MAC-aware Delay Optimization for Wireless Sensor Network Gateway with a microcontroller

H. Murat Gürsu*, Samuele Zoppi*, Eriza Fazli†, Wolfgang Kellerer*
*Chair of Communication Networks, Technical University of Munich, Germany
†Zodiac Inflight Innovations, Wessling, Germany
Email:*murat.guersu@tum.de, †eriza.fazli@zii.aero

Fig. 1: Different computation units on a typical sensor device. Transceiver does wireless reception while the USB controller handle forwarding of the packet where both have to share the microcontroller resources.

***Abstract*—Wireless sensor network devices are widely used for home automation and monitoring services. With low power consumption and low cost, IEEE 802.15.4 is highly attractive and accepted as the de facto standard wireless sensor networks. In order to benefit from the low cost similar chips are used also for the gateway. However, as the gateway has to forward all the accumulated packets this poses a resource sharing problem for the microcontroller between wireless tasks and forwarding tasks. In this work, we show this bottleneck with measurements. Then, we formulate the resource sharing as an optimization problem and solve it with an algorithm that has low complexity, fit for a microcontroller. Finally, we validate the analytical optimization with measurement and simulations. This simple optimization can result in a reduction of delay at level of seconds.**

I. Introduction

With home automation systems, the IEEE 802.15.4 have become the de-facto standard for long-term sensing applications such as heat management, light control and many more. Home automation use sensors all over the house to collect data and present it to customers through servers in the cloud [1]. This requires a wireless sensor network gateway that receives data through a wireless interface and forwards it to do wired world. In most of the cases, the gateway devices use Ethernet or USB communication for the forwarding task. In order to limit the costs, the gateways mostly has a single microcontroller. However, the microcontroller can either do wired or wireless communication as illustrated in Fig. 1, such that processing resources should be shared. Currently the use of the microcontroller resources is decided on an independent perspective and this results in unnecessary delays.

In this work we show that joint considerations of wireless delay and forwarding delay results in an improved delay performance. We formulate it as an optimization problem and solve it with lightweight algorithms that can be run on microcontrollers. We show the validity of the results through measurements and extend it with simulations. The results show that a delay reduction at the level of seconds is possible.

There has already been interest in optimizing the gateway delay problem [2]. However, the amount of work focusing on evaluating and improving delay for wireless sensor network gateways is limited. A high speed ethernet integration to the gateways for higher forwarding speed for industrial applications is investigated in [3], but for low cost devices this is not an option. The placement of the gateway for latency reduction is formulated as a facility placement problem in [4]. In home automation systems as the home user is setting the gateway, optimizing the placement is not always an option. A combined wireless-forwarding delay optimization for sharing of computation resources is not investigated and that forms the contribution of our work.

Our contributions can be summarized as:

1) Formulation of the joint MAC-forwarding gateway delay.
2) A lightweight solution fit for microcontrollers.
3) Validate of the analysis through simulations and measurements.

The paper is organized as follows: In Section II, scenario for the delay evaluation is described. In Section III, the delay model is evaluated against simulations and measurements. A solution is shared in Section V. The paper is concluded with a summary in Section VI.

II. Scenario

As we are focusing on the joint wireless-forwarding delay we are interested only in the last-hop behavior of any wireless sensor network. Thus, we consider a star topology composed of sensor nodes and a gateway. Our aim is to optimize the delay from sensor to cloud hence we consider an uplink only traffic. But the optimization can also be used for downlink without any changes. A

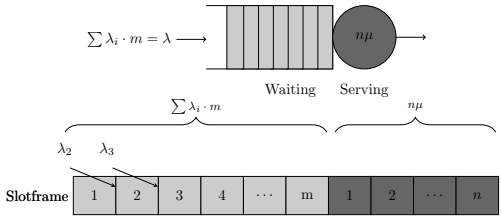


Fig. 2: The slotframe queue model, with a constant serving time of $n\mu$ and and arrival of $\lambda = \sum \lambda_i m$.

