DiSCA: a Distributed Scheduling for Convergecast in Multichannel Wireless Sensor Networks

Ridha Soua, Pascale Minet, Erwan Livolant

DiSCA: a Distributed Scheduling for Convergecast in Multichannel Wireless Sensor Networks

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

***Abstract*—The new $IEEE$ 802.15.4e standard does not specify how the schedule of medium accesses followed by wireless sensors is built. That is why, we propose a distributed interference-aware joint channel and time slot assignment, called DiSCA, for a traffic-aware convergecast in multichannel wireless sensor networks (WSNs). Unlike most previous studies, we consider two cases of transmissions: without acknowledgment and with immediate acknowledgment. We provide the minimum bound on the number of time slots needed for a convergecast with a sink equipped with multiple radio interfaces. Simulations results show that DiSCA is close to the optimal in terms of the number of slots and outperforms TMCP.**

I. Introduction

Convergecast is a relevant pattern of communication in Wireless Sensor Networks (WSNs): every node plays the role of data source and possibly of router node through a routing tree to deliver packets to the sink. Furthermore, in some scenarios, delay constraints for data packets do not allow intermediate processing on traffic in transit, like compression or aggregation. This data collection is called raw data convergecast. In this context, nodes that are near the sink should forward more packets than sensors far away. Hence, the medium access should be scheduled to take traffic load into account.

Two key issues for data convergecast raise: (1) minimized latencies and guaranteed packet delivery (2) energy saving. Minimized end-to-end delays ensure freshness of collected data. Besides, guaranteed packet delivery leads to a more accurate monitoring. As convergecast involves a large number of sensors that may transmit simultaneously, collisions represent a major challenge for bounded latencies. Indeed, collisions lead to data losses. Retransmissions increase packet latency and result on non-deterministic packet delivery times. Unlike contention-based protocols which suffer from inefficiency due to backoff and collisions, collision-free protocols guarantee bounded latencies. In fact, these protocols ensure that any transmission of a node does not interfere with any other simultaneous transmission. It is achieved by allocation of channels and time slots to nodes in such a way that these interferences are avoided. In addition, a node is active only when it is transmitting to its parent or receiving from its children. Nodes turn off their radio otherwise in order to save energy.

Limiting factors for a fast data collection are interferences. Incel et al [1] proved that scheduling parallel transmissions using multiple channels is more efficient than power control. Basically, multichannel communications enable concurrent transmissions on different channels. Hence, they drastically reduce the data gathering delays.

The new standard $IEEE$ 802.15.4e proposed the TimeSlotted Channel Hopping (TSCH) [2] mode where nodes perform channel hopping. This latter, combined with slotted medium access, ensures collision-free communications and high reliability against interferences. However, the standard does not propose a mechanism to built a traffic-aware schedule that assigns a channel and a time slot to each transmission. In this paper, we cover this gap by proposing a distributed collision-free scheduling algorithm that jointly optimizes the channel and time slot assignment used by data gathering applications.

II. Related work

We can distinguish two approaches to remove interferences between concurrent transmissions in the convergecast routing tree. The first approach consists of two distinct phases:
(1) Channel allocation: channels are allocated to nodes (usually receivers) or links.
(2) Time slot assignment: interferences, that are not removed by channel allocation, are avoided by assigning different time slots to concurrent senders.

The second approach deals jointly with the time and frequency allocation. This is the approach chosen in this paper.

Hereafter, we will only focus on distributed algorithms for channel and slot assignment in WSNs. Such algorithms are considered to be more scalable and reliable than centralized ones. For centralized solutions, the interested reader can refer to [1], [3], [4], [5].

Recently, many studies have resorted to multichannel communications in order to deliver collected data in a short time. In [6], Incel et al. derive a TDMA schedule that minimizes the number of slots required for convergecast. They prove that a lower bound for raw convergecast is $max(2 * n_k - 1, N)$ where n_k is the maximum number of nodes on any subtree and N is the number of nodes in the network. In [7], we extend this work considering heterogeneous traffic demands and a sink equipped with multiple radio interfaces. In this current work, we will compute these bounds for the immediate acknowledgment policy. The solution proposed in [6] includes two steps:
(1) a receiver based channel assignment: it removes all the interference links in an arbitrary network.
(2) a distributed slot assignment: each node is assigned an

initial state (i.e. transmit, receive, idle) based on its hop-count to the sink and the state of its branch (active or not).
The algorithm also assumes that either the parent orchestrates the schedule of its children or any node should know the number of remaining packets for each of its brothers. Besides, the authors assume that after channel allocation, the only remaining conflicts are inside the convergecast tree.

In [8], a distributed joint channel allocation of links and packet scheduling for Software-Defined Radio, called CLDS (Collision-free Distributed Scheduling) combines the use of an access hash function with the inductive scheduling technique. The hash function SHA-1 allows CLDS to know if a pair of nodes will communicate or not on a channel and in the current time slot. In addition, to be collision-free, a node needs to exchange its links utilization with its interfering nodes. However, CLDS is not designed for convergecast. Hence, the end-to-end delays can be large due to an assignment of slots that does not take into account the progression of data toward the sink.

