Distributed Cell Scheduling for Multichannel IoT MAC Protocols

Ahlem Hannachi
Lesia Lab.
Computer Science Department
Biskra University, Algeria
a.hannachi@univ-biskra.dz

Abdelmalik Bachir
Lesia Lab.
Computer Science Department
Biskra University, Algeria
a.bachir@univ-biskra.dz

***Abstract*—We provide a distributed solution to the scheduling of slots over multiple channels in low power wireless sensor networks organized in data gathering tree which are the topologies that are expected to be deployed in the Internet of Things (IoT). Our solution generates a dynamic and distributed schedule that achieves high throughput, low packet errors and collisions, as well as low energy consumption by avoiding deafness and reducing both overhead and overhearing. In our solution, nodes do not communicate large numbers of control message as local topology construction is only based on local link exchange to construct 2-hop neighborhood. To reach our goal, we chose to make use of Combinatorial Design theory, particularly Block Design. Within Block Design, we particularly focused on Latin Squares/Latin Rectangles (LS/LR), and Balanced Incomplete Block Design (BIBD). We evaluate the performance of our solution by comparing it with DiSCA (Distributed Scheduling for Convergecast in Multichannel WSN) and show that we obtain better results in terms of throughput, packet delivery rate, and energy consumption. Our results are based on our implementation of both protocols on NS-3.26.**

***Index Terms*—IoT, WSN, TSCH, Low-Power, Distributed Scheduling, Block Design.**

I. Introduction

The coalesce of the Internet into the various real world applications is in a persistent progress and is leading to the materialization of the Internet of Things where heterogeneous physical objects with different characteristics will be interconnected. The Internet of Things (IoT) is expected to have a significant impact on various aspects of our lives by modernizing industrial and agricultural sectors as well as improving the service industry [1], [2], [3]. According to one of the most prominent visions, the IoT is being designed to full comply with the exiting TCP/IP architecture while relying on IPv6 instead of traditional IPv4 to provide end-to-end reachability to several billions of objects and leverage the use of existing successful and robust Internet applications and services. While the use of IPv6 solves the problem of address shortage, it increases the header size as more bits are added to express larger addresses, which creates a burden for memory-constrained objects which rely on low power radios for communication such as those based on the IEEE 802.15.4 standard. Due to the packet size limitations imposed by 802.15.4 standard, these constrained objects typically use a compressed version of IPv6 to reduce the header size and organize themselves into networks, called the 6LowPAN networks, and get interconnected to the Internet through a border router called 6LBR (6LowPAN Border Router) [4].

This work is funded in part by the Algerian Ministry of Higher Education and Scientific Research under contract B*01420130132.

The objects forming the 6LowPAN network are typically very constrained in computation, memory, storage, and rely on a finite source of energy for their operation. As such, they rely on the good performance of the protocols dealing with their operation. The IEEE 802.15.4e is a recent amendment aiming at overcoming several shortcomings of the initial 802.15.4 standard such as low reliability, unbounded packet delays and the absence of protection against interference [5]. The IEEE 802.15.4e standard introduced several modes of operation among which the Time Synchronized Channel Hopping (TSCH) and Synchronous Multi-channel Extension (DSME) are particularly interesting in the management of energy [6], [7]. TSCH is based on slicing the time into equal slots and organizing the nodes activity on these slots. In every slot, a node is either in transmit, receive, or sleep state. The slot length should be long enough to handle the transmission of the longest frame (127 byte) and the reception of its acknowledgment. TSCH relies on the establishment and maintenance of a schedule that defines how each slot is used by each node which has not been defined by the standard and left to an entity called Logical Link Control [8].

