IoT

MDPI

Article

TSCH Evaluation under Heterogeneous Mobile Scenarios

Charalampos Orfanidis [1], Atis Elsts [2], Paul Pop [1] and Xenofon Fafoutis [1,*]

[1] Department of Applied Mathematics and Computer Science, Technical University of Denmark (DTU), 2800 Kgs. Lyngby, Denmark; chaorf@dtu.dk (C.O.); paupo@dtu.dk (P.P.)
[2] Institute of Electronics and Computer Science (EDI), LV-1006 Riga, Latvia; atis.elsts@edi.lv
* Correspondence: xefa@dtu.dk

Abstract: Time Slotted Channel Hopping (TSCH) is a medium access protocol defined in the IEEE 802.15.4 standard. It has proven to be one of the most reliable options when it comes to industrial applications. TSCH offers a degree of high flexibility and can be tailored to the requirements of specific applications. Several performance aspects of TSCH have been investigated so far, such as the energy consumption, reliability, scalability and many more. However, mobility in TSCH networks remains an aspect that has not been thoroughly explored. In this paper, we examine how TSCH performs under mobility situations. We define two mobile scenarios: one where autonomous agriculture vehicles move on a predefined trail, and a warehouse logistics scenario, where autonomous robots/vehicles and workers move randomly. We examine how different TSCH scheduling approaches perform on these mobility patterns and when a different number of nodes are operating. The results show that the current TSCH scheduling approaches are not able to handle mobile scenarios efficiently. Moreover, the results provide insights on how TSCH scheduling can be improved for mobile applications.

Keywords: IoT; TSCH; mobility; reliability; robustness

Citation: Orfanidis, C.; Elsts, A.; Pop, P.; Fafoutis, X. TSCH Evaluation under Heterogeneous Mobile Scenarios. *IoT* **2021**, *2*, 656–668. https://doi.org/10.3390/iot2040033

Academic Editors: Prem Prakash Jayaraman, Abdur Rahim Mohammad Forkan and Ali Hassani

Received: 20 September 2021
Accepted: 21 October 2021
Published: 22 October 2021

1. Introduction

Time Slotted Channel Hopping (TSCH) is a Medium Access Control (MAC) protocol for Internet of Things (IoT) applications, which has been used broadly in the industry since it can offer a high level of reliability and robustness [1]. TSCH is implemented as a channel-hopping mechanism to avoid interference and multi-path fading, which Low-power and Lossy Networks (LLNs) are prone to. Furthermore, all devices in a TSCH network are synchronized to a common time source: the coordinator node. Essentially, all the nodes are synchronized (with a tolerated offset), and a schedule is calculated. The purpose of the schedule is to dictate which node is supposed to perform an action (receive, transmit, sleep), on which channel and at which point in time.

The modular nature of TSCH has enabled high flexibility, and therefore, there are numerous application scenarios that have utilized the TSCH mechanism. Some examples include industrial monitoring [2], environmental monitoring [3], smart home for healthcare [4], smart buildings [5] and many more. TSCH is extended for sub-GHz and long range communications in [6] to monitor the temperature and humidity in an application scenario where the grain pile is monitored in order to ensure its quality in the long term but also during transportation. The evaluation results report a 99% packet delivery ratio, which proves that TSCH is a reliable solution for this application scenario. TSCH is also used in an e-Health application [7] installed in 29 long-term residential deployments. The applications include accelerometer data from wrist-worn wearables collected from inhabitants, localization services (better than room-level accuracy) and environmental monitoring (temperature and light levels) collected from the installed residencies. The sensor data are timestamped with sub-second accuracy; the average reliability in terms of packet delivery ratio is 99.96%; and the energy consumption is less than 9 mAh per day on environmental sensors. Since there are different requirements in every application scenario, there are several modified

versions of TSCH in this regard. One common way to tailor TSCH based on the application requirements is to have a different approach on the scheduling scheme. Thus, there is rich scientific literature describing different scheduling approaches for TSCH that can be summarized in three main categories: centralized, distributed and autonomous. In centralized scheduling [2], a node is responsible for calculating the schedule and distributing it to the rest nodes. In distributed scheduling approaches, the schedule is constructed based on the relations between neighboring nodes [8]. In the autonomous approaches, each node constructs its own schedule mostly based on routing information that is already present in the node [9].

Several aspects of TSCH have been explored, such as energy consumption [10], scalability [9,11] and latency [12]. However, it is still unclear how a TSCH network performs under a mobile scenario since that topic is not explored thoroughly in the scientific literature. The mobility feature in IoT is emerging. Application scenarios might operate in remote areas where there is no access to a cellular network [13], and protocols such as TSCH are used for local coordination. These applications can be diverse in terms of mobility patterns. For instance, an application might include an autonomous agricultural vehicle (AAV) [14] following a predefined trail. Other applications might operate in the context of warehouse logistics [15], where a synergy of moving machinery, autonomous robots/vehicles and workers might require low-power mobile communication to fulfill the application requirements. In the latter case, the mobility pattern is more random than the former, and the performance of a protocol, such as TSCH, would be different.

In this paper, the main research question we try to answer is: how reliable are the current TSCH schedulers under different mobile scenarios? We consider two mobile scenarios, and based on the simulations, we evaluate three different schedulers to examine the reliability of TSCH. Specifically, we evaluate how mobility affects its reliability in terms of packet received ratio (PRR), downtime and the initial joining time. In addition, we consider how a different number of nodes in a network architecture would impact the network reliability. The results illustrate that the current TSCH approaches do not operate efficiently in those cases, and we provide some proposals towards improving them. As a main evaluation tool, we use the Cooja [16] simulator.

