On Reliable Communication in Transmit-Only Networks for Home Automation

Philip Parsch*, Alejandro Masrur

Department of Computer Science, TU Chemnitz, Germany

Abstract

Home-automation applications such as intelligent illumination, heating, and ventilation allow reducing the overall energy consumption and improve comfort in our everyday lives. To implement such applications, multiple sensors and actuators often need to be connected into networks typically communicating over radio signals, i.e., wireless sensor networks (WSNs). Many available technologies are based on bidirectional devices with the capability of acknowledging packets and performing retransmissions if necessary. However, home-automation devices mostly report data to a sink for which they do not need any feedback channel or external control, thus, unidirectional devices can be used instead reducing costs and energy consumption. On the other hand, since unidirectional nodes are unable to perform carrier sensing or acknowledge packets, the resulting networks are often unreliable. To overcome this problem, we propose a medium access control (MAC) that consists in making each transmit-only node send a sequence of redundant packets. The proposed method guarantees reliability, i.e., at least one packet of each sequence reaches its destination within a specified deadline by carefully selecting inter-packet times. In contrast to similar approaches from the literature, our MAC is based on a more general model that allows describing arbitrary deadlines and packet sizes for each node in the network and can accommodate considerably more nodes into a reliable network as the ratio between the longest and the shortest deadline increases. We illustrate these and other benefits of the proposed MAC by means of extensive simulations based on the OMNeT++ framework.

Keywords: Home automation, reliability, cost efficiency, sensor/actuator networks, unidirectional/transmit-only nodes
2010 MSC: 68M12

1. Introduction

There is an increasing interest in home automation with the aim of improving comfort and energy efficiency in modern homes. As illustrated in Fig. 1, applications such as heating, ventilation and air-conditioning (HVAC), appliance management, etc. are typical of this domain. To this end, embedded devices need to be interconnected, which puts emphasis on wireless sensor networks (WSNs), since communication has to be flexible, reliable and cost-effective at the same time.

Home-automation WSNs are normally based on bidirectional nodes that are capable of transmitting and receiving data. However, the focus recently shifted towards unidirectional (i.e., transmit-only) protocols, as these allow reducing the energy consumption by avoiding the high overhead of bidirectional MAC protocols and do not require to monitor the communication channel [6][14][21]. Another important aspect of unidirectional networks is the reduced complexity as sensor nodes forgo the receiver circuitry and, hence, can use smaller batteries, less powerful processors, etc. [6]. This results in lower hardware costs, which is especially interesting for networks with high numbers of nodes. For example, a simple transmit-only light switch from HomeEasy [1] costs around 10 euros, whereas the price of its bidirectional equivalent from Z-Wave [3] is about 50 euros. Considering that home-automation networks typically contain 30 to 50 devices, using unidirectional nodes can result in considerable cost savings.

However, this comes at the cost of an increased unreliability, i.e., data is more likely to be lost; no carrier-sensing or synchronization is possible, hence, established solutions like CSMA (Carrier Sense Multiple Access) or TDMA (Time Division Multiple Access) are not applicable. To overcome this problem, most commercial available transmit-only systems send their data multiple times to increase the probability of a successful delivery. For example, HomeEasy [1] nodes send up to 12 redundant copies of their packets upon activation depending on the type of node. However, no special care has been taken on choosing inter-packet times and packets are rather sent subsequently with a fixed separation not always leading to good results.

To overcome this problem, MAC techniques based on sending a sequence of redundant data packets have been proposed in the literature [4][15]. Neglecting external interference — home automation nodes are well shielded by walls [15] — these methods guarantee that at least one packet of each sequence reaches its destination within a

*Corresponding author

Email address: philip.parsch@cs.tu-chemnitz.de (Philip Parsch)

 https://doi.org/10.1016/j.jnca.2017.10.023

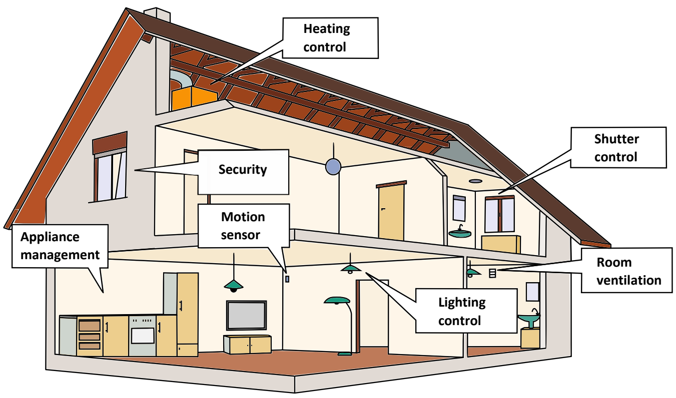

Figure 1: Example of a home-automation network and its typical application areas: heating, ventilation, air-conditioning (HVAC), appliance control and security [8]. More elaborate applications can further be realized. For example, lights can be automatically turned on at night when a user enters a room. Similarly, appliances, e.g., TV, radio, coffee machine, etc., can be switched off when a user leaves home, etc.

specified deadline in the worst case. To this end, suitable inter-packet times need to be found for each transmit-only node in the network.

However, these methods from the literature make restrictive assumptions to reduce the complexity of the problem and are, hence, not suitable for a wide range of applications. In particular, the deadlines, by which at least one packet of each sequence must reach its destination, and the packet sizes, i.e., the amount of data bytes to transmit, are enforced to be the same for all nodes [4][5][15][21]. Further, nodes are either assumed to be activated once within a sufficiently long time interval such that all sequences of packets finish before the next activation of any node, or to implement a transmission pause — equal to the longest deadline in the system — after each sequence of packets [4][5][15].

These assumptions do not reflect the actual requirements of home-automation applications, since networks typically contain different types of nodes with varying packet sizes and/or deadlines requirements [8]. For example, a light switch should turn on/off lights within 500 ms. Greater delays would negatively impact the quality of the system. In contrast, temperature sensors periodically transmit their data within a relatively long time interval in the order of minutes. In addition, temperature sensors usually require multiple data bytes, whereas light switches can encode their data within one byte.

The methods presented in [4] and [15] consequently incur in great pessimism when adapted for heterogeneous networks. On the one hand, this results in increased communication delay and, on the other hand, there will be more packet collisions as the number of nodes increases. As a result, less nodes can be accommodated in the network for a specified deadline — more details follow in Section 5.

1.1. Contributions

In this paper, we propose a reliable MAC for unidirectional (i.e., transmit-only) home-automation WSNs. Our technique consists in making each transmit-only node send a sequence of k_i redundant packets with constant inter-packet times p_i. Neglecting external interference and carefully selecting k_i and p_i, the proposed technique guarantees *full* reliability, i.e., at least one packet of each sequence reaches its destination in the worst case.

In contrast to existing approaches from the literature [5][15][21], the proposed MAC is based on a more general model that allows for modeling arbitrary deadlines and packet sizes. It also eliminates the need for transmission pauses after a sequence of packets, which reduces the communication delay and therefore increases the maximum possible network size.

In addition, we analyze the effect of practical factors such as external interference and clock drift on our MAC. In practice, whereas clock drift has almost no effect on reliability, it is not possible to guarantee *full* reliability in the presence of a high external interference. However, our MAC still shows a robust behavior.

We finally analyze the effect of sending less than k_i redundant packets with the aim of reducing the protocol's overhead while still guaranteeing a desired reliability. This allows designing heterogeneous networks with mixed reliability levels and enables nodes to dynamically adapt their energy consumption depending on the type of data to transmit.

1.2. Structure of the paper

The rest of this paper is structured as follows. Related work is discussed in Section 2. Next, Section 3 explains our system model and assumptions. Section 4 introduces the proposed MAC technique for home-automation WSNs with transmit-only nodes. Section 5 presents our simulation results, while Section 6 studies the effect of practical factors such as external interference and clock drift on the proposed MAC. Section 7 concludes the paper.

2. Related Work

There are many different approaches from the literature that are concerned in making WSN more reliable and energy-efficient. Most of them use bidirectional nodes that implement elaborate protocols such multi-hopping, automatic retransmission and routing of data packets. However, in scenarios, where simplicity and cost-efficiency are key factors, unidirectional communication has been used many times in the past: environmental monitoring [13], body area networks [11][17][20], Internet of Things [9][13] and RFID systems [9].

The simplicity of unidirectional nodes, however, comes at the cost of increased unreliability; no carrier-sensing or synchronization is possible, hence, generated traffic is completely uncoordinated [6]. As a consequence, established solutions like CSMA and TDMA cannot be used and special MAC protocols must be applied instead that do not rely on feedback from the sink node.

The two protocols presented in [6][16] use a hybrid approach, i.e., they are composed of a high number of transmit-only nodes forming clusters for cost reduction and so-called *cluster heads* with reception capability. Cluster heads collect packets from their corresponding transmit-only clusters and forward them to receivers. Since they can acknowledge packets and perform carrier sensing, more sophisticated communication schemes can be implemented upon them. For example in [16], cluster heads use a configurable receiver that only collects data packets complying with a pre-specified signal strength. By this, the strongest signal can be received at the event of a collision whereas otherwise it would be lost. However, if many cluster heads are used, costs and energy consumption increase rapidly. Moreover, these methods cannot provide any reliability guarantees and packets may potentially never reach their receivers due to collisions, in particular, within a transmit-only cluster.

Another hybrid approach presented in [21] also consists of two sensor node types: low-priority, transmit-only nodes (LP-nodes) and high-priority, bidirectional nodes (HP-nodes). Both node types are triggered periodically and transmit a number of redundant data packets with constant inter-packet times. These times are known by the sink, which uses them to schedule HP nodes to transmit in vacant time slots. As a result, HP-nodes do not collide with LP-nodes and the overall transmission reliability increases. However, this approach again incurs in increased costs and energy usage, since the resulting improvement in reliability strongly depends on the number of HP-nodes. Further, it assumes that nodes are triggered periodically in a known interval, which is not practicable in event-triggered applications. In contrast, in this paper we propose an approach that is applicable to both periodic and event-triggered scenarios.

Other approaches [9][10] use backscatter communication to reduce costs and energy consumption of nodes. For example in RFID systems [9], a sink node (reader) transmits a radio signal, which sensor nodes (tags) can use as a power source and as a carrier for reflecting back their encoded data. Similar to classic transmit-only systems, backscattering does not allow tags to detect transmissions from other tags [7]. However, in contrast, the reader's carrier signal allows synchronization or waking up tags with specific IDs. As a consequence, most existing backscattering systems either use variants of Aloha and TDMA or tree search methods that aim to avoid collisions by identifying only one tag at a time. These benefits, however, also come with some major drawbacks making backscattering impracticable for smart home networks. For example, many devices in smart homes have long idle times in the order of hours, but upon activation, data must be transmitted timely. Since backscattering is receiver initiated, the sink either has to pull for data periodically, which adds additional delay, or provide a continuous data signal, being especially problematic in (typical) settings with multiple sink nodes.

In the domain of pure unidirectional networks, Andersson et al. [4] presented a transmission scheme guaranteeing that data always reaches its destination within the shortest possible delay. To this end, each transmitter sends a sequence of redundant packets with carefully selected patterns such that at least one packet is not interfered by other transmitters. The transmission patterns are selected via ILP (integer linear programming) minimizing the transmission durations of all sequences of packets. However, due to the high complexity of the problem, patterns for only a small number of nodes can be found in an acceptable time.

To address this concern, Andersson et al. presented an alternative algorithm that heuristically searches for transmission patterns [5]. Although this second algorithm is considerably faster than their ILP-based approach, it is still time-consuming as shown later by our experimental evaluation. Both approaches in [4] and [5] assume that there is no interference from outside the network.

