LOST: Localized Blacklisting Aware Scheduling Algorithm for IEEE 802.15.4-TSCH Networks

Dimitrios Zorbas, Vassilis Kotsiou, Fabrice Théoleyre,
Georgios Z. Papadopoulos, and Christos Douligeris
Department of Informatics, University of Piraeus, Greece
CNRS, ICube, University of Strasbourg, France
IMT Atlantique, IRISA, UBL, France
email: dzorbas@unipi.gr

(this may not be the final version of the paper)

Abstract

Industrial applications require more and more low-power operations, low-delay, deterministic communications as well as end-to-end reliability close to 100%. IEEE 802.15.4-TSCH (Time-Slotted Channel Hopping) relies on a channel hopping technique while scheduling properly the transmissions to provide a high end-to-end reliability. Because of external interference, some channels may perform very poorly locally, which impacts negatively the reliability for some radio links. We propose here the first distributed scheduling solution which reactively allocates the cells to each pair of nodes while also considering local blacklists. These local blacklists are constructed on a per-radio link basis to reflect the actual performance encountered locally. Our simulations highlight the relevance of our distributed blacklisting aware scheduling algorithm to improve both the reliability and the delay efficiency compared with DeTAS, a state of the art distributed solution.

1 Introduction

Internet of Things (IoT) comprise numerous wireless sensor devices deployed for various applications ranging from logistics and smart transportation to e-health or firefighting systems [20]. For more than a decade there have been significant efforts in setting up deployments that require pertinent data collection. In many-to-one topologies, sensor readings are typically forwarded via relay nodes until they reach the sink station, i.e., root of the routing tree that is able to store and process the collected data.

Critical applications within the Industrial 4.0 era require efficient communication among the devices with end-to-end reliability close to 100% [20]. However, the constrained devices combined with the nature of wireless communications and the *a priori* unknown conditions over the deployment area, may present significant communication challenges and, thus, endanger the reliable data collection.

Slow channel hopping MAC such as IEEE 802.15.4-TSCH [1] has been proposed to provide high-reliability at the link-layer. They rely on a strict organization of the different transmissions to eliminate the collisions. A schedule is constructed, such that two interfering transmitters are allocated to different timeslots. Besides, channel hopping allows to combat external interference [19]. While the medium access under TSCH relies on scheduling the transmissions, the

actual algorithm to use is let unspecified in the standard. Therefore, several centralized and distributed scheduling algorithms have been proposed so far.

Because external interference may affect some specific channels [13], channel hopping may be insufficient: the transmissions using the *bad* channels impact negatively the reliability and the energy efficiency. With blacklisting (BL), the bad channels are identified and may be removed from the hopping sequence. By using only the most reliable channels, the reliability is globally improved. However, since a physical channel exhibits location-dependent characteristics [12], the blacklist is radio-link dependent, and should be updated locally. The idea of channel BL is not new since it has already used for other protocols, like WirelessHART. However, there is no localized mechanism to tackle local interference and it is assumed that it is a task globally performed by a systems administrator [14]. Thus, the localized approach presented in this paper may also be used in other communication standards.

To the best of our knowledge, we propose here the first complete localized scheduling algorithm combined with a reactive and localized BL mechanism.

The contributions of this paper are as follows:

1. we present a localized scheduling algorithm to assign the cells (matrix of channel offsets and timeslots) by exchanging information only with the neighbors;
2. we propose a distributed (i.e., per pair of nodes) BL scheme, that can modify its frequency hopping sequence to avoid the bad channels;
3. we use an adaptive over-provisioning scheme in the TSCH scheduler to increase the network reliability.
4. our performance evaluation highlights the relevance of this approach to increase the number of delivered packets while decreasing the end-to-end delay.

