Analysis of Low Latency TSCH Networks for Physical Event Detection

Alex Yang, Arvind Sundararajan, Craig B. Schindler, and Kristofer S. J. Pister
Berkeley Sensor & Actuator Center
Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
Berkeley, California, USA
Email: {yangalex, arvinds, craig.schindler, ksjp}@berkeley.edu

***Abstract*—Timeslotted Channel Hopping (TSCH) is a mode of the IEEE802.15.4 standard for low-power wireless sensor networks. We discuss the variation in performance of a 6TiSCH network implemented using OpenWSN, an open-source implementation of current low-power wireless sensor networking standards, by measuring end-to-end packet latency — the time between an event trigger signaling the transmitter node's microprocessor to create a packet, and the packet reception on the receiving node — with 0.5 millisecond accuracy. In TSCH networks, time is divided into repeated chunks known as slotframes, which are further divided into timeslots. We explore the effect the number of available transmission slots has on packet end-to-end latency for an 11 slot 6TiSCH network with 1, 3, 5, 8, and 11 active slots. Results are reported for a setup with one transmitter and one receiver.**

I. Introduction

Wireless sensor networks are pervasive in industrial process monitoring. Sensor nodes, or "motes," can measure processing variables such as temperature, pressure, humidity, light intensity, etc. and communicate wirelessly to execute and monitor complex processes. Low-power wireless sensor networks provide a cheaper, more robust alternative to comparable wired solutions, for which the cost of installation and maintenance is much greater. Although current state of the art low-power wireless sensor networks offer excellent reliability (>99.999%)[1], they make no hard guarantees on latency. Real time decision making often requires tight control loops and small latency between an event trigger and a corresponding action.

Initial research has shown the applicability of low-power wireless networks for industrial process control (discussed in Section III). However, exploring the tradeoffs of varying networking architecture and topology is still an open area of research.

In this paper we explore the effect of varying the radio duty cycle (by changing the number of TX/RX opportunities) on the end-to-end latency of packets sent through an IEEE 802.15.4 TSCH network [2] using OpenWSN [3] and 6TiSCH [4], [5]. The end-to-end latency is measured as the time difference between a "button press" event trigger to the transmitter mote, and a "packet received" event from the receiver mote(s) shown in Fig. 1.

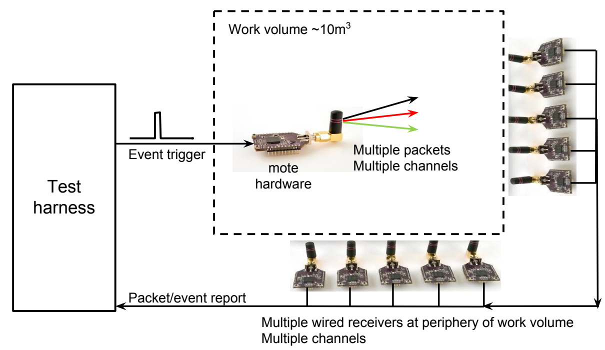

Fig. 1. An industrial workcell with multiple transmitting motes and multiple receiving motes. The test harness controls when packets are sent by sending an event trigger to the transmitting mote. Packets are sent on a specified channel to receiving motes. The receiving signal is reported back to the test harness and the next trigger occurs.

II. Background

First introduced in "Tsmp: Time synchronized mesh protocol" [6], the principle of Timeslotted Channel Hopping (TSCH) provides the basis for the MAC layer definition in the IEEE 802.15.4e standard [2].

In a TSCH network, all motes tightly synchronize their clocks and specify exactly when packet transmissions should occur. This allows the radios to turn on only when required to, and remain off otherwise. The radio draws the most power on the motes [7], and lowering the radio duty cycle thus improves individual mote lifetime and the energy efficiency of the whole network.

Furthermore, TSCH increases reliability in wireless sensor networks. Motes communicate with each other over different radio frequency channels, and re-transmit packets on different channels if previous attempts fail. Channel hopping in this manner reduces multipath fading and external radio interference, thereby increasing reliability [8].