In our previous work [15], we proposed a technique to design reliable single-hop WSNs for home automation based on transmit-only nodes. Similar to [4][5], the technique in [15] provides a guarantee that data always reaches its destination within a specified deadline — without external interference. This consists in sending a sequence of identical packets with constant inter-packet separations that are carefully selected for each node. In contrast to [4][5], we analytically derived upper bounds on the inter-packet times of transmit-only nodes in [15]. The inter-packet times resulting from [15] are more pessimistic than those of [4][5], however, the algorithm in [15] is considerable less complex.

Though allowing the design of reliable WSNs based on transmit-only nodes, these related approaches [4][5][15] assume that data packets and deadlines are the same for all nodes. As mentioned before, this does not match most home-automation networks, which are typically composed of multiple different devices with varying Quality of Service (QoS) requirements. In contrast, this paper enables modeling different packet sizes and deadlines for more general settings and applications. This information is also exploited to optimize delays and be able to accommodated more nodes into a reliable WSN as the ratio between longest to shortest packet/deadline increases.

Finally, the related approaches [4][5][15] enforce a transmission pause after each sequence of packets. This pause needs to be equal to d_{max}, the maximum allowable delay for data transmission, or to the duration of the longest sequence of packets. In this paper, we further remove the need for transmission pauses as discussed later in more detail.

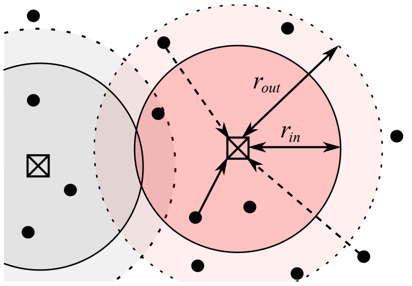

Figure 2: Example of a single-hop WSN with transmit-only nodes (solid circles) and sink nodes (checked boxes): r_{in} represents the range within which a sink collects packets, while r_{out} indicates the range in which transmitters can interfere with each other, e.g., at simultaneous transmissions.

3. System Model and Assumptions

We consider a single-hop WSN for home automation consisting of n transmit-only nodes and multiple receive-only sink nodes that are distributed within a house. For the rest of the paper, we refer by nodes to transmit-only and by sinks to receive-only nodes.

Nodes are typically connected to one or more sinks in a single-hop (star-topology) fashion. They can be activated either by a sporadic event or periodically depending on the application. To this end, sinks constantly monitor the communication channel to be able to receive the data directly and, hence, remove the need of sending wake-up messages, etc. before each transmission. Sinks in our case are lamps, heaters, different appliances, etc. and are normally attached to the house's electric network — having continuous power supply.

All nodes are spatially distributed within a radius r_{in} from a data sink as shown in Fig. 2. We assume that nodes are independent of each other and that there is no synchronization between them. Upon activation, they broadcast a sequence of k_i packets with $k_i \in \mathbb{N}_{>0}$. As a result, all transmitting nodes within a radius r_{out} from the sink being $r_{out} \geq r_{in}$ — see again Fig. 2 — can potentially interfere with each other. In addition, we consider that packets in a sequence have constant inter-packet times p_i with $p_i \in \mathbb{R}_{>0}$.

Depending on the application, different nodes may need to transmit different numbers of data bytes. For example, a climate sensor node usually transmits a relatively large data packet containing temperature, humidity values, etc., whereas a light switch will only contain an address byte and an on/off command. For this reason, in contrast to [4, 5] and [15], we allow for packets with different *lengths* l_i with $l_i \in \mathbb{R}_{>0}$. This l_i is the time required by node i to send a packet at a given transmission speed. Note that $l_i \leq p_i$ must hold for any i, i.e., each transmit-only node must be able to send its full packet within a time interval equal to its inter-packet separation.

For a given node, at least one packet has to reach the sink within a specified deadline that depends on the application. For example, in a home-automation setting, a light switch should turn on lights within half a second to one second from its activation, whereas a room temperature or humidity sensor may have a deadline in the order of minutes instead. To this end, we allow modeling nodes with different deadlines, which are then denoted by d_i where $d_i \in \mathbb{R}_{>0}$ and $p_i < d_i$ holds.

In the next section, we introduce the proposed MAC guaranteeing reliable communication as defined below. To this end, a technique to derive *safe* values for k_i and p_i is presented based on the assumption that interference from outside the network is negligible. Later, in Section 6 we extend our analysis to consider external interference among other practical factors.

4. Proposed MAC Technique

In this paper, we are concerned with reliable communication for transmit-only WSNs. For this purpose, let us first consider the following definition.

Definition: We define reliability of a WSNs as the probability that, in the worst case, at least one out of k_i packets of any node i reaches its destination within a specified deadline d_{max}.

In particular, the proposed MAC allows guaranteeing *full* reliability (when neglecting external interference), meaning that there is never data loss within the network. To this end, we have to carefully select packet numbers k_i and unique inter-packet times p_i for every node i. In the following lemmas, we state the necessary requirements and conditions for k_i for different cases.

Lemma 1. *Let us consider a set of n independent transmit-only nodes. Each node j, with $1 \leq j \leq n$, is activated once and sends a sequence of k_j packets with constant inter-packet times p_j within its deadline d_j. Further, we consider that p_i and p_j can be chosen such that there is at most one collision between any two sequences of any two nodes j and i. In the worst case, at most $\gamma_i - 1$ packets of node i, with $1 \leq i \leq n$ and $i \neq j$, will be lost due to starting sequences of every node j:*

$$\gamma_i = n + \sum_{j=1, j\neq i}^{n} \left\lfloor \frac{d_i}{d_j} \right\rfloor . \tag{1}$$

Proof. Since nodes are independent of each other, they start transmitting their sequences of packets at arbitrary points in time. Hence, it may happen that every time a node i transmits a packet, this gets interfered by a packet of a sequence of another node j — with $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$ — being sent at that time. Since there are $n-1$ nodes other than i in the system, $n-1$ packets of node i can be interrupted this way, provided that suitable p_i and p_j can be found such that there is at most one collision between any two sequences of nodes j and i.

In addition, since each of the other $n-1$ nodes is activated at most once within its deadline d_j, node i's transmissions can be interfered at most $\lfloor \frac{d_i}{d_j} \rfloor$ times more — in total $\lfloor \frac{d_i}{d_j} \rfloor + 1$ times — by the same node j. When considering all nodes in the system, we obtain the following expression:

$$\sum_{j=1, j\neq i}^{n} \left(\left\lfloor \frac{d_i}{d_j} \right\rfloor + 1 \right) = n - 1 + \sum_{j=1, j\neq i}^{n} \left\lfloor \frac{d_i}{d_j} \right\rfloor ,$$

which is $\gamma_i - 1$ as defined in (1). The lemma follows. □

From Lemma 1, we can conclude that a node i needs to transmit a minimum of γ_i packets in order that at least one of them reaches its destination within d_i in the worst case, i.e., this is a *safe* value for k_i. However, since (1) is based on the fact that nodes can be activated at arbitrary points in time, it may result in very pessimistic values. Note that Lemma 1 considers starting sequences of packets by a node j. As discussed later, collisions with subsequent packets in a node j's sequence can be prevented by selecting suitable values of p_i and p_j.

Let us consider the following example consisting of two nodes with $d_i = 1\,\text{min}$ and $d_j = 500\,\text{ms}$ respectively. From (1), we obtain $\gamma_i = 121$, i.e., in spite of having a two-node network, node i needs to sends at least 121 packets to achieve reliability in the worst case. If we now have another node with a 500 ms deadline, γ_i will further increase by 120, i.e., node i will have to send 241 packets.

To countervail this pessimism, it is possible to impose a transmission pause of at least d_{max} after each sequence of packets, where d_{max} denotes the maximum deadline in the network. This is formalized in the next lemma.

Lemma 2. *Let us consider a set of n independent transmit-only nodes. Each node j sends a sequence of k_j packets with constant inter-packet times p_j within its deadline d_j, after which it implements a transmission pause of at least d_{max}. Further, we consider that p_i and p_j can be chosen such that there is at most one collision between any two sequences of any two nodes j and i. In the worst case, at most $n-1$ packets of node i will be lost due to starting sequences of packets of every node j, where $d_{max} = \max_{j=1}^{n}\{d_j\}$, $1 \le i \le n$, $1 \le j \le n$, and $i \ne j$ hold.*

Proof. Since nodes are independent of each other, they start transmitting their sequences of packets at arbitrary points in time. As already mentioned, every time a node i transmits a packet, this can be interfered by a packet of a sequence of another node j — with $1 \le i \le n$, $1 \le j \le n$, and $i \ne j$ — being sent at that time. Since there are $n-1$ nodes other than i in the system, $n-1$ packets of node i can be interrupted this way, provided that suitable p_i and p_j can be found such that there is at most one collision between any two sequences of nodes j and i.

Now, since each of the other $n-1$ nodes can be activated not earlier than d_{max} time after having finished one sequence of packets, node i's transmissions cannot be interfered anew by a starting sequence of packets of any other node j, i.e., $\lfloor \frac{d_i}{d_{max}+l_j} \rfloor = 0$. As a result, if transmission pauses greater than or equal to d_{max} are enforced after each sequence of packets, in the worst case, at most $n-1$ packets can be lost by any node i. The lemma follows. □

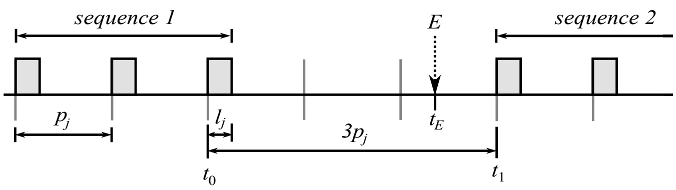

Figure 3: Delayed-activation scheme: An event E is not allowed to immediately trigger a new sequence of packets of a node j. Instead, this is delayed to the next time instant that is a multiple of p_j starting from the last packet of the previously sent node j's sequence.

Unfortunately, in contrast to the approaches from the literature, imposing a transmission pause of at least d_{max} after each sequence of packets is not a suitable solution due to considering arbitrary deadlines. In the above example, this would lead to node j being *blocked* by 1 min after each activation. If node j is a light switch and node i a temperature sensor in a home-automation setting, this means that lights would be blocked in an on- or off-state for 1 min, which is clearly an unacceptable delay.

4.1. Delayed-activation scheme

To reduce pessimism by (1) without enforcing long transmission pauses, we introduce the concept of delayed activation as illustrated in Fig. 3. In principle, after every sequence of packets of any node j, an event E is not allowed to trigger a new sequence of packets immediately. This is rather delayed to the closest time instant that is a multiple of p_j starting from the last packet of the previous sequence.

That is, if the last packet of node j's *sequence 1* is sent at time t_0 and an event E occurs at a later time t_E, then the next sequence of packets (triggered by E) starts at a t_1 being $t_E \le t_1$ where $t_1 - t_0$ is an integer multiple of p_j — in Fig. 3 this is $3p_j$.

Lemma 3. *Let us consider a set of n independent transmit-only nodes. Each node j sends a sequence of k_j packets with constant inter-packet times p_j within its deadline d_j, after which it implements a delayed activation as described above for a time interval of length d_{max}. In the worst case, at most $n-1$ packets of a node i will be lost due to starting sequences of packets of every node j provided that suitable p_i and p_j can be found, where $d_{max} = \max_{j=1}^{n}\{d_j\}$, $1 \le i \le n$, $1 \le j \le n$, and $i \ne j$ hold.*

Proof. Let us consider again the example of Fig. 3. In the case $t_E - t_0 - l_j > d_{max}$, if node j's *sequence 1* interfered with a packet of a sequence of node i, node j's *sequence 2* triggered by E cannot interfere with any other packet of the same node i's sequence as per Lemma 2.

