Accounting for Reliability in Unacknowledged Time-Constrained WSNs

PHILIP PARSCH and ALEJANDRO MASRUR, TU Chemnitz, Germany

Wireless sensor networks (WSNs) typically consist of nodes that collect and transmit data periodically. In this context, we are concerned with unacknowledged communication, i.e., where data packets are not confirmed upon successful reception. This allows reducing traffic on the communication channel — neither acknowledgments nor retransmissions are sent — and results in less overhead and less energy consumption, which are meaningful goals in the era of Internet of Things (IoT). On the other hand, packets can be lost and, hence, we do not know how long it takes to convey data from one node to another, which hinders any form of real-time operation and/or quality of service. To overcome this problem, we propose a medium access control (MAC) protocol, which consists in transmitting each packet at a random instant, but within a specified time interval from the last transmission. In contrast to existing approaches from the literature, the proposed MAC can be configured to meet reliability requirements — given by the probability that at least one data packet reaches its destination within a specified deadline — in the absence of acknowledgments. We illustrate this and other benefits of the proposed approach based on detailed OMNeT++ simulations.

CCS Concepts: • **Networks** → **Network protocol design**; *Network reliability*; • **Computer systems organization** → *Sensor networks*; Fault-tolerant network topologies;

Additional Key Words and Phrases: Wireless sensor networks, media access control, unacknowledged communication, reliability, energy efficiency

ACM Reference Format:
Philip Parsch and Alejandro Masrur. 2019. Accounting for Reliability in Unacknowledged Time-Constrained WSNs. *ACM Transactions on Cyber-Physical Systems* 3, 3, Article 26 (August 2019), 28 pages. https://doi.org/10.1145/3241746

1 INTRODUCTION

With the advent of IoT, an increasing number of devices start exchanging information in a variety of new applications. This puts emphasis on wireless communication and, especially, on MAC protocols that need to transport data in an efficient and reliable manner.

For this purpose, MAC protocols usually rely on acknowledgments (ACKs) that are sent for each data packet upon successful reception. This, however, also leads to several disadvantages. On the one hand, they do not increase the probability of a successful transmission, but this is rather achieved by retransmitting packets or taking provisions for future transmissions [14]. On the contrary, ACKs decrease reliability by producing additional traffic, i.e, increasing the probability of collision and data loss.

On the other hand, sending ACKs requires nodes to switch their radio transceivers twice: The receiving node has to switch from receive to transmit mode after receiving a packet to send an ACK and then switch back to receive mode to listen for further data. Similarly, if nodes perform carrier sensing, they have to switch from receive to transmit mode to send their data and then switch back to receive mode to listen for an ACK [13]. In this time, the channel is blocked and no further data can be sent, at least not to the same receiving node. For example, in the case of the CC2545 [19]

Authors' address: Philip Parsch; Alejandro Masrur, TU Chemnitz, Department of Computer Science, Chemnitz, 09111, Germany.

transceiver, each mode switch requires 130 μs — note that in this time a 16 byte-message can be sent at 2 Mbit/s. Together with the duration of the ACK itself, this is a relatively long time in which the channel is blocked and no other transmissions can take place.

In addition, since data packets are sent periodically in a WSN, losses can be tolerated to some extent [22]. As a consequence, ACKs become more an overhead than really necessary, which argues for *unacknowledged* communication protocols. For example, in wireless body area networks, sensors transmit heart rate, running speed, etc. to a health monitoring application, or in wireless home-automation networks, sensors report temperature/humidity data to a central processing unit, among others. Loosing some data packets in this context does not negatively impact the system's performance as long as a specified *age* is not exceeded.

Clearly, one can adapt existing MAC protocols for handling data transfers in unacknowledged WSNs. For example, TDMA is known to have a high throughput — the maximum channel utilization can theoretically reach 100% — and CSMA offers good flexibility and energy efficiency. However, TDMA requires synchronization and lacks flexibility, for example, when nodes enter or leave the network, which is necessary in the context of IoT. CSMA has the disadvantage of low efficiency at high loads. Similarly, other MAC protocols can be used, but these rarely allow assessing their worst-case behavior, which makes it difficult to provide any guarantees on communication reliability and are rather more suitable for a best-effort operation.

1.1 Contributions

In this paper, we propose a MAC protocol for unacknowledged, time-constrained WSNs. In particular, our MAC consists in sending each data packet at a random instant within a time interval $[t_{min}, t_{max}]$ from the last transmission. By adjusting t_{min} and t_{max}, we can influence the probability of a successful transmission and are, therefore, able to provide a guarantee on reliability, i.e., that at least one data packet reaches its destination within a specified deadline.

In the first half of the paper, we consider the case where all nodes have the same deadline and packet length, and neglect practical factors, such as clock drift and external interference. This allows us to derive a basic model of the proposed MAC and analyze its behavior in a comprehensible manner.

In the second half of the paper, this model is extended to consider arbitrary nodes types with individual deadlines and packet lengths. This has the advantage that (i) longer deadlines allow decompressing packets along the time line and (ii) smaller packets reduce interference on the communication channel. As a result, being able to model arbitrary deadlines and packet sizes allows computing the probability of packet loss with higher accuracy and, hence, increasing the achievable reliability in an unacknowledged WSN.

In addition, we consider the effect of practical factors such as external interference, and present an algorithm (of heuristic nature) to find an optimum configuration of our MAC protocol such that reliability is maximized for a given WSN.

1.2 Structure of the Paper

The rest of this paper is structured as follows. Related work is discussed in Section 2. Section 3 explains our system model and Section 4 introduces the proposed MAC. Section 5 introduces a use case consisting of an intelligent assembly line in the context of Industry 4.0, which is used for evaluating the proposed approach. Next, Section 6 extends our analysis to consider clock drift and external interference as well as arbitrary packet sizes and deadlines. Section 7 then discusses a simulation-based evaluation using OMNeT++ and Section 8 concludes the paper.

2 RELATED WORK

There are many existing MAC protocols in the literature that are concerned with making WSNs more reliable, delay-bounded, and energy-efficient. In the following, we briefly summarize those that are applicable to networks with recurring data traffic.

A large number of WSNs are based on the 802.15.4 standard [1], which defines two CSMA-based protocols: a hybrid superframe MAC (in slotted, beacon-enabled mode), and classic CSMA (in unslotted mode). While the first one is designed for applications that require low latencies and a guaranteed bandwidth, the second one offers low complexity. Both protocols can be adapted to a given network by adjusting parameters, such as backoff or retransmissions numbers. However, finding the optimal setting is difficult due to high complexity [9], while without adaption, i.e., with default parameters, performance is often low [2]. To solve this problem, numerous modifications have been proposed, for example, dynamic, learning-based adaption of parameters [4] or a modified backoff scheme [11]. These typically improve the average performance, however, also lead to increased complexity and a lower worst-case performance. In contrast, the proposed MAC of this paper offers a low-complex framework that allows easy network adaption, while maintaining a worst-case reliability and a bounded delay.

An unacknowledged MAC for slotted CSMA (as per 802.15.4) has been presented in [23]. Here, nodes use carrier sensing to locally measure busy channel probabilities and to estimate reliability. These values are then used to tune MAC parameters, in particular, contention window sizes are changed to prolong or shorten backoff times. This improves performance and allows reaching the same reliability as acknowledged CSMA, while in return having a lower energy consumption by missing ACKs. However, this protocol is based on a heuristic algorithm to estimate average reliability and cannot provide any guarantees on its worst-case behavior.

In [22], Zhang et al. presented an unacknowledged protocol taking advantage of capture effect, i.e., it assumes that there is a certain chance (i.e., a capture rate) that the strongest packet can be recovered in the event of a collision. To maximize the capture rate, the authors present policies to determine the optimal number of receivers in a network as well as their (physical) placement. Results show that packet loss can be reduced by 35 % when using three correctly-placed receivers instead of only one. However, this approach has a number of drawbacks. For example, it is not well suited for mobile networks, as receivers would have to be relocated continuously for good performance. In addition, capture effect causes poor fairness [6], meaning that nodes located far away from the receiver can potentially *starve*, because closer nodes always have higher signal strengths. Lastly, in contrast to our proposed approach, this protocol is unable to provide any kind of worst-case guarantees due to the highly stochastic nature of capture effect [7].

In [5], an unacknowledged CSMA mechanism was presented that uses a modified RSSI (received signal strength indication) to obtain additional information about interfering sources and decide whether to transmit or to back off instead. This increases the average performance compared to classic CSMA [5]; however, no analysis framework is provided for assessing the worst-case performance.

Another approach, presented in [24], consists of two node types: low-cost, unacknowledged nodes with low priority (LP-nodes), and more elaborate, high-priority nodes (HP-nodes). While LP-nodes send sequences of packets with constant inter-packet times, HP-nodes are scheduled to transmit in the vacant time slots in between LP transmissions. This way, LP-nodes provide good efficiency, but need to tolerate data loss, whereas HP-nodes provide high reliability, but incur high cost. However, again no framework is provided to analyze the worst-case performance either.

An asynchronous MAC protocol alternative to CSMA was presented in [13]. Thereby, nodes transmit packets within random time intervals of configurable length. This allows tuning reliability

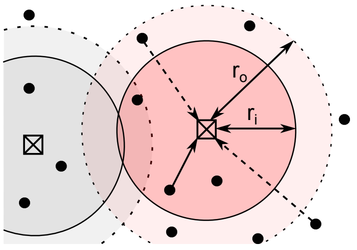


Fig. 1. Example of a WSN with data sources (solid circles) and sinks (checked boxes): r_i represents the range within which a sink collects packets, while r_o indicates the range in which sources can interfere with each other.

and energy consumption so as to meet desired goals [13]. Although this MAC results in a good performance, it also relies on acknowledgments and is less suited for high-traffic settings with recurrent data.

Further, techniques for unidirectional nodes are, in principle, applicable to unacknowledged networks.[1] These consist in each node sending a sequence of redundant data packets with carefully chosen inter-packet times to guarantee that at least one packet arrives in the worst case [3][12]. In [3], ILP (integer linear programming) is used to select inter-packet times, whereas these are selected randomly in [12].

In this paper, to account for reliability in unacknowledged WSNs, we make use of the fact that (i) most WSNs perform periodic transmissions and (ii) applications tolerate some amount of data loss. Similar to [12] and [13], packets are sent within random time intervals, based on which we derive a guarantee on the probability that at least one packet reaches its destination within a specified deadline, i.e., the maximum age of data. In contrast to the aforementioned approaches [3][12][13], we consider arbitrary deadlines and packet sizes taking practical factors such as external interference into account.

3 SYSTEM MODEL AND ASSUMPTIONS

We consider a WSN where a set of nodes (i.e., data sources) are activated periodically and send updates to one or more (data) sinks. To ensure that a maximum *age* of the data is not exceeded,[2] at least one data packet must be received before an upper time boundary or deadline d_{max} has elapsed, measured from the node's activation time. Since individual transmissions can be corrupted, multiple data packets need to be sent within d_{max}. We denote this number of packets by $k \in \mathbb{N}_{>0}$ and refer to it by (packet) sequences. Note that, in contrast to [3][12][13], packets within a sequence are not redundant copies, but rather contain updated data. This can be regarded as oversampling with an increase in quality, if all packets of the sequence are received, and minimum acceptable functionality, if only one packet is received within d_{max}.

Each node i waits a random time $t_{ix} \in \mathbb{R}_{>0}$ before sending any packet x — including the first packet of a sequence.[3] Here $1 \leq i \leq n$ and $1 \leq x \leq k$ hold and $n \in \mathbb{N}_{>0}$ denotes the number of transmitting nodes in the system. This t_{ix} is referred to as (random) inter-packet time and

[1] In contrast to unidirectional networks, an unacknowledged network essentially consists of bidirectional nodes, which has the advantage of facilitating remote reconfiguration.

[2] Many applications require timely data delivery, e.g., control loops or whenever a quick reaction is needed to some event. For example, in a home automation setting, a wireless light switch needs to convey its information within around half a second and temperature data is only valid for a few minutes. Longer delays are unacceptable and lead to a quality loss.

