MODESA: An optimized multichannel slot assignment for raw data convergecast in wireless sensor networks

Ridha Soua, Pascale Minet, Erwan Livolant

MODESA: an Optimized Multichannel Slot Assignment for Raw Data 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 aerospace applications, wireless sensor networks (WSNs) collect data from sensor nodes towards a sink in a multi-hop convergecast structure. The throughput requirement of these applications is difficult to meet with a single wireless channel. That is why, in this paper, we focus on a multichannel time slot assignment that minimizes the data gathering cycle. We first formalize the problem as a linear program and compute the optimal time needed for a raw data convergecast in various multichannel topologies. These optimal times apply to sinks equipped with one or several radio interfaces. We then propose our algorithm called MODESA and prove its optimality in various multichannel topologies. We evaluate its performances in terms of number of slots, maximum buffer size and number of active/sleep switches per node. Furthermore, we present variants of MODESA achieving a load balancing between the channels used.**

I. Context

Data gathering or convergecast applications represent the main part of applications supported by wireless sensor networks (WSNs). These applications generally require small delays for data gathering and time consistency of gathered data; this time consistency is usually achieved by a small gathering period. Collected data from sensors are transferred to a special entity, called sink, generally more powerful than other nodes. When the volume of data transmitted by any sensor is reduced, aggregation techniques are used to increase network efficiency and throughput. When several samples are transmitted in a single MAC frame, the length of the frame is usually close to the maximum length allowed by the MAC protocol (e.g.: 127 bytes for the IEEE 802.15.4 MAC protocol). As a consequence, no aggregation is possible in the intermediate nodes (i.e raw data convergecast). Other techniques must be investigated to achieve network throughput and efficiency. In this paper, we focus on multichannel techniques to ensure a small gathering cycle time and a higher throughput.

In many real deployments of WSNs, the channel used by the WSN usually encounters perturbations such as jamming, external interferences or noise caused by external sources (e.g. a polluting source such as a radar) or other coexisting wireless networks (e.g. WiFi, Bluetooth). Commercial sensor nodes can communicate on multiple frequencies as specified in the 802.15.4 standard. This reality has given birth to multichannel communication paradigm in WSN. Multichannel WSNs significantly expand the capability of single-channel WSNs by allowing parallel transmissions and avoiding channels that are congested or whose performances are degraded by interfering devices.

This work is partly funded by the FUI project SAHARA.

It is obvious that medium access protocols that are contention-based protocols are inefficient for periodic data collection under heavy traffic conditions which drastically increase the probability of collisions and retransmissions. In contrast, Time Division Multiple Access, TDMA, is a contention-free protocol where time is divided into cycles. A cycle is divided into slots. Interfering nodes are scheduled to transmit in different slots. Each node transmits data in its allocated slots. On the one hand, since the TDMA protocol removes idle listening and overhearing, which are the main sources of energy drain, TDMA deterministic scheduling is appropriate for low power devices since nodes turn off their radio in non scheduled time slots ensuring energy efficiency and prolonging network lifetime. On the other hand, minimizing the number of slots in the TDMA cycle is crucial if we want to preserve the small energy budget of sensors. In multichannel context, interfering nodes are scheduled to transmit on different channels for further throughput enhancement. Moreover, nodes near the sink suffer from heavy traffic. Therefore, we tackle in this paper the problem of ensuring to any node a medium access that is proportional to its traffic demand.

In this paper, we focus on a multichannel time slot assignment that minimizes the data gathering cycle. After a state of the art in Section II, we first formalize the problem as a linear program in Section III and determine the minimum number of slots for various multichannel topologies in Section IV. We then propose our algorithm called MODESA, prove its optimality for different topologies and present its variants achieving a load balancing between the channels used in Section V. Performances of MODESA are evaluated by simulation in Section VI. Finally, we conclude in Section VII.

II. State of the art

In multichannel WSNs, we distinguish two problems, namely channel assignment and node/link scheduling. These problems can be solved separately or jointly. In channel assignment, each sensor is assigned a physical channel. The channel can be assigned either to the sender (the receiver has to

978-1-4673-4883-6/12/$31.00 ©2012 IEEE

switch), or to the receiver (the sender has to switch, multicast is more complex), or both of them in case of frequency hopping. Existing multichannel assignment protocols can be classified [1] according to (1) the frequency of the channel assignment: (e.g. static, semi-dynamic or dynamic channel assignment), (2) the channel selection policy: (e.g. round robin, the least loaded), and (3) the channel coordination technique used (e.g. dedicated control channel, split phase, coloring like).

We now detail TDMA based scheduling protocols proposed in multichannel WSNs. WirelessHART [2] was the first communication standard specially designed to fit critical requirements of industrial applications. WirelessHART uses TDMA to arbitrate communications between devices. To enhance reliability, TDMA is combined with channel hopping on a per-transaction (packet + acknowledgment) basis. A fundamental shortcoming of this standard is that only a single device is scheduled for transmission in each channel at the same slot. Hence, there is no spatial reuse of the bandwith.

In [3], authors address jointly the link scheduling and channel assignment for convergecast in networks operating according to the WirelessHART standard. Authors have proven that for linear networks with N single buffer devices, the minimum schedule length obtained is $(2N-1)$ time slots with $\lceil N/2 \rceil$ channels. They present also an algorithm with time complexity $O(N^2)$ to generate the time and channel optimal convergecast schedule. The solution does not provide spatial reuse of the bandwith and is restricted to linear topologies which are not suitable for all real deployments. In addition, they focus only on single radio interface devices.

TMCP [4] was designed to support data collection traffic. It begins by partitioning the network into multiple subtrees and then assigns different channels to nodes belonging to different subtrees. Hence, it minimizes the interferences between subtrees. After the channel assignments, time slots are assigned to nodes. However, TMCP does not eliminate contention inside the branches of a subtree since nodes that belong to the same branch communicate on the same channel.

Y-MAC [5] is a multichannel MAC protocol for WSNs that requires that nodes share the same wake/sleep duty cycle. Time slots are not assigned to the senders but to the receivers. At the beginning of each time slot, potential senders for the same receiver contend for the medium. When a node needs to transmit multiple packets to a receiver, these packets are sent on different channels following a pre-determined hopping sequence. We notice that Y-MAC has not been designed for data gathering applications, where the contention around the sink quickly becomes severe especially in heavy traffic conditions.