The rest of the paper is organized as follows: Section 2 describes the basic principles of TSCH and the schedulers used in this paper. Section 3 lists the state of the art and how it relates to the presented approach. Section 4 illustrates the technical details and the motivation behind the selected mobility patterns and the representing applications. Section 5 presents the evaluation method and the obtained results, and finally, Section 6 discusses future work and concludes the paper.

2. Background

This section provides a technical overview of TSCH protocol and the TSCH schedulers, which are used in the evaluation part. TSCH is included in the IEEE 802.15.4 [17] standard, which is in an amendment in IEEE 802.15.4e-2015 [18].

TSCH is a link layer protocol that utilizes frequency hopping and synchronization techniques. Essentially, neighboring nodes can transmit and receive packets based on a scheduler that defines the time slot and the channel. In order to avoid collisions, it should not be allowed multiple packet transmissions at the same time and channel. Thus, the schedule is organized in a two-dimensional table called the slotframe. Figure 1 depicts a slotframe along with the assumed topology. The x-axis in the slotframe stands for the time offset, and the y-axis stands for the channel offset. A cell corresponding to a specific time and channel offset is also called timeslot and usually has a duration of 10 ms. In the depicted example in Figure 1, there are six timeslots and five channel offsets. The length of the slotframe is defined by the number of the timeslots, and it is associated with several tradeoffs concerning the network performance. In each cell, every node is allowed to perform one of the following actions, mandated by the schedule: (1) transmit a packet; (2) receive a packet; or (3) put the radio on sleep mode. For instance, in the mentioned

example, node B can send a packet to node E at cell (0, 4); node C can send a packet to node F at cell (0, 2); and the rest of the actions can be executed as defined from the schedule.

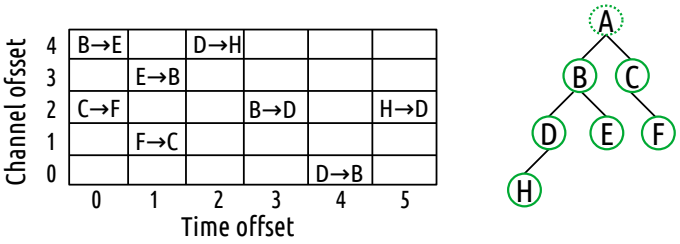

Figure 1. TSCH slotframe on the left and routing topology on the right.

The TSCH coordinator is responsible for the construction of a TSCH network. More specifically, it defines the network ID, slotframe size and the frequency-hopping sequence (FHS). It also initializes the absolute slot number (ASN) to 0, which is increased with each new timeslot. In order to join the TSCH network, a node has to receive an enhanced beacon (EB) packet, which is broadcast by the coordinator or other joined nodes. The EB contains important information required, such as the time source to synchronize with the other nodes of the network. A channel for a given cell can be computed through Equation (1) where CO stands for channel offset. If there is more than one transmission cell at the same time, there is a priority mechanism for the cell with more packets.

$$Channel = FHS(ASN + CO) \ mod \ ||FHS|| \tag{1}$$

2.1. Minimal Scheduling Function (MSF)

The RFC 9033 [8] defines a scheduling mechanism for TSCH implemented on the top of the 6 top protocol. MSF includes three different type of cells: *minimal*, *autonomous* and *negotiated* cells. The minimal cells are used to exchange EB packets and routing information; autonomous cells are used for unicast communication; and negotiated cells are used by a node to communicate and announce itself to other nodes that have just joined the TSCH network. We focus on autonomous cells in this paper. With the MSF scheduler, every node defines some cells autonomously. An Rx cell is defined for the node itself using a hash function to decide the timeslot and the channel offset. The other cells of a slotframe are defined as Tx and are defined based on the traffic demands, and they can be deleted if there is no traffic with a neighbor. That is the main difference with Orchestra, which is described in the next schedule we included in the evaluation. If there is no traffic to transmit to a neighbor, the cell is removed and can be used for other purposes. If there are more than one transmission cell allocated in the same timeslot, a mechanism gives priority to the cell with the more packets.

2.2. Orchestra

Orchestra [9] was introduced in 2015. It is an autonomous scheduler for TSCH. One of the basic principles of Orchestra is that it combines different slotframes for MAC, routing and application purposes. Orchestra includes several types of operation. The receiver-based (RB) and sender-based (SB) types are the most popular ones, which depend on storing the routing information to schedule unicast cells from parent to child. Each node is allocating a unicast cell for its own demands (in RB type, the cell is Rx and in SB, Tx). The rest cells are Tx and Rx accordingly and allocated on a cell by hashing the neighbor's address. The slotframes for the assuming node B from Figure 1 are illustrated in Figures 2 and 3 for RB and SB, respectively. For the RB case in Figure 2, we see that initially a cell schedules a transmission to its parent, then a cell is devoted to its own receiving demands and a transmit cell to each of its children. On the other hand, in Figure 3, we see that node B schedules a node to receive from its parent first, then a cell for itself to transmit and then a receiving cell for each child. A hash function using the neighbor address is constructing the schedule. For evaluation purposes, we use the RB type of operation. Every Orchestra

timeslot includes two more slotframes, one for the default timeslot, which is shared among all nodes, and another one for EB packets, which is broadcast unicast.

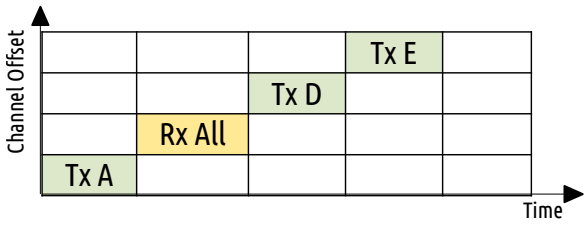

Figure 2. Slotframe for the receiver-based approach of orchestra for node B from Figure 1.