Several scheduling protocols have been proposed for multichannel networks [9] with the aim of increasing throughput, lowering delay, and improving reliability while reducing energy consumption. The scheduling consists in distributing slots per node or link. Per node slot distribution consists in assigning slots to nodes to receive frames from any other node or to transmit frames to any other node. A per node slot distribution can be transmitter-based, receiver-based, or a combination of both. Per link slot distribution consists in attributing slots to particular link, for example a transmission from one node to another node. Links can be either unidirectional or bidirectional. Several studies have been carried out to compare the performance of these protocols and have shown that in the general case where traffic and topology are unknown and dynamic, the receiver-based scheduling is more energy saving than the other schemes [10]. Although receiver-based scheduling achieves high energy savings as nodes only wake up during those reception slots to listen for incoming traffic, it has the possibility to lead to deafness situations which happens when a node starts a transmission and the intended receiver misses it as it is not listening to the right slot. The use of multichannel exacerbates the deafness problem as the intended receiver must be listening to the right slot and channel to correctly receive the packet intended to it. Deafness has been taken into account in [11] where we proposed a solution to avoid it in the general case. In this paper, we consider the particular case where the network is organized in a data gathering tree.

We focus on providing a distributed solution to the scheduling of slots over multiple channels that would be used by TSCH or similar modes. Our objective is to propose a dynamic and distributed schedule that achieves high throughput, low packet errors and collisions, as well as low energy consumption by avoiding deafness and reducing both overhead and overhearing. In our solution, nodes do not communicate large numbers of control message as local topology construction is only based on local link exchange to construct two-hop neighborhood. To reach our goal, we chose to make use of Combinatorial Design theory, particularly Block Design. Within Block Design, we particularly focused on Latin Squares/Latin Rectangles (LS/LR), and Balanced Incomplete Block Design (BIBD). We show how we make it possible to maintain these structures within the node's memory constraints and evaluate the performance of our solution by comparing it with DiSCA [12] and show that we obtain better results in major networking metrics. Our results are based on our implementation of both protocols on NS-3.26 [13].

II. Related Work

The main purpose of a good multichannel MAC schedule is to define the cell (timeslot, channel) where nodes can meet to exchange messages and avoid as much as possible events that have negative influence on the performance of the wireless communication. Research has been extensive in this area and involved various techniques to achieves this end with different levels of sophistication. A number of methods have made use of mathematical theories and results to define cell reservation algorithms. Other methods relied on rather simpler solutions based on message exchange.

MMSN (Multi-Frequency Media Access Control for WSN) [14] is among the first multichannel protocols for sensor networks. It is a receiver-based static channel allocation protocol that avoids the use RTS/CTS (request to send/clear to send) to reduce the overhead it generates. It uses an exclusive frequency assignment to eliminate the hidden node problem by avoiding the node to have the same frequency as its second hop neighbor. Although MMSN reduces the overhead by avoiding RTS/CTS, it generates a large overhead when nodes construct their second hop neighborhood topology and when it makes two hops broadcast of its assigned channel. The static channel allocation makes MMSN suffer from a constant interferer. The snooping mechanism makes the node switch frequently between channels which consumes more energy.

QLCH (Quorum and Latin Squares Channel Hopping) [15] uses quorum system to guaranty rendezvous by defining the channel hopping sequence and Latin square to determine slots. It assumes that there are n available channel in the network, and that the time is divided to series of frames with equable size of n slots. The settlement of n value is a very critical point in QLCH protocol, because it represents the number of channels and at the same time the number of slots. To prevent intra-channel interference and reduce collisions, the number of slots per frame need to be adequate with the network's density. During the slot allocation, QLCH does not consider neither the frame length minimization to reduce the delay, nor prevent neighbors transmission at the same cell to reduce collisions, nor guaranty fairness in medium access as the number of rendezvous per frame between every pair of nodes is not the same.

In DiSCA [12], the solution is based on determining the transmission cell for every packet. Every node n determines its conflict set which contains the nodes that their transmission may collide with its own transmission. The elements of the conflict set depends on the adopted policy, either the immediate acknowledgment or no acknowledgment. Besides discovering a large neighborhood topology, DiSCA does not envisage anything to allow the joining of new nodes. The parent and children of a node n are elements of its conflict set. In DiSCA, for every allocation, the node has to broadcast it in the entire network so its conflict set should consider it, which increases highly the number of control packets.