2 Related Work

2.1 IEEE 802.15.4-TSCH & Radio Channel BL Techniques

IEEE 802.15.4-2015 has proposed the TSCH mode, where the time is divided into timeslots of equal length. At each timeslot, a node may transmit or receive a frame, or it may turn to sleep mode for saving energy. A slotframe comprises a fixed set of timeslots, repeated cyclically. Each timeslot is labelled with an Absolute Sequence Number (ASN), which counts the number of timeslots since the network was established. Based on the ASN and the schedule, the nodes in the TSCH network decide when to transmit or receive a frame.

IEEE 802.15.4-2015 TSCH implements a channel hopping approach to combat noise and interference and, thus, to achieve high network reliability [19]. To do so, TSCH presents a deterministic scheduling approach in which each cell consists of a pair of timeslot and channel offset. The standard maintains a schedule, and assigns a set of cells to each radio link. At the beginning of a timeslot, the channel offset is translated into a physical channel using the ASN value:

$$frequency = F\Big((ASN + channelOffset) \ \% \ nFreq\Big) \tag{1}$$

where $channelOffset$ is the channel offset of the current cell, $nFreq$ is the number of available channels and $F()$ a bijective function mapping an int comprised between 1 and $nFreq$ into a physical channel [18].

BL consists in excluding the corresponding radio channels to be used for transmissions. To this aim, each pair of nodes must be able to identify accurately the bad channels which impact

the reliability of *its* radio link. Detecting bad channels may use the average ETX (per channel) value [15], or dedicated timeslots to measure the noise level [5].

A blacklist may be applied globally in the whole network [16]. Such approach may be sub-optimal since a physical channel exhibits very location-dependent characteristics [12]. Thus, a link-based blacklist should be preferred to avoid wasting the bandwidth.

With a local blacklist, the transmitter can decide to postpone its transmission [11]. The pseudo-random sequence will give after a while a whitelisted channel. Unfortunately, this technique increases the delay.

2.2 Scheduling algorithms for TSCH

While 6TiSCH defines how the cells are negotiated, any scheduling algorithm may be actually implemented.

The centralized approaches rely on a Path Computation Element (PCE). TASA proposes to construct a schedule for a multihop TSCH network [2]: the same slotframe may be repeated more frequently to increase the network capacity. Dobslaw *et al.* [6] propose to reserve additional timeslots for retransmissions, in order to improve the reliability. MABO-TSCH assigns in a centralized manner a collection of cells for each radio link, while considering local blacklists [9]. Several channel offsets are assigned within the same timeslot, so that a radio link can pick *one* of the channel offset which does not give a blacklisted physical frequency.

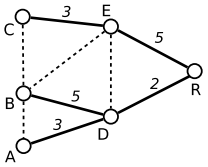
Under distributed algorithms, the nodes exchange control packets only with their neighbors to detect and to eliminate the collisions. DeTAS proposes a decentralized version of TASA [3]: the children of the border routers collect the radio topology of their subtree to compute independently the schedule of their descendants (called micro-schedule). Wave [17] constructs a schedule such that a packet is delivered before the end of the slotframe, even if it has to be relayed by intermediate nodes. Hosni *et al.* [10] organize the transmissions into stratums to upper bound the end-to-end delay, even in presence of retransmissions. SF0 [7] represents the default behavior of 6TiSCH. It reactively computes the number of cells to reserve based on the traffic it has to transmit. Over-provisionning and an hysteresis function help the network to be more stable and reliable. Furthermore, Orchestra was recently proposed [8] to construct a TSCH schedule in a distributed manner. Each node constructs pseudo-randomly its schedule without exchanging any control information with its neighbors.

3 Blacklisting aware scheduling algorithm

In this Section, we present LOST (LOcalized Scheduling algorithm for TSCH networks), an efficient algorithm that establishes schedules and assigns channels for IEEE 802.15.4 networks. LOST is a localized algorithm since it relies on information gathered by its 1-hop neighbors only.