[3] This design decision simplifies our analysis in a considerable manner, while it does not affect the functionality of the network.

is uniformly distributed in the interval $[t_{min}, t_{max}]$ where $t_{min}, t_{max} \in \mathbb{R}_{>0}$ are system design parameters and common to all nodes. Similarly, we assume that the packet length $l_{max} \in \mathbb{R}_{>0}$, i.e., the time to transmit a given amount of data at a specific speed, is the same for all nodes in the system.

Further, we assume that each sink constantly monitors the communication channel and, hence, no special measures have to be taken before sending data. For example, no wake-up message needs to be sent to activate a sink, etc. In many applications, this assumptions does not pose any additional restrictions, since sinks are usually attached to an actuator with a relatively big power supply. In other cases, the presented method can be extended to account for sleep/wake-cycling, e.g., by adjusting the packet length or increasing the number of packets sent.

Packets can be lost as a consequence of interference at the communication channel. For ease of exposition, we first assume that interference originates from simultaneous transmissions by neighboring nodes with overlapping space and frequency ranges (see r_o in Fig. 1). Later in Section 6, we extend our analysis to consider external interference. In addition, in Section 6, we extend our model to consider arbitrary packet sizes l_i and deadlines d_i and allow for $t_{min,i}$ and $t_{max,i}$, i.e., bounds that depend on the individual node i. This allows modeling packet loss more precisely and, hence, increases the achievable performance. Note that, similar to other approaches from the literature [5][13][23], we (pessimistically) assume that no packets can be recovered in case of collisions, i.e., there is no capture effect, since this is highly stochastic depending on signal strength, node placement and on which parts of the packets overlap [6][7]. For this reason, capture effect leads to erratic and non-reproducible results, making it impossible to provide any worst-case guarantees as aimed in this paper.

4 PROPOSED MAC

In this section, we obtain suitable values for the parameters introduced before in Section 3. In particular, we will select values for t_{min}, t_{max} and k that allow guaranteeing a specific worst-case reliability p for communication between nodes in our network.

Definition: We define *reliability* of a time-constrained wireless network as the probability that, in the worst case, at least one of a sequence of k packets of any node i reaches its destination within a specified deadline.

To compute the above probability, we need to consider the worst-case transmission conditions: (i) all n nodes in the network are sending packets, and (ii) every time a packet is sent by a node, there exists a maximum fraction of the interval $[t_{min}, t_{max}]$, denoted by Δ_{coll}, for which any selected value of t_{ix} leads to collisions. While condition (i) is straightforward, condition (ii) requires more analysis.

Recall that each node i in the network uniformly selects inter-packet times t_{ix} in $[t_{min}, t_{max}]$. Let us first consider the case where t_{min} is set such that there can be at most one packet of each node in an interval of length $t_{max} - t_{min}$. That is, the value of t_{min} has to fulfill the following condition:

$$\begin{aligned} t_{min} &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max}}{2}. \end{aligned}$$

Given this case, let us analyze the example of Fig. 2 for four nodes. Node 1 is activated at time t_0 and sends packets at $t_0 + t_{1,x}$ with $1 \leq x \leq k$. The probability of losing a packet is given by the probability of choosing a $t_{1,x}$ (from $[t_{min}, t_{max}]$) that leads to a collision with a packet of any of the

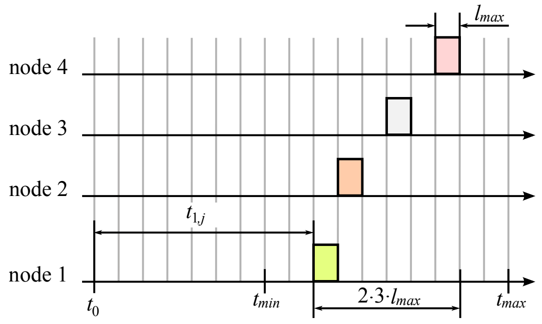


Fig. 2. Computing the worst-case probability of packet loss: This results from the ratio between the maximum number of inter-packet times that potentially yield a collision to the total number of possible inter-packet times.

other nodes. This probability is maximum, when the fraction of $[t_{min}, t_{max}]$ leading to potential collisions, i.e., Δ_{coll}, is the greatest possible.

In Fig. 2, there is a period of time in $[t_{min}, t_{max}]$ equal to $2l_{max}$, for which any $t_{1,x}$ leads to a collision with one of the other three nodes. This originates from the fact that even the smallest overlapping yields packet loss, i.e., a packet sent within $(t_{1,x} - l_{max}, t_{1,x} + l_{max}]$ will collide with packet x of node 1. In the worst case, the packets of the other three nodes are separated by at least a time equal to l_{max}. As a result, there will be three time intervals equal to $2l_{max}$ that lead to collisions. The sum of these time intervals results in the maximum value of Δ_{coll}, i.e., $2 \cdot 3 \cdot l_{max}$ in the example of Fig. 2. For n nodes, this leads to the following expression:

$$\Delta_{coll} = 2(n-1)l_{max}.$$

To generalize, let us remove the previous restriction allowing t_{min} to have smaller values than $\frac{t_{max}}{2}$. Now, each node can send more than one packet within an interval of length $t_{max} - t_{min}$, for example, if it (randomly) selects t_{min} multiple times. We denote this number of packets by $m \in \mathbb{N}_{>0}$ for which t_{min} has to fulfill the following condition:

$$\begin{aligned} m \cdot t_{min} &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max}}{m+1}. \end{aligned} \tag{1}$$

Since there can be m packets of each node in an interval of length $t_{max} - t_{min}$, the generalized expression of Δ_{coll} is given by:

$$\Delta_{coll} = 2m(n-1)l_{max}. \tag{2}$$

As a consequence, we can compute the maximum possible probability of packet loss every time a packet is sent using the proposed MAC scheme by the ratio between Δ_{coll} and $t_{max} - t_{min}$:

$$q = \frac{2m(n-1)l_{max}}{t_{max} - t_{min}}. \tag{3}$$

Clearly, the probability of a successful packet transmission in the worst case is given by $1 - q$. Note that, for (3) to be valid the following condition must be satisfied (i.e., $q \leq 1$ must hold):

$$\begin{aligned} 2m(n-1)l_{max} &\leq t_{max} - t_{min}, \\ t_{min} &\leq t_{max} - 2m(n-1) \cdot l_{max}. \end{aligned} \tag{4}$$

In addition, since m, n, l_{max}, t_{max} and t_{min} are system parameters (i.e., common to all nodes in the WSN), q is independent of the node and of the packet being sent. As a result, we can model

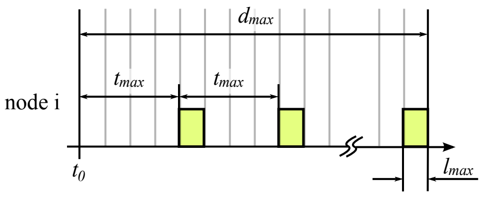


Fig. 3. Relation between d_{max} and t_{max}: In the worst case, t_{max} should fit k (number of packets sent) times in a time interval of length $d_{max} - l_{max}$.

the transmission of packets in the network using a binomial distribution. This way, we compute the probability p that, in the worst case, at least one packet out of a sequence of k reaches its destination for any node in the network.

To compute p, we need to consider all possible combinations, i.e., only one packet arrives, two packets arrive, etc., which is a cumbersome procedure. It is much easier to compute the sequence loss rate $1 - p$, i.e., the probability that, in the worst case, no packet of a sequence of k reaches its destination. This is the probability that k consecutive packets be lost and can be computed by the well-known equation $\binom{k}{z}q^z(1-q)^{(k-z)}$ where $\binom{k}{z} = \frac{k!}{z!(k-z)!}$ is the binomial coefficient. Replacing q as per (3) and selecting $z = k$, i.e., k out of k packets are lost, the binomial coefficient becomes 1 and we obtain:

$$1 - p = \left(\frac{2m(n-1)l_{max}}{t_{max} - t_{min}}\right)^k. \tag{5}$$

In order that p corresponds to our definition of reliability, we need to make sure that nodes always send k packets within the specified deadline d_{max}. Towards this, recall again that every node i waits a random time t_{ix} chosen from $[t_{min}, t_{max}]$ before sending any packet. In the worst case, node i will have to wait t_{max} before sending each of its k packets as illustrated in Fig. 3. To guarantee that each node i sends its k packets within d_{max}, the following must hold:

$$t_{max} \leq \frac{d_{max} - l_{max}}{k}. \tag{6}$$

Given a value of t_{max} as per (6), we can reshape (5) to compute the value of t_{min} that satisfies a desired reliability p for the whole WSN:

$$t_{min} \leq t_{max} - \frac{2m(n-1)l_{max}}{\sqrt[k]{1-p}}. \tag{7}$$

Note from (5) that a $p = 1$, i.e., 100% reliability, can only be achieved for $n = 1$, i.e., for only one node in the network, independent of all other parameters. For $n > 1$, if p tends to 1, t_{min} tends to minus infinity as per (7). In other words, 100% reliability as with [3] cannot be achieved with the proposed approach. However, our scheme allows for a reliability that is acceptably close to 100%, while considerably reducing the number of packets sent and, hence, making better use of energy.

In the next section, we introduce a use case based on an intelligent assembly line in the context of Industry 4.0. This provides us with exemplary data such as deadlines, packets sizes, etc., which we use in later sections to analyze and simulate the proposed MAC.

5 USE CASE - INTELLIGENT ASSEMBLY LINE

Modern assembly lines are expected to increasingly rely on mobile robots, as these can perform tasks quickly and efficiently. These robots are usually limited to a set of pre-programmed tasks and lack autonomy. As a result, they still require human workers for manipulating fragile parts or taking common sense decisions leading to a shared physical space.

This, of course, requires a high degree of safety, since weighty robots can potentially harm humans and/or other robots. For this reason, we assume that humans wear sensors to locate and track their positions. This data is periodically broadcast and received by a central control station that computes mobile robots' trajectories. If robots comes close to humans or other robots, i.e., at a *safety distance*, these are slowed down or even stopped to avoid accidents. In the case of communication loss, fail-save mechanisms need to be specified to guarantee safety. For example, robots may perform an emergency stop, if no updates from the central control station are received within a specified period of time, etc.

5.1 Timings and Data Structures

Workers and mobile robots, i.e., transmitting nodes in our network, periodically broadcast their current position and speed with a total of 10 bytes. For this they are equipped with a wireless transceiver, which handles transmissions and reception of data packets. We assume a basic frame format of 8 bytes preamble, 2 bytes Start Frame Delimiter (SFD) and 2 bytes Cyclic Redundancy Check (CRC). Assuming a transmission speed of 2 Mbit/s, the resulting length of a single data packet is:

$$\begin{aligned} l_{max} &= T_{preamble} + T_{SFD} + T_{payload} + T_{CRC} \\ &= 32\,\mu s + 8\,\mu s + 40\,\mu s + 8\,\mu s = 88\,\mu s. \end{aligned}$$

In addition, there will be a delay at the central control station, which is receiving all data packets. This delay, denoted by t_{host}, accounts for computation of trajectories, collision avoidance algorithms, and the reaction time of actuators and is estimated to be in the order of several tens of milliseconds. In the following, we assume $t_{host} \leq 80\,ms$.

5.2 Physical World

We consider moving robots with a maximum speed of $10\,km/h$, i.e., around $3\,m/s$, and further assume that the walking speed of human workers is no more than $5\,km/h$, i.e., around $1.5\,m/s$. In addition, robots have a stopping distance of $0.5\,m$ (from maximum speed to zero) and their safety distance is fixed to a radius of $4\,m$.