In [6], Ramen et al have proposed PIP : a joint TDMA-FDMA based bulk transfer protocol. When the sink needs data from a specific sensor, it establishes a connection with this latter and downloads data from that node at the highest rate possible. The major shortcoming of PIP is that the sink can only collect data from at most two sensors at the same time (i.e at most two simultaneous connections).

Incel et all [7], have proven that if all interfering links are removed (with the required number of channels), the schedule length for raw-data convergecast is lower bounded by $max(2n_k - 1, N)$ where n_k is the maximum number of nodes in any top-subtree of the routing tree and N is the number of source nodes. They have also proposed an optimal convergecast scheduling algorithm JFTSS that achieves this lower bound on any network topology where the routing tree has an equal number of nodes on each branch. In this paper, we generalize these theoretical results considering that the sink has one or more radio interfaces and at least two channels are available at each node. We also propose an algorithm called MODESA that reaches these bounds in many multichannel topologies.

III. Multichannel slot assignment problem

We are looking for a multichannel slot assignment model that minimizes the number of slots assigned, under the assumptions described below, and ensures that no two conflicting nodes transmit simultaneously on the same channel.

A. Assumptions

◦ **A1. Node type:** We assume two distinct types of node in the network. The sinks are in charge of gathering data from the other nodes. These other nodes, called sources, generate packets they have to transmit towards the sinks.

◦ **A2. Node radio interface:** The sinks are the only nodes that have $k \geq 1$ radio interfaces. All the source nodes have a single radio interface. Hence, two children of the same parent that is not a sink cannot send data simultaneously, even on different channels, since every source node has a single radio interface.

◦ **A3. Available channels:** For the sake of simplicity, we assume that at each node, $n_{channel} > 1$ channels are available. These channels are numbered from 1 to $n_{channel}$. Network connectivity is assumed on any of these channels and the 1-hop neighborhood of any node is the same on any available channel. Since any node u different from the sink has a single radio interface, at most one channel is active at any time on node u. For the sink, there are at most k active channels simultaneously.

◦ **A4. Data gathering cycle:** In each data gathering cycle, each node except the sink transmits its own data to its parent in the data gathering tree rooted at the sink and forwards the data received from its children. For the sake of simplicity, we assume that the slot size enables the transmission of a single packet corresponding to the data generated by a node. Moreover, each unicast transmission is acknowledged in the time slot of the sender, this is called immediate acknowledgment.

◦ **A5. Conflicting nodes:** Two nodes are said *conflicting on a given channel* if and only if they cannot transmit in the same time slot on this channel.

◦ **A6. Ideal environment:** In this paper, we assume there is neither message loss, nor node failure.

B. Formalization of the problem

The network is formalized as a graph $G = (V, E)$ where V is the set of vertices representing network nodes and E is the set of edges representing the communication links. Let $V = V_s \bigcup V_g$, where V_s is the set of source nodes in the network and V_g represents the set of gateways acting as sinks, with $V_s \bigcap V_g = \emptyset$.

For each node $v \in V$, we define $I(v)$ the set of nodes that interfere with v when transmitting on the same channel. Moreover, let i_v denote the number of physical interfaces available at the node v. For any source s, let p_s be the number of packets that it has to transmit in the TDMA cycle. For any link e, let $f_{e,s}$ denote the number of packets generated by the source s and sent over the link e during the TDMA cycle. Let $E^+(v)$ denote the set of links through which a node v can transmit. Let $E^-(v)$ be the set of links through which a node v can receive.

Let C be the set of the $n_{channel}$ channels available for any transmission. We define $a_{e,c,t}$ the activity of a link e on the channel c in the time slot t, ie $a_{e,c,t} = 1$ if and only if there is a transmission of a packet on the link e on the channel c in the time slot t and $a_{e,c,t} = 0$ otherwise. Furthermore, let u_t be the use of a slot t, in other words $u_t = 1$ means that there is at least one link activity on any channel in the slot t and $u_t = 0$ denotes an empty slot.

We can compute T_{max}, an upper bound of the cycle length. This bound is reached when all packets are sent sequentially on the same channel. We then have: $T_{max} = \sum_s \sum_e f_{e,s} * depth_s$ where $depth_s$ is the depth of node s in the data gathering tree. The objective is to minimize the number of slots $t \leq T_{max}$.

$$min \sum_{t \leq T_{max}} u_t$$

with the following constraints:

$$a_{e,c,t} \leq u_t \quad \forall e \in E, \forall c \in C, t \leq T_{max} \tag{1}$$

$$a_{e,c,t} + a_{e',c,t} \leq 1 \quad \forall v \in V, \forall e \in E^+(v), \forall w \in I(v), \forall e' \in E^+(w), \forall c \in C, t \leq T_{max} \tag{2}$$

$$\sum_{c \in C} \sum_{e \in E^+(v)} a_{e,c,t} + \sum_{c \in C} \sum_{e' \in E^-(v)} a_{e',c,t} \leq i_v \quad \forall v \in V, t \leq T_{max} \tag{3}$$

$$\sum_{s \in V_s} f_{e,s} = \sum_{c \in C} \sum_{t \leq T_{max}} a_{e,c,t} \quad \forall e \in E \tag{4}$$

$$\sum_{e \in E^+(s)} f_{e,s} = p_s \quad \forall s \in V_s \tag{5}$$

$$\sum_{g \in V_g} \sum_{e \in E^-(g)} f_{e,s} = p_s \quad \forall s \in V_s \tag{6}$$

$$\sum_{s \in V_s} \sum_{e \in E^+(i)} f_{e,s} = p_i + \sum_{s \in V_s} \sum_{e \in E^-(i)} f_{e,s} \quad \forall i \in V_s \tag{7}$$

$$\sum_{c \in C} \sum_{e \in E^+(i)} a_{e,c,t} \leq \sum_{c \in C} \sum_{e \in E^-(i)} \sum_{t' \in \{1..t-1\}} a_{e,c,t'} + p_i - \sum_{c \in C} \sum_{e \in E^+(i)} \sum_{t' \in \{1..t-1\}} a_{e,c,t'} \quad \forall i \in V_s, t \leq T_{max} \tag{8}$$

Constraint 1 binds the use of a time slot to at least the activity of one link on any channel in the slot. Constraint 2 ensures that two conflicting nodes do not transmit on the same channel in the same time slot. Constraint 3 guarantees that the number of simultaneous communications for a node is limited to its number of interfaces. Constraint 4 ensures the mapping between the activities on all channels and the packets sent on links. Constraints 5, 6 and 7 express the conservation of messages between the sources and the sinks. The last constraint guarantees that any node receives or generates a packet before transmitting it.

