit packets earlier in the slotframe than the leaf nodes, they will spend additional time in the queues

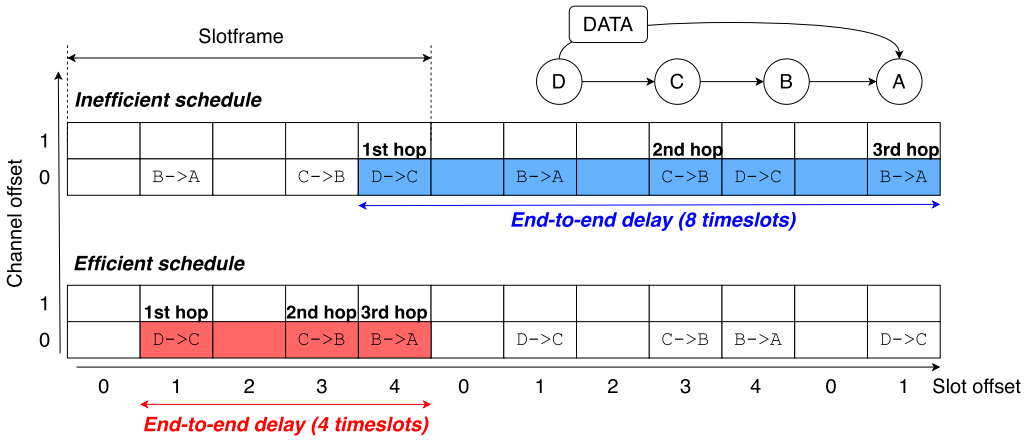


Fig. 4. Latency-inefficient vs -efficient scheduling, an example with Slotframe Length (SFL) = 5.

when they are forwarded. This happens because the closest transmission opportunity is already in the *next* slotframe, as visible from the blue schedule in Figure 4.

With the proper chaining of cells, however, a packet can reach the sink within a single slotframe, if the number of hops does not exceed the number of slots in the slotframe. This is why we employ the idea of cell daisy-chaining [5] to optimize the end-to-end routes across the whole network and to reduce latency. To tackle the reliability challenge, 6TiSCH-CLX utilizes the DODAG topology information, which is gathered by the RPL sink in order to split the network in the frequency domain and ensure that the probability of cell collisions is minimized, as explained in detail in the following sections.

3.2 General Principles

Cross-layer communication between RPL and TSCH is realized mostly by the 6P interface and the SF, which processes and forwards information between layers. The SF also implements the schedule management logic and is accompanied by 6P as a sort of flexible Application Programming Interface (API) for schedule manipulations between neighbors. The default 6TiSCH stack with MSF (6TiSCH-MSF) serves as a starting point for our improvements, as it represents a standardized ubiquitous solution, suitable for most consumer-level WSNs. The MSF has been an actively researched topic in the field of low-power, lossy wireless networks for the past few years (refer to Table 1). 6TiSCH-CLX can be decoupled into two main components:

— The **Cross-Layer Scheduling Function** (**CLSF**).
— RPL (in storing mode), extended with an interface to the SF and modified DIO and DAO packets.

Although RPL in a non-storing mode can also be used for 6TiSCH-CLX, it introduces a large overhead during the daisy-chaining of the transmission cells, as explained in more detail in the further sections. Although our improvements mostly focus on the end-to-end delay and PDR optimization, the framework itself can be further extended with other target metrics, such as throughput, energy consumption and duty-cycle. To reach the required QoS, 6TiSCH-CLX proceeds in three phases: schedule establishment, daisy-chaining, and continuous maintenance. The first phase is required to obtain a stable, converged topology in a typical DODAG form. The second phase optimizes all uplink routes by using a daisy-chaining mechanism to reduce latency, while also keeping

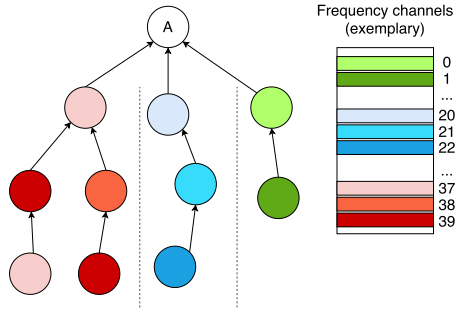


Fig. 5. Channel-disjoint branches, different colors refer to different frequency channels used in the HS.

the collision probability at minimum. Lastly, the maintenance phase, which follows the MSF, continues throughout the network lifetime. During this phase the cell performance and utilization is monitored to adapt for the traffic changes and mitigate interference.

3.3 Phase I — Schedule Establishment

During the first phase, 6TiSCH-CLX behaves similar to 6TiSCH-MSF. Initially, communication among the neighbors takes place by using only the shared auto cells. Upon joining a DODAG, a dedicated cell is scheduled between the node and its preferred RPL parent. The cell utilization is monitored to keep up with increased traffic demand, e.g., caused by DAO message forwarding. If a new cell is scheduled during this phase, it is done in the same randomized manner as in the MSF.

After an RPL sink receives DAOs from its one-hop children, the scheduling differs from MSF. In order to keep the schedule virtually collision-free, we split the topology into functional branches and allocate an equal number of *distinct* channel offsets for each branch. These branch-specific channel offsets are then used by the nodes of this branch as a starting point for the hopping sequence, as shown in Figure 5. As a result, any collisions between nodes of different branches are eliminated, as long as free cells are available for scheduling.

The RPL sink performs disjoint channel allocation using the modified *DAO_ACK* packet, which is shown in Figure 6. The sink equally and sequentially splits all available channel offsets into sets, one per branch. For example, given 40 channels in WAIC and a topology with 10 branches, each branch will be assigned 4 channels, where the first branch gets the channels with offsets 0–3, the second branch—channels offset by 4–7 and so on. The channel offset range advertised for a branch is carried by the *DAO_ACK* option header, which is not utilized in 6TiSCH by default. Therefore, the root of each branch knows about the assigned channel offset range and will disseminate this information downwards during the second scheduling phase.

To ensure reliable distribution of channel offsets per branch, a dedicated *downlink* cell is additionally scheduled between the RPL sink and each of its one-hop children. The channel distribution has to take place after all the one-hop children of the sink have joined the DODAG in order to equally distribute the available frequency spectrum. Considering the lossy nature of most WSNs, the exact way to kick off the channel advertisement process highly depends on the use case and may involve some heuristics, such as a timeout or another trigger condition, e.g., no DAOs from the one-hop neighbors are received by the sink since the last DIO broadcast. The timeout approach is suitable when there is no mobility in the network and it is safe to assume that all one-hop neighbors of the sink have joined the DODAG after some predefined amount of time. For the *static* avionic scenarios investigated in this work, we employ a timeout approach, which guarantees a proper channel offset distribution among the branches.

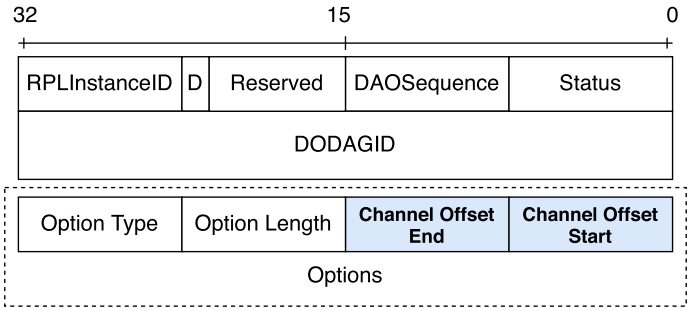


Fig. 6. DAO_ACK packet modified for 6TiSCH-CLX scheduling.

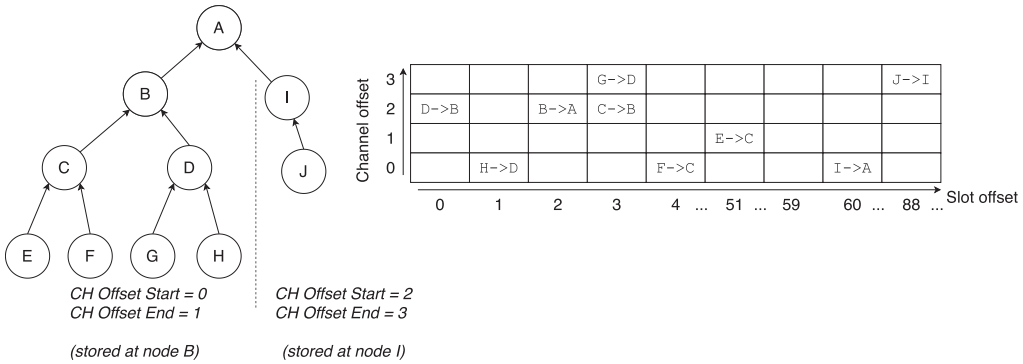


Fig. 7. Example of the schedule after phase I.