A. Slotframes and schedules

In TSCH, each mote in the network synchronizes using a known slotframe structure. The slotframe is a repeating unit of time subdivided into slots. Each slot is just long enough

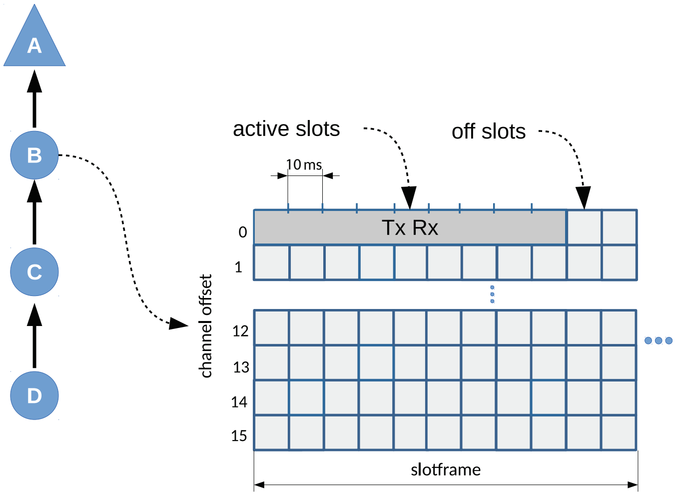

Fig. 2. The 6TiSCH minimal schedule, shown here with 9 active slots and an 11 slot slotframe. Channel offset is used as an index into a pseudo-random channel hopping pattern.

for a mote to transmit a radio packet and receive a packet reception acknowledgment from the recipient. Within each slot of a TSCH network, a mote either 1) transmits data, 2) receives data, 3) listens idly (doesn't receive anything, but remains awake and listening), or 4) keeps its radio off [4], [5]. During an "active slot" a mote will transmit data if it needs to, and listen otherwise. During an off slot a mote will keep its radio off. In the minimal 6TiSCH configuration, the number of active slots and off slots in the schedule is a tunable parameter, and all active slots are shared TX/RX slots, i.e. a pair of motes can both transmit or receive. Fig. 2 shows a schedule with a minimal 6TiSCH configuration, containing eleven 10 ms slots per slotframe and nine active slots.

For a TSCH network, the absolute slot number (ASN) is a counter initialized to zero when the network is formed, and incremented with every slot (10 ms). Formally,

$$\text{ASN} = kS + t \tag{1}$$

where k is the number of slotframe repetitions since the network was formed, S is the number of slots in the slotframe, and t is the slot offset of the current slotframe. Each mote in the network synchronizes to the same ASN. The number of slots in the slotframe determines the radio duty cycle; more active slots mean more opportunities to send and receive packets (lower end-to-end latency), but greater mote power consumption. A model to compute the amount of energy a mote consumes per slot of a TSCH network has been studied [7], [9].

B. Channel hopping

Channel hopping is implemented by IEEE802.15.4 TSCH within the active slots of the slotframe. In the network schedule, each packet transmission/reception is specified by a slotOffset (the slot within the slotframe when transmission will occur) and a channelOffset (the channel over which the packet is transmitted). Each active slot is pseudorandomly assigned a channel on which to transmit from the total number of channels available [10], [11]. The frequency f is determined by the channelOffset, ASN, and total number of frequency channels numFreq as per the following scheme:

$$f = F\{\text{mod}((\text{ASN} + \text{channelOffset}), \text{numFreq})\} \tag{2}$$

where $\text{mod}(a, b)$ is defined as a modulo b and F is a table mapping an integer to a frequency. The number of slots in the slotframe, 11, is relatively prime to the number of possible channels in an IEEE802.15.4e network, 16. This ensures that 1) a new frequency is tried for the same slotOffset on successive slotframes, and 2) a new frequency is tried for different active slots in the same slotframe.

If a mote transmits a packet in a shared TX/RX slot and fails to receive an acknowledgment, it increments a backoff exponent BE and waits a random number of active slots between $[0, 2^{BE}]$ before retransmitting over a different channel. This prevents transmission collisions between motes transmitting in the same slot.

III. Related Work

Chang et al. demonstrates an algorithm called the Low Latency Scheduling Function which can be used to daisy chain timeslots together in a 6TiSCH network, resulting in lower latency across a multihop network [12]. An average end-to-end latency of 320 ms across a five hop 6TiSCH network is demonstrated.

ABB's WISA platform, designed specifically for factory automation, implements the IEEE 802.15.1 standards for the physical (PHY) networking layer and shows sub-10 ms latencies with 2-ms repeating frame cycles [13]. Using low-power IEEE 802.15.4 wireless networks for industrial process control has also been investigated [14].