In the case $t_E - t_0 - l_j < d_{max}$, if node j's *sequence 1* interfered with a packet of a sequence of node i, by properly selecting p_j and p_i, it can be avoided that *sequence 2* and its following node j's sequences in $[t_0 + l_j, t_0 + l_j + d_{max}]$ interfere with any other packet of same node i's sequence. Note that if proper p_i and p_j can be found for any i and j with $i \neq j$, in the worst case, at most $n-1$ packets of any node i can be lost due to starting sequences of every other node j. The lemma follows. □

As per Lemma 3, the delayed-activation scheme allows us to reduce the pessimism introduced by Lemma 1 in the same way Lemma 2 does, but without enforcing long transmission pauses. For this, suitable values of p_i and p_j must be configured for every i and j where $i \neq j$ as discussed in the next section.

To further clarify this, let us again consider an exemplary system composed of 9 light switches with $d_{max} = 500\,\text{ms}$ and one temperature sensor with $d_{max} = 1\,\text{min}$. If we implement a transmission pause after each sequence, like in [4][15], each light switch will have a pause time of $1\,min$ after each activation, which is clearly unacceptable. In case of the delayed activation scheme, this pause time reduces to roughly $\frac{500\,\text{ms}}{10} = 50\,\text{ms}$ for the light switch (and 6 s for the temperature sensor), discussed later. Now, if the user accidentally switches a light on and wants to switch it immediately back off, he must wait at most 500 ms (instead of 1 min + 500 ms), since a node can be activated at most once within its deadline. As a result, the delay incurred by our delay-activation scheme does not play any role and can typically be neglected.

4.2. Selecting inter-packet times

So far, we have considered collisions caused by the first packet sent by other nodes in the network. However, any of their subsequent packets may also produce collisions leading to further packet loss. In other words, sending sequences of n packets in the delayed-activation scheme, i.e., making $k_i = n$, is necessary but not sufficient to guarantee a reliable communication.

To guarantee full reliability, i.e., that at least one packet of a node reaches its receiver in the worst case, we also need to select *safe* values of p_i for each node in the system. Towards this, note that a packet from one node can be interfered by a packet of another node sending simultaneously, and that the minimum overlapping between any two packets leads to packet loss. The following theorem states a necessary and sufficient condition for guaranteeing that, among the n packets sent by a node i, at least one of them reaches its destination.

Theorem 1. *Let us consider a set of n independent transmit-only nodes, each of which is activated once and sends a sequence of $k_i = n$ packets with constant inter-packet times p_i within its deadline d_i. In the worst case, at least one packet of each node can be guaranteed to reach its destination, if the delayed-activation scheme is used and the following condition holds for $1 \leq i \leq n$, $1 \leq j \leq n$, $i \neq j$, and $1 \leq k_i \leq n-1$:*

$$\mathrm{mod}\left(\frac{k_i \times p_i}{p_j}\right) \geq l_i + l_j, \tag{2}$$

where $\mathrm{mod}(\cdot)$ is the modulo operation, l_i and l_j are the packet lengths of node i and j respectively, while p_i and p_j are their corresponding inter-packet times.

Algorithm 1 Searching for values of p_i

Require: n, list of (l_i, d_i)
sort list of (l_i, d_i) according to non-decreasing d_i
for $i = 1$ to n **do**
  found = false
  $p_{temp} = \frac{d_i - l_i}{n}$
  while $p_{temp} > 0$ **do**
    if check_period (i, p_{temp}) == true **then**
      found = true
      $p_i = p_{temp}$
      break
    else
      $p_{temp} = p_{temp} - step$
    end if
  end while
  if found == false **then**
    return (failed)
  end if
end for
return (list of p_i)

The proof of Theorem 1 can be found in Appendix A. This allows us to guarantee that at least one packet of each node reaches its destination within d_i, provided that each node follows our delayed-activation scheme and sends n packets within d_i. It does not help in selecting suitable values of p_i, but only verifies provided values of p_i.

In the following, we are concerned with finding *safe* values for p_i for any i and $1 \leq i \leq n$. However, deriving p_i analytically is difficult due to non-linear dependencies in (2). We therefore propose an algorithm which heuristically searches for valid values of p_i. To this end, let us first consider the following lemma providing an upper bound on the values of p_i.

Lemma 4. *If a node i follows our delayed-activation scheme and sends n packets within d_i according to Lemma 3, then its (constant) inter-packet time is upper bounded by $\hat{p}_i$, where l_i is node i's packet length:*

$$\hat{p}_i = \frac{d_i - l_i}{n}. \tag{3}$$

Proof. Without loss of generality, let us assume that node i is activated at time t_0. In order that n packets can be sent within $[t_0, t_0 + d_i]$, the n-th packet has to be sent at latest at $t_0 + d_i - l_i$. This way, node i's n-th packet finishes being sent exactly at $t_0 + d_i$.

On the other hand, according to our delayed-activation scheme, the transmission of a sequence of packets can be

Algorithm 2 Function *check_period()*

```
Require: n, i, p_temp, list of (l_j, p_j) for j < i
1: for j = 1 to i − 1 do
2:   if p_j < p_temp then
3:     p_long = p_temp
4:     p_short = p_j
5:   else
6:     p_long = p_j
7:     p_short = p_temp
8:   end if
9:   for k = 1 to n − 1 do
10:    if mod((k × p_long) / p_short) < l_long + l_short then
11:      return (false)
12:    else if p_short − mod((k × p_long) / p_short) < l_long + l_short then
13:      return (false)
14:    end if
15:  end for
16: end for
17: return (true)
```

delayed by at most p_i — see again Fig. 3. Hence, the first node i's packet will be sent at $t_0 + p_i$. Since p_i is assumed to be constant, it should fit $n-1$ times in an interval of length $d_i - l_i - p_i$ in the worst case. Now replacing p_i by $\hat{p}_i$ to denote the greatest possible p_i, we obtain $\hat{p}_i = \frac{d_i - l_i - \hat{p}_i}{n-1}$, which leads to $\hat{p}_i = \frac{d_i - l_i}{n}$. The lemma follows. □

4.3. Proposed algorithm

Alg. 1 computes values of p_i for a set of n transmit-only nodes, given packet lengths l_i and deadlines d_i with $1 \le i \le n$. The algorithm iterates over a list of ordered pairs (l_i, d_i) — see line 2, which has been sorted according to non-decreasing d_i. For each i, Lemma 4 is applied at line 4 to compute the longest possible p_i, temporally stored in p_{temp}, that allows meeting the deadline d_i. This way, the algorithm intends to maximize p_i. The greater the value of p_i, the less packets will be sent per time unit leading to less collisions.

This t_{temp} is checked by function *check_period()* as discussed later. If this check is successful, the value of t_{temp} is adopted for the current i — see line 8. If the check fails, t_{temp} is decremented in line 9 by a amount equal to *step*[1] as long as it remains greater than zero — see while-loop at line 5. If t_{temp} becomes zero, it means that *check_period()* was never successful and the algorithm returns *failed* at line 15. Otherwise, if the algorithm finds valid values of p_i for every i, these are returned as a list.

Alg. 2 shows the function *check_period()*, which is invoked from Alg. 1 and verifies a given value of p_{temp}. The algorithm is based on Theorem 1, i.e., it iteratively computes (2). If (2) holds for p_{temp} and all p_j for $j \le i-1$ in lines 9 to 15, i.e., the previously computed inter-packet times, p_{temp} is a valid value for p_i and *check_period()* returns *true* — otherwise it returns *false*. For this, *check_period()* requires the list of ordered pairs (l_j, p_j), i.e., the inter-packet times obtained so far, as well as index i of the node whose p_i is currently being computed. To simplify the computation of (2), p_{temp} and p_j are compared and stored in p_{long} and p_{short} depending on their values — see lines 2 to 8 in Alg. 2.

Note that, according to Theorem 1, (2) needs to be computed once with p_{long} and once with p_{short} in the numerator. The case with p_{long} in the numerator is checked in line 10, whereas the case with p_{short} in the numerator is checked in line 12. This way, it is possible to reduce the number of iterations that would be otherwise necessary.

Complexity. Note that the proposed algorithm iterates over the list of ordered pairs (l_i, d_i) — see again Alg. 1, which has a length of n elements. Each time, it computes p_{temp} and calls the function *check_period()*. This is repeated until a valid value of p_{temp} is found or p_{temp} becomes zero or less, where p_{temp} is decremented — starting from $\hat{p}_i$ in (3) — by a constant amount *step*. In other words, *check_period()* is called a maximum number of $\lfloor \frac{\hat{p}_{max}}{step} \rfloor$ times, where $\hat{p}_{max} = \max_{i=1}^{n}(\hat{p}_i)$.

On the other hand, *check_period()* shown in Alg. 2 iterates over the list of ordered pairs (l_j, p_j), i.e., the inter-packet times that have already been found in previous iterations, which can have a length of $n-1$ elements. In each iteration, it computes (2) a number of $n-1$ times. In other words, *check_period()* has a complexity $\mathcal{O}(n^2)$.

As a result, the overall complexity is $\mathcal{O}(\lfloor \frac{\hat{p}_{max}}{step} \rfloor \times n^3)$. This is a pseudo-polynomial complexity, since $\lfloor \frac{\hat{p}_{max}}{step} \rfloor$ depends on $\hat{p}_{max}$, i.e., on nodes' parameters.

5. Experimental Evaluation

In this section, we compare our proposed MAC protocol with other state-of-the-art approaches from the literature. To this end, we created an exemplary network and used parameters that are commonly found in the home-automation domain. That is, we assume a transmission speed of 128 kbit/s, short packet sizes of 3 bytes (resulting in $l_{max} = 187.5\,\mu s$) and a deadline of $d_{max} = 500$ ms unless otherwise noted. These parameters are used to evaluate the different MAC protocols with respect to their performance such as delay, computation time, etc. In particular, the maximum possible network size n_{max} is of importance, i.e., the number of nodes that can be safely accommodated in a network for a given deadline. This n_{max} results from the incurred delay, channel utilization and throughput of a MAC and constitutes a quality measure. The higher the resulting n_{max}, the better the corresponding MAC is.

5.1. Compared MAC protocols

Let us first discuss the different MAC protocols that are compared in the following experiments. For this, we have selected those that are designed for reliable communication

[1]The smallest meaningful value of *step* is equal to the time required for transmitting one bit on the communication channel.

in pure transmit-only networks and do not require special transceiver hardware. Note that the approaches in [6], [14], and [21] fail to meet these criteria and, hence, our selection reduces to the following three: The *proposed* approach from this paper, the *analytic* scheme from [15] and the *extensive* protocol introduced in [4][5].

These three MAC protocols consist in nodes sending their data as sequences of redundant packets with carefully chosen inter-packet times. This way, they generate unique transmission patterns ensuring that at least one packet of a sequence reaches its destination in the worst case. The main difference between these approaches is the methodology for selecting inter-packet times, which yields different results. In case of the *analytic* scheme, these are derived analytically using a simple formula [15]. However, results are typically more pessimistic leading to prolonged delays and reduced network sizes (i.e., a small n_{max}).

On the other hand, the *extensive* scheme uses Integer Linear Programming (ILP) to shorten inter-packet times and therefore improve n_{max}. However, due to the high complexity of the problem, only small network sizes with $n \leq 6$ could be calculated by the authors [4]. As a result, the same authors introduced a heuristic algorithm in [5], which greatly reduces computation times, however, at the cost of lower performance.

The proposed approach in this paper also implements a heuristic search as shown in Alg. 1. As explained in Section 4.3, instead of first starting with short inter-packet times and gradually increasing them as it becomes necessary, we instead start with the maximum possible inter-packet times that allow meeting deadlines and stepwise decrease them as much as possible. In contrast to the other two, the *proposed* approach allows for arbitrary node types with varying deadlines and packet lengths and does not require enforcing transmission pauses after a sequence, which makes it more general and effective.

