A Distributed Joint Channel and Time Slot Assignment for Convergecast in Wireless Sensor Networks

Ridha Soua, Pascale Minet, Erwan Livolant

To cite this version:

Ridha Soua, Pascale Minet, Erwan Livolant. A Distributed Joint Channel and Time Slot Assignment for Convergecast in Wireless Sensor Networks. the Sixth IFIP International Conference on New Technologies, Mobility and Security, Mar 2014, Dubai, United Arab Emirates. pp.5. hal-00939005

HAL Id: hal-00939005

https://hal.archives-ouvertes.fr/hal-00939005

Submitted on 10 Feb 2014

A Distributed Joint Channel and Time Slot Assignment for Convergecast in Wireless Sensor Networks

Ridha Soua, Pascale Minet, Erwan Livolant
INRIA Rocquencourt, 78153 Le Chesnay cedex, France
Email: firstname.name@inria.fr

***Abstract*—In this work, we study raw convergecast in multichannel wireless sensor networks (WSNs) where the sink may be equipped with multiple radio interfaces. We propose $Wave$, a simple and practical distributed joint channel and time slot assignment. We evaluate the number of slots needed to complete the convergecast by simulation and compare it to the optimal schedule and to a centralized solution.**

I. Context and motivations

A wireless sensor network (WSN) is typically composed of many tiny low-cost low-power on-chip sensors. The individual devices sense the surrounding environment and send their data, directly or via multiple hops, to a central device, namely the sink for processing. This many-to-one communication pattern is called raw data convergecast when intermediate nodes do not aggregate packets. Most of data gathering applications have strict latency requirements. However, the intrinsic characteristics of WSNs such as limited bandwidth and scarce energy budget coupled with unreliable wireless links, channel contention and interferences, raise great challenges with regard to end-to-end delays. One of the main reasons for increased delays is retransmissions due to collisions. In [1], authors show that using multiple channels is more efficient than transmission power control. Indeed, distributing transmissions across multiple channels enhances channel spatial reuse and decreases the convergecast delay. In addition, resorting to multichannel communication mitigates interferences by avoiding bad channels and hence reduces packet losses.

On the other hand, contention-free access protocols, such as Time Division Multiple Access (TDMA), eliminate collisions by assigning different time slots to conflicting nodes. Therefore, these protocols are able to deliver packets with deterministic delay bounds.

Furthermore, in their allocated slots, nodes can transmit (respectively receive) data to their parents (respectively from their children). This deterministic schedule allows nodes to turn off their radio during time slots where they are not involved. Hence, contention-free protocols eliminate overhearing, and idle listening, which are the main sources of energy depletion. Notice that WirelessHART [2], frequently adopted in control applications, uses TDMA to control medium access.

Recently, the MAC amendment, $IEEE$ 802.15.4e TSCH [3] adds functionality to better meet industrial markets requirements. The TimeSlotted Channel Hopping (TSCH) mode ensures robustness and high reliability against interferences by channel hopping. $IEEE$ 802.15.4e highlights how the MAC layer executes the nodes schedules. Nevertheless, the policy that provides such scheduling is not specified.

In this paper, we propose $Wave$, a simple, practical and traffic-aware distributed joint channel and time slot assignment for convergecast. Our target is to minimize the data gathering delays and ensure that all packets transmitted in a cycle are delivered to the sink in this cycle, assuming no packet loss at the physical layer.

II. Related work

The focus on the state of the art will be limited only to distributed joint channel and slot assignment approaches in WSNs. Authors of [4] tackle data intensive applications for WSNs. Their proposal GLASS, a distributed conflict-free schedule access, operates in three phases: first, each sensor associates itself with one virtual grid cell. Then, GLASS assigns different sets of time slots to nodes that belong to adjacent cells. Finally, the Latin square matrices are run to assign time slots for nodes in the same grid cell. Nevertheless, GLASS does not meet the requirements of raw data convergecast: only a single slot is granted per node.

