Experimental Characterization of Joint Scheduling and Routing Algorithm over 6TiSCH

Gordana Gardašević, Dragan Vasiljević
University of Banja Luka
Faculty of Electrical Engineering
Banjaluka, Bosnia and Herzegovina
gordana.gardasevic@etf.unibl.org
dragan.vasiljevic@mtel.ba

Chiara Buratti, Roberto Verdone
DEI, University of Bologna
Bologna, Italy
{c.buratti, roberto.verdone}@unibo.it

***Abstract* - Recently, the Time-Slotted Channel Hopping (TSCH) mode was introduced as an amendment to the Medium Access Control (MAC) part of the IEEE 802.15.4 standard. TSCH is the emerging standard for industrial automation and process control for Low-power and Lossy Networks (LLNs) that uses time synchronization to achieve low-power operation and channel hopping to enable high reliability. This paper presents the preliminary results obtained from the COST Action CA15104 - IRACON activity. We have integrated the research related to the implementation of Internet of Things (IoT) networks based on the newly proposed "IPv6 over TSCH - 6TiSCH" standard, and the research dealing with the definition of novel joint scheduling and routing algorithms for centralized IoT networks. For experimentations, we have used the OpenMote hardware and OpenWSN software platform for IoT applications.**

Keywords* - *Internet of Things (IoT); TSCH; 6TiSCH; OpenMote platform; OpenWSN; Scheduling; Routing

I. Introduction

Recent trends in Internet of Things (IoT) activities are directed towards the open-software and open-hardware prototyping and development. The OpenWSN operating system (OS) [1] and the OpenMote hardware platform [2] are examples of such an initiative, and are particularly suitable for Industrial IoT (IIoT) applications. Both OpenWSN OS and OpenMote have the support for new Time-Slotted Channel Hopping (TSCH) mode of IEEE 802.15.4e Medium Access Control (MAC). TSCH differs from other low-power MAC protocols because of its scheduled nature [3]. Due to its time synchronization and channel hopping, it enables ultra-low power operations and highly reliable mesh networks. All nodes in a TSCH network are synchronized. Time is divided into timeslots which are grouped into one of more slotframes, where a slotframe continuously repeats over time. The size of slotframe (i.e., the number of timeslots within the slotframe) is tightly related to the application requirements. The shorter the slotframe, the more often a timeslot repeats, resulting in more available bandwidth, but also in a higher power consumption. TSCH is a deterministic protocol since a node only wakes up at its own assigned timeslots.

The 6TiSCH (IPv6 over the TSCH mode of IEEE 802.15.4e) mechanisms are of a particular importance for the further adoption of IPv6 in industrial standards [4]. However, the standard does not define or propose a particular scheduling and routing algorithms, whose definition is left to the designers. Therefore, we have tested the performance of the Joint Scheduling and Routing Algorithm (JSRA) proposed in [5], when applied to the 6TiSCH standard, and we have compared results with a simple Benchmark algorithm [5].

The structure of the paper is as follows. Section II gives an overview on the OpenWSN protocol stack implementation based on 6TiSCH architecture, as well as the description of JSRA. The details on the experimental platform and components are provided in Section III. The results are presented and discussed in Section IV. Finally, we conclude the paper with indicating the future direction in Section V.

II. The Implemented Protocol Stack

A. *OpenWSN Protocol Stack*

The OpenWSN OS is a recently released open-source implementation of a fully standards-based protocol stack for IoT capillary networks, rooted in the new IEEE 802.15.4e TSCH standard, as shown in Fig. 1. OpenWSN implements, on top of IEEE 802.15.4e, IoT standards such as IPv6 over Low power Wireless Personal Area Networks (6LoWPAN), Routing Protocol for Low power and Lossy Networks (RPL) and Constrained Application Protocol (CoAP), thus enabling a seamless network connectivity to the IPv6 Internet.

The IEEE 802.15.4e TSCH standard replaces the traditional MAC protocol, without changing the underlying physical layer. In TSCH, time is divided into groups of timeslots called slotframe that repeats over time. A single timeslot is long enough for the transmitter to send a maximum length packet, and for the receiver to send back an acknowledgment (ACK). All nodes in the network share the same Absolute Slot Number (ASN). When the coordinator/root initiates the network formation, it sets ASN to zero. From that point on, ASN increments by 1 in each timeslot. Once a node is synchronized to a network, it sets its ASN to the network's ASN. Through advertisement (ADV) packets, the current ASN is shared.

