Journal of
Sensor and Actuator Networks

MDPI

Article

Relayer-Enabled Retransmission Scheduling in 802.15.4e LLDN—Exploring a Reinforcement Learning Approach

Andreas Willig *, Yakir Matusovsky and Adriel Kind

Department of Computer Science and Software Engineering/Wireless Research Centre,
University of Canterbury, Private Bag 4800, Christchurch 8041, New Zealand; yakir.m@gmail.com (Y.M.);
adriel.kind@canterbury.ac.nz (A.K.)
* Correspondence: andreas.willig@canterbury.ac.nz; Tel.: +64-3-364-2987 (ext. 7869)

Academic Editor: Màrio Alves
Received: 7 April 2017; Accepted: 31 May 2017; Published: 3 June 2017

Abstract: We consider the scheduling of retransmissions in the low-latency deterministic network (LLDN) extension to the IEEE 802.15.4 standard. We propose a number of retransmission schemes with varying degrees of required changes to the LLDN specification. In particular, we propose a retransmission scheme that uses cooperative relayers and where the best relayer for a source node is learned using a reinforcement-learning method. The method allows for adapting relayer selections in the face of time-varying channels. Our results show that the relayer-based methods achieve a much better reliability over the other methods, both over static (but unknown) and over time-varying channels.

Keywords: IEEE 802.15.4e; LLDN; retransmission scheduling; cooperative relayers; reinforcement learning

1. Introduction

Industrial wireless sensor networks (IWSN) have recently received considerable attention [1–3]. In particular, the IEEE 802.15.4 standard [4] is of great interest to researchers and practitioners, not the least due to its maturity and the commercial availability of components [5]. In 2012, the IEEE approved the IEEE 802.15.4e amendment to the IEEE 802.15.4 standard [6,7]. The amendment includes the *low latency deterministic network* (LLDN) extension, which targets applications in the domain of factory automation where determinism in the time domain is important. In a nutshell, the LLDN extension specifies a star network in which a number of sensors are associated with a coordinator, and where deterministic and low-latency transmission is achieved through a TDMA-like (Time-Division Multiple Access) medium access control scheme. In this scheme, time is partitioned into subsequent superframes, which, in turn, are partitioned into time slots, most of which are allocated for exclusive use to sensors (for uplink traffic) and to the coordinator (for downlink traffic). LLDN also addresses reliability concerns by including both individual and group acknowledgements and explicit retransmission slots.

In this paper, we explore how to use these retransmission slots to improve reliability for uplink traffic. According to the IEEE 802.15.4e amendment, each sensor gets one timeslot for an initial uplink transmission and at most one more timeslot for a retransmission (we call a source node whose initial transmission failed a **failed node** or a **failed source**). We argue that this allocation scheme is inappropriate, for the following reasons: (i) a failed source node can not use more than one retransmission slot, even if other retransmission slots are unused; and (ii) it ignores that the channels between different sources and the coordinator can be very different and can vary over time. We propose and evaluate the performance of a number of different schemes for allocating retransmission slots to

failed nodes. One of the proposed schemes requires no changes to the IEEE 802.15.4e packet formats (only a behavioural change), other schemes only require modifications of one particular LLDN packet, the **group acknowledgement** (GACK). Most of our schemes are designed to maximize the probability that all source packets are eventually received within one superframe (we refer to this as the **success probability**). To achieve this, the coordinator continuously maintains estimates of the current packet error rates on the source-coordinator channels to inform retransmission slot allocation.

Our results indicate that it is possible to improve the success probability significantly already by simply allowing the system to use all available retransmission slots (and without requiring any changes to packet formats). However, much more substantial gains can be made by adding **cooperative relayers**, i.e., dedicated helper nodes tasked to overhear source transmissions and performing retransmissions on their behalf, leveraging spatial diversity [8]. It has already been shown in several other works that adding relayers to TDMA-based systems can substantially improve reliability (e.g., [9]). However, in many of these works, relay scheduling has been carried out under the assumption that all channels are known and time-invariant, or relayer selection is made instantaneously without exploiting what might have been learned about the efficacy of different relayers in the past. In this paper, we consider the situation where the channels are generally unknown and time-varying, and we propose a scheme in which the coordinator **learns** about the quality of different relayers to support the sources. The proposed scheme is based on algorithms to solve the multi-armed bandit problem, a well-studied problem in the field of reinforcement learning (RL) [10,11], and is designed to adapt to time-varying channels. Depending on the number of relayers, substantial improvements in the success probability can be achieved. We explore the performance characteristics of our learning-based scheme under a range of channel models and parameter settings.

The paper is structured as follows: in the next Section 2, we provide background information on LLDN and describe our system model. In Section 3, we introduce the retransmission scheduling schemes that do not employ relayers, and, in Section 4, we introduce the relayer-based retransmission schemes: one involving a practically feasible learning algorithm, and a genie-aided scheme for performance comparisons. All performance results have been obtained by simulation, and the simulation framework is explained in Section 5. The following Section 6 then contains our results. The learning scheme proposed in this paper uses one particularly important system parameter for generating actions, the so-called system temperature. In Section 6.1, we present simulation results justifying our choice of the system temperature value for the remaining paper. In Section 6.2, we compare the proposed schemes for unknown but static channels, in Section 6.3, we consider the case of time-varying channels, and in Section 6.4, we compare variants of the learning-based scheme with larger spaces of available actions against a genie-aided relaying scheme, and, through this, we are able to quantify the "system loss" from the operation of the learning scheme. Related work is summarized in Section 7 and our conclusions are given in Section 8.

This paper is a substantially extended and revised version of the conference contribution [12]. The main differences include the following: (i) the learning-based scheme has been generalized to allow for larger action spaces; (ii) another baseline scheme, the genie-aided scheme, has been added, allowing for assessment of the performance of the revised learning-based scheme in more detail; (iii) additional results exploring the choice of the system temperature parameter for the learning-based scheme and investigating the effect of a larger action space for the learning-based scheme have been added.

2. Background and System Model

2.1. Background on IEEE 802.15.4e LLDN

LLDN uses a TDMA-like medium access control scheme where time is partitioned into superframes. Two different superframe structures are specified for LLDN, which, for the purposes of this paper, are equivalent. We use the superframe structure shown in Figure 1. It starts with a beacon packet transmitted by the coordinator, followed by two optional management slots (e.g., for node

association). Next, there are K uplink slots in which each of the K sensors has an initial transmission of its uplink data packet. In the following group acknowledgement (GACK) packet, the coordinator broadcasts a bitmap indicating the received uplink packets, using one bit for each source node. A configurable number N of retransmission uplink slots follows the GACK. According to the amendment, the first retransmission slot is allocated to the failed source coming first in the bitmap, the second slot is allocated to the failed node coming second, and so on. Hence, a failed source will not get more than one retransmission slot. Up to N retransmission slots are available, and if there are more than N failed sources, then the last ones will miss out. Lastly, there is optionally a number of so-called bi-directional slots that can be allocated to the coordinator or to sensor nodes. In this paper, however, we focus entirely on the uplink and retransmission slots and ignore the management and bi-directional slots.

In the second superframe format specified in the amendment, the group acknowledgement is part of the beacon packet and retransmission slots are placed at the beginning of the next superframe.