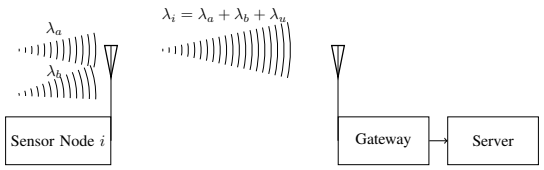


Fig. 3: At the last hop of the wireless sensor network, the traffic from previous hops are aggregated and forwarded to the gateway. Following, the gateway forwards the packets to the server. This defines the evaluation scope of this paper.

joint uplink downlink or round-trip optimization is left for future work.

Sensor nodes forwards their packets through wireless connection to the gateway. And the gateway forwards the packets through a wired connection to a server. The task of forwarding the packet from the gateway to the server is referred as gateway-forwarding.

A. Physical Layer

We assume a Rayleigh block fading channel with constant packet size that can be modelled as a Bernoulli random variable p that represents the success probability, as shown with measurements in an aircraft scenario [5]. Such a parameter should be adjusted with respect to the distance between devices and the transmission power. We use a fixed transmission power of 1 mW and no coding. A cyclic redundancy check is used for error detection.

B. Medium Access Layer

A well established technique for end-to-end delay guarantees in WSN networks is network-wide synchronization for coordination [6]. Accordingly, we consider a time-slotted MAC of IEEE 802.15.4 Time-Slotted Channel Hopping (TSCH) [7].

The use of resources is decided on a schedule called slotframe. Slotframe is a repeating pattern of multiple slots. Each slot can be allocated to a user or a task. As the resources of the microcontroller is shared, the network has to be aware when the gateway is ready to receive wireless packets and when not. This is done via allocating some of the slots to wireless transmission and some of the slots to gateway-forwarding. There are m slots for wireless transmission of m sensor nodes and n slots for the gateway-forwarding. Thus, a slotframe is in total $m + n$ slots. As each wireless packet arriving at any of the m slots has to be forwarded with any of the n slots before the next slotframe begins. Otherwise, queued until next n slots begin. The slotframe is illustrated in Fig. 2.

C. Firmware

We use OpenWSN [8] as the firmware for gateway and the sensor node. OpenWSN[1] is a firmware for a complete communication stack that serves as the de-facto open source project for WSN embeddings of 6TiSCH. The resource sharing for the microcontroller is reflected also in the OpenWSN firmware and serial slots are a part of the wireless schedule (slotframe).

[1]It has been awarded the Google IoT prize in 2016.

D. Hardware

We use Zolertia 1 (Z1) [9] for our measurements. Z1 motes are equipped with CC2420 transceiver chip for PHY layer and a separate micro-controller MSP430 to run the state machine of the communication stack. Same hardware is used for gateway and sensor nodes in order to reflect the low cost capability of a gateway.

E. Traffic

Each sensor node has an arrival rate with a Poisson mean of λ_i in units of packets per slotframe. The arrival rate represents the uplink packets the sensor node has to forward per slotframe and is the aggregated traffic at the last hop as illustrated in Fig. 3. The arrival rate λ_i of node i represents the total of the traffic it is forwarding from other devices e.g., $(\lambda_a + \lambda_b)$ and its own traffic (λ_u).

There are m wireless transmission slots, we consider a scenario where all users have their specific Poisson arrival rate that can be used to calculate the total arrival rate per slotframe. The overlap of multiple independent Poisson processes can be represented with a new Poisson mean as the summation of all Poisson means. The channel success probability can be used to convert transmitted packet rate to received packet rate as in

$$\lambda := \sum_{i=1}^{m} \lambda_i \cdot p. \tag{1}$$

III. Average Delay Analysis

The maximum payload of a wireless packet allowed in IEEE 802.15.4 is 133 bytes. The reception of a wireless packet with the maximum payload takes around 4 ms and 3 ms for forwarding with USB 2.0. Thus, when we are optimizing on the level of ms's, the MAC delay, wired or wireless, is the main delay contributor and processing delay that is below ms values can be neglected.

