Robust Topology using Reconfigurable Radios in Wireless Sensor Networks

Yueshi Wu and Mihaela Cardei
Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, FL 33431, USA
E-mail: {wuy2013, mcardei}@fau.edu

***Abstract*—Using SDR technology, the distributed algorithm RMA is used by sensor nodes to reconfigure their radio according to some predefined radio-modes, such that the resulting topology is connected to the sink. This mechanism reduces interference and increases data delivery rate. In this paper we extend RMA by adding a low overhead mechanism to be used in the presence of a primary user. Sensor nodes operating on the same frequency as the primary user will reassign their radio-mode such that the resulting topology does not interfere with the primary user. We analyze the performance of our distributed algorithm using ns-3 simulations.**

Keywords: wireless sensor network, radio mode assignment, multi-radio sink, distributed algorithm, primary user, robust topology.

I. Introduction and Related Works

With the development of Software Defined Radio (SDR) technology, different levels of reconfiguration within a transceiver are allowed. Multi-band, multi-channel, multi-standard, multi-service systems can be achieved with SDR. Cognitive radios enable wireless sensor networks (WSNs) to use vacant licensed channels. The advanced technology makes multi-channel communication realistic.

A cognitive radio can be programmed to transmit and receive on a variety of frequencies and to use different transmission technologies supported by its hardware design. A radio is *reconfigurable* if it has the capability to adjust its transmission parameters on the fly, without any modifications to the hardware. These parameters include spectrum band, transceiver parameters, and modulation scheme.

Cognitive radio networks can operate in both licensed and unlicensed bands [1]. When operating on licensed bands, the objective is to exploit spectrum holes through cognitive communication, giving priority to the primary users (PUs). In unlicensed bands, all users have the same priority. ISM bands are used nowadays by many radio technologies and has started to decrease in efficiency with an increase in interference. Cognitive radio networks can be used to increase efficiency and QoS through intelligent spectrum sharing.

Wireless sensor motes are mostly equipped with dipole antennas. Compared to channels at higher frequency, those at lower frequency have better propagation characteristics and achieve larger transmission range when they use the same transmitting power.

Existing commercial sensor motes operating on different channel frequencies achieve different transmission ranges and different data rates. For example, sensor motes using lower frequency bands such as 868/900 MHz achieve lower data rate than those at 2.4 GHz band and they have a larger transmission range.

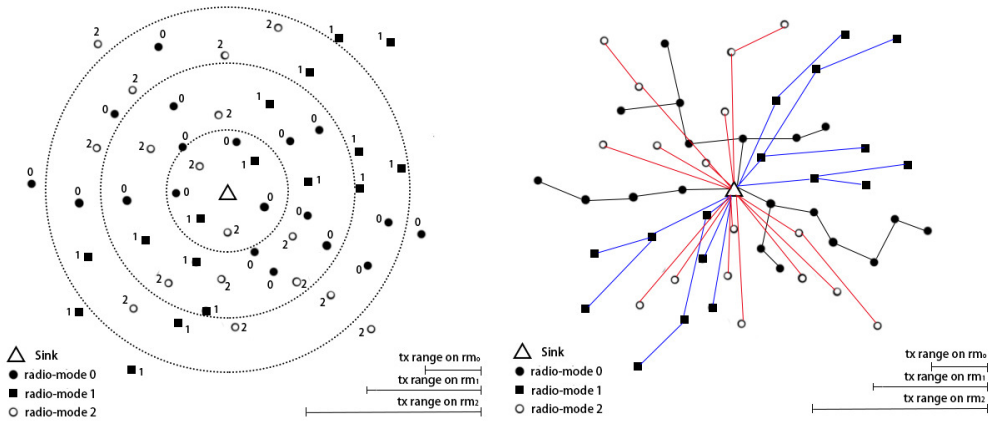
TABLE I
Comparison of Various Wireless Sensor Radio-Modes

SensorMotes	TX Range	Frequency Band	Data Rate(max)	Radio Module
Mica2 [3] 868/916MHz	152m,outdoor	868/916MHz	38.4Kbps	CC1000
Mica2 [3] 433MHz	304m,outdoor	433MHz	38.4Kbps	CC1000
MicaZ [7]	75-100m,outdoor	2.4GHz	250Kbps	CC2420

Table I shows some RF modules of Mica motes. We observe that sensor motes transmitting on lower frequency bands are characterized by a larger transmission range and have a smaller data rate. Using SDR, radios can be configured to different schemes as used in the existing RF modes, according to different needs.