The first phase concludes with a converged RPL topology, where each node has a dedicated, randomly placed TX cell to its preferred parent, as shown in Figure 7. Additionally, the roots of the branches are informed about their assigned channel offsets and are then ready for the next scheduling phase, the daisy-chaining.

3.4 Phase II — Daisy-Chaining

The network in Figure 7 is already nearly collision-free, with guaranteed connectivity between neighbors through auto cells, as well as dedicated uplink cells. Without any external interference, collisions are only possible if at least two pairs of nodes in the mutual interference range schedule the exact same cells for communication. Considering the recommended slotframe size of 101 timeslots [31] and 40 WAIC frequency channels, the probability of at least two pairs of nodes scheduling overlapping cells is almost negligible. However, the goal of this phase is to reduce the latency across the network, while keeping the collision probability at a minimum.

To improve the end-to-end delay, we now apply the idea of daisy-chaining transmission cells along multi-hop paths to reduce the time that packets spend waiting in queues for the next transmission slot. Note that only the *dedicated* uplink cells are considered for daisy-chaining, as they're used to forward QoS-constrained application data. Minimal shared as well as any downlink cells are not specially managed by 6TiSCH-CLX, but follow the general scheduling rules from the MSF. This potentially allows *any* packet to reach the sink within a *single* slotframe, given the number of intermediate hops doesn't exceed the slotframe length. In the best-case scenario, with a slotframe of 101 timeslots, a packet can be transmitted along a tree of up to 101 hops within a *single* slotframe. A shorter example of a network with a daisy-chained schedule of 4 hops is visualized in Figure 8.

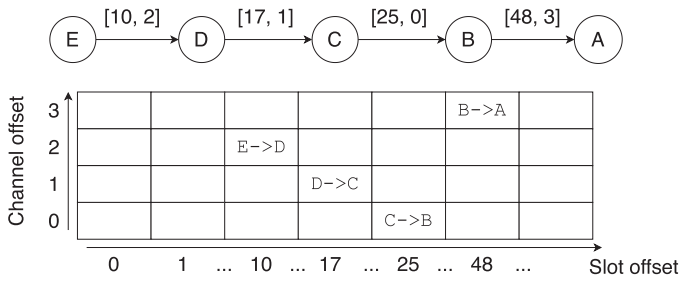


Fig. 8. Example of a daisy-chained schedule.

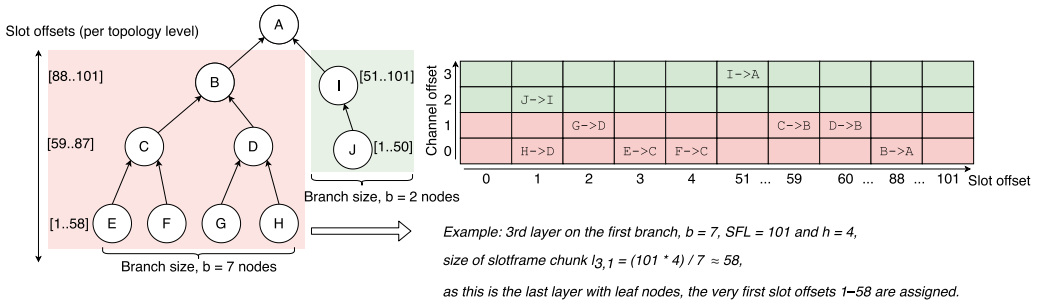


Fig. 9. The network schedule after the second phase.

To achieve this kind of cell ordering, we proceed as follows:

— For each branch the *whole* slotframe length is used.
— Branches are split into *layers* made up of nodes with the same hop-distance to the sink.
— For each layer, a portion of the slotframe, called *slotframe chunk*, is assigned.
— Layers near the sink get slotframe chunks closer to the end of the slotframe assigned, while layers further away are using slotframe chunks closer to the start of the slotframe
— The total number of slot offsets allocated to a ith layer on the jth branch (i.e., slotframe chunk size) is denoted as $l_{i,j}$ and provided in Equation (2), where S is the sloftframe length, h_i is the number of nodes in the ith layer (i.e., the layer size), and b_j is the total number of nodes in the jth branch (i.e., branch size).

$$l_{i,j} = \frac{S\ h_i}{b_j}. \tag{2}$$

The size of each slotframe chunk is proportional to the total number of nodes in a branch, as well as to the layer size. A working example of the assignment of slot offsets per layer is given in Figure 9. As a result, each layer is allocated a portion of the slotframe earlier than what is allocated to the preceding layer(s). Furthermore, regardless of the traffic or topology changes, the nodes always find themselves within a certain branch and layer, prescribing a defined portion of the schedule for operation. This is due to the fact that whenever nodes within a specific layer need to schedule additional cells to accommodate more traffic, they are limited in the following way:

— Time-wise limitation to this slotframe chunk.
— Frequency-wise limitation to the branch-specific channel offsets.

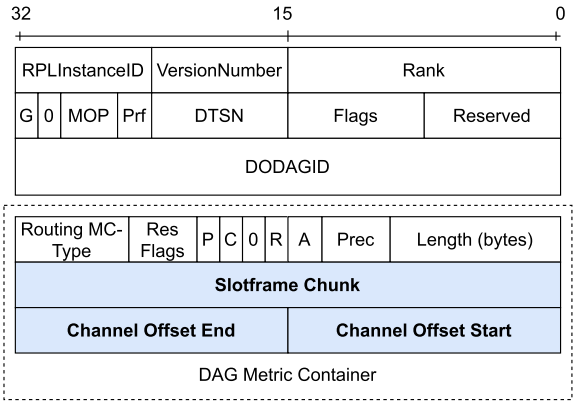


Fig. 10. DIO packet modified for 6TiSCH-CLX scheduling.

Phase II starts simultaneously for all branches, triggered by a special DIO message, which is broadcasted by the sink and identified by the first bit set in the reserved field. When such a message is received, the sub-root of each branch enumerates the number of known downlink destinations or branch size b. Using this value and Equation (2), the sub-root calculates a slotframe chunk for itself as well as one it advertises to its immediate children. This calculated slotframe chunk, alongside branch-specific channel offsets, is then included in a modified DIO packet extended with a custom DAG metric container as shown in Figure 10. Each node acts in the same manner when processing this cross-layer DIO: It first extracts the advertised slotframe chunk and the channel offsets of the branch and afterwards passes them to the CLSF module to be used for all future scheduling operations. After receiving the slotframe chunk and the branch-specific channel offsets from RPL, the Cross-Layer Scheduling Function (CLSF) ensures that the dedicated transmission cells are within the bounds given to the particular node and relocates them if necessary. The SF furthermore ensures that the candidate cell list proposed in all future 6P ADD and RELOCATE requests complies with these bounds. For example, when a candidate cell list for a 6P RELOCATE is being created, the slot offsets are chosen randomly only from the *branch-specific* channel offset range. Thus, if the CLSF detects that dedicated cells should be relocated to ensure the daisy-chaining property, a standard 6P RELOCATE request is issued to the parent node, with only a candidate cell list modified to ensure the proposed cells are within the slotframe chunk and branch-specific channel offset range.

While the CLSF takes care of the schedule, RPL recalculates the slotframe chunk to be included in further DIOs using Algorithm 1, where $[t_{chunk\_start},\ t_{chunk\_end}]$ is the slotframe chunk of the node itself, n_{desc} is the number of downwards-routable RPL descendants and n_{nb} is the number of 1-hop neighbors. The *storing* mode of operation for RPL is assumed, where each node itself estimates the number of downlink destinations, which corresponds to the branch size. However, also in the *non-storing* mode it is possible to employ 6TiSCH-CLX. In this case the *sink* handles cross-layer data dissemination by itself. A large DIO with multiple DAG metric containers is broadcasted by the sink multiple times, at least once for *each* branch, filled with branch-specific channel offsets and slotframe chunks for each single layer. These DIOs are then flooded downwards, advertising the necessary daisy-chaining information along the DODAG. As a possible drawback, packet error probability increases due to the larger DIOs, which may in turn delay the termination of phase II and congest the network.

