Simple Distributed Scheduling with Collision Detection in TSCH Networks

Kazuki Muraoka, Thomas Watteyne, Nicola Accettura, Xavier Vilajosana, Kristofer Pister

Simple Distributed Scheduling With Collision Detection in TSCH Networks

Kazushi Muraoka, Thomas Watteyne, Nicola Accettura, Xavier Vilajosana, and Kristofer S. J. Pister

***Abstract*— The IETF IPv6 over the time synchronized channel hopping mode of IEEE 802.15.4e (6TiSCH) working group standardizes a distributed mechanism for neighbor motes to agree on a schedule to communicate, driven by a scheduling function. This letter introduces the notion of housekeeping to the schedule function, in which motes relocate cells in the schedule to build a collision-free schedule in a distribute manner. The solution, based on detecting underperforming cells and listening for unexpected packets, does not require additional signaling traffic or state. This letter shows that simple housekeeping rules introduced allow a network to contain 17% more motes and suffer from 64% fewer collision, while maintaining end-to-end reliability above 99.5% in a typical industrial environment. This solution is now being discussed for standardization at 6TiSCH.**

***Index Terms*— Time synchronized channel hopping (TSCH), distributed scheduling.**

I. Introduction

THE IEEE802.15.4e-2012 standard introduces the Time Synchronized Channel Hopping (TSCH) mode. TSCH combines time division multiplexing with channel hopping to achieve ultra low power and ultra high reliability. Motes in a TSCH network communicate by following a schedule which indicates in which time slots to transmit, receive or sleep. We call "cell" a time slot offset and channel offset in the TSCH schedule. The IETF IPv6 over the TSCH mode of IEEE802.15.4e (6TiSCH) working group is standardizing the 6TiSCH operation sublayer (6top) which manages the TSCH schedule.

The 6TiSCH architecture allows for distributed TSCH schedule management [1] in which a protocol, called the "6top Protocol" (6P), enables neighbor motes to negotiate to add/remove cells to one another. The challenge is that two pairs of nearby neighbors might choose the same cell(s) to communicate, resulting in in-network packet collisions.

We propose a technique in which neighbor motes schedule communication cells to one another at random locations in the schedule, and combine that with a "housekeeping" mechanism which detects schedule collisions and relocates each colliding cell to a different position in the schedule.

Manuscript received March 21, 2016; revised May 20, 2016; accepted May 22, 2016. Date of publication May 25, 2016; date of current version July 6, 2016. This work was supported by the Spanish Ministry of Economy and European Regional Development Fund through the SINERGIA Project under Grant TEC2015-71303-R. The associate editor coordinating the review of this paper and approving it for publication was Dr. Ashish Pandharipande.

K. Muraoka is with NEC Corporation, Kawasaki 211-8666, Japan (e-mail: k-muraoka@fq.jp.nec.com).

T. Watteyne is with Inria, Paris 75012, France (e-mail: thomas.watteyne@inria.fr).

N. Accettura is with the Laboratory for Analysis and Architecture of Systems, Toulouse 31400, France (e-mail: nicola.accettura@laas.fr).

X. Vilajosana is with the Universitat Oberta de Catalunya, Barcelona 08018, Spain (e-mail: xvilajosan@uoc.edu).

K. S. J. Pister is with the Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, Berkeley, CA 94720 USA (e-mail: ksjp@berkeley.edu).

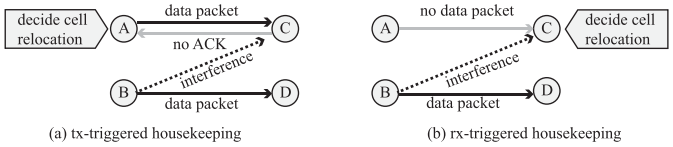

Fig. 1. The two housekeeping approaches.