III. Block Design-based Cell Distribution

A. Block Design Overview

Block design is a variation of combinatorial designs [16], [17]. It mainly consists in randomly gathering elements of a given set into subsets according to different parameters such as the number of subsets, and the number of elements per subset. There are many variants of block designs. The most relevant to our case are BIBD and RCBD (Row-Column Block Design) which includes Latin Squares (LS) and Latin Rectangles (LR). For a brief definition, let us consider tow positive integers v, k such that $v > k \geq 2$. A (v, b, k, r)-Balanced Incomplete Block Design is a design (X,A) such that the following properties are satisfied: (i) $|X| = v$; (ii) each block contains exactly k points. r is often called the replication number of the BIBD and b is the number of blocks. For a detailed description of these concepts refer to [18].

B. LS/LR Generation

We follow a receiver-based cell scheduling, i.e. a node wakes up either during its cells to listen to incoming traffic or its receivers cells to transmit, and goes to sleep mode outside these cells. We focus on eliminating deafness, contention and reduce collisions. We extract the Latin Rectangles from the Latin Square to make sure that all nodes have the same opportunities of being allocated to a cell independently of the number of nodes or channels in the network.

In our solution, all nodes generate the same LS/LR in a distributed way to avoid conflicting cell reservations. To do so, we start by forming a LS from which we extract a set of LRs, where a LS represents a super-slotframe (i.e. composed of a set of slotframes), and every LR represents a slotframe. To generate our final LS, we follow these steps:

1) Node classification in sets: we classify nodes in s sets using the BIBD. $D = 2 - (v, b, k, r)$ where: v is the number of source states, v = ceil(nb_nodes/nb_channels)*nb_channels is the number of source state and, b represents number of channels, k is the number number of frames, k= ceil(nb_nodes/nb_channels), r = 1 because we need that each source state appears only in one block.
2) Generation of the partial LS: after organizing nodes in sets, we will deal with block id instead of node id, to generate an initial Latin Square using the method in [15], that will give a $b * b$ matrix. After that, in the resulted LS, we replace every block i in the initial Latin Square with its partial LS. Partial LS of block i is the LS of its k elements. It is generated using the same method described in (1).

$$\text{LS}(i, j) = (i + j) \mod n \tag{1}$$

3) Extraction of LR: we could use a simple method, where every k columns represent one LR, to get b LR. If we do so, whatever the value of n is, we will have b frames, which has a very limited value. We will note that there is not a considerable change of node's position between two successive columns which makes the reservation not dynamic. Thereby, we thought to make a column permutation to increase the number of frames in a super frame. Thus, we generate k LRs instead of b to make the reservation of cells more dynamic and node finds the position of every neighbor in the LR after that it runs the statement `if` n times at most. To generate LR LR_i, we gather columns j satisfying (2).

$$j \mod \text{nb\_frames} = i \tag{2}$$

In the resulting LR, if we consider that column identifier refers to channel number, channel allocation will be almost static. For that reason, in networks of nb_channels orthogonal channels, we set Id of channel represented by the column j in the LR of frame_id is c where:

$$c = (j + \text{frame\_id}\ ) \bmod \text{nb\_channels} \tag{3}$$

We abbreviate steps listed above, and set algorithm 1 which is used by node to generate LS of nb_frames LRs in networks of nb_nodes and nb_channels. We define the function Round as the following:

$$\text{Round}(\text{nb\_nodes}, \text{nb\_frames}) = \left\lceil \frac{\text{nb\_nodes}}{\text{nb\_frames}} \right\rceil * \text{nb\_frames} \tag{4}$$