The process continues until all nodes within a branch have aligned or relocated their cells to meet both, the slotframe chunk bounds, as well as the channel offset requirements of the particular

ALGORITHM 1: Advertised slotframe chunk calculation

Input: $rank$, t_{chunk_start}, t_{chunk_end}, n_{desc}, n_{nb}
Output: t_{adv_start}, t_{adv_end}

if $n_{desc} == 0$ **then**
    $t_{adv\_start} \leftarrow 0$
    $t_{adv\_end} \leftarrow 0$
    return
end if
if $rank == 2$ **then**
    $t_{chunk\_end} \leftarrow S$
    $t_{chunk\_start} \leftarrow \lceil t_{chunk\_end}(1 - \frac{n_{nb}}{n_{desc}}) \rceil$
end if
$t_{adv\_end} \leftarrow t_{chunk\_start} - 1$
if $t_{adv\_end} < 0$ **then**
    $t_{adv\_end} \leftarrow S$
end if
$t_{adv\_start} \leftarrow \lceil t_{adv\_end}(1 - \frac{n_{nb}}{n_{desc}}) \rceil$

branch, which results in daisy-chained multi-hop paths, as shown in Figure 9. Only DIOs from a preferred parent are processed during the dissemination of cross-layer information along the branches. This limitation is enforced to avoid any accidental involvement in the daisy-chaining procedure of the adjacent branches.

Note that 6TiSCH-CLX also does not affect RPL parent selection in any way. If a node loses connectivity to the DODAG, all RPL-/ and cross-layer—related information is wiped, so that upon hearing a DIO from a new candidate parent, the node re-joins the DODAG and uses both, a *new* slotframe chunk and *new* branch-specific channel offsets. Furthermore, a node maintains a list of candidate parents as usual, additionally storing their associated cross-layer information. In case a preferred parent becomes unreachable, a node can still switch to the next feasible successor, even if it belongs to another branch. A dedicated cell to this new preferred parent will then be scheduled using the slotframe chunk and branch-specific channel offsets contained in the last DIO received from this new parent. If there is no related cross-layer information, a dedicated cell location to the new parent will be selected randomly, as in the MSF.

If a node switches branches *after* phase II has concluded, the slotframe chunks will not be re-calculated, which might introduce a bottleneck issue in the affected branch. This issue, as well as mobility in general, may be addressed as a part of the future work. The 6TiSCH-CLX framework is mainly developed for *static* networks, where a timeout triggering the start of phase II ensures that the topology has *converged* based on the RPL Objective Function (OF) and the sink has discovered all routable destinations through DAOs. **Objective Function 0** (**OF0**), used in this work for RPL route optimization, ensures that each node is the least possible number of hops away from the sink. Without mobility, this number would not change throughout the network lifetime and the DODAG structure remains constant.

3.5 Phase III — Schedule Maintenance

After all nodes have relocated their dedicated TX cells in line with the branch-specific channel and layer-specific slotframe chunk requirements, the network enters the continuous schedule maintenance phase. In this phase, the cross-layer framework works similar to MSF and monitors both cell performance and cell utilization to avoid interference and to match traffic requirements.

Table 3. Notation Glossary

Notation	Description
n	number of nodes in the network
d	total end-to-end delay
d_{trans}	transmission delay
d_{qc}	queuing delay with collisions
d_{qn}	queuing delay without collisions
deg	number of neighbors of a node
h	number of hops from the sensor node to the sink
R	maximum number of retransmissions per link
C	number of available frequency channels
p_c	link collision probability
t	timeslot duration (ms)
S	slotframe length (in timeslots)
N_a	number of ways to arrange the possible collisions on the number of hops

Unlike MSF, if the need for more uplink cells is detected, CLSF limits the node to the portion of the schedule defined by the slotframe chunk and by the branch-specific channels. Thus, the daisy-chaining property of the branch as well as the branch separation in the frequency domain is kept intact, even if nodes schedule additional cells to handle more uplink traffic.

4 THEORETICAL EVALUATION

This section introduces the proposed analytical model to evaluate the performance of the SFs mathematically by using KPIs such as end-to-end delay and Packet Reception Probability (PRP). This model is used to compare the proposed 6TiSCH-CLX with 6TiSCH-MSF by varying parameters such as link collision probability and number of hops to the sink. The contribution of this section may also serve as a basis for further extension and parametrization to get even more detailed and precise analytical estimations of the SF performance. Table 3 summarizes the commonly used terminology of this section. Our main assumptions about the evaluated network are the following:

— Static multi-hop topology with a fixed schedule.
— Each node has a *single* dedicated uplink TX cell (one transmission opportunity per slotframe).
— Periodic uplink application traffic.
— Each packet can be retransmitted up to R times after a collision (retransmission threshold).
— All link collisions are independent and identically distributed (i.i.d.) and the collision probability is the same across all the links.

4.1 End-to-End Delay

Total end-to-end delay d: It consists of two components in this analysis, the transmission delay d_{trans} and the queuing delay d_q:

$$d = d_{trans} + d_q. \tag{3}$$

Total transmission delay d_{trans}: The sum of transmission delays d_{trans_i} for each hop i:

$$d_{trans} = \sum_{i=1}^{h} d_{trans_i}, \tag{4}$$

where h is the number of hops from the sender node to the sink. It is determined entirely by the timeslot duration t and has a constant impact on the end-to-end delay. For this reason, we leave it out of the scope of this investigation. **Total queuing delay** d_q: The sum of all queuing times a packet experiences at the originating and each intermediate node,

$$d_q = \sum_{i=1}^{h} d_{q_i}. \tag{5}$$

Unlike the transmission delay, the queuing delay depends on two variable factors: the *link collisions* and the *temporal ordering of TX cells along the path*. Each of these factors is also directly tied to the way the SF handles the schedule organization and will be explored in more detail in the next sections.

4.2 Impact of Link Collisions

Let us assume, there is only one dedicated TX cell to the preferred parent of a node. If a collision occurs and the packet has not been acknowledged, retransmission takes place in the *next* slotframe, effectively adding up to a full slotframe to the queuing delay of the packet. To differentiate between the normal queuing delay and the one caused by collisions, we will refer to them as d_{qn} and d_{qc}, respectively. Therefore, to simplify the analysis further, we can estimate d_{qc} in terms of discrete *slotframes*, which is easily translatable into metric time as follows:

$$d_{qc}\ [\text{ms}] = S\, t\, d_{qc}\ [\text{slotframes}], \tag{6}$$

where S is the number of timeslots in a slotframe and t is the timeslot duration in ms. Considering that d_{qc} increases by exactly 1 slotframe for each collision, the **Probability Mass Function** (**PMF**) of queuing delay with collisions, p_{coll}, can be derived based on the link collision probability as

$$p_{coll}(i) = P(c = i) = p_c^i \binom{i + h - 1}{h - 1} (1 - p_c)^h, \tag{7}$$

where $p_{coll}(i)$ denotes the probability that the queuing delay caused by collisions equals i slotframes, p_c is the link collision probability, and h is the number of hops to the sink.

Since p_c is the same across all links and link collisions are independent events, the probability that i collisions occur equals p_c^i. The end-to-end delay can only be estimated for packets, which are received at the *sink*, i.e., when there are h successful transmissions. The binomial coefficient of $\binom{i+h-1}{h-1}$ enumerates all possible ways to arrange i collisions on h hops, regardless of the order, as visualized in Figure 11. However, the fact that the packet will be dropped if the number of collisions on a single link exceeds the retransmission threshold R is not taken into account. For example, taking $R = 3$ in Figure 11, the case at the top with 4 collisions on the link $C \rightarrow B$ results in a lost packet and should, therefore, be excluded from the enumeration by the binomial coefficient in Equation (7). In order to account for the retransmission threshold R, we need to calculate the number of possible collisions for the number of hops under the condition that no link gets more than R collisions. Furthermore, while arranging collisions across links, the order of the links matters, which leads us to the concept of *restricted integer compositions* [30]. The number of possible ways to arrange c collisions on h links, such that each link has $\leq R$ collisions, is the same as the number of restricted integer compositions $F(c, h; 1, R; 1, R; \ldots 1, R)$ of $c \in \mathbb{Z}$ into *at most* h parts such that each part h_i is within the range $1 \leq h_i \leq R$. This can be partially achieved using the formula [2]:

$$F(c, h; 1, R; 1, R; \ldots 1, R) = \sum_{i=0}^{h} (-1)^i \binom{h}{i} \binom{c - iR - 1}{h - 1}. \tag{8}$$

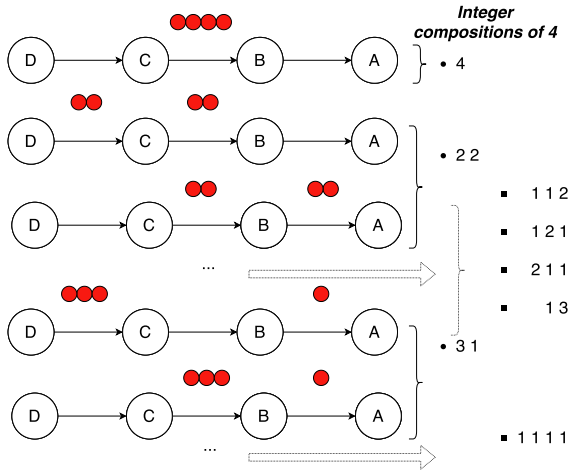


Fig. 11. Example of the possible integer compositions of a packet which suffers exactly $i = 4$ collisions (in red) for $h = 3$ hops.