II. Random Scheduling and Housekeeping

To obtain some bandwidth with a neighbor in distributed scheduling algorithm, a mote uses 6P to negotiate adding a number of cells corresponding to the bandwidth requirements. The mote builds a 6P ADD Request that includes a *randomly* picked candidate list of cells. The 6P ADD Request is sent to the neighbor, which checks which cells from the candidate list are not used in its current schedule. Among those, the neighbor *randomly* selects one, and sends back a 6P Response. Without *a priori* knowledge of the available cells on a pair of negotiating motes, this results in simple distributed scheduling. Yet, collisions may happen when a cell is scheduled for transmission between more than one close pair of neighbors. The 6top sublayer is responsible for keeping good link reliability; it hence has to detect collisions and start a new negotiation for relocating the collision-prone cells. This letter refers to such a periodic process as "6top housekeeping". We introduce two housekeeping approaches. **tx-housekeeping** runs on the transmitting mote. It detects schedule collisions by comparing the performance of different cells to the same neighbor. **rx-housekeeping** runs on the receiving mote. It detects schedule collisions when overhearing packets from a mote which is not the neighbor it expects packets from.

A. Tx-Housekeeping

In Fig. 1(a), motes `A` and `B` transmit in the same cell to `C` and `D`, respectively. This is the collision condition tx-housekeeping detects. Since `B` and `C` are in the same interference domain, packet transmissions can collide. Mote `A` tracks the Packet Delivery Ratio (PDR) of all the cells it has to `C`. The PDR of a cell is calculated using (1).

$$PDR = \frac{\#\ of\ ACK}{\#\ of\ Tx}. \tag{1}$$

Mote `A` can have multiple cells to the same neighbor; these cells form a "bundle". Normally, all cells in a bundle have the same PDR. Tx-housekeeping detects that a cell suffers from collisions when that cell has a PDR significantly lower than the others in the bundle. If that is the case, tx-housekeeping triggers a relocation of that cell to a different (random) cell in the schedule.

It might also happen that all cells in the bundle suffer from collisions. To detect this situation, tx-housekeeping compares the average PDR over the bundle to the expected PDR, given the Received Signal Strength Indicator (RSSI) of the packets received over those cells. If there is a significant difference, all cells in the bundle are relocated. The following rules trigger the relocation of a cell or a bundle:

$$\text{relocate a cell if } PDR_{cell} < PDR_{others}/T \quad (2)$$

$$\text{relocate a bundle if } PDR_{bundle} < PDR_{RSSI}/T \quad (3)$$

Eqs. (2) and (3) formalize the two conditions which trigger cell relocation. T is a threshold value greater than 1 specifying how frequently a relocation should be triggered. A small value of T makes tx-housekeeping more sensitive to collisions, while increasing false positives and thus the signaling overhead. A large value of T makes the housekeeping more conservative. When collisions are rare, the PDR reduction is not detectable by the tx-housekeeping, although reliability is still degraded.

B. Rx-Housekeeping

In Fig. 1(b), assume `A` has no packets for `C`, while `B` transmits packets to `D`. `C` powers on its radio to receive packets from `A`, but receives packets from `B`. The rx-housekeeping function on `C` relocates that cell to avoid future collisions. tx- and the rx-housekeeping are complementary, this letter presents a hybrid approach in which both are used.

III. Simulation Results

We use the open-source 6TiSCH simulator.[1] A Directed Acyclic Graph (DAG) root node of the Routing Protocol for Low-Power and Lossy Network (RPL) is placed at the center of a 1 km $\times$ 1 km deployment area. It is the sink for all data generated in the network. At each simulation run, motes are randomly positioned while ensuring that each mote has at least 3 neighbors to which it connects with a link with PDR $\geq$ 50%.

Time slot duration is 10 ms, a slotframe is 101 time slots long. The simulated network channel hops on 4 frequencies, which is equivalent to using 16 channels on a network with 4 times more motes and 4 DAGroots. Each mote can have at most 3 RPL parents. The application on each mote generates a data packet periodically, with a jitter of $\pm$ 5% of the period.