In the worst-case, two robots move at full speed towards each other, which results in a relative speed of $6\,m/s$. Since we have to guarantee that both robots stop on time to avoid collisions, the central control station has to receive their packets within:

$$t_{WC} = \frac{4\,m - 0.5\,m}{6\,m/s} = 580\,ms,$$

from the time they enter each others safety distance. Subtracting t_{host} as a safety tolerance and to account for processing times, we obtain an upper bound on communication delay:

$$d_{max} = t_{WC} - t_{host} = 500\,ms.$$

This is the time by which data must be received at the central control station to avoid triggering fail-save mechanisms and cause an emergency stop. In the next section, we use this data to evaluate the proposed MAC.

5.3 Numerical Evaluation

In this section, we perform a detailed evaluation of the worst-case behavior of the proposed MAC. For this we use the parameters obtained before in the presented use case, i.e., a packet length

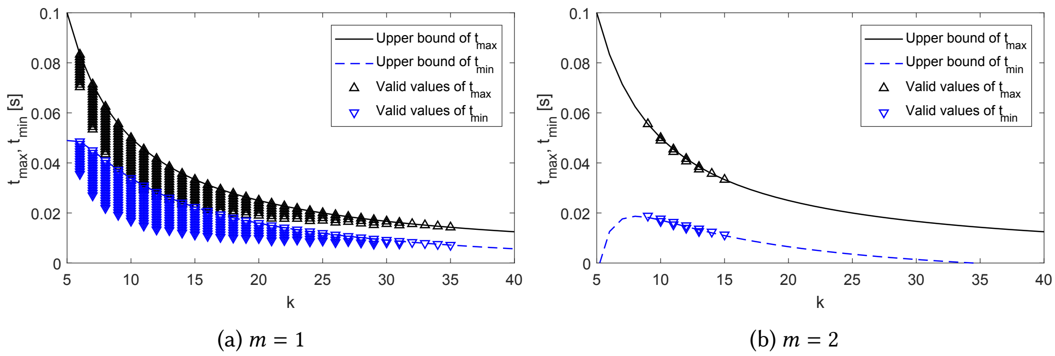


Fig. 4. Valid values of t_{max} and t_{min} for different m

$l_{max} = 88\ \mu s$ and a deadline $d_{max} = 500\ ms$. Further we set the number of nodes, i.e., workers and mobile robots, to $n = 30$ and assume a reliability of $p = 99.999\%$ (equal to a sequence loss rate $1 - p$ of 10^{-5}). For simplicity, we do not take external interference and clock drift into account now, as this is discussed later in Section 6. However, observations and conclusions remain valid.

In particular, we evaluate the performance of the proposed MAC by analyzing the maximum possible network size n_{max}, i.e., the number of nodes that can be safely accommodated in a network for the given parameters. This n_{max} results from delay, channel utilization and throughput of the MAC and often constitutes the limiting factor for many applications.

Selecting t_{max} and t_{min}

Given the parameters n, l_{max}, and d_{max}, we first need to determine the number of packets k that guarantees the desired reliability p. This implies finding valid values of t_{max} and t_{min} for a given k that not only satisfy (6) and (7), but also (1) and (4).

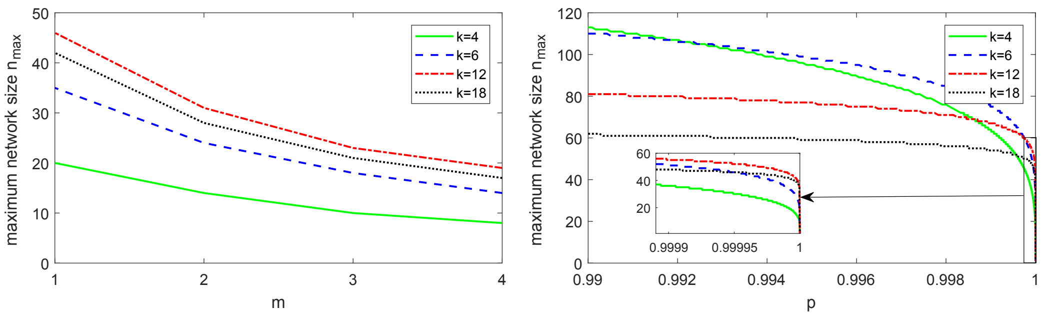
Fig. 4a and Fig. 4b show plots of (6) (solid line) and (7) (dashed line) for $p = 1 - 10^{-5}$ and different values of k. Upward- and downward-pointing triangles identify valid values of t_{max} and t_{min}. The value of m has been set to 1 in Fig. 4a and to 2 in Fig. 4b. Recall that this parameter determines the lower bound of t_{min} and, hence, the number of packets from the same node in an interval of length $t_{max} - t_{min}$.

From Fig. 4a and Fig. 4b, we can see that $p = 1 - 10^{-5}$ cannot be guaranteed for every k. In the case of $m = 1$ in Fig. 4a, the system is only feasible for k in $[6, 35]$. It should be noted that the number of valid values for t_{max} and t_{min} reaches its maximum for $k = 12$. For higher k this number decreases until having only one valid value for $k = 35$. That is, a higher k does not increase the reliability anymore, since the increasing interference between nodes starts to be the dominating effect from this point on.

Now, in the case of $m = 2$ in Fig. 4b, the system is only feasible for k in $[9, 15]$ and, in general, there are less valid values of t_{max} and t_{min}. This is analyzed next in detail.

Effect of m

Let us now study the effect of m on the maximum number of nodes n_{max} that can be reached for a specified p. Fig. 5a shows how n_{max} varies with m and different values of k. For $k = 6$, $p = 1 - 10^{-5}$ and $m = 1$, around 35 nodes are possible, which reduces to 14 for $m = 4$. A similar behavior can be observed for other k.



(a) Different values of m and k (b) Different values of p and k

Fig. 5. Maximum network size n_{max} for different m, p and k

In other words, m allows for more flexibility in selecting values of t_{min}. The greater m is chosen, the closer t_{min} may be to zero — see (1). As a result, the interval $[t_{min}, t_{max}]$ becomes longer decreasing the probability of packet collision — see (3). On the other hand, m also increases the number of packets in $[t_{min}, t_{max}]$ from the same node, which again increases the probability of packet collision — see again (3). In general, the second effect dominates such that a greater m negatively impacts the attainable size of the network. We therefore consider $m = 1$ and $t_{min} = \frac{t_{max}}{2}$ — minimizing the probability of packet collision for $m = 1$ — for the rest of the paper.

Reliability vs. number of nodes

Fig. 5b shows the dependency of the maximum possible number of nodes n_{max} on the desired reliability p for different values of k. We can observe that sending a small number of packets, i.e., $k = 4$, allows an acceptably big n_{max} for low values of p. With $k = 4$ and $p = 0.95$, it is possible to achieve $n_{max} = 170$, but this rapidly reduces for very high p, for example to $n_{max} = 20$ for $p = 1 - 10^{-5}$. For these high reliabilities, more packets are required to achieve an acceptably high n_{max}. For example, for $p = 1-10^{-5}$ we need at least $k = 6$ for achieving a network size of > 30 nodes. As mentioned previously, the only way of guaranteeing a reliability of 100 % with our scheme, i.e., $p = 1$, is with only one node, i.e., $n = 1$.

For a given reliability value, there exists an optimal k that maximizes n_{max}. In Fig. 5b, for example, $k = 4$ performs best for p up to $p \leq 1 - 10^{-2}$ %. Similarly, $k = 6$ is optimal for $p \leq 1 - 10^{-4}$, $k = 12$ for $p \leq 1 - 10^{-6}$ and $k = 18$ for $p > 1 - 10^{-6}$. Note that if multiple values of k satisfy a given p and n, it is meaningful to select the lowest k, clearly, leading to less energy consumption.

6 PRACTICAL FACTORS

In this section, we extend our proposed analysis to consider practical factors, such arbitrary packet sizes and deadlines for each node as well as external interference.

6.1 External Interference

Since nowadays a growing number of devices communicate wirelessly, we observe an increasing level of noise on the wireless medium. In this section, we analyze the effects of external interference on the proposed MAC, i.e., how noise from outside the network affects our system's performance.

To begin with, we first need a model of external interference to characterize its behavior. For this, we assume that the maximum *duty cycle* by external interference — denoted by σ — can be

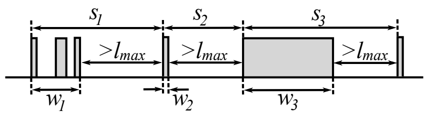


Fig. 6. An exemplary sequence of external interference pulses at the communication channel. In this case, the maximum possible duty cycle σ is equal to $\frac{w_3}{s_3}$. Note that external interference pulses that are separated by less than l_{max} are considered to be one single large pulse, since no packet can be sent in this short time interval. This is the case of the first three pulses in the example, which are merged into on single pulse of width w_1.

determined, i.e., the greatest possible ratio between pulse width w_i to inter-pulse separation s_i of external interference — see Fig. 6:

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

where $i \in \mathbb{N}_{>0}$ is an index identifying the particular pulse. This σ 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} 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 shown in Fig. 6.

This σ gives also the greatest probability of encountering an external interference pulse at the communication channel [21] — note that $\sigma \leq 1$ holds. This probability is clearly independent of q in (3), i.e., the probability of packet loss due to internal interference, since external interference is independent of any of the (internal) nodes in the network. As a result, when considering external interference, the probability of packet loss is given by:

$$\hat{q} = q + \sigma - q \cdot \sigma = q + (1 - q) \cdot \sigma, \tag{9}$$

i.e., the probability that a packet is lost at the communication channel either by internal or by external interference.

Note that we can still apply the binomial distribution, since $\hat{q}$ in (9) is independent of the node and the packet being sent. As a result, proceeding as for the case with no external interference, we obtain the probability that k consecutive packets are lost:

$$1 - p = (q + \sigma - q \cdot \sigma)^k, \tag{10}$$

and, hence, replacing q as per (3) we can solve for t_{min} such that p, the desired reliability requirement, is satisfied under external interference:

$$t_{min} \leq t_{max} - \frac{2m(n-1)(1-\sigma)l_{max}}{\sqrt[k]{1-p} - \sigma}. \tag{11}$$

If t_{min} becomes negative or greater than t_{max}, it will not be possible to fulfill the reliability requirement p for the given external interference σ. The other constraints on t_{min}, i.e., (1) — or (23) when considering clock drift — and (4) still have to be satisfied. The value of t_{max} is again given by (6) — or (24) when accounting for clock drift.

In summary, (6) and (7) can be used in the absence and (24) and (11) in the presence of external interference. However, dynamically adjusting t_{max} and t_{min} to the interference level is not possible, since unidirectional nodes cannot sense the channel and are therefore not able to detect any changes. The configuration must hence be performed in the deployment phase or manually changed later.

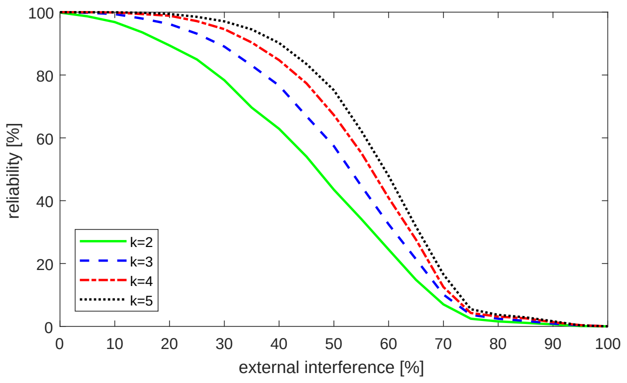


Fig. 7. Simulated reliability for different values of k and an increasing level of external interference

Effect of external interference. Let us now examine the effect of external interference on the transmission reliability of the proposed scheme by means of simulation. We again regard an exemplary network with $n = 30$ nodes and the same simulation parameters as in Section 7.

Fig. 7 shows the resulting reliability for different values of k and an increasing duty cycle of external interference pulses. We can see that an increasing noise level leads to a higher loss of packet sequences, i.e., a lower reliability. With 100% interference, i.e., the channel is fully blocked, data transmission is not possible anymore and the resulting reliability is zero. Note that curves bend at around 75 % external interference due to saturation effects. In other words, interference pulses are so long that successful transmission (for the given packet length) is hardly possible in the pauses between them.

