Cluster-based Scheduling for Cognitive Radio Sensor Networks

Hanen Idoudi, Ons Mabrouk, Pascale Minet, Leila Saidane

To cite this version:

Hanen Idoudi, Ons Mabrouk, Pascale Minet, Leila Saidane. Cluster-based Scheduling for Cognitive Radio Sensor Networks. Journal of Ambient Intelligence and Humanized Computing, Springer, 2018. hal-01870909

HAL Id: hal-01870909

https://hal.archives-ouvertes.fr/hal-01870909

Submitted on 10 Sep 2018

Cluster-based Scheduling for Cognitive Radio Sensor Networks

Hanen Idoudi[1], Ons Mabrouk[1], Pascale Minet[2], Leila Azouz Saidane[1]

[1] National School of Computer Science, University of Manouba, 2010 Manouba, Tunis, Tunisia
[1] Emails: hanen.idoudi@ensi.rnu.tn, mabrouk.ons@gmail.com, leila.saidane@ensi-uma.tn
[2] Inria de Paris, France
[2] Email: pascale.minet@inria.fr

Abstract. In this paper, we define a cluster based scheduling algorithm for Cognitive Radio Sensor Networks (CRSNs). To avoid inter-clusters collision, we assign fixed channels only to nodes having one-hop neighbors out of their clusters. We denote these nodes as specific nodes. Previous studies assign distinct channels to whole neighbor clusters to avoid inter-clusters collision. Our objective is to optimize the spatial reuse and to increase the network throughput while saving sensors energy. We start by assigning channels only to the specific nodes. Once the problem of inter-clusters collision is solved, each cluster head (CH) schedules the transmissions in its cluster independently. For the cluster members that are specific nodes, the CH assigns only time slots because the channel assignment is already done. For other cluster members (CMs) (not specific nodes), the CH assigns the pair (channel, slot). Two solutions are proposed in this paper to schedule the CMs: The Frame Intra Cluster Multichannel Scheduling algorithm denoted Frame-ICMS and the Slot Intra Cluster Multichannel Scheduling algorithm denoted Slot-ICMS. We evaluate the performance of these algorithms in case of accurate PUs activity detection and in case of bad PUs activity estimation. We prove that our proposals outperform an existing one especially in terms of energy saving.

Keywords: Cognitive radio sensor networks, scheduling, clustering

1 Introduction

The increasing usage of wireless communications raises the challenge of spectrum utilization efficiency challenge. Cognitive radio technology has emerged as an effective solution to allow other users, called cognitive radio users or secondary users (SUs), to share the underutilized spectrum provided that there is no interference with primary users (PUs). Once a SU detects the presence of PU, it has to switch to another available channel, not occupied by PU.
Dynamic spectrum access stands as a spectrum-efficient communication paradigm

for Wireless Sensor Network (WSN). These latter face an increased level of interferences from various wireless systems operating over the unlicensed frequency bands such as WiFi, WIMAX, Bluetooth, etc.
Cognitive Radio Sensor Network (CRSN) is a new sensor networking paradigm that adopts the cognitive radio capabilities to sensor networks. CRSNs come as a solution to opportunistically exploit the unused parts of licensed spectrum. Offering to sensor nodes temporary usage of the available licensed channels improves the utilization efficiency of the spectrum itself and provides improved quality of service with respect to existing wireless technologies.
Cognitive radio users are able to access any portion of spectrum. Significant interference can be caused to licensed users and to other secondary users. Schedule based MAC protocols for cognitive radio networks have been proposed to address this issue. Also, there are still numerous challenges that need to be addressed. Firstly, the common control channel problem is still mostly unsolved in cognitive radio technologies. Secondly, it has been proved (OFCOM, 2009) that single-user detection strategies do not perform well enough to detect primary user activity. Thirdly, many solutions have been designed for only limited-size networks. A possible solution to address these issues is to divide the network into clusters. Unlike previous studies, that assigns distinct channels to neighbor clusters to avoid collision, we profit from the totality of the spectrum left available by the PUs to increase the parallelism of transmissions made by SUs. In this paper, we focus on how to build a schedule for cluster-based CRSN. Our objectives are to: *(i)* avoid inter-clusters collisions, *(ii)*avoid interference with PUs and other cluster members (CMs), *(iii)* minimize the number of slots used to schedule the CMs which minimizes the data gathering delay and the switching to another available channel if the PU notifies its presence, as well as *(iv)* reduce the activity period of nodes, ensuring, therefore, energy efficiency and prolonging network lifetime.

In order to avoid the inter-clusters collision, our cluster-based channel assignment algorithm affects channels to the nodes having one-hop neighbors out of their clusters. We denote these nodes as specific nodes in the remaining of this paper. After assigning fixed channels to specific nodes, each cluster head (CH) can schedule the transmissions of its CMs. The CH tries to schedule non interfering nodes in the same time slot and on the same available channel to minimize the schedule length and so, the switching to another available channel if the PU notifies its presence.
To allocate spectrum to SUs, the CH senses the channel for a certain period of time and stores all the channel states. Then, it estimates the parameters of the PU behavior model. It is widely accepted that the PU behavior follows a Markov model (Tumulura et al, 2011; Rashid et al, 2009). The transition probabilities are assumed to be known to the SUs. However, in real applications, it is almost impossible for a SU to obtain these parameters in advance. That is why, in this paper, we propose two solutions. The Frame Intra Cluster Multichannel Scheduling algorithm, denoted Frame-ICMS and the Slot Intra Cluster Multichannel Scheduling algorithm, denoted Slot-ICMS. The Frame-ICMS solution is based on dividing time into frame periods. The frame period consists of an

access control slot and a number of time slots proportional to nodes traffic. In this solution, the CH estimates the parameters of the PU behavior model. Then, it forwards the scheduling algorithm during the control slot. For the remaining of the frame period and at each slot, only selected nodes wake up to send or receive packets. In the Slot-ICMS solution, since the transition probabilities of the PUs cannot be known by the CH in real application, this latter senses the available channels at the beginning of each slot and selects the list of nodes to transmit. Not selected nodes turn off their radio for the remaining time slot. The two traffic-aware intra cluster multichannel scheduling algorithms assign only a time slot for the specific nodes that already have a fixed channel assigned, and a pair (time slot, channel) for other nodes. The purposes are to allocate a number of time slots proportional to nodes traffic and to minimize the delay for data gathering.
In this respect, our work is different from prior works about the assignment of the channels in each cluster. In addition, our work takes advantages from spatial reuse and allows non interfering SUs to access the same available channel in the same time slot which minimizes the data gathering delay and the switching to another available channel in case of PU presence. In our study, we assume ideal physical layer. To the best of our knowledge, ours is the first study on scheduling in cluster-based CRSN taking into consideration inter-cluster collisions.
The rest of this paper is organized as follows. After a state of the art in Section 2, we solve in Section 3 the problem of inter-clusters collision. Then, we propose two solutions for the intra cluster multichannel scheduling algorithm in Section 4. Performance in terms of schedule length, energy consumption and throughput are evaluated by simulation in Section 5. Finally, we conclude in Section 6.

2 Related work

Many algorithms for dynamic spectrum management in cognitive radio networks were proposed (Tragos et al, 2013). The majority of these solutions aimed at gaining more quality of service in terms of throughput, delay, interference or fairness. To reduce the scheduling delay, authors in (Gozupek and Alagoz, 2009) propose to order nodes transmissions according to their number of packets. Authors in (Gozupek et al, 2012) aim to maximize the number of satisfied users by, in priority, assigning resources to the unsatisfied SUs who would either come closer to their satisfaction limit or become satisfied without gaining much additional resources. In (Tumuluru et al, 2010), the Central Scheduler finds the channel-user pair with the highest value of the expected frame throughput and broadcasts it to all the users.
Unlike in CRN, spectrum access in cognitive radio sensor networks (CRSN) has to take into consideration the energy efficiency and so, network lifetime. The total residual energy of the whole network was considered as the metric for channel assignment in (Li et al, 2011). Authors in (Li et al, 2014) further improve the energy efficiency by suggesting to adapt packets size and to prolong network lifetime by residual energy balancing. As the dynamic spectrum access is greatly

affected by primary user behavior, we proposed in our previous works (Mabrouk et al, 2014a,b) various scheduling algorithms that minimize the schedule length. With this metric, sensor nodes with cognitive radio transmit all their packets as soon as possible before the PUs start of activity, which minimizes the switching to another available channel, saving thus, cognitive radio sensors energy.