Each mote's transmission queue can hold up to 10 packets. A packet is dropped when (i) the application generates the packet but the queue is full, or (ii) the queue is full and the packet has been retransmitted 5 times or more. The tx-housekeeping function is only applied to cells having 10 transmitting attempts, to let the measured PDR be a reliable statistic and avoid an unnecessary relocation. Threshold T is set to 1.5. On the basis of the number of packets in the queue, a mote determines the number of cells to be scheduled to a neighbor mote, as described in [2]. Results are presented alongside the interference-free case, in which an infinite number of channel offsets are available. This scenario, while unrealistic, represents the best possible case. Results are average over 400 simulation runs and presented with a 95% confidence interval.

Fig. 2 shows how network size affects end-to-end packet loss ratio. The packet period is set to 2 s. At these low data rates, the simplest random scheduling without 6top relocation housekeeping is sufficient in networks with less than 50 motes to obtain an end-to-end packet loss ratio below 10^{-3}. With bigger networks, collisions are more frequent, causing an increased packet loss. In this case, 6top housekeeping improves the network performance. In networks with $\leq$ 100 motes, rx-housekeeping prevents collisions better than the tx-housekeeping. In network with $>$100 motes, tx-housekeeping function detects collisions better than rx-housekeeping. The joint use of tx- and rx-housekeeping ("tx-rx-housekeeping" in Fig. 2) combines their advantages, and performs better in all networks. Using tx-rx-housekeeping allows 17% larger networks while keeping the same end-to-end reliability – defined as 1 minus end-to-end packet loss ratio – above 99.5%.

[1] Available at `https://bitbucket.org/6tisch/simulator`.

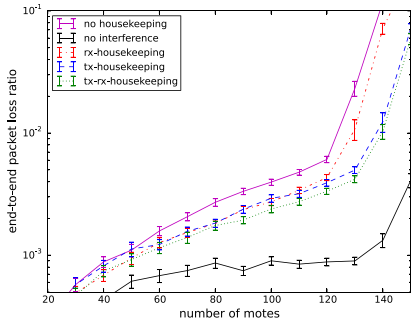

Fig. 2. For a given target end-to-end packet loss, using housekeeping increases the number of motes that can be in the network.

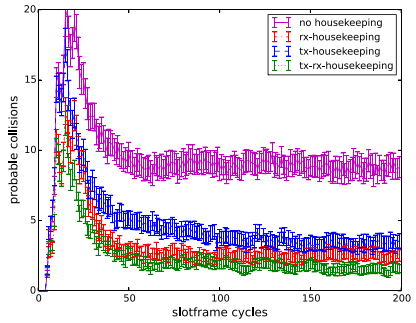

Fig. 3. Convergence of the probable collisions per a slotframe cycle.

To plot Fig. 3, we simulate an 80-mote network where all motes are powered on at the same time and start generating packets immediately. At the beginning of the life of the network, motes discover one another, start forming a multi-hop structure, and add cells to one another using 6P. This results in an increased number of "probable" collisions, the number of interfering packets received by the motes (Fig. 3). The number of collisions decreases to a steady state as this initial activity stops and all the required cells are installed. This convergence to steady-state happens regardless of whether housekeeping is used, but the amount of collisions in steady state drops by 64%, from 9.5 without housekeeping to 3.4 with tx-rx-housekeeping. We are currently working on large scale experimentation, which includes a thorough comparison with other existing techniques.

References

[1] Q. Wang, X. Vilajosana. "6top Protocol (6P)," IETF I-D. draft-ietf-6tisch-6top-protocol (Work in Progress). Jun. 2016. [Online]. Available: https://tools.ietf.org/html/draft-ietf-6tisch-6top-protocol-00

[2] M. R. Palattella *et al.*, "On-the-fly bandwidth reservation for 6TiSCH wireless industrial networks," *IEEE Sensors J.*, vol. 16, no. 2, pp. 550–560, Jan. 2016.