Incel et al [5] propose a TDMA-based schedule that minimizes the number of slots required for convergecast. They extend the work of Ghandam et al. [6] to multichannel WSNs. Their approach includes two steps:
(1) a receiver based channel assignment that aims to remove all the interference links. Thus, the only remaining conflicts are inside the convergecast tree. (2) a distributed slot assignment: where each node is assigned an initial state (i.e. transmit T_x, receive R_x or idle) based on its hop-count to the sink and the state of its branch. To eliminate conflict between two children of the same node, the algorithm assumes that any node should know the number of remaining packets for its brothers to schedule them in round robin order.

In [7], authors combine the use of an access hash function with the inductive scheduling technique proposed by Kumar et al [8] to assign jointly channels and time slots to nodes. Their proposed technique, called CLDS (Collision-free Distributed Scheduling), allows each node to know if it will communicate or not with its neighbor on a specific channel and in a given time slot, assuming that nodes have previously exchanged their links utilization with their interfering nodes.

Authors of [9] propose DeTAS, a distributed traffic aware scheduling solution for $IEEE$ 802.15.4e $TSCH$ networks. This solution is the distributed mode of TASA proposed in [10]. In DeTAS, all nodes follow a common schedule, called macro-schedule, that is the combination of micro-schedules of each routing graph. Authors claim that their solution is optimum. However, this is not true when the sink child needs at least two slots to transmit its own data packets due to the alternation of transmit and receive slots, even there is no more packets to receive.

III. Problem statement

The joint channel and slot assignment problem for raw data data convergecast in multichannel WSNs consists in assigning each node different from the sink, the number of couples (time slot, channel) needed by the data gathering. We focus on algorithms whose goals are to:

- *produce a valid time slot and channel assignment.* By definition a valid schedule ensures that 1) no two conflicting transmissions are scheduled in the same time slot and on the same channel and 2) no transmission is scheduled when either the sender or the receiver has no available interface in this time slot.
- *take advantage of multichannel communications* to reduce interferences and enable parallel transmissions on different channels.
- *reduce data gathering delays.* A shorter delay reduces the activity period and allows nodes to sleep longer to save energy, while meeting the application requirements.
- *ensure that all data produced by sensor nodes in a cycle are delivered to the sink at the latest in the next cycle.* This ensures freshness and a better time consistency of data gathered.

To this end, the slot and channel assignment must first minimize the total number of slots assigned for the data gathering and second schedule the transmissions in an order that ensures that any data collected in a cycle is delivered to the sink in the same cycle, with the following assumptions.

∘ $A1$: The network topology and the routing tree associated with the raw data gathering are given. Notice that additional links to the routing tree may exist in the topology creating additional conflicts.

∘ $A2$: The topology of the network may differ from one channel to another. Connectivity is assumed on any channel.

∘ $A3$: Any node $u \neq sink$ generates locally $Gen(u) > 0$ packets per cycle and these $Gen(u)$ packets are present in the buffer of u at the beginning of the raw data gathering. In addition, to theses packets, node u forwards the packets received from its children; The total number of packets transmitted by u, in a data gathering cycle, is denoted $Trans(u)$. $Trans(u) = \sum_{v \in Subtree(u)} Gen(v)$.

∘ $A4$: The size of the time slot is fixed to enable the transmission of one packet.

Modeling interferences for data gathering: Any node $u \neq sink$, should determine $Conflict(u)$ the set of nodes whose transmission conflicts with its own on the same channel and in the same time slot.

Property 1: In a raw data convergecast and in the absence of acknowledgment, $Conflict(u)$ contains for any node $u \neq sink$, the node itself, its parent, its children, all nodes that are 1-hop away from its parent and all nodes whose parent is 1-hop away from u.

Proof: On the same channel and in the same time slot, any node $u \neq sink$ cannot:

- transmit in parallel; hence it conflicts with itself;
- transmit and receive; hence $Parent(u)$ cannot transmit while it is receiving from u. Similarly u cannot receive from its children, while it is transmitting to its parent.
- receive from two different nodes; hence $Parent(u)$ cannot receive from u and a node v one-hop away from $Parent(u)$. Similarly a node $Parent(v)$ 1-hop away from u cannot receive simultaneously from u and v.