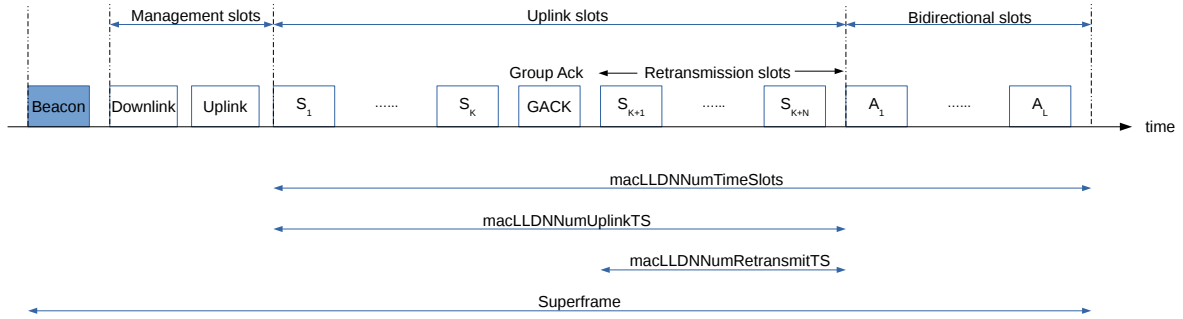

Figure 1. Superframe structure (with separate group acknowledgement), compare ([6] Figure 11g).

2.2. System Model

We look at a single LLDN network having a star topology. There is a single PAN (Personal Area Network) coordinator, a number K of sensor nodes, and R relay nodes. We assume that the relay nodes are truly distinct from the sensor nodes, but they can also be integrated with source nodes [13]. The network has already reached steady-state operation, i.e., all nodes are registered with the coordinator and have started transmitting data packets. There are N retransmission slots, no management slots and no bi-directional slots. To cover the worst case, we assume that each source node has a new packet at the beginning of each superframe, which needs to be successfully transmitted before the superframe ends.

To focus this study on the effects of the actual retransmission scheduling methods, we assume that all transmissions coming from the coordinator (beacons, GACK packets) are completely and reliably received by all other nodes. However, the other channels (uplink channels from sources and relays to the controller, channels between sources and relays) can introduce packet errors. We assume that these channels are pairwise statistically independent, and in order to simplify the channel model, we assume that all data packets have the same length, so that it is meaningful to assign packet error rates to individual channels. In applications where sensor nodes transmit sensor data, this assumption is realistic, as sensor readings are often very small and make up only a relatively small part of a packet, which needs to have physical layer preambles, headers, checksums and other overheads. There is no external interference in the system.

We use two channel models. In the **static channel model**, the packet error rate for each channel is drawn randomly from a uniform distribution between 0% and 100% and remains fixed throughout. In the **time-varying channel model**, each channel changes between two different packet error rates (each drawn randomly from a uniform distribution between 0% and 100%) following a two-state time-homogeneous Markov chain with two identical average state holding times.

3. Non-Relaying Schemes

In this section, we describe the retransmission slot allocation schemes that do not use relay nodes.

3.1. Standard-Based Schemes

The **standard scheme** (Std) implements the allocation of retransmission slots according to the IEEE 802.15.4e LLDN specification (see Section 2.1). Each failed source gets at most one retransmission slot (exactly one if there are at least as many retransmission slots as there are failed sources). When there are fewer failed sources than retransmission slots, some of the retransmission slots remain unused. The **enhanced standard scheme** (EnhStd) does not need any changes to the GACK packet format, but a change in the behaviour of sources and the controller. If we have N retransmission slots and sources $s_1, \ldots, s_m$ have failed (ordered according to their position in the GACK bitmap), the retransmission slots are allocated in a repeating cycle as $s_1, s_2, \ldots, s_m, s_1, s_2, \ldots, s_m, s_1, s_2, \ldots$ until all N slots are exhausted. Hence, sources can get more than one retransmission slot and all N retransmission slots are fully utilized.

3.2. The Optimal(PAR) Scheme

In this subsection and the next, we introduce two schemes that use an estimate of the current packet error rate (PER) for each source-controller channel. For each source node i, the controller maintains a PER estimate $p_i(t)$ that is updated in each superframe immediately after the initial source transmissions and before the allocation of retransmission slots and subsequent transmission of the GACK frame (the time index $t \in \mathbb{N}$ counts the superframes). If the controller did not receive the packet from source i, it encodes the outcome as $o_i(t) = 1$—otherwise (in case of successful reception) as $o_i(t) = 0$. The PER estimate is updated using an exponentially-weighted moving average (EWMA) algorithm:

$$p_i(t) = \alpha \cdot o_i(t) + (1 - \alpha) \cdot p_i(t - 1), \tag{1}$$

where $0 < \alpha < 1$ is a parameter and the initial values $p_i(0)$ are set to zero. This is a well-known method for PER estimation and has the ability to adapt to changing channels. In this paper, we assume throughout that $\alpha = 0.03$, i.e., most weight is put on the "history" summarized in $p_i(t-1)$.

In the **optimal(PAR) scheme** (OptPAR, PAR is a shorthand for "Probability that All packets are Received"), the controller considers all possible allocations of N slots to the M failed nodes, and calculates for each such allocation the probability that the controller receives all packets successfully (see Equation (3)). The controller retains the allocation which maximizes this measure. Due to its computational complexity, we do not consider this scheme as an option for practical implementation, since the number of such allocations is ([14] Section II.5):

$$\binom{M + N - 1}{N}. \tag{2}$$

However, we have included this scheme to compare its performance with that of the (much more practical) heuristic(PAR) scheme, which is discussed next.

3.3. The Heuristic(PAR) Scheme

In the **heuristic(PAR) scheme**, the current PER estimates $p_i(t)$ again play a role in allocating the N available retransmission slots to the M failed sources in the current superframe. For our presentation, we simply number the failed sources from 1 to M.

According to our assumptions, we have M statistically independent wireless links, and on each link i packet errors happen independently of each other with PER p_i (we drop the dependence on time for notational convenience). We are furthermore given N retransmission slots and we assume

that $N > M$ holds. (For $N \leq M$, we simply assign one retransmission slot to each of the first N failed sources.). Suppose that station i gets allocated n_i retransmission slots, in which station i simply repeats its packets n_i times. (Note that, for simplicity, we ignore feedback from the controller here. In practice, when a source node gets positive feedback after fewer than n_i retransmissions, it may stop. However, this will only impact the energy consumption of the source node and not its reliability, which is the main focus of this paper.). Then, the probability that node i's packet fails to reach the controller is $p_i^{n_i}$, and the probability that the controller receives **all** packets becomes (by independence of channels/sources):

$$\Pr\left[\text{Success}\right] = \prod_{i=1}^{M} \left(1 - p_i^{n_i}\right). \tag{3}$$