C. Illustrative Examples

An optimal multichannel time slot assignment can be obtained by linear programming tools such as GLPK (GNU Linear Programming Kit) [8] based on this model. Figure 1 shows the optimal number of slots nb_s for different single-sink topologies (linear 1(a), multiline 1(b), balanced tree 1(c) and tree 1(d)) with various number of sink interfaces k and channels $n_{channel}$. These optimal results are reached by the MODESA algorithm presented in Section V.

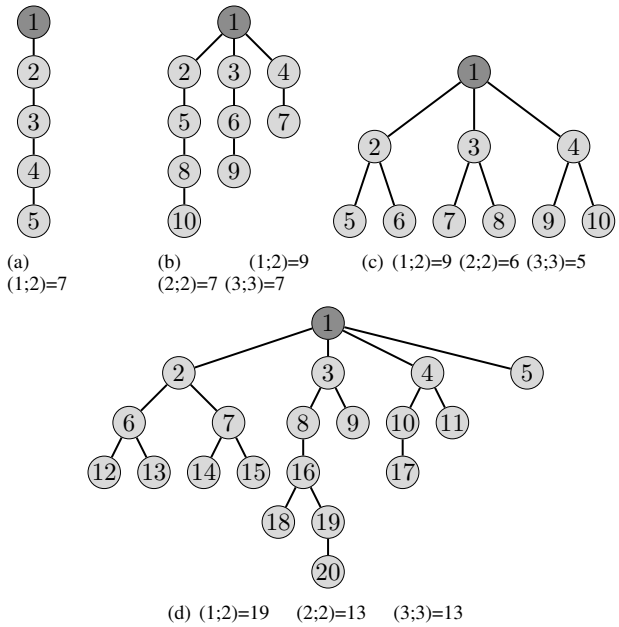


Fig. 1. The optimal number of slots nb_s for various topologies with different number of sink interfaces k and channels $n_{channel}$ with the notation: (k;$n_{channel}$)=nb_s.

We will see in the next section that we distinguish two types of network topologies where the optimal number:

- is imposed by the most populated subtree: see for instance Figures 1(b) and 1(d), both with two sink interfaces. In Figure 1(d), there are two most populated subtrees rooted at nodes 2 and 3 respectively.
- depends only on the number of nodes and the number of interfaces of the sink: see for instance Figures 1(c) and 1(d) both with a single sink interface.

We can observe that a given topology may belong to one type or another depending on the number of sink interfaces (e.g. Figures 1(c) and 1(d) for a single sink interface belong to the first type and for two sink interfaces to the second type).

IV. Theoretical bounds on the number of slots

A. Additional assumptions

∘ **A6. Neighborhood:** Two nodes u and v are 1-hop neighbors if and only if their distance is lower than or equal to the transmission range R. For any integer $h > 1$, any two nodes u and v are h-hop neighbors if and only if u is $(h-1)$-hop away from a 1-hop node of v.

∘ **A7. Interferences:** We assume that interferences are limited to 2 hops. Consequently, we assume that any two nodes u and v within 2-hop neighborhood from each other do not transmit in the same time slot on the same channel.

∘ **A8. Topology links:** We also assume that the only topology links are those represented in the convergecast tree.

Theorem 1: In any WSN with N nodes, a lower bound on the number of slots required by a raw data convergecast is $\lceil \frac{N-1}{min(k,n_c,n_{channel})} \rceil$, where $k \geq 1$ is the number of interfaces of the sink, n_c is the number of children of the sink and $n_{channel} > 1$ the number of available channels at each node.

Proof: In any network with N nodes, the sink has to receive $N-1$ messages from its children. The number of simultaneous transmissions to the sink is limited by the number of children and the number of sink interfaces as well as the number of available channels (each interface using its own channel). Hence, the number of slots needed is higher than or equal to $\lceil \frac{N-1}{min(k,n_c,n_{channel})} \rceil$. Hence the theorem. ■

B. Linear networks

Theorem 2: In linear networks where each node has $n_{channel} > 1$, the minimum number of slots for a raw data convergecast is $2N-3$, where N is the number of nodes including the sink, whatever the number of interfaces of the sink.

Proof: Consider a linear network with N nodes, where nodes are numbered from 1 to N, starting by the sink node. Any node $i > 1$ is at a distance $i - 1 < N$ from the sink. Node 2 needs to transmit $N-1$ packets to the sink (node 1) and needs to receive $N-2$ packets from node 3. Since node 2 has a single interface, these transmissions cannot overlap. As a consequence, a lower bound on the number of slots is equal to $N - 1 + N - 2 = 2N - 3$.

This bound is reached by the algorithm that schedules:

- in any odd slot, any node that is $(2h+1)$-hop away from the sink, with $h \in [0, \lfloor \frac{N-1}{2} \rfloor]$;
- in any even slot, any node that is $2h$-hop away from the sink, with $h \in [1, \lfloor \frac{N-1}{2} \rfloor]$.

Hence the theorem. ■

C. Tree networks

Theorem 3: In tree networks, a lower bound on the number of slots for a raw data convergecast is Max($\lceil \frac{N-1}{min(k,n_c,n_{channel})} \rceil$, $2n_1 - 1 + \delta$), where N is the number of nodes including the sink, n_1 is the maximum number of nodes in a subtree rooted at a sink child and n_c the number of sink children and $\delta = 1$ if the number of nodes of the $(1 + min(k, n_c, n_{channel}))^{th}$ most populated subtree rooted at a sink child is equal to the number of the most populated one, 0 otherwise.