Many approaches proposed for traditional multi-channel WSNs mainly focus on reducing the interference caused by simultaneous transmissions and are mainly exploiting the 802.15 channels. A tree-based multichannel scheme is proposed in [10]. This approach partitions the network into different subtrees communicating on different channels thus eliminating the inter-tree interference. Another tree-based joint channel selection and routing scheme is proposed in [9], which aims to improve the network lifetime by reducing the energy consumed on overhearing.

An application based clustering mechanism is proposed in [5]. Nodes with similar sensed data (e.g. temperature) are assigned to the same channel, forming a data plane. It is assumed that geographical proximity



(a) Radio-mode assignment by the sensor nodes

(b) Multiple tree construction

Fig. 1. RMA: example of radio-mode assignment by the sensors and trees construction.

implies high data correlation. Cluster heads (CHs) in each data plane and the sink communicate though a common control channel while sensor nodes send data to their CH though the assigned intra-cluster channel. The application dependent assumption makes this approach hard to extend for different sensing mechanisms.

Work [2] proposes a distributed algorithm, called RMA (Radio-Mode Assignment) which is used by nodes to assign their radio to different radio-modes, such that the resultant topology is connected to the sink and data throughput at the sink is increased. If shortest-path communication is used, then the resulting topology can be seen as multiple shortest-path trees operating on different frequencies, see Figure 1b.

The mechanism [2] does not address the case when a PU occupies a certain frequency, thus some of the nodes cannot continue to operate on the same radio-mode. In this paper we extend the mechanism [2] to deal with the presence of a PU.

The rest of the paper is organized as follows. Section II presents motivation and the problem that we seek to solve in this paper. Section III contains a brief description of the RMA protocol [2] followed by a detailed description of the way we enhance RMA to deal with the presence of a PU. The resulting distributed algorithm is called PUawareRMA. Section IV presents simulation results and section V concludes the paper.

II. Motivation and Problem Definition

We consider a WSN consisting of n homogeneous sensor nodes $s_1, s_2, ..., s_n$ and a sink node S. We assume that sensor nodes are densely deployed and the WSN is connected. The sink node S is used to collect data and is connected to the network of sensors. Data collection follows a *convergecast* communication model, where data flow from many nodes (e.g. the sensors) to one (the sink). Sensor nodes are resource constrained devices, while the sink is a more resource-powerful device. Each sensor node is equipped with one reconfigurable radio and the sink is equipped with k reconfigurable radios.

We assume that both the sensors and the sink have reconfigurable radios which can adjust the transmission parameters of their radios to radio-modes from the set $C = \{rm_0, rm_1, ..., rm_{m-1}\}$, where $m \geq k$.

Using the RMA [2] distributed algorithm, data is collected by the sink in parallel on k different topologies (or k different trees), where each topology is operating on a different radio-mode. Data throughput received at the sink is increased. Such an example is illustrated in Figure 1b.

We consider that any of the radio-modes used by sensor nodes can be reclaimed by a PU. In the presence of a PU, secondary users (SUs), in our case sensor node, must vacate the spectrum. This may result in network partition and data not being forwarded to the sink. In this paper we assume that at most a PU is present at any time.

The problem that we solve in this paper is topology robustness to the presence of a PU. The contribution of this paper is to extend RMA by adding a mechanism to be used in the presence of a PU. Sensor nodes operating on the same radio-mode as the PU will reassign their radio-mode such that the resulting topology does not interfere with the PU. The objective is that the algorithm has a small overhead and involves a small number of sensor nodes. In this way the topology becomes *robust* to the presence of a PU.

III. Distributed Algorithm for a Topology Robust to the Presence of a PU

Section III-A presents a brief description of the RMA distributed algorithm [2]. In section III-B we enhance

the RMA algorithm to deal with the presence of a PU. We call the new algorithm PUawareRMA.