LOST's features are summarized as follows:

1. 1-hop only information is required. No centralized mechanism is involved in both slot allocation and offset assignment;
2. the algorithm multiplexes the different transmissions across different channels by allocating properly the channel offsets;
3. LOST prioritizes the nodes which have a larger number of packets in their queue. This action reduces the probability of buffer overflows, and decreases the delay;
4. No initial or global knowledge of the number of packets of the predecessor nodes is required. This makes the algorithm adaptive to topology changes;



Timeslots	1-5	6-8	9-18	19-21
Links	B-D	A-D	D-R	E-R
	E-R	C-E		
Channel offsets	1	1	0	0
	0	0		

Figure 1: Example topology and the corresponding LOST scheduling.

5. A localized BL method is employed in the scheduler to avoid using the *bad* radio channels;

6. An over-provisioning scheme allocates additional timeslots for retransmissions within the same slotframe to bound the end-to-end delay.

We rely on IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) to construct the routes. LOST allocates the transmission opportunities along these routes to forward all traffic. Each node selects a preferred parent, from the list of possible parents, to which it will forward all its traffic. Each node is also aware of its rank in RPL tree. Inversely, a node receives Destination Advertisement Object (DAO) control packets from its descendants, and forwards them to the border router through its own preferred parent. It exploits these DAO to maintain the list of its children. Each node in the network also acquires the information regarding the maximum degree of the DODAG (Destination Oriented Directed Acyclic Graph) during its construction. Finally, a node maintains a list of its physical neighbors, updated when it receives an Enhanced Beacon (EB).

The scheduling process of LOST consists of several rounds each of them comprising three phases. In each round, each node decides one of the three available states to follow depending on the state of its neighbors.

We consider three types of states: the *requester* requests timeslots from its parent who will act as *respondent*. The respondent is in charge of assigning timeslots and replying to the requester. If a node has no request to send and at the same time it is not respondent, it remains in *sleep* mode. We will discuss about this last state later.

Next, we describe the phases of LOST and how a node individually decides if it will be requester or respondent. Note that the first two phases can be combined in a single phase (transmitting a single control packet), however, we separate the two phases for presentation purposes. To further facilitate the understanding of the algorithm, we introduce the example of Figure 1 consisting of 5 nodes (A-E) all of them transmitting packets to the root (R). The solid lines correspond to the RPL links (node→parent) and the dashed lines to the physical radio links. The number above each link denotes the number of generated packets per node (q_i).

3.1 Dissemination of the Amount of Traffic

The purpose of this phase is to let the neighboring nodes (parent and children) know about the amount of packets that each particular node intends to transmit. 1-hop information is exchanged among neighbors during this phase. The information that is exchanged includes the node id i, the amount of packets q_i that exist in its (virtual) buffer and a priority number $prior_i$. LOST gives priority to the nodes that transmit a higher number of packets and are close to the root in order to avoid buffer overflows. Thus, $prior_i$ is computed as a function of q_i as follows: $prior_i = \frac{q_i}{r_i} + \epsilon$, where r_i is the rank of node i and ϵ is a real very small random number. In our example, nodes B and E have the highest priorities.

3.2 Slots Requests

Every node that receives the neighboring piece of information described in the previous phase can now decide if it will be a requester or respondent in this round. A node can be a requester if it has a higher priority than its parent. A requester sends its request to its parent using its id, priority, the number of requested slots, and the last already assigned timeslot (if any). In our example, nodes A, B, and E will be requesters and D, R will be respondents. Finally, C will wait the next round to request slots.

3.3 Slots Allocation

Finally, a respondent who has already received one or more requests, will rank the requests according to their priority, starting from the highest. It will then check its first available slot and will assign a number of slots equal to the requested slots for each of the (priority-ranked) requested children. A reply is sent to its children with the allocated slots. In the example of Figure 1, D will allocate slots 1-5 for B and slots 6-8 for A. On the other hand, R will allocate slots 1-5 for E.