IV. Data Collection

A. Experiment Logic

In order to examine the relationship between the number of active slots and end-to-end latency, we measure the end-to-end latency of 10,000 packets in 6TiSCH networks with 1, 3, 5, 8, and 11 active slots in each slotframe. The experiments utilize one transmitter mote and one receiver mote. In each experiment, the slotframe consists of 11 total slots of 10 ms each, with all active slots grouped consecutively at the start of the slotframe.

In each experiment, the following sequence of events repeats for every packet:

1) The test harness waits for a random amount of time, uniformly distributed between 0 and 110 ms, to simulate an asynchronous event trigger within the slotframe.
2) The test harness outputs a digital signal to a GPIO pin on the transmitter mote, simulating the button press/event trigger.
3) The transmitter mote enters interrupt mode upon the rising edge of the above digital signal. The interrupt handler invokes a custom firmware application which creates a packet to be transmitted.

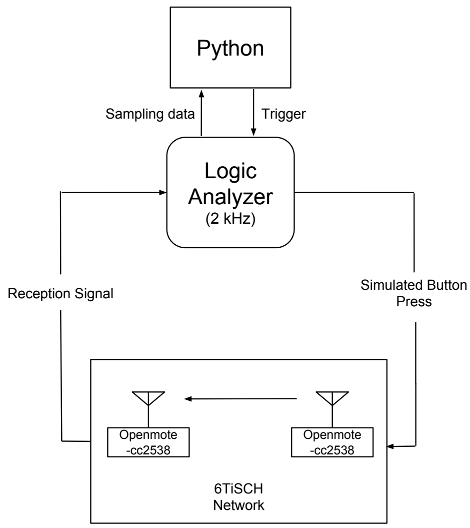

Fig. 3. Block diagram of the experimental setup with a single transmitting mote and a single receiving mote.

4) The created packet is transmitted by the transmitter mote's radio at the next available active slot.
5) The packet is received by the receiver mote's radio. If the packet was intended for the receiver mote, the packet is forwarded to the application. If the packet was not received, the transmitter mote retries transmission up to 5 times until it is received.
6) Upon receiving the packet in its custom application, the receiver mote toggles a GPIO pin to indicate successful reception.
7) The test harness senses the reception signal and measures the end-to-end latency – the time between button press event and packet reception event.
8) If there is no packet reception signal within 2 seconds, the test harness records a missed packet.

B. Experiment Implementation

Fig. 3 shows a block diagram of the implemented test harness and 6TiSCH network. The Digilent Analog Discovery 2 [15], a multifunction instrument for generating and sampling signals, functions as the test harness. We use the onboard digital I/O module to output the event trigger digital signal, and the logic analyzer to sample mote GPIOs at 2 kHz, thereby providing 0.5 ms resolution. We use a custom Python script to interface with the Analog Discovery 2 and automate the test harness logic.

We use the OpenMote-CC2538 [9], a hardware prototyping ecosystem designed for IoT, as the motes in the 6TiSCH network. The OpenMote boards are programmed to run OpenWSN [3], an open-source networking protocol stack implementing IEEE 802.15.4 TSCH [4], [5] in addition to upper-stack IETF standards: 6LoWPAN [16], [17], RPL [18], and CoAP [19]. The OpenWSN stack layers are shown in Fig. 4.

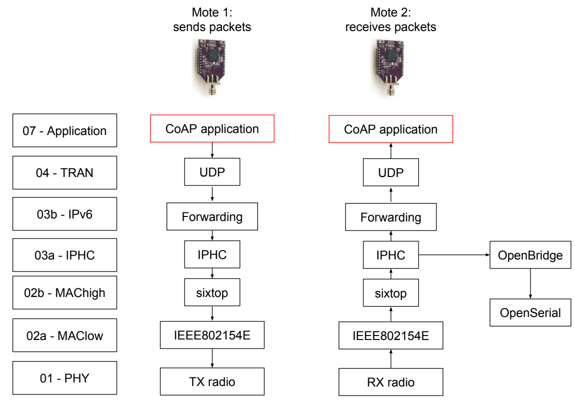

Fig. 4. A block diagram of the OpenWSN network stack. A CoAP application on the transmitter mote creates and sends a packet upon receiving a trigger from the Analog Discovery 2. A CoAP application on the receiver mote toggles one of its GPIO pins when it receives the packet.