Algorithm 1: LS_Generation

Input: nb_nodes, nb_channels
Output: Latin Square matrix

```
nb_nodes ← Round (nb_nodes, nb_channels) ;
nb_frames ← nb_nodes/nb_channels;
for frame_id ← 0... nb_frames - 1 do
    for i ← 0...nb_nodes - 1 do
        for j ← 1...nb_channels do
            ls_column_id ← frame_id * nb_frames + j ;
            LS(i, ls_column_id) ←
              (Round(i, nb_frames) + (i +
              frame_id)%nb_frames + (j − 1) ∗
              nb_frames)%nb_nodes ;
        end
    end
end
```

C. Cell Allocation Description

Our algorithm started by an initial phase, in which every node sends to its first hop neighbors the list of its 1-hop neighbors, where it distinguishes between its parent and its children. The first concern of our cell reservation approach is to avoid deafness problem by preventing the simultaneous reservation from a node and its parent or its child and the reservation of the same cell by the node and its first hop neighbor which is not its parent nor its child. The second concern is to reduce idle listening by scheduling the sleep mode, so node listen to the channel only during the transmission and reception cells and spend the rest of time in sleeping. The third concern is to eliminate the contention between senders of the same receiver by using sub slots.

1) Deafness Avoidance: At first, a node runs the reservation algorithm 2 for itself and for every 1-hop neighbor. After that it broadcasts the max value of slots of every frame. So nodes will have the same number of slots in a super frame and synchronize with each other. To obtain the same reservation (adjust the predicted reservation) of a neighbor i as the reservation obtained by the neighbor i itself, the node reruns the reservation algorithm, but this time the node puts itself in the position of i (run reservation algorithm for i and for every 1-hop neighbor of i). During the reservation adjustment, in frame i, if the node finds that it receives in the same slot as its neighbor, it deletes allocation of this neighbor to prevent deafness. At the end every node re-broadcasts the max value of slots of every frame. For every frame the allocation of the node rarely coincides with the same neighbor. If this neighbor is a parent or a child, both nodes add a cell at the end of the last frame and allocate it to the parent. In our reservation algorithm, the cell slot of frame_id is reserved to node_id, only if this slot is not allocated by its parent or one of its children what ever was the channel. And if this cell is not allocated to a 1-hop neighbor of node_id which is not a parent nor a child. Therefore, deafness problem is completely avoided and

Algorithm 2: Reservation Algorithm

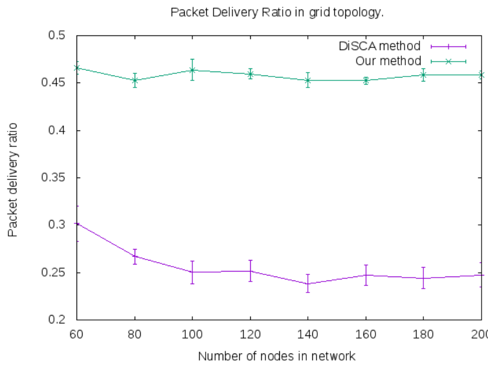
```
Input: node_id
Output: Assignment(node_id, frame_id, slot_id, channel_id)
Data: Constants: nb_nodes, nb_channels, 1st hop neighbors of node_id
n ← Round(nb_nodes, nb_channels) ;
nb_frames ← ⌈n/nb_channels⌉ ;
list(1st_hop_neighbors) ← first hop neighbors of node_id;
for frame_id ← 0 ... nb_frames -1 do
    for i ← 0 ... n - 1 do
        /* If we set the loop (j ← 1 ... nb_channels) at first, most of allocation in
           every frame will be in the channel referred by the first column of the LR.
           */
        for j ← 1 ... nb_channels do
            node_id_to_be_allocated ← (Round(i, nb_frames) + (i+frame_id)%nb_frames + (j-1)*nb_frames)%n;
            if node_id_to_be_allocated = node_id or node_id_to_be_allocated ∈ list(1st_hop_neighbors) then
                channel_id ← (column_id+frame_id)%nb_channels;
                if Assignment[frame_id][node_id_to_be_allocated] = null then
                    slot_id ← 1;
                    /* In order to minimize frame length, we start from the first slot
                       every time and jump to next slot only if the present slot is
                       reserved by its parent or its child or if the present cell is
                       assigned by its 1st hop neighbor.                              */
                    is_slot_free ← false;
                    while (is_slot_free = false) do
                        is_slot_free ← true;
                        for Every neighbor from list(1st_hop_neighbors) do
                            if (neighbor is a parent or child) AND ((slot(Assignment[frame_id][neighbor]) = slot_id) then
                                slot_id ← slot_id +1;
                                is_slot_free ← false;
                                Break;
                            end
                            else if (neighbor is not a parent nor a child) AND ( slot(Assignment[frame_id][neighbor]) =
                             slot_id And channel(Assignment[frame_id][neighbor]) = channel_id ) then
                                slot_id ← slot_id +1;
                                is_slot_free ← false;
                                Break;
                            end
                        end
                    end
                    Assignment[frame_id][node_id_to_be_allocated] = (slot_id,channel_id);
                end
            end
        end
    end
end
```