Proof: Let $g = min(k, n_c, n_{channel})$. The sink requires $\lceil \frac{N-1}{g} \rceil$ time slots to receive all the packets generated in the network as seen in Theorem 1. Furthermore, each child of the sink has at least one packet to transmit. Moreover, let us consider the child of the sink with the highest number of descendants. Let n_1 be the number of nodes in the subtree rooted at this child. This latter requires $n_1 - 1$ slots to receive the packets from its children and n_1 slots to transmit its packets to the sink. Since all these transmissions are sequential, at least $2n_1 - 1$ slots are needed. If the $(g+1)^{th}$ most populated subtree has a number of nodes equal to n_1, 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 $(g+1)^{th}$ sink child starts to transmit at the second slot, that is a slot later. Indeed, at the first slot, all the available interfaces of the sink are used by the g children of the sink having the most populated subtrees. Consequently, the schedule of this subtree will end a slot after. Hence, the value of δ and the theorem. ■

Notice that a multi-line topology can be seen as a specific case of a tree topology.

D. Optimal schedule

In this section, we build an algorithm that reaches the bounds given in the previous theorems. As a consequence, these bounds are optimal as well as the algorithm. The basic idea of this optimal algorithm is to maintain the $g = min(k, n_c, n_{channel})$ interfaces of the sink busy as long as possible. We first notice that for a linear topology, the algorithm given for the proof of theorem 2 meets this requirement. We now extend this algorithm to tree topologies. This algorithm, called FlipFlop, proceeds as follows:

First, it orders all the subtrees rooted at a sink child according to the decreasing number of nodes. The g first subtrees form the first group, the next g subtrees form the secong group and so on until the last one. We first consider the case where $g \leq n_c \leq 2g$, there are at most 2 groups.

This algorithm schedules in the odd slots:

- the nodes of odd depth in the subtrees 1 to g, including the sink children belonging to the first group,

- the nodes of even depth in any other subtree.

It schedules in the even slots:

- the nodes of even depth in the subtrees 1 to g,
- the nodes of depth 1 in subtrees $g+1$ to $2g$,
- the nodes of odd depth > 1 in any subtree $> g$.

Furthermore, this schedule meets the following rules:

- If several nodes have the same parent that is not the sink, they are scheduled round robin.
- In the same subtree, two 2-hop nodes that are scheduled in the same time slot transmit in different channels. For instance, channel 1 is used at depth 1, 5, and 9, whereas channel 2 is used at depth 3, 7 and 11...
- As soon as the schedule of a subtree is completed, the first sink child that has never been scheduled is scheduled.

A slot where the g interfaces of the sink are not busy is said uncomplete.

Theorem 4: In a multichannel WSN the optimal number of slots for a raw data convergecast is:

- $2n_1 - 1$ if $n_c = g$;
- $2n_1 - 1 + \delta$ if $g + 1 \leq n_c \leq 2g$;

with $g = min(k, n_c, n_{channel})$, n_1 is the maximum number of nodes in a subtree rooted at a sink child, n_c the number of sink children and $\delta = 1$ if the number of nodes of the $(g+1)^{th}$ most populated subtree rooted at a sink child is equal to n_1, 0 otherwise. In both cases, the FlipFlop algorithm is optimal.

Proof: When there is only one group, $n_c = g$, it is clear that the FlipFlop algorithm requires exactly $2n_1 - 1$ slots, that is the lower bound. Hence, the FlipFlop algorithm is optimal. If there are two groups, $g + 1 \leq n_c \leq 2g$, the FlipFlop algorithm schedules in the odd slots the sink children of the first group and in the even slots the sink children of the second group. We distinguish two cases:

- if the size of the $(g + 1)^{th}$ subtree is identical to the size of the first subtree, the FlipFlop algorithm requires an additional slot to complete the schedule of the second group. $2n_1$ slots are used, that is the optimal number.
- otherwise, the FlipFlop algorithm does not need any additional slots. Hence, it uses $2n_1 - 1$ slots for its schedule.

In both cases, the FlipFlop algorithm is optimal. ■

Theorem 5: In a multichannel WSN, the FlipFlop algorithm is not optimal for 3 groups.

Proof: We just point out an example where the FlipFlop algorithm is not optimal. We consider a multiline topology with 16 nodes, 2 sink interfaces, 2 channels and 5 sink children. Hence, $g = min(2, 2, 5) = 2$. The topology is depicted in Figure 2. We notice that the third group contains only one subtree rooted at a sink child. The FlipFlop algorithm needs 9 slots, whereas the optimal slot number is $8 = \lceil \frac{N-1}{g} \rceil$. This can be explained by the fact that there are $3 > g = 2$ slots where only the last sink child transmits. ■

Theorem 6: In a multichannel WSN, the optimal number of slots for a raw data convergecast is: $max(\lceil \frac{N-1}{g} \rceil, 2n_1 - 1 + \delta)$, if $n_c > 2g$, where N is the number of nodes including the sink, $g = min(k, n_c, n_{channel})$, n_1 is the maximum number of nodes in a subtree rooted at a sink child, n_c the number of sink children and $\delta = 1$ if the number of nodes of the $(g+1)^{th}$ most populated subtree rooted at a sink child is equal to n_1, 0 otherwise.

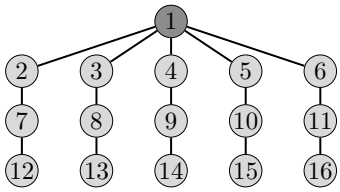


Fig. 2. An example of topology with 16 nodes and 2 sink interfaces where FlipFlop is not optimal.

Proof: We now consider the case $n_c > 2g$ and prove by induction that there exists an algorithm that uses $n_{slot} = max(2n_1 - 1 + \delta, \lceil \frac{N-1}{g} \rceil)$. This is true for $n_c = N - 1$, in this case the sink has n_c children which are leaves. Each sink child has exactly one message to transmit. In each slot, the algorithm schedules a group. Consequently, the number of slots needed is $n_{slot} = \lceil \frac{N-1}{g} \rceil$. Since $n_1 = 1$, we have $max(2n_1 - 1 + \delta, \lceil \frac{N-1}{g} \rceil) = \lceil \frac{N-1}{g} \rceil$.
Let us consider any topology with N nodes and $n_c > 2g$. From this topology, we can build a topology with $N - 1$ nodes by removing a node in the last group, while maintaining the non increasing order of the number of descendants in the subtrees. According to our induction assumption, there exists an optimal algorithm that schedules any topology with $N - 1$ nodes and $n_c \geq 2g$ in $n_{slot} = max(2n_1 - 1 + \delta, \lceil \frac{N-2}{g} \rceil)$. With regard to the scheduling of the topology with $N - 1$ nodes, the node inserted and all its ascendants require the transmission of an additional message. These transmissions, except the transmission by the root of this subtree, do not require any sink interface, we schedule them at the earliest (i.e. in the first slot where it is possible), in parallel with other nodes, without requiring any additional slot. It follows that the only transmission that remains to be scheduled is the transmission done by the root of the subtree involved. Let $child$ denote this node. We consider two cases:

- $2n_1 - 1 + \delta \geq \lceil \frac{N-1}{g} \rceil$: the schedule length is imposed by the two first groups.
  Since the associated topology with $N - 1$ nodes is obtained by removing a node in the last group and the number of groups is strictly higher than two, we also have $2n_1 - 1 + \delta \geq \lceil \frac{N-2}{g} \rceil$. According to our induction assumption, there exists an algorithm that reaches this bound of $2n_1 - 1 + \delta$. We modify this schedule to insert the additional transmissions required by the insertion of a node, as explained previously. The transmission of $child$ can be scheduled in the last uncomplete slot where $child$ is not transmitting. Such a slot exists, since $child$ does not belong to the two first groups that impose the schedule length and the message originated from the inserted node has already reached $child$.

- $\lceil \frac{N-1}{g} \rceil > 2n_1 - 1 + \delta$.
  Since the associated topology with $N-1$ nodes is obtained by removing a node in the last group and the number of groups is strictly higher than two, we also have $\lceil \frac{N-2}{g} \rceil \geq 2n_1 - 1 + \delta$. According to our induction assumption, there exists an algorithm that reaches this bound of $\lceil \frac{N-2}{g} \rceil$. We modify this schedule to insert the additional transmissions required by the insertion of a node, as explained previously. For the transmission of $child$, we consider two cases:
  - $\lceil \frac{N-2}{g} \rceil = \lceil \frac{N-1}{g} \rceil$, there are uncomplete slots in the schedule of the $N-1$ topology. We distinguish again two subcases:
    * If in the last uncomplete slot, $child$ is not transmitting then the transmission of $child$ can be scheduled in this slot. Notice that the message originated from the inserted node has already reached $child$. Hence, the number of slots required for the N node and the $N-1$ node topologies are identical.
    * Otherwise, we go backward to find a slot such that a sink child $other \neq child$ that does not transmit in the last slot is transmitting and no child of $child$ is transmitting. We then exchange the transmissions of $child$ and $other$. Consequently, we have found a schedule for the N node topology with the same number of slots than the $N-1$ topology, such that all messages sent by the new inserted node are transmitted to the sink.
  - $\lceil \frac{N-2}{g} \rceil = \lceil \frac{N-1}{g} \rceil - 1$, there is no slot in the schedule of the $N-1$ topology where the transmission of this sink child can be done. Hence, an additional slot is inserted at the end of the schedule.

Consequently, we have established a schedule for the N topology that reaches the bounds. Hence, the theorem. ■

V. MODESA: Multichannel Optimized DElay time Slot Assignment

The aim of this section is to propose a centralized raw data convergecast scheduling, called MODESA, that takes into account the availability of multiple channels to reduce the TDMA cycle length while ensuring a fair medium access.

A. Principles

MODESA builds the multichannel scheduling slot by slot applying the following rules:

1) Any node has a *dynamic priority*. The priority is equal to $remPckt * parentRcv$ where $remPckt$ means the number of packets the node has in its buffer at the current iteration. $parentRcv$ is the total number of packets the parent of the node has to receive in a cycle. The idea behind this heuristic is to reduce the number of buffered packets by favoring nodes having packets to transmit to a parent with a high number of descendants.
2) Nodes compete for the current time slot if and only if they have data to transmit.
3) In addition to be allowed to transmit in a slot, a node and its parent must have an available interface.
4) For any slot, the first scheduled node is the node having the highest priority among all the nodes having data to transmit. If two nodes have the same priority, MODESA chooses the node with the smallest identifier. The selected node is scheduled on the first channel c in the greedy variant.
5) Any node can be scheduled in the current time slot on channel c if it does not interfere with nodes already scheduled on channel c in this slot.
6) Conflicting nodes that interfere with nodes already scheduled in this slot are scheduled on a different channel.

B. The MODESA algorithm

MODESA pseudo-code is given by Algorithm 1. The algorithm iterates over N the set of nodes having data to transmit and sorted according to their priorities. In each iteration, the algorithm determines among these nodes the set of nodes that are assigned to the current time slot t. The node u with the highest priority that has an available interface as well as its parent is scheduled first. Then MODESA iterates on the set of nodes sorted according to their priority. Nodes which are non conflicting are allocated to the same channel. In contrast, any other node in the sorted set N is assigned the same time slot but on a different channel if it conflicts with nodes already assigned to the current slot.

C. Optimality of MODESA

A tree is said balanced if and only if at each level l, the number of children is the same for all nodes belonging to level l.

Theorem 7: When $n_{channel} > 1$, MODESA is optimal for linear, multiline and balanced tree topologies.

Proof: We consider three cases:

- First case: linear topologies
  The behavior of MODESA and FlipFlop are identical: odd (even respectively) slots schedule the transmissions of nodes at an odd (even respectively) distance from the sink. Hence, MODESA is optimal for linear topologies.

- Second case: multiline topologies
  - When there is a single group, all sink children are scheduled in parallel. The number of slots is the one needed to schedule the first sink child. Hence, MODESA is optimal.
  - When there are two groups, MODESA schedules in the first slot the sink children of the first group because of their higher number of packets to receive and in the second slot the sink children of the second group. With two groups, a child of a sink child has never a priority higher than its parent, when this parent has at least one message to transmit. Hence, MODESA behaves exactly like FlipFlop. Since FlipFlop is optimal, MODESA is optimal too.