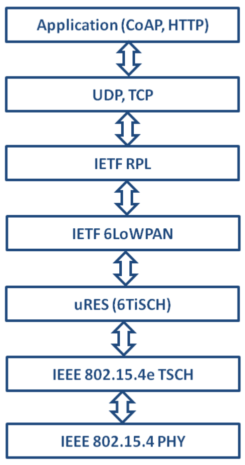

Fig. 1. OpenWSN protocol stack

Assigned timeslots within the slotframe can be categorized as beacon transmit timeslots, Enhanced Beacon (EB) listen timeslots, shared timeslots, dedicated transmit timeslots and dedicated receive timeslots. When a node wishes to join a network, it listens for EBs advertized by nodes which have been already synchronized. This node then continuously advertises EBs to other joining nodes for maintaining network synchronization. Beacons carry slotframe timing information, channel hopping sequence, beacon and shared timeslot schedule, transmit and receive timeslots. Control frames, such as assocation/diassocation, are transmitted and received in the shared timeslots. In the dedicated timeslots, the node transmit or receive a packet, depending on whether the timeslot is a transmit or receive timeslot. Note that during dedicated timeslots, only nodes assigned to communicate within these timeslots are active, thus, there is no interference from other nodes. On the contrary, in case of shared timeslots, each node can contend for transmission of packets.

The next upper layer, uRES, can be considered as a Logical Link Layer (LLC). uRES is a partial implementation of IETF 6TiSCH standard that is still in development phase. This protocol is developed within the OpenWSN to fill the gap between IEEE 802.15.4e and the upper part of protocol stack. IEEE 802.15.4e does not specify the policies to build and maintain the communication schedule. Therefore, agreeing is necessary because the transmitter will need to transmit at the same time and frequency as the receiver is listening. uRES allows each node to establish bidirectional connectivity with all of its neighbors [6].

The 6LoWPAN layer enables the integration of IPv6 protocol in low rate WPAN. 6LoWPAN removes redundant fields from IPv6 and User Datagram Protocol (UDP) protocols taking into account characteristics of IEEE 802.15.4, IPv6 and UDP protocols.

The IPv6 Routing Protocol for Low-Power and Lossy Networks (RPL) is the routing protocol responsible for packet forwarding between nodes that are multiple hops away, as well as for creation and maintenance of routing tables in network nodes. In our case, RPL is used for initial setup in order to synchronize all nodes in the network.

At the transport layer, UDP is used due to its lightweight implementation. OpenWSN supports CoAP at the application layer, both for configuration and interaction with coordinator node.

The OpenVisualizer (OV) is a Python-based debugging and visualization program. The OV creates IPv6 TUN (network TUNnel) interface that is a virtual IPv6 interface tunneled through the serial port. This interface enables the communication between IoT devices and IPv6 Internet. The OV also provides the graphical web interface for network management purpose and various traffic and node parameter statistics (through the CoAP interaction based on RESTful architecture).

B. The Joint Scheduling and Routing

As stated above, the 6TiSCH standard does not define how to assign dedicated timeslots. Therefore, we implement, on top of IEEE 802.15.4e, the JSRA protocol.

The JSRA algorithm derives a tree-based topology and a scheduling strategy in a joint way. This proposal is conceived similarly to the Dijkstra's algorithm, which builds the tree iteratively by progressing from the root, incrementally outward; the optimality of the algorithm is exploited, introducing the consideration of interference in the weights. At each iteration, when a new link is added to the tree, a colour (i.e., a timeslot) is assigned to it according to the actual interference that it will generate on the previously defined links. In this way, it is ensured that packet losses are avoided, as the Signal to Interference Ratio (SIR) is kept above the capture ratio for all links, since we are able to account for the sum of all possible interferences. Results are compared with those obtained when applying a benchmark solution, where routing is implemented via Dijkstra, using as link cost a linear function of the power received over the link, and for the scheduling DSATUR colouring algorithm [7].

III. Experimental Platform and Setup

A. OpenMote hardware platform

The OpenMote is a representative of new generation open-hardware platforms that is particularly adapted to the IIoT applications. The OpenMote hardware is shown in Fig. 2. It is composed of four boards: OpenMote-CC2538, OpenBase, OpenBattery, and the OpenUSB (from the left-side to the right-side, respectively).

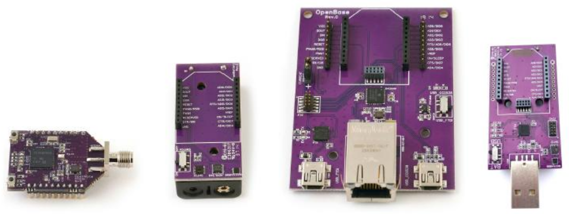

Fig. 2. OpenMote hardware

The OpenMote-CC2538 is based on TI CC2538 SoC functionalities and has the following peripherals: step-down DC/DC converter, two crystal oscillators, 4 LEDs, antenna connector, and two programmable buttons. The microcontroller runs up to 32 MHz and includes 32 Kbytes of RAM and 512 Kbytes of Flash memory, as well as the common peripherals (GPIOs, ADC, timers, etc.). The radio operates at 2.4 GHz band and is fully compatible with IEEE 802.15.4-2006 standard. The OpenMote is also fully compliant with the XBee form factor. The OpenBattery enables battery-powered supply for sensor board with three digital sensors: a temperature/humidity sensor, an acceleration sensor, and a light sensor. The OpenBase enables programming, debugging and the communication with a computer through the serial port or USB port. The OpenUSB provides the USB connection for easier programming and debugging capabilities.