However, Equation (8) still does not give us the required number of ways to arrange i collisions on h hops, because a single integer composition of c collisions may be placed on h hops in several ways, as visualized in Figure 11. Moreover, we need to consider not just h-compositions of c, but rather all possible h_i-compositions of c, for $1 \leq h_i \leq h$. The latter means that we are not only interested in the number of ways to arrange c collisions on *exactly* h hops, but rather on *up to* h hops. Therefore, the right-hand side in Equation (8) has to be additionally summed over the hop range $h_i = 1 \ldots h$ and each term multiplied by $\binom{h}{h_i}$ as follows:

$$N_a(i, h; 0, R) = \sum_{j=1}^{h} \sum_{k=0}^{j} (-1)^k \binom{j}{i} \binom{i - kR - 1}{j - 1} \binom{h}{j}, \tag{9}$$

where $N_a(i, h; 0, R)$ states the number of ways to arrange i collisions on the number of h hops, such that no link has $\geq R$ collisions. From Equations (7) and (9) the Probability Mass Function (PMF) of the number of collisions can be derived:

$$p_c(i) = P(c = i) = p_c^i \, (1 - p_c)^h \sum_{j=1}^{h} \sum_{k=0}^{j} (-1)^k \binom{j}{i} \binom{i - kR - 1}{j - 1} \binom{h}{j}. \tag{10}$$

4.3 Expected Queuing Delay With Collisions

Knowing the PMF of the queuing delay with the collisions $p[d_{qc_i}]$, the expected queuing delay with collisions is defined as

$$E[d_{qc}] = \sum_{i=1}^{hR} p[d_{qc_i}] d_{qc_i} \quad \text{[slotframes]}. \tag{11}$$

Only *up to* hR collisions are summed up as the lost packets are not considered. If $hR+1$ collisions occur, at least one link will have $\geq R$ collisions, resulting in a lost packet. Having established a direct relation between the collision-impacted queuing delay d_{qc} and the number of collisions c,

Equation (10) extends the expected queuing delay d_{qc} in Equation (11) as follows:

$$E[d_{qc}] = (1 - p_c)^h \left(\sum_{i=1}^{R} i\, p_c^i \binom{i+h-1}{h-1} + \sum_{i=R+1}^{hR} i\, p_c^i\, N_a(i, h; 0, R) \right), \tag{12}$$

which allows us to estimate the average queuing delay, added by collisions on a multi-hop path of length h in a network with the retransmission threshold R, based on the link collision probability p_c. The latter is a key parameter, which depends on a number of factors, such as application use-case, channel model, SF, and so on.

In the vast majority of situations, p_c is not uniform across all links in the network. The link collision probability varies for example depending on the SF used, which means that Equation (12) can be used to roughly compare different SFs based on their scheduling scheme. For instance, collisions in MSF are possible if at least two pairs of nodes have scheduled the exact same cell as their dedicated uplink. In this case, p_c can be estimated as

$$p_{c_{MSF}} \geq \frac{1}{(C(S - deg))^2} \binom{n}{2}, \tag{13}$$

where C is the number of frequency channels, S is the slotframe size, deg is the number of neighbors of the node as the vertex degree in a topology graph and n gives the total number of nodes in the interference range.

4.4 Queuing Delay Without Collisions

This section highlights the queuing delay without collisions d_{qn}, which is crucial for understanding the impact of the 6TiSCH-CLX framework. The term d_{qn} refers to the total time packets spent in queues at each node while waiting for the next transmission slot. As shown in the previous sections, if TX cells along the multi-hop path are not *daisy-chained*, each hop may provoke an additional delay of up to one full additional slotframe. Several widely used SFs [4, 9] utilize random cell allocation, which means this kind of delay is likely to have a considerable impact on the total end-to-end delay. Let us assume that the sequence of TX cell slot offsets along a multi-hop path is a sequence of natural numbers, then there is only *one* way to put these numbers in ascending order. Therefore, regardless of the slotframe size, the probability that any kind of randomized cell selection mechanism will avoid this additional queuing delay is

$$p_a = \frac{1}{h!}, \tag{14}$$

where p_a gives the probability sequence that h slot offsets are randomly drawn from a slotframe of size S in ascending order. With an increasing number of hops, the probability that randomized SF do *not* cause any kind of additional queuing delay is almost negligible.

To understand how much delay this unordered cell allocation causes, let us assume that the sequence of links along a multi-hop path is a sequence of natural numbers (slot offsets), with each pair of slot offsets placed in descending order. Each such pair adds *up to* a single slotframe of queuing delay for the full transmission. Under this condition, without considering the queuing delay caused by other packets, the PMF of the queuing delay without collisions d_{qn} can be estimated based on the number of link pairs with their slot offsets sorted in descending order along the path of h hops

$$p_{d_{qn}}(i) = P(d_{qn} = i, p_c = 0) = \binom{h-1}{i} \frac{1}{2^{h-1}}. \tag{15}$$

Only *adjacent* links are considered and therefore, given a total of h hops, there are $h - 1$ possible adjacent link pairs, as illustrated in Figure 12. Without any additional queuing, the link pair

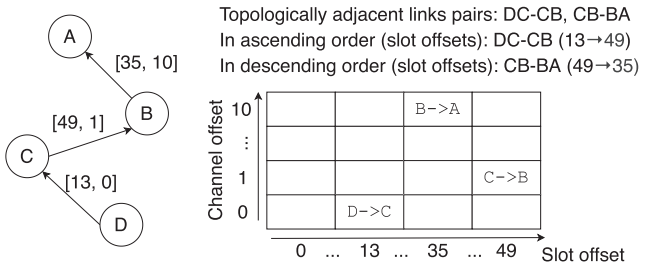


Fig. 12. Example of the link pairs ordered in ascending/descending order based on the slot offsets.

DC–CB does not require a packet to wait until the next slotframe to reach node *B*. However, the link pair *CB–BA* has a descending sequence of slot offsets: 49 for $C \rightarrow B$ transmission followed by slot offset 35 for $B \rightarrow A$, which causes additional waiting time even with an *empty* queue.

Extending this to the expected queuing delay without collisions gives

$$E[d_{qn}] = \sum_{i=1}^{h-1} i \binom{h-1}{i} \frac{1}{2^{h-1}}. \tag{16}$$

As shown in Equation (14), the probability that a number of slot offsets selected randomly is daisy-chained by chance, is negligibly small. Therefore, the queuing delay $E[d_{qn}]$ is certain to increase if a SF with random cell allocation is employed, compared to the CLSF allocation in ascending order. One other important aspect highlighted by Equation (16) is that the expected number of "'out-of-order" slot offset pairs does not depend on the slotframe length, but only on the total number of pairs considered. Therefore, the end-to-end delay improvement from daisy-chaining by 6TiSCH-CLX depends entirely on the average path length in the network, irrespective of the slotframe length. 6TiSCH-CLX performs exactly like 6TiSCH-MSF in the worst case, when either the slotframe length or the average path length equals 1. In all other cases, there is an opportunity to daisy-chain cells and lower the end-to-end delay.

If the depth of a topology branch exceeds the slotframe length, the daisy-chaining process is reset, i.e., the chunk allocation restarts from the end of the slotframe, as highlighted in line 12 of Algorithm 1. The schedule in Figure 13 illustrates how this may result in several daisy-chained slotframes within a branch. In such a case, a packet from the furthest leaf node is not able to reach the sink within a single slotframe. Nevertheless, compared to 6TiSCH-MSF, cell daisy-chaining of 6TiSCH-CLX presents a lower bound for latency.