Parameter	Explanation
T_{slot}	Time slot duration in a slot frame
T_w	Wireless Transmission time
R_w	Wireless data rate
R_b	Serial data rate
T_{init}	Average waiting time before first transmission
T_{retx}	Average waiting time due to re-transmissions
T_{g-f}	Average gateway-forwarding waiting time
m	Number of wireless transmission slots
n	Number of serial transmission slots
p	Receiver based Bern. reception probability
α_1	Initial waiting coefficient
α_2	Re-transmission waiting coefficient
μ	Packets forwarded through serial per slot
ρ	Utilization of the gateway forwarding
λ_i	Arrival rate to slot i
λ_0	A fixed arrival rate for all slots
λ	Arrival rate to slotframe
$T(n)$	MAC and gateway related delay
$\Delta T(n)$	Derivative of function $T(n)$

Tab. I: Summary of variables used for explanation of gateway related wireless sensor network delay.

A. Average MAC Delay

The application and the MAC timer are independent from each other. The sensor node can transmit a single packet per slotframe. With a slotframe length $m+n$ slots the average initial waiting time T_{init} for a packet is

$$T_{init} = \frac{m+n}{2}. \tag{2}$$

T_{init} is the uniform expectation of the possibility to wait a full slotframe, i.e., $m+n$ slots and not waiting at all i.e., 0. This covers, the application to MAC delay.

In lossy world of the wireless medium access, re-transmissions are expected. Each packet that can not pass the error detection is not acknowledged thus re-transmitted. And has to wait a full slotframe i.e., $m+n$ slots. In order to model the delay added due to the re-transmissions we use the success probability p as in

$$T_{retx} = \frac{(1-p)(m+n)}{p}. \tag{3}$$

With a perfect channel, i.e., $p = 1$ we have no re-transmission delay.

The summation of these two delay contributors is the average MAC delay

$$T_{mac} = T_{init} + T_{retx} = \frac{(2-p)(m+n)}{2p}. \tag{4}$$

Thus, the packet has reached the gateway and we want to further investigate the gateway-forwarding delay.

B. Gateway-Forwarding Delay

We adopt a queuing model to calculate the gateway-forwarding delay with an $M/D/1$ queue on a slotframe basis as summarized in Fig. 2. We assume Markovian arrivals with λ arrivals per slotframe. And h serving time per arrival. We use this model to calculate the average gateway-forwarding delay.

Parameter	Value
T_{slot}	10 ms
T_w	5 ms
R_w	250 kbit/s
R_b	115200 baud/s
r	0.8 bit/baud

Tab. II: Parameter settings for a typical Wireless Sensor Network with OpenWSN and Z1

The queuing model is represented in Fig. 2. In the queueing model there are λ_i packets per slotframe from each m sensor node. This results in a with a total Poisson mean arrival λ packets per slotframe.

The gateway can forward μ packets per slot thus, in total it has a capacity to serve $n\mu$ packets per slotframe. The forwarding rate of a packet is deterministic as we have the same packet size for all sensor in the network. We calculate the forwarding rate μ in units of packets per slot that is

$$\mu = \frac{T_{slot} \cdot R_b \cdot r}{T_w \cdot R_w}, \tag{5}$$

where R_b and R_w are the serial baudrate and wireless rate in kbits/s. r is the conversion rate from baud to bits. The T_{slot} is the duration of a serial slot and T_w is the wireless transmission duration. In TSCH T_{slot} is same for all slots in a slotframe and motes do not use the whole duration but only a chunk of it for wireless transmission which is given as T_w here. These are not the same due to different processing time for both[2]. As there are n slots, the serving time per slotframe is $n \cdot \mu$. This results in a serving time of $h = \frac{1}{n \cdot \mu}$ per slotframe. The values of the variables are summarized in Tab. II.

Through this we can calculate the utilization as

$$\rho = \frac{\lambda}{n\mu}. \tag{6}$$

The average queuing delay in units of slotframe for an $M/D/1$ queue can be calculated as

$$\frac{\rho h}{2(1-\rho)}. \tag{7}$$

As we based our model on units of slotframe we have to scale it back to slots. Thus we multiply the delay with the number of slots per slotframe i.e., $(m+n)$. That represents the gateway forwarding delay T_{g-f} in terms of slots as,

$$T_{g-f} = \frac{(m+n)}{2n\mu}\left(\frac{\lambda}{n\mu - \lambda}\right). \tag{8}$$

It is known that we have an unstable queue with a utilization greater than 1. Thus the minimum number of serial slots n for a stable gateway-forwarding delay is

$$n > \frac{\lambda}{\mu}. \tag{9}$$

[2]Usually T_w is almost half of T_{slot}.