An OpenMote-CC2538 consumes 34mA when transmitting at 7dBm, and consumes 20mA when receiving at -50dBm. A 10 ms active slot in which a mote transmits a packet consists of transmitting for 4ms and listening for 1ms to receive the acknowledgment [9].

C. Simulation

We construct a simple model and algorithm in MATLAB to generate data from a simulated experiment that follows the same rules from IV-A, with the exception that there is no limit to the number of transmission retries. We define the following variables:

- S: The number of slots in the slotframe.
- H: An array of length S.
- n: The number of active slots in the slotframe.
- l: The length of a single slot, in milliseconds.
- t: The slot offset.
- m: The shortest allowable end-to-end latency.
- Z: A random variable which takes an integer value from $U \sim [-(l - \text{round}(m)), \text{round}(m-1)]$. Z accounts for the sum of two effects: the fact that a real button press occurs randomly within the slotframe, drawn from $U \sim [0, Sl]$, and the fact that the smallest empirical end-to-end latency that can be seen is m.
- p: The probability a packet is received during a transmission attempt.
- w: The number of elapsed slots since a packet was created.

A simulated button press and packet sent consists of the following steps:

1) Initialize H to contain *True* in the first n entries of the array, and *False* in all other entries.
2) Initialize t by picking a random integer from $U \sim [1, S]$.
3) Initialize w to 1.
4) If the value of H at index t is *True*, initialize a floating point number x from $U \sim [0, 1]$

- If $x \geq p$, the packet has not been received. Increment both t and w by 1. If t is greater than S, then reinitialize t to 1. Jump to step 4.
- If $x < p$ the packet has been received. The end-to-end latency is calculated as $lw + Z$.

5) If the value of the array H at index t is *False*, increment both t and w by 1.
   - If t is greater than S, reinitialize t to 1.
   - Jump to step 4.

In our simulations, we set $p = 0.95$. The results of the simulations are presented in Section V.

V. RESULTS

The summary statistics of experimental data for 1, 3, 5, 8, and 11 active slots with 10,000 packets sent are shown in Fig. 5, and the latency histograms for the varying number of active slots are presented in Figs. 6,7,8,9, and 10. The histograms show that end-to-end latency is inversely related to the number of active slots. More packets have lower end-to-end latencies in schedules with more active slots. This is consistent with the idea that a transmitter can send a packet more quickly after packet creation with more active slots in the slotframe.

The observed minimum latencies for all active slots are < 10 ms; this occurs when the event triggers at the start of an active slot and the packet is successfully transmitted in that slot. No packets arrive faster than 6.5 ms because the minimum latency is limited by the radio tasks required in every active slot. Some values for maximum latency packets are greater than expected latency given a maximum of 5 retransmission attempts. This is due to 1) the next available active slot being used for network management packets instead of packets from our application, and 2) our simulation not implementing a backoff mechanism for failed transmissions.

The decreasing plateau heights in the figures show that the frequency of packet latencies decreases significantly after certain values. This is due to the distribution of active slots in the slotframe. For example, with 1 active slot, Fig. 6 features the first plateau from approximately 10 ms to 120 ms, a second plateau from approximately 120 ms to 230 ms, and so on. If a packet is transmitted in the first available active slot after the event trigger and is received on the first attempt, the packet latency will fall between approximately 10 ms and 120 ms. If the packet is received on the second transmission attempt, the packet latency will approximately fall between 120 ms and 230 ms, because a full slotframe elapses before the next available active slot. The data in the other histograms can be explained similarly for each active slot configuration. With more active slots, packet retransmissions occur sooner, causing the plateau ranges to decrease.

The complimentary cumulative distribution function (CCDF) plots for each number of active slots are presented in Fig. 11. Generally, the latency values in the experimental CCDF curves for each active slot are worse than the latencies in the corresponding simulated CCDF curves. This is due to the same reasons as described for the histograms. The CCDF curves also display periodic cusp-like features. Since the distribution of packets is uniform across certain latency ranges, the CCDF decreases linearly over that range. Plotting the CCDF with logarithmic axes thus produces the features shown in the figure.