As can be seen in Fig. 7, the proposed MAC protocol is generally robust against external interference. For example with $k = 2$ and a desired reliability of $p = 90\%$, up to 20 % of interference can be tolerated. For $k = 3$ this rises to 30 % and for $k \geq 5$ to over 40 %. In general, a greater k leads to better performance in case of external interference, because it is less likely to lose all k consecutive packets of a sequence. With $k = 1$, reliability in shows a linear behavior, whereas this is non-linear for $k > 1$ as per (10).

6.2 Arbitrary Packet Sizes and Deadlines

So far, the proposed technique considers the same packet size for all nodes, i.e., the maximum packet size l_{max}. This incurs pessimism when nodes' packet sizes l_i greatly differ from each other, i.e., some packet sizes are very small, some others very large. In addition, deadlines were assumed to be the same for all nodes (denoted by d_{max}), which further increases pessimism when real deadlines d_i greatly differ from each other. In this section, we remove these two restrictions.

Let us consider a node i with packet size l_i and a set of nodes j with packet sizes l_j that are sending packets simultaneously. The interval of time leading to collisions between node i and any node j is equal to $l_i + l_j$. Here again node i waits for a random time t_{ix} before sending a packet. However, t_{ix} is now uniformly distributed in $[t_{min,i}, t_{max,i}]$, where $t_{min,i}, t_{max,i} \in \mathbb{R}_{>0}$ are node-specific parameters. If every node j sends at most one packet in an interval of length $t_{max,i} - t_{min,i}$, we obtain the following:

$$\Delta_{coll,i} = (n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j.$$

And, if every node j sends m packets in $t_{max,i} - t_{min,i}$, we further obtain:

$$\Delta_{coll,i} = m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big), \tag{12}$$

where m has to fulfill the following condition for each node i and j in the network, i.e., at most m packets of node j can interfere with one packet of node i:

$$\begin{aligned} m \cdot t_{min,j} &\geq t_{max,i} - t_{min,i}, \\ t_{min,i} &\geq t_{max,i} - m \cdot t_{min,j}. \end{aligned} \tag{13}$$

As a consequence, we can compute node i's maximum possible probability of packet loss based on the proposed MAC scheme. This is given by the ratio between $\Delta_{coll,i}$ and $t_{max,i} - t_{min,i}$:

$$q_i = \frac{m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big)}{t_{max,i} - t_{min,i}}. \tag{14}$$

Clearly, the probability of a successful packet transmission by node i is $1 - q_i$ in the worst case. Note that, for (14) to be valid the following condition must be satisfied (i.e., $q_i \leq 1$ must hold):

$$t_{min,i} \quad \leq \quad t_{max,i} - m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big). \tag{15}$$

Note that q_i in (14) depends on node i; however, it is independent of the packet being sent by node i. As a consequence, we can again make use of binomial distribution to compute the probability p_i that at least one out of a sequence of k node i's packets reaches its destination in the worst case. Proceeding as before, we compute $1 - p_i$, i.e., the probability that none of the k node i's packets reaches its destination in the worst case:

$$1 - p_i = \left(\frac{m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big)}{t_{max,i} - t_{min,i}}\right)^k. \tag{16}$$

In order that p_i corresponds to our definition of reliability, we need to make sure that nodes always send k packets within their specified deadlines d_i:

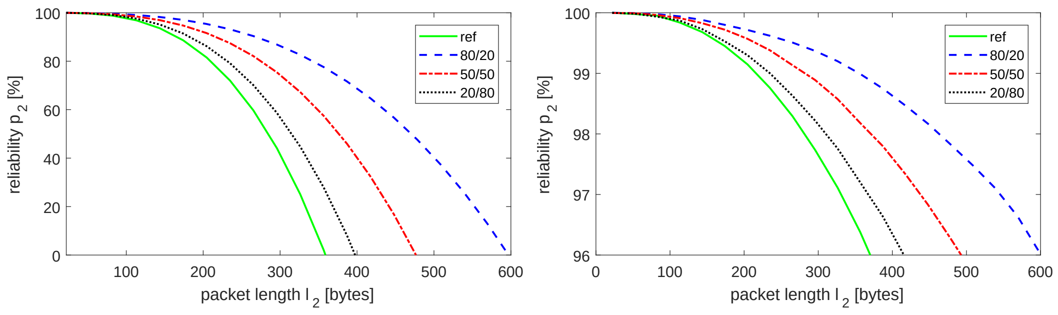
$$t_{max,i} \leq \frac{d_i - l_i}{k}, \tag{17}$$

and, finally, solving (16) for $t_{min,i}$, we have:

$$t_{min,i} \leq t_{max,i} - \frac{m\Big((n-1)l_i + \sum_{j=1;j\neq i}^{n} l_j\Big)}{\sqrt[k]{1-p_i}}. \tag{18}$$

In summary, the previous equations allow calculating safe values for $t_{min,i}$ and $t_{max,i}$ for each node i of a set of different nodes with arbitrary deadlines and packet sizes.

Effect of arbitrary packet sizes. We now analyze the effect of arbitrary packet sizes on reliability. For this purpose, let us again consider (15), which gives an upper bound on $t_{min,i}$. This bound depends on $t_{max,i}$ and $\Delta_{coll,i}$ as per (12). Now, by increasing the packet sizes l_i, the collision interval increases and, hence, $t_{min,i}$ decreases. However, $t_{min,i}$ is lower bounded by (13) and can therefore only decrease to a certain extent. Otherwise, the system stops being feasible.



(a) worst case (computed) (b) average (simulated)

Fig. 8. The maximum possible reliability p_2 for a system with two node types 1 and 2: $l_1 = 22$ bytes , $22 \leq l_2 \leq 600$ bytes. The different curves show p_2 for different n_1/n_2-ratios, i.e., *20/80* means 20 % type 1 and 80 % type 2 nodes with $n_1 + n_2 = n = 30$. The *ref* curve shows the behavior of the basic scheme from Section 4, which does not allow modeling different packet sizes for each node.

Reliability is then conditioned by the packet sizes l_i of all nodes as per (12) and by the length of the shortest $t_{min,j}$ in the system — as per (13). In summary, allowing for arbitrary packet sizes instead of assuming the longest possible l_{max} for every node reduces $\Delta_{coll,i}$ and, hence, increases reliability according to (16).

Let us now consider an exemplary system with two node types as displayed in Fig. 8: type 1 with short packets of $l_1 = 22$ bytes (88 μs as before in Sec. 5) and type 2 with varying packets of $22 \leq l_2 \leq 600$ bytes. For simplicity, the sub indexes 1 and 2 represent the *node type* 1 and 2 respectively and not directly the *node* 1 and 2. The remaining parameters were set to $n = 30$, $k = 3$, $d_1 = d_2 = 500\,ms$ and $m = 1$. The figure on the left (Fig. 8a) shows the computed worst-case reliabilities using the formulas mentioned above, whereas the figure on the right (Fig. 8b) shows the simulated average reliabilities — more details on the simulation can be found in Section 7.

As expected, an increasing l_2 leads to lower reliability, which, in the worst case in Fig. 8a, reaches $p_2 = 0$ for high values of l_2, i.e., when (18) does not hold and, hence, the system is not feasible for any valid p_2. We can further observe that a higher ratio of type 1 to type 2 nodes, i.e., when the number of type 2 nodes with their relatively big packets is low, allows for a higher reliability (of the remaining type 2 nodes in the system).

By taking arbitrary packet sizes into account, it is possible to reach higher reliabilities than for packets with the same size. In the latter case, the corresponding packet lengths are set to be equal to the longest possible packet in the system to guarantee the desired reliability (see Section 4), incurring pessimism as shown by the *ref* system. In the worst case, in Fig. 8a, this system achieves a reliability of 65 % for $l_2 = 256$ bytes, whereas the *80/20* system — 80 % of type 1 and 20 % of type 2 nodes — allows for 91% reliability. This is an improvement of 40 % over the *ref* system from Section 4. Even for small differences in packet sizes, for example $l_2 = 44$ bytes, reliability can be improved considerably. In this case, p_2 can be increased from 99.81 to 99.9 %, which corresponds to a 50 % reduction of the sequence loss rate $(1 - p)$. On average, in Fig. 8b, similar improvements can be observed. For example, for $l_2 = 256$ bytes, the *20/80* systems achieves a reliability of 99.5 % compared to 98.4 % of the *ref* system. This corresponds to a 70 % reduction of the sequence loss rate. Note that, as shown by experiments in Fig. 8, reliability is always higher on average than in the worst case validating the proposed approach, i.e., simulated behavior never falls below the worst case.

Extension to different m. So far, we have considered that any node j can send at most m packets within an interval of length $t_{max,i} - t_{min,i}$, where m is the same for all nodes. According to (13), for any node i, this means that the shortest $t_{min,j}$ must fit no more than m times in $t_{max,i} - t_{min,i}$. As a consequence, nodes with long deadlines will be severely restricted by nodes with short deadlines and, hence, they cannot benefit from their long deadlines. To solve this problem, let us now consider the case of different m for every node in the network:

$$m_{ij} = \left\lceil \frac{t_{max,i} - t_{min,i}}{t_{min,j}} \right\rceil, \tag{19}$$

where m_{ij} is the number of packets a node j can send in nodes i's interval $t_{max,i} - t_{min,i}$. As a result, we can extend (12) as follows:

$$\Delta_{coll,i} = l_i \Bigg(\sum_{j=1;j\neq i}^{n} m_{ij} \Bigg) + \sum_{j=1;j\neq i}^{n} m_{ij} \cdot l_j, \tag{20}$$

and (16) now becomes:

$$1 - p_i = \left(\frac{l_i \Big(\sum_{j=1;j\neq i}^{n} m_{ij} \Big) + \sum_{j=1;j\neq i}^{n} m_{ij} \cdot l_j}{t_{max,i} - t_{min,i}} \right)^k . \tag{21}$$

This can be reshaped to obtain $t_{min,i}$:

$$t_{min,i} \leq t_{max,i} - \frac{l_i \Big(\sum_{j=1;j\neq i}^{n} m_{ij} \Big) + \sum_{j=1;j\neq i}^{n} m_{ij} \cdot l_j}{\sqrt[k]{1 - p_i}} . \tag{22}$$

Note that (17) reduces to (6) for $l_i = l_{max}$ and $d_i = d_{max}$. Similarly, (22) reduces to (7) for $t_{max,i} = t_{max}$, $l_i = l_j = l_{max}$ and $m_{ij} = m$. Here, p_i in (21) becomes p as per (5), i.e., the probability that at least one out of k packets reaches its destination in time is the same for all nodes in the network.

Effect of arbitrary deadlines. Next, we discuss the effect of arbitrary deadlines on reliability. As noted above, it is not meaningful to use the same m for every node, since a fixed m limits the benefits of modeling arbitrary deadlines in the network.

Similar to packet sizes l_i, deadlines d_i have an impact on nodes' reliability. Whereas different l_i have an influence on the collision interval as per (20), different d_i have an impact on $t_{max,i}$ as shown in (17). As a result, the length of the interval $t_{max,i} - t_{min,i}$ varies with d_i and, correspondingly, this influences p_i as per (21).

However, nodes will also have higher m_{ij} for greater deadline ratios as described in (19). That is, although the denominator in (21) increases, the numerator also increases resulting in a non-linear behavior. However, the positive effect predominates in most cases. This is exploited by the proposed algorithm shown in Alg. 1 to find a value of $t_{min,i}$ that maximizes reliability p_i — starting from a minimum required value — for a given set of n nodes.