5.2. Maximum number of nodes versus deadline

Next, let us analyze n_{max} for the case of both the deadline and packet size being the same for all nodes, as displayed in Fig. 4. Clearly, for an increasing deadline, the maximum number of nodes n_{max} increases with all methods, since longer deadlines allow sending more packets with less collisions. The greatest number of nodes can be realized using the *extensive* scheme, since it has the most elaborate search optimization and allows multiple period values for a single node. It is, however, computationally very expensive — as we discuss later — and yields only around 20 % better results than the *proposed* scheme. For a deadline of 1500 ms, it allows for around 32 nodes, whereas the *proposed* method can reach approximately 25 nodes. The *analytic* approach only allows 17 nodes for this deadline.

Our proposed scheme is depicted twice in Fig. 4: The *proposed* curve depicts the results from the unmodified algorithm as shown in Alg. 1, whereas the *proposed** curve shows a slightly optimized variant. To explain the difference

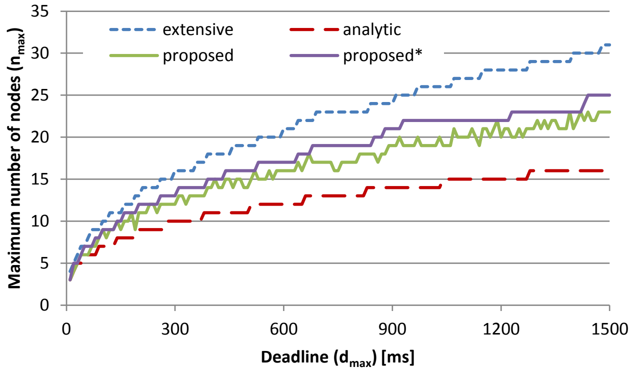

Figure 4: The maximum number of nodes n_{max} versus deadline is depicted. The *proposed* curve shows the behavior of Alg. 1, whereas the *proposed** curve shows a slightly optimized variant Alg. 1.

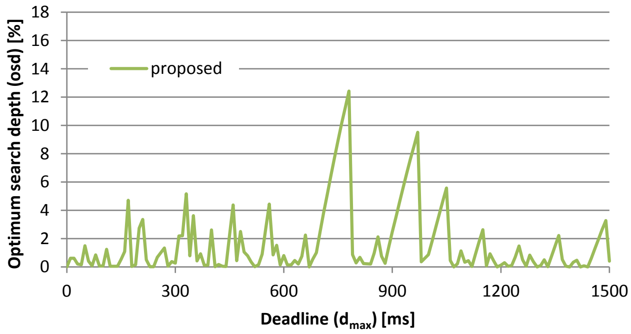

Figure 5: The *optimal search depth* (*osd*) is depicted for the *proposed* scheme. Different *osd* can be used to find optimized values of n_{max} as shown by the *proposed** curve in Fig. 4.

between them, let us again consider Alg. 1. This algorithm gradually searches for periods for each node and takes the first value that is proven valid by function *check_period()*. However, this first choice can be further optimized. In particular, we observe in Fig. 4 that *proposed* sometimes finds a greater n_{max} for a smaller deadline. This suggests making Alg. 1 search for a local optimum in the close vicinity of a deadline, which leads to the *proposed** curve in Fig. 4. That is, for a given deadline d_i, the local optimum n_{max} is searched in $[d_i \times (1 - osd), d_i]$, where $osd \in [0, 1]$ denotes what we call the *optimum search depth*. This gives the length of the search interval in relation to d_i: an $osd = 0$ means no search for an optimum; an $osd = 1$ means a full search in $[0, d_i]$. Clearly, the greater the value of *osd*, the more computation time is required.

Fig. 5 shows the values of *osd* that leads to a locally optimum n_{max}. As it can be noted, an $osd = 0.13$ is the greatest *osd* required to find the optimum n_{max} at a 750 ms deadline. On the other hand, the average *osd* is around 0.05, i.e., 5 % of the deadline, as illustrated in Fig. 5.

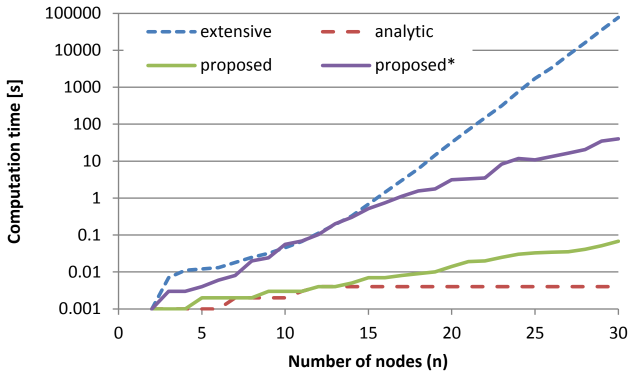

Figure 6: Computation time versus number of nodes is depicted. The *proposed* curve shows the computation time without optimization, whereas *proposed** shows the computation time with an *osd* of 10 %.

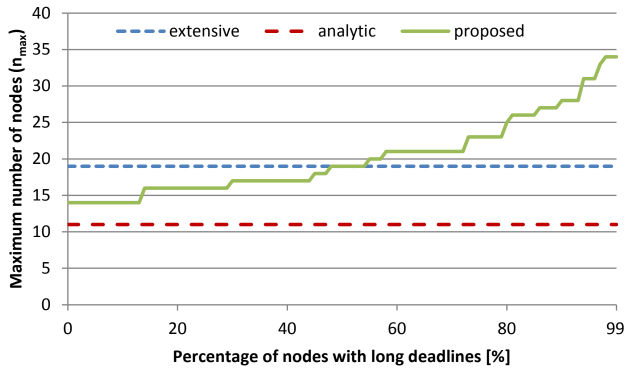

Figure 7: The maximum number of nodes n_{max} is depicted for a system with varying ratios of two node types: fast nodes with a short deadline $d_{short} = 500$ ms and slow nodes with a long deadline $d_{long} = 10$ s.

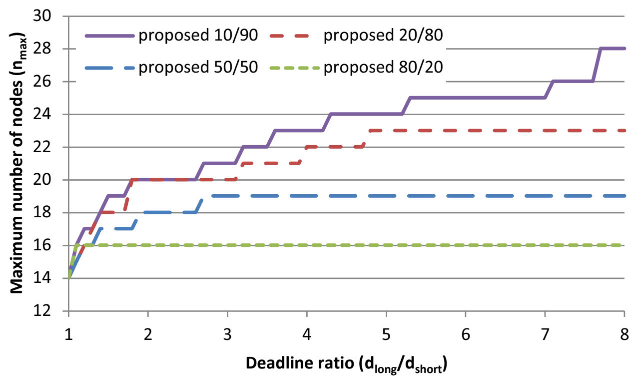

Figure 8: The maximum numbers of nodes n_{max} is shown for varying deadline ratios. Each system contains two node types: fast nodes with a short deadline $d_{short} = 500$ ms and slow nodes with a long deadline d_{long} that vary between 500 ms (ratio of 1) and 4 s (ratio of 8). The different curves show the results for different amounts of fast/slow nodes in the system, e.g., the *proposed 20/80* curve represents a system with 20 % fast nodes and 80 % slow nodes.

5.3. Computation time

Let us now discuss the computation overhead by the different approaches, i.e., the time that is needed to calculate inter-packet times for all nodes. All calculations were performed on an AMD A8-6500 processor at 3.5 GHz and with 8 GB of memory. Results are depicted in Fig. 6.

As expected, the *analytic* scheme has the lowest computation time, since inter-packet times can be calculated analytically with linear complexity. Its computation time for 30 nodes is below 5 ms. The *proposed* scheme, i.e., without optimization, offers the second lowest computation time. Its computation time slowly rises and is around 33 ms for 30 nodes. With optimization and an *osd* of 10 % for the *proposed** scheme, the computation time rises to 40 s for 30 nodes.

In case of the *extensive* scheme, the multiple inter-packet times per single node and the extensive optimization lead to considerably higher computation overhead. For 30 nodes, its computation time is around 28 hours as shown in Fig. 6. Although these algorithms normally run off-line on a fast computer, high computation times negatively impact testing and debugging of a system, in particular, during the production and early deployment phase.

5.4. Effect of different deadlines

Next, we examine how n_{max} is influenced when nodes with different deadlines are considered. Fig. 7 shows an exemplary system with two types of nodes: fast nodes with a deadline d_{short} of 500 ms (e.g., light switches in a home-automation setting) and slow nodes with a deadline d_{long} of 10 s (e.g., temperature sensors).

Since both the *extensive* and *analytic* scheme are not designed for systems with different deadlines, they cannot take advantage of the long deadline and have to assume the short deadline for all nodes. Their n_{max} consequently do not change for an increasing number of nodes with long deadlines and stay at the value of 0 % slow nodes (i.e., only fast nodes). In contrast, the *proposed* scheme allows a higher n_{max} for a rising percentage of nodes with long deadlines, since it was specially designed to benefit from this. For 50 % or more slow nodes, the *proposed* scheme results in the greatest n_{max} — see Fig. 7, even greater than the *extensive* scheme with its considerably more elaborate optimization.

In Fig. 7, the deadlines d_{short} and d_{long} are fixed to $d_{short} = 500$ ms and $d_{long} = 10$ s, i.e., a deadline ratio d_{long}/d_{short} of 20. Since this ratio also influences the maximum number of nodes n_{max} by the *proposed* scheme, we further investigate this effect in Fig. 8. Here, we again have two node types: fast nodes with a deadline d_{short} of 500 ms and slow nodes with a deadline d_{long}, which in this case, can be varied from 500 ms (ratio of 1) to 4 s (ratio of 8). The number of fast nodes compared to the number of slow nodes within the system is fixed, e.g., the *proposed 20/80* system is composed of 20 % fast nodes and 80 % slow nodes. As depicted in Fig. 8, the greater the amount of nodes

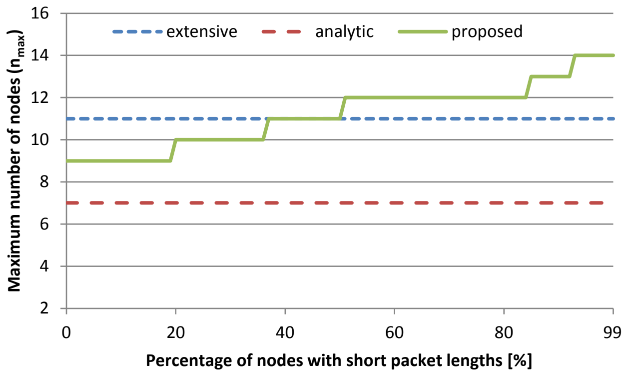

Figure 9: The maximum number of nodes n_{max} is depicted for a varying ratio of the two node types: slight nodes with a short packet length l_{short} corresponding to a 3-byte packet and intense nodes with a long packet length l_{long} corresponding to 12 bytes.