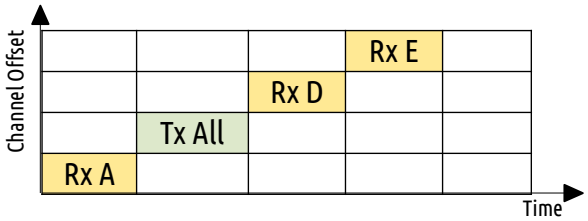

Figure 3. Slotframe for the sender-based approach of orchestra for node B from Figure 1.

2.3. Alice

Alice [19] is based on Orchestra but instead of calculating a schedule based on the node perspective, it calculates it based on a link basis. Thus, it deploys a link-based channel offset instead of regular channel offset. ALICE interacts more tightly with RPL routing, and a node can have more RPL neighbors (parent and children) in order to ensure more unique TSCH cells. Moreover, upstream and downstream traffic is not interfering with each other. Then, several channels can be used at the same time, and there is no additional overhead for calculating the cell schedule. Another difference is that the unicast cells are updated once in every slotframe based a hash function that is time-dependent. A slotframe is represented in Figure 4, which considers node B from Figure 1. We see that it is first scheduled a cell for transmitting towards its parent, and the next cell is for receiving from its parent. Afterwards, the node transmits to one of its children, and the next cell is scheduled to receive from it. Finally, we see that in the same timeslot, a transmission and a reception cell is allocated at different channels. This is one of the new features of ALICE that enables upstream and downstream traffic at the same time.

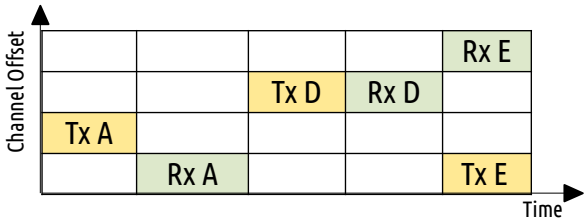

Figure 4. Slotframe for the link-based approach in the Alice schedule for node B from Figure 1.

3. Related Work

This section presents the state of the art of TSCH operating in mobile scenarios. Unfortunately, there are very few instances focusing on this topic; thus, this motivates us to investigate this topic further in order to provide insight and new solutions to these research questions.

Al-Nidawi et al., in [20], investigate the performance of TSCH and LLDN protocols under mobility scenarios. More specifically, the authors focus on the overhead that joining and leaving the network causes in terms of delay and energy. One of the observations is that if a slotframe has a large channel offset, the scanning procedure will take more time

and cost more energy since there are more channels to scan. This issue can be countermeasured with a suitable timeslot and scheduling approach. In addition, it is illustrated that TSCH can operate in mobile scenarios and does not consume more power if there are a sufficient number of nodes to provide good network coverage. In a follow up paper [21], the same authors introduce Mobile TSCH, which is designed to reduce the delay that occurs when joining a TSCH network. The authors take advantage of ACK packets, which are transmitted on a specific channel to advertise the TSCH network. In other words, the ACK packets behave as an enhanced beacon (EB) packet as well. The EB packets in TSCH are used to advertise the TSCH network, and a node requires to successfully receive one in order to join the network. Thus, the scanning procedure takes place in a single channel, and the delay and the energy consumption decreases.

A more recent attempt to investigate TSCH mobility is described in [22]. The authors focus on cases where mobile nodes are leaving and re-joining a TSCH network due to issues that are related with the synchronization degree. They carry out an evaluation based on simulations that takes into account a combination of stationary and mobile nodes. The metrics used are downtime (the time a node is disconnected from the TSCH network), the energy consumption and the amount of keep alive (KA) packets (KA packets are used to update the synchronization of a node). Moreover, they regulate the number of the nodes and the speed to see their impact on TSCH performance. The results describe a tradeoff between the downtime and the consuming energy connected with the operating speed. Essentially, an increased speed might decrease the downtime, but it will increase the energy consumption. Among other results, they present that when the coverage of the network is sufficient with an adequate number of mobile nodes, TSCH performance is very close to a static TSCH network.

Instant, a TSCH schedule for mobile nodes, is introduced in [23]. Instant focuses on a combination of stationary and mobile nodes to to provide both indoor localization and data collection services in the context of healthcare for residential environments. The novel part in this schedule is that the mobile nodes are able to know information about reservations of subsequent unicast cell block of the stationary nodes in advance through a simple probe–ACK transaction. Thus, if they wish to transmit a large amount of data, they use this information to select an available stationary node.

An older approach in the same spirit, with a combination of mobile and stationary nodes, is described in [24]. The main strategy described here is to divide the routing strategy between the stationary and mobile nodes. The stationary nodes use the RPL protocol and ETX metric to evaluate its link. The mobile nodes use their end-to-end ETX estimation in combination with a blacklisting process that considers future position inaccuracies. A number of simulations show that the reliability of the network increases compared to the classical TSCH approach.

The aforementioned approaches describe TSCH performance based on very basic mobility patterns and do not consider the state of the art of the TSCH schedulers. Furthermore, these approaches consider a single mobility pattern in their evaluation method, but as mentioned later, mobility in IoT can vary a lot as it is dictated from several spatio-temporal variables. In contrast, the results presented in this paper are obtained considering the latest TSCH schedulers, and we consider two mobile scenarios that differ in terms of randomness between them. Furthermore, the mobility patterns are inspired and designed based on real-life application scenarios; thus, the results reflect these cases more.

4. Mobility Patterns and Applications

This section describes the details of the mobility patterns used and the motivation behind it. The mobility patterns among IoT application scenarios vary a lot and include several spatio-temporal variables that are difficult to model in a general context. To this end, we defined two scenarios that we will focus on. The first one represents an application where an AAV operates on the field, and the nodes follow a specific trail. The nodes are supposed to be part of a fleet collaboration system, enabling and allowing multiple

autonomous agricultural robots to collaborate with each other as well as human-operated tractors and machinery. TSCH is envisioned as part of a multiple-protocol solution with robust safety degradation features.