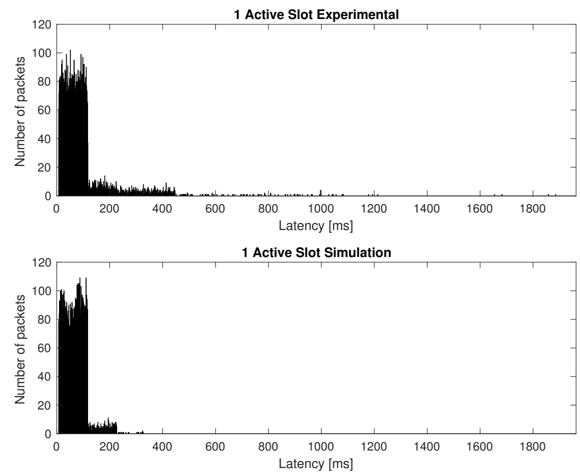

Fig. 6. Histogram of end-to-end latency for 10,000 packets sent with 1/11 active slots.

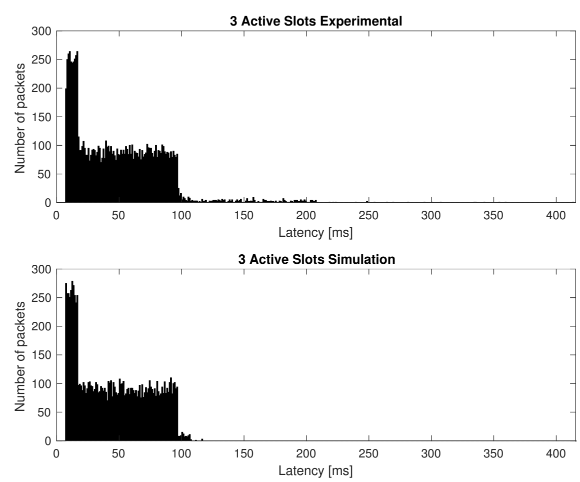

Fig. 7. Histogram of end-to-end latency for 10,000 packets sent with 3/11 active slots.

Latency Statistics

Active Slots	*Min.*	*Max.*	*Mean*	*Median*	*Std. Dev.*	*NumRx*	*NumTx*	*Reliability*
1/11 Exp.	7.0	1,962.5	93.9	71.0	109.3	9,996	10,000	99.96%
1/11 Sim.	7.0	326.0	67.7	66.0	40.9	10,000	10,000	100%
3/11 Exp.	7.0	415.5	49.0	44.5	35.6	10,000	10,000	100%
3/11 Sim.	7.0	116.0	45.1	42.0	28.6	10,000	10,000	100%
5/11 Exp.	7.0	345.0	32.4	24.5	22.9	10,000	10,000	100%
5/11 Sim.	7.0	94.0	31.4	23.0	22.1	10,000	10,000	100%
8/11 Exp.	6.5	80.0	17.9	14.0	11.1	10,000	10,000	100%
8/11 Sim.	7.0	71.0	17.6	14.0	10.9	10,000	10,000	100%
11/11 Exp.	7.0	47.0	12.3	12.0	3.7	10,000	10,000	100%
11/11 Sim.	7.0	35.0	11.9	12.0	3.7	10,000	10,000	100%

Fig. 5. Latency statistics for 10,000 packets sent with 1, 3, 5, 8, and 11 active slots. Both experimental and simulated data are shown. All latency statistics are reported in milliseconds. NumRx is the number of packets received, and NumTx is the number of packets transmitted. The reliability is calculated as NumRx / NumTx.

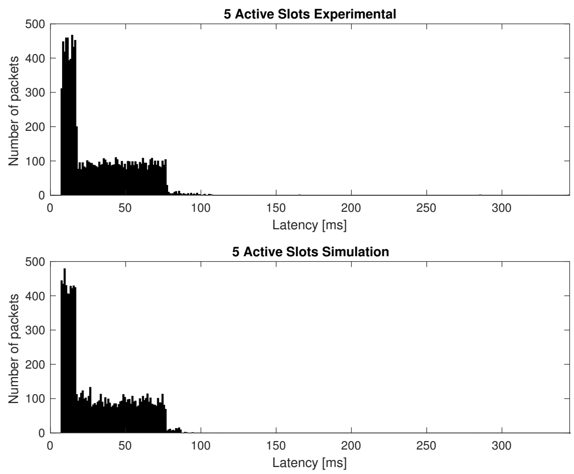

Fig. 8. Histogram of end-to-end latency for 10,000 packets sent with 5/11 active slots.