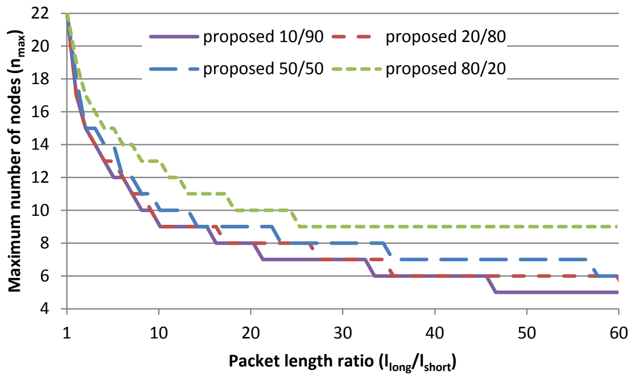

Figure 10: The maximum number of nodes n_{max} is shown for varying packet length ratios. Each system contains two node types: slight nodes with a short packet length l_{short} corresponding to a one-byte packet and intense nodes with a long packet length l_{long} that can be varied between the length of 1 byte (ratio of 1) and of 60 bytes (ratio of 60). The different curves show the results for different amounts of slight/intense nodes in the system, e.g., the *proposed 20/80* curve is a system with 20 % slight nodes and 80 % intense nodes.

with long deadlines in the system, the higher n_{max} is for an increasing d_{long}/d_{short} ratio. However, n_{max} saturates for high deadline ratios, i.e., making d_{long} greater has no positive effect on n_{max} from a certain point onwards.

This saturation point is reached earlier, i.e., for smaller deadline ratios, the lower the number of slow nodes is. That is, systems with a small amount of slow nodes, for example, the *proposed 80/20* system with 20 % slow nodes, have little benefit from a greater d_{long}. Systems with a high number of slow nodes, for example, the *proposed 10/90* system with 90 % slow nodes, benefit from greater values of d_{long}.

The reason is that, in the worst case, a fast node with a short deadline can be activated multiple times within the longer deadline of a slow node. This means that the slow node can potentially collide with more packets of the fast node within its long deadline. This effect increases with the number of fast nodes and the ratio of d_{long}/d_{short} — as shown in Fig. 8 — leading to the observed saturation.

5.5. Effect of different packet lengths

Let us now analyze the effect of different packet lengths on n_{max}. In Fig. 9, we regard a system composed of two node types: slight nodes with short packet lengths $l_{short} = 62.5\,\mu s$ corresponding to a one-byte packet, and intense nodes with a longer packet lengths $l_{long} = 750\,\mu s$ corresponding to a 12-byte packet.

Varying the percentage of nodes with short deadlines does not influence n_{max} of the *analytic* and *extensive* schemes, as both of them are not designed for different packet lengths and, therefore, have to assume l_{long} for all nodes. In contrast, as illustrated in Fig. 9, the *proposed* scheme leads to improved results when combining different packet lengths. The values of n_{max} increase as the percentage of nodes with a short packet lengths increases, outperforming the *extensive* scheme from 40 % slight nodes — with short packet lengths — onwards. Note that long packets leads to a higher channel utilization and, hence, n_{max} decreases for all algorithms. That is, values of n_{max} in Fig. 9 are smaller than those of Fig. 7 although curves look similar.

Further, n_{max} is also influenced by the packet length ratio between different nodes, i.e., how big the packet length of a node is compared to another node. In Fig. 10, similar to Fig. 9, we use a system with two different node types: slight nodes with a short packet length l_{short} corresponding to a one-byte packet and intense nodes with a long packet length l_{long} that can be varied between the length of 1 byte (ratio of 1) and the length of 60 bytes (ratio of 60). Now, the amount slight compared to intense nodes is fixed, e.g., the *proposed 20/80* curve is a system with 20 % slight and 80 % intense nodes.

We can clearly see the negative impact of the packet size on n_{max}: As the packet length ratio l_{long}/l_{short} rises, l_{long} and, hence, the channel utilization increase. A high channel utilization implies less time for other nodes to send their packets, therefore, the maximum number of nodes n_{max} decreases in a considerable manner.

A l_{long}/l_{short} ratio of 1 means the packet lengths are the same for both node types and all curves displayed in Fig. 10 have the same n_{max}. When the packet length ratio rises, the n_{max} values of the systems with more intense nodes decrease more rapidly, e.g., the *proposed 10/90* with 90 % intense nodes falls deeper than the *proposed 80/20* system with only 20 % intense nodes. In contrast to Fig. 8, note that there is no saturation effect for high packet length ratios, but n_{max} tends to 0. In summary, long packets should be avoided in order to achieve an acceptable n_{max}. Alternatively, the deadline of nodes with long packets should be increased to compensate this negative effect.

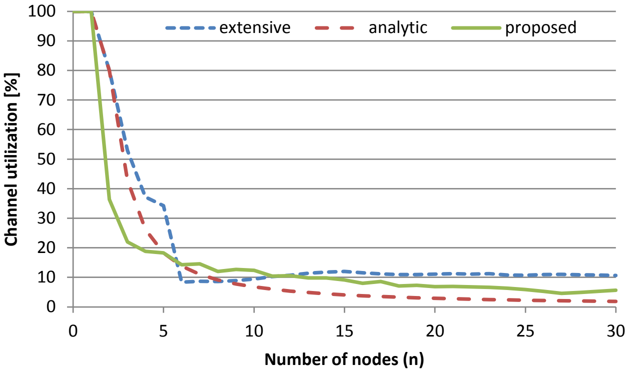

Figure 11: The maximum channel utilization of the network, i.e., the ratio of the transmission times of all nodes in one sequence to the length of one sequence.

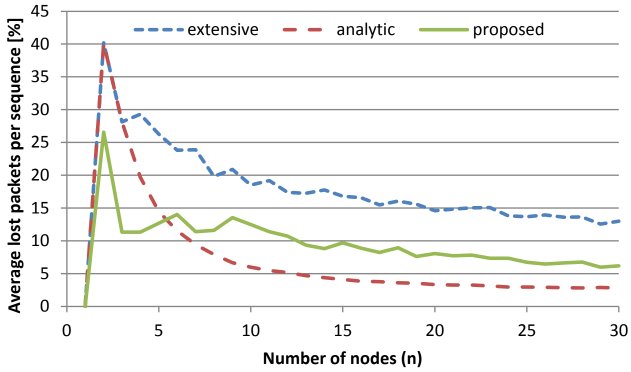

Figure 12: The amount of lost packets per sequence is depicted for the compared transmission schemes. Each scheme was simulated for different numbers of transmitters n; each time 100,000 packet sequences have been sent.

5.6. Channel utilization

Let us analyze the maximum channel utilization of the network, i.e., the ratio of the transmission times of all nodes in a sufficiently large time interval to the length of the interval. For the sake of comparison between approaches, we again assume that deadlines and packet sizes are the same for all nodes. In particular, this states how much of the channel capacity is used by the different MAC protocols. Note that channel utilization can be easily converted to throughput by multiplying it by the transmission speed, i.e., 128 kbit/s in our example.

In Fig. 11, we can see that the channel utilization starts at 100 % for $n = 1$ and decreases for higher n, since more redundant packets are sent and inter-packet times become larger. For example, for $n = 30$, the utilization drops to 10 % for the *extensive*, 6 % for the *proposed* and 2 % for the analytic scheme, which translates to a throughput of 13, 8 and 3 kbit/s respectively. In case of the *proposed* scheme, the channel utilization for $n \leq 4$ is lower compared to the *analytic* scheme. This is due to the search strategy of Alg. 1, which, in contrast to the other two approaches, first assumes long inter-packets times and then gradually decreases them. For $n \geq 5$, however, this effect reverses. Note that the rippling of the *proposed* and *extensive* schemes in Fig. 11 is due to their non-linear nature.

Although the channel utilization (and consequently the throughput) of the compared MAC protocols is low with regard to other throughput-optimized approaches, e.g., TDMA, it is sufficient for home-automation networks; these are characterized by the transmission of small control packets rather than large bulks of data. In particular, one packet must be received before a certain deadline passes, for which low complexity and cost-effectiveness are of more importance. In addition, note that the *proposed* scheme expectedly outperforms the others when considering different deadlines and/or packet sizes per node.

5.7. Simulation-based comparison

Complementary to the numerical analysis in the previous sections, we further evaluate the different MAC protocols with respect to their simulated average performance. To this end, we used OMNeT++ [18] and MiXiM [12], an extension for mobile and wireless networks, to simulate our network with different physical parameters and to record statistical values for a large numbers of transmissions.

The simulated network consists of one receiver and a selectable number of n transmitters that are all within range of one another and, hence, interfere with one another. To this end, we assume that the network has been set up correctly in a way that physical effects, such as fading, shading and reflection of radio waves do not cause packet loss and can therefore be neglected. The recording and processing of simulation data is done by the framework at runtime. In particular, the time stamps of the different packets sent are compared to determine whether packets overlap and, hence, are lost.

Fig. 12 shows how many packets are lost on average within a packet sequence as n increases, which illustrates the amount of unnecessary redundancy. A lower packet loss means that more packets from a sequence reach their destination, i.e., sending less packets would have been sufficient to guarantee reliability in this case. However, Fig. 12 shows the average case, whereas all compared transmission schemes were designed to guarantee reliability in the worst case.

All three transmission schemes start with 0 % packet loss for $n = 1$, which then rises to a peak value for $n = 2$ and gradually decreases for higher $n > 2$. The packet loss depends on both the inter-packet separations, i.e., how much time is in between two consecutive packets of a sequence, the amount of packets each node sends, and the number of nodes transmitting simultaneously. A greater number of nodes n will lead to a greater amount of packets per sequence — equal to n, which will therefore increase the amount of packet loss. On the other hand, the inter-packet

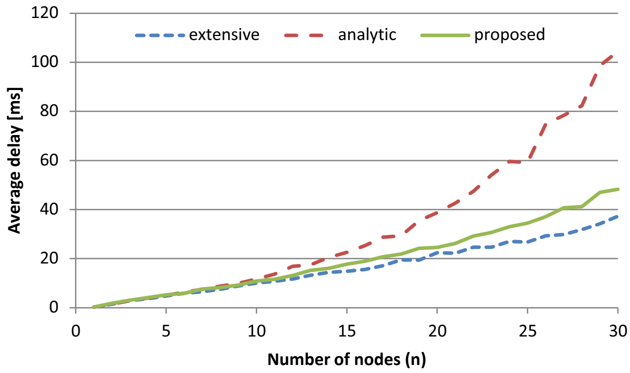

Figure 13: The average delay for receiving a packet is depicted with respect to an increasing number of transmitters n; each time 100,000 packet sequences have been simulated.

times p_i also increase for higher n and, consequently, the amount of packet loss reduces. In this case, the effect of longer inter-packet times dominates over that of increasing numbers of packets and nodes. Therefore, the amount of packet loss decreases for higher values of n. With $n = 1$, we do not have any collisions, hence, the peak value of lost packets occurs at $n = 2$.

The inter-packet times by the *analytic* scheme are obtained analytically and grow linearly for higher n leading to a smooth curve in Fig. 12. Both the *extensive* and *proposed* scheme are of non-linear nature instead. They are able to shorten the inter-packet times and achieve better maximum numbers of nodes n_{max} per deadline — see Fig. 4. However, since p_i grown non-linearly for a rising n, curves in Fig. 12 slightly fluctuate. In addition, shorter inter-packet times lead to a higher number of collisions than with the *analytic* scheme, whereby the *extensive* scheme has the highest number of packet collisions, since its inter-packet times are the shortest among all the three algorithms. Finally, it should be noticed that the *analytic* approach results in shorter periods for $n < 5$ than the *proposed* scheme, hence, leading to more packet collisions as depicted in Fig. 12.

Let us now consider the average communication delay, i.e., the time measured from the activation of a node until the first successful reception of a packet. As shown in Fig. 13, the delay rises exponentially for an increasing n independent of the algorithm used. Although the *analytic* scheme has the lowest packet loss per sequence, which will typically result in one of the first packets of a sequence to be received, the inter-packet times are greater than those of the other algorithms. This makes the *analytic* approach have the greatest delay of all three transmission schemes.

The *proposed* and the *extensive* schemes have shorter inter-packet times p_i and, therefore, shorter average delays. The *extensive* scheme has the shortest delay, whereas the delay of the *proposed* scheme is comparably low, however, at a lower computation overhead.