4.5 Packet Reception Probability (PRP)

Let us consider the assumptions and results from the previous sections including an ideal channel model, where the only cause for packet loss is given by independent and identically distributed packet collisions along with the hops from the sender to the sink. In this case, a packet reaches the destination if it has not experienced more than *R* collisions on any link of its path with *h* hops and, therefore, the PRP can be computed as

$$PRP = (Pr[\leq R \text{ collisions}])^h = (1 - Pr[> R \text{ collisions}])^h = \left(1 - p_c^R\right)^h. \tag{17}$$

5 SIMULATION RESULTS

This section discusses an overview of the 6TiSCH stack with cross-layer information exchange (6TiSCH-CLX) implementation in OMNeT++ [25], describes the simulation scenarios, and concludes with the results.

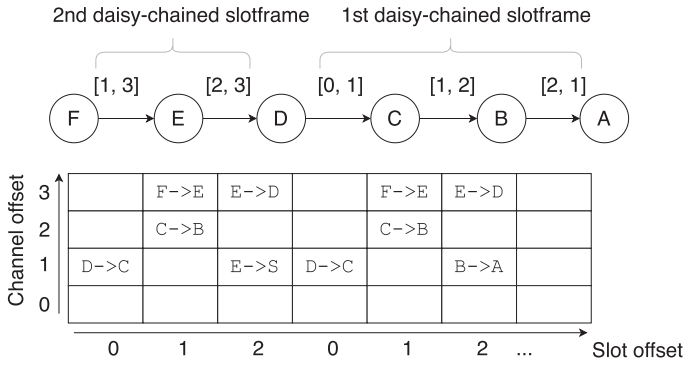


Fig. 13. Daisy-chaining under slotframe length smaller than the branch size.

5.1 Implementation

The proposed framework is implemented in the event-based simulation environment OMNeT++, based on the INET framework for wired, wireless and mobile networks [16]. The INET framework features an extensive suite of TCP/IP network models, including various layers and protocols of the TCP/IP Protocol Stack. The 6TiSCH implementation presented in this work follows the modular approach of INET and includes numerous layers, such as IEEE 802.15.4 PHY, TSCH, 6P with the dedicated SF module, and RPL. The simulation code is publicly available on GitHub [7]. A block diagram of the implementation work is shown in Figure 14. The WAIC PHY is based on the IEEE 802.15.4 PHY with **Quadrature Phase Shift KeyingDirect Sequence Spread Spectrum** (**OQPSK-DSSS**) modulation, but utilizes a higher number of channels, different bandwidth and the TX power limits summarized in Table 2. For the evaluation in OMNeT++, the WAIC PHY is modeled by extending IEEE 802.15.4 PHY with the parameters from Table 5. The TX power is set to an extremely low 0.05 nW as the worst case bound, since the exact in-cabin propagation model is not available and the limits imposed by the WAIC standard are not finalized yet. This allows for TX ranges of up to 3–4 m.

The channel and propagation models, most of the IPv6 and UDP functionality are inherited directly from the INET framework. The block diagram in Figure 14 shows the relationships between the individual modules, as well as the TCP/IP layers they belong to. The modules have a clear task separation, which allows for fine granularity in terms of control management. Each module is accountable for a small number of tasks only, which simplifies the development and improves flexibility. This way, both the MSF and the CLSF are implemented using the common SF interface, universally compatible with the 6P sublayer. The RPL extensions of 6TiSCH-CLX are contained within the core RPL class due to their tight coupling with its basic mode of operation. The areas of main implementation work for the 6TiSCH-CLX framework are highlighted in red in Figure 14.

5.2 Scenarios

The goal of the simulation campaign is to evaluate the performance of the proposed 6TiSCH-CLX framework and benchmark against 6TiSCH-MSF. Several WAIC application scenarios are compared, which represent different traffic intensity and priorities, as summarized in Table 4. The scenarios are defined together with partners from the aviation industry as part of the research projectReSA [27] and they reflect possible application use cases for WSNs.

The main KPIs investigated are end-to-end delay and PDR, since they are of utmost importance for safety-critical avionic applications and therefore are chosen as primary goals for optimization

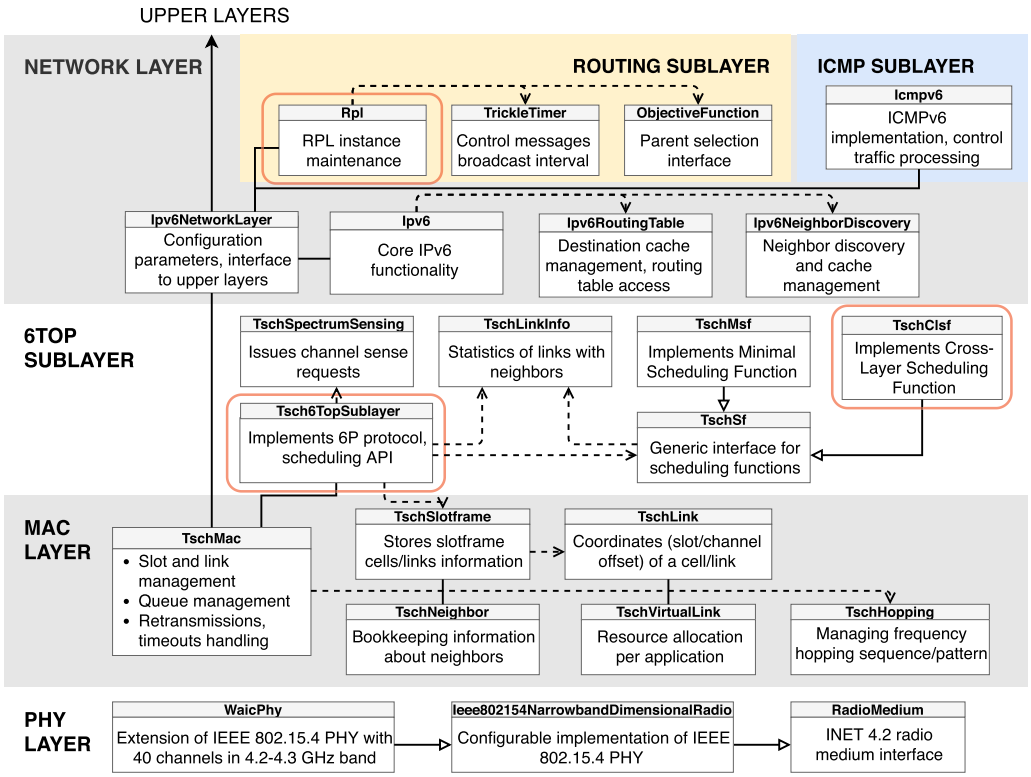


Fig. 14. The 6TiSCH implementation block diagram.

Table 4. Avionic Scenarios Details

Name	Latency (upper bound)	Nodes	Priority	Pattern
Seat belt status	1s	up to 200	average	aperiodic
Smoke alarm	1s	30	high	aperiodic
Cabin humidity monitoring	60 s	20–30	low	periodic

in 6TiSCH-CLX. Although the energy consumption is also relevant in the IoT context, for WAIC applications it is of secondary priority compared to reliability and low latency of sensitive data. RPL utilizes the Objective Function 0 (OF0), meaning that the rank of a node represents the distance to the sink in hops.

In the seat belt status scenario, 200 nodes are deployed within a rectangular area with the dimensions 60m × 10m, imitating a grid of passenger seats with sinks arranged evenly along the middle corridor, as shown in Figure 15. Choosing the appropriate minimum number of sinks is the key to avoid network congestion. Bottlenecks are expected to be prevalent in the dense seat belt scenario, given the fixed slotframe length, a defined application traffic pattern, and a high number of nodes. For this reason, the KPIs are collected for a variable number of sinks.

For the smoke alarm and humidity monitoring scenarios, a tree-like topology with a single sink is created, represented in Figure 15. It aims at resembling a possible placement of sensor nodes within an aircraft. The emphasis here is on the long, multi-hop connections, which allow to evaluate the transmission performance based on the distance of a node to the sink, expressed

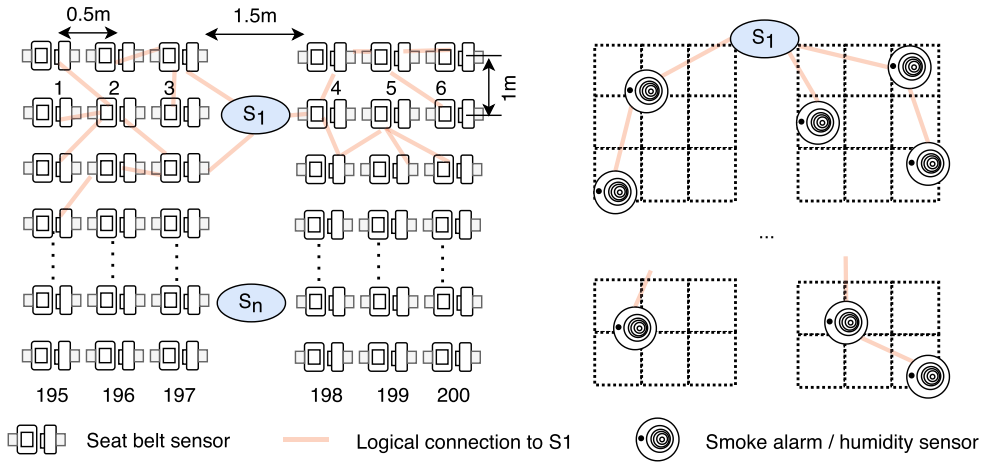


Fig. 15. Layout of the simulation scenarios with n sinks for the seat belt status (left) and 1 sink for the smoke alarm and humidity monitoring scenario (right).