Authors of [9] propose DeTAS, a distributed traffic aware scheduling solution for $IEEE\ 802.15.4e\ TSCH$ networks. In DeTAS, all nodes follow a common schedule, called macro-schedule, that is the combination of micro-schedules of each destination-oriented tree graph rooted at the sink. Micro-schedules minimize the number of slots needed to deliver data to the sink. Their solution provides the smallest number of slots for a sink with a single radio interface. However, DeTAS can only operate correctly when all interfering links that do not belong to the convergecast tree are eliminated.

In [10], we propose a simple distributed algorithm, called Wave, based on successive waves, whose complexity is that of a coloring algorithm. The macro-schedule consists of a juxtaposition of the micro-schedules provided by each wave. To reduce the number of slots needed by raw data gathering, we will propose in this paper an optimized distributed scheduling algorithm that allows micro-schedules to overlap.

This algorithm, called DiSCA, is self adpative to various raw data convergecast features, such as homogeneous or heterogeneous traffic generated per node, possible presence of topology links in addition to those present in the routing tree, number of sink's children in the routing tree, number of radio interfaces of the sink and number of available channels. In addition, we also compute theoretical bounds for raw data convergecast in case of immediate acknowledgment.

III. Joint channel and slot assignment problem

The joint channel and slot assignment problem in raw data gathering supported by a multichannel WSN consists in assigning to each node a time slot and a channel for each of its transmissions toward the sink. All these transmissions are needed to collect all the data produced by sensor nodes under the assumptions given hereafter. This assignment must minimize the number of slots allocated while ensuring that all data produced by sensor nodes are delivered to the sink in a single data gathering frame.

A. Definitions

We first introduce some definitions:

Definition 1: We focus on raw data convergecasts in a WSN where $nchannel > 1$ channels are available at each node.

Definition 2: We define $ninterf$ as the number of radio interfaces of the sink, $ninterf \geq 1$. Any other node has only one radio interface.

Definition 3: For any node u, we define $Parent(u)$ the parent of u in the routing tree. We denote $Subtree(u)$ the subtree of the routing tree that is rooted at node u. Let $nchild$ denote the number of children of the sink.

Definition 4: For any node u, we define $Gen(u)$ the number of packets generated by u to transmit its own data produced in a data gathering frame. This number of packets may differ from one node to another, that is why we speak of heterogeneous traffic. Similarly, $Trans(u)$ denotes the total number of packets transmitted by u in a data gathering frame. This number includes $Gen(u)$ packets and the number of packets received from its children that are forwarded to its parent. Hence, $Trans(u) = \sum_{v \in Subtree(u)} Gen(v)$.

Definition 5: For any node u, we define $Conflict(u)$ the set of nodes that are not allowed to transmit in the same slot and on the same channel as u. Notice that in a raw data convergecast, all the data transmissions are from a node to its parent in the routing tree.

Definition 6: We consider two policies of acknowledgment: either there is no acknowledgment or an immediate acknowledgment is sent in the same time slot and on the same channel as the packet it acknowledges.

B. Assumptions

In this paper, we adopt the following assumptions:

Assumption 1: The network topology and the routing tree associated with the data gathering are given. Since the links of the routing tree are used upstream to gather user data and downstream to broadcast the computed schedule to all nodes, these links must be symmetric. In addition, if the immediate acknowledgement policy is adopted, symmetric links are required.

Assumption 2: For any node $u \neq sink$, $Gen(u) > 0$ and these $Gen(u)$ packets are present in the buffer of u at the beginning of the raw data gathering frame.

Assumption 3: For the sake of simplicity, the size of time slots is constant and enables the transmission of one packet and its acknowledgement if any.

C. Modeling interferences for data gathering

For any node u, the determination of $Conflict(u)$ depends on the policy used for the acknowledgment (immediate acknowledgment or no acknowledgment).

Lemma 1: In data convergecast and in the absence of acknowledgment, $Conflict(u)$ contains:

a) the node u itself,
b) the node $Parent(u)$,
c) all the children of u,
d) all the nodes that are 1-hop away from $Parent(u)$,

e) all the nodes whose parent is 1-hop away from u.

Proof: Any node $u \neq sink$ cannot on the same channel simultaneously:

- transmit to different nodes: case a in Figure 1.
- transmit and receive: case c in Figure 1.
- receive from two different children: case d in Figure 1.

These rules also apply for $Parent(u)$ (see case b for simultaneous transmission and receipt and case d for simultaneous receipts in Figure 1). In addition, when u transmits, it interferes with the transmission of any node v, whose parent is one-hop away from u (see case e in Figure 1). ■

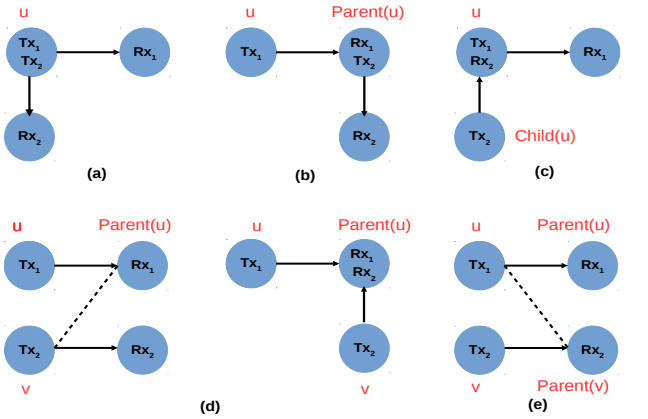


Fig. 1. Conflicting transmissions without acknowledgment.