Algorithm 1 MODESA algorithm with the greedy variant

Input: $n_{channel}$ channels, a spanning tree T, where each node u has i_u radio interfaces, d_u packets to transmit and a set of conflicting nodes $Conflict(u)$.
Output: The scheduling of nodes in the TDMA cycle
/* Initialization phase */
Initialize priority and traffic demand d_u for each node u
$t \leftarrow 0$ // current time slot
/* Scheduling phase */
while $\sum_u d_u$ **do** // there are packets to transmit
  Initialize number of available interfaces of nodes
  Initialize conflicting nodes on channel c, $conflict_c \leftarrow \emptyset, \forall c = 1..n_{channel}$
  $N \leftarrow$ list of nodes having data to transmit and sorted according to their priorities.
  $t \leftarrow t + 1$
  /* Assignment of slot t */
  while $N \neq \emptyset$ **do**
    $Tx \leftarrow False, nChannelReached \leftarrow False$
    repeat
      Select node v with the highest priority in N
      $N \leftarrow N \setminus \{v\}$
    until $i_v > 0$ and $i_{parent(v)} > 0$
    $c \leftarrow 1$ // selected channel
    repeat
      if $v \notin conflict_c$ **then**
        Node v transmits in slot t on the channel c
        $d_v \leftarrow d_v - 1$
        $d_{parent(v)} \leftarrow d_{parent(v)} + 1$
        $i_v \leftarrow i_v - 1$
        $i_{parent(v)} \leftarrow i_{parent(v)} - 1$
        $conflict_c \leftarrow conflict_c \cup Conflict(v)$
        $Tx \leftarrow True$
      else
        if $c < n_{channel}$ **then**
          $c \leftarrow c + 1$ // change of selected channel
        else
          $nChannelReached \leftarrow true$
        end if
      end if
    until Tx || $nChannelReached$
  end while
  Update priority of nodes
end while

  - When there are more than two groups, we distinguish two cases:
    * If $2n_1 - 1 + \delta \geq \lceil \frac{N-1}{g} \rceil$, the most populated line determines the schedule length. Let c_1 be the sink child corresponding to this line. In any slot, MODESA schedules either c_1 or the child of c_1, keeping c_1 always active, either transmitting to the sink or receiving from its child. This gives the optimal schedule for this line. Furthermore, MODESA completes the schedule of any slot with the other sink children that have not yet received the slots they need, while keeping busy the g sink interfaces as long as possible. Hence, MODESA gets the optimal number of slots.
    * Otherwise $2n_1 - 1 + \delta < \lceil \frac{N-1}{g} \rceil$, the schedule length is determined by the number of nodes and g. As long as g sink children have not received the slots they need, MODESA maintains busy the g sink interfaces. Furthermore, MODESA uses a variable Round Robin that ensures that the schedule of the $n_c \ modulo \ g$ last sink children is never entirely postponed at the end. This is the difference with FlipFlop, where more than g uncomplete slots (i.e. slots with some sink interfaces inactive) may exist.

    In both cases, MODESA maintains busy the g interfaces of the sink as long as possible. It is not possible to use a lower number of slots to schedule the sink children. Hence MODESA is optimal for multilines.

- Third case: balanced trees
  - When there is a single group, all subtrees are scheduled in parallel. The number of slots is the one needed to schedule the first subtree. Hence, MODESA is optimal.
  - When there are two groups, MODESA schedules in the first slot the sink children of the first group because of their higher number of packets to receive and in the second slot the sink children of the second group. Notice that depending on the number of sink children in the second group, some of g of the interfaces of the sink may be unused. In the third slot, it is again the sink children of the first group. The sink children of the second group occupy the fourth slot. Hence, MODESA schedules in round robin the sink children of the first group and the sink children of the second group. Other nodes are scheduled in the slots where they do not conflict. MODESA uses the same number of slots as the FlipFlop algorithm. It is optimal.
  - When there are more than two groups, MODESA schedules successively in the $\lfloor \frac{n_c}{g} \rfloor$ first slots the sink children of the $\lfloor \frac{n_c}{g} \rfloor$ first groups. We distinguish two subcases:
    * If the last group contains exactly g sink children, MODESA repeats the $\frac{n_c}{g}$ first slots to schedule the sink children until they have transmitted all their messages.
    * Otherwise the behavior of MODESA differs a little insofar as in the $\lceil \frac{n_c}{g} \rceil^{th}$ slot, it schedules the $p < g$ sink children of the last group with the $g-p$ first sink children of the first group. It continues with in the next slot, the p last sink children of the first group with the $g-p$ first sink children of the second group...

    In both cases, MODESA maintains busy the g interfaces of the sink as long as possible. It is not possible to use a lower number of slots to schedule the sink children: MODESA is optimal for balanced trees.

■

Notice that in the example given in the proof of Theorem 5, MODESA provides 8 slots, the optimal number.

D. Variants of MODESA

MODESA has been presented with the greedy variant for channel allocation. In this variant, channels are allocated in arbitrary order that is the same for all time slots. We notice that the first channel can be saturated, whereas the others can be empty. In order to provide load balancing, we propose several variants:

1) Round Robin: channels are considered in a circular order, depending on the current time slot.
2) Least used channel: we first favor the least used channel among the whole network (i.e. the channel with the smallest number of transmissions).
3) Least used 2-hop channel: on any node up to 2-hop from the selected node, we compute the maximum or average load of any channel and select the channel with the least load.

VI. Performance evaluation of MODESA

We developped a GNU Octave [9] based simulation tool and performed simulation with different WSN topologies. We suppose that all 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 channels available at each node is equal to the maximum of 2 and the number of sink interfaces is equal to one, unless otherwise stated. We make vary N the number of nodes from 10 to 100 and generate random trees where the maximum number of children is limited to 3. We use Galton-Watson process as a branching stochastic process to generate random trees: each node gives birth to a random number of children independently of the others and according to the same distribution. In addition, we assume that the only existing links are those in the tree. In the following, each result is an average of 20 runs for small topologies ($\leq$ 30 nodes) and 100 runs for large topologies.

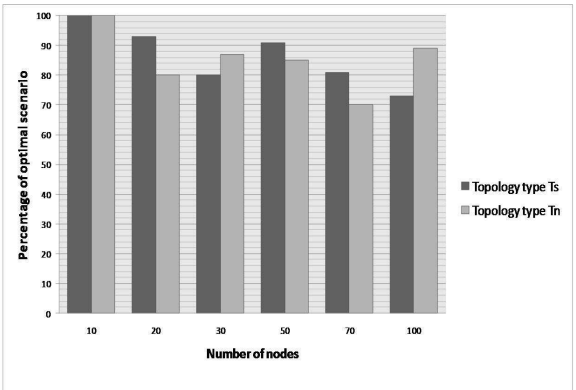


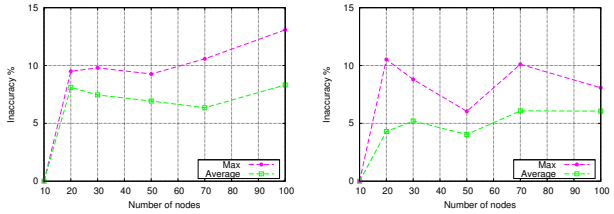
Fig. 3. Optimality of MODESA in T_S and T_N configurations.