Table 5. Physical & MAC Parameters

Parameter	Value
Carrier frequency	4.3 GHz
Bitrate	250 kbps
Channel bandwidth	2.6 MHz
Receiver sensitivity	−91 dBm
Background noise	−109 dBm
Transmit power	0.05 nW
Slotframe size	101 timeslots
Frequency channels	40
Packet queue size	20

Table 6. RPL & SF Configuration

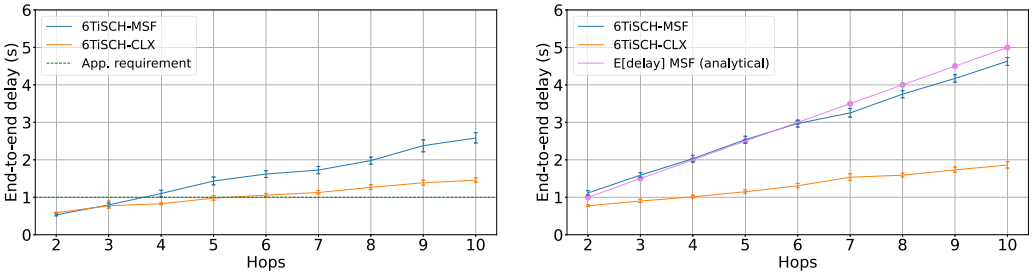
Parameter	Value
RPL mode of operation	storing
CLSF phase 2 timeout	100 s (smoke alarm, humidity), 500 s (seat belt status)
Number of minimum cells	7 timeslots
Cell utilization period	30 cells
Cell utilization bounds	[0.1, 0.75]

by the *rank* of the node. In each simulation round, 3 nodes of the same rank are actively sending application data to the sink with the target of assessing the impact of the hop distance on the end-to-end delay, without introducing queuing delays from intermediate nodes. Therefore, the results are comparable to the analytical expectation from Equation (16).

Common physical and link layer parameters are represented in Table 5, followed by the RPL and SF configurations in Table 6 as well as application parameters summarized in Table 7. The physical layer parameters reflect the latest drafts from the **Radio Technical Commission for Aeronautics** (**RTCA**) Working Group SC-236, 19th Joint Meeting, Minutes of Meeting [28] as well as the ITU-R M.2283-0 report [18]. Given the size of the network, the length of the slotframe is set to the recommended 101 timeslots from the respective RFC document [32]. Seven minimum cells are used for the control and broadcast traffic of IPv6 and RPL, because during preliminary evaluations it showed the most promising results in ensuring faster topology convergence, while it avoids spending much idle time listening after the topology is converged. The cell utilization period is set to 40 cells, which means that every 40 cells that elapsed in transmissions with a neighbor, the cell usage is estimated by the SF to decide whether more cells are required to handle the traffic. At the same time, if the lower cell utilization bound is not reached, a cell is deleted from the schedule of both neighboring nodes.

Table 7. Application Layer and Simulation Parameters

Parameter	Value
Simulation duration	30 min
Simulation repetitions	20
Packet size	50 B
Send interval	
Cabin humidity monitoring	uniform(10s, 20s)
Smoke alarm	uniform(0.5s, 1s)
Seat belt status	uniform(30s, 50s)
Number of transmitting nodes	
Cabin humidity monitoring	3 (of the same rank)
Smoke alarm	3 (of the same rank)
Seat belt status	200



(a) Smoke Alarm (multiple TX cells per hop)

(b) Humidity Monitoring (single TX cell per hop)

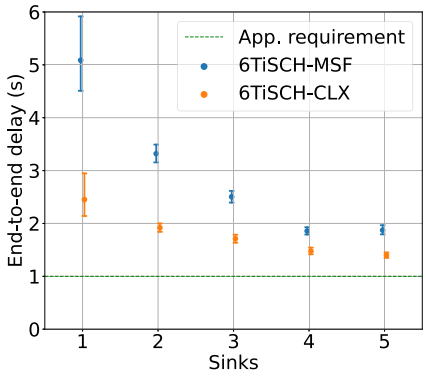
Fig. 16. End-to-end delay in the smoke alarm and humidity monitoring scenarios.

5.3 Results Analysis

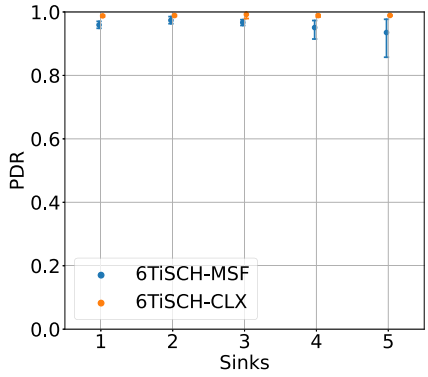
The results in this section were collected from simulation runs with a simulated duration of 1 hour. All values displayed are averaged over 20 repetitions and plotted with 95% confidence intervals. The end-to-end delay for the smoke alarm and humidity monitoring scenarios are visualized in Figure 16. Additionally, the latency required by the smoke alarm application, as well as the expected delay of the MSF are plotted. The latter is calculated as given by Equation (16) in combination the with an average service time per hop, which equals $1/N_{TX}$, where N_{TX} is given as the number of TX cells scheduled per node. The expectation curve is not shown for the smoke alarm sensor, since Equation (16), used to estimate the expected delay, only considers a *single* TX cell per hop. The traffic intensity of the smoke alarm forces the SF to schedule multiple cells, though.

In comparison to 6TiSCH-MSF, a significant performance improvement brought by 6TiSCH-CLX can be observed: the end-to-end delay is up to four times lower in the humidity monitoring case. The positive effect is less noticeable in the smoke alarm case due to a higher traffic intensity. In fact, both the MSF and the CLSF perform cell monitoring to reactively satisfy traffic demand by allocating more uplink TX cells. During this process, the CLSF always ensures the daisy-chaining property along the multi-hop path, while the MSF selects cell locations randomly. A higher number of TX cells at each intermediate link assists the MSF by increasing the probability of cell daisy-chaining.

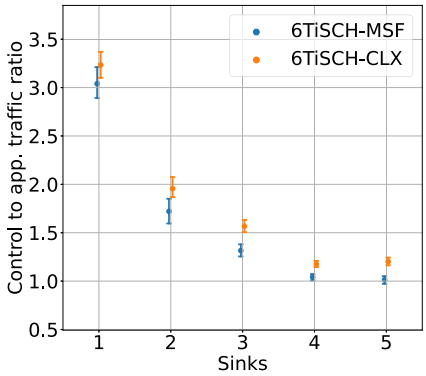
Figure 17(a) and (b) show the end-to-end delay and PDR of the seat belt scenario for a variable number of sinks, respectively. The results clearly indicate improvements in the end-to-end delay



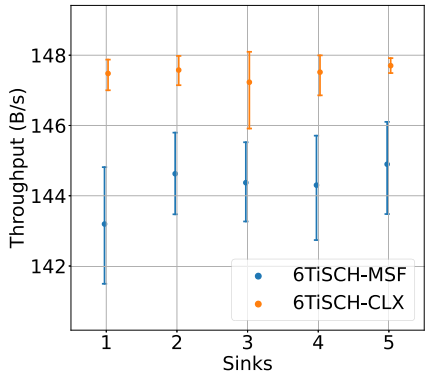
(a) End-to-end delay (s)



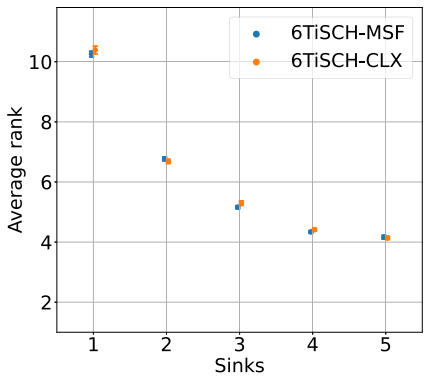
(b) Packet Delivery Ratio



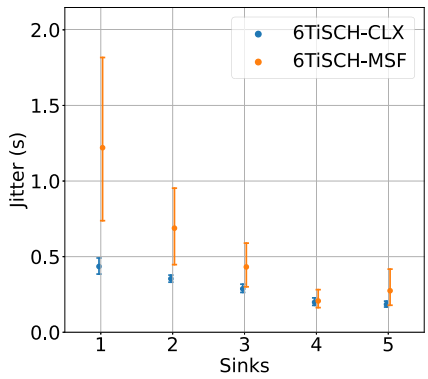
(c) Control traffic overhead



(d) Throughput (B/s)



(e) Average node rank



(f) Jitter (s)

Fig. 17. Simulation results of the seat belt scenario.

due to the fact that each packet is always traversing a daisy-chained path with minimal queuing delays. The PDR stays mostly the same for both 6TiSCH-MSF and 6TiSCH-CLX, as there are no reasons that may lead to packet losses, such as external interference or link collisions.