A. RMA Distributed Algorithm

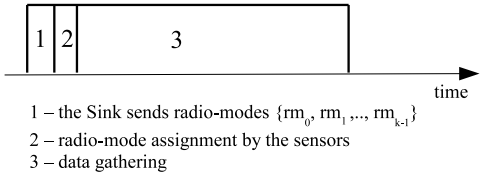


Fig. 2. Network Organization

Network organization has three phases as presented in Figure 2. The sink S is equipped with k radios and chooses k radio-modes from the set C, $C_{sink} = \{rm_0, rm_1, ..., rm_{k-1}\}$, where $C_{sink} \subseteq C$. Let us assume that the radio-modes in C_{sink} are sorted in increasing order of their transmission range, which is the reverse order of the frequency. Thus $rm_0 \in C_{sink}$ is the radio-mode with the smallest transmission range (and the highest frequency). The sink assigns each radio a radio-mode from C_{sink}. All sensor nodes in the network operate on rm_0 during phases 1 and 2.

In the *phase 1*, the sink broadcasts a message containing C_{sink} by flooding. In the *phase 2*, each sensor node selects a radio-mode from C_{sink} such that the overall topology remains connected, see the example in Figure 1a. Each sensor assign its radio to the selected radio-mode at the end of phase 2. *Phase 3* is the data gathering phase. Data is generated by sensor nodes and collected by the sink. The overall topology contains multiple connected topologies using different radio-modes. Any data collection mechanism can be used in this phase. One such example is data collection using shortest-path trees, see Figure 1b.

Phase 2 has two steps:

- Step 1 - neighbor discovery and setting up the distance (hop count) to the sink
- Step 2 - radio-mode selection by the sensor nodes.

Let $N_k(u)$ be the k'th neighborhood of a node u, $N_k(u) = \{v | dist(u, v) \leq k \ hops\}$.

Step 1 is accomplished as follows. Sink S constructs $N_1(S)$ and $N_2(S)$, while each other sensor node v constructs $N_1(v)$ using *Hello* messages. All the nodes broadcast a *Hello* message containing the node ID. The message is sent with a small random delay to avoid collisions. After a short time interval, each node u which is 1-hop away from S sends a second *Hello* message, containing node ID and $N_1(u)$. Based on this information, S computes $N_2(S)$. Each other sensor node v computes $N_1(v)$.

In the second part of this step, the sink S broadcasts a message *Hops* which contains a parameter *hops* - the number of hops to the sink. A sensor node receiving a *Hops* message retransmits the message in two cases: (i) if this is the first *Hops* message received, or (ii) if this message contains a shorter distance to the sink. In both cases the node updates its shortest distance to the sink, increments the *hops* counter, and retransmits the *Hops* message. At the end of this step, each sensor node knows its smallest number of hops to the sink using rm_0.

Step 2 is performed as follows. The number t of topologies connected to the sink S is upper-bounded by $k = |C_{sink}|$ and by the number of nodes in $N_1(S)$, that is $t =$*min*$\{\ k,\ |N_1(S)|\ \}$.

First, the sink S assigns radio-modes to sensor nodes in $N_1(S)$ with the objective of balancing the number of nodes using each radio-mode, in order to distribute the traffic on multiple frequencies. The details of the algorithm are presented in [2]. After all the nodes in $N_1(S)$ have been assigned radio-modes, S broadcasts a message *SinkRMSet*N_1*(S,* $N_1(S)$*, radio-modes assigned to* N_1*(S), TTL* = 1*)*.

Each sensor node v assigns a radio-mode as follows. If $v \in N_1(S)$, then it assigns the radio-mode selected by the sink in the message *SinkRMSet*N_1. Once a sensor selects its radio-mode, it broadcasts a *RMSet* message, informing its neighbors of its radio-mode decision. After broadcasting the *RMSet* message, the node configures its radio to the new radio-mode selected.

All other nodes assign their radio-modes in increasing order of their distance (number of hops) to the sink. A node v sets up a waiting time t_{wait} during which it waits to receive messages from neighbors. t_{wait} is proportional with the distance to the sink v_{hops} - the number of hops to the sink. The waiting time is computed as $Time(v_{hops}) = v_{hops} \times hopDelay$, where $hopDelay$ is the delay per hop and it must account for the propagation delay, algorithm execution time, and the maximum waiting time of a node before sending the *RMSet* message. In this way the nodes at distance 1 will set up their channels first, followed by the nodes at distance 2, then 3, and so on.

When the timer expires, the node takes a decision on selecting a radio-mode among those already advertised by the neighbors. Node v maintains a map with pairs containing node ID and radio-mode selected by neighbors. A threshold value *th* is predefined, e.g. $th = 3$. If the map contains radio-modes assigned to less than th nodes, then the selected radio-mode is the one assigned to the smallest number of nodes. In case of a tie, a radio-mode is selected arbitrarily.