Lemma 2: In data convergecast with immediate acknowledgment, $Conflict(u)$ contains:

a) the node u itself,
b) the node $Parent(u)$,
c') all the nodes that are 1-hop away from u,
d) all the nodes that are 1-hop away from $Parent(u)$,
e) all the nodes whose parent is 1-hop away from u.
f) all the nodes whose parent is 1-hop away from $Parent(u)$.

Proof: The set $Conflict(u)$ with immediate acknowledgment is equal to the set $Conflict(u)$ without acknowledgment $\bigcup$ the set of nodes that are 1-hop away from u (see case c' in Figure 2) $\bigcup$ the set of nodes whose parent is 1-hop away from $Parent(u)$ (see case f in Figure 2). ■

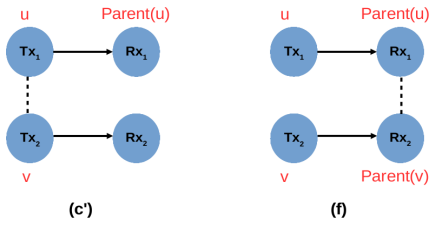


Fig. 2. Additional conflicting transmissions with immediate acknowledgment.

Corollary 1: With the immediate acknowledgment policy, $Conflict(u)$ contains the two additional cases depicted in Figure 2.

IV. Theoretical Bounds

In this section and only in this section, we adopt some additional assumptions that are used only for the computation of theoretical bounds:

Assumption 4: No message loss and no node failure.

Assumption 5: No topology link in addition to those belonging to the routing tree.

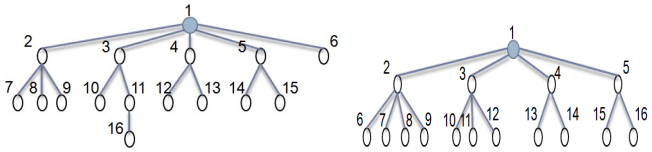
We have shown in Corollary 1 that for any node u, the set $Conflict(u)$ with immediate acknowledgment is a superset of $Conflict(u)$ without acknowledgment. Hence, the number of slots for a raw data convergecast with immediate acknowledgment is higher than or equal to this obtained without acknowledgment. Therefore, the lower bounds computed without acknowledgment still hold for immediate acknowledgment see Section IV-A. In Section IV-B, we then identify configurations for which the number of slots is the same with both acknowledgment policies.

We order the children of the sink by the decreasing order of the number of their transmissions. The first child transmits more messages to the sink than any other. It is called the most transmitting child and it is denoted $c1$. A subtree is said active in a slot t if and only if the node at which it is rooted is either transmitting or receiving in this slot.

With regard to the minimum number of slots required, we define two types of configurations:

- The T_n $configurations$ are traffic-balanced. By traffic-balanced, we mean that each subtree of the sink delivers approximately the same number of messages to the sink. See Figure 3.a. The sink has to receive $\sum_{u \neq sink} Gen(u)$ messages from its children. The number of simultaneous transmissions to the sink is limited by the number of children, the number of sink interfaces, as well as the number of available channels (each interface using a channel different from the channels used by the other active interfaces). Hence, the number of slots needed is higher than or equal to $S_n = \lceil \frac{\sum_{u \neq sink} Gen(u)}{min(n_{interf}, n_{child}, n_{channel})} \rceil$;
- The T_t $configurations$ are dominated by the subtree requiring the highest number of transmissions. See Figure 3.b. Each child, i, of the sink has $\sum_{v \neq i \in subtree(i)} Gen(v)$ packets to receive and $\sum_{v \in subtree(i)} Gen(v)$ packets to transmit. Moreover, let us order the sink's children according to the decreasing order of the number of slots they need, that is, $Gen(i) + 2\sum_{v \neq i, v \in subtree(i)} Gen(v)$ for the sink child i. If the $(min(n_{interf}, n_{child}, n_{channel}) + 1)^{th}$ sink child requires the same number of slots as $c1$ the first one, then its schedule will require an additional slot. Indeed, the schedule of this subtree requires the same number of slots as the first one. However, the $(min(n_{interf}, n_{child}, n_{channel}) + 1)^{th}$ sink child starts to transmit one slot later than the first one, because at the first slot, all the available interfaces of the sink or all the available channels are used by the $min(n_{interf}, n_{child}, n_{channel})$ first children of the sink. Consequently, the

$(min(n_{interf}, n_{child}, n_{channel})+1)^{th}$ sink child will end one slot after. Hence the number of slots needed is given by $S_t = Gen(c1) + 2\sum_{v \neq c1, v \in subtree(c1)} Gen(v) + \delta$, where $\delta = 1$ if the $(min(n_{interf}, n_{child}, n_{channel})+1)^{th}$ sink child requires the same number of transmissions as $c1$ and $\delta = 0$ otherwise.



(a) A T_n configuration. (b) A T_t configuration.

Fig. 3. An example of two configurations with the same number of nodes but whose number of slots differs.

A. Lower bounds valid whatever the acknowledgment policy

Lemma 3: The minimum number of slots required for raw data gathering in a multichannel WSN generating heterogeneous traffic, whatever the acknowledgment policy, is lower bounded by $max(S_n, S_t)$.