When a node receives a reply from its respondent, it sets its q value to 0. On the other hand, the respondent increases its q value accordingly: a given flow will reserve hop by hop a sufficient amount of cells. This 3-step procedure is repeated until all the nodes have allocated a number of slots at least equal to the number of packets of their predecessor nodes plus their initial packets. Note that each node is aware of this information as the algorithm evolves from the leaf nodes to the root. A node knows that its allocation role is completed in the algorithm if it does not receive any other requests within the next round from the time it set its q value to 0.

Then, a node has to decide which channel offset to use for each of its cells.

3.4 Channel offset assignment

We have now to assign the channel offsets so that two interfering nodes which selected the same timeslot use a different channel offset. We consider a collision may arise because (i) at least one node from one link is physical neighbor with at least one node of the other link, or (ii) the sender of one link is 2-hop RPL neighbor with the sender of the other link.

LOST follows a distributed approach to assign the channel offsets and avoid the collisions. Initially all the nodes have assigned a channel offset equal to 0. Each node follows a two-step procedure as soon as it has already allocated the timeslots for its packets. A node has the right to decide which channel offset to use, when it has already received the channel offsets of all its predecessor RPL nodes. This means that the channel assignment is a recursive procedure that starts from the leaf nodes and moves on to the root.

During the first step, we keep on exploiting the local priorities to solve the conflicts with the physical neighbors. A node extracts the list of timeslots already reserved by its neighbors with an higher priority. If the timeslots reserved by the node, and these higher priority neighbors are the same, it has to select a different channel offset. Thus, it selects the next available offset (e.g., B will choose the channel offset 1 in figure 1). By exploiting the set of priorities, we are sure to order the decisions, and that only one node will change its schedule when a conflict is detected.

The second step ensures that no 2-hop neighbors transmit at the same time with the same offset. To achieve this, a node checks the reservations received from its children (which contain also the reservations of the grand-children). Similarly, a node checks if the same timeslots have been already reserved by one of these 2-hop neighbors. If an overlap is detected, the node selects a different channel offset. Once a node has reserved its cells (timeslots and channel offset) and has already completed the two steps, it can safely notify the root with a control packet transmitted

in unicast to the preferred parent, forwarded along the routing tree. This way, we ensure to avoid the conflicts: a node *locks* virtually its 2-neighborhood.

3.5 Adaptive timeslot provisioning

LOST allocates additional timeslots per link in order to increase the communication reliability and to bound the end-to-end delay in the case of failures. These extra timeslots are used as backup slots giving the opportunity to the sender to re-transmit a packet that was not acknowledged previously. The strong point of LOST's over-provisioning mechanism is that it is adaptive to network traffic demands and communications failures.

In LOST, the number of extra timeslots depends on the Packet Error Rate (PER) between the sender and the receiver. PER can be computed by evaluating the link quality at the beginning of the process. Note, that if PER is not available, other link quality estimators can be used like the RSSI and the SNR. Eq. (2) describes the extra amount of slots q_i^{PR} that will be requested by the requester i during the second phase of the algorithm.

$$q_i^{PR} = int\left(\alpha_i \left(\frac{PER_{ij}}{max\_PER}\right)^2 q_i\right), \tag{2}$$

where α_i is a constant that describes the necessity of the provisioning ($0 \leq \alpha_i \leq 1$), PER_{ij} is the packet error rate of transmitter i and receiver j, and max_PER is the maximum error packet rate. Note that α_i is a node-dependent constant and it characterizes the link quality between i and its parent. α_i can be modified in future executions of the algorithm (re-schedules) according to the number of extra slots that were actually used during the previous slotframes. In particular we update α_i as follows:

$$\alpha_i^{FS+1} = \frac{\alpha_i^{FS} FS + \frac{reserved\_slots - unused\_slots}{reserved\_slots}}{FS+1}, \tag{3}$$

where $reserved\_slots$ is the number of slots reserved for transmitting q packets ($q \leq reserved\_slots$), $unused\_slots$ is the number of assigned slots that have finally not used, and FS is the number of the frameslot.