Considerably larger confidence intervals are caused in the case of a single sink by bottleneck links at the nodes close to this sink, due to the fact that all DAOs as well as application traffic from descendants have to be relayed through these nodes. This requires additional bandwidth, which has to be negotiated with the sink. With an increasing number of simultaneous 6P transactions, the contention for the available slot offsets grows, which results in more failed requests. Furthermore, due to the rare race conditions within a node participating in multiple 6P transactions, some scheduled cells may overlap in time. To resolve such cases, a SF inconsistency handling mechanism is invoked, using the 6P *CLEAR* command to reset the schedule between conflicting neighbors. This results in bursty traffic of 6P control packets, which increases the queuing time for application packets given the absence of traffic prioritization.

As visible from Figure 17(b), the PDR stays closely to 100% for both 6TiSCH-CLX and 6TiSCH-MSF. This is achieved by the low-intensity traffic in this scenario. Occasional packet drops are mostly caused by the previously described 6P inconsistencies, during which the neighboring nodes have to reset their schedules. Especially the queues of nodes closer to the sink may get filled rapidly during the inconsistency resolving process, which is caused by the high number of child nodes generating application data. To alleviate the issue, the frequency of cell usage estimation, denoted as MAX_NUM_CELLS in the MSF terminology [4, p. 10], maybe adjusted to improve the traffic adaptation mechanism. Figure 17(e) depicts the average node rank in the generated DODAG. Considering that the OF0 is used in RPL, the rank directly reflects the number of hops, which an application packet traverses to the nearest sink. With more than 10 intermediate hops, even with daisy-chaining, it is difficult to achieve the required 1s end-to-end delay for application packets, because of both bandwidth limitations, as well as the absence of traffic prioritization.

Figure 17(c) visualizes the ratio between control and application packets transmitted during the network lifetime. It is calculated as the total number of RPL and 6P control packets emitted by all nodes, divided by the total number of application packets received by all sinks. The mean value of the ratio is between 3 and 3.5 when a single sink is used, due to the fact that no DAO aggregation mechanism according to [1, p. 83] is implemented in the simulation framework. This means that each time a node discovers a new reachable destination, it immediately forwards a DAO to its preferred parent, which results in excessive traffic at the bottleneck links close to the sink. Although 6TiSCH-CLX marginally increases the control traffic overhead, due to the overlapping confidence intervals between 6TiSCH-MSF and 6TiSCH-CLX it remains unclear whether this would be noticeable on practice.

In Figure 17(d) and (f), the results for 6TiSCH-MSF and 6TiSCH-CLX in terms of throughput and jitter are plotted. Both are commonly considered QoS parameters alongside the PDR and latency. While the main focus of 6TiSCH-CLX lies on delay and reliability optimizations, neither throughput nor jitter experience any degradation when compared to 6TiSCH-MSF. In fact, both metrics are improved and tend to be more consistent under 6TiSCH-CLX thanks to the optimized end-to-end routes. Especially prominent is the jitter, which stays below 0.5s even with a single sink.

The convergence time of the daisy-chaining phase is also estimated, as it can be considered to be a part of the network bootstrapping process. During this time nodes relocate their dedicated uplink cells according to the assigned slotframe chunk and the branch-specific channel range. The convergence time is estimated as the difference between the earliest start and latest termination time of the daisy-chaining phase across the entire network. Results show that the convergence time for all considered scenarios varies between 25 and 125 seconds, regardless the number of

sinks involved. The latter point can be explained by the fact that the daisy-chaining phase starts simultaneously in all branches and triggers nodes to start the cell relocation, which then proceeds mostly in parallel at different layers in each branch. This results in similar convergence times. A high variation in the convergence time is caused by a few nodes struggling to finish the daisy-chaining in the mean 25s–50s range. This issue can be attributed to failed or lost 6P RELOCATE requests, which can only be detected after a standardized 6P timeout [4, p. 13]. It is calculated depending on the use case and is set to 60s in our evaluation. Adjusting the timeout value and the number of proposed cells in the 6P request, the convergence time can be reduced considerably.

Daisy-chaining can be done in the presence of other traffic, however, it is advised to be performed right after the network bootstrapping for the constructed DODAG, before the application traffic starts to be transmitted. For this case, a 2min to 3min network setup phase is recommended in practice.

6 CONCLUSION

In this work, the end-to-end QoS in avionic WSNs with the 6TiSCH framework is investigated. It highlights the challenges of guaranteeing low-latency, reliable communication that is required in industrial use-cases. For avionic networks, the industry solution WAIC is in the process of being standardized. In contrast to the default IEEE standard for Low-Rate Wireless Networks *IEEE 802.15.4*, WAIC uses a 4.3 GHz frequency band and enforces strict transmit power limits, which in turn reduce the transmission range for intra-aircraft networks and hence require multi-hop communication for sensor networks.

A solution labeled as 6TiSCH-CLX is proposed to accomplish the required performance in safety-critical avionic applications. 6TiSCH-CLX extends the 6TiSCH framework with cross-layer information exchange mainly between the network and the link layers. Together with a dedicated SF and a daisy-chaining of network paths, the network performance is improved by reducing end-to-end delays and eliminating intra-network interference. By combining the SF with the centralized topology overview from the RPL and with the distributed schedule management of the 6TiSCH Operation Sublayer (6top) Protocol (6P), this hybrid solution intervenes in several phases to optimize the network performance.

6TiSCH-CLX is evaluated using both an analytical model and simulations of a simplified WSN with TSCH in WAIC on the 4.3 GHz band. The proposed solution can similarly be realized on the default IEEE 802.15.4 physical layer, since it only operates on layer 2 and above of the OSI model. With 95% confidence in the significance of the simulation results, the evaluation demonstrates that up to 4 times latency reduction is possible, compared to default 6TiSCH with MSF in scenarios with 30 nodes. In a large-scale case with 200 nodes, the proposed cross-layer solution is able to achieve better results in terms of latency and throughput, while minimizing jitter and packet losses. Furthermore, the growth in the control traffic overhead compared to 6TiSCH-MSF and the duration of the daisy-chaining phase can be considered negligible even in the scenario with 200 nodes, which is promising for the scalability of the proposed framework. 6TiSCH-CLX is developed with extensibility in mind, which allows it to target other QoS metrics, such as energy consumption, link quality, etc. by modifying the scheduling function CLSF.

Another contribution of this work is the analytical model of a static, multi-hop WSN with a fixed TSCH schedule and regular traffic. This model allows conducting fast and transparent estimations of the PRP and end-to-end delays based on the hop distance to the sink and the link collision probability. This can in turn be utilized for comparisons of SFs. The model itself can be extended further to account for dynamic scenarios and traffic flows.

Overall, 6TiSCH-CLX is able to meet the 1s end-to-end delay requirement of a typical safety-critical avionic application in WAIC with 30 nodes, if the sensor node is at most 5 hops away from

the sink. Nevertheless, the delays are higher in the 200 node case due to increased network congestion at nodes close to the sink and due to longer multi-hop paths. As future work, we consider implementing a more sophisticated mechanism to manage the link resources at the bottleneck links, as well as adding priority queues to handle critical application traffic in a timely manner.

ACKNOWLEDGMENTS

The authors would like to thank the lab engineer Frank Laue from the *Institute of Communication Networks, Hamburg University of Technology* for his support in implementation and for providing background knowledge.