collisions are reduced.

2) Collision Avoidance: Since our method is receiver-based, the contention between transmitters toward the same receiver will be very high which increases the probability of collisions. To resolve this, we resort to devise the slot to sub-slots, where the length of the sub-slot depends on the number of children. And the local allocation of the sub-slots depends on the order of the child Id in the list of children of the receiver. For example, if a node 0 has three children with the following identifiers: $\{1, 4, 5\}$ and the length of the slot is 1 time unit, then the length of the sub-slot equals: $1/3 = 0.33$ time unit. Accordingly, node 1 (resp. 4 and 5) allocates the first (resp. second and third) sub-slot. Since every node already knows about the children of the receiver, it knows its order in children's list (its sub slot) and there is no need to redistribute the sub slot allocation. Thus every node defines its sub slot autonomously. The weakness aspect is the empty sub-slot. After allocation deletion or new slot addition, if node informs its neighbors by broadcasting, other children will not consider it during slot division. But the broadcast generates more overhead which drains more energy. For that reason, we did not use it and we recourse to not inform neighbors and scarifies latency to save energy.

D. Mobility Management

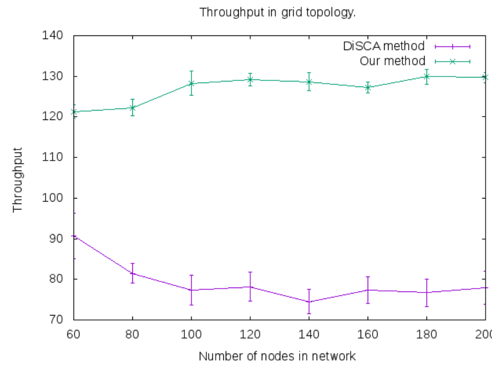
To take into account mobile or highly dynamic networks our method needs to schedule the initial phase at the beginning of every super frame. Thus, nodes can detect the neighborhood alteration and the new joining nodes. So they have to schedule the initial phase periodically to update its schedule. The same



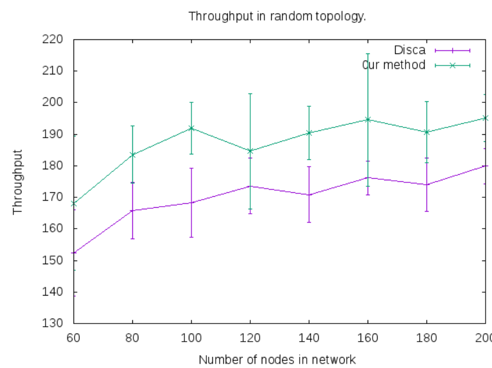
(a)

(b)

Fig. 1. Packet delivery ratio in different topologies. (a) Grid topology. (b) Random topology.