In the second one, a random movement mobility pattern represents moving machinery, autonomous robots/vehicles and workers who communicate in a smart warehouse environment. The patterns were implemented in Python and then imported in the Cooja simulator for the evaluation part. For the random movements in the smart warehouse pattern, we used the random library from Python with a uniform distribution.

4.1. Autonomous Agricultural Vehicle Mobility Pattern

In this mobility pattern, the nodes move in a 1000 × 1000 m frame in a predefined trail, as shown in Figure 5. A node can start at any point of the predefined trail and move in any direction. When it reaches the end, it will start moving to the opposite direction. We assume that when two nodes are moving against each other, they keep a safe distance between them. The pattern will continue until the end of the simulation. In a 4-h simulation, which we execute, the nodes repeat the trail approximately 20 times. The purpose here is to check if the different TSCH schedulers are able to cope with the dynamic links that are created from such a mobility pattern. This particular mobility pattern can be described as more deterministic since the nodes follow a repeating pattern several times.

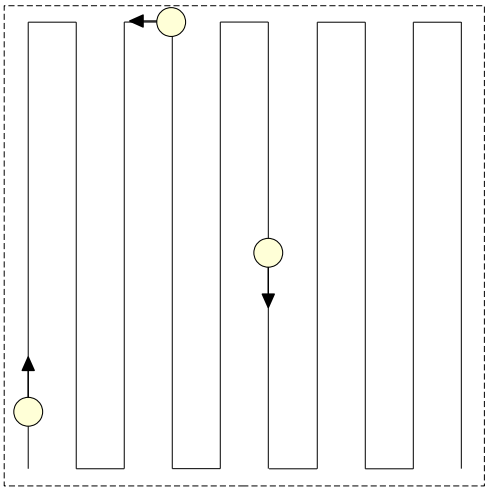

Figure 5. The mobility pattern that represents the function of autonomous agricultural vehicles operating on a field.

4.2. Smart Warehouse Mobility Pattern

The nodes in this mobility pattern move in a same frame of 1000 × 1000 m, and it can start from any position. Then there are some limitations on which directions a node can move in order to stay within the defined frame. For example, when a node is the upper left corner, the available next moves are either right or down, and the probabilities to choose each direction is 50%. Accordingly, when a node is on the limit of the left side for instance, the available moves are right, down and up, and the probability to choose in that case is 33.3%. If the node is at any other position, it can move up, down right or left with a probability of 25%. The probabilities are summarized in Table 1. A uniform distribution was used to ensure an equal probability for each direction every time. The distance a node is moving every time is from 50 m to 500 m, with steps of 10 m and a probability of 2.22% for selecting each resulting distance. The probability here is calculated from a uniform distribution where the node may choose 45 available options from 50 m to 500 m with steps of 10 m. With this pattern, we intended to represent how a node will move in a warehouse, with different movements every time, and create a more random pattern to compare how the TSCH schedulers perform in this scenario.

Table 1. Available moves of the nodes and probabilities to choose them in the warehouse mobility pattern.

Current Position	Available Moves	Probability
Corner	2	50%
Side	3	33.3%
Not corner and not side	4	25%

5. Evaluation

This section presents the details of the evaluation method we used, the results we obtained and a discussion about the findings.

To simulate the proposed scenarios, we used Cooja [16], a popular network simulator that is part of Contiki-OS [25]. To enable mobility, we also used a mobility plugin for Cooja [26]. We first created several trails in Python following the mobility patterns described in Section 4. In this sense, every node will follow its own trail following a specific mobility pattern. The field a node can move is 1000 × 1000 m, the transmission range is 450 m and the mobility speed 2 m/s. The simulation was based on Cooja motes, a virtual platform offered by the simulator that is able to execute Contiki-OS as a native process and directly utilize all the hardware accesses. The duration time of each simulation was 4 h. The slotframe size and the channel offset for the MSF scheduler are defined using a hash function, which takes into account the amount of neighbors. Orchestra in our evaluation has a size of 397 slots for the EB slotframe, 31 for the broadcast slotframe and for the Unicast slotframe, it is constructed by the same hash function described before. For Alice, the EB and Broadcast slotframe sizes are the same the Unicast; however, it is constructed again every time the slotframe is completed based on directional link information and multiple channel offsets.

We focus on a scenario where the traffic is directed from nodes to the coordinator. In this scenario, every node is transmitting a packet every 6 s. To have statistically valid results, we executed the same simulation with 20 different seeds for each selected scheduler, and we changed the number of nodes to observe how it can affect the performance of TSCH in each one. We conducted 360 simulations in total, which correspond to 1440 simulation hours and more than 3,161,000 packets that were transmitted. However, the initial positions of the nodes remain the same in order to systematically investigate when coverage issues may or may not occur. For instance, the mobility pattern representing the AAV case includes three to nine nodes. In every simulation for the AAV case, there is a node with severe coverage issues, as after a point, it moves away from the rest of the nodes. This node has a hard time re-joining the network due to the fact that there is an increased time delay to join the TSCH network, and the time this node interacts with the rest of the nodes is not enough to complete this process. This is not happening in the smart warehouse mobility pattern, where none of the nodes spent significant time outside the network.

5.1. Packet Received Ratio

The first metric we focus on is the PRR, which is the number of transmitted packets to the number successfully received, including the retransmissions. Figures 6 and 7 represent the obtained results for the two mobility patterns accordingly. The results are presented with box plot figures where we group the TSCH scheduling approaches as we differentiated the amount of operating nodes in the x-axis.