Assuming that $p_1, \ldots, p_M$ are given, we want to pick numbers $n_1, \ldots, n_M \in \mathbb{N}$ that maximize this expression (equivalently: its logarithm) and which obey $n_1 + \ldots + n_M = N$ to make sure that all N slots are allocated. Hence, we get the following integer optimization problem:

$$\begin{aligned} \text{maximize} \quad & f(n_1, \ldots, n_M) = \sum_{i=1}^{M} \log\left(1 - p_i^{n_i}\right), \\ \text{s.t.} \quad & \sum_{i=1}^{M} n_i = N, \\ & n_i \in \mathbb{N}, \\ & n_i \geq 1 \qquad (i \in \{1, \ldots, M\}), \end{aligned} \tag{4}$$

where the last condition ensures that each failed source gets at least one retransmission slot. Since integer optimization in general is NP-hard [15], we have developed a heuristic based on the relaxation of this problem (ignoring the constraint $n_i \geq 1$ for the time being) and the Lagrange multiplier method [16]. The Lagrangian of this problem is

$$L(n_1, \ldots, n_M, \lambda) = f(n_1, \ldots, n_M) + \lambda \left(\sum_{i=1}^{M} n_i - N \right), \tag{5}$$

where $\lambda \in \mathbb{R}$ is the Lagrange multiplier. To apply the Lagrange multiplier theorem ([16] Thm. 3.1.1), the partial derivatives of the Lagrangian are needed:

$$\begin{aligned} \frac{\partial}{\partial n_i} L(n_1, \ldots, n_M, \lambda) &= \frac{-\log(p_i) \cdot p_i^{n_i}}{1 - p_i^{n_i}} + \lambda, \\ \frac{\partial}{\partial \lambda} L(n_1, \ldots, n_M, \lambda) &= \sum_{i=1}^{M} n_i - N. \end{aligned}$$

With the abbreviation $c_i = \log(p_i)$, the equation $\frac{\partial}{\partial n_i} L(\cdot) = 0$ can be solved for $n_i = n_i(\lambda)$ as:

$$n_i(\lambda) = \frac{\log\left(\frac{\lambda}{c_i + \lambda}\right)}{c_i}. \tag{6}$$

Plugging this expression for n_i into the second condition $\frac{\partial}{\partial \lambda} L(\cdot) = 0$ yields:

$$0 = \sum_{i=1}^{M} \frac{\log\left(\frac{\lambda}{c_i + \lambda}\right)}{c_i} - N. \tag{7}$$

This needs to be solved for λ to find the multiplier and subsequently the n_i (from Equation (6)). For $M > 1$, there is in general no closed-form expression for λ and we need to resort to numerical

computation. Since the functions $n_i(\cdot)$ from Equation (6) should return positive values, the fact that $c_i < 0$ implies that $0 < \frac{\lambda}{c_i+\lambda} < 1$ must hold, which in turn requires $\lambda < 0$. Furthermore, noting that for $\lambda < 0$ the function $n_i(\cdot)$ is monotonically increasing, and using $\lim_{\lambda\to-\infty} n_i(\lambda) = 0$ and $\lim_{\lambda\to 0-} = \infty$, we can conclude that the right-hand side of Equation (7) has exactly one root λ^*, which can be found efficiently using a bisection method. Please note that solutions with $n_i(\lambda^*) < 1$ cannot be ruled out.

With these building blocks in place, we can now describe our heuristic(PAR) algorithm. Assume that there are M failed sources and N retransmission slots. The current PER estimates for the failed nodes are given by $p_1, \ldots, p_M$, and we first compute the Lagrange multiplier λ^* as the unique negative root of Equation (7). Then, set

$$n_i^* = \lfloor n_i(\lambda^*) \rfloor , \tag{8}$$

as our initial guess for the optimal $n_1, \ldots, n_M$. The function $\lfloor x \rfloor$ returns the largest integer $\leq x$.

There can be cases where

$$N' := \sum_{i=1}^{M} n_i^* < N$$

holds and some slots are not used. We first allocate one of the unused slots to each source i with $n_i^* = 0$ while possible, and update their slot counters to $n_i^* = 1$. After doing this, when there are still $N'' = N - \sum_{i=1}^{M} n_i^* > 0$ unused slots, we run the following algorithm to allocate these N'' slots based on the "allocation gap":

---

while $N'' > 0$:
  $j := \arg\max_{i\in\{1,\ldots,M\}} (n_i(\lambda^*) - n_i^*)$
  allocate slot to failed source j
  $n_j^* := n_j^* + 1$
  $N'' := N'' - 1$

Computationally, the most complex step in this algorithm is finding the root of a strictly monotonically increasing function (Equation (7)), which can be done efficiently to arbitrary precision using a bisection method. We surmise that the entire algorithm is quick enough so that the resulting allocation is available when the controller starts to construct the GACK packet. However, the GACK packet needs to be extended: beyond the acknowledgement bitmap, it needs to indicate for each failed source i the number n_i^* of retransmission slots it gets. When N is not too large, these numbers only need a few bits per failed source.

4. Relaying Schemes

In this section, we introduce the two relaying schemes considered in this paper. The first is the learning-based scheme, the second is an idealized genie-aided scheme introduced for comparison purposes.

4.1. Learning-Based Scheme

Diversity schemes, in particular spatial diversity schemes, are a key approach to improving transmission reliability over wireless channels [8,17]. In spatial diversity schemes, multiple spatially separated antennas are used to transmit or receive information. In the special case of cooperative communications, the required antennas are "borrowed" from third-party nodes, called **relayers** [18,19]. A particularly interesting application of the cooperative communications concept is to incorporate relay nodes into retransmission-based error-control schemes, where a relayer performs retransmissions on behalf of a source node, provided the relay node has managed to overhear the original data packet (see [9,20] for relaying in a TDMA context).

We assume that, besides the K source nodes, there are R separate relay nodes in the network. The relays are switched on all the time to overhear packets transmitted by source nodes. In our scheme, it is the controller which decides which relay will have to support a given source node. Note, however,

that the controller has initially no information about the (generally time-varying) channels between the sources and relays, and between the relay nodes and the controller.

The decision scheme used to allocate a relayer to a failed source has some similarities to learning-based algorithms used to solve the multi-armed bandit problem, a well-known problem in reinforcement learning (RL) [10,11]. Broadly, the controller needs to balance two different goals: on the one hand, it should consistently apply the action (choosing a relay node and its number of slots) that is currently known to be the best one (this is called **exploitation** in the literature); on the other hand, it has to test other actions from time to time to see whether they give better results than previously thought and to update the knowledge about the quality of other actions (this is called **exploration**). Exploration is particularly important for time-varying channels. A standard approach is to switch probabilistically between exploration and exploitation, and, in this paper, we do this by using the Boltzmann distribution for action selection (see below).

Our learning-based scheme uses the heuristic(PAR) scheme as a starting point and we refer to it as the **learning(PAR) scheme**. Important design goals are simplicity (so it can be executed in real time) and quick convergence towards optimal actions for given channel conditions. In the light of the convergence requirement, we have decided to keep the space of available actions small; in particular, we allow only one relayer to support a failed source within a superframe and ignore the possibility to build elaborate relaying chains involving two or more distinct relayers.

The controller runs a separate instance of our algorithm for each source node, and, in the following, we consider a fixed failed source node i. After the initial transmissions, the controller first invokes the heuristic(PAR) scheme to calculate an initial allocation of retransmission slots to failed nodes (note that here the current channel PER estimates come in). For our failed node i, we denote by n_i the number of allocated retransmission slots, and regard this as its *state*. Therefore, the state space is given by the possible number of retransmission slots a failed node can get, which ranges from 1 to N.