If the map has at least *th* nodes for each radio-mode and the advertised radio-modes are $\{rm_{i1}, rm_{i2}, ..., rm_{ij}\}$, then v computes the probability of selecting the radio-mode rm_{ir} as $p_{ir} = \frac{d_{ir}}{\sum_{x=1}^{j} d_{ix}}$, where $r = 1, 2, .., j$ and d_{ir} is the data rate of the

radio-mode rm_{ir}. The radio-mode rm_i with the highest probability is selected. Node v waits a random time to avoid collisions and transmits a *RMSet* message to inform neighbors of its selection.

At the end of this step, each sensor node v switches its radio to the selected radio-mode. [2] shows that the resultant topology is connected, assuming it is initially connected on rm_0.

Once radio-modes have been selected, data can be collected by the sink using any data collection algorithm. Figure 1b shows shortest-path trees on each radio-mode: rm_0, rm_1, and rm_2.

B. PUawareRMA Distributed Algorithm

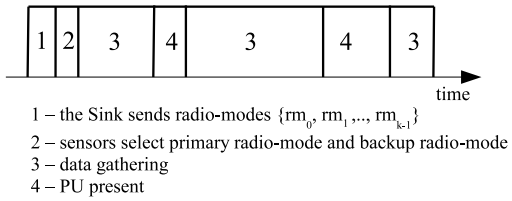


Fig. 3. Network Organization

Network organization has 4 phases, as illustrated in Figure 3. Phases 2 and 4 are different than the RMA algorithm and they will be described in detail.

After $C_{sink} = \{rm_0, rm_1, ..., rm_{k-1}\}$ is broadcasted by the sink on rm_0, all sensors have this information. Recall that in phases 1 and 2 all sensor nodes operate on rm_0. In *phase 2*, sensor nodes select a primary radio-mode (denoted prm) and a backup radio-mode (denoted brm). The primary radio-mode is used when there is no PU affecting it. The backup radio-mode is intended to be used when there is a PU operating on prm.

Phase 2 has the following steps:

- Step 1 - neighbor discovery and setting up the distance (hop count) to the sink
- Step 2 - sink S selects prm and brm for the nodes in $N_1(S)$ and sends this information to $N_1(S)$
- Step 3 - sensor nodes select their prm
- Step 4 - sensor nodes select their brm

Step 1 is similar to the step 1 of the RMA algorithm, see Section III-A.

Next, we describe the **step 2**. The number t of topologies connected to the sink S is t = *min*$\{k, |N_1(S)|\}$. S computes prm and brm for the nodes in $N_1(S)$ using the algorithm *SinkAssignRadioModes*$N_1(S)$ which is presented using pseudocode.

The objective of *ComputePrimaryRadioModes*$N_1(S)$ algorithm is to balance the number of radio-modes used by nodes in $N_1(S)$. For example if $|N_1(S)|$ = 10 and $k = 3$, then $Nrm_0 = 4$, $Nrm_1 = 3$, and $Nrm_2 = 3$. When choosing nodes for rm_i, the objective is to reduce

Algorithm 1 SinkAssignRadioModesN_1(S)

1: ComputePrimaryRadioModes$N_1(S)$
2: ComputeBackupRadioModes$N_1(S)$
3: broadcast *SinkRMSet*N_1(S, $N_1(S)$, primary and backup radio-modes assigned to sensors in $N_1(S)$, TTL = 1)

Algorithm 2 ComputePrimaryRadioModes$N_1(S)$

1: t = min{ k, $|N_1(S)|$}
2: let $C_t = \{rm_0, rm_1, ..., rm_{t-1}\}$
3: **if** $t == |N_1(S)|$ **then**
4: assign nodes in N_1 primary radio-modes $rm_0, rm_1, ..., rm_{t-1}$
5: return
6: **end if**
7: for each $rm_i \in C_t$ compute Nrm_i - number of sensors in $N_1(S)$ with primary radio-mode rm_i, such that $|Nrm_i - Nrm_j| \leq 1$ for any $rm_i, rm_j \in C_t$
8: $U = N_1(S)$ /* sensors with no prm assigned */
9: **for** each radio-mode rm_i, i = 0 to $k-1$ **do**
10: /* assign rm_i to Nrm_i nodes */
11: **for** each node x in U **do**
12: $x.confl = 0$
13: **end for**
14: **for** i = 1 to Nrm_i **do**
15: pick up a node x in U with the smallest $x.confl$ value
16: $x.prm = rm_i$
17: $U = U - \{x\}$
18: **for** each node y in U which is neighbor of x **do**
19: $y.confl = y.confl + 1$
20: **end for**
21: **end for**
22: **end for**

the interference. We select them based on a *conflict* field, denoted $x.confl$. $N_1(S)$ and $N_2(S)$ are used in calculating the conflict. For a sensor x, we denote $x.prm$ - x's primary radio-mode and $x.brm$ - x's backup radio-mode.