■

IV. $Wave$ Algorithm

A. $Wave$ in centralized mode

The $Wave$ algorithm schedules nodes in successive waves. In each wave, each node having a packet to transmit is assigned a time slot and a channel. The first wave constitutes the (slot x channel) pattern. Each next wave is an optimized subset of the first wave: only the slots that will contain transmissions are repeated and they always occur in the same order as in the first wave. As a result, the joint channel and time slot assignment produced by $Wave$ contains for each time slot and for each available channel, a list of sender nodes, such that their transmissions to their parent do not conflict.

More precisely, the $Wave$ algorithm proceeds as follows:

- *Building of the first wave*: Each node u has a priority equal to $Trans(u)$ the number of slots needed to transmit its own data and the data received from its children. $Wave$ schedules nodes in the order of their decreasing priority, granting them the smallest available time slot and the first available channel (channels being visited according to a round-robin strategy) (see Algorithm 2). Let P be the (slot x channel) pattern obtained and T be the number of time slots in P.
- *Computation of the next waves*: $Wave$ repeats the pattern P of the first wave a number of times equal to W. However, the repeated pattern is optimized according to the following rule.

Rule R: Any node u having slot j and channel ch in the first wave, with $1 \leq j \leq T$, has also the slot $s(k)$ and channel ch in any wave k with $1 \leq k \leq Trans(u)$,

$s(k) = \sum_{w=1}^{k-1} \sum_{t=1}^{T} \delta_{t,w} + \sum_{t=1}^{j} \delta_{t,k}$, where $\delta_{t,w} = 1$ if and only if $Maxtrans(t) \geq w$ and 0 otherwise, where $Maxtrans(t)$ is the maximum number of transmissions of any node transmitting in the slot t.

The $Wave$ algorithm produces W successive waves, where W is equal to the maximum number of transmissions done by a node: $W = max(Trans(u))$. In each wave, each node having not acquired all its slots requested, is granted exactly one time slot. The assignment produced by $Wave$ contains exactly $\sum_{1\leq t\leq T} Maxtrans(t)$ slots.

Let us consider the example depicted in Figure 1 with a pattern of 5 slots, $P = A\ B\ C\ D\ E$ such that $Maxtrans(A) = 6$, $Maxtrans(B) = 5$, $Maxtrans(C) = 3$, $Maxtrans(D) = 2$ and $Maxtrans(E) = 1$. In this figure, the number inside the slot indicates the maximum remaining transmissions of nodes scheduled in this slot at the beginning of the i^{th} wave. The schedule provided by the $Wave$ algorithm contains 6 waves and 17 slots that are $A\ B\ C\ D\ E$, $A\ B\ C\ D$, $A\ B\ C$, $A\ B$, $A\ B$, A.

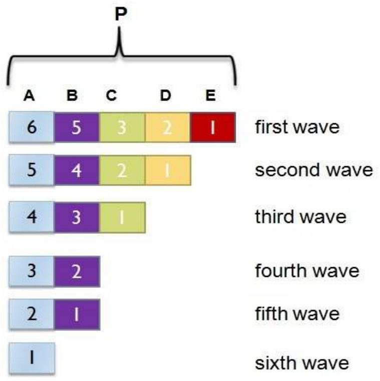


Fig. 1. Illustrative example for the $Wave$ algorithm

Algorithm 1 ScheduleNode Function (v, $Conflict(v)$, $nchannel$, $ScheduledNodes$)

```
1: Input: node $v$, set of conflicting nodes $Conflict(v)$, $nchannel$ channels and table of scheduled nodes $ScheduledNodes$ per slot and channel.
2: Output:
3: - $slot$: the slot assigned to node $v$
4: - $ch$: channel allocated to node $v$
5: - $ScheduledNodes$: scheduled nodes per slot and channel.
6: Initialization:
7: $ch \leftarrow 1$
8: $tx \leftarrow false$
9: $nChannelReached \leftarrow false$
10: $t \leftarrow 0$
11: while ($AvailInter(v,t) = 0 \,||\, AvailInter(Parent(v),t) = 0 \,||\, tx = false$) do
12:     /*find a time slot with available interface for $v$ & $Parent(v)$*/
13:     $t \leftarrow t + 1$
14:     repeat
15:         if ($Conflict(v) \bigcap ScheduledNodes(t,ch) = \emptyset$) then
16:             /*Node v can be scheduled */
17:             $tx \leftarrow true$;
18:             Node $v$ transmits in slot $t$ on channel $ch$
19:             $AvailInter(v,t) \leftarrow AvailInter(v,t) - 1$
20:             $AvailInter(Parent(v),t) \leftarrow AvailInter(Parent(v),t) - 1$
21:             $ScheduledNodes(t,ch) \leftarrow ScheduledNodes(t,ch) \cup \{v\}$
22:         else
23:             if ($ch < nchannel$) then
24:                 $ch \leftarrow ch + 1$ // try the next channel
25:             else
26:                 $nChannelReached \leftarrow true$
27:                 $tx \leftarrow$ false
28:             end if
29:         end if
30:     until ($tx \,||\, nChannelReached$ )
31: end while
32: return ($t$, $ch$, $ScheduledNodes$)
```