Proof: The number of slots needed is determined by the highest value between S_n and S_t. ■

B. Accurate bounds for the immediate acknowledgment

We now detail cases where the bounds given in the previous section, are still valid with the immediate acknowledgment policy.

Lemma 4: For raw data gathering in a multichannel WSN, the only additional conflicts created by the choice of the immediate acknowledgment policy are those occurring between a node and one of its nephews (i.e. a child of a brother of the node considered).

Proof: The additional conflicts introduced by the immediate acknowledgment policy are those illustrated by Figures 2.f and 2.c'. We distinguish two cases:

- Conflict illustrated by Figure 2.f: Since in a data gathering tree, any node different from the sink transmits only to its parent in the tree and since there is no radio link other than those present in the data gathering tree (Assumption A5), this case can occur only when $Tx1$ is a node u different from the sink, $Rx1$ is the parent of u, whereas $Rx2$ is a brother of u and $Tx2$ is a child of $Rx2$.
- Conflict illustrated by Figure 2.c': Such a conflict can never occur, since in the tree no two one-hop neighbors can transmit simultaneously to two different parents. Hence, the only additional conflicts are those occurring between a node different from the sink and one of its nephews (see the collision caused by the transmissions of nodes 2 and 7 in Figure 4). ■

Lemma 5: The minimum number of slots required for raw data gathering in a multichannel WSN having a star or line topology, whatever the acknowledgment policy, is equal to $max(S_n, S_t)$.

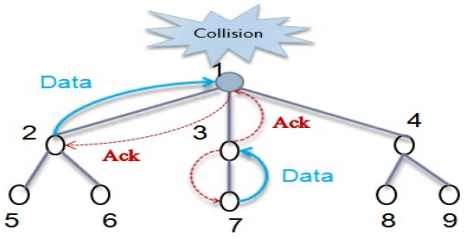


Fig. 4. Collision between a node and its nephew.

Proof: A star topology or a line topology does not create any conflict between a node and its nephew because no node has a nephew. Hence, the minimum number of slots does not depend on the acknowledgment policy. From Lemma 3, it is equal to $max(S_n, S_t)$. ■

Theorem 1: When $nchild \leq min(nchannel, ninterf)$ or $ninterf < min(nchild, nchannel)$, the minimum number of slots required for raw data gathering in a multichannel WSN generating heterogeneous traffic is the same with and without the immediate acknowledgment.

Proof: We distinguish two cases.

- First case: If $nchild \leq min(nchannel, ninterf)$: We can build a schedule where the subtree rooted at any sink child c is active in any slot t with $1 \leq t \leq Trans(c)$ and nodes are scheduled according to their depth in the tree. Each child of the sink is assigned both its own interface and its own channel. In each brotherhood of depth $4h+1$, with h integer ≥ 1, one node u (the selection policy of this node in its brotherhood is round robin) is assigned the same odd slot t and the same channel as the root of its subtree (if u has still packets to transmit). Similarly, in each brotherhood of depth $4h+3$, one node is assigned the same odd slot as the root of its subtree (if this slot is needed) but with a different channel (e.g. the channel of the root of the previous subtree). Alternatively in the even slots and as long as they have packets to transmit, in each brotherhood of even depth, one node is scheduled on the channel used by its parent in the previous slot. Since in the absence of acknowledgment, this schedule provides the minimum number of slots required by the most demanding subtree, given by S_t, this schedule is optimal. Since there is no slot where a node and its nephew are scheduled in the same slot and on the same channel, according to Lemma 4, this schedule is also valid if the immediate acknowledgment is used. Hence, the first part of the theorem.
- Second case: If $ninterf < min(nchild, nchannel)$: At any slot at most $min(nchannel, nchild, ninterf) = ninterf$ children of the sink are scheduled. Let us consider any valid schedule without acknowledgment that provides the minimum number of slots and build a valid schedule for the immediate acknowledgment. We distinguish two subcases:
  - First subcase: there is no slot where a node and its nephew transmit in the same slot and on the same channel. Hence, according to Lemma 4, this schedule is also valid if the immediate acknowledgment is used.

◦ Second subcase: there is a slot where a node and its nephew, denoted $neph$, transmit in the same slot and on the same channel. We focus on t the smallest time slot where this occurs. Since $nchannel > ninterf$, at most $ninterf$ uncles of $neph$ are scheduled in the same slot but each on its own channel. It is necessary to schedule the transmission of node $neph$ on a channel ch that is unused by the uncles of $neph$ scheduled in this slot. Such a channel exists since $nchannel > ninterf$. Furthermore, we have to avoid the creation of new conflicts: for instance a conflict between $neph$ and $child_{neph}$, a child of a child of $neph$, when $child_{neph}$ is scheduled in the same slot and on the same channel ch as $neph$. We then apply the following rules: any descendant of $neph$ at a depth = $4h + 4$ with h integer ≥ 0 is scheduled on a channel already used by a child of the sink, whereas any descendant of $neph$ at a depth = $4h + 6$ with h integer ≥ 0 is scheduled on the channel ch. Hence, the conflict caused by $neph$ or any of its descendant is avoided. This method can be applied to any conflicting node $neph$. Finally, we get a valid schedule using the same number of slots as the initial one. Hence, the immediate acknowledgment does not require any additional slot. Hence, the second part of the theorem. ■

Theorem 2: If $nchannel \leq ninterf < nchild$ and $nchannel \times max(S_t, S_n) \geq \sum_{u \in\ WSN} Gen(u) + Rcv(c)$, the minimum number of slots required for raw data gathering in a multichannel WSN generating heterogeneous traffic is equal to $max(S_t, S_n)$, with and without the immediate acknowledgment, where $Rcv(c)$ denotes the number of packets received by the most receiving child of the sink.