(a)



(b)

Fig. 2. Throughput in different topologies. (a) Grid topology. (b) Random topology.

principle of our method can be used to avoid hidden node problem in sender-based allocation. In this case nodes share their first and second hop neighbors. And node reserves slot_id of frame_id to node_id, only if the related cell is not allocated to a second hop neighbor of node_id.

IV. Simulation

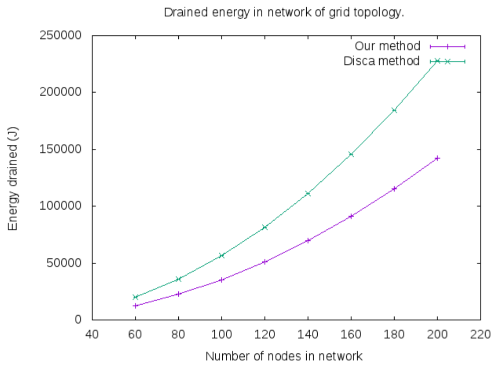
In this section we evaluate the performance of our proposal by comparing it with DiSCA due to the availability of implementation details particularly the length of the superframe which affects the performance significantly. In our simulations, we implemented both protocols in NS-3.26 [13] by modifying the existing WiFi module and implementing the multichannel mode. We used different topologies to measure the performance of our method under various situation with different network densities expressed in terms of the number of nodes per m^2. We used a grid topology with a maximum number of neighbors of 8 and that of children of 3, and a minimum number of neighbors of 3. We also used a random topology where x nodes are spread over an area of $x * 10m^2$ where the number of neighbors is not uniform as in the first grid topology. We validated the performance according to three metrics: packet delivery ratio, throughput, and power consumption without considering the initial phase.

A. Performance in Terms of Packet Delivery Ratio Metric

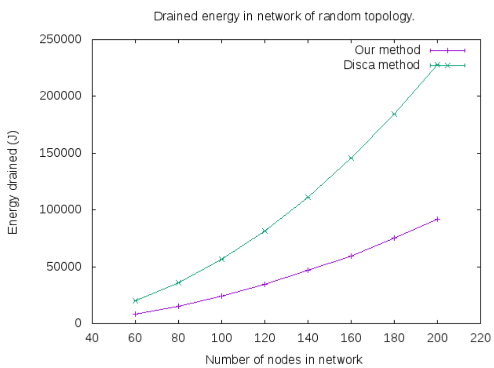
In this experiment, we calculate the ratio of the number of correctly received packets compared to the number of transmitted packets. Figs.1(a), and 1(b) represent the resulted plots from our simulation of the PDR in grid and random topology respectively. As shown, our method provides better performance than DiSCA. Collisions are the main reason why packets do not get received correctly as we did not model other adjacent channel interference. Collisions occur when two nodes choose to transmit to the same node. This is exacerbated in receiver-based cell scheduling as all traffic intended for given node gets synchronized at the beginning of its wake-up cell. However, our use of the sub slot reduces contention and reduce collisions efficiently. It also eliminates the deafness problem to increase the packet delivery ratio.

B. Performance in Terms of Throughput Metric

To evaluate the throughput we calculate the ratio of the number of correctly received bytes during the time of simulation. The throughput metric represent the symmetry between the portion of the received packet and the delay. As shown in Fig. 2(a) and 2(b), our method provides better performance than



(a)



(b)

Fig. 3. Drained energy in different topologies. (a) Grid topology. (b) Random topology.

DiSCA due to our consideration of frame length minimization in conjunction with collision avoidance (deafness and contention elimination) in our method compared to DiSCA which increases the ratio of received packets during the simulation in the case of our method.

C. Performance in Terms of Power Consumption Metric

DiSCA did not consider the energy consumption. In the Figs. 3(a) and 3(b), we remark that our method has the ultra low energy consumption in comparison to DiSCA. In channel hopping methods, the overhead of channel switching should be considered because it drains energy. The energy consumption in our method has two sides. The first one saves energy through the use of sleep mode. The second one drains energy in the switching between the different channels. The higher number of channel gives lower number of wake up slots and lower energy drain.