6. Practical Factors

In this section, we extend our analysis to consider practical factors, in particular, clock drift, external interference and reduced packet numbers per sequence.

6.1. Clock Drift

All clocks used in electronic devices show a deviation in frequency with respect to each other, i.e., they *count* time at different rates. This deviation is known as clock drift and normally depends on a number of different factors such fabrication-induced variability, operating temperature, etc. As a result, since transmit-only nodes cannot be synchronized [6], they will unavoidably have different time scales.

Since a node *counts* for p_i time before sending a packet, a clock drift leads to an *absolute* waiting time $\bar{p}_i$ different than p_i, i.e., the time without clock drift. As a result, this needs to be considered when selecting inter-packet times. In particular, (3) needs to be modified to guarantee that the node sends its n packets within d_i independent of clock drift. To this end, let us denote by $\Delta\hat{p}_i$ the maximum possible clock deviation (with respect to an ideal, non-drifting clock) in an interval of length $\hat{p}_i$. Analogously to Lemma 4, we proceed as follows to incorporate clock drift into (3):

$$\begin{aligned} (n-1)\cdot(\hat{p}_i + \Delta\hat{p}_i) &\leq d_i - (\hat{p}_i + \Delta\hat{p}_i) - l_i, \\ \hat{p}_i &\leq \frac{d_i - l_i - n\cdot\Delta\hat{p}_i}{n}. \end{aligned} \tag{4}$$

Similarly, (2) needs to be modified to consider clock drift as shown in the following:

$$\mathrm{mod}\left(\frac{k_i \times p_i}{p_j}\right) \geq l_i + l_j + \Delta\hat{p}_i + \Delta\hat{p}_j, \tag{5}$$

where again $\Delta\hat{p}_i$ and $\Delta\hat{p}_j$ denote the maximum possible clock deviation in an interval of length $\hat{p}_i$ and $\hat{p}_j$ respectively.

Note that (4) and (5) require knowing $\Delta\hat{p}_i$ and $\Delta\hat{p}_j$ (which depend on $\hat{p}_i$ and $\hat{p}_j$ that are unknown). However, we know that clock deviation due to clock drift increases with the length of the considered time interval. Since $\hat{p}_i < \tilde{p}_i$ and $\hat{p}_j < \tilde{p}_j$ hold for $\tilde{p}_i = \frac{d_i}{n}$ and $\tilde{p}_j = \frac{d_j}{n}$, we have that $\Delta\hat{p}_i < \Delta\tilde{p}_i$ and $\Delta\hat{p}_j < \Delta\tilde{p}_j$ also holds, where $\Delta\tilde{p}_i$ and $\Delta\tilde{p}_j$ are the maximum clock deviations in the intervals of length $\tilde{p}_i$ and $\tilde{p}_j$. As a result, to resolve the above dependency, we can safely replace $\Delta\hat{p}_i$ and $\Delta\hat{p}_j$ in (4) and (5) by $\Delta\tilde{p}_i$ and $\Delta\tilde{p}_j$.

If clock drift is not considered, when selecting p_i for all nodes, it might happen that there are multiple collisions between any two different nodes in one sequence of packets, which results in a transmission reliability less than 100 % in the worst case. However, since our transmission scheme transmits n redundant packets, it is generally robust against clock drift as shown next.

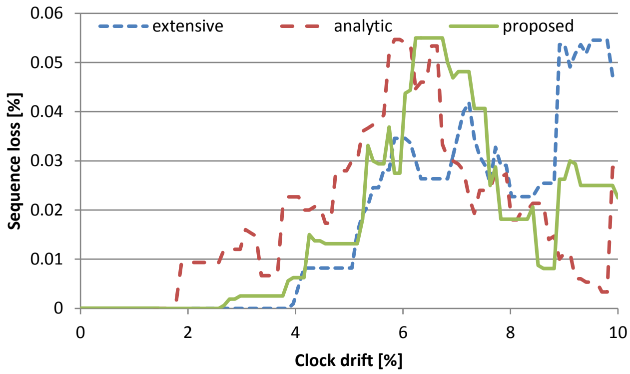

Figure 14: The average sequence loss, i.e., when all packets of a transmission are lost, is depicted with respect to an increasing clock drift; each time 100,000 packet sequences have been simulated.

Simulation of clock drift. We now examine the effects of clock drift on the *proposed*, *extensive* and *analytic* scheme by simulation. To this end, we regard an exemplary network with $n = 10$ nodes and the same simulation parameters as used in Section 5.7.

We consider that each node runs on a local clock with a random drift in $[-\Delta_{drift}, +\Delta_{drift}]$. Here, Δ_{drift} is the maximum drift of the current iteration, which is slowly increased from zero to 10 %. Note that a drift of 10 % means that the clock deviates 10 % from its nominal frequency. For example, if we consider a 1 MHz crystal, a 10 % drift would cause it to run at either 1,100,000 Hz or 900,000 Hz.

Fig. 14 shows how clock drift affects the average loss of packet sequences, i.e., when all packets are lost. For this, we have varied Δ_{drift} in 100 steps between 0 and 10 % simulating 100,000 packet sequences at each step. As expected, a rising clock drift leads to a slowly rising loss of packet sequences for all three MAC protocols. This growth is strongly non-linear, both because of the randomness of the experiment and the dependency of inter-packet times on clock drift. That is, some values of clock drift generate more collisions for specific inter-packet times leading to local peaks in the loss of packet sequences. For example, the *proposed* scheme loses more packets for a drift of 6 % than for 8 % on average. However, due to high non-linearity, this changes if a another setting is used, for example, a different n.

For drift values < 1.5 %, we have observed no loss of packet sequences. As a result, since even cheap crystal oscillators can guarantee a 100 ppm = 0.01 % accuracy, we can safely neglect clock drift when designing a home automation network with any of the compared approaches.

6.2. Packet numbers vs. reliability

So far, we considered the case that each node in the system sends n packets leading to fully reliable communication within the network, i.e., without external interference. In some applications, however, it is favorable to trade reliability for a reduced number of packets in order to save energy. For example, pressing a light switch is not safety critical and the user can press it again, if the light does not turn on/off. Clearly, the reliability should still be high enough not to negatively impact the system quality.

In this section, we present a method to calculate the transmission reliability for each node i, if it sends a reduced number of $k_i \leq n$ packets. This allows us to adjust the packet numbers individually for each node to save energy whenever data loss can be tolerated. Note that, by changing the number of packets per sequence and keeping the periods found by Alg. 1, we ensure that nodes with mixed reliability can coexist without affecting each other's performance. Further, this allows us to change packet numbers dynamically depending on the message priority. For example, a node can use higher k for important alarm messages and a lower k for less important messages to save energy.

Now, let us assume that all nodes in the network are activated and sending packets and, hence, produce interference. Since nodes send packets periodically, the probability of interfering with a packet of a node i at the communication channel is $\sigma_i = \frac{l_i}{p_i}$ [19], i.e., the node's duty cycle. However, by properly selecting inter-packet times, we know that the proposed technique guarantees that, within one sequence of packets, there will be at most one interference with another node in the network.

Without loss of generality, let us assume that nodes are sorted in order of decreasing σ_i, i.e., $\sigma_1 \geq \sigma_2$ and $\sigma_2 \geq \sigma_3$, etc. In other words, the smaller a node's index, the higher the probability of interfering with it. As a result, the greatest probability of losing $n - 1$ packets is that of node n.

Let us now define $q_i(x)$, i.e., the probability that exactly x out of k_i packets sent by node i are lost due to internal interference. This results in:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j \neq i}^{n} \sigma_j \left(\sum_{l=1;\ l \neq i,j}^{n} \sigma_l \cdots \right) \right), \quad (6)$$

where $1 \leq x \leq (n-1)$ and $q_i(n) = 0\ \forall i$, i.e, the probability of losing all n packets is zero for every node i. Note that there are exactly x summations in the above equation.

A detailed derivation of (6) can be found in Appendix B. In particular, $\binom{k}{x}$ is the binomial coefficient and accounts for the different combinations of packets that may collide within a sequence. For example, if we want the probability of $x = 1$ packet out of $k = 3$ being interfered, we have $\binom{3}{1} = 3$ possibilities: Either the first, the second or the third packet may be interfered. Further, there can be at most one collision with any other node per sequence, hence, $q_i(x)$ does only depend on k_i and not on k_j of the other nodes j with $j \neq i$.

The remaining part of (6) considers the combinations of different nodes that can cause collisions within the sequence. For example, if we have 4 nodes, each packet of node 4 can be either corrupted by node 1, 2 or 3. In case multiple packets are lost ($x > 1$), different combinations of nodes

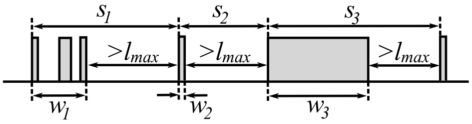

Figure 15: An exemplary sequence of external interference pulses at the communication channel. In this case, the maximum possible duty cycle σ_{ext} is equal to $\frac{w_3}{s_3}$. Since no packet can possibly be sent in a time interval that is less than l_{max}, external interference pulses that are separated by less than l_{max} are considered to be one single large pulse. This is the case of the first three pulses in the example, which are merged into one single pulse of width w_1.

can cause these losses. For example, if two packets are lost ($x = 2$), this can be because of collisions with node 1 and 2 or node 2 and 3, etc.

Finally, equation (6) can be simplified to (7) by assuming the same σ_i for all nodes, i.e., $\sigma_i = \sigma\ \forall i$:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\prod_{j=1}^{x} (n-j) \right) \cdot \sigma^x. \tag{7}$$

Example. To illustrate the previous equations, let us calculate the loss rate of an exemplary network, i.e., the probability of loosing all k packets per sequence $q_i(k)$ with $1 \leq k \leq (n-1)$. To this end, we set $n = 4$ and the duty cycle of each node to $\sigma = 0.035$ — this is greatest duty cycle found by Alg. 1 for $n = 4$, $l_i = 187.5\,\mu s$, and $d_i = 500\,\text{ms}$. Using (7), we obtain loss rates of $q(4) = 0\,\%$, $q(3) \leq 0.026\,\%$, $q(2) \leq 0.74\,\%$ and $q(1) \leq 10.5\,\%$. As we can see, a higher k results in a smaller $q(k)$. However, only relatively low k are required to achieve loss rates below $1\,\%$, e.g., only $k = 2$ in this case. This once again shows that full reliability, i.e., a loss rate of $q(n) = 0\,\%$, requires high costs in the form of increased packets numbers and delays. On the other hand, whenever data loss can be tolerated to some extent, the number of redundant packets can be reduced to save energy. For example, reducing k to 2 halves the energy consumption and only results in an average expected packet loss of less than $1\,\%$.

However, reducing the packet numbers has further consequences, for example, when external interference is present. This will be discussed in the following section.

6.3. External Interference

External interference can occur, for example, when microwaves, wireless toys, etc. are turned on, or when there exist neighboring WSNs that have not been regarded during the design phase. Existing approaches [4][15] assume that this interference is negligible, since nodes are shielded by walls and a careful selection of carrier frequencies can be performed. However, in most applications this might not be the case and system performance can be jeopardized by external interference.

In order to model external interference, we assume that its maximum *duty cycle* — denoted by σ_{ext} — can be determined, i.e., the greatest possible ratio between pulse width w_i to inter-pulse separation s_i of external interference — see Fig. 15:

$$\sigma_{ext} = \max_{\forall i} \left(\frac{w_i}{s_i} \right), \tag{8}$$

where $i \in \mathbb{N}_{>0}$ is an index identifying the particular pulse. This σ_{ext} can be obtained, for example, by measuring at the communication channel for a sufficiently large time window; or this may also be known from previous experience.