In each state, the controller picks an action from an action set. An action a in state $s \in \{1, \ldots, N\}$ is given by a pair $a = (r, m)$ specifying a relayer r and a number m of contiguous slots given to the relayer. For $s = 1$, we require $m = 0$ and for $s > 1$, we restrict m to be from the set $\{1, \ldots, \min\{s-1, \Delta\}\}$, where Δ is a protocol parameter specifying the maximum number of retransmission slots that can be allocated to a relayer. Note that the first retransmission slot is always allocated to the failed source i. When such an action is chosen, in the resulting slot allocation, the first $s - m$ slots are allocated to the failed source i, and the remaining m slots are allocated to relayer r (which is assumed to be only transmitting i's packet if it has overheard it previously, and otherwise remains silent). Besides the actions involving a relay node r, there is one further action in state s, which is to give all retransmission slots to the failed source i.

The controller stores for each source node i a separate table with all allowed state/action pairs. The table entries contain the average reward $Q(s, a)$ for state s and action a. Suppose that, in superframe t, the data packet of a particular source i has been received by the controller already after the initial transmissions. In this case, the table entries for this source remain unchanged. Otherwise, if source i is a failed source and has received s retransmission slots according to the heuristic(PAR) algorithm, the controller picks the action a for state s randomly, using a Boltzmann distribution (see below). After executing the s retransmission slots according to action a, the controller determines the outcome o: if the controller has eventually received the data packet of source i, it assigns $o = 1$ (indicating success); otherwise, it assigns $o = 0$. The value $Q(s, a)$ is then updated following an EWMA-type approach as

$$Q(s, a) := \alpha_r \cdot o + (1 - \alpha_r) \cdot Q(s, a), \tag{9}$$

where $0 < \alpha_r < 1$ is an adjustable parameter. Again, the EWMA-type reward update scheme helps with adaptation to time-varying channels. After a preliminary simulation-based performance evaluation, in this paper, we have fixed α_r as 0.05, i.e., most weight is on the "history".

There are different (randomized) methods to pick an action a in a given state s. In the conference version [12], we have used a scheme in which, with a fixed probability ϵ, the best available action for

state s is chosen (i.e., the action a maximizing $Q(s,a)$), and, with probability $1-\epsilon$, an action is chosen randomly with uniform distribution. However, we found that better performance could be achieved when we use a Boltzmann distribution with a given temperature parameter $\tau > 0$ to pick an action, and this is what we use in this paper. In this method, the action a for given state s is always chosen randomly according to the following distribution:

$$\Pr\left[\text{Action} = a^*\right] = \frac{\exp\left(\frac{Q(s,a^*)}{\tau}\right)}{\sum_a \exp\left(\frac{Q(s,a)}{\tau}\right)}, \tag{10}$$

where the sum in the denominator extends over all actions a available in state s. We will consider the choice of the temperature parameter τ in Section 6.1. Overall, we found that the Boltzmann method can converge much quicker towards a good-quality action and is better suited for time-varying channels.

To implement this scheme, the GACK packet needs to be extended. In particular, for each failed source, we need to indicate the number of allocated retransmission slots (as in the heuristic(PAR) method), the chosen relayer r and the number m of slots allocated to the relayer. When the number of relayers and the parameter Δ are kept small, a few bits will suffice to encode the chosen action. Furthermore, the protocol needs to be extended by methods allowing relays to register themselves with the controller.

4.2. Genie-Aided Scheme

In the learning(PAR) scheme, we have deliberately kept the action space small (e.g., by using only one relayer instead of two or more for the same packet, and by limiting the number of slots the relayer can get), which potentially reduces its performance. Furthermore, the controller does not have any information about the channel quality between the source node and the relayer.

To assess the impact of limiting the number of slots a relayer can get, we consider an idealized relaying scheme, called the **genie(PAR)** scheme. Again, we discuss the behaviour for an individual failed node. The controller first runs the heuristic(PAR) scheme to obtain an initial allocation based on the current PER estimates, returning a number n_i of retransmission slots allocated to failed node i. Then, the controller uses divine insight to obtain the current true PERs of **all** the channels in the system and then calculates for each relayer r and each possible split of the n_i retransmission slots between the source node i and the relayer r (such that the first slot is always allocated to i and the relayer gets contiguous slots at the end) the success probability as

$$(1 - p_{i,r}^{n_i - n_r}) \cdot (1 - p_{r,c}^{n_r}),$$

where n_r is the number of slots allocated to the relayer, $p_{i,r}$ is the true channel PER between source i and relayer r, and $p_{r,c}$ is the true channel PER between relayer r and the controller. The best such allocation is then selected.

4.3. Overheads

The retransmission schemes presented in this paper are aimed at resource-constrained platforms, and so it is prudent to briefly consider their computational, memory and energy overheads.

Let us first consider the case of simple sensor nodes. As compared to the standard scheme, they do not have to carry out any additional computations and only require very little additional memory to store the slot indices of the retransmission slots allocated to them. In terms of energy, for the heuristic(PAR) and learning(PAR) schemes the sensor nodes will have to process slightly larger GACK frames (as these now contain additional bits describing the retransmission slot allocation) and will possibly have to carry out more than one retransmission. Sensor nodes can sleep outside beacon frames, GACK frames and their own retransmission frames. Relay nodes, however, would need to be awake during all initial sensor transmissions and need to engage in retransmissions, if called upon.

The case of the coordinator is slightly more complex, but it is an often-made assumption that centralized coordinator nodes are less resource-constrained than simple sensor nodes. The computational overhead of the heuristic(PAR) scheme involves some modest amount of floating-point calculations: after each initial sensor transmission in a superframe, the coordinator will have to update the PER estimate for this sensor according to Equation (1)—note that this also requires some memory to store the PER estimates, one floating-point value per sensor. Furthermore, before sending the GACK packet, it has to calculate the retransmission slot allocation, which involves root-finding of a strictly monotonic function (Equation (7), the number of terms corresponds to the number of sensors), running through the algorithm filling the allocation gap (see Section 3.3) and the actual construction of the GACK packet. The computational overhead of the learning-based scheme consists of updating the Q-values (Equation (9)) and generating random actions following the Boltzmann distribution (Equation (10)), which both happen once per superframe and for each failed node. Furthermore, the coordinator will require for each sensor node a table space of $1 + R\sum_{s=2}^{N} \min\{s-1, \Delta\} \leq 1 + R(N-1)\Delta$ floating point values to store the Q-values.

5. Simulation Framework

All performance results in this paper have been generated with a custom simulation tool written in the Haskell programming language. To explain the rationale for this, recall that the main focus of this paper is to understand the performance of the retransmission slot scheduling schemes in isolation, and so we have deliberately chosen to exclude several other system aspects that could possibly confound the performance results, such as channel coding and modulation, hardware aspects, complex channel models using fading at sub-superframe timescales, etc. We wanted to keep our system model as simple as possible (but not simpler), and correspondingly have opted to create a comparatively small custom simulator and avoid the complexities of simulation frameworks like ns-3 or OMNet++.