We first evaluate the optimality of MODESA considering two types of configurations, depending on whose factor imposes the optimal schedule length:

- The size of the most populated subtree, denoted type T_S, where $2n_1 - 1 + \delta > \lceil \frac{N-1}{g} \rceil$;
- The number of nodes and g, denoted type T_N, where $2n_1 - 1 + \delta \leq \lceil \frac{N-1}{g} \rceil$.

As depicted in Figure 3, we notice that in random trees with 100 nodes, MODESA is optimal in respectively 89% of the T_S configurations tested and 74% of the T_N configurations tested, respectively. This illustrates the merit of MODESA.

We now only consider configurations where MODESA is not optimal and quantify the drift between MODESA and an optimal schedule with regard to the number of slots needed. For this purpose, we evaluate for each type of configuration the schedule length obtained by MODESA, denoted L_{MODESA} and the optimal one, $L_{Optimal}$. We compute the inaccuracy of MODESA as $\frac{L_{MODESA} - L_{Optimal}}{L_{Optimal}}$. The inaccuracy of MODESA is depicted in Figures 4(a) and 4(b) for topologies of type T_S and T_N respectively. The maximal inaccuracy is 13% for the T_S and 10.5% for the T_N configurations, demonstrating the very good behavior of MODESA. The average inaccuracy in both T_S and T_N configurations is below 8.5%.



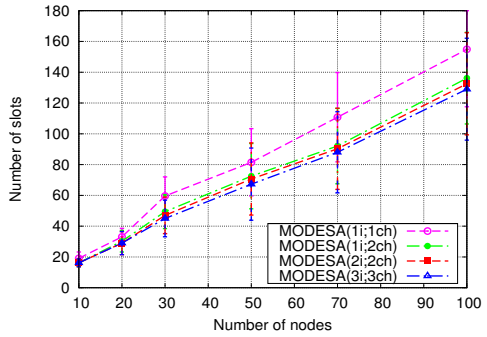
(a) (b)

Fig. 4. Inaccuracy of MODESA in (a) T_S configurations (b) T_N configurations.

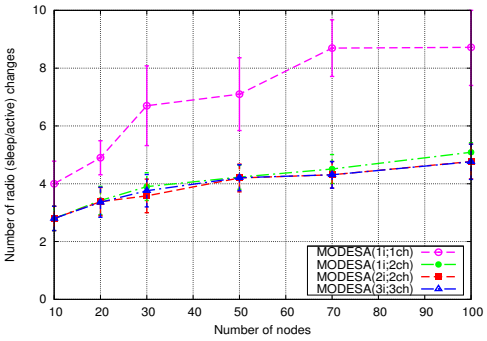
We now detail the performances of MODESA in terms of schedule length, throughput, radio switches per node in a schedule and buffer size, considering various multichannel configurations.

Figure 5(a) depicts the total number of slots for MODESA considering different numbers of channels. We observe that even when the sink has a single radio interface, the use of multichannel drastically decreases the TDMA cycle length: for example with 100 nodes, the use of only two channels decreases the number of slots by 12.82% (20 slots). With a single interface of the sink, the best performances are achieved when the number of channels is equal to two. When the sink is equipped with multiple interfaces, we observe also a reduction of the number of slots. Moreover, we notice that there is no interest to equip the sink with a number of radio interfaces greater than the number of its children. We also observe that it is useless to have a number of channels higher than the number of sink interfaces when this latter is greater than 1.

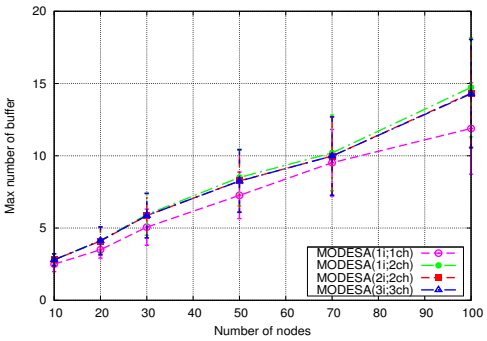
In addition, reducing the radio state switches is crucial to save the energy of sensors. Therefore, for each node, we compute its number of switches as the number of times the node alternates between the sleep and active radio states in a cycle. Figure 5(b) shows that the number of switching between active ad sleep states is decreased by the only use of two



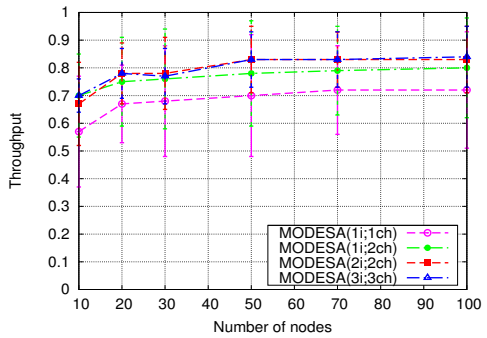
(a)



(b)



(c)



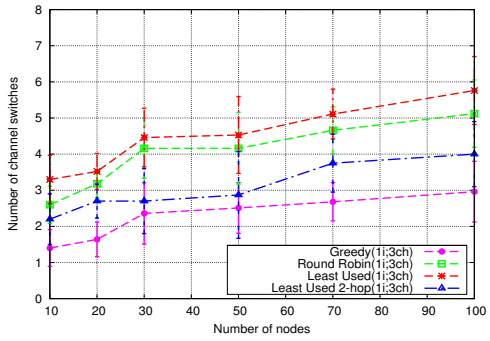
(d)

Fig. 5. MODESA performance regarding (a) the number of required slots (b) the number of radio switches (c) the maximum buffer size (d) the throughput.