All the former proposals perform dynamic spectrum management either in a centralized or completely cooperative way, whereas, in large scale or dense networks, applying a hierarchical topology is known to be more effective.

Clustering is a topology management mechanism that organizes nodes into logical groups in order to provide network-wide performance enhancement. Each cluster comprises two kinds of nodes, namely cluster head (CH) and member nodes or cluster members (CM). CH and its CMs communicate regularly among themselves. It is the intra-cluster communication.

Many clustering approaches were proposed for WSNs, most of them aim at improving scalability of network functioning and reducing energy consumption of sensors (Abbasia and Younis, 2007). One of the well known and studied protocols for WSNs is LEACH (Heinzelman et al, 2000).

Clustering techniques were adopted in cognitive radio networks (Yau et al, 2014) due to their great advantages. Firstly, clustering improves scalability through the reduction of communication overhead while CMs exchange information with their respective CHs only. Secondly, clustering improves stability since any changes on network dynamics cause local updates among member nodes and their respective CHs. Thirdly, in clustered networks, CHs and member nodes can cooperate more efficiently to detect the primary user activity. Hence, clustering can facilitate the SUs scheduling and enhance greatly the spectrum detection and management. Traditionally, the cluster structure changes with network topology. In CR networks, the channel availability of each node changes with time and location. This brings new challenges to clustering in CR networks. The lack of common channels among nodes in a cluster may cause loss of connectivity between CH and its CMs.

In CRNs, clustering formation and maintenance can be performed according to four types of metrics:

– channel availability (Ozger and Akan, 2013; Li et al, 2012; Liu et al, 2012; Lazos et al, 2009; Asterjadhi et al, 2010),
– geographical location (Ozger and Akan, 2013; Zhang et al, 2011; Wei and Zhang, 2010),
– signal strength and channel quality (Li et al, 2012; Ramli and Grace, 2010), as well as
– node degree (Ozger and Akan, 2013; Li and Gross, 2011).

Nodes may form clusters with single hop (Ozger and Akan, 2013; Li and Gross, 2011; Liu et al, 2012) or multiple hops (Huang et al, 2011; Asterjadhi et al, 2010). Single-hop clusters enhance network stability but suffer from inter-cluster communication delays. However, multiple-hop clusters reduce the number of clusters in the network and provide lower inter-cluster communication overhead. To avoid collision, CHs can be organized in a tree topology and wake up in sequential order according to their depth in this tree (Zhang et al, 2009). Also, neighboring clusters can use distinct channels. Each cluster C_i has its subset of spectrum S_i (Sudhanshu et al, 2015).

Some clustering approaches specifically dedicated to CRSN were recently proposed. In (Pei et al, 2015), authors consider that energy is the ultimate parameter to consider for clustering in CRSN. They define, at this end, a low-energy adaptive uneven clustering hierarchy for CRSN, which optimizes energy consumption through creating uneven clusters to balance energy consumption among the cluster heads under multiple hops transmission. In (Gajanan and Pingat, 2016), authors define a clustering technique for energy reducing and spectrum management in CRSN. First, CHs are elected according to the highest level of residual energy. After, each CH will assign higher amount of resources to closer nodes than to cell edge nodes. Authors base on the fact that closer nodes to the CH have higher frequency range. Cooperative spectrum sensing (ECS) for CRSNs (Rauniyar and Shin, 2015) is an energy-efficient clustering aiming at increasing energy efficiency, network lifetime, network stability, and optimal cluster-head selection process. This schema exploits a duty cycling approach to let sensors switch between Awake and Sleep modes to save their energy. More clustering approaches for CRSNs can be found here (Joshi and Kim, 2016; Wu and Cardei, 2016).

Regarding spectrum management, most of the cited works assign to each cluster its subset of spectrum to avoid collision between neighbouring clusters and allow the CMs to access to their set of spectrum dynamically and independently. However, it limits the number of channels used in each cluster, especially as the channel availability depends on the activity of PUs. In addition, this solution decreases the performances of the proposed multi-channel CRSN scheduling algorithms. None of the studied works exploited at most spatial re-use between neighbouring clusters. Moreover, no former study considered the inter-clusters interferences problem.

In the following sections, we detail our solution for clustering and transmissions scheduling in CRSN that tackles both intra and inter cluster collisions while enhancing the spectrum management by the mean of a more efficient channels re-use. We define at first our clustering technique then the proposed algorithms for inter and intra-clusters scheduling.

3 Clusters formation and channel allocation for specific nodes

We define in this section our clustering model which allows first to determine the clusters topologies and identify the specific nodes in order to assign them fixed channels.

3.1 Clusters formation

We use the *k-means* (Kanungo et al, 2002) algorithm to form our clusters. *k-means* clustering aims to partition N nodes into k clusters using these steps:

1. Define k centers, one for each cluster,
2. Associate each node to the nearest center,
3. Update clusters' centers: the new center is the barycenter of the cluster formed in the previous step,
4. Iterate (steps 2) and 3)) until centers do not move anymore.

Algorithm 1 illustrates the *k-means* clustering algorithm. In figure 1, the *k-means*

Algorithm 1 *k-means* clustering algorithm

```
1: Input:
     – $N = \{n_1, n_2, n_3, ..., n_N\}$ : the set of nodes.
     – $k$: the number of clusters.
2: Output: The partition of $N$ nodes into $k$ clusters: each node belongs to only one cluster.
3: Initialization phase
4: $M \leftarrow \{m_1, m_2, ..., m_k\}$ // Select randomly $k$ cluster centers
5: /* Let $difference$ denote a vector of $k$ elements initialized to 1
6: while $i \neq k$ do
7:    $difference(i) \leftarrow 1$
8: end while
9: while $\sum_{i=1}^{k} difference(i) > 0$ do //while other cluster centers are reassigned
10:    while $i \neq k$ do
11:       /* Assign each node to the cluster $S_i$ whose distance $d$ from the cluster center $m_i$ is minimum of all the cluster centers. */
12:       /* The distance $d$ can be a Squared Euclidean distance, a Sum of absolute differences, etc. */
13:       $S_i \leftarrow \{n_p : d(n_p, m_i) \leq d(n_p, m_j) \forall j, 1 \leq j \leq k\}$
14:       $oldPosition_i \leftarrow m_i$
15:       /* Recalculate the new cluster center */
16:       /* Set the position of each cluster center to the mean of all nodes belonging to that cluster */
17:       $m_i \leftarrow \frac{1}{|S_i|} \sum_{n_j \in S_i} n_j$
18:       $newCenter_i \leftarrow m_i$
19:       $difference(i) \leftarrow \|oldPosition_i - newCenter_i\|^2$
20:       /* $difference = 0$ if there are no changes in any cluster centers */
21:    end while
22: end while
```

algorithm partitions 20 nodes, randomly scattered over a region ($50m \times 50m$), into 2 clusters. Nodes with the same color belong to the same cluster. The center of each cluster is marked by the symbol '**x**'.

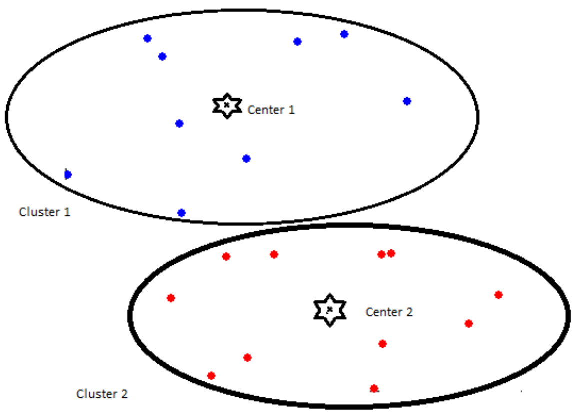


Fig. 1: K-means partition: N=20,k=2.

3.2 Channel allocation for specific nodes

In this section we study the channel assignment problem for nodes having one-hop neighbors out of their clusters.

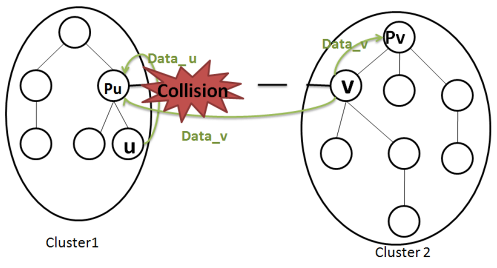


Fig. 2: Interfering nodes belonging to different clusters.