3.6 Overhead

LOST exhibits a low overhead mainly because 1-hop only information is exchanged. The number of exchange messages per round depends on the number of participating nodes. In fact, $N-1$ messages are needed for the first two phases of the algorithm, where N is the number of nodes including the sink. Moreover, each parent replies to its children requests sending a message with the schedule to each child. Finally, for the offset assignment two messages are required per node (excluding the sink); one for the 1-hop physical neighbors and one to forward the reservation to the parent. Thus, the total number of exchange messages per round is $4N-3$.

We must note here that the number of exchange messages can be highly reduced if we assume that each node waits for the reception of the schedules of its children before allocating its own timeslots. However, this could dramatically increase the number of data packets in the buffer, causing buffer overflows, delays, and longer schedules.

3.7 Blacklisting Aware Channel Hopping

At the beginning of each timeslot, an active node (either transmitter or receiver) has to select the actual physical frequency to use. The standard specifies it is derived from the channel offset

Algorithm 1: Localized Blacklisting method, multiple channel offsets assignment

require: $max\_degree, first\_offset$
1 $OFFSETS = \emptyset$;
2 $step = max\_degree$;
3 **for** $s = first\_offset; s < 16; s \mathrel{+}= step$ **do**
4 $\quad OFFSETS = OFFSETS \cup \{s\}$;
5 **return** $OFFSETS$;

and the ASN using the eq. 1 (cf. section 2.1). A pair of nodes has to agree on the blacklist to use: inconsistent blacklists may make the receiver *deaf*, impacting negatively the reliability. On the other hand, even if two parallel links have different channel offsets, eq. (1) may still generate the same physical frequency for both links from time to time in absence of a local BL mechanism.

To tackle this problem, two solutions are possible: either globally blacklisting bad channels or the list of blacklisted channels is exchanged locally. The disadvantage of the first case is that all the nodes must have a consistent view of the blacklists used by the other ones, thus, the information needs time to be propagated to the whole network. Moreover, since interferences exhibit a very local pattern, it is not necessary to distribute these channels to the rest of the deployed areas. Finally, extra messages need to be injected in the network increasing the energy cost.

In this paper, we use a localized BL method to keep bad channels locally without distributing them to the whole network. This method enhances the channel offset assignment phase described in Section 3.4 by allowing a node to assign multiple offsets. This procedure is described in Algorithm 1. In particular, every node keeps track of a set of available channel offsets (i.e., $OFFSETS$) generated by a simple function based on the maximum vertex degree in network. The maximum vertex degree is used so that an adequate number of channel offsets is produced even for high density areas (areas with possibly many parallel links) [4]. If $first\_offset$ is the channel offset assigned by the channel offset process of Section 3.4, a few other offsets are generated with distance *step* with each other. Since $first\_offset$ is different for each parallel link and *step* is the same for all the nodes, Eq. 1 always generates different channels for all parallel links. If a generated channel is blacklisted, a new channel is generated using the next available offsets until a whitelisted channel is finally selected. If none whitelisted channel is generated using all the possible offsets, the node postpones its transmission. Note that each pair of nodes must retain the same blacklist to communicate properly. To do so, neighboring nodes can encapsulate blacklists in a data or in an acknowledgement packet without increasing considerably the payload and thus the energy consumption [9].

Thus, B and A of Figure 1 will assign offsets 1, 5, 9, and 13, while the rest of the nodes will assign 0, 4, 8, and 12.

4 Performance Evaluation

In this Section, we evaluate the LOST algorithm and we compare its performance against DeTAS [3]. We use 4 flavors of LOST; one with both BL and provisioning (designated as **LOST**), one with BL and without provisioning (**LOST-NPR**), one with provisioning and without BL (**LOST-NBL**), and one without BL or provisioning (**LOST-NBL-NPR**). The purpose of distinguish these four flavors is to assess the affect of BL and over-provisioning schemes to the algorithm performance.