Proof: The quantity $nchannel \times max(S_t, S_n)$ denotes the number of transmission opportunities available for the children of the sink. The number of messages that must be transmitted to the sink is equal to $\sum_{u \in\ WSN} Gen(u)$. Each of this transmission uses a distinct interface and channel. If $nchannel \leq ninterf < nchild$, the only possibility for any child of a sink child is to transmit on a channel already used by one of its uncles. To avoid such a conflict, we should have: $nchannel \times max(S_t, S_n) \geq \sum_{u \in\ WSN} Gen(u) + Rcv(c)$, where c denotes the most receiving child of the sink. In such a case, the conflict is avoided and the number of slots is the same with and without immediate acknowledgment; it is equal to $max(S_n, S_t)$. Hence, the theorem. ■

We now prove the existence of T_t and T_n configurations that both require different numbers of slots, depending on the acknowledgment policy.

Let us consider the example depicted by Figure 3 where both configurations have the same number of nodes. Furthermore, we assume two channels, two radio interfaces for the sink and an homogeneous traffic. We have $nchannel \leq ninterf < nchild$ in both configurations. For the T_n configuration (see Figure 3.a), we have $S_n = 8 > S_t = 7$, 8 slots are needed without acknowledgment and 9 slots with immediate acknowledgment. For the T_t configuration (see Figure 3.b), we have $S_t = 9 > S_n = 8$. Hence, 9 slots are needed without acknowledgment, but 10 slots are required with immediate acknowledgment. We notice that neither T_t nor T_n meets $nchannel \times max(S_t, S_n) \geq \sum_{u \in\ WSN} Gen(u) + Rcv(c)$, explaining why the number of slots differs according to the acknowledgment policy.

V. A DISTRIBUTED CHANNEL AND SLOT ASSIGNMENT

We are interested in a joint channel and slot assignment algorithm that minimizes the number of slots needed by raw data convergecast, taking advantage of the multichannel WSN but without making any assumption regarding:

- the network links: additional links to the routing tree may exist in the topology creating additional conflicts.
- the multichannel topology: the topology of the network may differ from one channel to another. However, the connectivity is assumed on any channel.
- the acknowledgment policy: it is optimized for the absence of acknowledgment and for the presence of immediate acknowledgment.

DiSCA is a distributed algorithm where any node only knows the information related to itself and its conflicting nodes. The i^{th} iteration schedules the i^{th} transmission of each node having at least i transmissions to perform. Each iteration provides a micro-schedule and micro-schedules can overlap to reduce the total number of slots. It proceeds in successive iterations applying the following rules:

Rule 1: For any node $u \neq sink$, DiSCA defines the static priority of any node as follows: $Prio(u) = \sum_{v \in Subtree(u)} Gen(v) = Trans(u)$. If two nodes have the same priority, the node with highest depth in the routing tree is scheduled first. If both nodes have the same depth, the node with the smallest identifier is favored.

Rule 2: DiSCA in distributed mode proceeds in successive iterations too. As illustrated in Algorithm 1, on any node u, the i^{th} iteration schedules the i^{th} transmission of any node $v \in Conflict(u)$ with unscheduled transmissions. More precisely, the i^{th} transmission of any node $u \neq sink$ with $Trans(u) \geq i \geq 1$ is scheduled in iteration i if and only if:

- the i^{th} transmission of any node $v \in Conflict(u)$ such that $Prio(v) > Prio(u)$, with $1 \leq i \leq Trans(v)$ is already scheduled,
- and if $i > 1$, the $(i-1)^{th}$ transmission of any node $v \in Conflict(u)$, whatever its priority, with $1 \leq i-1 \leq Trans(v)$, is already scheduled.

Rule 3: The i^{th} transmission of u is scheduled in the first slot t and on the first channel ch (channels being visited in round robin order) where the following conditions are met:

- u has a packet to transmit in this slot,
- u has an available interface,
- the parent of u has an available interface,
- the transmission of u does not conflict with the transmission of any node already scheduled in the slot t and on the channel ch.

Rule 4: Each node u whose i^{th} transmission is scheduled in slot t and on channel ch notifies its conflicting nodes with the $SlotChAssigned$ message (see Algorithm 1).