We assume in our model that a communication tree has been established in each cluster, such that the root of the tree is the CH. As shown in Figure 2, v and P_u, the parent of u, are two 1-hop neighbors not in the same cluster. So, v and u cannot transmit in the same time slot on the same channel. A collision can occur. To allow each CH to run its scheduling independently, we have to assign to nodes, having neighbors out of their clusters, different channels to avoid collision when these nodes transmit packets during the same time slot.

Assumptions and basic concepts

- **Network:** The network is divided into k clusters. Each cluster is composed by a CH and secondary users (SUs). The sink executes the *k-means* algorithm to form the clusters. k is a network parameter.
- **Spectrum:** The available spectrum is divided into $nb_{channel}$ non overlapping channels. The channels are assumed to be stable and equivalent for the SUs, so for any SU there is no preference for a channel or another.
- **Primary Users:** Each PU randomly accesses its licensed channel to transmit its messages.
- **Secondary Users:** SUs are randomly deployed in the network and profit from the vacant licensed channels to transmit their packets. Each SU is equipped with a single radio interface. The sensors' position is known in advance by using the GPS system. In addition, these positions do not change since the network is static. Also, SUs have the same behavior in terms of access type, connection quality, number of packets to send, etc.
- **Application**: In the intra-cluster communication we focus on data gathering applications where SUs sense data and transmit them to their CH. We build a spanning tree rooted at the CH. Any node should transmit its packets to its parent. Only CHs have to aggregate packets and transmit the aggregated packets to the sink.
- **Slot size and immediate acknowledgment:** We assume that the size of a time slot allows the transmission of one packet and its acknowledgment. During the slot, the sender transmits its packet to its receiver. This latter, denoted, the parent of the sender, acknowledges the received packet in the same time slot. We refer to it as immediate acknowledgment.
- **Conflicting model:**The conflicting model defines which set of links can be active (transmission/reception) simultaneously. We adopt the graph-based model. Two nodes are said conflicting if and only if they can not transmit in the same time slot and on the same channel. Each SU has conflicting nodes in its cluster. Some SUs have also conflicting nodes out of their clusters. Since we assume an immediate acknowledgment, the set of nodes conflicting with any node u, is given by:
  1. the node u itself,
  2. the node $Parent(u)$, the parent of u,
  3. all the nodes that are 1-hop away from u or $Parent(u)$,
  4. all the nodes whose parent is 1-hop away from u or $Parent(u)$.
- **Conflicting clusters:** Two clusters $c1$ and $c2$ are said conflicting if and only if there is at least one node $u \in c1$ and a node $v \in c2$ that are conflicting.
- **Traffic model**: Each node, except the sink, generates a random number of packets (p>1). Each node, except the sink,transmits to its parent in the routing tree all the packets it generates as well as the packets received from its children.
- **Physical layer**: In this paper, we assume that there is no message loss. That is why the number of slots assigned to a node is equal to the number of message it has to transmit. Notice that this assumption can be relaxed by assigning a number of slots equal to the number of messages to transmit multiplied by a link reliability coefficient.

The goal of the channel assignment problem is to attribute a fixed channel to each node having at least one 1-hop neighbor out of its cluster. The channel assignment problem must meet the following rule: two conflicting nodes are not assigned the same channel.

Discovery protocol for neighbors out of the cluster In order to discover neighbors out of its cluster, each SU needs to provide information about all its 1-hop neighbors. This process starts with a SU broadcasting a Hello message. This message contains the identity of SU and the identity of its cluster. Receiving Hello messages from all SUs in its neighborhood, each SU builds the list of its 1-hop neighbors out of its cluster.
Let NOC_u denote the set of neighbors of u that are out of its cluster. If $|NOC_u| \neq 0$, u must transmit to its CH the pair $(n_{id}, n_{cluster-id})$ for any node n in NOC_u. Hence, $n_{cluster-id}$ and $u_{cluster-id}$ are two conflicting clusters. Let $ClusterConf_c$ denote the set of clusters in conflict with the cluster c and

$$SetClusConf_c = ClusterConf_c \bigcup \{c\}. \qquad (1)$$

Cluster-based channel assignment for nodes having neighbors out of clusters

Principles Our aim is to design an algorithm that assigns channels to nodes having neighbors out of their clusters. Let $NNOC$ denote the set of nodes having neighbors out of their clusters. This algorithm is based on the following principles:

1. Each cluster has its list of conflicting clusters.
2. Each cluster has a priority. This latter indicates the order according to which the CHs assign the available channels to its nodes having neighbors out of their clusters.
3. The CH of the cluster with the highest priority, among all clusters in its list of conflicting clusters, assigns the available channels to its nodes having neighbors out of its clusters and forwards a control packet containing the pair (channel assigned to each node, the parent of the node).

We will explain why the CH has to send the identity of the parent of each node in $NNOCc$ based on the example illustrated by Figure 2. Let CH1 (respectively CH2) be the cluster head of cluster 1 (respectively cluster 2). Let u be a member of cluster 1, and v a member of cluster 2 such that v is 1-hop away from P_u, the parent of u. Suppose that the cluster 1 has the highest priority. Suppose that CH_1 assigns channel 1 to node u. v cannot use channel 1 because v is 1-hop away from P_u. To inhibit v to use channel 1, CH_2 needs to know the channel assigned to the parent of u.

Algorithm Using the notations given in Section 3.2, we now present the channel assignment algorithm for $NNOC$ illustrated by Algorithm 2. To assign channels to the $NNOC$, each CH calculates the priority of its cluster c, denoted $PrioCluster_c$. The priority of a cluster is equal to the number of nodes in this cluster having one-hop neighbors out of the cluster. In other words, let $NNOCc$ denote the set of nodes having neighbors out of their cluster c.

$$PrioCluster_c = |NNOCc|. \tag{2}$$

The CHs of clusters in the same $SetClusConf_c$ must broadcast a control packet containing the priority of their clusters. The cluster with the highest priority is the cluster with the highest $|NNOCc|$. The CH of this cluster assigns the available channels to its nodes having neighbors out of the cluster and forwards a control packet containing the channel assigned to each node in $NNOCc$ and its parent.

Algorithm 2 Channel assignment for {NNOC}

```
step 1: Each CH builds $SetClusConf_c$ the list of clusters in conflict with itself

step 2: Each CH computes the priority of its cluster $PrioCluster_c$

step 3: Let $c^*$ be the cluster with the highest priority in $SetClusConf_c$. Its cluster head:

- assigns the available channels $chan_{c^*}$ to its nodes having neighbors out of its cluster
- forwards a control packet containing the information about these nodes ($node_{id}, channel_{id}, parent_{id}$)

step 4: Each CH in $SetClusConf_c \setminus \{c^*\}$ inhibits its conflicting nodes to use $chan_{c^*}$

step 5: Each CH updates its list $SetClusConf_c = SetClusConf_c \setminus \{c^*\}$

step 6: If $SetClusConf_c \neq \emptyset$, go to step 3
```

3.3 Illustrative example of channel assignment

We now explain the principles of channel assignment for $NNOC$, using the example depicted in Figure 3. The network consists of 5 clusters. Each CH calculates its priority and determines the set of its conflicting clusters.

Table 1 summarizes the priority and the set of conflicting clusters of each cluster. For instance, in the cluster $c1$ the nodes 48 and 7 have as 1-hop neighbor nodes 22 and 46 that belong to cluster $c17$. Also, in the same cluster the node 41 has as 1-hop neighbor nodes 10 and 39 that belong to cluster $c10$.
So, the cluster $c1$ has two conflicting clusters: $c10$ and $c17$. Clusters in the same

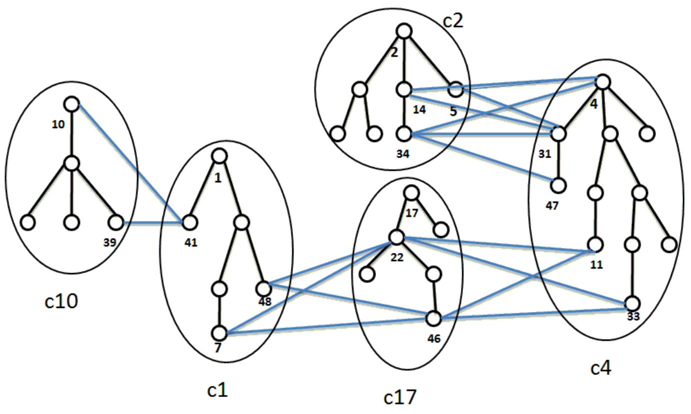


Fig. 3: Example of a network.