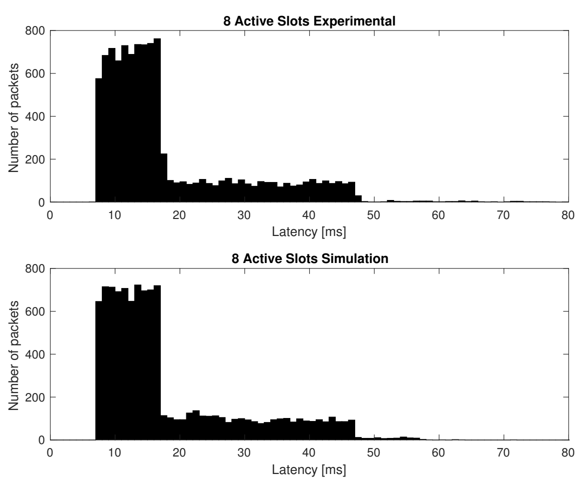

Fig. 9. Histogram of end-to-end latency for 10,000 packets sent with 8/11 active slots.

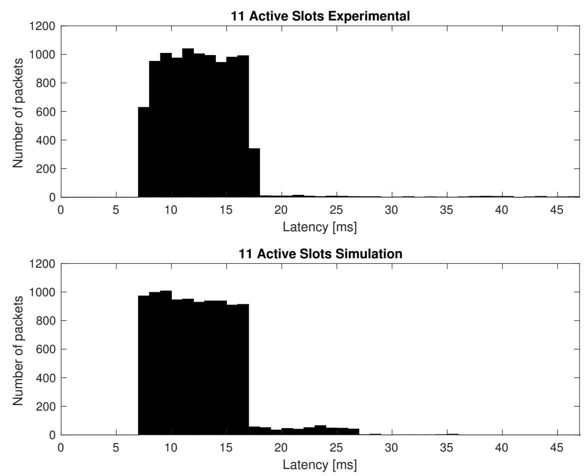

Fig. 10. Histogram of end-to-end latency for 10,000 packets sent with 11/11 active slots.

VI. Conclusion

The scope for low-latency event detection in low-power wireless sensor networks is vast. This paper provides end-to-end packet latency distributions for a one transmitter to one receiver topology within a 6TiSCH network. Additionally, the study performed in this paper provides a framework to conduct future studies with more complicated network topologies, as shown in Fig. 1. We conclude that it is possible to achieve low latency event detection with careful selection of the TSCH scheduling parameters. Characterizing the tradeoff between active slots (i.e. energy consumption) and end-to-end latency offers useful benchmarks for configuring real-world wireless sensor networks.

Acknowledgment

The authors would like to thank Azbil Corporation and the Berkeley Sensor & Actuator Center for supporting and funding this project.