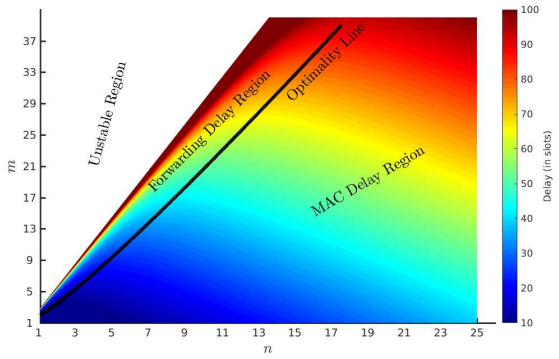


Fig. 4: The delay is given as a heatmap with varying number of serial and wireless slots. The arrival rate is fixed to $\lambda_0 = 0.5$ and the success probability is $p = 0.5$. The contribution of MAC and forwarding delay can be depicted with distinct regions and also the unstable region is given separately.

The combined delay from wireless Eq. (4) and gateway-forwarding delay (8) is

$$T(n) = (m+n)\left(\frac{2-p}{2p} + \frac{\lambda}{2n^2\mu^2 - 2\lambda n\mu}\right). \quad (10)$$

With increasing n, the combined delay $T(n)$ is in one part linearly increasing and on the other part decreasing as seen from Eq. (4). Intuitively, we expect to observe a concave behavior with varying n. Thus, there is an optimal n that minimizes the delay.

IV. Evaluation

In this section, we observe the behavior of the total delay. Initially we plot Eq. (4) with varying m and n. Later on we validate the analysis with measurements. We extend the results with simulations to easily control the loss probability p and investigate its effects.

For the ease of evaluation we assume that the arrival rate is fixed for each node to λ_0 and we will not use λ_i. Hence, the total arrival rate per slotframe is $\lambda = m \cdot \lambda_0 \cdot p$.

A. Analytical Results

In Fig. 4 we have plotted $T(n)$ with varying m and n. In general, there are three important regions. First, the unstable region where delay reaches infinite values due to insufficient number of serial slots per slotframe. Secondly, the forwarding delay region where the delay is high due to insufficient number of serial slots. And lastly, the MAC delay region where the forwarding delay is negligible compared to MAC layer due to initial waiting time and re-transmissions. This shows that the behavior is indeed concave with varying n as expected from Eq. (10). Thus, optimizing n for lower average combined delay is validated.

B. Measurements

We have implemented the gateway forwarding measurements using Z1 nodes and OpenWSN firmware. We have generated packets in the sensor and measured gateway-forwarding with no packet loss mimicking the scenario of $p = 1$. The number of wireless slots m are set to $\{4, 6, 8, 10\}$ and the number of serial slots are varied. [3]

C. Comparison to Measurements

We investigate the behavior of the delay (y-axis) with varying number of serial slots n (x-axis) comparing Eq. 10 with measurements. Fig. 5 shows that delay increases with low and high number of serial slots and the optimal n varies for different scenarios. In Fig. 5a, measurement results differ slightly from analysis and simulations for increasing m. This is due to the simplification in the model. We have assumed that arrivals are per slotframe basis and can be solved as a slotframe queue. On the other hand, for each slotframe, each packet has to wait for an average of $m/2$ wireless slots until the serial slots start. In Fig. 5b this added value is subtracted and we see that for both m values we observe a perfect match. The in-slotframe waiting time can be solved with adding each serial slot directly after the wireless slot, instead of grouping them together as illustrated in Fig. 2.

D. Implementation-Oriented Simulations

The simulations are performed in a discrete event based simulator developed in MATLAB. The traffic is generated per user per slotframe on a Poisson distribution basis. However, the constraint of maximum one packet transmission per slot is imposed so that not more than one packet is transmitted per slotframe per user. This is done to observe the validity of the summation of Poisson arrivals for multiple slots in a slotframe Reception of more than m packets per slotframe is not possible. But Poisson distribution is not truncated thus, this limittion is not represented in the analysis. And we hope to investigate its effect through simulations.