Table 1: The priority and conflicting clusters of each cluster

Cluster id	Priority	Conflicting clusters ($ClusterConfc$)
c1	3	c10,c17
c2	3	c4
c4	4	c2,c17
c10	2	c1
c17	2	c1, c4

$SetClusConf_c$ are sorted according to their decreasing priority. Cluster heads assign the channels used by nodes having conflicting nodes out of their cluster in the decreasing order of priority. $c1$, that has the highest priority in its $SetClusConf_c$, according to Table 2, is selected to assign channels to $NNOC$ in its cluster. However $c2$ has not the highest priority in its $SetClusConf_c$. So, in this iteration $c2$ cannot assign channels to $NNOC$ in its cluster. Only $c1$ and $c4$ can assign the channels to their $NNOC$. Once information about channels assigned in $c1$ and $c4$ are received, $c2$, $c10$ and $c17$ inhibit their $NNOC$ to use these channels. Then $c2$, $c10$ and $c17$ update their $SetClusConf_c$ and the clusters with the highest priority among the clusters in $SetClusConf_c$, having not yet assigned their channels, assigns them.

4 Frame-ICMS and Slot-ICMS

In a cluster, as we have mentioned in the previous sections, we have two types of nodes: (i) nodes and all their one-hop neighbors are in the same cluster, and (ii) nodes that have at least 1-hop neighbor out of their cluster. In Section 3 we have fixed channels for nodes belonging to the second category ((ii)), denoted

Table 2: Conflicting clusters sorted according to their decreasing priority.

Cluster id	$SetClusConfc$
c1	c1, c17, c10
c2	c4, c2
c4	c4, c2, c17
c10	c1, c10
c17	c4, c1, c17

specific nodes. In this section, our aim is to design two multichannel scheduling algorithms in a clustered based network. Both Frame-ICMS & Slot-ICMS algorithms avoid collisions between SUs and also between SUs and PUs. These scheduling algorithms must allow any node to transmit data it generates and data it receives from its children in the data gathering tree.

The Frame-ICMS solution allocates the idle channels, to nodes having data to transmit, based on the estimation of the PU spectrum occupancy. However, it is almost impossible to estimate the transition probabilities of the PU. That is why, in the slot-ICMS solution, the CH senses the activity of PUs at the beginning of each slot.

4.1 Frame-ICMS: Frame Intra Cluster Multichannel Scheduling

Based on the estimation of the PU spectrum occupancy, the Frame-ICMS, illustrated by Algorithm 3, iterates over the set of SUs in a cluster having data to transmit sorted according to their decreasing priority. We define the priority of any SU as the number of packets it has to transmit (i.e sum of generated and received packets). The selected node is the SU having the highest priority among all nodes having data to transmit. Ties are broken by the smallest identifier if there is no specific nodes, else the specific node is selected. In each iteration, in a current time slot t, the algorithm selects the node having the highest priority. This node must have an available interface as well as its parent. If so, the Frame-ICMS algorithm determines the category of this node. If it is not a specific node, it is scheduled on the first available channel provided that it does not interfere with nodes already scheduled in t. Otherwise (case of specific node), Frame-ICMS tries to schedule this node provided that it does not conflict with nodes already scheduled in its fixed channel. Frame-ICMS ends when all nodes in the cluster have sent all their packets.

4.2 Slot-ICMS: Slot Intra Cluster Multichannel Scheduling

Since, in real applications, the CH cannot obtain accurate transition probabilities parameters of PUs in advance, in the Slot-ICMS solution, each CH senses the channels, at the beginning of each slot. It detects the activity of PUs, and

Algorithm 3 Frame-ICMS scheduling algorithm

```
1: Input:
     – A directed graph $G_c = (\mathcal{V}_c, \mathcal{E}_c)$ composed of $nb_cSU$ secondary nodes in a cluster $c$, where $\mathcal{V}_c$ is the set of vertices representing the $CMs$ and $\mathcal{E}_c$ is the set of edges representing the communication links between these $CMs$. Each secondary node $u$ has $nbPacket_u$ to transmit.
     – Each secondary user $u$ has $i_u$ radio interfaces, a set of conflicting nodes $I(u)$ and one receiver, its parent $parent(u)$.
     – $nb_{channel}$ channels, each channel $ch$ has a set $Scheduled_{ch}$ of scheduled nodes on this channel per time slot.
     – Each secondary user $u$ has its fixed channel $channel_u$ if it has at least 1-hop neighbor out of its cluster $c$ ($u \in \{NNOC_c\}$)
2: Output: The scheduling of secondary nodes: a list of couples (time slot, channel) for each secondary node $v$.
3: Initialization phase
4: $t \leftarrow 1$ // current time slot
5: while $\sum_{v\in\mathcal{V}} nbPacket_v \neq 0$ do /* $\exists$ nodes having packets to transmit */
6:   For each secondary node $v$, initialize the number of available interfaces.
7:   For each channel $ch$, initialize the set of nodes scheduled on this channel: $Scheduled_{ch} \leftarrow \emptyset, \forall ch = 1..nb_{channel}$.
8:   $N_{SU} \leftarrow$ list of $SUs$ having data to transmit sorted according to the decreasing order of their priorities.
9:   /* Scheduling nodes in the time slot $t$ */
10:  $v \leftarrow$ first node in $N_{SU}$
11:  while $v$ do
12:    while ($v$ & ( $i_v = 0$) OR ($i_{parent(v)} = 0$)) do
13:      $v \leftarrow$ next node in the list $N_{SU}$
14:    end while
15:    if $v$ then
16:      /* try to schedule $v$ */
17:      (scheduled,channel) = FramOneNodSch($v, t, nb_{channel}$, $Scheduled_{ch}$, $channel_v$, $\{NNOC_c\}$);
18:      if (scheduled = true) then
19:        $N_{SU} \leftarrow N_{SU} \setminus \{v\}$ /* $v$ has been successfully scheduled*/
20:        $i_v \leftarrow i_v - 1$
21:        $i_{parent(v)} \leftarrow i_{parent(v)} - 1$
22:        $nbPacket_v \leftarrow nbPacket_v - 1$
23:        $nbPacket_{parent(v)} \leftarrow nbPacket_{parent(v)} + 1$
24:        $Scheduled_{channel} \leftarrow Scheduled_{channel} \cup \{v\}$
25:      end if
26:      $v \leftarrow$ next node in $N_{SU}$
27:    end if
28:  end while
29:  /* at this level all possible transmissions are scheduled in the current time slot $t$ */
30:  $t \leftarrow t + 1$ //next slot
31: end while
```

Algorithm 4 FramOneNodSch(node v, slot t, int $nb_{channel}$, $Scheduled_{ch}$,$channel_v$, $\{NNOC_c\}$)

```
1: $ch \leftarrow 1$
2: $tx \leftarrow false$
3: if $v \in \{NNOC_c\}$ then
4:   /*Node $v$ has at least 1-hop neighbor out of its cluster $c$ */
5:   $channel \leftarrow channel_v$
6:   /* $v$ has its fixed channel $channel_v$ */
7:   if $((I(v) \bigcap Scheduled_{channel_v} = \emptyset)$ & ( no PU on channel $channel_v$)) then
8:     /*Node $v$ can be scheduled in $t$*/
9:     $tx \leftarrow true$
10:  end if
11: else
12:   /*Node $v$ can be scheduled in $(t, ch)$*/
13:   repeat
14:     if $((I(v) \bigcap Scheduled_{ch} = \emptyset)$ & ( no PU on channel $ch$)) then
15:       /*Node $v$ can be scheduled in $(t, ch)$*/
16:       $tx \leftarrow true$
17:       $channel \leftarrow ch$
18:     else
19:       if $(ch < nb_{channel})$ then
20:         $ch \leftarrow ch + 1$ // try the next channel
21:       else
22:         $nChannelReached \leftarrow true$
23:         $tx \leftarrow false$
24:       end if
25:     end if
26:   until ($nChannelReached$ OR $tx$)
27: end if
28: return ($tx$, $channel$)
```

determines the available channels. Based on the same definition of priority as in Frame-ICMS, the CH selects the node having the highest priority and an available interface as well as its parent. Slot-ICMS illustrated by Algorithm 5 tries to schedule nodes provided that they do not conflict with nodes already scheduled in the selected channel.