Alg. 1 is based on the following principles: First, a node j with a short deadline d_j and consequently a short $t_{min,j}$ produces a high m_{ij} for node i with a long d_i. This is because the short $t_{min,j}$ fits multiple times in $t_{max,i} - t_{min,i}$. Conversely, we make only one packet of node i fit in $t_{max,j} - t_{min,j}$, i.e., $m_{ji} = 1$ implying $t_{min,i} > t_{max,j} - t_{min,j}$. Second, we know from Section 5.3 that a greater m

Algorithm 1 Optimizing $t_{min,i}$ for each node i

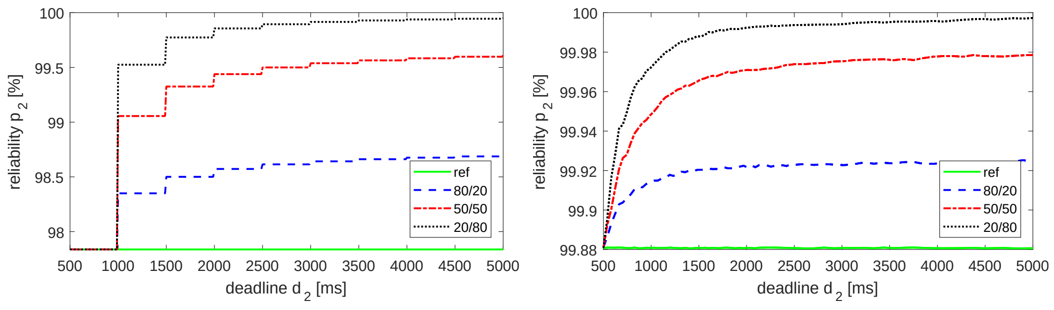
```
Require: set of nodes with $l_i$, $d_i$, minimum required $p_i$
Require: $n$ and $k$
 1: sort nodes in order of non-decreasing deadlines $d_i$
 2: $t_{max,1} = \frac{d_1 - l_1}{k}$
 3: $t_{min,1} = \frac{t_{max,1}}{2}$
 4: if (22) does not hold for $i = 1$ and $m_{1j} = 1 \,\forall j$ then
 5:     return (not feasible)
 6: end if
 7: for $i = 2$ to $n$ do
 8:     $t_{max,i} = \frac{d_i - l_i}{k}$
 9:     $m_{i1} = 0$
10:     while 1 do
11:         $m_{i1} = m_{i1} + 1$
12:         $t_{min,i} = t_{max,i} - m_{i1} \cdot t_{min,1}$
13:         $m_{ij} = \left\lceil \frac{t_{max,i} - t_{min,i}}{t_{min,j}} \right\rceil \forall\, j < i$ and $m_{ij} = 1 \,\forall\, j > i$
14:         if (22) does not hold or $t_{min,i} < \frac{t_{max,i}}{2}$ then
15:             if $m_{i1} = 1$ then
16:                 return (not feasible)
17:             else
18:                 restore last (valid) values of $m_{ij}$ and $t_{min,i}$
19:                 break
20:             end if
21:         end if
22:     end while
23: end for
24: return (feasible)
```

generally leads to less reliability. As a result, Alg. 1 limits any node j's own m_{jj} to be equal to 1, which means that $t_{min,j} \geq \frac{t_{max,j}}{2}$ has to hold as per (19). All other m_{ij} for $i \neq j$ will be greater than 1 for any node i with $d_i > d_j$ and equal to 1 for $d_i < d_j$.

Alg. 1 starts by sorting nodes in non-decreasing order of d_i. It then calculates $t_{max,1}$ and $t_{min,1}$ of the node with the shortest deadline, where $t_{min,1}$ is set to $\frac{t_{max,1}}{2}$ with $m_{11} = 1$ as mentioned above. By setting $t_{min,1}$ to the lowest possible value, we maximize the interval $t_{max,1} - t_{min,1}$ and consequently maximize p_1 — see (21). This $t_{min,1}$ is checked for validity using (22) in line 4 assuming $m_{1j} = 1 \,\forall j$, since $d_1 \leq d_j$ holds for $2 \leq j \leq n$. Clearly, in the case that (22) does not hold for the chosen $t_{min,1}$, the algorithm will exit with an error.

From line 7 onwards, Alg. 1 iterates over the remaining nodes computing $t_{max,i}$ in line 8 and $t_{min,i}$ in line 9 to 21. More specifically, $t_{min,i}$ is calculated by gradually incrementing m_{i1} and subtracting $m_{i1} \cdot t_{min,1}$ from $t_{max,i}$. This reduces $t_{min,i}$ in steps of $t_{min,1}$ making (21)'s denominator greater. A greater m_{i1} also increases (21)'s numerator, however, the positive effect predominates and p_i improves, i.e., it increases, in most cases.

The value of $t_{min,i}$ obtained this way has to be checked for validity. To this end, Alg. 1 first calculates all m_{ij} in line 13 by either using (19) for $j < i$, i.e., for the nodes with shorter deadlines, or setting $m_{ij} = 1$ for all $j > i$ according to the above discussion. In line 14, the value of $t_{min,i}$ is



(a) worst case (computed) (b) average (simulated)

Fig. 9. The maximum possible reliability p_2 is shown for a system with two node types 1 and 2: $d_1 = 500\,ms$, $500 \leq d_2 \leq 5000\,ms$ and m_{ij} are computed by Alg. 1. The different curves show p_2 for different n_1/n_2-ratios, i.e., *20/80* means 20 % type 1 and 80 % type 2 nodes with $n_1 + n_2 = n = 30$. The *ref* curve shows the behavior of the basic scheme from Section 4, which does not allow modeling different deadlines for each node.

checked to be between the bounds given by (22) and $\frac{t_{max,i}}{2}$ respectively. If this is not the case, the program exits with an error when $m_{i1} = 1$, i.e., there was no previous value of m_{i1} for which a valid $t_{min,i}$ could be found. Otherwise, it continues with the next node i until valid $t_{min,i}$ have been found for all i or the program exists with an error.

Let us now analyze an exemplary system consisting of two node types in Fig. 9: type 1 with short deadlines of $d_1 = 500\,ms$ and type 2 with varying deadlines of $500 \leq d_2 \leq 5000\,ms$. Again, for simplicity, the subindexes 1 and 2 represent the *node type* 1 and 2 respectively as defined above. The remaining parameters were set to $n = 30$ and $k = 3$; m_{ij} are computed based on Alg. 1. Note that, for the sake of illustration, we have set $l_1 = l_2 = 400\,\mu s$ in this simulation.[4] The figure on the left (Fig. 9a) shows the computed worst-case reliabilities, whereas the figure on the right (Fig. 9b) shows the simulated average reliabilities — more details on the simulation can be found in Section 7.

By modeling arbitrary deadlines, we can reach higher reliabilities than in the case of assuming that all deadlines are the same. In this case, all deadlines are set to be equal to the shortest possible one in the system (see Section 4), which incurs pessimism as shown by the *ref* system. In the worst case, in Fig. 9a, this allows for at most a reliability of 97.8 % for all d_2, which equals a sequence loss rate $1 - p \approx 2 \cdot 10^{-2}$. In case of the *20/80* system — 20 % of type 1 and 80 % of type 2 nodes — the reliability increases to around 99.95 % and the sequence loss rate $(1 - p)$ decreases to $5 \cdot 10^{-4}$ for $d_2 = 5000\,ms$. This is an improvement by a factor of 40. On average, in Fig. 9b, the *ref* system achieves a reliability of 99.88 % (sequence loss rate $1 - p \approx 1 \cdot 10^{-3}$) and the *20/80* system a reliability of 99.997 % (sequence loss rate of $(1 - p) = 3 \cdot 10^{-5}$) for $d_2 = 5000\,ms$. This is an improvement by a factor of 33. Again, reliabilities are always higher on average than in the worst case, which validates the proposed theory.

It can also be observed for the worst case in Fig. 9a that the reliability p_2 increases stepwise every $500\,ms$ until slowly saturating for a large d_2. At every step, viz., when d_2 is a multiple of d_1, m_{i1} can be increased by one as per Alg. 1. This allows reducing the value of $t_{min,2}$ and, hence, results in a better reliability p_2 as shown in Fig. 9. As expected, the more type 2 nodes there are in the system, the higher the value of p_2 we can reach, since there will be more nodes with longer

[4]With $l_1 = l_2 = 88\,\mu s$ as before, reliability values are so high (around $1 - 10^{-8}$) that it becomes difficult to reproduce in simulation. As a result, curves fully coincide making it hard to see any difference.

deadlines. Similarly, in Fig. 9b, reliability also increases with rising d_2, however, there are no sharp steps, but these are rather smoothed by the simulation.

7 SIMULATION AND COMPARISON

Complementary to Section 5.3, where the worst-case behavior is illustrated in a numerical evaluation, in this section, we analyze the average-case behavior of our MAC and compare it to other protocols as explained below. More specifically, we perform a simulation with different parameters in OMNeT++ [20] and record statistical values for very large numbers of transmissions — at least 1,000,000 packets were simulated for each of the presented curves.

The simulated network is based on the intelligent assembly line from Section 5 and consists of one sink, i.e., the central control station, and n identical transmitting nodes, i.e., mobile robots and human workers. These are randomly distributed in a field of 50 m x 50 m and periodically transmit their position and speed to the sink located in the center of the field. All data is processed by the OMNeT++ framework at runtime, comparing time stamps of simulated packets to determine whether they overlap and, hence, interfere with each other.

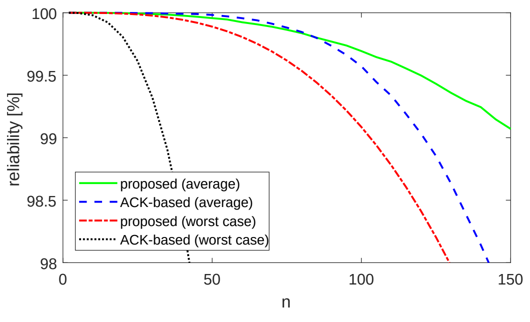
We compare the following MAC protocols in simulation:

- The *proposed* scheme is our MAC protocol as presented in Section 4 of this paper.
- The *ACK-based* scheme is a MAC protocol based acknowledgments and carrier sensing as per [13].
- The *TDMA* (time division multiple access) scheme is a synchronous protocol, in which nodes transmit during dedicated time slots to avoid collisions. For this, they share a common clock by periodically receiving synchronization beacons from the sink.
- The *CSMA* (carrier sense multiple access) scheme is an asynchronous protocol that implements non-persistent carrier sensing and a random backoff scheme, similar to IEEE 802.15.4.

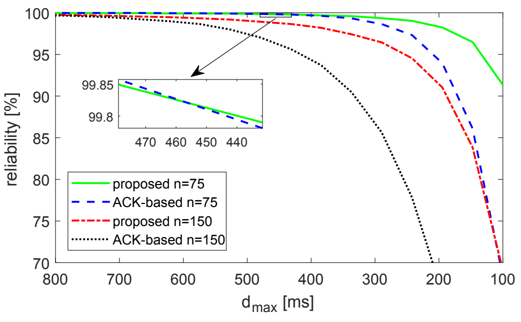
The *ACK-based* scheme is a protocol based on our previous work [13] that follows almost the same principles as the *proposed* scheme. That is, nodes transmit k packets within a deadline d_{max} to guarantee a specific reliability p, whereby inter-packet times are randomly selected from an interval $[t_{min}, t_{max}]$. In contrast to the *proposed* scheme, the *ACK-based* scheme uses carrier sensing for collision avoidance and ACKs to reduce the average number of packets sent. For the sake of comparison, we selected $k = 3$ for the *proposed* and the *ACK-based* schemes, since this is sufficient for a good average performance. As discussed before, a bigger k would be necessary to guarantee a high reliability in the worst case.

In the *TDMA* scheme, communication is organized in cycles. Each such cycle starts with a beacon message for synchronization, followed by n dedicated time slots for data transmission, where n is the number of nodes to transmit in the current cycle. In our experiments, the beacon is a normal data packet, which contains synchronization and network information, having a length of 88 μs. Time slots have a length of 404 μs, i.e., they contain a data packet, an ACK and processing and switching times. In addition, a guard time interval t_{guard} is added before every slot and beacon to account for clock drift.