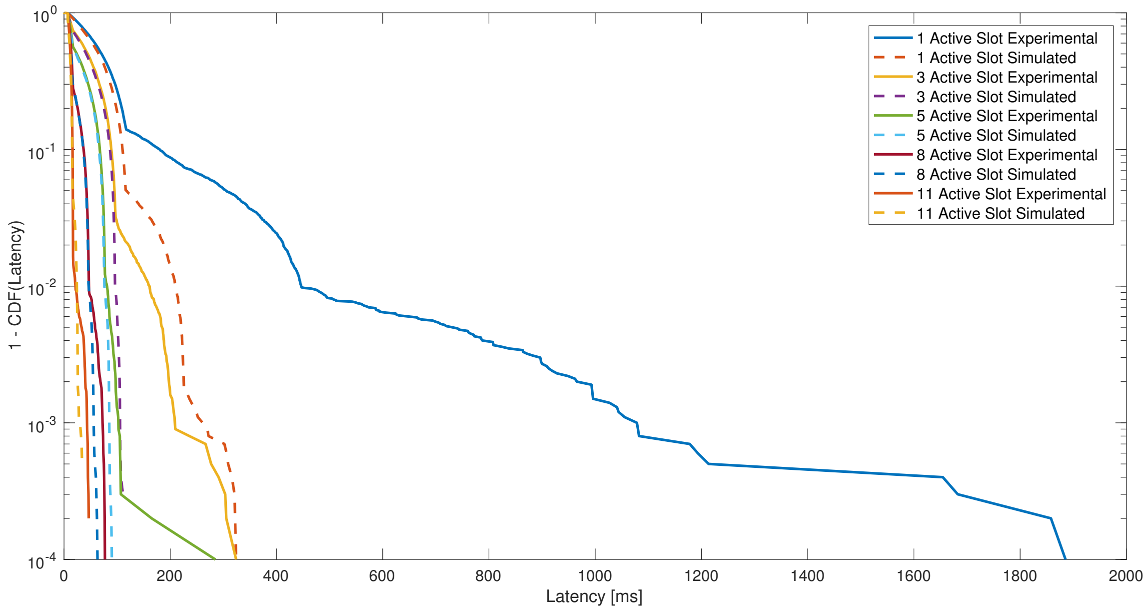

Fig. 11. Complimentary cumulative density functions (CCDFs) of end-to-end latency for 10,000 packets sent with 1/11, 3/11, 5/11, 8/11, and 11/11 active slots. The CCDFs only include packets that were received.

REFERENCES

[1] T. Watteyne, J. Weiss, L. Doherty, and J. Simon, "Industrial ieee802. 15.4 e networks: Performance and trade-offs," in *Communications (ICC), 2015 IEEE International Conference on*. IEEE, 2015, pp. 604–609.

[2] *IEEE Std 802.15.4e-2012*.

[3] 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.

[4] D. Dujovne, T. Watteyne, X. Vilajosana, and P. Thubert, "6tisch: deterministic ip-enabled industrial internet (of things)," *IEEE Communications Magazine*, vol. 52, no. 12, pp. 36–41, 2014.

[5] *IPv6 over the TSCH mode of IEEE 802.15.4e (6tisch)*, IETF.

[6] K. Pister and L. Doherty, "Tsmp: Time synchronized mesh protocol," *IASTED Distributed Sensor Networks*, pp. 391–398, 2008.

[7] X. Vilajosana, Q. Wang, F. Chraim, T. Watteyne, T. Chang, and K. S. Pister, "A realistic energy consumption model for tsch networks," *IEEE Sensors Journal*, vol. 14, no. 2, pp. 482–489, 2014.

[8] T. Watteyne, A. Mehta, and K. Pister, "Reliability through frequency diversity: why channel hopping makes sense," in *Proceedings of the 6th ACM symposium on Performance evaluation of wireless ad hoc, sensor, and ubiquitous networks*. ACM, 2009, pp. 116–123.

[9] X. Vilajosana, P. Tuset, T. Watteyne, and K. Pister, "Openmote: open-source prototyping platform for the industrial iot," in *International Conference on Ad Hoc Networks*. Springer, 2015, pp. 211–222.

[10] M. R. Palattella, N. Accettura, X. Vilajosana, T. Watteyne, L. A. Grieco, G. Boggia, and M. Dohler, "Standardized protocol stack for the internet of (important) things," *IEEE communications surveys & tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.

[11] T. Watteyne, M. Palattella, and L. Grieco, *Using ieee 802.15. 4e time-slotted channel hopping (TSCH) in the Internet of Things (IoT): Problem statement*, 2015.

[12] T. Chang, T. Watteyne, Q. Wang, and X. Vilajosana, "Llsf: Low latency scheduling function for 6tisch networks," in *Distributed Computing in Sensor Systems (DCOSS), 2016 International Conference on*. IEEE, 2016, pp. 93–95.

[13] R. Steigmann and J. Endresen, "Introduction to wisa: Wisa-wireless interface for sensors and actuators," *White paper, ABB*, 2006.

[14] C. B. Schindler, T. Watteyne, X. Vilajosana, and K. S. Pister, "Implementation and characterization of a multi-hop 6tisch network for experimental feedback control of an inverted pendulum," in *Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOpt), 2017 15th International Symposium on*. IEEE, 2017, pp. 1–8.

[15] "Analog Discovery 2," https://store.digilentinc.com/.

[16] Z. Shelby and C. Bormann, *6LoWPAN: The wireless embedded Internet*. John Wiley & Sons, 2011, vol. 43.

[17] *IPv6 over Low power WPAN (6lowpan)*, IETF.

[18] T. Winter, "Rpl: Ipv6 routing protocol for low-power and lossy networks," *IETF*, 2012.

[19] Z. Shelby, K. Hartke, and C. Bormann, "The constrained application protocol (coap)," *IETF*, 2014.