The results are obtained using a set of Monte Carlo simulations with a simulator written in Perl programming language. The topologies were generated using the DeTAS' terrain generator with a 200x200 m^2 terrain size, random node positions, and a communication range of 50m [3]. Each node generates a random number of packets per slotframe in the range [1,5]. Due to the high number of generated packets, the length of the schedule may be higher than the 101 timeslots described by the IEEE 802.15.4 TSCH standard (even without provisioning). Therefore, we have increased the number of available timeslots per slotframe to 301. Moreover, we use a probability P that a packet is dropped due to external interference for each of the 16 available channels. The following values were used: P_{11}=0.3, P_{12}=0.4, P_{13}=0.4, P_{14}=0.3, P_{15}=0.01, P_{16}=0.3, P_{17}=0.4, P_{18}=0.4, P_{19}=0.01, P_{20}=0.01, P_{21}=0.2, P_{22}=0.4, P_{23}=0.4, P_{24}=0.01, P_{25}=0.01, and P_{26}=0.01. Finally, we set α=0.5 for all the nodes. A channel is blacklisted if the Packet Delivery Ratio (PDR) falls bellow 0.9. We vary the number of nodes and we measure the PDR and the total packets delayed within 50 slotframes.

4.1 PDR

In Figure 2, the network-wide PDR performance under various densities is illustrated. As it can be observed, the proposed LOST scheme achieves PDR above 99%, more than any other LOST configuration or DeTAS scheme. The results demonstrate the significant adding value of both BL and over-provisioning mechanisms, i.e., LOST-NBL-NPR presents a PDR below 80%. It is worth mentioning that LOST outperforms DeTAS even without these mechanisms. This is mainly because of the low channel diversity of DeTAS which results to higher number of collisions.

4.2 Delay

Next, we evaluate the delay performance of LOST scheme. For successfully delivered packets, LOST's end-to-end delay is at the same level as that of DeTAS. However, on average it is considerably affected by failures on the path to the sink. We assume that a packet is delayed if it was not delivered within an interval of one slotframe. Our simulation results show that the network density and, consequently, the additional traffic in the network do not impact negatively LOST in terms of delay. As it can be observed from Figure 3, LOST presents a very low and stable performance. On the other hand, the results show that without employing BL and over-provisioning, LOST-NBL-NPR linearly degrades its performance with increase of density. Finally, DeTAS once again presents very poor results, it is straightforward that it can not handle high level of traffic in the presence of interference.

4.3 Impact of Over-provisioning

We study here the impact of over-provisioning method on the size of LOST's schedule. Note that in this case α is initially set to 0.5 and it is updated according to Eq. 3. Figure 4 illustrates LOST's schedule size during the initialization phase, at the end (i.e., re-scheduling after 50 slotframes), and without over-provisioning. As it can be observed, after 50 slotframes, the schedule converges and adapts to the requested traffic conditions. Moreover, the more dense is the network, the more efficient is the converged schedule. Indeed, in case of 50 nodes in the network, the size of the schedule after re-scheduling and without over-provisioning is similar. As a result, this evaluation shows the adaptability of our method and the minimum impact of over-provisioning on the schedule, while improving essentially the network performance in terms of PDR and delay.

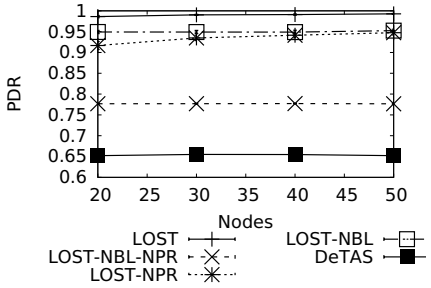


Figure 2: Packet Delivery Ratio for different node populations.