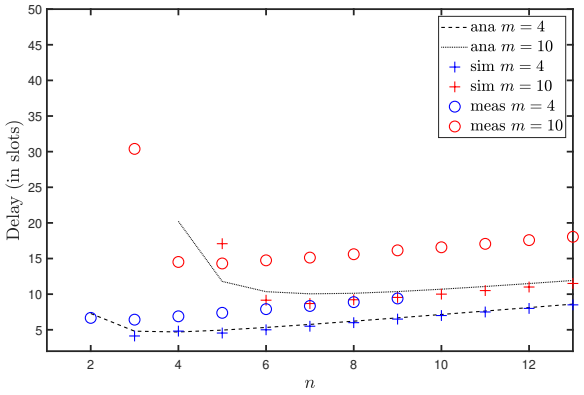
The delay in the simulator is more granular compared to analysis. The average waiting time for forwarding in the analysis is calculated with step size of one slotframe[4], while the delay on simulations are calculated with units of slots.

E. Comparison to Simulations

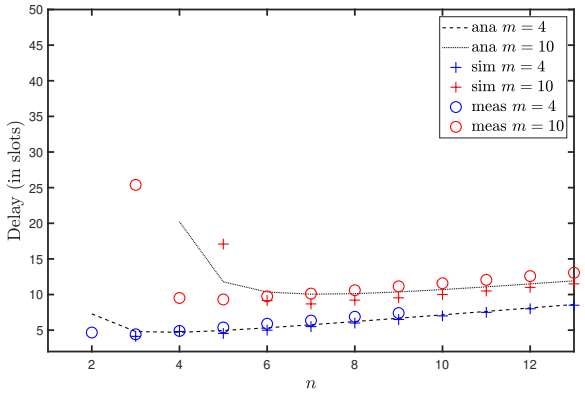
As we have validated the forwarding delay through measurements, we extend our results with simulations. We further relax the assumption of perfect channel conditions to investigate the effect of re-transmissions. In Fig.6 analytical results are compared against simulation results. We see that with $\lambda_0 = 0.25$ there is a close match and with increased arrival rate of $\lambda_0 = 1$ the fit diverges. This is due to maximum one packet received per slot

[3] We plan to make the measurement results available on GitHub for the camera ready version of the paper.

[4] If a user is not served in one slotframe, it has to wait for the other slotframe. However, there is also in-slotframe forwarding delay which is not taken into account in analysis. For instance if the packet received at slot 1 is served at slot 13, there is a 12 slot waiting delay, which is not represented by the analytical evaluation.



(a) $\lambda_0 = 0.25$ and $p = 1$.



(b) $\lambda_0 = 0.25$ and $p = 1$ with compensating for the in slot waiting time until serial slots.

Fig. 5: The average delay $T(n)$ varying with number of serial slots allocated and the optimal value. Number of users in the last-hop is varied as $m = [4, 6, 8, 10]$. Simulation results are averaged over 10000 trials.

Algorithm 1 Gradient descent algorithm for the calculation of the optimal number of allocated serial slots for minimum gateway forwarding delay. ϵ is the termination tolerance for ΔT, γ is the step size (learning rate) and I is the maximum number of iterations.

Input: $\{m, \mu, \lambda, p\}$
Output: n

$i = 0$
$n_0 = \left\lceil \frac{\lambda}{\mu} \right\rceil + 2$ ▷ Initial guess
while $\Delta T(n_i) > \epsilon$ **and** $i < I$ **do**
    $i = i + 1$
    $n_i = n_{i-1} + \gamma \Delta T(n_i)$
end while
return n

which is overruled with a Poisson arrival assumption. Even though the trend is similar, the difference between the analytical and simulated values increases.

Also, we see that the impact of adding serial slots is a lot more severe. The more lossy the channel is, the closer we are to the stability limit for fixed number of serial slots. This is indeed an interesting outcome and explains why providing low number of serial slots has been working as expected for lossy wireless sensor network setups [5].