Algorithm 2 Wave algorithm: first wave

```
1: Input: $nchannel$ channels; a routing tree $T$ where each node $u$ has a set of conflicting nodes $Conflict(u)$, a number of available radio interfaces $AvailInter(u)$ and a priority $Trans(u)$=number of packets that node $u$ should transmit.
2: Output: $ScheduledNodes$: Channel and time slot assignment for all nodes
3: Initialization:
4: $scheduledNodes \leftarrow \emptyset$
5: Channel and slot assignment for all nodes:
6: $SortedNodesList \leftarrow$ nodes sorted by decreasing priorities
7: $u \leftarrow first(SortedNodesList)$
8: while $SortedNodesList \neq \emptyset$ do // there are nodes to schedule
9:     ($slot$,$ch$,$ScheduledNodes$)=ScheduleNode($u$,$Conflict(u)$, $nchannel$,$ScheduledNodes$)
10:    remove($u$) from $SortedNodesList$
11:    $u \leftarrow next(u)$
12: end while // no pending demand
```

B. *Wave in distributed mode*

We now focus on the distributed mode of $Wave$. The main difference between the two modes concerns the knowledge required. In centralized mode, the central entity running $Wave$ has the knowledge of any information related to each node. In distributed mode, any node u knows only the information related to its set of conflicting nodes. To acquire this information, messages are exchanged between neighboring nodes. Furthermore, a specific message, called $SlotChAssigned$, is created to notify the neighboring nodes of a (time slot, channel) assignment (see Algorithm 4). In distributed mode, the global time slot and channel assignment is built by assembling all the partial assignments known by nodes. More practically, to compute the first wave, it is sufficient for each node u to know: its conflicting nodes $Conflict(u)$ and the number of transmissions $Trans(v)$ of each node $v \in Conflict(u)$.

Then any node u sends to its parent the maximum number of transmissions for each slot it knows. At the end of the first wave, the sink computes T the total number of slots in the pattern and for each slot t with $1 \leq t \leq T$, its maximum number of transmissions $Maxtrans(t)$. It sends this information to all nodes via the routing tree. Nodes are now able to compute the next waves according to Rule R.

Algorithm 3 processRecSlotChAssigned Procedure ($SlotChAssigned$)

```
1: Input: message $SlotChAssigned(slot, ch, v, Parent(v))$
2: if $v \in Conflict(u)$ then
3:     update $ScheduledNodes$
4: else if $v = Child(u)$ then
5:     update $AvailInterf(u, slot)$
6:     update $EarliestPcktSlot(u)$
7: else if ($v = Parent(u)$) || ($Parent(v) = Parent(u)$ & $v \neq u$) then
8:     update $AvailInterf(Parent(u), slot)$
9: end if
```

C. *Properties*

Under the assumptions given in Section III, the $Wave$ algorithm exhibits the following properties:

Algorithm 4 Wave algorithm: distributed mode

```
1: Input: $nchannel$ channels; a routing tree $T$ where the local node $u$ has a set of conflicting nodes $Conflict(u)$, a number of available radio interfaces $AvailInter(u)$ and a priority $Trans(u)$=number of packets that node $u$ should transmit.
2: Output: $ScheduledNodes$: Channel and time slot assignment for local node $u$ and $Conflict(u)$ per slot and channel
3: Initialization:
4: $Slot(u) \leftarrow 0$ /*number of slots already assigned to node $u$*/
5: Channel and slot assignment for node $u$
6: $SortedNodesList(u) \leftarrow$ the set $\{u\} \cup Conflict(u)$ sorted by decreasing priorities.
7: if reception of a $SlotChAssigned(slot, ch, v, Parent(v))$ Message then
8:     $processRecSlotChAssigned()$ /* see algorithm 3
9:     if ($u$ has child) & ($u$ is 1-hop away from $v$) then
10:        forward $SlotChAssigned$
11:    end if
12: end if
13: if ($Slot(u) < Trans(u)$ ) & ( $\forall v \in SortedNodesList(u)$ such that $Trans(v) > Trans(u)$, $Slot(v) > Slot(u)$) then
14:    ($slot$,$ch$,$ScheduledNodes$)=ScheduleNode($u$,$Conflict(u)$, $nchannel$, $ScheduledNodes$)
15:    $Slot(u) \leftarrow Slot(u) + 1$
16:    transmission of the $SlotChAssigned(slot, ch, u, Parent(u))$ Message to the 1-hop neighbors of $u$.
17: end if
```

Property 2: Equivalence of centralized and distributed modes: For any considered topology of WSN, for any raw data convergecast and for any data traffic, the distributed and the centralized modes of $Wave$ provide the same slot and channel assignment.

Proof: First, we prove that both modes of $Wave$ produce the same first wave. All nodes $\neq$ sink have at least one packet to transmit. Hence, they are all scheduled in the first wave in both modes. We have only to prove the equality of the assignments produced. We proceed by contradiction. Let t be the first slot where the assignments differ: there is a node v that is scheduled in slot t and on channel ch by one mode and in slot t' and on channel ch' by the other mode, with $t' > t$ or $ch \neq ch'$. Whatever the mode considered, if node v is scheduled in slot t and on channel ch, it means that no node $w \in Conflict(v)$ is scheduled in slot t and on channel ch. Since both modes schedule nodes in $Conflict(v)$ in the same priority order and assign them the first available channel in the current time slot, they should reach the same decision: a contradiction. Since both modes apply the same rule to provide the next wave from the first one, they provide the same assignment. ■

Property 3: Efficiency: $Wave$ ensures that:

- no slot is empty;
- if the transmission of a node is scheduled in a given slot, this node has a message to transmit;
- if a packet is transmitted in a data gathering cycle, it reaches the sink in this cycle.

Proof: According to the principles of the algorithm, only nodes having at least one packet to transmit are scheduled in the current wave. Furthermore, the number of remaining packets of a node at the beginning of a wave is such that it reaches 0 only when this node has completed all the transmissions required by the convergecast. We deduce that no slot is empty. Furthermore, in each wave, the transmitted messages progress toward the sink. The number of waves W is computed to allow any transmitted message to reach the sink. Hence the property. ■

Property 4: Simplicity and adaptivity: $Wave$ easily adapts to traffic changes.

Proof: When the traffic changes, the first wave is kept. Only the computation of the next waves is redone, preserving the initial pattern while taking into account the new values of $Trans(u)$ for any node $u \neq sink$. Hence the property. ■

V. Performance evaluation

With our simulation tool based on GNU Octave [11], we evaluate the performance of $Wave$ (i.e. number of slots required) in various configurations and compare it to the optimal one [12] and a centralized solution MODESA [13]. In this latter, the sink computes the dynamic priority of nodes in each slot then builds an interference-free schedule for convergecast. The number of nodes ranges from 10 to 100. Node 1 denotes the sink, equipped with 1, 2 or 3 radio interfaces. All other nodes have a single radio interface and generate heterogeneous traffic toward the sink. The number of available channels is 2 or 3. The random tree topology is generated according to the Galton-Watson branching stochastic process: any node has at most 3 children. Each result depicted is averaged on 20 simulations for topologies with less than 30 nodes and on 100 for others. We will distinguish two types of configurations: T_t configurations where the number of slots is dictated by the most demanding subtree and T_n configurations otherwise.