Algorithm 1 DiSCA algorithm

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.
Output: $ScheduledNodes$: Channel and time slot assignment for local node u and $Conflict(u)$ per slot and channel
Initialization:
$Slot(u) \leftarrow 0$ /*number of slots already assigned to node u*/
$LastTxSlot(u) \leftarrow 0$ /*the last slot assigned to u*/
$EarliestPcktSlot(u) \leftarrow 0$ /*the smallest slot in which u has generated / received a packet not yet transmitted.*/

Channel and slot assignment for node u
$SortedNodesList(u) \leftarrow$ the set $\{u\} \cup Conflict(u)$ sorted by decreasing priorities.
if Reception of a $SlotChAssigned(slot, ch, v, Parent(v))$ Message **then**
  Forward the $SlotChAssigned$ Message to the conflicting nodes
  if $v \in Conflict(u)$ **then**
    Update $ScheduledNodes$
  end if
  if $v = Child(u)$ **then**
    Update $AvailInterf(u, slot)$
    Update $EarliestPcktSlot(u)$
  else if $(v = Parent(u))$ || $(Parent(v) = Parent(u)$ & $v \neq u)$ **then**
    Update $AvailInterf(Parent(u), slot)$
  end if
end if
if $(Slot(u) < Trans(u)$) & ($\forall v \in SortedNodesList(u)$ such that $Trans(v) > Trans(u)$, $Slot(v) > Slot(u)$) **then**
  $ch \leftarrow 1$, $tx \leftarrow false$
  $nChannelReached \leftarrow false$
  $t \leftarrow max(LastTxSlot(u), EarliestPcktSlot(u))$
  while $(AvailInter(u,t) = 0$ || $AvailInter(Parent(u),t) = 0$ || $tx = false)$ **do**
    /* Find a time slot with available interface for u & $Parent(u)$ */
    $t \leftarrow t + 1$
    repeat
      if $((Conflict(u) \bigcap ScheduledNodes(t, ch) = \emptyset)$) **then**
        /* Node u can be scheduled in slot t on channel ch */
        $tx \leftarrow true$;
        $AvailInter(u,t) \leftarrow AvailInter(u,t) - 1$
        $AvailInter(Parent(u),t) \leftarrow AvailInter(Parent(u),t) - 1$
        $ScheduledNodes(t, ch) \leftarrow ScheduledNodes(t, ch) \cup \{u\}$
      else
        if $(ch < nchannel)$ **then**
          $ch \leftarrow ch + 1$ // try the next channel
        else
          $nChannelReached \leftarrow true$
          $tx \leftarrow$ false
        end if
      end if
    until $(tx$ || $nChannelReached$)
  end while
  Update of $LastTxSlot(u)$
  Update of $EarliestPcktSlot(u)$
  $Slot(u) \leftarrow Slot(u) + 1$
  Transmit the $SlotChAssigned(slot, ch, u, Parent(u))$ Message to the 1-hop neighbors of u.
end if

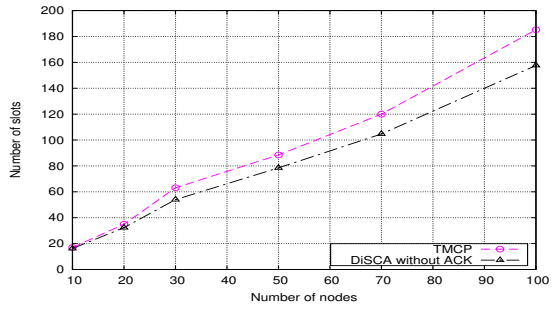
Notice that for any node u, its conflicting nodes $Conflict(u)$ are an input of the DiSCA algorithm. The determination of this set depends on the policy used for the acknowledgment (immediate acknowledgment or no acknowledgment). Hence, the same algorithm is applied whatever the policy used for the acknowledgment, but with different inputs. This algorithm is given (see Algorithm 1).

VI. Performance evaluation

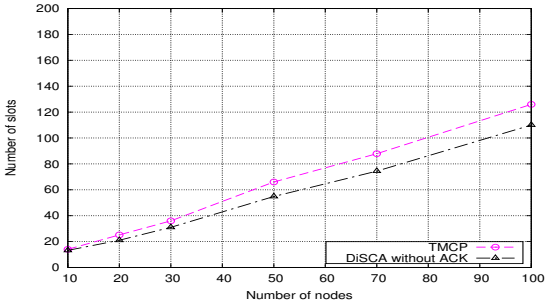
We evaluate the performance of DiSCA using our simulation tool based on GNU Octave [11]. We compare the number of slots required by DiSCA with the optimal (i.e. the theoretical bound) and that obtained by TMCP [3], MODESA [5] and Wave [10]. The number of nodes varies from 10 to 100. The first node is designated as the sink and all other nodes generate packets and send them to the sink. We use the Galton-Watson process as a branching stochastic process to generate random trees. The number of children per node is at most equal to 3. In addition, we assume that the only existing links are those in the tree. We suppose that all the nodes except the sink have a single radio interface and we vary the number of sink radio interfaces from 1 to 3. The number of available channels vary from 2 to 3. We also distinguish two subcases: homogeneous and heterogeneous traffic generated by nodes. In addition, in order to show how the convergecast structure can impact the schedule length, we differentiate the two types of configurations T_t and T_n. In the following, each result is the average of 20 simulation runs for small topologies (≤ 30 nodes) and 100 runs for large topologies.

A. Comparison with TMCP

We first compare DiSCA with TMCP [3], a famous centralized scheduling algorithm for data convergecast.