We observe high variance in Figure 6, where we evaluate the AAV scenario. This is happening because most of the nodes achieve a similar amount of PRR, but the one that moves to a remote position has a significant lower value and the boxes in the plot represent the values of all the nodes in the network.

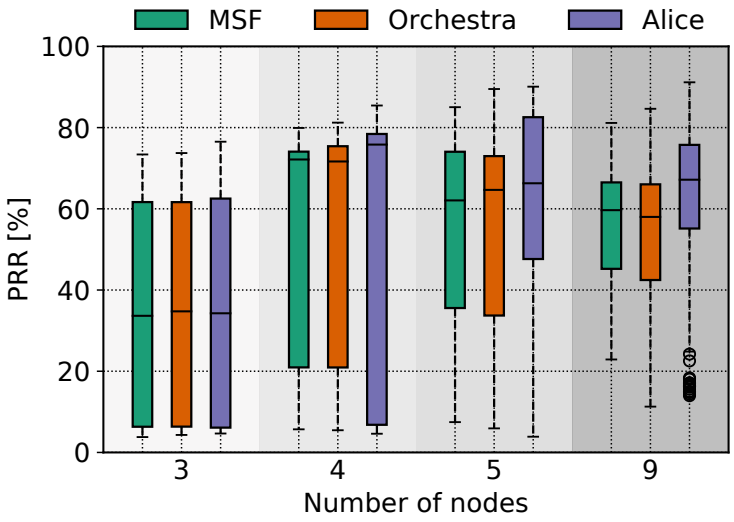

Figure 6. PRR for different TSCH schedulers under the autonomous agricultural vehicles' mobility pattern for a different number of nodes.

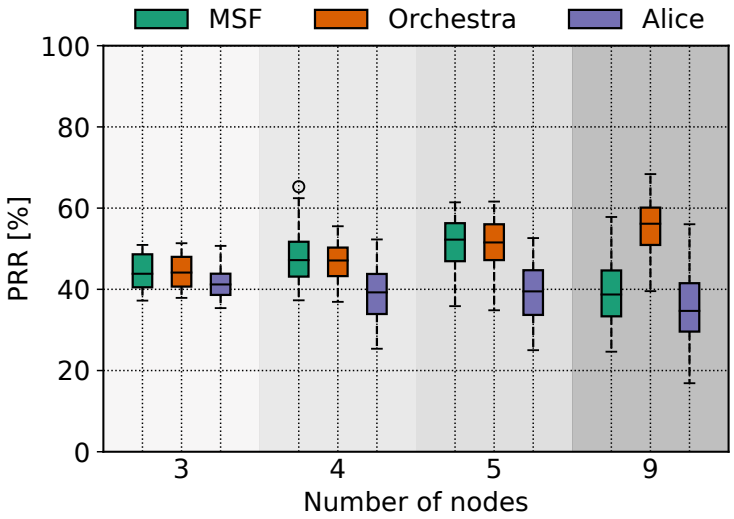

Figure 7. PRR for different TSCH schedulers under the smart warehouse mobility pattern for a different number of nodes.

The next observation has to do with the number of operating nodes. One might have assumed that if you increase the number of nodes, the coverage would increase, and therefore, the PRR would be higher [22]. While we believe this is a valid argument and this is a trend we see in Figures 6 and 7, we also observe that the PRR value decreases in Figure 6 when we increase the number of nodes from 4 to 5 and 5 to 9. This is not happening because one the new nodes has an increased downtime value, which is confirmed by Figure 8, but rather because of the TSCH mechanics and new traffic demands because of the new nodes joining the network. As downtime, we define the time a node is not associated with the TSCH network. This can happen either due to the fact that a node is out of range because of the mobility pattern but also due to synchronization or high traffic and scheduling issues. It is important to highlight that when TSCH is used in mobile scenarios, the downtime is not only occurring out of coverage issues but also because of scheduling approaches and low synchronization issues. Apparently, these problems contribute to a decrease in the network performance as well.

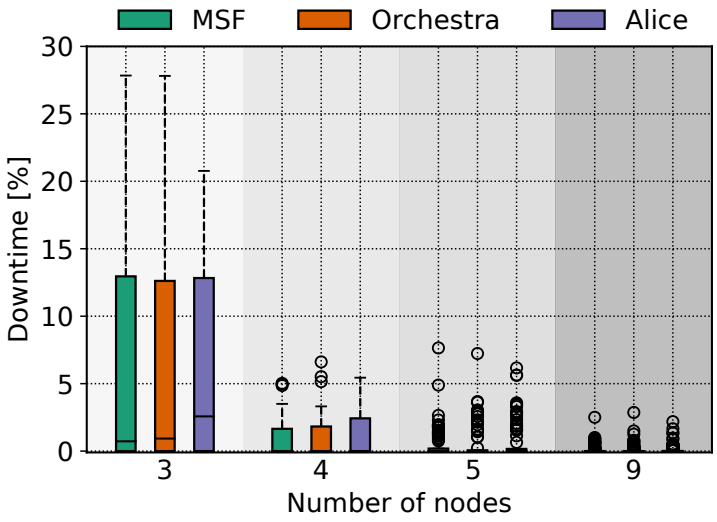

Figure 8. Downtime for different TSCH schedulers under the autonomous agricultural vehicles' mobility pattern for a different number of nodes. The amount of downtime is the percentage of a 4-h simulation.

Regarding the different schedulers we used, we do not observe any significant difference in the PRR. ALICE performs slightly better for the AAV mobility pattern, but for the case of the smart warehouse mobility pattern, ALICE performs slightly worse than the other two scheduling approaches. We speculate that the directional link approach that ALICE employs is more efficient for a more deterministic mobility pattern than a more stochastic one, such as the smart warehouse one.