Algorithm 5 Slot-ICMS scheduling algorithm

```
1: Input:
     – A directed graph $G_c = (\mathcal{V}_c, \mathcal{E}_c)$ composed of secondary nodes in a cluster $c$, where $\mathcal{V}_c$ is the
       set of vertices representing the $CMs$ and $\mathcal{E}_c$ is the set of edges representing the communi-
       cation links between these $CMs$. Each secondary node $u$ has $nbPacket_u$ to transmit.
     – Each secondary user $u$ has $i_u$ radio interfaces, a set of conflicting nodes $I(u)$ and one
       receiver, its parent $parent(u)$.
     – $nb_{channel}$ channels, each channel $ch$ has a set $Scheduled_{ch}$ of scheduled nodes on this
       channel.
     – Each secondary user $u$ has its fixed channel $channel_u$ if it has at least 1-hop neighbor out
       of its cluster $c$ ($u \in \{NNOC_c\}$)
2: Output: List of scheduled SUs in the current slot.
3: Initialization phase
4: For each secondary node $v$, initialize the number of available interfaces.
5: For each channel $ch$, initialize the set of nodes scheduled on this channel: $Scheduled_{ch} \leftarrow$
   $\emptyset, \forall ch = 1..nb_{channel}$.
6: $N_{SU} \leftarrow$ list of $SUs$ having data to transmit sorted according to the decreasing order of their
   priorities.
7: /* Scheduling nodes */
8: $v \leftarrow$ first node in $N_{SU}$
9: while $v$ do
10:   while ($v$ & ( $i_v = 0$) OR ($i_{parent(v)} = 0$)) do
11:     $v \leftarrow$ next node in the list $N_{SU}$
12:   end while
13:   if $v$ then
14:     /* try to schedule $v$ */
15:     (scheduled,channel) = SlotOneNodeSchedule($v, nb_{channel}$, $Scheduled_{ch}$, $channel_v$,
        $\{NNOC_c\}$);
16:     if (scheduled = true) then
17:       $N_{SU} \leftarrow N_{SU} \setminus \{v\}$ /* $v$ has been successfully scheduled*/
18:       $i_v \leftarrow i_v - 1$
19:       $i_{parent(v)} \leftarrow i_{parent(v)} - 1$
20:       $nbPacket_v \leftarrow nbPacket_v - 1$
21:       $nbPacket_{parent(v)} \leftarrow nbPacket_{parent(v)} + 1$
22:       $Scheduled_{channel} \leftarrow Scheduled_{channel} \cup \{v\}$
23:     end if
24:     $v \leftarrow$ next node in $N_{SU}$
25:   end if
26: end while
```

Algorithm 6 SlotOneNodeSchedule(node v, int $nb_{channel}$, $Scheduled_{ch}$,$channel_v$, $\{NNOC_c\}$)

```
1: $ch \leftarrow 1$
2: $tx \leftarrow false$
3: **if** $v \in \{NNOC_c\}$ **then**
4:   /*Node $v$ has at least 1-hop neighbor out of its cluster $c$ */
5:   $channel \leftarrow channel_v$
6:   /* $v$ has its fixed channel $channel_v$ */
7:   **if** $((I(v) \bigcap Scheduled_{channel_v} = \emptyset)$ & ( no PU on channel $channel_v$)) **then**
8:     /*Node $v$ can be scheduled*/
9:     $tx \leftarrow true$
10:  **end if**
11: **else**
12:   **repeat**
13:     **if** $((I(v) \bigcap Scheduled_{ch} = \emptyset)$ & ( no PU on channel $ch$)) **then**
14:       /*Node $v$ can be scheduled in $ch$*/
15:       $tx \leftarrow true$
16:       $channel \leftarrow ch$
17:     **else**
18:       **if** $(ch < nb_{channel})$ **then**
19:         $ch \leftarrow ch + 1$ // try the next channel
20:       **else**
21:         $nChannelReached \leftarrow true$
22:         $tx \leftarrow false$
23:       **end if**
24:     **end if**
25:   **until** ($nChannelReached$ OR $tx$)
26: **end if**
27: **return** ($tx$, $channel$)
```

5 Performance evaluation

We implemented the Frame-ICMS & Slot-ICMS algorithms using MATLAB. Our network is modeled by a connected random graph. In this graph, a set of nodes is randomly scattered over a region ($100m \times 100m$). Two nodes are connected by an edge if they are separated by a distance less than the communication range R. In our simulations, $R = 30m$. Based on this graph, we build in each cluster a shortest path spanning tree rooted at the CH. In the inter cluster communication, the CHs use higher power to transmit the aggregated information to the sink.
In our simulations, we set the number of channels $nb_{channel}$ to 4. The number of secondary users vary from 50 to 200. All nodes have a single radio interface. Each node, except the sink and the CHs, generates a random number of packets ranging from 1 to 3. In the Frame-ICMS solution, as in (Gozupek and Alagoz, 2010), the CH uses a finite state model to simulate the PU spectrum occupancy behavior. Figure 4 illustrates the finite state model of the spectrum usage behavior of the PU. The PU can be either in the ON state or in the OFF state. In the ON state, it uses a frequency f^i from a subset of F frequencies, $i \in \{1, .., F\}$. The probability of staying in the ON or OFF states is p_s. At the end of each period (for example 10 slots), each PU can either stay in the same state with probability p_s or change its state with probability $1-p_s$. We selected the p_s value as 0.9 in our simulations as in (Gozupek and Alagoz, 2010). In the following,

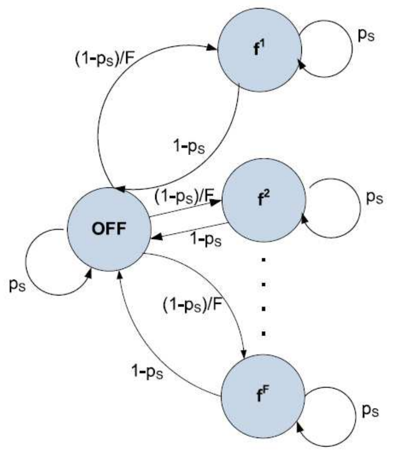


Fig. 4: PU spectrum occupancy model (Gozupek and Alagoz, 2010).

each result is the average of 20 simulation runs.

5.1 Cluster head selection

Since the Frame-ICMS & Slot-ICMS algorithms take advantage of the multi-channel aspect and allow non interfering nodes to transmit in the same time slot on the same channel, we form multi hop clusters. Let us consider the CRSN routing tree in Figure 5, where additional links are depicted by dotted blue lines. While node u is transmitting data to its parent, only node v is allowed to trans-

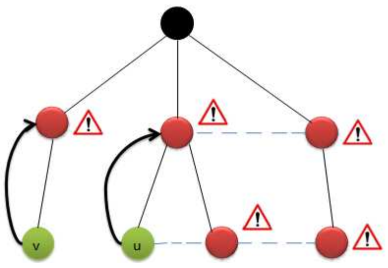
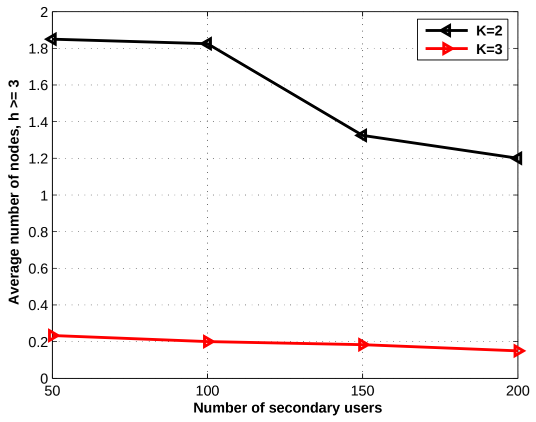