The shared optimization algorithm with the underlying delay equation can then be implemented in the gateway to optimally allocate the required number of serial slots n. This update can be included in the schedule and distributed in the network over beacons, which can be done hourly or daily depending on the variations on the traffic. We are currently working on such a deployment and doing in-depth analysis with real world measurements.

V. Optimization

The optimization problem for the allocation of the number of serial slots with minimum combined wireless-forwarding delay $T(n)$ can be formulated as,

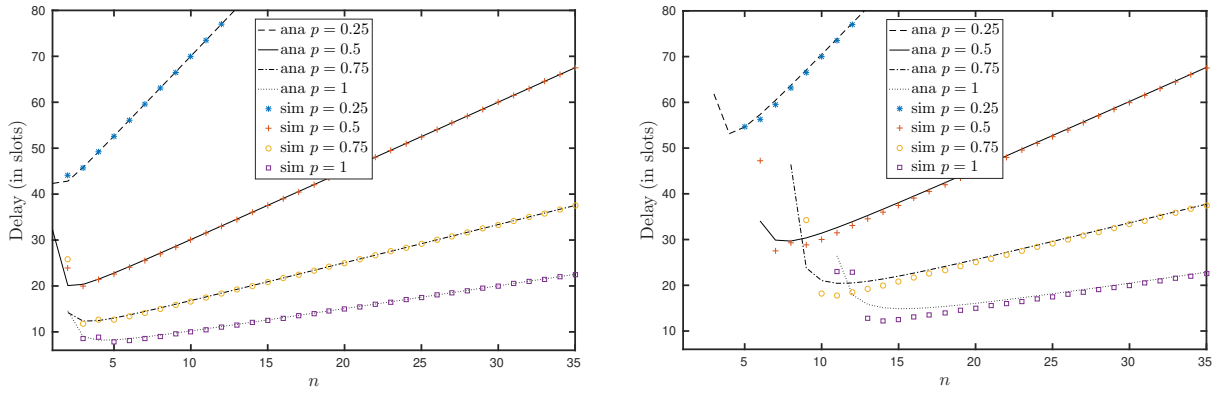
$$\begin{aligned} \underset{n}{\text{minimize}} \quad & T(n) \\ \text{subject to} \quad & n \geq \left\lceil \frac{\lambda}{\mu} \right\rceil . \end{aligned} \tag{11}$$

The problem at hand is polynomial and a closed form solution can be calculated. However, the root is quite complex to calculate. It is possible to show that $T(n)$ is convex in the region of interest, i.e., $n > \frac{\lambda}{\mu}$. However, this is not done here due to space limitations. Any convex optimization can be used for the selection of n implemented in the gateway.

We have implemented a gradient descent algorithm to solve this optimization problem, and we present its pseudo-code in Alg. 1. The algorithm requires the knowledge of λ, p, μ and m. λ and p can be easily calculated in the gateway implementing specific counters. μ is a constant value which depends on the serial port implementation and known to the gateway, and m is known to the gateway as it is allocated by the network wireless resource allocation mechanism. Afterwards, before starting its iterations, the algorithm selects its initial guess as $\left\lceil \frac{\lambda}{\mu} \right\rceil + 2$ as delay exponentially falls after the stability limit. Then, the algorithm iterates over different steps of the derivative of $T(n)$ and terminates when it reaches a sufficient precision ϵ or a maximum number of iterations I.

We would like to point out that, although the calculation of the derivative $\Delta T(n)$ is not a simple operation, it is simpler compared to the calculation of the root.

Additionally, we evaluate the performances of our implementation of the gradient descent algorithm. The results are summarized in Table III. The evaluation is performed for different values of p as it highly affects the shape of $T(n)$ as shown in Fig. 6. For every p, the average number of iterations and the average optimality gap are calculated for linearly spaced values of m between 1 and m_{max}. The values of m_{max} and ϵ have been selected experimentally for different values of p. The



(a) $\lambda_0 = 0.25$.

(b) $\lambda_0 = 1$.