Next, if we compare the PRR between the two mobility patterns, we will observe that there is a different performance, and it is difficult to compare in a holistic manner. For instance, if we compare the PRR values when three nodes were operating in the AAV case in Figure 6, the PRR values are very low in that case because one of the three nodes is having significant downtime and consequently very low PRR compared to the ones in the smart warehouse case in Figure 7. If we compare now the case where there are four nodes operating in Figure 6, we see that the PRR values are increased by a large degree because the newly introduced node operates within the network range and its PRR value is able to compensate in the mean value depicted in the box plot.

The main remark is that the PRR values are generally very low for all cases, and the current TSCH approaches cannot provide a reliable and robust performance when they are used in a mobile scenario.

5.2. Downtime

Downtime is a crucial metric that points out the time a node might spend not associated with the TSCH network. As it was mentioned before, besides coverage issues, other issues, such as going out of synchronization or high traffic, might lead a node to leave the network. Being out of coverage is a problem that cannot be solved in a straight forward way, but leaving the TSCH network because of synchronization issues or because the scheduler does not include enough EB cells during a mobile scenario is a less complicated problem to solve.

In the cases we investigated, the downtime was rather low in general, specifically for the smart warehouse mobility pattern, as it is reported very close to 0 in Figure 9, because the initial positions and the mobility pattern was such to enable adequate coverage for the TSCH network. For the AAV case in Figure 8, the downtime is increased in some cases since one of the nodes has increased downtime on purpose.

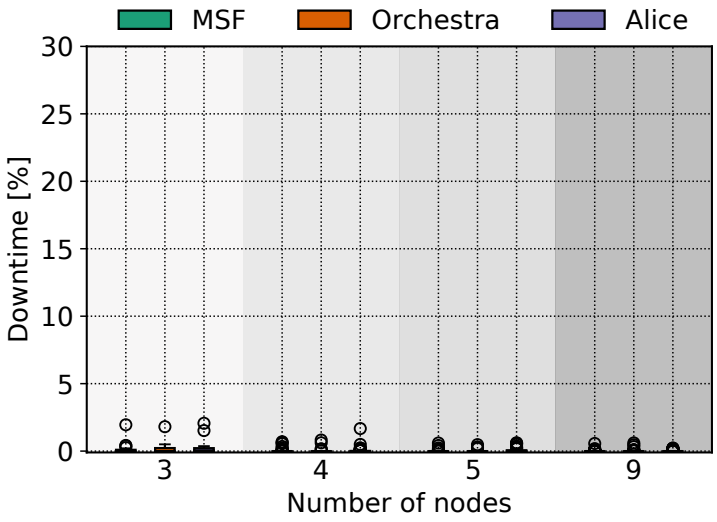

Figure 9. Downtime for different TSCH schedulers under the smart warehouse mobility pattern for a different number of nodes. The amount of downtime is the percentage of a 4-h simulation.

5.3. Initial Join Time

We also report the initial jointime metric, which is the amount of time it takes a node to join the TSCH network from the time it boots. This is an important metric since it is known that for some TSCH approaches, it takes a notable amount of time to join the TSCH network. If we combine this with a mobile scenario, the amount of time can be increased. Figures 10 and 11 illustrate the results.

The average jointime in the AAV case in Figure 10 is lower compared with the warehouse case in Figure 11; even though the downtime is larger for the AAV, the jointime is lower due to the fact that the node is moving out of coverage after joining the TSCH network.

In Figure 11, we see that when the operating nodes are three, the initial jointime is increased, but when the nodes are increased to four, the jointime drops very close to 0. We speculate that in the second case, the new node act as the mediator to provide coverage to the rest of the nodes in the beginning and helps them to associate with the TSCH network faster.

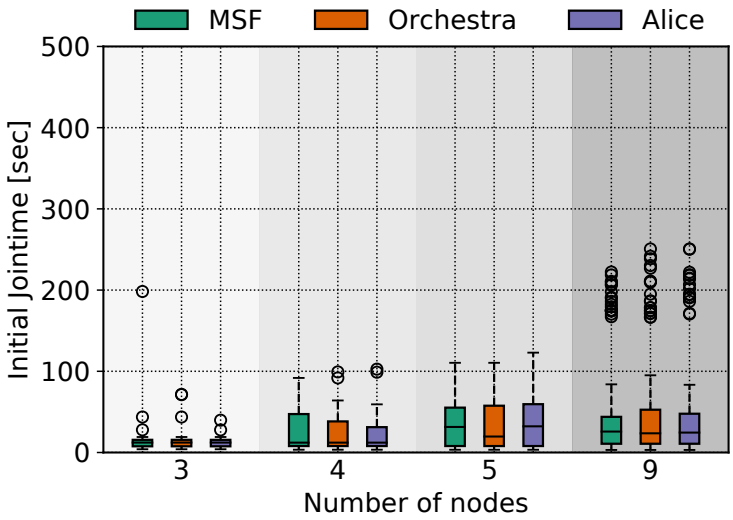

Figure 10. Jointime for different TSCH schedulers under the autonomous agricultural vehicles mobility pattern for a different number of nodes.

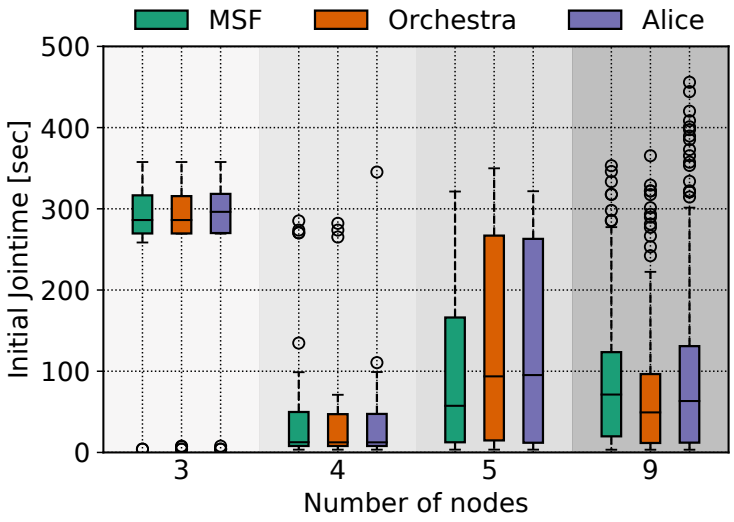

Figure 11. Jointime for different TSCH schedulers under the smart warehouse mobility pattern for a different number of nodes.