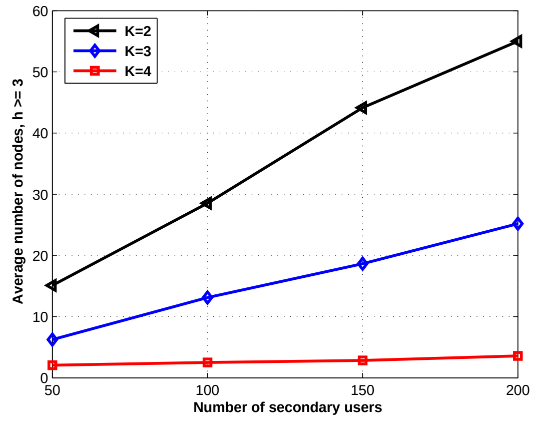
Fig. 5: Routing tree and additional links in dotted line.

mit in the same time slot and on the same channel (see the conflicting rules given in Section 3.2), since we assume an immediate acknowledgement. We can conclude that to take advantage of parallelism on the same channel, many CM must be at least 2-hops away from their CH. Also, based on the same figure (Figure 5), not all nodes that are 2-hops away from their CH can transmit simultaneously in the same channel. In other words, with a tree of depth equal to 2, we cannot profit from the spatial reuse of the channel. So, to improve the performance of our algorithm, we vary the number of clusters, k, and find the number of nodes h-hops away from their CH, $h \geq 3$ in the cluster.
We propose two strategies for the selection of the CHs:

First strategy: Nearest CM from the barycenter of the cluster In Figure 6(a), we partition our network firstly into two clusters, then into three clusters. Our CHs are the nearest nodes from their barycenter. With this strategy of CHs selection, the average number of CMs h-hops away from their CH with, $h \geq 3$, is not important ($<$ 2 in average). Figure 6(a) also shows that, if the number of nodes increases, the number of nodes h-hops away from their CH, $h \geq 3$, decreases because the density of the cluster increases and the number of nodes near (h=1) to their barycenter increases, the network deployment region being constant. Also, if the number of clusters increases, the number of CMs decreases which decreases the number of nodes h-hops away from their CH, $h \geq 3$.
We can conclude that with this strategy, we cannot profit from the advantages of our algorithms. The CH should maximize the depth of its tree.



(a) First strategy.



(b) Second strategy.

Fig. 6: Average number of nodes h-hops away from their CH, h $\geq$ 3.

Second strategy: The CM with the highest depth tree We propose, in this second strategy, that, each CM acts as a CH, builds its tree and calculates the number of nodes h-hops away from itself, $h \geq 3$. Then CMs broadcast a control packet containing this information. The CM with the highest value is elected as a CH.
As shown in Figure 6(b), with this strategy the number of nodes that are far from the CH is more important than with the first strategy. Thus, it ensures a higher channel reuse ratio. According to the Figures 6(b) and 6(a), even if the number of clusters increases, the second strategy outperforms the first one. In fact, with k=4 and N=200, the average number of nodes h-hops away from their CH, $h \geq 3$ is equal to 3.58 with the second strategy and equal only to 0.1499 when k=2 in the first strategy. Also, Figure 6(b) shows that decreasing the number of clusters increases the number of nodes in the cluster h-hops away from their CH, $h \geq 3$. It is equal, in average, to 3.58 if k=4 and to 25.18 if k=3, with a topology of 200 nodes. Since we take more advantages of spatial reuse if the number of nodes which are far from the CH is more important (Mabrouk et al, 2015), we adopt the second strategy in the remaining of this paper and assume that there are three clusters (k=3) in our topology.

5.2 Impact of the channels set allocation strategy on the schedule length

Different channel allocation strategies are possible. The first strategy allocates distinct channels to neighboring clusters. In our simulations, we have 4 channels and 3 clusters. Each cluster has its own channel. We denote this strategy, *One channel per cluster*. In the second strategy, all the clusters use the same set of channels. Only conflicting nodes use distinct channels to avoid collisions. We denote this strategy, *The same set of channels for all clusters*.

Based on the Frame-ICMS algorithm, we compare the schedule length provided by the *The same set of channels for all clusters* and by the *One channel per cluster* respectively. As depicted in Figure 7, our Frame-ICMS solution minimizes the average total number of slots compared to the previous solutions. For example, with 200 nodes, our Frame-ICMS solution decreases the number of slots by 46.16%. This is because, our solution allows many nodes to transmit simultaneously in the same slot. Only conflicting nodes have to use different channels. We also notice that the gap between the two solutions increases with the number of SUs.

5.3 Performance of Frame-ICMS and Slot-ICMS

We evaluate the performances of the Frame-ICMS and Slot-ICMS solutions in terms of schedule length, throughput and energy consumption for the two cases:

1. *Good estimation of the PU behavior*: in this scenario, the CH has correctly estimated the PU spectrum occupancy.

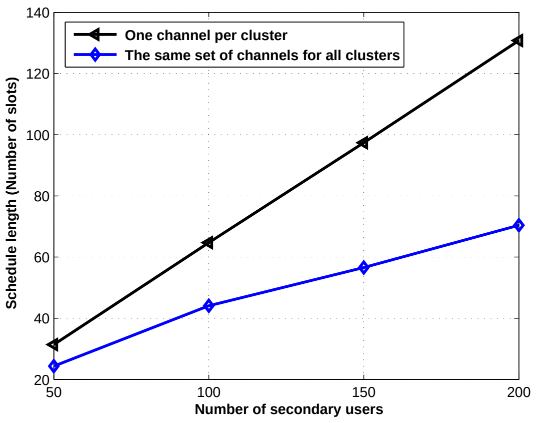


Fig. 7: Impact of the channels set allocation strategy on the schedule length.

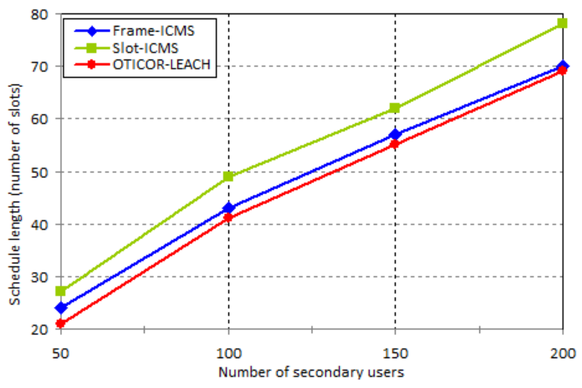
2. *Bad estimation of the PU behavior*: in this scenario, the CH has not correctly estimated the PU spectrum occupancy. For example, the CH estimates that, at a slot t, a PU channel will be available to transmit data. However, a PU can decide to use its channel at this slot. Whenever the PU returns to use its channel, the data sent by SUs on this channel are lost.

To assess the performances of Slot-ICMS and Frame-ICMS, we execute the same scenarios with OTICOR combined to LEACH in order to compare their performances. While OTICOR is a centralized scheduling mechanism, we assume that the network is clustered according to LEACH, such as, the root of LEACH tree is the sink executing the scheduling algorithm and disseminating the schedule through the hierarchy.

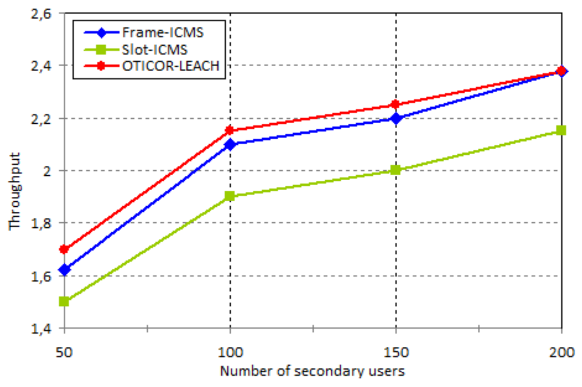
Good estimation of the PU behavior

Schedule length As shown in Figure 8(a), the average number of slots provided by the Frame-ICMS solution is smaller than the Slot-ICMS. In fact, in Frame-ICMS, the entire slot is used to transmit/receive packets. However, in Slot-ICMS, a portion of the slot is reserved to the sensing of the result of the schedule and, only, in the remaining of the slot, selected nodes can transmit. That is why, the number of slots increases in the Slot-ICMS solution. The combination of OTICOR-LEACH performs better in terms of overall schedule length, since spectrum management and slots allocation is performed in a centralized way. The sink has a complete knowledge of the network and the channels state and can perform the scheduling in an optimized way.