The algorithm *ComputeBackupRadioModes*$N_1(S)$ computes a backup radio-mode for each sensor in $N_1(S)$. Lines 3 to 12 address the case when each node in $N_1(S)$ has a distinct primary radio-mode. In this case a node $x \in N_1(S)$ chooses $x.brm$ to be any $y.prm$ such that $y \in N_1(S)$ and $y \notin N_1(x)$. This criterion is used in order to reduce interference between x and y in the event that x switches to its brm. If no such radio-mode exists, then $x.brm$ is assigned any radio-mode in C_t. Note that for any sensor x, $x.prm \neq x.brm$.

Lines 13 to 37 address the case when $k < |N_1(S)|$, thus $t = k$. When computing the backup radio mode the objective is to balance the number of radio-modes used and reduce the conflict (which basically reduces the interference). Let us consider the same example $|N_1(S)|$ = 10 and $k = 3$, with the number of primary radio-modes $Nrm_0 = 4$, $Nrm_1 = 3$, and $Nrm_2 = 3$. Assume that we want to compute the brm of the 4 nodes with $prm = rm_0$. This corresponds the case when a PU on rm_0 is present. To balance the number of users on each radio-mode, 2 nodes will assign $brm = rm_1$ and 2 nodes will assign $brm = rm_2$. When selecting nodes, the one with the smallest conflict is selected in each iteration in

Algorithm 3 ComputeBackupRadioModes$N_1(S)$

```
1:  t = min{ k, |N_1(S)|}
2:  let C_t = {rm_0, rm_1, ..., rm_{t-1}}
3:  if t == |N_1(S)| then
4:      for each node x ∈ N_1(S) do
5:          if there is a node y ∈ N_1(S), y ≠ x, and y ∉ N_1(x) then
6:              x.brm = y.prm
7:          else
8:              x.brm = any radio-mode in C_t − {x.prm}
9:          end if
10:     end for
11:     return
12: end if
13: for each radio-mode rm_i ∈ C_t do
14:     let N'_1(S) = {x ∈ N_1(S) : x.prm = rm_i}
15:     /* compute backup radio-modes for all nodes in N'_1(S) */
16:     let C' = C_t − {rm_i}
17:     compute Nrm_j - the number of sensors in N'_1(S) with backup radio-
        mode rm_j, for each rm_j ∈ C', such that |Nrm_j − Nrm_p| ≤ 1
        for any rm_j, rm_p ∈ C'
18:     U = N'_1(S)    /* sensors with no brm assigned */
19:     for each rm_j ∈ C' do
20:         for each node x ∈ U do
21:             x.confl = 0
22:         end for
23:         for each node x ∈ N'_1(S) do
24:             for each node y ∈ N_1(S) − N'_1(S) such that y.prm = rm_j
                and y ∈ N_1(x) do
25:                 x.confl = x.confl + 1
26:             end for
27:         end for
28:         for i = 1 to Nrm_j do
29:             pick up a node x in U with the smallest x.confl value
30:             x.brm = rm_j
31:             U = U − {x}
32:             for each node y such that y ∈ U and y ∈ N_1(x) do
33:                 y.confl = y.confl + 1
34:             end for
35:         end for
36:     end for
37: end for
```

Algorithm 4 AssignBackupRadioMode(v)

```
1: if v receives a brm assignment in the message SinkRMSetN_1 from the
   sink S then
2:     v.brm = the brm assigned in the message
3:     wait a random time and broadcasts BRMSet(v, v.brm, 0, S, TTL =
       1)
4:     return
5: end if
6: set a waiting time t_wait = Time(v_hops)
7: record brm assigned by neighbor nodes based on BRMSet messages
   received
8: when t_wait expires, v examines brm recorded by neighbors and Neigh-
   borPRMMap and selects a brm (see explanation in the text)
9: wait a random time and broadcasts BRMSet(v, v.brm, v.sdist,
   v.connector, TTL = 1)
```

order to reduce interference.