B. *Experimental Setup*

In order to have a fair comparison among different solutions, the experiments were set up within the Flextop platform at UNIBO [5]. This platform allows the proper characterization and control of the experimental environment, thus providing the fair comparison among protocols even though tests are performed at different time instances.

Ten OpenMote-CC2538 nodes were located into boxes on the walls of a corridor at UNIBO. The boxes were deployed in the corridor, with nodes deployed at fixed positions (as shown in the map in Fig. 3). Node 4f, at the end of the corridor (left side of the map in Fig. 3), acts as the coordinator of the network. The host computer is running Ubuntu 14.04 LTS with OpenWSN and OV installed. The node that is directly connected to host is declared as Destination Oriented Acyclic Graph (DODAG) root. DODAG represents the hierarchical organization of nodes in the network, based on RPL protocol specifications. All nodes use one assigned frequency channel, in our case - channel 26. In order to properly describe the environment, before the start of experiments, the average received power matrix was measured.

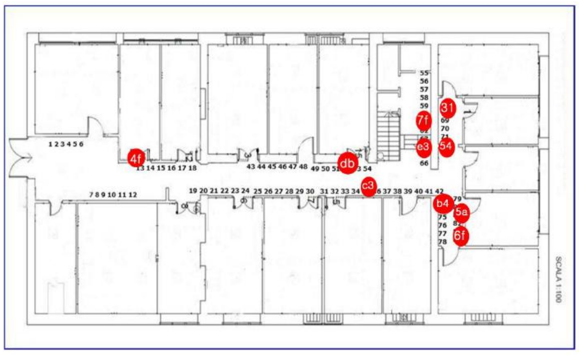

Fig. 3. Position of nodes in testbed

The setup for these measurements is as follows:

- nodes are at fixed and known positions;
- channel gains between each pair of nodes are measured at the beginning of each test;
- experiments are performed during the night, when nobody is present, thus avoiding uncontrollable channel fluctuations.

The matrix of channel gains has been obtained as follows: each node sends a burst of 500 packets to let other nodes compute the average power received. The results are presented in Table 1. This matrix is the input for the JSRA algorithms. The sign '-' means that there is no connectivity between two nodes.

TABLE I. AVERAGE RECEIVED POWERS (dBm) MATRIX FOR −5 dBm

IDs RSSI	4f	5a	b4	db	e3	7f	6f	54	31	c3
4f	-	-	-	-96	-87	-98	-	-	-	-78
5a	-	-	-66	-57	-87	-79	-	-	-78	-87
b4	-	-64	-	-74	-79	-81	-98	-	-76	-84
db	-96	-56	-72	-	-84	-86	-	-	-77	-87
e3	-87	-87	-79	-83	-	-77	-96	-	-78	-62
7f	-98	-78	-80	-85	-77	-	-97	-88	-70	-74
6f	-	-78	-88	-85	-86	-95	-	-	-84	-82
54	-	-	-95	-	-98	-89	-	-	-93	-90
31	-	-	-	-	-	-88	-	-	-	-91
c3	-78	-84	-83	-86	-61	-73	-90	-91	-80	-

C. *Scheduling in OpenWSN*

All sensor nodes in an IEEE 802.15.4e network are synchronized. The timeslot duration is 15 ms, and supports the transmission / reception of a single packet along with its acknowledgment. The coordinator (root) is entitled to transmit the EB during the first timeslot (denoted as slot 0 in OV) containing the scheduling of the entire network. At least three timeslots in slotframe must be reserved for node's serial data transfer. Data transmission is performed in active timeslots. For measurement purposes, nodes are sending 500 query packets towards the coordinator. The packet payload size is 20 bytes. The interpacket time is 165 ms.

IV. RESULTS

We have first applied the Benchmark algorithm and, as a result, this algorithm generates the routing topology (in OV) presented in Fig. 4. Moreover, this algorithm gives, as the output, seven timeslots used for data communication: five dedicated slots (slots 4, 5, 6, 7, and 10), and two shared slots (slots 8 and 9). The scheduling is performed according to the TSCH specifications, Fig. 4. Slot 0 is reserved for exchanging EB messages, while slots 1, 2 and 3 are reserved for serial activity. The transmission of data packets is organized in slots 4-10, Fig 5.