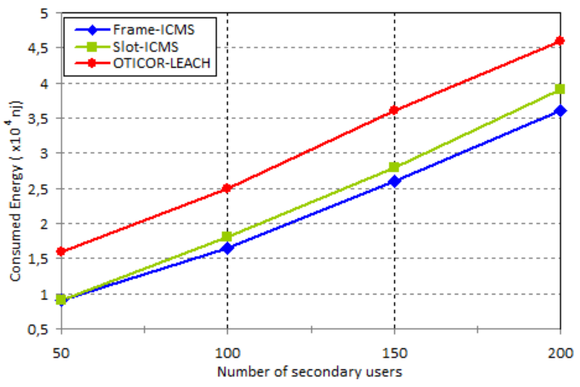
Throughput We define the throughput as the number of packets received by the CH per time slot. In other words, the throughput is the total number of packets



(a) Average number of slots.



(b) Average throughput.



(c) Average consumed energy.

Fig. 8: Performances of Frame-ICMS and Slot-ICMS solutions with good estimation of PU behavior.

transmitted by the children of the CH divided by the total number of slots. With the same amount of data to be transmitted, if the number of slots increases, the throughput decreases, as depicted in Figure 8(b).

Energy consumption In the Frame-ICMS solution, if a CM is not assigned to any channel during a time slot, it will turn to sleep state to save energy. Otherwise, it tunes its radio to the designated channel either to transmit its packet to its parent or to receive a packet from one of its children.
In the Slot-ICMS solution, at the beginning of each slot, all CMs tune their radios to the common control channel to receive the Slot-ICMS schedule. For the remaining of the slot, if a CM is not assigned to any channel, it will turn to sleep state to save energy. Otherwise, it tunes its radio to the designated channel either to transmit its packet to its parent or to receive a packet from one of its children.
To transmit a message of l bits over a distance d, the node consumes:

$$E_{Tx}(l) = lE_{elec} + l\epsilon_{fs}d^2 \tag{3}$$

Where E_{elec} represents the energy consumption in the electronics for sending or receiving one bit, and $\epsilon_{fs}d^2$ is the amplifier energy that depends on the transmitter amplifier model.
To receive this message, the node consumes:

$$E_{Rx}(l) = lE_{elec} \tag{4}$$

To receive the Slot-ICMS schedule, the node consumes E_{Rss} at the beginning of each slot. Here we use the typical values E_{elec} = 50 nJ/bit, ϵ_{fs} = 10 pJ/bit/m2 and E_{Rss}= 5nJ. The energy consumption of sensing used for the experiments is 190 nJ (Maleki et al, 2011).
We calculate the average energy consumed by CRSN nodes for the Frame-ICMS and Slot-ICMS solutions. From Figure 8(c), we find that the average energy consumed in the Slot-ICMS solution is more important than the energy consumed in the Frame-ICMS solution. It results from the additional energy that each node consumes at the beginning of each slot to receive the schedule in the Slot-ICMS solution. Both Frame-ICMS and slot-ICMS preserve better nodes energy than OTICOR-LEACH. This is mainly due to the well known bad energy performances of LEACH which requires CHs to consume high energy for clusters maintenance.

Bad estimation of the PU behavior

In this scenario, we assume that after a time equal to 40% of the schedule length, a PU uses its channel for the remaining of the schedule period whereas its channel was estimated available for the remaining of the schedule. In this scenario, in the Frame-ICMS solution, when a node is selected to transmit its packet it cannot detect the presence of the PU. With the activity of the PU, the parent of the selected node cannot receive the packet, so it cannot acknowledge the sender.

All nodes, which have not received the acknowledgement packet, have to wait until the CH broadcasts the new schedule for the remaining packets.
In the Slot-ICMS solution, the CH detects at the beginning of each slot, the activity of the PU and selects the active nodes based on this result.

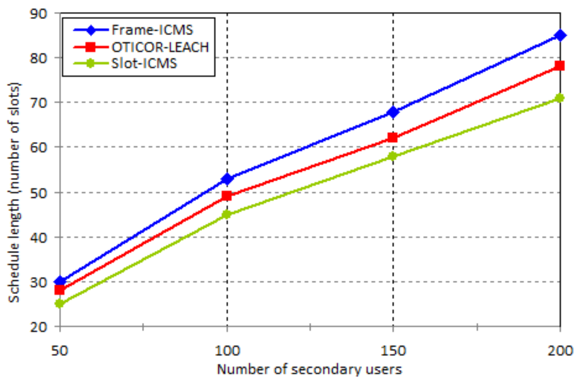
Schedule length When a CH has not correctly estimated the PU spectrum occupancy, several nodes have to retransmit their packets, which increases the schedule length. In this scenario, the Slot-ICMS solution provides an average number of slots smaller than the Frame-ICMS solution as shown in Figure 9(a). In average, the number of slots increases from 77.5167 in Slot-ICMS to 84.65 in Frame-ICMS with 200 nodes. We notice also that OTICOR-LEACH performs better than Frame-ICMS but worst than Slot-ICMS. This is explained by the fact that both in OTICOR and Slot-ICMS, the sink or CH senses the spectrum at the beginning of each slot which limits the effect of bad estimation to a slot. Slot-ICMS adapts better since a bad estimation can be locally treated within a cluster, whereas in OTICOR, the sink has to recompute and retransmit the entire schedule through the LEACH hierarchy.

Throughput When several nodes need additional slots to retransmit their packets while a PU occupies the channel, in the Frame-ICMS solution, the throughput decreases more than with the Slot-ICMS and OTICOR-LEACH, as depicted in Figure 9(b). Slot-ICMS adapts more dynamically to the PUs activity than Frame-ICMS and OTICOR-LEACH.

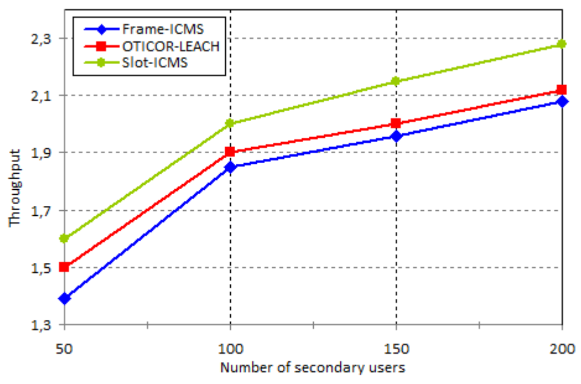
Energy consumption Figure 9(c) shows that energy consumed with both Slot-ICMS and Frame-ICMS is slightly the same but that they both outperform the OTICOR-LEACH solution. This shows that the clustering scheme adopted in our proposals conserves better nodes energy. Moreover, distributing the scheduling function among CHs balances better the energy consumption unlike in centralized approaches such as OTICOR.

6 Conclusion

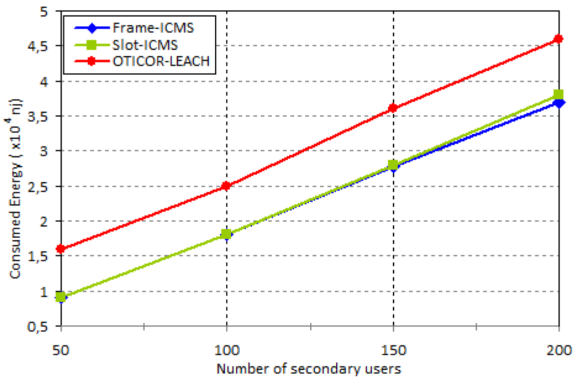
Two neighboring nodes belonging to two distinct clusters can interfere if they transmit in the same slot on the same channel. In this paper, unlike previous studies and to avoid inter-clusters collision, we do not assign fixed channels to neighboring clusters, but we assign fixed channels only to nodes having neighbors out of their clusters. So, firstly, we proposed a novel cluster based channel allocation strategy that affects fixed channels to the specific nodes. After solving the problem of inter-clusters collision, by channel allocation, each CH can schedule its CMs independently. For this purpose, we have proposed two scheduling algorithms that take into consideration the activity of PUs. In the Frame-ICMS solution, each CH uses the estimation of the PU spectrum occupancy over a frame to assign the channels sensed idle at the beginning of the frame, to the



(a) Average number of slots.



(b) Average throughput.



(c) Average consumed energy.

Fig. 9: Performances of Frame-ICMS and Slot-ICMS solutions with bad estimation of PU behavior.