Step 3, when sensor nodes select their primary radio-mode, is similar to the step 2 in the RMA algorithm, see Section III-A, thus it will not be further discussed here. Each node stores a table (or map) *NeighborPRMMap* with the prm selected by each 1-hop neighbor, and the distance (number of hops on rm_0) of that neighbor to the sink.

Next we describe **step 4**, where sensor nodes select their backup radio-mode. The backup radio-mode of a sensor must be different than its primary radio-mode. The backup radio-mode will be used when the appearance of a PU makes the usage of the primary radio-mode impossible.

The algorithm *AssignBackupRadioMode* is presented using pseudocode.

In lines 1 to 5, if $v \in N_1(S)$, then $v.brm$ is set-up to the value received in the *SinkRMSet*N_1 message.

If $v \notin N_1(S)$, then v sets up a waiting time t_{wait} during which it waits to receive *BRMSet* messages from neighbors. t_{wait} is proportional with the distance to the sink v_{hops} - the number of hops to the sink. The waiting time is computed as $Time(v_{hops}) = v_{hops} \times hopDelay$, where $hopDelay$ is the delay per hop and it must account for the propagation delay, algorithm execution time, and the maximum waiting time of a node before sending the *BRMSet* message. In this way the nodes at distance 1 will set up their brm first, followed by the nodes at distance 2, then 3, and so on.

When the timer expires, v decides its brm and sends a message *BRMSet*(v, $v.brm$, $v.sdist$, $v.connector$, TTL = 1). The field $v.connector$ is the new parent of v and basically will communicate with v on $v.brm$. The field $v.sdist$ indicates the *"switch distance"* - how many ancestors have to switch to their brm in order to avoid network partition.

There are two cases that we distinguish:

- v has at least one neighbor $u \in N_1(v)$ such that $u.prm \neq v.prm$. In this case v selects a neighbor $u \in N_1(v)$ with the smallest distance to S such that $u.prm \neq v.prm$. Then assign $v.brm = u.prm$ and broadcast *BRMSet*(v, $v.brm$, 0, u, TTL = 1).
- Each neighbor u of v has $u.prm = v.prm$. Node v sets-up $v.brm$ to one of the brm of its neighbors. v has information about brm of some of its neighbors, including those which are at a smaller distance to the sink. Node v keeps a *NeighborBRMMap* from all the messages *BRMSet* received. It will store the first four field in the message. Node v selects a node from the *NeighborBRMMap* with the smallest $sdist$. Let us assume that the node u selected has the *NeighborBRMMap* entry: u, $u.brm$, d, w. Then v sets $v.brm = u.brm$ and broadcasts the message *BRMSet*(v, $v.brm$, $d+1$, u, TTL = 1).

At the end of phase 2, each sensor v has computed $v.prm$ and $v.brm$. At the end of this phase v switches its radio-mode to $v.prm$.

Phase 3 is the data gathering phase. Any data collection mechanism can be used. For simplicity, we consider that data is collected along the shortest-path trees, one shortest-path tree for each radio-mode in C_t.

Phase 4 corresponds to the case when a PU on radio-mode rm_i affects an area of the network. We assume data collection using shortest-path trees. We distinguish the following three cases:

• Sensor v with $v.prm = rm_i$ senses the presence of the PU. Node v sends a message $SwitchRM(v, v.prm, v.brm, v.sdist, v.connector, \text{TTL} = 1)$ on $v.prm$ and $v.brm$ radio-modes. Then v switches to $v.brm$ radio-mode during the presence of the PU. Node $v.connector$ becomes the new parent of v.

• Sensor u with $u.prm = rm_i$ is not in PU's transmission range and it receive a message $SwitchRM(v, v.prm, v.brm, v.sdist, u, \text{TTL} = 1)$ from v. If $v.sdist = 0$ then u becomes the new parent of v and u will not switch its radio-mode. If $v.sdist > 0$, then u becomes v's parent and u has to switch in order to ensure v's connectivity. u sends a message $SwitchRM(u, u.prm, u.brm, u.sdist, u.connector, TTL = 1)$, on $u.prm$ and $u.brm$ and then switches to $u.brm$. Node $u.connector$ becomes the new parent of u. In this case $v.prm = u.prm = rm_i$, $v.brm = u.brm$, and $u.sdist = v.sdist - 1$.

