Competition: Adaptive Time-Slotted Channel Hopping

Atis Elsts, Xenofon Fafoutis,
Adeyinka Adeleke,
Robert Piechocki, George Oikonomou
Electrical and Electronic Engineering Department
University of Bristol

Simon Duquennoy
INRIA Lille, France
SICS Swedish ICT

Antonio Liñán, Marc Fàbregas
Zolertia

Abstract

Time-Slotted Channel Hopping (TSCH) from the IEEE 802.15.4-2015 standard uses channel hopping to combat interference and frequency-selective fading. It has attracted large attention from the research community due to its properties: high reliability in terms of packet delivery rates, and increased predictability in terms of energy consumption and latency, as compared to commonly used low-power CSMA MAC protocols.

This work makes use of the Contiki OS implementation of the TSCH protocol. We extend the standardized TSCH protocol with adaptive channel selection, adaptive time synchronization, and adaptive guard time selection to improve its energy efficiency and reliability properties.

1 Background

There is a growing need to make low-power wireless networks more reliable and more predictable in order to open them up to a wider range of applications (*e.g.*, industrial, automotive, e-health applications). Time-Slotted Channel Hopping (TSCH), specified in the IEEE 802.15.4-2015 standard [1], offers these features and has recently attracted attention from both industry and academia.

TSCH makes use of pseudorandom channel hopping to combat external interference and frequency-selective multipath fading. The core observation here is that several major causes of bad performance, such as WiFi interference or deep fading are very unlikely to equally affect all of IEEE 802.15.4 MAC-layer channels. Since a retransmission of an unacknowledged packet in TSCH with high probability takes place on a different channel, having just a few low-quality channels do not render the whole network unusable. In a similar way, changing the channel allows the nodes to get out of frequency-specific deep fading pockets caused by the multipath effect. In this way, the reliability of the networking plane in increased by making use of frequency diversity. Since the low-power wireless environment is notoriously dynamic and unpredictable, TSCH typically shows better results than adaptive selection of a single channel, as seen, for example, in ZigBee.

OpenWSN [7] is the *de facto* TSCH open-source reference implementation. In the recent years, Contiki port of TSCH has emerged as a competitor [2]. OpenWSN and Contiki versions of TSCH were demonstrated to be interoperable in Internet Engineering Task Force (IETF) plug-tests in Prague (2015) and in Paris (2016). Similarly to OpenWSN, the Contiki implementation supports multiple hardware platforms: NXP JN516x, Texas Instruments CC2538 and CC2650, Zolertia Z1, and others, including TelosB.

The Contiki implementation of TSCH is characterized by energy-efficient wake up pattern. Before entering sleep mode, the microcontroller calculates the time to the next active slot (transmission or reception), and schedules a wake up. In this way, no energy is wasted waking up at the start of every idle slot.

2 Our enhancements to TSCH

The objectives of this work are to increase energy efficiency, decrease latency, and keep high reliability of TSCH networks even in harsh conditions. To achieve these objectives, we will experiment with a number of techniques.

2.1 Adaptive time synchronization

This technique consists of learning the drift of the local clock compared to the drift of the network's time source and adaptively compensating for this expected drift [5]. In this way, the desynchronization between nodes can be kept small; in particular, it is bounded as long as the temperature changes in the environment remain bounded as well.

2.2 Adaptive guard time

Idle listening in the Rx guard time is know to have high impact on TSCH energy consumption [3]. By increasing the accuracy of time synchronization, this idle listening can be reduced. We plan to dynamically reduce guard time depending on the state of time synchronization. Our results show that decreasing guard time leads to higher energy efficient compared to the more conventional method of decreasing the rate of network synchronization packets [3].

2.3 Adaptive channel selection

To avoid transmissions on low-quality IEEE 802.15.4 channels, either the hopping sequence itself may be changed

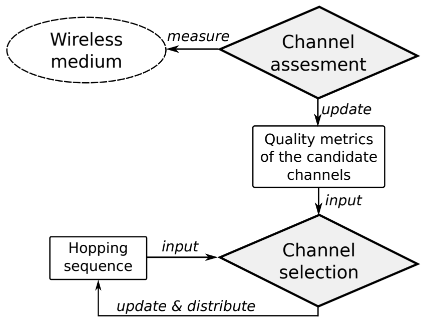

Figure 1: Overview of the adaptive channel selection process [4].