Note that external interference pulses separated by less than l_{max}, i.e., the maximum packet length in the system, are considered to be one single large pulse. This is because the minimum overlapping with an external interference pulse may already yield packet loss. Hence, no data packet can be sent between such pulses as illustrated in Fig. 15.

This σ_{ext} gives also the greatest probability of encountering an external interference pulse at the communication channel [19] — note that $\sigma_{ext} \leq 1$ holds. A $\sigma_{ext} = 1$ means that external interference occupies the full channel and, hence, no reliable communication is possible. This probability is clearly independent of $q(n)$ in (6), i.e., the probability of packet loss due to internal interference.

In general, data packets can be either lost by external or by internal interference. If we consider an exemplary network with $n = 4$ nodes and $k = 2$ data packets per sequence, we have the following possible combinations that lead to full packet loss: (i) all packets are lost due to external interference, (ii) one packet is lost due to internal and one due to external interference and (iii) all packets are lost internally. Changing k to higher values results in more combinations of mixed interference in (ii). For example, $k = 3$ packets can be either lost by 1 or 2 internal collisions and 2 or 1 external ones. If $k = n$, there is no full packet loss due to internal collisions, since the *proposed* scheme prevents this by construction.

Now, let us generalize the previous example and calculate $\hat{q}_i$, i.e., the probability to loose all k_i packets of a sequence of node i by internal and external interference. This results in:

$$\hat{q}_i = \sigma_{ext}^{k_i} + \sum_{j=1}^{k_i} q_i(j) \cdot \sigma_{ext}^{k_i - j} \cdot \bar{\sigma}_{ext}^{j}, \tag{9}$$

where $1 \leq i \leq n$, $1 \leq k_i \leq n\ \forall i$ and $q_i(j)$ is the probability of node i to loose j packets internally as per (6) or (7).

The first part of (9) considers the probability to lose all packets externally. The second part combines internal and external interference as well as just internal interference when $j = k$, i.e., the last term of the summation.

Simulation of external interference. We now analyze the effects of external interference on the *proposed*, *extensive* and *analytic* scheme by simulation. For this, we again regard an exemplary network with $n = 10$ nodes and the same parameters as used in Section 5.7. Fig. 16 shows the loss of packet sequences for an increasing duty cycle of external interference pulses.

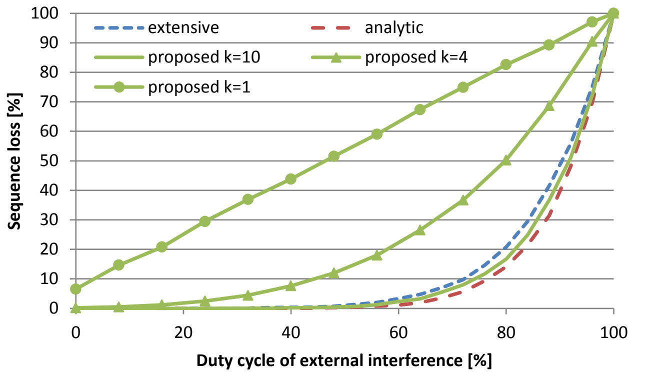

Figure 16: The average sequence loss, i.e., when all packets of a transmission are lost, is depicted for an increasing level (duty-cycle) of external interference; each time 100,000 packet sequences have been simulated.

We can see that an increasing noise level leads to a higher loss, or lower reliability for all transmission schemes. With 100 % interference, i.e., the channel is blocked, data transmission is not possible anymore and the resulting loss is 100 %. In general, a higher number of packets (i.e., k) results in a higher robustness against both internal and external interference, hence, loss rates are lower. In case of the *analytic* and *extensive* schemes — both send $n = 10$ packets per sequence — loss rates are similar to the *proposed* protocol with $k = 10$. The slight differences between them are related to the techniques used to select inter-packet times. That is, the inter-packet times are shorter in the case of the *extensive* scheme leading to a slightly higher loss of packet sequences, and longer for the *analytic* scheme leading leading to a slightly lower loss.

At $\sigma_{ext} = 0$, we can see again how many sequences of packets are lost by the different algorithms without external interference. This is zero for the *extensive*, *analytic* and *proposed* schemes with $k = 10$, i.e., these are fully reliable without external interference. However, 1% and 8% of packet sequences are lost when sending $k = 4$ and $k = 1$ packets with the *proposed* scheme. As discussed before, sending less packets allows reducing energy consumption.

In summary, a higher number of packets results in excellent reliability even under high noise levels of up to 60 % duty cycle. Considering that the average interference within a building is well below this level [4][15], external interference will not significantly affect the reliability by the above MAC protocols. On the other hand, if the *proposed* scheme sends less packets to save energy, its robustness against (external) interference decreases, which must regarded at design time.

6.4. Network reconfiguration

Network reconfiguration poses a problem for unidirectional networks, since sensor nodes are transmit-only and can therefore not receive configuration data. This makes network changes, for example, when additional nodes are deployed, a cumbersome procedure requiring nodes to be reconfigured manually. A way to counteract this problem is to design the network for more nodes than those actually deployed to allow for later extensions. This, however, reduces the energy efficiency and may not always be feasible for certain deadline constraints.

Changing the network size, changes the number of packets n per sequence, as well as the nodes' periods p_i. In [4][15], these periods must be recalculated completely, which is computational intensive, especially, in [4]. When adding a new node to the network, our scheme allows keeping existing periods. One can then use Alg. 1 to find additional periods that are compatible. However, the network size must still be reconfigured, therefore, a manual change cannot be avoided. Similar to existing home-automation systems, such as Intertechno [2], this can be easily realized via a small rotary switch at the back of each node.

6.5. Hidden Terminal

The hidden node or hidden terminal problem poses a challenge for many transmission schemes and leads to unavoidable packet loss, increased delay, etc., especially for bidirectional transmission. In unidirectional systems, however, nodes are not aware of each other, since no synchronization or carrier sensing can be performed. Reliability is achieved by sending redundant data packets to guarantee that one reaches its receiver in the worst case. This means that potential collisions with hidden nodes are already regarded and, hence, the occurrence of hidden terminals does not affect the unidirectional approaches from [4][15] to the proposed one in this paper.

7. Concluding Remarks

In this paper, we proposed a MAC protocol for designing highly reliable home-automation WSNs based on unidirectional nodes. More specifically, our technique consists in each transmit-only node sending a sequence of n packets — proven to be safe — with constant inter-packet times p_i, being n the number of transmit-only nodes in the network. This is suitable for home-automation networks, which typically consist of a set of WSNs with a reduced number of nodes.

Neglecting external interference, we showed that it is possible to select inter-packet times to guarantee *full* reliability, i.e., at least one packet of each node reaches its destination within a specified deadline in the worst case. In contrast to similar approaches from the literature [5][15][21], our technique is based on a more general model that allows describing arbitrary deadlines and packet sizes for each node. This makes our approach particularly suitable for heterogeneous home-automation networks improving not only delay and robustness, but also allowing us to safely accommodate more nodes in a network.

We further analyzed the effects of practical factors such as clock drift, external interference, etc. and evaluated how reducing the number of redundant packets affects reliability and energy consumption. To this end, we performed extensive experiments based on OMNeT++. These show that the proposed MAC technique is more general and outperforms known approaches from the literature being, at the same time, robust against clock drift and external interference.

As future work, we plan to extend the proposed MAC towards a hybrid network, i.e., where bidirectional nodes coexist with unidirectional ones. Bidirectional nodes can enhance functionality of the system, for example, by data forwarding, clustering, etc., while the majority of the network remains unidirectional for cost and energy efficiency. Further, we plan to include application specific data to estimate when a certain node most likely starts transmitting. This can further improve inter-packet times and therefore increase the overall performance of the protocol.

References

[1] HomeEasy. http://www.elro.eu/en, October 2017.

[2] Intertechno. http://www.intertechno.at/, October 2017.

[3] Z-Wave. http://www.z-wave.com, October 2017.

[4] B. Andersson, N. Pereira, and E. Tovar. Delay-Bounded Medium Access for Unidirectional Wireless Links. In *Proceedings of International Conference on Real-Time Networks and Systems (RTNS)*, 2007.

[5] B. Andersson, N. Pereira, and E. Tovar. Delay-Bounded Medium Access for Unidirectional Wireless Links. Technical report, CISTER - Research Centre in Real-Time and Embedded Computing Systems, 2007.

[6] B. Blaszczyszyn and B. Radunovic. Using Transmit-only Sensors to Reduce Deployment Cost of Wireless Sensor Networks. In *Proceedings of the IEEE Conference on Computer Communications (INFOCOM)*, 2008.

[7] M. Buettner and D. Wetherall. A Software Radio-based UHF RFID Reader for PHY/MAC Experimentation. In *Proceedings of the IEEE Conference on RFID (IEEE RFID)*, 2011.

[8] C. Gomez and J. Paradells. Wireless Home Automation Networks: A Survey of Architectures and Technologies. *IEEE Communications Magazine*, 48:92–101, 2010.

[9] K. Han and K. Huang. Wirelessly Powered Backscatter Communication Networks: Modeling, Coverage, and Capacity. *IEEE Transactions on Wireless Communications*, 16:2548–2561, 2017.

[10] P. Hu, P. Zhang, and D. Ganesan. Laissez-Faire: Fully Asymmetric Backscatter Communication. *Computer Communication Review (SIGCOMM)*, 45:255–267, 2015.

[11] H. Keong, K. Thotahewa, and M. Yuce. Transmit-only Ultra Wide Band Body Sensors and Collision Analysis. *IEEE Sensors Journal*, 13:1949–1958, 2013.

[12] A. Köpke, M. Swigulski, K. Wessel, D. Willkomm, P. T. K. Haneveld, T. E. V. Parker, O. W. Visser, H. S. Lichte, and S. Valentin. Simulating Wireless and Mobile Networks in OMNeT++: The MiXiM Vision. In *Proceedings of the International Conference on Simulation Tools and Techniques for Communications, Networks and Systems (SIMUTools)*, 2008.

[13] M. T. Lazarescu. Design of a WSN Platform for Long-Term Environmental Monitoring for IoT Applications. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, 3:45–54, 2013.

[14] Q. Lin, S. Mohan, and M. A. Weitnauer. Interference-insensitive Synchronization Scheme of MSK for Transmit-only Wireless Sensor Network. In *Proceedings of the IEEE International Conference on Communications (ICC)*, 2016.

[15] P. Parsch, A. Masrur, and W. Hardt. Designing Reliable Home-Automation Networks based on Unidirectional Nodes. In *Proceedings of the IEEE International Symposium on Industrial Embedded Systems (SIES)*, 2014.

[16] B. Radunovic, H. L. Truong, and M. Weisenhorn. Receiver Architectures for UWB-Based Transmit-Only Sensor Networks. In *Proceedings of the IEEE International Conference on Ultra-Wideband (ICU)*, 2005.

[17] G. Santagati, T. Melodia, L. Galluccio, and S. Palazzo. Medium Access Control and Rate Adaptation for Ultrasonic Intrabody Sensor Networks. *IEEE/ACM Transactions on Networking*, 23:1121–1134, 2015.

[18] A. Varga. The OMNeT++ Discrete Event Simulation System. In *Proceedings of the European Simulation Multiconference (ESM)*, 2001.

[19] M. Weisenhorn and W. Hirt. Uncoordinated Rate-Division Multiple-Access Scheme for Pulsed UWB Signals. *IEEE Transactions on Vehicular Technology*, 54:1646–1662, 2005.

[20] C. Yi, L. Wang, and Y. Li. Energy Efficient Transmission Approach for WBAN Based on Threshold Distance. *IEEE Sensors Journal*, 15:5133–5141, 2015.