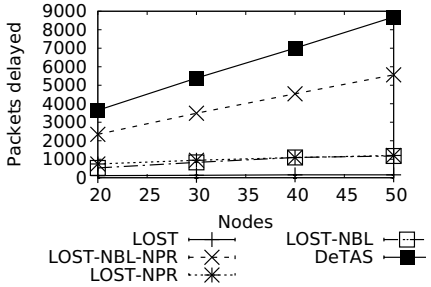


Figure 3: Number of delayed packets for different node populations.

4.4 Impact of α

We focus here on the impact of α on the network reliability. To this aim, we evaluate the PDR performance of LOST scheme with and without over-provisioning over α. We present two different use-cases: *i)* each node transmits on average three packets per slotframe (i.e., slotframe size of 301 timeslots), *ii)* each node transmits on average one packet per slotframe (i.e., slotframe size of 101 timeslots). As it was expected, LOST without over-provisioning, presents stable PDR performance, since it has no impact from α. On the other hand, low α values do not allow many packet retransmissions leading to a slightly lower PDR compared to middle-range values. On the contrary, large α values lead to very long schedules increasing the number of nodes that are unable to send their packets within the desired slotframe.

4.5 Overhead

Finally, Figure 6 presents the overall LOST's overhead for different node populations. We can observe that the number of messages increases linearly with the number of nodes. About 6 messages are needed per node to build the schedule (allocate slots and channels offsets).

5 Conclusions & Future Work

We presented here a distributed scheduling solution, which allocates the timeslots and channel offsets. We assign a set of priorities so that a cell is reserved by the highest priority node, which *locks* its usage in its neighborhood. Besides, we also combine a BL technique to not use the *bad* channels, which provide a poor reliability. Our simulations highlight the superiority of this approach compared with DETAS, a reference solution for distributed scheduling in TSCH. By avoiding both collisions and the usage of the locally bad channels, LOST is able to multiplex

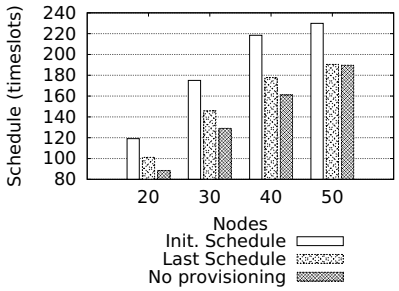
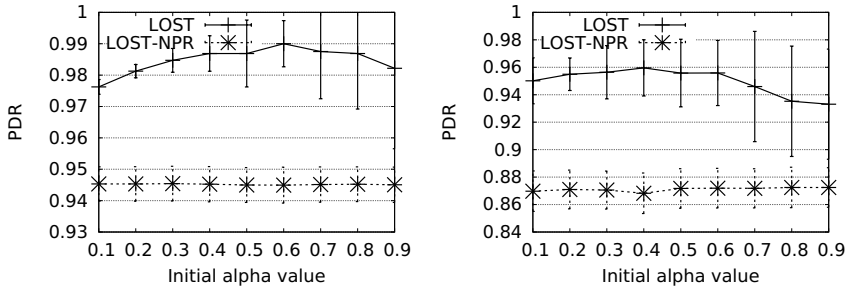


Figure 4: Schedule lengths before and after rescheduling.



(a) 3-packets per node on average (slotframe size = 301 timeslots)

(b) 1-packet per node (slotframe size = 101 timeslots)

Figure 5: PDR for different α values and slotframes sizes.

efficiently the transmissions and to improve the network reliability. In the future, we plan to evaluate the performance of LOST in a testbed, where the conditions are dynamic. We conjecture that our over-provisioning and BL mechanisms will help to guarantee high-reliability.

References

[1] IEEE Standard for Low-Rate Wireless Personal Area Networks (LR-WPANs). IEEE Std 802.15.4-2015 (Revision of IEEE Std 802.15.4-2011), April 2016.

[2] M.R. Palattella, et al. On optimal scheduling in duty-cycled industrial iot applications using IEEE802.15.4e TSCH. *Sensors Journal, IEEE*, 13(10):3655–3666, Oct 2013.