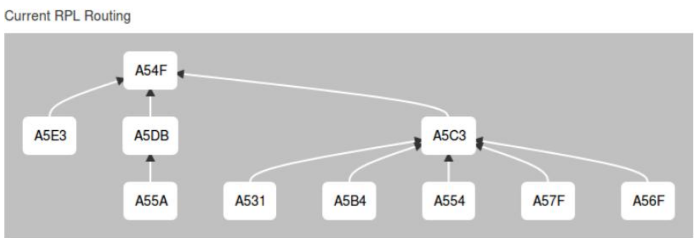

Fig. 4. Routing topology for Benchmark algorithm in OV

The Benchmark scheduling, implemented in OpenWSN, gives the following seven active slots, thus indicating which node is transmitting packet in particular slot:

slot 4	slot 5	slot 6	slot 7	slot 8	slot 9	slot 10
31	54	6f	b4	5a,e3	db,7f	c3

Fig. 5. Benchmark scheduling

The JSRA algorithm provides the following routing topology (in OV), Fig. 6:

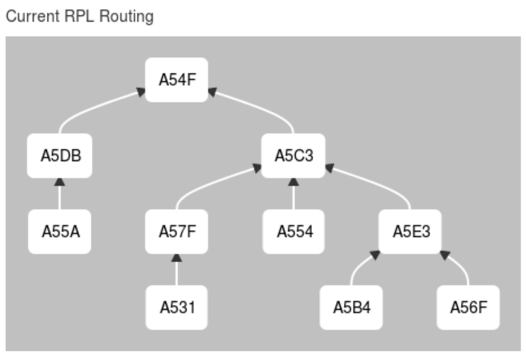

Fig. 6. Routing topology for JSRA algorithm in OV

The JSRA scheduling gives six active timeslots: three dedicated and three shared, Fig. 7:

slot 4	slot 5	slot 6	slot 7	slot 8	slot 9
54	6f,31	b4	db,7f	5a,e3	c3

Fig. 7. JSRA scheduling

To provide the comparison of three different protocols, the results in terms of packet delivery ratio (PDR) and throughput are summarized in Table 2.

TABLE II. PERFORMANCE EVALUATION

Protocol	Packet Delivery Ratio	Throughput [kb/s]
6TiSCH	100%	12
Benchmark	95%	13.2
JSRA	99%	15

The routing and scheduling strategy for 6TiSCH network requires careful planning and optimization. In our case, the scheduling is based on hybrid scheme where shared (contention-based timeslots) and dedicated timeslots (contention-free timeslots) coexist within the same slotframe. The results show that 6TiSCH provides high-reliability packet delivery. For obtaining a higher throughput, some additional policies on resource allocation or dynamic scheduling approach should be applied. The additional resource allocation is particularly required for multihop topologies due to the fact that nodes with more children need to forward more data towards the root.

V. Conclusion

This paper presents the integration of JSRA with OpenWSN 6TiSCH protocol stack, where the performances in terms of PDR and throughput are evaluated for Benchmark and JSRA algorithms. Based on the measured average received power matrix, JSRA algorithm produces a certain routing topology and timeslot scheduling. The trade-off between the number of active slots and application requirements is necessary, especially in multihop networks where additional resources are required. 6TiSCH protocol-stack development is still in progress and is open to contributions. We will further investigate the routing and scheduling strategies for 6TiSCH networks.

Acknowledgment

This work has been performed under the framework of COST Action CA15104, Inclusive Radio Communication Networks for 5G and beyond (IRACON).

References

[1] OpenWSN project. http://www.openwsn.org/.

[2] OpenMote platform. http://www.openmote.com/.

[3] IETF, Using IEEE 802.15.4e Time-Slotted Channel Hopping (TSCH) in the Internet of Things (IoT): Problem Statement. https://tools.ietf.org/html/rfc7554.

[4] IPv6 over the TSCH mode of IEEE 802.15.4e (6tisch). https://datatracker.ietf.org/wg/6tisch/documents/.

[5] C. Buratti, R. Verdone, Joint scheduling and routing with power control for centralized wireless sensor networks, Wireless Networks, pp. 1-16, 2016.

[6] Gardašević, G., Veletić, M., Maletić, N. et al. Wireless Pers Commun (2017) 92: 127. doi:10.1007/s11277-016-3842-3.

[7] Z. Walczak and J. Wojciechowski, "Transmission scheduling in packet radio networks using graph coloring algorithm," in Wireless and Mobile Communications, 2006. ICWMC '06. International Conference on, July 2006, pp. 46–46.