When using static channels, on an individual channel, all packets are erroneous independently of each other and with the same probability (the packet error rate). For a given set of simulation parameters, we draw for each of the $(K+R+1)^2$ channels its packet error rate (PER) from a uniform distribution and run the system for 40,000 superframes or rounds. After finishing these 40,000 superframes, we compute the main performance measures considered in this paper: the most important measure is the **success probability**, defined as the fraction of superframes where **all** source packets have been received by the coordinator (a number between 0 and 1). Occasionally, we also consider the fraction of source packets received by the coordinator (again between 0 and 1). This procedure is repeated 100,000 times for the same parameter settings, so we run 100,000 replications with different random instantiations of the channel error probabilities. The output statistics of these replications are averaged, and we report these averages. Due to the large number of 100,000 replications the confidence interval half-widths for the success probability at a 99% confidence level are all below 0.3%, and we do not show the confidence intervals in the figures.

For time-varying channels, we use a modified version of this procedure. Each channel in the system is modeled as a two-state (time-homogeneous) Markov chain [21] with states s_1 and s_2. The state can only change at the start of a superframe and remains constant until the next superframe. In state s_i, the channel behaves like a static channel with packet error rate e_i. The packet error rates e_1 and e_2 are chosen randomly from a uniform distribution at the start of a replication, and, in each replication, 40,000 superframes are again simulated. The channel state transition probability matrix is the same for all channels and has the form

$$\mathbf{P} = \begin{pmatrix} p & 1-p \\ 1-p & p \end{pmatrix} \tag{11}$$

with parameter p. It is well-known that the state holding times have a geometric distribution [21] and the average state holding time in either state is $\frac{1}{1-p}$. In this paper, we consider four different values for p, giving channels with different rates of change: by picking $p \in \{0.9, 0.99, 0.999, 0.999999\}$, the state changes on average every 10th, 100th, 1,000th, or 1,000,000th superframe, respectively. Clearly, for

larger values of p, the channels become more stable, and we loosely refer to p as the channel stability. As before, a single replication extends over 40,000 superframes and we have used 100,000 replications in total (except for Section 6.1, where due to the large number of parameter combinations, we have restricted ourselves to 10,000 replications).

6. Results

6.1. Choosing the System Temperature for the Learning(PAR) Scheme

In our first study, we assess the impact of the temperature parameter τ governing the Boltzmann distribution for selecting actions (see Section 4.1) on the achievable performance of the learning(PAR) scheme, using the time-varying channel model (see Section 5). The main performance parameter considered here is the success probability.

We have chosen a scenario with $K = 6$ sources and $N = 9$ retransmission slots. The learning(PAR) scheme runs with a maximum of $\Delta = 1$ retransmission slots that can be allocated to a relayer (see Section 4.1). We have varied the system temperature τ in the set $\tau \in \{0.05, 0.075, 0.1, \ldots, 0.225, 0.25, 0.3, 0.35, 0.4, 0.45, 0.5, 0.6\}$ and the number of relayers as $R \in \{1, 2, 3, 4, 5\}$. The success probability results for the four considered values of the channel stability p (see Section 5) are shown in Figure 2a–d, respectively. It can be seen that the choice of the temperature value has a quite significant impact on the success probability, and the variation over different temperatures becomes larger as the number of relayers increases. Somewhat to our surprise, for all considered channel stability values p and relayer numbers R, the temperature $\tau = 0.1$ has displayed the optimal or close-to-optimal performance, so, in the remaining part of the paper, we will only consider the learning(PAR) scheme with a temperature value of $\tau = 0.1$.

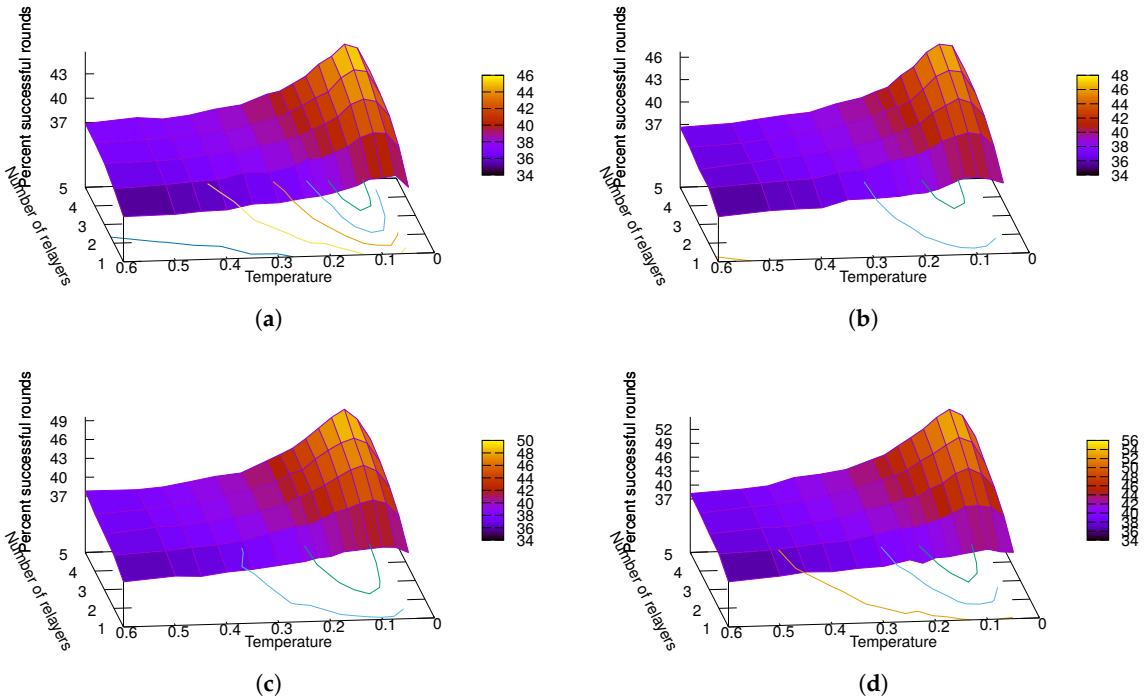

Figure 2. Average fraction of successful rounds for learning(PAR)-τ schemes for varying τ and varying numbers of relayers over two-state channels with different probabilities p to stay in the same state. (**a**) $p = 0.9$; (**b**) $p = 0.99$; (**c**) $p = 0.999$; (**d**) $p = 0.999999$.

6.2. Performance on Static Channels

In this section, we present simulation results for the case where all channels in the system follow the static channel model described in Section 5.

We have generated results for all the non-relaying schemes and the learning(PAR) scheme with a temperature parameter of $\tau = 0.1$ and, by setting $\Delta = 1$, restricting the relayer to a maximum of one slot. We have considered three different deployments:

- $K = 8$ source nodes, $N = 12$ retransmission slots,
- $K = 6$ source nodes, $N = 9$ retransmission slots,
- $K = 4$ source nodes, $N = 6$ retransmission slots.

In all of these deployments, the ratio of retransmission slots to source nodes is the same, giving each source 1.5 retransmission slots on average. For the learning(PAR) scheme, we have varied the number of relayers between 1 and 5.