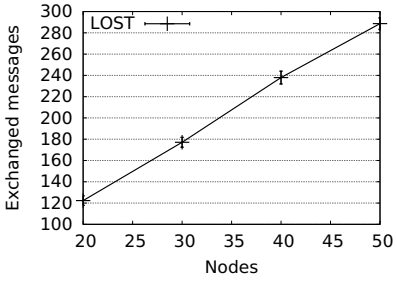


Figure 6: LOST's overhead for different node populations.

[3] N. Accettura, M.R. Palattella, G. Boggia, L.A. Grieco, and M. Dohler. Decentralized traffic aware scheduling for multi-hop low power lossy networks in the internet of things. In *WoWMoM*, pages 1–6. IEEE, 2013.

[4] Béla Bollobás. Chromatic number, girth and maximal degree. 24:311–314, 1978.

[5] F. Chiti, R. Fantacci, and A. Tani. Performance evaluation of an adaptive channel allocation technique for cognitive wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 2017, in press.

[6] F. Dobslaw, T. Zhang, and M. Gidlund. End-to-end reliability-aware scheduling for wireless sensor networks. *IEEE Transactions on Industrial Informatics*, 12(2):758–767, April 2016.

[7] D. Dujovne, L. A. Grieco, M. R. Palattella, and N. Accettura. 6tisch 6top scheduling function zero (sf0). draft 4, IETF, July 2017. `https://tools.ietf.org/html/draft-ietf-6tisch-6top-sf0-04`.

[8] Simon Duquennoy, Beshr Al Nahas, Olaf Landsiedel, and Thomas Watteyne. Orchestra: Robust mesh networks through autonomously scheduled tsch. In *Sensys*, pages 337–350. ACM, 2015.

[9] P H Gomes, T. Watteyne, and B. Krishnamachari. MABO-TSCH: Multihop and blacklist-based optimized time synchronized channel hopping. *Transactions on Emerging Telecommunications Technologies*, August 2017.

[10] Inès Hosni and Fabrice Théoleyre. Self-healing distributed scheduling for end-to-end delay optimization in multihop wireless networks with 6tisch. *Computer Communications*, 110:103 – 119, 2017.

[11] ISA-100.11a-2011:. Wireless systems for industrial automation:process control and related applications. *International Society of Automation (ISA) Std.*, 1, May 2011.

[12] Vasileios Kotsiou, Georgios Z. Papadopoulos, Periklis Chatzimisios, and Fabrice Theoleyre. Is Local Blacklisting Relevant in Slow Channel Hopping Low-Power Wireless Networks? In *ICC*. IEEE, 2017.

[13] G. Z. Papadopoulos, A. Gallais, G. Schreiner, E. Jou, and T. Noel. Thorough IoT testbed Characterization: from Proof-of-concept to Repeatable Experimentations. *Computer Networks*, 119:86–101, 2017.

[14] S. Petersen and S. Carlsen. Wirelesshart versus isa100.11a: The format war hits the factory floor. *IEEE Industrial Electronics Magazine*, 5(4):23–34, Dec 2011.

[15] M. Sha, G. Hackmann, and C. Lu. Arch: Practical channel hopping for reliable home-area sensor networks. In *RTAS*. IEEE, 2011.

[16] Jianping Song, Song Han, A.K. Mok, Deji Chen, M. Lucas, and M. Nixon. WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control. In *RTAS*. IEEE, 2008.

[17] Ridha Soua, Pascale Minet, and Erwan Livolant. Wave: a distributed scheduling algorithm for convergecast in IEEE 802.15.4e TSCH networks. *Transactions on Emerging Telecommunications Technologies*, 27(4):557–575, October 2016.

[18] T. Watteyne, M. Palattella, and L. Grieco. Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement. RFC 7554, 2015.

[19] Thomas Watteyne, Ankur Mehta, and Kris Pister. Reliability through frequency diversity: Why channel hopping makes sense. In *PE-WASUN*. ACM, 2009.

[20] L. D. Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, Nov 2014.