(a) in T_t configurations



(b) in T_n configurations

Fig. 5. DiSCA versus TMCP: homogeneous traffic

We assume a homogeneous traffic where each node, except the sink, generates one data packet. As illustrated by Figure 5,

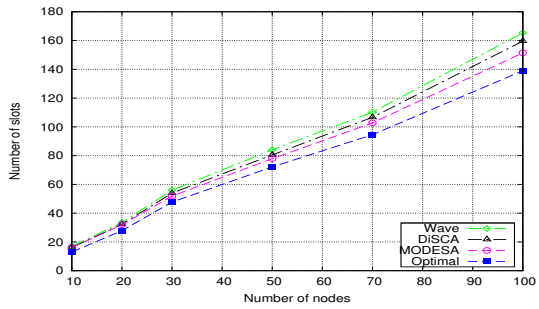
DiSCA outperforms clearly TMCP. Indeed, in data gathering trees with 100 nodes, DiSCA requires 12% (respectively 10.5%) slots less than the number required by TMCP in T_t configurations (respectively T_n configurations).

Indeed, TMCP partitions the network on multiple disjoint subtrees (vertex-disjoint). There is no channel reuse in the same subtree. In contrast, DiSCA allows non-conflicting nodes (even in the same subtree) to use same channels. Another drawback of TMCP is that the number of channels should be equal or higher to the number of subtrees. This is because, in TMCP, each subtree operates on a different channel. There is a problem when we do not have a sufficient number of channels to schedule all subtrees in parallel. The scheduling of a whole (or multiple) subtree should be delayed. In contrast, DiSCA uses the available channels to schedule nodes. Besides, unlike TMCP, DiSCA does not require that the sink is equipped with a number of radio interfaces equal to the number of sink children.

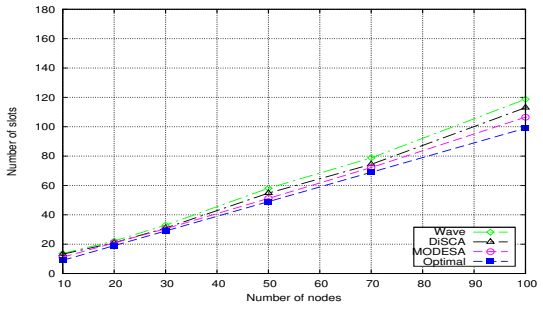
Notice however that in the comparative performance evaluation, the number of available channels and the number of sink interfaces are always higher than or equal to the number of sink children. In other words, we are always in a favorable context for TMCP.

B. Comparison with optimal schedule, Wave and MODESA

We compare here DiSCA with the optimal schedule and our previous published studies Wave and MODESA.



(a) in T_t configurations



(b) in T_n configurations

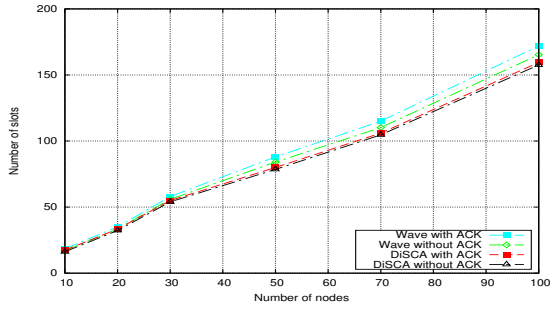
Fig. 6. DiSCA versus optimal schedule, Wave and MODESA: homogeneous traffic

Results depicted in Figure 6 show that DiSCA is at most 11% away from the optimal. This is due to the very accurate definition of the conflicting links, taking into account both the transmitter and the receiver. Moreover, DiSCA needs less slots than $Wave$ in both types of configuration to complete data convergecast. Hence, these simulation results show that DiSCA is more efficient than $Wave$.

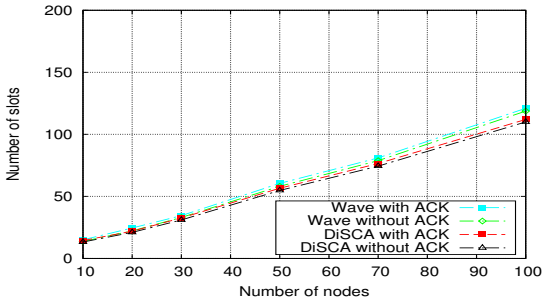
MODESA, a centralized time slot and channel assignment, provides better results than DiSCA as shown in Figure 6, because it uses a dynamic priority to schedule nodes. Nevertheless, the distributed mode of MODESA would require many more control messages than DiSCA because, after each slot, the priority of each transmitter and each receiver is updated and transmitted to the conflicting nodes. Whereas, the priority is sent only once in DiSCA.

C. Impact of immediate acknowledgment

We evaluate the impact of immediate acknowledgement. As expected the number of slots required when the immediate acknowledgement is used is higher than without, for both DiSCA and $Wave$ (see Figure 7). However, the gap remains small thanks to the accurate definition of conflicting transmissions (less than 3% for DiSCA).