In Figure 3a, we show the results for the success probability, and the fraction of received packets is displayed in Figure 3b The following points are interesting:

- The results for the success probability show a much wider spread (both when varying the number of sources and among the different schemes) than the fraction of successful packets. For the non-relaying schemes (except the standard scheme), the difference in the average fraction of successful packets is small, and the increase of that fraction for increasing numbers of relayers is moderate. Similar findings apply for all of the other scenarios studied in this paper, and we will not report further results on the fraction of successful packets.
- The standard scheme shows consistently and by some margin the poorest success probability performance. By comparing the standard scheme with the enhanced standard scheme, we can conclude that not utilizing all available retransmission slots significantly reduces the success probability.
- The success probability achieved by the heuristic(PAR) and optimal(PAR) schemes is very close, confirming that the heuristic proposed in Section 3.3 gives a very good approximation to the true optimum.
- Somewhat to our surprise, the heuristic(PAR) scheme shows almost the same success probability performance as the enhanced standard scheme—for six and eight sources, the advantage of heuristic(PAR) over the enhanced standard scheme is only on the order of 1% to 1.5% in absolute percentages.
- The biggest improvements can be achieved with the learning(PAR) scheme, in particular as more relayers are added to the system. For $K = 8$ sources and $R = 5$ relayers, the learning(PAR) scheme achieves almost twice the success probability of the heuristic(PAR) scheme; for smaller numbers of source nodes, the relative advantage is smaller but still significant. These results are even more encouraging when noting that the channels are completely random—with a carefully planned deployment of relayers further performance, improvements can be expected.

In summary, when only considering the non-relaying schemes, the enhanced standard scheme achieves almost the same performance as the heuristic(PAR) and optimal(PAR) schemes, while not requiring any changes to the LLDN packet formats. However, adding relayers to the system can achieve much more substantial gains.

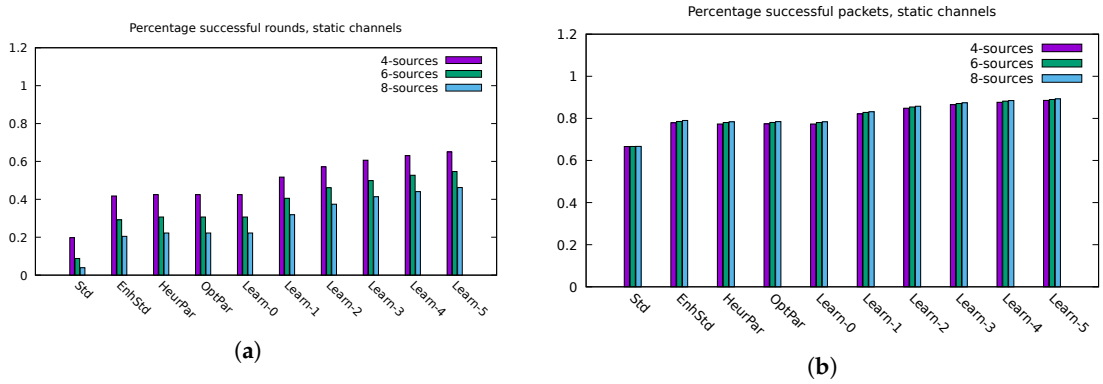

Figure 3. Average fractions of successful rounds and successful packets for all non-relaying schemes and learning (PAR). (**a**) success probability/fraction of successful rounds; (**b**) fraction of successful packets.

6.3. Performance on Time-Varying Channels

We next explore the success probability performance over time-varying channels. We have chosen a scenario with $K = 6$ sources and $N = 9$ retransmission slots, comparable with one of the scenarios discussed in Section 6.2 for static channels.

In Figure 4, we compare the success probability for all non-relaying schemes and the learning(PAR) scheme with a system temperature $\tau = 0.1$ and a maximum of $\Delta = 1$ retransmission slots allocated to a relayer. We present results for both static channels and the time-varying channels with the four different values for p, the channel stability (compare Section 5). A number of interesting observations can be made:

- The standard and enhanced standard schemes show more or less no sensitivity to the channel stability. The other two non-relaying schemes (heuristic(PAR), optimal(PAR)) show light performance improvements as the channel stability increases. We attribute this to the time required for the EWMA-based PER estimator (Equation (1)) after a channel change to adapt to the new channel PER. During this transient adaptation phase, sub-optimal allocation decisions can be made.
- When compared to static channels, the learning(PAR) scheme shows a reduced success probability performance over time-varying channels, particularly for smaller channel stability values. When the channel stability value becomes larger, the success probability of the learning(PAR) scheme approaches that for static channels, since, for larger channel stability values, the channels remain stable longer, and the fraction of time spent by the learning(PAR) scheme to learn the new channels becomes relatively smaller.
- Despite the performance loss observed over time-varying channels, it is still true for the learning(PAR) scheme that adding relayers gives significant success probability gains over the non-relaying schemes.

In summary, the learning(PAR) scheme can adapt to changing channels while successfully exploiting the presence of relayers. Higher channel stability leads to better performance for the learning schemes.

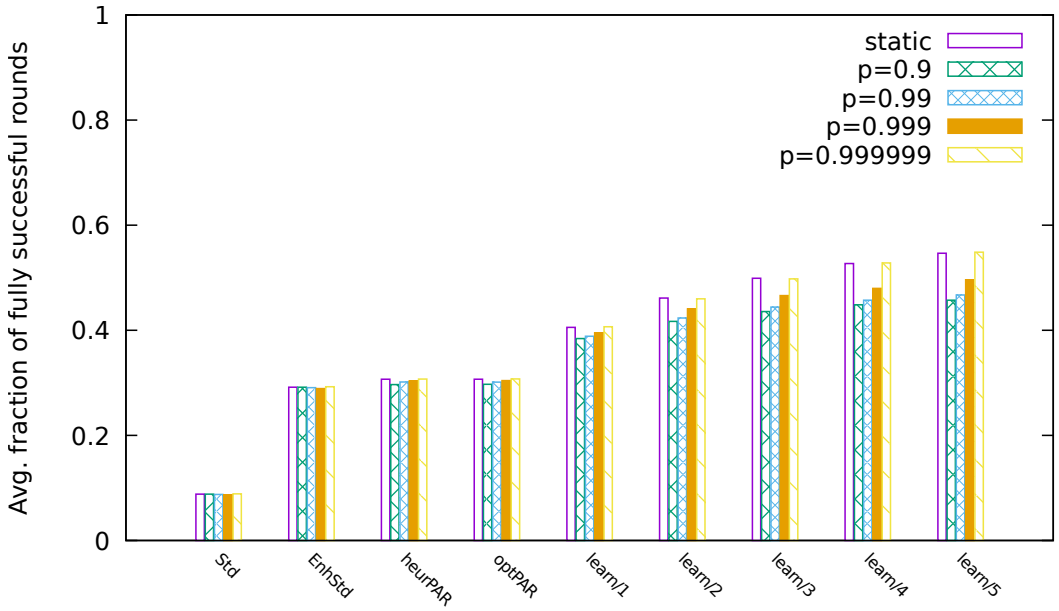

Figure 4. Average fraction of successful rounds.