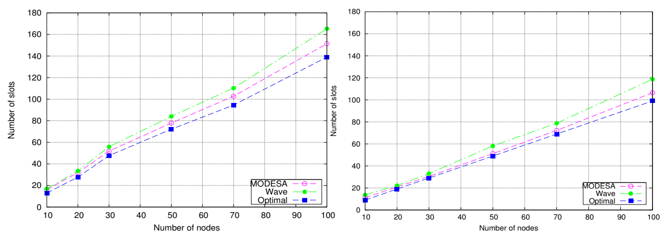
- Comparison with optimal schedule and MODESA: homogeneous traffic

Assuming that any node generates one packet and 2 channels are available at each node, we compare the number of slots provided by the optimal schedule, MODESA and $Wave$. We notice that T_t configurations are more greedy than T_n ones: T_t configurations of 100 nodes need in average 175 slots with $Wave$ while T_n need only 120 slots (see Figure 2. $Wave$ is 18% (respectively 17%) away from the optimal in T_t configurations (respectively T_n configurations). Besides, $Wave$ is slightly less worse than MODESA but much simpler. MODESA is 11% (respectively 10%) away from the optimal in T_t configurations (respectively T_n configurations) as depicted in Figure 2.

- Impact of additional links:

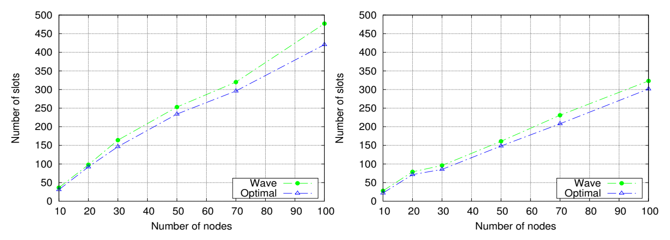
With the same previous parameters, we add other links: for each node at even depth d in the tree, an additional link is generated with a node at depth $d-1$ different from its parent. Furthermore, with a probability equal to 0.5, another link is added with a node of depth $d+1$ different from its children. In average, 60% additional links are added.

The existence of topology links that do not exist in the routing tree induces more conflicts. Hence the possibilities for parallel transmissions are reduced leading to higher number of slots: for 100 nodes, added links result on 8% (respectively



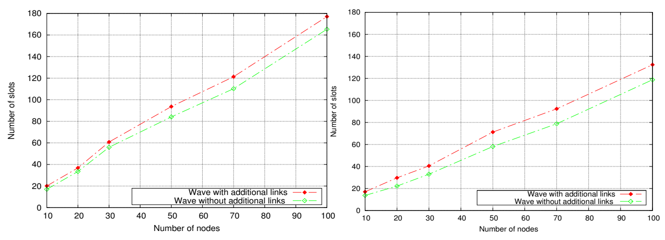
(a) in T_t configuration (b) in T_n configurations

Fig. 2. $Wave$ versus optimal schedule and MODESA: homogeneous traffic



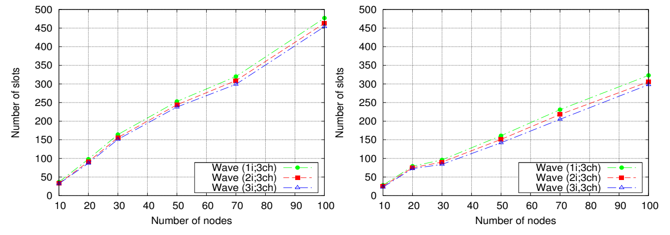
(a) in T_t configuration (b) in T_n configurations

Fig. 4. $Wave$ versus optimal schedule: heterogeneous traffic



(a) in T_t configuration (b) in T_n configurations

Fig. 3. $Wave$ with additional interfering links



(a) in T_t configuration (b) in T_n configurations

Fig. 5. Impact of multiradio sink on number of slots.

11%) extra slots in T_t configurations (respectively T_n configurations).

• Comparison with optimal schedule: heterogeneous traffic Each node generates a random number of packets between 1 and 5. The sink is equipped with one radio interface and three channels are available at each node. We notice in Figure 4 the same trend of curves as Figure 3 when nodes generate a single packet. $Wave$ is 13% (respectively 11%) away from the optimal in T_t configurations (respectively T_n configurations).