For increased robustness, there are three TDMA cycles within a deadline, which allows nodes to retransmit packets or receive another beacon, if the previous ones were corrupted. These cycles are evenly spaced within the deadline, i.e., these start at times 0, 167 ms and 333 ms from the beginning of d_{max}. Lastly, the beacon rate needs to be configured, i.e., the time after which a node has to receive a beacon to resynchronize before its clock drift becomes too large and it starts violating slot boundaries. Assuming a standard oscillator with 100 PPM, the beacon rate will mainly depend on t_{guard}. The greater the beacon rate, the shorter t_{guard} can be, resulting in short delays. However, since delay is generally less important as long packets arrive before d_{max}, we select t_{guard} to be as



(a) For increasing number of nodes



(b) For a decreasing deadline

Fig. 10. Reliability of the *proposed* and *ACK-based* schemes for different system parameters

long as possible to lower the beacon rate and decrease energy consumption. That is, it is meaningful to select a beacon rate of 3.5 s and $t_{guard} = 700\,\mu$s. Note that the sink sends a beacon in every cycle, but sensor nodes only listen for them if they need to synchronize, i.e., every 3.5 s.

The *CSMA* scheme is based on the non-persistent, non-slotted CSMA protocol as defined in IEEE 802.15.4. Prior to every packet transmission, nodes first perform a random backoff and then use carrier sensing to assess the channel state. If it is free, they start transmitting. Otherwise, if the channel is blocked or whenever a transmission failed, i.e., no ACK was received, nodes perform another random backoff before sensing the channel again. The backoff time is calculated using the binary exponential backoff formula $\text{rand}(1, 2^c - 1) \cdot t_{base}$ with c being the number of failed transmissions. According to IEEE 802.15.4, we set the maximum backoff and retransmission numbers to 4 and 3 respectively and use a base multiplier of $t_{base} = 320\,\mu$s. The contention window — a value which defines the minimum and maximum length of the backoff time — was increased to [32, 1024] to prolong backoff times and allow *CSMA* to use the full deadline (with default parameters as per IEEE 802.15.4, only a short fraction of the deadline is used). This increases the overall performance of *CSMA* in the simulation and, in particular, reduces collision rates and energy consumption.

Note that *CSMA* and *TDMA* are the core technologies for many other approaches [8][17][25]. As a result, during high contention — which we simulate in our experiments — these fall back to the performance of either *CSMA* or *TDMA* and are, hence, not further considered in the presented comparison. For example, [8][17] reduce to *CSMA* and [25] reduces to *TDMA* at high congestion.

Finally, each node is assumed to be equipped with a CC2545 [19] transceiver IC for handling data transmissions and receptions. The packet length is again $l_{max} = 88\,\mu s$ and the transmission power is set high enough to ensure good link quality — we assume a packet error rate of PER = 0 %.[5] The deadline, i.e., the maximum tolerable delay in which the central control station must receive a packet from every node, is set to $d_{max} = 500\,ms$. Further, ACKs have the same frame format as data packets, but a reduced payload of 2 bytes containing the source node address. Their length is consequently $l_{ACK} = 56\,\mu s$. The transceiver switching time, i.e., the time required by a node to switch between receive and transmit mode, is $t_{switch} = 130\,\mu s$ and the carrier sensing has been fixed to $t_{sen} = 150\,\mu s$, i.e., slightly longer than t_{switch} to be able to detect the *gap* between data packet and ACK — see [13] for more details.

[5]This assumption facilitates the understanding of the simulation results, since these are not distorted by packet reception errors. We later analyze the effects of PER separately in Section 7.3.

7.1 Communication with and without ACK

As previously mentioned, ACKs provide a feedback channel from sink to source allowing for retransmission schemes or dynamic adaption of network parameters. Although ACKs typically improve the average performance of the network, they also generate additional overhead. This does not only increase the overall complexity of the network, but also affects its performance, especially, during high contention phases as shown next.

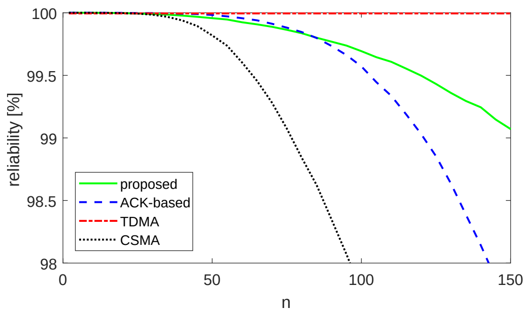
Fig 10a shows the calculated worst-case and simulated average reliabilities of the *proposed* and *ACK-based* schemes. We can observe that using an ACK scheme greatly deteriorates the worst-case reliability by introducing two additional error sources: the ACK packet itself and the transceiver switching times in which the sink can potentially miss packets. This deterioration is independent of k, meaning that the worst-case reliability with ACKs is always lower than without ACKs.

The average reliability, on the other hand, behaves differently. In Fig. 10a, the *ACK-based* scheme first offers slightly higher reliability for small n. That is, in small networks, the data traffic is low enough that ACKs prevent more collisions than they create. For higher n, however, this effect reverses as the traffic load increases and the channel starts to saturate. In the above example, the *ACK-based* scheme offers a lower reliability from $n = 80$ onwards than the *proposed* scheme.

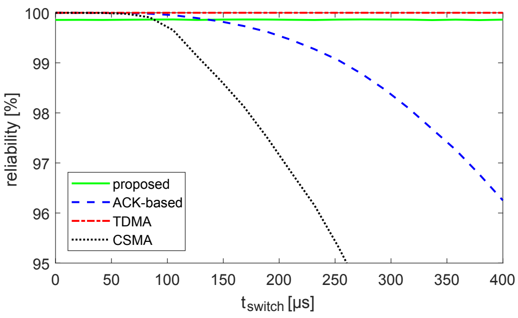
The same effect can be observed when varying other system parameters, for example, the deadline, as shown in Fig. 10b. Here, we can see that the average reliability decreases when the deadline is reduced, since nodes have to transmit their packets in a shorter period of time. This increases traffic load, which, in return, leads to higher collision rates. For $n = 75$, the *ACK-based* scheme offers a slightly higher reliability than the *proposed* scheme for large deadlines, which reverses when the deadline becomes smaller than 460 ms. Again, from this point onwards, the traffic load becomes large enough such that ACKs cause more collisions than they prevent. For $n = 150$, the same effect can be observed, however, the turning point is reached at $d_{max} = 1000$ ms, i.e., for a larger deadline due to the higher n. Lastly, for short deadlines below 100 ms, the *proposed* scheme with $n = 150$ starts performing better than the *ACK-based* scheme with $n = 75$. For more details, see Appendix B.

The number of packets k also influences average reliability. In general, the greater k, the more efficient the *ACK-based* scheme is, since more packets can be skipped in the event of a successful transmission. For example, with $k = 12$, up to 11 packets can be omitted if the first one is successful. With small packet numbers, however, savings are lower and the additional overhead outweighs benefits depending on the number of nodes n. The larger the network, the more traffic there is and the *ACK-based* scheme incurs in more collisions as shown in Fig. 10a. With $n = 50$, for example, *ACK-based* outperforms the *proposed* scheme for $k = 3$ on average — note again that *proposed* is always better in the worst case. Although not shown in Fig. 10a, with $n = 100$ and $n = 200$, *ACK-based*'s average performance does not become better until $k = 4$ and $k = 6$ respectively.

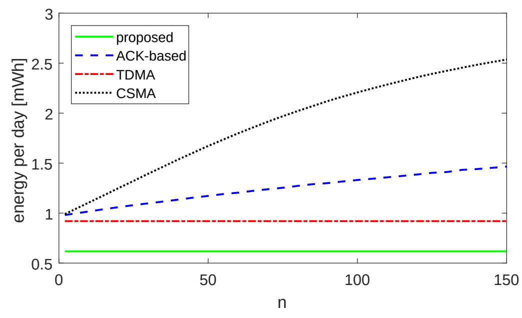
In summary, while the worst-case reliabilities are always higher for the *proposed* scheme, the average performance can be higher for an acknowledged protocol (such as ACK-based in this comparison) depending on network parameters such as deadline, packet numbers, etc. In general, the *proposed* scheme is well suited for networks with a large number of nodes and high traffic load, i.e., scenarios in which ACKs are more a burden than of help. For very large packet numbers, however, acknowledged protocols can be better suited, since these reduce the number of packets sent on average. Nevertheless, the proposed scheme still achieves very high performance for a wide range of network settings, while offering other advantages such as low complexity, independence of the transceiver quality (i.e., switching times) and lower energy consumption. In the next section, we further compare the *proposed* scheme to the well established *TDMA* and *CSMA*.



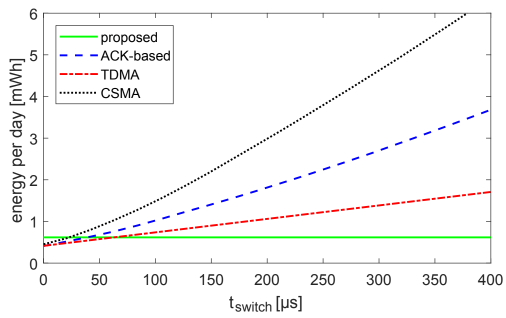
(a) Reliability for increasing number of nodes



(b) Reliability for increasing transceiver switching time ($n = 75$)



(c) Energy consumption for increasing number of nodes



(d) Energy consumption for increasing transceiver switching time ($n = 75$)

Fig. 11. Average reliability and per-node energy consumption in one day of the different MAC protocols for varying number of nodes and transceiver switching times.

7.2 Reliability and energy efficiency

In the following, we compare the *proposed* scheme to *CSMA* and *TDMA* as illustrated in Fig. 11 for different settings. In the case of average reliability in Fig. 11a, *TDMA* offers a very high value of 100 % without external interference as it effectively prevents internal collisions between nodes by its time slot arbitration. In contrast, *CSMA* offers the lowest reliability due to its backoff mechanism that is designed for fast data transfer rather than reliability. In particular, backoff times are initially very short and only increased if retransmissions fail. Short backoff times cause packets to be sent faster and, therefore, lead to a higher channel utilization, i.e., a big amount of data per time unit. This generally increases collision probability, in particular, if multiple nodes are triggered simultaneously [2].[6] The proposed scheme, on the other hand, balances network load by distributing packets uniformly along the full deadline. This avoids peaks in data traffic and leads to a higher reliability, e.g., more than 99 % for $n \leq 150$ in Fig. 11a.

Next, let us analyze the average reliability for increasing t_{switch}, i.e., the time for switching between receive and transmit mode. This mainly depends on the settling time of the transceiver's frequency synthesizer, which in turn depends on modulation and circuit speed [15]. Most transceivers using DSSS (Direct Sequence Spread Spectrum), an interference-insensitive modulation scheme

[6]Please note that nodes are activated at random time instants. However, it may still (randomly) happen that nodes are triggered simultaneously.

typically used in 802.15.4 [16], have relatively long switching times, for example, 192 μs for the commonly-used CC2420 [18]. Other modulation schemes such as OFDM (Orthogonal Frequency-Division Multiplexing) allow much faster switching speeds, for example, 2 μs in the case of the transceiver MAX2837 [10]. However, different types of modulation have different advantages and disadvantages and might not be well suited for all applications. For example, OFDM results in a typically higher vulnerability to external interference and increased costs compared to DSSS [16].

As can be observed in Fig. 11b, the *proposed* scheme is not affected by changing t_{switch}, since nodes do not switch modes. With *TDMA*, where nodes have to switch twice per slot to receive an ACK, reliability also remains unchanged (however, slots increase in length to accommodate t_{switch} affecting delays). Both *CSMA* and the *ACK-based* scheme show a decreasing reliability, since nodes are active for a longer time, resulting in a higher probability that other nodes have to back off. Again, *CSMA* shows a greater decrease in reliability due to higher internal collision rates, similar as in Fig 11a. On the contrary, if t_{switch} is reduced, for example, by selecting a faster transceiver, the acknowledged schemes can achieve a higher reliability than the *proposed* scheme on average. In the above example, this happens for a switching time $< 140\,\mu$s for the *ACK-based* scheme and $< 90\,\mu$s for *CSMA*.