REFERENCES

[1] Roger A., Anders B., J. P. Vasseur, J. Hui, K. Pister, P. Thubert, P. Levis, R. Struik, R. Kelsey, and T. Winter. 2012. RPL: IPv6 Routing Protocol for Low-Power and Lossy Networks. *RFC* 6550. (2012). DOI:https://doi.org/10.17487/RFC6550

[2] Morton Abramson. 1976. Restricted combinations and compositions. *Fibonacci Quart* 14, 5 (1976), 439.

[3] N. Accettura, M. R. Palattella, G. Boggia, L. A. Grieco, and M. Dohler. 2013. Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things. In *Proceedings of the 2013 IEEE 14th International Symposium on "A World of Wireless, Mobile and Multimedia Networks"*. 1–6.

[4] T. Chang, M. Vučinić, X. Vilajosana, S. Duquennoy, and D. R. Dujovne. 2021. 6TiSCH Minimal Scheduling Function (MSF). *RFC* 9033. (2021). DOI:https://doi.org/10.17487/RFC9033

[5] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana. 2016. LLSF: Low latency scheduling function for 6TiSCH networks. In *Proceedings of the 2016 International Conference on Distributed Computing in Sensor Systems*. 93–95.

[6] D. Chen, M. Nixon, and A. Mok. 2010. *WirelessHARTTM: Real-Time Mesh Network for Industrial Automation*. Springer. Retrieved from https://books.google.de/books?id=zIY9s_KyvGgC.

[7] ComNets. 2021. Wireless Avionics Intra-Communications (WAIC) simulation model for OMNeT++, utilizing IEEE 802.15.4 Time Slotted Channel Hopping (TSCH). (2021). Retrieved 27 Aug., 2021 from https://github.com/ComNetsHH/omnetpp-tsch/tree/6tisch.

[8] M. Domingo-Prieto, T. Chang, X. Vilajosana, and T. Watteyne. 2016. Distributed PID-based scheduling for 6TiSCH networks. *IEEE Communications Letters* 20, 5 (2016), 1006–1009.

[9] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne. 2015. Orchestra: Robust mesh networks through autonomously scheduled TSCH. In *Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems*. 337–350.

[10] A. J. Fahs, R. Bertolini, O. Alphand, F. Rousseau, K. Altisen, and S. Devismes. 2017. Collision prevention in distributed 6TiSCH networks. In *Proceedings of the 2017 IEEE 13th International Conference on Wireless and Mobile Computing, Networking and Communications*. 1–6. DOI:https://doi.org/10.1109/WiMOB.2017.8115798

[11] Á. A. Farías and D. Dujovne. 2015. A queue-based scheduling algorithm for PCE-enabled industrial internet of things networks. In *Proceedings of the 2015 6th Argentine Conference on Embedded Systems*. 31–36.

[12] Final Acts World Radiocommunication Conference 2015. Final Acts World Radiocommunication Conference 2015. (2015). Retrieved from https://www.itu.int/dms_pub/itu-r/opb/act/R-ACT-WRC.12-2015-PDF-E.pdf.

[13] T. Hamza and G. Kaddoum. 2019. Enhanced minimal scheduling function for IEEE 802.15.4e TSCH networks. In *Proceedings of the 2019 IEEE Wireless Communications and Networking Conference*. (2019). DOI:https://doi.org/10.1109/wcnc.2019.8885940

[14] L. Hanschke, L. Krüger, T. Meyerhoff, C. Renner, and A. Timm-Giel. 2017. Radio altimeter interference mitigation in wireless avionics intra-communication networks. In *Proceedings of the 2017 15th International Symposium on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks*. IEEE, 1–8.

[15] I. Hosni and F. Theoleyre. 2017. Self-Healing distributed scheduling for end-to-end delay optimization in multi-hop wireless networks with 6TiSCH. *Computer Communications* 110 (2017), 103–119. DOI:https://doi.org/10.1016/j.comcom.2017.05.014

[16] inet 2021. INET Framework. (2021). Retrieved from https://inet.omnetpp.org [Accessed on: 29.3.2021].

[17] ISA 2011. ANSI/ISA-100.11A-2011 wireless systems for Industrial Automation: Process Control and related applications. Retrievd on 27 Aug., 2021 from https://www.isa.org/products/ansi-isa-100-11a-2011-wireless-systems-for-industr.

[18] International Telecommunication Union (ITU) Recommendation ITU-R M.2283-0 (2013). Retrieved 27 Aug., 2021 from https://www.itu.int/dms_pub/itu-r/opb/rep/R-REP-M.2283-2013-PDF-E.pdf.

[19] Y. Jin, P. Kulkarni, J. Wilcox, and M. Sooriyabandara. 2016. A centralized scheduling algorithm for IEEE 802.15.4e TSCH based industrial low power wireless networks. In *Proceedings of the 2016 IEEE Wireless Communications and Networking Conference*. 1–6.

[20] J. Jung, D. Kim, T. Lee, J. Kang, N. Ahn, and Y. Yi. 2020. Distributed slot scheduling for QoS guarantee over TSCH-based IoT networks via adaptive parameterization. In *Proceedings of the 2020 19th ACM/IEEE International Conference on Information Processing in Sensor Networks*. 97–108.

[21] P. Levis, T. H. Clausen, O. Gnawali, J. Hui, and J. Ko. 2011. The Trickle Algorithm. *RFC* 6206. (2011), 3. DOI:https://doi.org/10.17487/RFC6206

[22] S. Mersch, T. Meyerhoff, L. Krüger, and A. Timm-Giel. 2018. Coexistence of wireless avionics intra-communication networks. In *Proceedings of the 2018 6th IEEE International Conference on Wireless for Space and Extreme Environments*. IEEE, 18–23.

[23] J. Nowotsch and M. Paulitsch. 2012. Leveraging multi-core computing architectures in avionics. In *Proceedings of the 2012 9th European Dependable Computing Conference*. 132–143. DOI:https://doi.org/10.1109/EDCC.2012.27

[24] M. Ojo and S. Giordano. 2016. An efficient centralized scheduling algorithm in IEEE 802.15.4e TSCH networks. In *Proceedings of the 2016 IEEE Conference on Standards for Communications and Networking*. 1–6.

[25] omnet 2021. OMNeT++ Discrete Event Simulator. (2021). Retrieved from https://omnetpp.org [Accessed on: 29.3.2021].

[26] M. R. Palattella, N. Accettura, M. Dohler, L. A. Grieco, and G. Boggia. 2012. Traffic aware scheduling algorithm for reliable low-power multi-hop IEEE 802.15. 4e networks. In *Proceedings of the 2012 IEEE 23rd International Symposium on Personal, Indoor and Mobile Radio Communications*. IEEE, 327–332.

[27] ReSA 2021. ReSA Project. (2021). Retrieved from https://tore.tuhh.de/cris/project/pj00090 [Accessed on: 31.3.2021].

[28] RTCA 2020. DRAFT Minutes of Meeting EUROCAE WG-96 RTCA SC-236 19th Joint Meeting. (2020). Retrieved from https://www.rtca.org/wp-content/uploads/2021/01/236sum19.pdf [Accessed on: 11.8.2021].

[29] R. Soua, P. Minet, and E. Livolant. 2016. Wave: A distributed scheduling algorithm for convergecast in IEEE 802.15.4e TSCH networks. *Transactions on Emerging Telecommunications Technologies* 27, 4 (2016), 557–575. DOI:https://doi.org/10.1002/ett.2991

[30] E. Steffen. 2013. Restricted weighted integer compositions and extended binomial coefficients. *Journal of Integer Sequences* 16 (2013), 3. Retrieved from https://cs.uwaterloo.ca/journals/JIS/VOL16/Eger/eger6.pdf.

[31] P. Thubert. 2020. *An Architecture for IPv6 over the TSCH mode of IEEE 802.15.4*. Internet-Draft draft-ietf-6tisch-architecture-30. Internet Engineering Task Force. Retrieved 27 Aug., 2021 from https://datatracker.ietf.org/doc/html/draft-ietf-6tisch-architecture-30. Work in Progress.

[32] X. Vilajosana, K. Pister, and T. Watteyne. 2017. Minimal IPv6 over the TSCH Mode of IEEE 802.15.4e (6TiSCH) Configuration. *RFC* 8180. (2017), 24. DOI:https://doi.org/10.17487/RFC8180

[33] Q. Wang, X. Vilajosana, and T. Watteyne. 2018. 6TiSCH Operation Sublayer (6top) Protocol (6P). *RFC* 8480. (2018). DOI:https://doi.org/10.17487/RFC8480

Received 31 March 2021; revised 21 August 2021; accepted 27 September 2021