Fig. 6: The average delay $T(n)$ varying with number of serial slots allocated and the optimal value. The success probability in the last-hop is varied as $p = [0.25, 0.5, 0.75, 1]$ and number of users is fixed to $m = 6$. Simulation results are averaged over 10000 trials.

p	$m_{\max}$	ϵ	avg. iterations	avg. optimality gap
0.25	48	0.939	4.60	0.389
0.50	53	1.171	1.40	0.109
0.75	64	0.707	1.60	0.060
1.00	80	0.350	3.40	0.036

Tab. III: Performance of the gradient descent algorithm for different values of packet success probability p. Learning rate $\gamma = 0.1$. For every p, the average number of iterations and the average optimality gap are calculated for linearly spaced values of m between 1 and $m_{\max}$. The values of $m_{\max}$ and ϵ have been selected experimentally for the different values of p.

table shows that the implemented algorithm converges to optimality with a very low number of iterations. In facts, for the evaluated scenarios, the difference to the optimal delay is always lower than one time slot, and the number of iterations is maximum 14. These results show that the proposed implementation is compatible with the constrained computational resources of a WSN device.

VI. Conclusion

In this paper we have analysed the wireless sensor to wired delay for a gateway in Wireless Sensor Networks where we base our analysis on the off-the-shelf sensor devices. We have validated our analysis through measurements and simulations. The analysis is used to formulate an optimization problem that is solved with a lightweight algorithm fit for microcontrollers. Through this analysis our main contribution is providing the ideal scheduling of the microcontroller resources between wireless and wired communication for optimal total delay.

Future work can investigate possible hardware improvement, i.e., multiprocessors and multiradio for the gateway, that can improve the delay of gateway-forwarding. These solutions can have limitations again due to their effect on MAC delays like varying processing times for wireless and wired communication.

Furthermore, another possible extension is integrating the downlink and providing a model for the round trip. As the forwarding from the server to the gateway has to be also taken into account it can result in a different optimal point.

References

[1] C. O. Rolim, F. L. Koch, C. B. Westphall, J. Werner, A. Fracalossi, and G. S. Salvador, "A cloud computing solution for patient's data collection in health care institutions," in *eHealth, Telemedicine, and Social Medicine, 2010. ETELEMED'10. Second International Conference on*, pp. 95–99, IEEE, 2010.

[2] Y.-C. Li, S. H. Hong, X. Huang, G. Chen, and X. Liang, "Implementation of a powerlink-wirelesshart gateway for industrial automation," in *Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON), 2016 13th International Conference on*, pp. 1–6, IEEE, 2016.

[3] J. von Hoyningen-Huene, A. Mueller, S. Dietrich, and G. May, "Comparison of wireless gateway concepts for industrial real-time-communication," in *Emerging Technologies and Factory Automation (ETFA), 2016 IEEE 21st International Conference on*, pp. 1–4, IEEE, 2016.

[4] J. L. Wong, R. Jafari, and M. Potkonjak, "Gateway placement for latency and energy efficient data aggregation [wireless sensor networks]," in *Local Computer Networks, 2004. 29th Annual IEEE International Conference on*, pp. 490–497, IEEE, 2004.

[5] M. Gürsu, M. Vilgelm, S. Zoppi, and W. Kellerer, "Reliable coexistence of 802.15. 4e tsch-based wsn and wi-fi in an aircraft cabin," in *Communications Workshops (ICC), 2016 IEEE International Conference on*, pp. 663–668, IEEE, 2016.

[6] K. Pister and L. Doherty, "Tsmp: Time synchronized mesh protocol," *IASTED Distributed Sensor Networks*, pp. 391–398.

[7] "IEEE Standard for Local and metropolitan area networks–Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC sublayer," *IEEE Std 802.15.4e*.

[8] T. Watteyne, X. Vilajosana, B. Kerkez, F. Chraim, K. Weekly, Q. Wang, S. Glaser, and K. Pister, "Openwsn: a standards-based low-power wireless development environment," *Transactions on Emerging Telecommunications Technologies*, vol. 23, no. 5, pp. 480–493, 2012.

[9] Zolertia, *Z1 Datasheet*, 1 2018. RevC.