CMs in its cluster. In the Slot-ICMS solution, each CH senses the activity of the PU at the beginning of each slot to take more accurate decisions and assigns idle channels to the selected nodes for the remaining slot. The CH can either succeed in its estimation or fail. We studied these two cases and compared the results with the Slot-ICMS solution.

In CRSN, we face a dilemma in building conflict-free schedules. Frequent channel sensing improves PU protection performances and throughput but depletes drastically the scarce energy budget of sensor nodes. From the PU point of view, Slot-ICMS ensures the best guarantee of free channel. Frame-ICMS may lead to the loss of data sent by the PU during a frame. From the SU point of view, Frame-ICMS that uses infrequent sensing requires less overhead, in case of good estimation. However, if the estimations are bad, Slot-ICMS provides the best result with pseudo-continuous sensing.

Bibliography

Abbasia AA, Younis M (2007) A survey on clustering algorithms for wireless sensor networks. Computer Communications 30:2826–2841

Asterjadhi A, Baldo N, Zorzi M (2010) A cluster formation protocol for cognitive radio ad hoc networks. IEEE Wireless Conference pp 955–961, DOI 10.1109/EW.2010.5483442

Gajanan H, Pingat SP (2016) Energy aware routing in next generation wireless networks using efficient resource allocation. International Journal of Science and Research (IJSR) 5

Gozupek D, Alagoz F (2009) Throughput and delay optimal scheduling in cognitive radio networks under interference temperature constraints. Journal of Communications and Networks 11:147–155, DOI 10.1109/JCN.2009.6391389

Gozupek D, Alagoz F (2010) An interference aware throughput maximizing scheduler for centralized cognitive radio networks. International Symposium on Personal Indoor and Mobile Radio Communications pp 1527–1532, DOI 10.1109/PIMRC.2010.5671962

Gozupek D, Eraslan B, Alagoz F (2012) Throughput satisfaction based scheduling for cognitive radio networks. IEEE Transactions on Vehicular Technology 61:4079–4094, DOI 10.1109/TVT.2012.2210257

Heinzelman W, Chandrakasan A, Balakrishnan H (2000) Energy-efficient communication protocols for wireless microsensor networks. 33rd Hawaaian International Conference on Systems Science (HICSS)

Huang XL, GWang, Hu F, Kumar S (2011) Stability-capacity-adaptive routing for high-mobility multihop cognitive radio networks. IEEE Transactions on Vehicular Technology 60:2714–2729, DOI 10.1109/TVT.2011.2153885

Joshi GP, Kim SW (2016) A survey on node clustering in cognitive radio wireless sensor networks. Sensors (Basel) 16

Kanungo T, Mount DM, Netanyahu NS, Piatko CD, Wu AY (2002) An efficient k-means clustering algorithm: Analysis and implementation. IEEE Transactions on Pattern Analysis and Machine Intelligence 24:881–892, DOI 10.1109/TPAMI.2002.1017616

Lazos L, Liu S, Krunz M (2009) Spectrum opportunity-based control channel assignment in cognitive radio networks. IEEE SECON pp 1–9, DOI 10.1109/SAHCN.2009.5168974

Li D, Gross J (2011) Robust clustering of ad-hoc cognitive radio networks under opportunistic spectrum access. IEEE international conference on communications pp 1–6, DOI 10.1109/icc.2011.5963426

Li X, Wang D, McNair J, Chen J (2011) Residual energy aware channel assignment in cognitive radio sensor networks. Wireless Communications and Networking Conference pp 398–403, DOI 10.1109/WCNC.2011.5779196

Li X, Hu F, Zhang H, Zhang X (2012) A cluster-based mac protocol for cognitive radio ad hoc networks. Wireless Personal Communications 69:937–955

Li X, Wang D, McNair J, Chen J (2014) Dynamic spectrum access with packet size adaptation and residual energy balancing for energy-constrained cognitive radio sensor networks. Journal of Network and Computer Applications 41:157–166

Liu S, Lazos L, Krunz M (2012) Cluster-based control channel allocation in opportunistic cognitive radio networks. IEEE Transactions on Mobile Computing 11:1436–1449, DOI 10.1109/TMC.2012.33

Mabrouk O, Idoudi H, Amdouni I, Soua R, Minet P, Saidane L (2014a) Oticor: Opportunistic time slot assignment in cognitive radio sensor networks. IEEE International Conference on Advanced Information Networking and Applications pp 790–797, DOI 10.1109/AINA.2014.96

Mabrouk O, Minet P, Idoudi H, Saidane L (2014b) Conflict-free opportunistic centralized time slot assignment in cognitive radio sensor networks. IEEE International Conference on High Performance Computing and Communications pp 421–427, DOI 10.1109/HPCC.2014.72

Mabrouk O, Minet P, Idoudi H, Saidane L (2015) Intra-cluster multichannel scheduling algorithm for cognitive radio sensor networks. International Wireless Communications & Mobile Computing

Maleki S, Pandharipande A, Leus G (2011) Energy-efficient distributed spectrum sensing for cognitive sensor networks. IEEE Sensors Journal 11:565–573, DOI 10.1109/JSEN.2010.2051327

OFCOM (2009) Digital Dividend: Cognitive Access Consultation on Licenceexempting Cognitive Devices using Interleaved Spectrum. http://www.ofcom.org.uk/consult/condocs/cognitive/cognitive.pdf

Ozger M, Akan OB (2013) Event-driven spectrum-aware clustering in cognitive radio sensor networks. IEEE INFOCOM pp 1483–1491, DOI 10.1109/INFCOM.2013.6566943

Pei E, Han H, Sun Z, S B, Zhang T (2015) Leauch: low-energy adaptive uneven clustering hierarchy for cognitive radio sensor network. EURASIP Journal on Wireless Communications and Networking

Ramli A, Grace D (2010) Rf signal strength based clustering protocols for a self-organizing cognitive radio network. International Symposiumon Wireless Communication Systems pp 228–232, DOI 10.1109/ISWCS.2010.5624375

Rashid M, MJHossain, Hossain E, VKBhargava (2009) Opportunistic spectrum scheduling for multiuser cognitive radio: a queueing analysis. IEEE Transactions on Wireless Communications 8:5259–5269, DOI 10.1109/TWC.2009.081536

Rauniyar A, Shin SY (2015) A novel energy-efficient clustering based cooperative spectrum sensing for cognitive radio sensor networks. International Journal of Distributed Sensor Networks 2015

Sudhanshu T, Sudeep T, Neeraj K, Joel J (2015) Cognitive radio-based clustering for opportunistic shared spectrum access to enhance lifetime of wireless sensor network. Pervasive and Mobile Computing

Tragos E, Zeadally S, Fragkiadakis A, Siris V (2013) Spectrum assignment in cognitive radio networks: A comprehensive survey. IEEE Communications Surveys & Tutorials 15:1108–1135, DOI 10.1109/SURV.2012.121112.00047

Tumulura VK, Wang P, Niyato D (2011) A novel spectrum-scheduling scheme for multichannel cognitive radio network and performance analysis. IEEE Transactions on Vehicular Technology 60:1849–1858, DOI 10.1109/TVT.2011.2114682

Tumuluru VK, Wang P, Niyato D (2010) An opportunistic spectrum scheduling scheme for multi-channel cognitive radio networks. IEEE Vehicular Technology Conference Fall pp 1–5, DOI 10.1109/VETECF.2010.5594393

Wei J, Zhang X (2010) Energy-efficient distributed spectrum sensing for wireless cognitive radio networks. INFOCOM IEEE Conference on Computer Communications Workshops pp 1–6, DOI 10.1109/INFCOMW.2010.5466680

Wu Y, Cardei M (2016) Multi-channel and cognitive radio approaches for wireless sensor networks. Computer Communications 94:30–45

Yau A, Ramli N, Hashim W, Mohamad H (2014) Clustering algorithms for cognitive radio networks:a survey. Journal of Network and Computer Applications 45:79–95

Zhang H, Zhang Z, Dai H, Yin R, Chen X (2011) Distributed spectrum-aware clustering in cognitive radio sensor networks. IEEE global telecommunications conference pp 1–6, DOI 10.1109/GLOCOM.2011.6134296

Zhang Z, Yu F, Zhang B (2009) A depth-based tdma scheduling for clustering sensor networks. International Conference on Frontier of Computer Science and Technology pp 261–266, DOI 10.1109/FCST.2009.19