We also have to mention the case when a node will be deployed in an initial position, which might be out of coverage regarding the rest nodes. The chances of this node joining the TSCH network is based on how much time it will interact with the rest of the nodes, and if the joining time of the regarding TSCH approach is increased, the node will have fewer chances to join the network. To this end, it is important to have a short joining time because in cases such as these, the chances of this node reporting the sensed data will be based on if it will manage to join the network during this short interval.

6. Conclusions

In this paper, we report how TSCH schedulers perform during heterogeneous mobile scenarios and where we should focus to improve them for such cases. We defined two mobility patterns representing two different mobile applications. The first one is more deterministic, as it consisted of repetitive patterns, and the second one is more stochastic. We run simulations after selecting TSCH approaches from the state of the art and perform evaluation terms of PRR, downtime and the initial join time. The results show that the current approaches cannot be utilized in mobile scenarios as their performance is inadequate.

One of the main conclusions is that the PRR was on average less than 80% for all combinations of mobility patterns and schedulers. The downtime metrics were different among the two mobility patterns but less than 3% for all cases. The downtime values for the smart warehouse scenario in Figure 9 are lower than the AAV one in Figure 8. Even though the nodes in the smart warehouse scenario spent less time out of coverage, their regarding PRR values are lower than the AAV one. This indicates that the downtime metric is not the only factor to diminish the performance of TSCH during mobile scenarios. The joining time is on average 20*sec* for the AAV case, which is a considerable time interval if the nodes are moving fast and have short interaction intervals to join the TSCH network.

The main contribution of this paper is identifying the directions researchers should concentrate on in order to improve TSCH performance towards mobile scenarios. There are several approaches to improve TSCH performance utilized in mobile scenarios. As we highlight in this paper, the mobility patterns and the application scenarios are important factors to consider. In cases where the downtime is high, the mobility pattern should be examined for nodes that operate too much time out of coverage. A solution in that case might be the deployment of a static node close to the isolated node or a different approach might be if an adjacent node can change its mobility pattern to keep the isolated node within the network coverage. If the downtime is low, reducing the joining time or

implementing a different scheduling approach might improve the performance since, in that case, the problem is that the nodes are moving in a pattern that the generated schedule is able to follow and it is outdated most of the times. We plan to explore these approaches in future work.

Author Contributions: Conceptualization, C.O. and X.F.; methodology, C.O. and X.F.; software, C.O.; validation, C.O., X.F., A.E. and P.P.; formal analysis, C.O.; investigation, C.O.; resources, C.O.; data curation, C.O.; writing—original draft preparation, C.O.; writing—review and editing, C.O, A.E., P.P. and X.F.; visualization, C.O.; supervision, X.F. and P.P.; project administration, X.F. and P.P.; funding acquisition, X.F. and P.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially funded by Innovation Fund Denmark, grant 9092-00007B AgroRobottiFleet. It was also supported by the ERDF Activity 1.1.1.2 "Post-doctoral Research Aid" (No. 1.1.1.2/VIAA/2/18/282).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The published data are available in the following repository https://github.com/haorfani/Results.git (accessed on 21 October 2021).