[21] J. Zhao, C. Qiao, R. S. Sudhaakar, and S. Yoon. Improve Efficiency and Reliability in Single-Hop WSNs with Transmit-Only Nodes. *IEEE Transactions on Parallel and Distributed Systems*, 24:520–534, 2013.

Appendix A. Proof of Theorem 1

According to Lemma 3, in order that at least one packet of a node i reaches its destination in the worst case, it needs to send $k_i = n$ packets under the delayed-activation scheme. Taking a node j into consideration with $i \neq j$, Lemma 3 assumes that suitable inter-packet times p_i and p_j can be found such that node j cannot interfere with more than one packet within any node i's sequence of packets.

Now, for p_i and p_j to comply with Lemma 3, if the first packet of a node j's sequence intercepts a packet of node i, none of the subsequent node j's packets should affect another packet of the same node i's sequence, independent of which node i's packet has been first interfered by node j.

Let us assume that node i starts sending its sequence of n packets at time t_0 with a constant inter-packet time p_i. Clearly, the n-th packet of node i is sent at time $t_0 + (n-1) \times p_i$. Let us further assume that a packet of node j is also sent at time t_0 interfering with the first packet of node i. In addition, let us assume that node j continuously sends packets in $[t_0, t_0 + (n-1) \times p_i]$ following our delayed-activation scheme, i.e., node j's packets are separated by an integer multiple of p_j.

In order that the n-th packet of node i is not affected by node j, the remainder of $\frac{(n-1)\times p_i}{p_j}$ must allow for enough *space* to send a node j's packet before the n-th packet of node i starts being sent, i.e., before $t_0 + (n-1) \times p_i$: $\mathrm{mod}\left(\frac{(n-1)\times p_i}{p_j}\right) \geq l_j$.

On the other hand, a packet of node j can still affect the first packet of node i, if this is sent at time $t_0 - l_j + \varepsilon$ or $t_0 + l_i - \varepsilon$ where ε is an infinitesimally small amount of time. This is because there will be some amount of packet overlapping (given by ε) between the corresponding packets of node i and j. As a result, the remainder of $\frac{(n-1)\times p_i}{p_j}$ should allow for enough space to send a node j's packet considering all possible initial overlapping between the first packet of node i and the packet of node j: $\mathrm{mod}\left(\frac{(n-1)\times p_i}{p_j}\right) \geq l_i + l_j$.

In a similar manner, for the $(n-1)$-th packet of node i not to be affected by node j, the following has to hold: $\mathrm{mod}\left(\frac{(n-2)\times p_i}{p_j}\right) \geq l_i + l_j$. For any two nodes i and j where $1 \leq i \leq n$, $1 \leq j \leq n$, and $i \neq j$, this translates in that $\mathrm{mod}\left(\frac{k_i \times p_i}{p_j}\right) \geq l_i + l_j$ has to hold for $1 \leq k_i \leq n-1$. The theorem follows.

Note that, if $\left\lfloor \frac{(n-1)\times p_i}{p_j} \right\rfloor = 0$ holds, node i can only be interfered once by node j, which is already considered in Lemma 3. As a result, if $\left\lfloor \frac{(n-1)\times p_i}{p_j} \right\rfloor = 0$ holds for all i and j, Lemma 3 becomes necessary and sufficient for a reliable communication.

Appendix B. Deriving $q_i(x)$

This section covers the derivation of $q_i(x)$, i.e., the probability that exactly x out of k_i packets of node i are lost due to internal interference. To this end, we consider different exemplary combinations of n, k, i and x to stepwise show the assumptions made to derive (6) and (7).

Example 1. Let us first consider the simple case of each node sending just a single packet per sequence $k_i = 1\ \forall i$. Further, we set $n = 4$, $i = 4$ and $x = 1$. The probability of losing this packet can be calculated as:

$$\begin{aligned} q_4(1) = &[\sigma_1\bar{\sigma}_2\bar{\sigma}_3 + \sigma_2\bar{\sigma}_1\bar{\sigma}_3 + \sigma_3\bar{\sigma}_1\bar{\sigma}_2] \\ &+ [\sigma_1\sigma_2\bar{\sigma}_3 + \sigma_1\sigma_3\bar{\sigma}_2 + \sigma_2\sigma_3\bar{\sigma}_1] \\ &+ [\sigma_1\sigma_2\sigma_3]\,, \end{aligned} \tag{B.1}$$

where $\bar{\sigma}_j = 1 - \sigma_j$ is the probability of having no interference by node j with $1 \leq j \leq 3$.

In more detail, (B.1) is composed of 3 different parts, separated by square brackets. The first part contains the probabilities that node 4 has only 1 collision with any other node, for example, $\sigma_1\bar{\sigma}_2\bar{\sigma}_3$ means that there is a collision with node 1, but not with node 2 and 3. The second part considers double collisions, i.e., when interference is caused by 2 nodes simultaneously. And finally, the third part contains triple collisions. Note that node 4 must be transmitting for possible interference, hence, $q_4(x)$ does not depend on σ_4.

Calculating $q_i(x)$ is complex, since all combinations of collisions must be regarded. This complexity further increases for higher n, k and x, hence, in order to simplify calculations, let us consider:

$$q_4(1) \leq \sigma_1 + \sigma_2 + \sigma_3. \tag{B.2}$$

This equation only sums up probabilities σ_j of having a (single) collision with any node j. Multiple, simultaneous collisions, on the other hand, are included in the form of intersections between any of those σ_j — see a, b, c, d in Fig. B.17. However, by simply adding all σ_j, some of these intersections are included multiple times. In case of (B.2), a, b and c are added twice and d three times, therefore, we calculate a $q_i(x)$ that is always greater than the actual (correct) value. In general, these intersections are very small, since $\sigma_j << 1$ holds, and, hence, this error does not significantly affect results.

Clearly, for (B.2) to be valid, the following must hold:

$$\sum_{\forall i} \sigma_i \leq 1. \tag{B.3}$$

This is a logical assumption, since the *utilization* of the communication channel cannot exceed 100 %, otherwise reliable communication is not possible anymore. Note that periods found by Alg. 1 comply with (B.3).

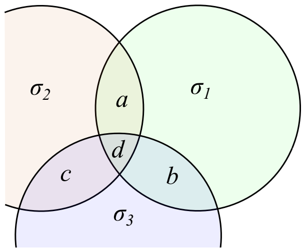

Figure B.17: A Venn diagram showing the relation between collision probabilities of three different nodes. The different sets represent the probabilities of having interference with one node, e.g., σ_1 means that there is a collision with node 1. The intersections a, b and c consider double collisions, i.e., an interference with two nodes simultaneously. Finally, d accounts for the case of a triple collision.

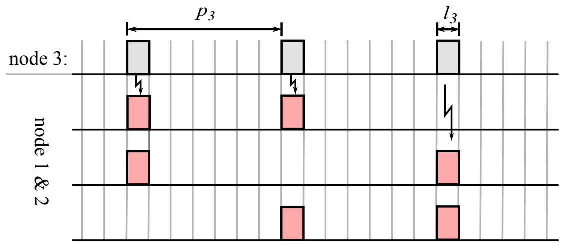

Figure B.18: Possible combinations of interference that lead to 2 packet collisions within a sequence of node 3: Either packets 1 and 2, packets 1 and 3 or packets 2 and 3 collide with packets of node 1 and 2. Further, the order in which nodes collide, e.g., node 1 first and node 2 second or vice versa, must be regarded as well, leading to 6 possible combinations in total.

Example 2. Next, we consider the case of two packets being sent per sequence, i.e., $k_i = 2\ \forall i$, $n = 4$ and $x = 1$. This yields the following expressions:

$$\begin{aligned} q_4(1) &\leq [\sigma_1 + \sigma_2 + \sigma_3] + [\bar{\sigma}_1\sigma_1 + \bar{\sigma}_2\sigma_2 + \bar{\sigma}_3\sigma_3]\,, \\ &\leq \sigma_1(1+\bar{\sigma}_1) + \sigma_2(1+\bar{\sigma}_2) + \sigma_3(1+\bar{\sigma}_3). \end{aligned} \tag{B.4}$$

The terms in the square brackets in (B.4) describe the collision probabilities for each packet within node 4's sequence. For example, σ_1 in the first term describes the probability that node 4's first packet collides with a packet of node 1. If that happens, the probability of the second packet interfering again with node 1 is zero ($\bar{\sigma}_1 = 1$). In case of the second term, however, we have to account for all previously sent packets: For example, $\bar{\sigma}_1\sigma_1$ in the second term describes the chance that node 1 does not interfere with the first, but with the second packet.

Again, we can simplify the calculation of $q_i(x)$:

$$q_4(1) \leq 2\sigma_1 + 2\sigma_2 + 2\sigma_3. \tag{B.5}$$

Since we know the ratios of packet lengths to period times are very small, i.e., $\sigma_i \ll 1$, we can approximate the probability of missing a packet $\bar{\sigma}_i = (1 - \sigma_i) \approx 1$. By doing so, we calculate a greater and more pessimistic $q_i(x)$, however, the error is again very small. In order for (B.5) to be valid, the following must hold:

$$\sum_{\forall i} k_i\sigma_i \leq 1. \tag{B.6}$$

This is a necessary condition in order that the simplification by (B.5) does not result in a probability that is greater than one. Since periods by Alg. 1 increase with k and n, (B.6) usually holds making the use of the simplified (B.5) possible.

Example 3. Let us consider the effects of changing the number of packets x that are allowed to be lost. To this end, we set our network parameters to $x = 2$, $n = 3$, $i = 3$ and $k_i = 3\ \forall i$.

Taking the simplifications of (B.2) and (B.5) into account, the probability of losing 2 packets within a sequence can be upper bounded by $\sigma_i\sigma_j$ with $j \neq i$. In other words, this is the probability of losing a first *and* a second packet. As illustrated in Fig. B.18, there are 6 possible combinations of losing 2 out of 3 packets, which can be computed using the binomial coefficient. We have $\binom{3}{2} = 3$ possibilities of being first interfered by node 1 and then by node 2 and another $\binom{3}{2} = 3$ in the reverse case. This results in:

$$q_3(2) \leq \binom{3}{2}(\sigma_1\sigma_2 + \sigma_2\sigma_1) = 6\,\sigma_1\sigma_2. \tag{B.7}$$

Let us now generalize $q_i(x)$ for an arbitrary number of packets sent k_i, packets lost x and number of nodes n. This results in (6) as stated in Section 6.2:

$$q_i(x) \leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j\neq i}^{n} \sigma_j \left(\sum_{l=1;\ l\neq i,j}^{n} \sigma_l \cdots \right)\right), \tag{B.8}$$

where $1 \leq x \leq (n-1)$ and $q_i(n) = 0\ \forall i$. The number of summations equals x. Again, the first part is the binomial coefficient and accounts for the different combinations of packets that collide within the sequence. The remaining part considers the combinations of different nodes that can cause collisions within the sequence. By assuming $\sigma_i = \sigma\ \forall i$, (B.8) can be further simplified to (7):

$$\begin{aligned} q_i(x) &\leq \binom{k_i}{x} \cdot \left(\sum_{j=1;\ j\neq i}^{n} \sigma \left(\sum_{l=1;\ l\neq i,j}^{n} \sigma \cdots \right)\right), \\ &\leq \binom{k_i}{x} \cdot \Big((n-1)\sigma\Big((n-2)\sigma \cdots\Big)\Big), \\ &\leq \binom{k_i}{x} \cdot \left(\prod_{j=1}^{x} (n-j) \right) \cdot \sigma^x. \end{aligned} \tag{B.9}$$

This equation allows to calculate the transmission reliability for each node i , if it sends a reduced number of $k_i \leq n$ packets. This allows us to adjust the packet numbers individually for each node to save energy whenever data loss can be tolerated by the application.