Next, let us analyze the energy consumption by the different MAC protocols. In Fig. 11c, we can see the average energy consumption per node on one day, in which the system was first active for 12 h and then in sleep for the remaining 12 h. The amount of energy consumed was calculated by multiplying the recorded times each node spent in receive, transmit and sleep mode with the corresponding power levels from the CC2545 transceiver [19]. That is, a node requires 62.5 mW during transceiver switches and in receive mode, 80.5 mW in transmit mode and 4.5 μW in sleep mode at $V_{DD} = 3\,V$. Note that during the 12 h sleep period, *TDMA* is deactivated and no beacons are transmitted to save energy.

The energy consumption of a node depends on the number of packets sent, as well as on the packets themselves. For the *proposed* scheme, a single packet transmission causes a node to be in transmit mode for $l_{max} = 88\,\mu$s, which requires 7 μWs. When sending ACKs, a node has to switch to receive mode after sending a packet, listen for the ACK and switch back to transmit mode. In our example, each node switch takes 130 μs and receiving the ACK takes 56 μs, resulting in additional 316 μs in receive mode and a total energy consumption of 26.8 μWs. For the *ACK-based* scheme and *CSMA*, nodes have to spend additional 150 μs in receive mode for carrier sensing, increasing the total receive time to 466 μs and energy consumption to 36 μWs.

As shown in Fig. 11c, energy consumption of the *proposed* scheme is independent of n, since nodes always transmit 3 packets per deadline. Similarly, with *TDMA*, nodes always send 1 packet per deadline, since there are no (internal) collisions and external interference is neglected. However, since transmitting one packet with ACK requires more energy than transmitting 3 packets without ACK, TDMA consumes more energy than the *proposed* scheme in total. In addition, nodes have to receive beacons periodically, which makes up for around 18 % of the total energy demand. The *ACK-based* scheme and *CSMA*, on the other hand, show an increasing energy consumption for rising n, as more collisions occur and packets have to be retransmitted more often. This first rises linearly for low n, but starts to saturate as retransmission and backoff retries become a limiting factor. In case of *CSMA*, energy increases more steeply due to higher internal collisions rates, therefore, reaching saturation more quickly, i.e., for lower n.

Fig. 11d shows the average energy consumption per node for varying transceiver switching times. As we can see, all modes except the *proposed* scheme require more energy when switching times increase, since data transmissions require more time. In the case of *TDMA*, energy rises linearly, whereas for the *ACK-based* scheme and *CSMA*, a slight curvature can be observed. This is due to the higher number of collisions — and lower reliability as shown previously in Fig. 11b — requiring

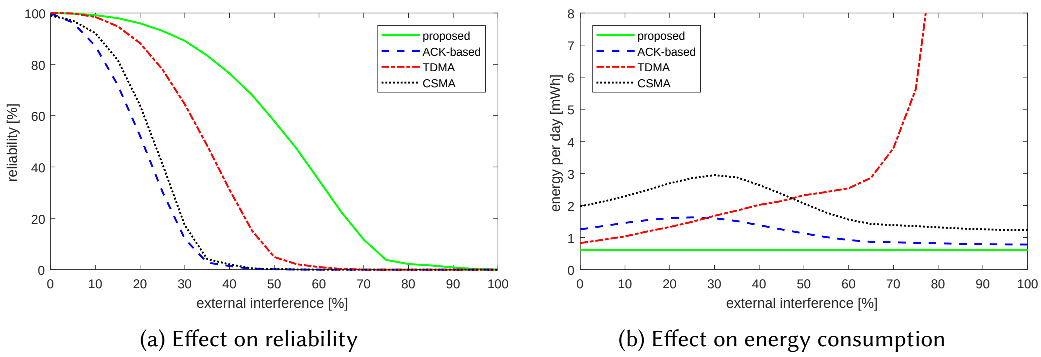


Fig. 12. Effect of external interference on reliability and energy consumption ($n = 75$)

more retransmissions. For short switching times, the acknowledged schemes can reach a lower energy consumption than the *proposed* scheme. This is reached for switching times below 60 μs for *TDMA*, 22 μs for *CSMA* and 40 μs for the *ACK-based* scheme.

7.3 Packet reception errors

This section analyzes how packet reception errors (PER $> 0\,\%$) affect the performance of the different MAC protocols, i.e., when sources external to the network cause additional packet loss. In real-world scenarios, there are numerous physical effects that might cause packet loss, for example, fading, shadowing, etc. Another, common source of error is noise on the communication channel. This typically originates from neighboring networks, for example, devices using WiFi, Bluetooth, etc., or other sources generating electromagnetic emissions. Depending on the signal strength, these can cause data corruption when overlapping with transmissions.

In the following, we evaluate reception errors by using our model of external interference from Section 6.1. That is, we describe the maximum probability of external interference by σ independent of the concrete source. In our simulation, a separate node randomly emits interference pulses of variable length from 48 to 304 μs, which corresponds to the duration of data packets with payloads between 0 and 64 bytes. The level of interference, i.e., σ, can be changed by varying the duty cycle of the interfering node, i.e., the ratio of time it actively transmits to the inter-pulse separation. This is stepwise increased from 0 % (no interference) to 100 % (blocked channel) in the following experiments. Finally, we again make the pessimistic assumption that any overlapping of transmissions leads to packet loss, i.e., there is no capture effect and data cannot be recovered.

Fig. 12a shows the average reliability of the different MAC protocols for an increasing level of external interference. As expected, a higher level of interference decreases the reliability of all protocols. That is, transmissions are disrupted more often leading to a higher likeliness that data cannot be conveyed anymore within the specified deadline, i.e., with a maximum number of retransmissions. The *proposed* scheme is the most robust protocol due to its low overhead, i.e., neither ACKs, nor carrier sensing nor beacons are used, hence, the probability of being disrupted is lower. For very noisy environments with interference levels of 40 %, it still offers a relatively high reliability of $\geq 80\,\%$, whereas the *ACK-based* scheme and *CSMA* drop to almost zero. These generally achieve a much lower reliability due their ACK-mechanism and carrier sensing: when an ACK is corrupted, the node assumes its packet transmission was corrupt and retransmits, while interference during carrier sensing causes the node to back off. In the above example, *CSMA* offers a sightly better reliability than the *ACK-based* scheme due to higher retransmission (and backoff) numbers. Lastly, with *TDMA*, robustness is lower than for the *proposed* scheme due to its larger overhead (ACKs and beacons), but higher than for *CSMA* and the *ACK-based* scheme due to lower internal collision rates and the lack of carrier sensing.

Fig. 12b shows the average energy consumption per node and day for an increasing level of (external) interference. Again, the *proposed* scheme offers constant energy consumption, since it always transmits $k = 3$ packets per deadline. *CSMA* and the *ACK-based* scheme, on the other hand, first show an increasing energy consumption, which then decreases from 25 to 30 % interference onwards until finally reaching a lower level than for zero interference. This is because, at this level of interference, the channel starts saturating and nodes start backing off without transmitting, which requires less energy. For very high interference, i.e., when nodes almost only back off, energy is even lower than for zero interference, i.e., when typically only a single packet is transmitted. Note that *CSMA*'s energy consumption is higher than that of the *ACK-based* scheme due to the larger retransmission (and backoff) numbers. Finally, *TDMA* shows a linearly rising energy consumption until around 60 % interference. From then onwards, energy drastically increases as most beacons are corrupted and nodes start to lose synchronization. In this case, nodes have to stay in receive mode longer and listen for another beacon in order to re-synchronize with a drastic impact on energy consumption.

8 CONCLUDING REMARKS

In this paper, we proposed a MAC protocol for assessing reliability in unacknowledged, time-constrained WSNs with periodic data traffic. More specifically, we make nodes send data packets at random instants within a time interval $[t_{min}, t_{max}]$ from the last transmission. By adjusting t_{min} and t_{max}, we can provide a certain probability on packet delivery in the worst case and thereby guarantee a maximum age for data.

Since the presented MAC forgoes acknowledgments, carrier-sensing, and synchronization, the resulting overhead is low. This makes the presented approach ideal for applications that require high flexibility and reduced energy consumption. In contrast to existing approaches from the literature, the presented MAC accounts for different node types with arbitrary packet lengths and deadlines. This allows modeling packet loss more accurately and results in an overall higher performance.

Based on extensive simulations, we further evaluated the performance of the proposed MAC in comparison to acknowledged protocols such as CSMA and TDMA. Our experiments show that the *proposed* approach generally leads to a higher worst-case reliability. The average performance, however, can be lower depending on network parameters such as deadline, number of packets, etc., and, in particular, if (typically more expensive) transceivers with fast switching times are used instead. Overall, the proposed MAC is best suited for networks with a large number of nodes and high traffic load, i.e., scenarios in which ACKs are more a burden than of help. For all other settings, it still achieves acceptable performance on average, while offering other advantages such as low complexity, independence of the transceiver's quality (i.e., switching times), lower energy consumption, and robustness against external interference.

REFERENCES

[1] Standard for Low-Rate Wireless Networks, IEEE 802.15.4-2015. page 411.

[2] G. Anastasi, M. Conti, and M. D. Francesco. A Comprehensive Analysis of the MAC Unreliability Problem in IEEE 802.15.4 Wireless Sensor Networks. *IEEE Transactions on Industrial Informatics*, 7:52–65, 2011.

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

[4] S. Brienza, M. Roveri, D. D. Guglielmo, and G. Anastasi. Just-in-Time Adaptive Algorithm for Optimal Parameter Setting in 802.15.4 WSNs. *ACM Transactions on Autonomous and Adaptive Systems*, 10:27, 2016.

[5] E. Celada-Funes, D. Alonso-Roman, C. Asensio-Marco, and B. Beferull-Lozano. A Reliable CSMA Protocol for High Performance Broadcast Communications in a WSN. In *Proceedings of the IEEE Global Communications Conference (GLOBECOM)*, 2014.

[6] B. Dezfouli, M. Radi, K. Whitehouse, S. A. Razak, and H.-P. Tan. CAMA: Efficient Modeling of the Capture Effect for Low Power Wireless Networks. *ACM Transactions on Sensor Networks*, 11:20, 2014.

[7] B. Firner, C. Xu, R. Howard, and Y. Zhang. Multiple Receiver Strategies for Minimizing Packet Loss in Dense Sensor Networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc)*, 2010.

[8] M. H. S. Gilani, I. Sarrafi, and M. Abbaspour. An Adaptive CSMA/TDMA Hybrid MAC for Energy and Throughput Improvement of Wireless Sensor Networks. *Elsevier Journal of Ad Hoc Networks*, 11:1297 – 1304, 2013.

[9] D. D. Guglielmo, F. Restuccia, G. Anastasi, M. Conti, and S. K. Das. Accurate and Efficient Modeling of 802.15.4 Unslotted CSMA/CA through Event Chains Computation. *IEEE Transactions on Mobile Computing*, 15:2954–2968, 2016.

[10] Maxim Integrated. MAX2837 Datasheet. URL: https://datasheets.maximintegrated.com/en/ds/MAX2837.pdf.

[11] E. D. N. Ndih and S. Cherkaoui. Adaptive 802.15.4 Backoff Procedure to Survive Coexistence with 802.11 in Extreme Conditions. In *Proceedings of the IEEE Consumer Communications & Networking Conference (CCNC)*, 2016.

[12] P. Parsch and A. Masrur. A Reliability-Aware Medium Access Control for Unidirectional Time-Constrained WSNs. In *Proceedings of the International Conference on Real Time and Networks Systems (RTNS)*, 2015.

[13] P. Parsch and A. Masrur. A Reliable MAC for Delay-Bounded and Energy-Efficient WSNs. In *Proceedings of the 23rd IEEE International Conference on Embedded and Real-Time Computing Systems and Applications (RTCSA)*, 2017.