Acknowledgments: We thank Athanasios Theocharis for the assistance with the simulation scripts, which he prepared during their MSc thesis.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Duquennoy, S.; Elsts, A.; Nahas, B.A.; Oikonomou, G. TSCH and 6TiSCH for Contiki: Challenges, Design and Evaluation. In Proceedings of the 2017 13th International Conference on Distributed Computing in Sensor Systems (DCOSS), Ottawa, ON, Canada, 5–7 June 2017; pp. 11–18. [CrossRef]
2. Chen, D.; Nixon, M.; Mok, A. *WirelessHART(TM): Real-Time Mesh Network for Industrial Automation*; Springer: Berlin/Heidelberg, Germany, 2010.
3. Watteyne, T.; Diedrichs, A.L.; Brun-Laguna, K.; Chaar, J.E.; Dujovne, D.; Taffernaberry, J.C.; Mercado, G. PEACH: Predicting Frost Events in Peach Orchards Using IoT Technology. *EAI Endorsed Trans. Internet Things* **2016**, *2*. [CrossRef]
4. Elsts, A.; Fafoutis, X.; Woznowski, P.; Tonkin, E.; Oikonomou, G.; Piechocki, R.; Craddock, I. Enabling Healthcare in Smart Homes: The SPHERE IoT Network Infrastructure. *IEEE Commun. Mag.* **2018**, *56*, 164–170. [CrossRef]
5. Brun-Laguna, K.; Diedrichs, A.L.; Dujovne, D.; Taffernaberry, C.; Léone, R.; Vilajosana, X.; Watteyne, T. Using SmartMesh IP in Smart Agriculture and Smart Building applications. *Comput. Commun.* **2018**, *121*, 83–90. 03.010. [CrossRef]
6. Piyare, R.; Oikonomou, G.; Elsts, A. TSCH for Long Range Low Data Rate Applications. *IEEE Access* **2020**, *8*, 228754–228766. [CrossRef]
7. Elsts, A.; Fafoutis, X.; Oikonomou, G.; Piechocki, R.; Craddock, I. TSCH Networks for Health IoT: Design, Evaluation, and Trials in the Wild. *ACM Trans. Internet Things* **2020**, *1*. [CrossRef]
8. Chang, T.; Vučinić, M.; Vilajosana, X.; Duquennoy, S.; Dujovne, D. 6TiSCH Minimal Scheduling Function (MSF). RFC 9033. 2021. Available online: https://rfc-editor.org/rfc/rfc9033.txt (accessed on 21 October 2021).
9. Duquennoy, S.; Al Nahas, B.; Landsiedel, O.; Watteyne, T. Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH. In Proceedings of the 13th ACM Conference on Embedded Networked Sensor Systems, Seoul, Korea, 1–4 November 2015; Association for Computing Machinery: New York, NY, USA, 2015; pp. 337–350. [CrossRef]
10. Vilajosana, X.; Wang, Q.; Chraim, F.; Watteyne, T.; Chang, T.; Pister, K.S.J. A Realistic Energy Consumption Model for TSCH Networks. *IEEE Sens. J.* **2014**, *14*, 482–489. [CrossRef]
11. Hajizadeh, H.; Nabi, M.; Tavakoli, R.; Goossens, K. A Scalable and Fast Model for Performance Analysis of IEEE 802.15.4 TSCH Networks. In Proceedings of the 2019 IEEE 30th Annual International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC), Istanbul, Turkey, 8–11 September 2019; pp. 1–7. [CrossRef]
12. De Guglielmo, D.; Al Nahas, B.; Duquennoy, S.; Voigt, T.; Anastasi, G. Analysis and Experimental Evaluation of IEEE 802.15.4e TSCH CSMA-CA Algorithm. *IEEE Trans. Veh. Technol.* **2017**, *66*, 1573–1588. [CrossRef]
13. Codeluppi, G.; Cilfone, A.; Davoli, L.; Ferrari, G. LoRaFarM: A LoRaWAN-Based Smart Farming Modular IoT Architecture. *Sensors* **2020**, *20*, 2028. [CrossRef] [PubMed]
14. Zhao, W.; Wang, X.; Qi, B.; Runge, T. Ground-Level Mapping and Navigating for Agriculture Based on IoT and Computer Vision. *IEEE Access* **2020**, *8*, 221975–221985. [CrossRef]
15. Liu, X.; Cao, J.; Yang, Y.; Jiang, S. CPS-Based Smart Warehouse for Industry 4.0: A Survey of the Underlying Technologies. *Computers* **2018**, *7*, 13. [CrossRef]

16. Osterlind, F.; Dunkels, A.; Eriksson, J.; Finne, N.; Voigt, T. Cross-Level Sensor Network Simulation with COOJA. In Proceedings. 2006 31st IEEE Conference on Local Computer Networks, Tampa, FL, USA, 14–16 November 2006; pp. 641–648. [CrossRef]
17. IEEE Standard for Low-Rate Wireless Networks. IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011). 2016. Available online: https://ieeexplore.ieee.org/document/7460875 (accessed on 16 June 2021).
18. IEEE 802 Working Group. IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Std.* **2011**, *802*, 1–314. [CrossRef]
19. Kim, S.; Kim, H.S.; Kim, C. ALICE: Autonomous Link-based Cell Scheduling for TSCH. In Proceedings of the 2019 18th ACM/IEEE International Conference on Information Processing in Sensor Networks (IPSN), Montreal, QC, Canada, 16–18 April 2019; pp. 121–132. [CrossRef]
20. Al-Nidawi, Y.; Yahya, H.; Kemp, A.H. Impact of mobility on the IoT MAC infrastructure: IEEE 802.15.4e TSCH and LLDN platform. In Proceedings of the 2015 IEEE 2nd World Forum on Internet of Things (WF-IoT), Milan, Italy, 14–16 December 2015; pp. 478–483. [CrossRef]
21. Al-Nidawi, Y.; Kemp, A.H. Mobility Aware Framework for Timeslotted Channel Hopping IEEE 802.15.4e Sensor Networks. *IEEE Sens. J.* **2015**, *15*, 7112–7125. [CrossRef]
22. Raza, S.; Lee, T.V.D.; Exarchakos, G.; Güneş, M. A Reliability Analysis of TSCH Protocol in a Mobile Scenario. In Proceedings of the 2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC), Las Vegas, NV, USA, 11–14 January 2019; pp. 1–6. [CrossRef]
23. Elsts, A.; Pope, J.; Fafoutis, X.; Piechocki, R.; Oikonomou, G. Instant: A TSCH Schedule for Data Collection from Mobile Nodes. In Proceedings of the 2019 International Conference on Embedded Wireless Systems and Networks, Beijing, China, 25–27 February 2019; Junction Publishing: Junction, TX, USA, 2019; pp. 35–46.
24. Barcelo, M.; Correa, A.; Vilajosana, X.; Vicario, J.L.; Morell, A. Novel Routing Approach for the TSCH Mode of IEEE 802.15.14e in Wireless Sensor Networks with Mobile Nodes. In Proceedings of the 2014 IEEE 80th Vehicular Technology Conference (VTC2014-Fall), Vancouver, BC, Canada, 14–17 September 2014; pp. 1–5. [CrossRef]
25. Dunkels, A.; Gronvall, B.; Voigt, T. Contiki—A lightweight and flexible operating system for tiny networked sensors. In Proceedings of the 29th Annual IEEE International Conference on Local Computer Networks, Tampa, FL, USA, 16–18 November 2004; pp. 455–462. [CrossRef]
26. Mobility of Nodes in Cooja. Available online: https://anrg.usc.edu/contiki/index.php/Mobility_of_Nodes_in_Cooja (accessed on 16 June 2021).