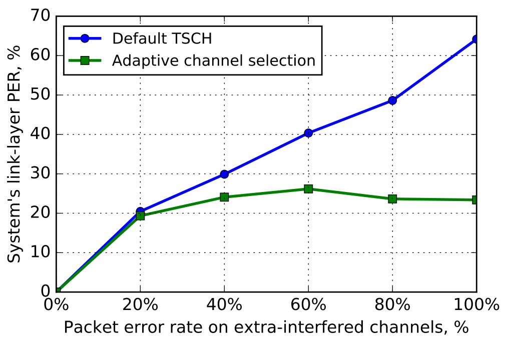

Figure 2: The effect from adaptive channel selection on link-layer packet error rate. The adaptive algorithm is able to select and use the less interfered channels (with 20 % PER), keeping the overall packet error rate low.

on the network nodes, or the nodes can pseudorandomly remap the set of low-quality channels to a different set of channels every time a transmission is done (Fig. 1). We have experimented with multiple techniques [4]: upstream-based selection and downstream-based selection, as well as multiple channel quality metrics: PRR and idle-channel energy levels. Assuming a situation where some of the channels are more interfered than other channels, the adaptive selection helps to decrease link-layer packet error rate (Fig. 2). Since the dependability competition requires relatively fast adaptations compared to the number of packets being sent, upstream-based channel selection using the idle-channel energy level metric is expected to perform better than the downstream-based selection using the PRR metric.

2.4 Network join process improvements

The robustness of TSCH join process can be improved by increasing the number of physical channels, as this increases the probability that some of the less-interfered channels will be used for sending enhanced beacon (EB) packets. However, using more channels also makes the join process slower in the average case. Similarly, the speed of the process can be improved by sending EB packets more frequently; however, too fast EB scheduling is going to lead to packet collisions and actually decrease performance. We will investigate practical solutions for these optimization problems, *i.e.*, selecting the number of channels to use and the EB frequency. We also will experiment with sending EB packets on multiple channels simultaneously, and sending EB packets in multiple slots in the same slotframe.

2.5 Routing and scheduling

We plan to use flooding or multi-path forwarding to additionally exploit the spatial diversity of nodes besides the frequency diversity of channels already naturally provided by TSCH. Link selection will be based on EB packet reception probabilities and other factors. The design of an efficient TSCH schedule for this competition is an open question that we plan to tackle at a later stage.

2.6 Improvements in hardware-specific code

The Contiki hardware driver for the CC2420 radio chip can be improved in terms of energy efficiency, for example, by turning the chip off completely during idle periods – such as is done, for example, in the ManOS operating system [6], rather that keeping it in idle mode, with its voltage regulator remaning powered. This "radio off" mode compared to the "radio idle" mode has the drawback of a much slower wakeup. However, since the time of the next active TSCH slot is known beforehand, techniques such as pre-emptive wakeup can be used to work around this problem. These techniques are already present in the code of more recent Contiki hardware platforms, for example, CC2650 [3] and JN516x; we will backport them to CC2420.

Similarly, MCU energy usage can be optimized by putting the MCU in sleep during radio packet reception and transmission. This requires changes both in the CC2420 driver and the core TSCH code.

3 Concluding remarks

We plan to build on the open-source implementation of TSCH in Contiki already known to provide high reliability in terms of packet delivery rate even in challenging environments [2]. We plan to enhance it in several ways to make the implementation more energy-efficient and have lower latency.

Acknowledgments

This work was performed under the SPHERE (a Sensor Platform for Healthcare in a Residential Environment) IRC funded by the UK EPSRC, Grant EP/K031910/1 and a grant from CPER Nord-Pas-de-Calais/FEDER DATA.

4 References

[1] IEEE Standard for Local and metropolitan area networks—Part 15.4. IEEE Std 802.15.42015, 2015.

[2] S. Duquennoy, B. Al Nahas, O. Landsiedel, and T. Watteyne. Orchestra: Robust Mesh Networks Through Autonomously Scheduled TSCH. In *ACM SenSys*, pages 337–350, 2015.

[3] A. Elsts, S. Duquennoy, X. Fafoutis, G. Oikonomou, R. Piechocki, and I. Craddock. Microsecond-Accuracy Time Synchronization Using the IEEE 802.15.4 TSCH Protocol. In *IEEE SenseApp*, 2016.

[4] A. Elsts, X. Fafoutis, R. Piechocki, and I. Craddock. Adaptive Channel Selection in IEEE 802.15.4 TSCH Networks, 2017. Unpublished manuscript, `http://user.it.uu.se/~atiel485/tsch_channels_submitted_2017.pdf`.

[5] D. Stanislowski, X. Vilajosana, Q. Wang, T. Watteyne, and K. S. Pister. Adaptive synchronization in IEEE802.15.4e networks. *Industrial Informatics, IEEE Transactions on*, 10(1):795–802, 2014.

[6] G. Strazdins, A. Elsts, and L. Selavo. MansOS: easy to use, portable and resource efficient operating system for networked embedded devices. In *ACM SenSys*, pages 427–428, 2010.

[7] 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*, 23(5):480–493, 2012.