(a) in T_t configurations



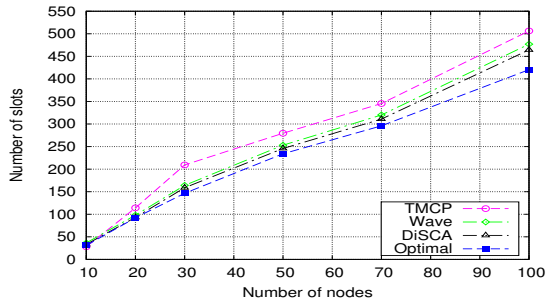
(b) in T_n configurations

Fig. 7. Impact of immediate acknowledgement

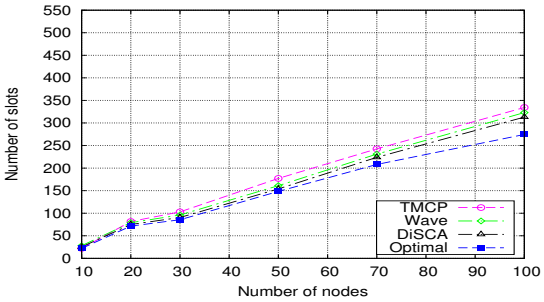
D. Impact of heterogeneous traffic

In this set of simulations, each node generates a number of packets randomly drawn in $[1, 5]$ per slotframe. We compare

DiSCA with TMCP, $Wave$ and the optimal schedule in terms of number of slots needed to complete convergecast.



(a) in T_t configurations



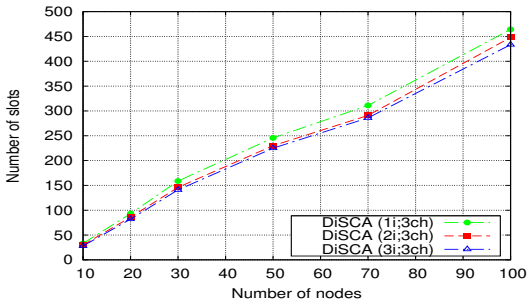
(b) in T_n configurations

Fig. 8. DiSCA versus optimal schedule, TMCP and Wave: heterogeneous traffic

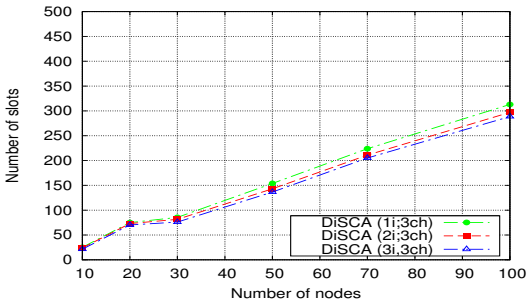
As illustrated in Figure 8, the number of slots per slotframe needed by all algorithms is higher than in the traffic homogeneous scenario. However, we observe the same trend as with homogeneous traffic. We also notice that T_t configurations are more greedy in number of slots than T_n. For example, for 100 nodes, T_n configurations need 67% of the number of slots needed by T_t configurations. This motivates the need to avoid an overloaded branch in the routing tree and to prefer routing trees balancing traffic over branches. Moreover, DiSCA needs 14% (respectively 10%) less slots than TMCP in T_s configurations (respectively T_n configurations).

E. Impact of the number of sink interfaces and channels

Figure 9 depicts the results obtained with DiSCA when the number of sink interfaces varies. When the sink is equipped with multiple interfaces, we also observe a reduction of the number of slots: with 3 interfaces, the number of slots is reduced by 8% for T_n configurations and 6% for T_t configurations. The minimum number of sink interfaces should be equal to the minimum of $nchannel$ and $nchild$ to obtain the best performances.



(a) in T_t configurations



(b) in T_n configurations

Fig. 9. Impact of sink radio interfaces on the number of slots

VII. Conclusion

In this paper, we have studied raw data convergecast in multichannel WSNs, where wireless sensors generate heterogeneous traffic collected by a sink that may be equipped with multiple radio interfaces. We have defined very accurately the conflicting transmissions, taking into account the acknowledgement policy. We have determined bounds on the minimum number of slots required when the immediate acknowledgment policy is used. We have proposed DiSCA, a distributed joint time slot and channel assignment algorithm. DiSCA does not suppose that all interfering links have been removed by channel allocation. Furthermore, DiSCA supports scheduling with a sink equipped by multiple radio interfaces. Unlike TMCP, DiSCA is self-adaptive to various raw data convergecasts (e.g. homogeneous/heterogeneous traffic generated by nodes, number of radio interfaces of the sink, presence of topology links in addition to those present in the routing tree, number of available channels). It obtains the best performances when the number of radio interfaces of the sink is equal to the minimum between the number of available channels and the number of sink's children.

Simulations results show that DiSCA is close to the lower bounds for raw data convergecast in various configurations and for different acknowledgement policies. This performance evaluation shows that DiSCA is able to schedule medium access in multi-hop $IEEE$ 802.15.4e TSCH based networks.

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] Institute of Electrical and Electronics Engineers (IEEE). 802.15.4e-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.

[3] Y. Wu, J. Stankovic, T. He, S. Lin, *Realistic and efficient multi-channel communications in wireless sensor networks*, INFOCOM 2008, Phoenix, AZ, USA, 2008.

[4] 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 2012, Sydney, Australia, 2012.

[5] 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.

[6] 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.

[7] 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.

[8] 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*, INFOCOM 2009, Rio de Janeiro, Brazil, April 2009.

[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 2013, Madrid, Spain, 2013.

[10] R. Soua, E. Livolant, P. Minet, *A distributed joint channel and time slot assignment for convergecast in wireless sensor networks*, NTMS 2014, Dubai, UAE, March 2014.

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