[14] W. B. Pottner, S. Schildt, D. Meyer, and L. Wolf. Piggy-Backing Link Quality Measurements to IEEE 802.15.4 Acknowledgements. In *IEEE International Conference on Mobile Ad-Hoc and Sensor Systems (MASS)*, 2011.

[15] W. Rhee. *Wireless Transceiver Circuits: System Perspectives and Design Aspects.* CRC Press, 2015.

[16] M. Sellars and D. Kostas. Comparison of QPSK/QAM OFDM and Spread Spectrum for the 2-11 GHz PMP BWAS. Technical report, Adaptive Broadband Corp., 2000.

[17] B. Shrestha, E. Hossain, and K. W. Choi. Distributed and Centralized Hybrid CSMA/CA-TDMA Schemes for Single-Hop Wireless Networks. *IEEE Transactions on Wireless Communications*, 13:4050–4065, 2014.

[18] Texas Instruments. CC2420 Datasheet. URL: http://www.ti.com/lit/ds/symlink/cc2420.pdf.

[19] Texas Instruments. CC2545 Datasheet. URL: http://www.ti.com/lit/ds/symlink/cc2545.pdf.

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

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

[22] Y. Zhang, B. Firner, R. Howard, R. Martin, N. Mandayam, J. Fukuyama, and C. Xu. Transmit Only: An Ultra Low Overhead MAC Protocol for Dense Wireless Systems. In *Proceedings of the IEEE International Conference on Smart Computing (SMARTCOMP)*, 2017.

[23] Y. Zhang, Y. Zhou, L. Gao, Y. Qian, J. Li, and F. Shu. A Blind Adaptive Tuning Algorithm for Reliable and Energy-Efficient Communication in IEEE 802.15.4 Networks. *IEEE Transactions on Vehicular Technology*, 66:8605–8609, 2017.

[24] 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.

[25] S. Zhuo, Z. Wang, Y. Q. Song, Z. Wang, and L. Almeida. A Traffic Adaptive Multi-Channel MAC Protocol with Dynamic Slot Allocation for WSNs. *IEEE Transactions on Mobile Computing*, 15:1600–1613, 2016.

A PRACTICAL FACTORS: 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 nodes do not synchronize in the proposed scheme, they will unavoidably have different time scales.

Recall that a node generates random inter-packet times t_{ix} in the interval $[t_{min}, t_{max}]$ — with a uniform distribution — and waits for these t_{ix} before sending any packet. A clock drift does not affect the generation of random inter-packet times, since bounds t_{min} and t_{max} are computed off-line. In other words, the length of the interval $[t_{min}, t_{max}]$ remains constant and, hence, (3) is still valid.

However, since a node *counts* for t_{ix} before sending a packet, a clock drift leads to an *absolute* waiting time $\bar{t}_{ix}$ different than t_{ix}, i.e., the time without clock drift. As a result, this needs to be considered when selecting the bounds t_{min} and t_{max} for the random inter-packet times.

Towards this, recall that a node may send m packets in an interval of length $t_{max} - t_{min}$. Again, the node waits for a random t_{ix} before sending each packet where t_{ix} is always greater than or equal to t_{min}. In an extreme case, it may happen that the node waits for t_{min} time before each of the above mentioned m packets.

We need to consider clock drift so as to guarantee that at most m packets be in an interval of length $t_{max} - t_{min}$, otherwise (3) will stop being valid. To this end, let us denote by Δt_{min} the maximum possible clock deviation (with respect to an ideal, non-drifting clock) in an interval of length t_{min}. As a result, we proceed as follows to incorporate clock drift into (1):

$$\begin{aligned} m \cdot (t_{min} - \Delta t_{min}) &\geq t_{max} - t_{min}, \\ t_{min} &\geq \frac{t_{max} + m \cdot \Delta t_{min}}{m+1}. \end{aligned} \tag{23}$$

Similarly, t_{max} is selected such that we can guarantee that k packets be sent within the specified deadline d_{max}. To consider clock drift in this case, let us denote by Δt_{max} the maximum possible clock deviation in an interval of length t_{max}. In the worst case, a node may need to wait for t_{max} before sending each of the k packets. We proceed as follows to incorporate clock drift in (6):

$$\begin{aligned} k \cdot (t_{max} + \Delta t_{max}) &\leq d_{max} - l_{max} \\ t_{max} &\leq \frac{d_{max} - l_{max} - k \cdot \Delta t_{max}}{k}. \end{aligned} \tag{24}$$

Note that this new bound for t_{max} can be used instead of (6) to compute the t_{min} that satisfies the reliability requirement p in (7). The value of t_{max} in (24) needs to be considered in computing the other bounds on t_{min} such as (4) and (23) taking clock drift into account.

Clearly, (23) and (24) requires us to know Δt_{min} and Δt_{max}, which depend on t_{min} and t_{max} respectively. However, we know that clock deviation due to clock drift will increase with the length of the considered time interval. Since $t_{min} < t_{max} < \hat{t}_{max}$ holds for $\hat{t}_{max} = \frac{d_{max}}{k}$, we have that $\Delta t_{min} < \Delta t_{max} < \Delta \hat{t}_{max}$ also holds where $\Delta \hat{t}_{max}$ is the maximum clock deviation in an interval of length $\hat{t}_{max}$. As a result, to resolve the above dependency, we can safely replace Δt_{min} and Δt_{max} by $\Delta \hat{t}_{max}$ in (23) and (24).

Effect of clock drift. Let us now examine the effect of clock drift on the proposed scheme by means of simulation. For this we regard an exemplary network with $n = 30$ nodes and the same simulation settings as used later in Section 7. To better assess the impact of clock drift on the network reliability, the values of t_{min} and t_{max} were computed using (7) and (6), i.e., without taking clock drift into consideration.

Fig. 13 shows how the clock drift affects the transmission reliability for different values of k. For this purpose, we have simulated a local clock for each node that randomly selects its drift value within $[-\Delta_{drift}, +\Delta_{drift}]$ after each sequence. Here, Δ_{drift} is the maximum drift of the current iteration, which is slowly increased from zero to 10%. Note that a clock drift of 10 % means that the frequency of the drifting clock is 10 % different than that of the non-drifting clock. For example, if we consider a 1 MHz crystal, the drift would cause it to run at either 900,000 Hz or 1,100,000 Hz.

As expected, we can see in Fig. 13 that a rising clock drift leads to a lower reliability for all numbers of k. This is caused by a superposition of the following effects: First, when the clock drift of a node is negative, i.e.,

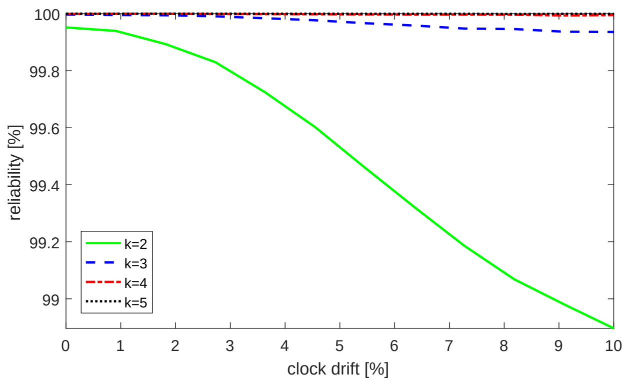
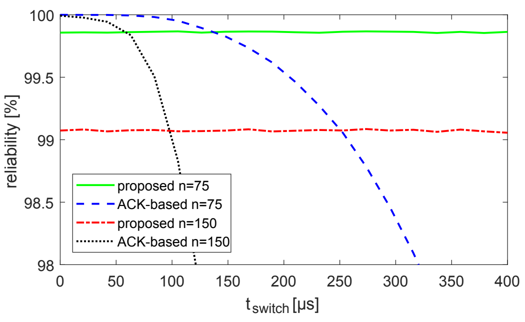


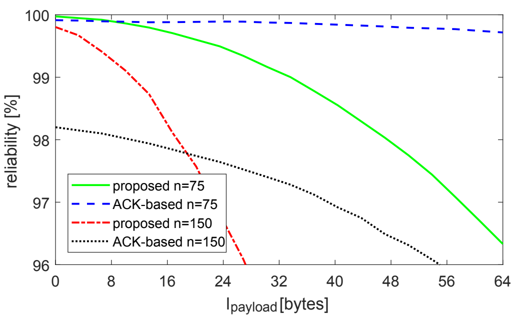
Fig. 13. Simulated reliability of the *proposed* algorithm for different values of k and an increasing clock drift

the clock frequency is lower, it counts time at a slower rate. This will enlarge the interval $t_{min} - t_{max}$ and therefore increase reliability. However, since t_{max} also increases, packets of a node can now miss the deadline as (6) is violated. Second, if the clock drift of a node is positive, it counts time at a higher rate, which results in a shorter interval $t_{min} - t_{max}$ and, hence, a decreased reliability. Additionally, (1) can start being violated and therefore reliability decreases further.

In summary, we can see in Fig. 13 that negative effects dominate on average and reliability decreases for a higher clock drift. This happens for all values of k, however, a greater k is more robust, as it is less probable to lose all packets of a sequence. On the other hand, even cheap crystal oscillators can guarantee a 100 ppm = 0.01 % accuracy. For this accuracy, the simulated reliability in Fig. 13 deteriorates by less than 0.02 % for $k = 2$ and 0.001 % for $k = 3$. For $k \geq 4$ the change is too small to be measured. Thus, in most of the cases, we can safely neglect clock drift when designing a network with the proposed scheme. That is, we do not need to use (23) and (24).



(a) For increasing transceiver switching time



(b) For increasing payload of data packets

Fig. 14. Average reliability of the *proposed* and *ACK-based* schemes for different system parameters

B COMMUNICATION WITH AND WITHOUT ACK — CONTINUED

Fig. 14a shows the average reliability when increasing the transceiver switching time t_{switch}, i.e., the time nodes require to switch between receive and transmit mode. In the case of the *proposed* scheme, this does not affect performance, since nodes do not change, but stay in transmit mode during data transfers. The *ACK-based* scheme, on the other hand, requires two mode switches[7] for each packet transmission and is consequently affected by this parameter. In general, increasing switching times lowers reliability, since the sink cannot listen for incoming data for a longer time, increasing the probability of missing packets. In addition, since other nodes cannot transmit in the *gaps* between mode switches, for example, between data packet and ACK, a greater switching time yields a longer blocking of the channel, leading to more backoffs and, ultimately, to a lower reliability. If the switching time is reduced, for example, by selecting a faster transceiver, the *ACK-based* scheme can achieve a higher reliability than the *proposed* scheme depending on the number of nodes in the network. For $n = 75$, this happens for a switching time $< 140\,\mu$s, while for $n = 150$ a faster transceiver with $< 95\,\mu s$ is required.

Next, Fig. 14b shows the average reliability when varying the payload $l_{payload}$ of a packet — note, the total packet length is the sum of payload plus 12 bytes overhead, see Section 5.1. As expected, increasing the payload reduces the reliability for both schemes, as network traffic increases and more collisions occur. The *ACK-based* scheme, however, loses reliability less quickly, due to carrier sensing. That is, if multiple packets overlap, the first one typically succeeds, as all later ones have to back off. The *proposed* scheme, on the other hand, loses all packets involved in the collision. In the above example, the *ACK-based* scheme offers a higher reliability for payloads greater than 9 bytes ($n = 75$) and 19 bytes ($n = 150$). For smaller payloads, however, the relatively large overhead of the *ACK-based* scheme starts limiting performance, yielding lower reliability than for the *proposed* scheme.

Received September 2017; revised July 2018; accepted July 2018

[7]Each data transfers starts with carrier sensing, followed by a switch to transmit mode to send a data packet and a switch back to receive mode to be able to receive an ACK. Similarly, the sink also has to switch modes twice: first to transmit mode to send an ACK and then back to receive mode to be able to receive further packets from other nodes.