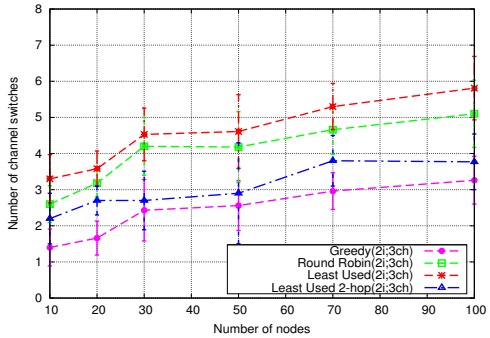
channels for a sink with one or two interfaces, and a number of channels equal to the number of sink interfaces otherwise. Another parameter which directly affects the execution of MODESA is the maximum number of buffers. So we also evaluate the maximum number of buffers required in a node during a TDMA cycle. Figure 5(c) demonstrates that MODESA with a single channel ensures the smallest number of required buffers. In multichannel wireless networks, the single radio interface uses more buffers than multi radio interfaces. This can be explained by the parallel transmissions allowed by the presence of at least two channels.

Finally, we evaluate the throughput. This latter is defined as the number of times where the sink receives packets divided by the total number of slots. As illustrated in Figure 5(d), the use of multi radio interfaces achieves higher throughput. Moreover, in case of sink equipped with one radio interface, the use of multichannel guarantees higher throughput than the single channel network.

We now focus on different variants of MODESA that tend to balance a channel load as said in Section V. We notice that all these variants of MODESA provide the same number of slots. On the first hand, we analyse the impact of variants of MODESA on the number of channel switches per node.



(a)



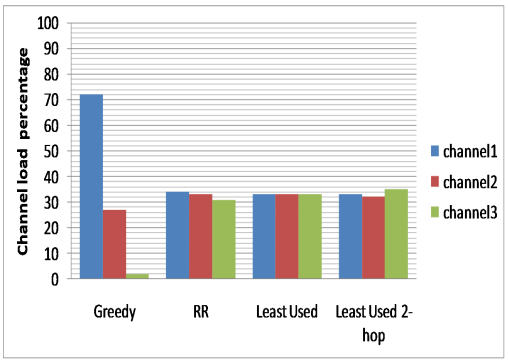
(b)

Fig. 6. Channel switching in case of sink equipped with (a) 1 interface and 3 channels (b) 2 interfaces and 3 channels

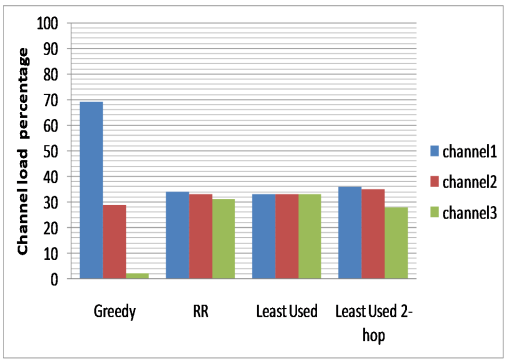
Figure 6 shows that all these variants of MODESA achieve a small number of channel switches leading to a minimized medium access time. As expected, the greedy variant achieves

the lowest number of switches. As illustrated in the Figures 6(a) and 6(b), greedy has the same behaviour in the two curves. This can be explained by the fact that with one or two interfaces for the sink, two channels are sufficient to schedule nodes transmissions. The greedy variant does not use the third channel.

Figure 7 depicts channels loads achieved by different MODESA variants. As illustrated, the least used variant outperforms greedy, Round Robin and least used 2-hop in balancing the number of times a particular channel is used. Hence, it minimizes the co-channel interferences. However, Round Robin provides the best trade off between the implementation simplicity and a channel load balancing.



(a)



(b)

Fig. 7. Channel load in topology with 100 nodes and sink equipped with a) 1 interface and 3 channels (b) 2 interfaces and 3 channels

VII. Conclusion

In this paper we have shown how multichannel communications contribute to achieve the application requirements in an aerospace WSN. They increase network capacity by allowing parallel transmissions and improve communication reliability by avoiding noisy channels. Our key results in this paper are twofold. First, we have generalized the state of the art results for the optimal number of slots required by a raw data convergecast in multichannel WSNs in case of a sink equipped with $k \geq 1$ radio interfaces. We have shown also that for a sink with a single interface, it is useless to have a number of channels higher than two, since the optimal number of slots is already reached with two channels. For a sink with k multiple radio interfaces, $k > 1$, k channels are sufficient to get the optimal schedule length. As a second contribution, we have proposed our MODESA algorithm and have proved its optimality in many multichannel topologies of WSNs. Simulation results show that MODESA needs a small buffer size and reduces the number of radio active/sleep switches per node in a cycle. In addition, we described variants of MODESA that balance traffic load between channels. Furthermore, we can improve MODESA by adopting the behavior of FlipFlop when there are exactly one or two groups, making it optimal even in case of unbalanced trees.

References

[1] R. Soua, P. Minet, *A survey on multichannel assignment protocols in wireless sensor networks*, IFIP Wireless Days, Niagara Falls, Ontario, Canada, October, 2011.

[2] J. Song, S. Han, A. K. Mok, D. Chen, M. Lucas, M. Nixon, *WirelessHART: Applying Wireless Technology in Real-Time Industrial Process Control*, In. Proc. IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS'08), St. Louis, MO, United States, April, 2008.

[3] H. Zhang, P. Soldati, M. Johansson, *Optimal Link scheduling and channel Assignment for convergecast in linear WirelessHART Networks*, In. Proc. international conference on Modeling and Optimization in Mobile, Ad Hoc, and Wireless Networks (WiOPT'09), Seoul, Korea, June, 2009.

[4] Y. Wu, J. Stankovic, T. He, S. Lin, *Realistic and efficient multi-channel communications in wireless sensor networks*, In. Proc. INFOCOM'08, Phoenix, AZ, USA, 2008.

[5] Y. Kim, H. Shin, H. Cha, *Y-MAC: An Energy-Efficient Multi-channel MAC Protocol for Dense Wireless Sensor Networks*, In. Proc. IPSN'08, St. Louis, Missouri, USA, 2008.

[6] B. Raman, K. Chebrolu, S. Bijwe, V. Gabale, *PIP: A Connection-Oriented, Multi-Hop, Multi-Channel TDMA-based MAC for High Throughput Bulk Transfer* , In. Proc. ACM Conference on Embedded Networked Sensor Systems (Sensys 2010), Zurich, Switzerland, November, 2010.

[7] 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.

[8] http://www.gnu.org/software/glpk/

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