6.4. Enlarging the Action Space

In the final experiment, we wanted to get some deeper insight into the performance characteristics of the learning(PAR) scheme. We have considered the learning(PAR) scheme for different numbers Δ of retransmission slots that can be allocated to a relay ($\Delta \in \{1, 2, 3, 4\}$—we denote the resulting scheme as learningPAR(Δ)) and compared it against the genie(PAR) scheme, both over the static channel model and the time-varying channel model. We have used a deployment with $K = 6$ sources, $N = 9$ retransmission slots and $R = 3$ relay nodes. Note that increasing Δ for the learning(PAR) scheme enlarges the space of available actions in a given state (compare Section 4.1), and it will on average require a longer time for the average rewards of all actions to settle close to their new values. The results are shown in Figure 5. The following points are interesting:

- Extending the action space for the learning(PAR) scheme has diminishing returns beyond $\Delta = 2$ for all considered channel models. The improvement from $\Delta = 1$ to $\Delta = 2$ is visible (most for the case of static channels), but, beyond this, it becomes marginal.
- In the case of static channels (and the time-varying channel with the largest channel stability), there is still a noticeable performance gap between the best learningPAR(Δ) scheme and the genie(PAR) scheme. We suspect that this gap is the price paid for the process of exploration, i.e., for the Boltzmann-based action selection scheme not selecting the best available action (which will have been learned after some time) throughout, but only with higher probability than other actions. Another possible explanation could have been the limited size of the action space when compared to the genie(PAR) scheme, but our finding of diminishing returns for increasing Δ does not support this hypothesis.
- The performance gap between the best learningPAR(Δ) scheme and the genie(PAR) scheme is even larger for time-varying channels with lower channel stability. The additional performance losses compared to static channels can be attributed to the transient times where the learningPAR(Δ) scheme needs to adjust to changed channels.

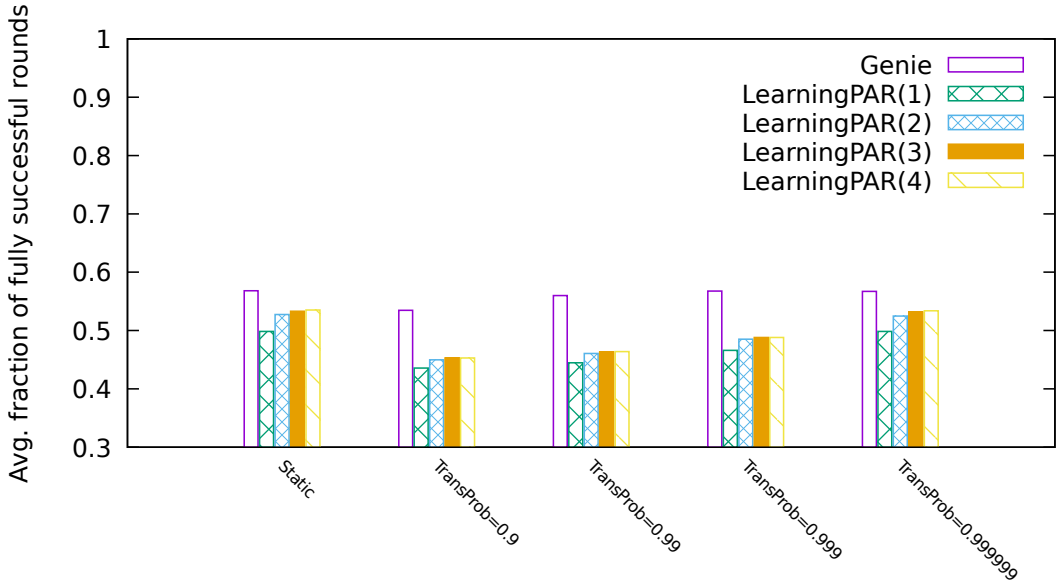

Figure 5. Average fraction of successful rounds.

7. Related Work

In the last few years, relay-assisted retransmission protocols (also known as cooperative ARQ protocols [22]) have received some interest in the area of industrial wireless (sensor) networks. The problem of optimal scheduling (with respect to the average number of packets successfully received before their deadline for a number of periodic sources with different periods) of relayer-based retransmissions has been explored in [9] for a TDMA-based system running over static (and known!) channels, and both (computationally heavy) optimal solutions and approximation algorithms have been investigated. The authors of [23] have considered relay-assisted communications in mining vehicle safety applications. Particularly, transmission scheduling for a multi-user MIMO system, including a set of relays, has been discussed. The application of relaying approaches in the context of IEEE 802.15.4 has been considered in [24], with a focus on relay selection. Three different selection schemes have been introduced, one in which relays are chosen periodically, one in which relay selection is triggered when the loss rate exceeds a threshold, and one in which each failed packet triggers a new relay selection. These schemes have been assessed experimentally for their effectiveness and their coordination overhead. All the proposed schemes require new control frames and involve explicit signaling. Note that the learning(PAR) scheme proposed in this paper differs from the schemes proposed in [24] by constantly maintaining quality information about all relayers, so that, after a degradation of the relayer that was best so far, the next best relayer is known quickly. The authors of [25] discuss the issue of cooperative relayer selection in an IEEE 802.15.4e variant. In their approach, the relayer for a given source node is selected only once and then not changed afterwards.

The LLDN extension to IEEE 802.15.4 has also been the focus of other works. The authors of [26] propose a method incorporating a time-diversity scheme to improve reliability. In this scheme, a packet is repeated a number of times in a superframe, and the impact on the (application-level) control performance of a networked control system is investigated. In [27], the performance of LLDN is analyzed in some detail; in particular, overheads and latencies are carefully explored.

The IEEE 802.15.4e amendment contains further extensions beyond LLDN. One of them, the time-slotted channel-hopping (TSCH) extension where the system periodically changes its frequency channel, has been considered in [28]. The link reliability of IEEE 802.15.4 in an industrial environment has been assessed experimentally in [29].

In summary,, in this paper, we have introduced and investigated a novel retransmission- and relay scheduling scheme based on Q-learning, which constantly keeps track of channel quality information and can identify new relayers quickly after channel changes. We have also carried out a detailed performance investigation.

8. Conclusions

The results in this paper suggest that the retransmission scheduling method described in the LLDN extension is sub-optimal and can be improved substantially. A substantial improvement can already be made when simply allowing the system to make use of all available retransmission slots and giving more than one slot to a failed source —as discussed, the enhanced standard scheme does not even require any changes to packet formats and is almost as good (in the considered scenarios) as more optimized non-relaying schemes. Significant further gains can be achieved by adding relayers, at the cost of changing the GACK packet format, some additional signaling to announce the presence of relay nodes to the controller, and the introduction of some additional configuration values (α, α_r, Δ and the temperature parameter τ). However, our learning-based scheme does **not** require any a priori channel state information and can adapt to changing channels. The price paid for the ability to adapt can be quantified by the performance difference between the learningPAR(Δ) scheme and the genie(PAR) scheme.