• Impact of the number of sink interfaces and channels: As illustrated in Figure 5, the number of slots to complete convergecast is reduced and so we have shorter convergecast latency. Indeed, when passing from $(1i;3ch)$ to $(3i;3ch)$, the number of slots is reduced by 6% (respectively 13%) in T_t configurations (respectively T_n configurations). This can be explained by the fact that in T_t configurations, the dominating subtree is the only subtree scheduled in the last time slots so the number of radio interfaces has no effect. Nevertheless, in T_n configurations, the demand is balanced among subtrees and the sink can receive simultaneously from its children even in the last time slots.

VI. Conclusion

In this paper we have proposed $Wave$, a distributed joint time slot and channel assignment for raw convergecast in WSNs. $Wave$ is simple to implement and efficient. Simulations results indicate that our heuristic is not far from the optimal bound for raw convergecast. Unlike most previously published papers, $Wave$ does not suppose that all interfering links have been removed by channel allocation. In addition, $Wave$ is able to easily adapt to traffic changes [12]. $Wave$ could be used to provide the schedule applied in the $802.15.4e$ TSCH based networks. In the future, we will extend this work considering that some nodes can play the role of aggregator.

References

[1] O. D. Incel,A. Gosh, B. Krishnamachari, K. Chintalapudi, *Fast data Collection in Tree-Based Wireless Sensor Networks*, IEEE Transactions on Mobile computing, vol. 1, pp. 86-99, 2012.

[2] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, M. Nixon, *WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control*, RTAS'08, MO, United States, April, 2008.

[3] Institute of Electrical and Electronics Engineers (IEEE). 802.15.4*e*-2012: IEEE Standard for Local and Metropolitan Area Networks-Part 15.4: Low Rate Wireless Personal Area Networks (LR-WPANs) Amendment 1: MAC Sublayer, April 16, 2012.

[4] Ch. Lin, V I. Zadorozhny, P.V. Krishnamurthy, Ho. Park, Ch. Lee, *A Distributed and Scalable Time Slot Allocation Protocol for Wireless Sensor Networks*, IEEE Transactions on mobile computing, Vol. 10, No. 5, April 2011.

[5] O.D. Incel, A. Gosh, B. Krishnamachari, K. Chintalapudi, *Multi-Channel Scheduling for Fast Convergecasst in Wireless Sensor Networks*, USC CENG Technical report CENG-2008-9.

[6] S. Ghandham, Y. Zhang, and Q. Huang, *Distributed Time-Optimal Scheduling for Convergescast in Wireless Sensors Networks*, Computing Network, vol. 52, no. 3, pp. 610-629, 2008.

[7] Bo. Han, V.S.A. Kumar, M.V. Marathe, S. Parthasarathy, A. Srinivasan, *Distributed Strategies for Channel Allocation and Scheduling in Software-Defined Radio Networks*, IEEE Infocom, April 2009, Rio de Janeiro

[8] V. S. A. Kumar, M. V. Marathe, S. Parthasarathy, and A. Srinivasan, *Algorithmics aspects of capacity in wireless sensors networks*, In proceedings of SIGMETRICS 2005, pp. 133-144, 2005.

[9] N. Accettura, M. R. Palattela, G. Boggia, L. A. Grieco, M. Dohler, Decentralized Traffic Aware Scheduling for Multi-hop Low Power Lossy Networks in the Internet of Things, WoWMoM'13, Madrid, Spain, 2013.

[10] M. Rita, N. Accettura, M. Dohler, L. A. Grieco, and G. Boggia, *Traffic Aware Scheduling Algorithm for reliable low-power multi-hop IEEE 802.15.4e networks*, PIMRC'12, Sydney, Australia, 2012.

[11] http://www.gnu.org/software/octave/

[12] R. Soua, E. Livolant, P. Minet, *Adaptive Strategy for an Optimized Collision-Free Slot Assignment in Multichannel Wireless Sensor Networks*, Journal of Sensor and Actuator Networks, Special Issue on Advances in Sensor Network Operating Systems, July 2013.

[13] R. Soua, P. Minet, E. Livolant, *MODESA: an Optimized Multichannel Slot Assignment for Raw Data Convergecast in Wireless Sensor Networks*, IPCCC 2012, Austin, Texas, December 2012.