V. Conclusions

Our algorithm is a receiver based, dynamic, distributed, and even cell reservation based on the concepts of Block Design BIBD and Latin Squares/Rectangles. Our method does not impose constraints on the number of channels or geographic position. All that the node need to develop its schedule and those of its 1-hop neighbors is the number of nodes in the entire network and 2-hop neighborhood view. Compared to DiSCA, the obtained results from our implementation of both methods in NS-3.26 simulator show that ours is more effective than DiSCA in terms of PDR, throughput, and power drain, even in large network.

References

[1] J. A. Stankovic, “Research directions for the internet of things,” *IEEE Internet of Things Journal*, vol. 1, pp. 3–9, Feb 2014.

[2] L. D. Xu, W. He, and S. Li, “Internet of things in industries: A survey,” *IEEE Transactions on Industrial Informatics*, vol. 10, pp. 2233–2243, Nov 2014.

[3] D. Bandyopadhyay and J. Sen, “Internet of things: Applications and challenges in technology and standardization,” *CoRR*, vol. abs/1105.1693, 2011.

[4] 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, pp. 1389–1406, Third 2013.

[5] D. D. Guglielmo, S. Brienza, and G. Anastasi, “{IEEE} 802.15.4e: A survey,” *Computer Communications*, vol. 88, pp. 1 – 24, 2016.

[6] T. Watteyne, V. Handziski, X. Vilajosana, S. Duquennoy, O. Hahm, E. Baccelli, and A. Wolisz, “Industrial wireless ip-based cyber physical systems,” *Proceedings of the IEEE*, vol. 104, pp. 1025–1038, May 2016.

[7] I. Juc, O. Alphand, R. Guizzetti, M. Favre, and A. Duda, “Energy consumption and performance of ieee 802.15.4e tsch and dsme,” in *2016 IEEE Wireless Communications and Networking Conference*, pp. 1–7, April 2016.

[8] 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,” RFC 7554, RFC Editor, May 2015. http://www.rfc-editor.org/rfc/rfc7554.txt.

[9] R. Soua and P. Minet, “Multichannel assignment protocols in wireless sensor networks: A comprehensive survey,” *Pervasive and Mobile Computing*, vol. 16, pp. 2–21, jan 2015.

[10] A. Bachir, M. Dohler, T. Watteyne, and K. Leung, “Mac essentials for wireless sensor networks,” *Communications Surveys Tutorials, IEEE*, vol. 12, pp. 222–248, Second 2010.

[11] A. Hannachi and A. Bachir, “Block design multichannel mac protocol for wsns,” in *2015 First International Conference on New Technologies of Information and Communication (NTIC)*, pp. 1–6, Nov 2015.

[12] R. Soua, P. Minet, and E. Livolant, “Disca: A distributed scheduling for convergecast in multichannel wireless sensor networks,” in *2015 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, pp. 156–164, May 2015.

[13] https://www.nsnam.org. Last updated on Oct., 2016.

[14] G. Zhou, C. Huang, T. Yan, T. He, J. Stankovic, and T. Abdelzaher, “Mmsn: Multi-frequency media access control for wireless sensor networks,” in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, pp. 1–13, April 2006.

[15] C.-M. Chao and H.-Y. Fu, “Providing complete rendezvous guarantee for cognitive radio networks by quorum systems and latin squares,” in *Wireless Communications and Networking Conference (WCNC), 2013 IEEE*, pp. 95–100, April 2013.

[16] D. Stinson, *Combinatorial Designs*. Springer, 2004.

[17] C. Colbourn and J. Dinitz, *Handbook of Combinatorial Designs, Second Edition*. Discrete Mathematics and Its Applications, CRC Press, 2007.

[18] M. Huber, *Combinatorial Designs for Authentication and Secrecy Codes*. Foundations and trends in communications and information theory, Now Publishers, 2010.