There are several avenues for future work. One is the more systematic offline (or perhaps even online) optimization of the system parameters (α, α_r, Δ, τ). For example, the two parameters α and α_r determining the "learning speed" for the channel PER estimator and the rewards can be adapted according to the observed rate of change in the underlying wireless channels. When this rate of change becomes small, one may also consider "cooling down" the system temperature (i.e., making τ smaller) to more strongly prefer the best available action. Furthermore, it would be very interesting to implement the learning(PAR) scheme in an experimental testbed to obtain better insight into implementation issues and to perform performance measurements. Finally, to eliminate the need for dedicated relay nodes, one could also extend the learning(PAR) scheme to use source nodes as relayers, energy consumption concerns permitting.

Acknowledgments: We would like to thank the anonymous reviewers and the editor for their helpful and constructive comments.

Author Contributions: Andreas Willig contributed to algorithm and conceptual design, conducted the simulations, evaluated the results and wrote the article. Yakir Matusovsky and Adriel Kind contributed to algorithm and conceptual design.

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

References

1. Galloway, B.; Hancke, G.P. Introduction to Industrial Control Networks. *IEEE Commun. Surv. Tutor.* **2013**, *15*, 860–880.
2. Gungor, V.C.; Hancke, G.P. Industrial Wireless Sensor Networks: Challenges, Design Principles, and Technical Approaches. *IEEE Trans. Ind. Electron.* **2009**, *56*, 4258–4265.
3. Willig, A. Recent and Emerging Topics in Wireless Industrial Communications: A Selection. *IEEE Trans. Ind. Inform.* **2008**, *4*, 102–124.
4. LAN/MAN Standards Committee of the IEEE Computer Society. *IEEE Standard for Local and Metropolitan Area Networks—Part 15.4: Low Rate Wireless Personal Area Networks (LR-WPANs)*; Revision of 2011; Institute of Electrical and Electronics Engineers: Piscataway, NJ, USA, 2011.
5. Toscano, E.; Bello, L.L. Multichannel Superframe Scheduling for IEEE 802.15.4 Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2012**, *8*, 337–350.
6. IEEE Computer Society. *IEEE Standard for Local and Metropolitan Area Networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs)— Amendment 1: MAC Sublayer*; IEEE Std 802.15.4e-2012; Institute of Electrical and Electronics Engineers: Piscataway, NJ, USA, 2012.

7. Chen, F.; German, R.; Dressler, F. Towards IEEE 802.15.4e: A Study of Performance Aspects. In Proceedings of the 8th IEEE International Conference on Pervasive Computing and Communications (PERCOM Workshops), Mannheim, Germany, 29 March–2 April 2010; pp. 68–73.
8. Diggavi, S.N.; Al-Dhahir, N.; Stamoulis, A.; Calderbank, A.R. Great Expectations: The Value of Spatial Diversity in Wireless Networks. *IEEE Proc.* **2004**, *92*, 219–270.
9. Willig, A.; Uhlemann, E. Deadline-Aware Scheduling of Cooperative Relayers in TDMA-Based Wireless Industrial Networks. *Wirel. Netw.* **2014**, *20*, 73–88.
10. Sutton, R.S.; Barto, A.G. *Reinforcement Learning—An Introduction*; MIT Press: Cambridge, MA, USA, 1998.
11. Bertsekas, D.P.; Tsitsiklis, J.N. *Neuro—Dynamic Programming*; Athena Scientific: Belmont, MA, USA, 1996.
12. Willig, A.; Matusovsky, Y.; Kind, A. Retransmission Scheduling in 802.15.4e LLDN—A Reinforcement Learning Approach with Relayers. In Proceedings of the International Telecommunication Networks and Applications Conference (ITNAC) 2016, Dunedin, New Zealand, 7–9 December 2016.
13. Girs, S.; Willig, A.; Uhlemann, E.; Bjoerkman, M. Scheduling for Source Relaying with Packet Aggregation in Industrial Wireless Networks. *IEEE Trans. Ind. Inform.* **2016**, *12*, 1855–1864.
14. Feller, W. *An Introduction to Probability Theory and Its Applications—Volume I*, 3rd ed.; John Wiley: New York, NY, USA, 1968.
15. Korte, B.; Vygen, J. *Combinatorial Optimization—Theory and Algorithms*, 3rd ed.; Springer: Berlin, Germany, 2005.
16. Bertsekas, D.P. *Nonlinear Programming*, 2nd ed.; Athena Scientific: Belmont, MA, USA, 1999.
17. Rappaport, T.S. *Wireless Communications—Principles and Practice*; Prentice Hall: Upper Saddle River, NJ, USA, 2002.
18. Liu, K.J.R.; Sadek, A.K.; Su, W.; Kwasinski, A. *Cooperative Communications and Networking*; Cambridge University Press: Cambridge, UK, 2009.
19. Kramer, G.; Maric, I.; Yates, R.D. Cooperative Communications. *Found. Trends Netw.* **2006**, *1*, 271–425.
20. Laneman, J.N.; Tse, D.N.C.; Wornell, G.W. Cooperative Diversity in Wireless Networks: Efficient Protocols and Outage Behaviour. *IEEE Trans. Inf. Theory* **2004**, *50*, 3062–3080.
21. Norris, J.R. *Markov Chains*; Cambridge University Press: Cambridge, UK, 1997.
22. Marchenko, N.; Bettstetter, C. Cooperative ARQ with Relay Selection: An Analytical Framework Using Semi-Markov Processes. *IEEE Trans. Veh. Technol.* **2014**, *63*, 178–190.
23. Ni, W.; Collings, I.B.; Liu, R.P.; Chen, Z. Relay-Assisted Wireless Communication Systems in Mining Vehicle Safety Applications. *IEEE Trans. Ind. Inform.* **2014**, *10*, 615–627.
24. Marchenko, N.; Andre, T.; Brandner, G.; Masood, W.; Bettstetter, C. An Experimental Study of Selective Cooperative Relaying in Industrial Wireless Sensor Networks. *IEEE Trans. Ind. Inform.* **2014**, *10*, 1806–1816.
25. Momoda, M.; Hara, S. Use of IEEE 802.15.4 for a Cooperator-Assisted Wireless Body Area Network. In Proceedings of the 8th International Symposium on Medical Information and Communication Technology (ISMICT), Firenze, Italy, 2–4 April 2014; pp. 1–5.
26. Yen, B.X.; Hop, D.T.; Yoo, M. Redundant Transmission in wireless networked control system over IEEE 802.15.4e. In Proceedings of the International Conference on Information Networking (ICOIN), Bangkok, Thailand, 28–30 January 2013; pp. 628–631.
27. Dariz, L.; Malaguti, G.; Ruggeri, M. Performance Analysis of IEEE 802.15.4 real-time Enhancement. In Proceedings of the IEEE 23rd International Symposium on Industrial Electronics (ISIE), Harbiye Istanbul, Turkey, 1–4 June 2014; pp. 1475–1480.
28. Palattella, M.R.; Accettura, N.; Grieco, L.A.; Boggia, G.; Dohler, M.; Engel, T. On Optimal Scheduling in Duty-Cycled Industrial IoT Applications Using IEEE802.15.4e TSCH. *IEEE Sens. J.* **2013**, *13*, 3655–3666.
29. Yadong, W.; Shihong, D. Study on IEEE802.15.4 Link Reliability in Industrial Environments. In Proceedings of the IEEE International Conference on Wireless for Space and Extreme Environments (WiSEE), Baltimore, MD, USA, 7–9 November 2013; pp. 1–7.