• A sensor w with $w.prm = rm_i$ is not in PU's transmission range and it receives the message $SwitchRM(v, v.prm, v.brm, v.sdist, v.connector, \text{TTL} = 1)$ from its current parent v. In this case $v.connector \neq w$. The issues is that if v changes its radio-mode, w cannot continue to send its data to the sink through v on rm_i.

If w has another neighbor x working on rm_i and x is not switching, then x could potentially become the new parent of w. If this option is used, then a mechanism for preventing loop formation must be used.

If w does not have any neighbor on rm_i, then w switches to $w.brm$ after sending $SwitchRM(w, w.prm, w.brm, w.sdist, w.connector, TTL = 1)$ on $w.prm$ and $w.brm$.

Data collection resumes after the nodes have switched their radio-modes. Next we discuss the way in which nodes restore their radio-mode after the PU leaves. We assume that PU activity on a channel is dynamic and it is represented as a continuous ON/OFF random process [4]. For example a TV transmission tower transmits for a certain time period and remains silent for some time before transmitting again.

Data gathering in WSNs can be categorized as time-driven, event-driven or query-driven. For applications that require periodic data monitoring, data gathering is time-driven. In event-driven and query-driven models, sensor nodes react immediately to the detection of a certain event or a query generated by the sink [6]. The overhead and delay caused by channel switching is not negligible and the switching during data gathering period may cause data loss. To alleviate these problems, in time-driven applications the channel restoration is done during the idle period.

Each sensor node in the network keeps a record of its original primary radio-mode and parent node id. When a node v senses the leaving of the PU, it automatically switches from $v.brm$ to $v.prm$ and uses the original parent. Before switching, node v sends a message $RestoreRM(v, v.prm, v.brm, v.sdist, v.connector, \text{TTL} = 1)$ on $v.prm$ and $v.brm$. In this way both the original parent and children that switched because of v are able to restore their original primary radio-mode. If $v.sdist > 0$, then $v.connector$ node switches to its prm, update its parent, and re-sends a ***RestoreRM*** message on its prm and brm radio-modes.

Number of grids	Deployment square side length, m	Total number of sensors
5 x 5 = 25	89.44	75
9 x 9 = 81	161.00	243
13 x 13 = 169	232.55	507
17 x 17 = 289	304.11	867
21 x 21 = 441	375.66	1323
25 x 25 = 625	447.21	1875
29 x 29 = 841	518.77	2523

Fig. 4. WSN deployment parameters.

IV. Simulation

In this section we use ns-3 network simulator [8] to evaluate the performance of our PUawareRMA distributed algorithm.

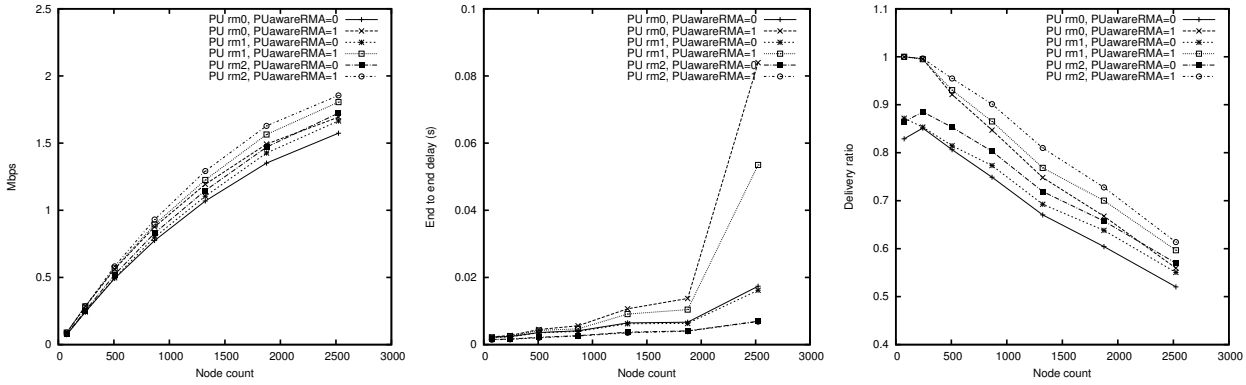
A. Simulation Environment

The current ns-3 release does not provide full support for wireless IEEE 802.15.4 networks. To test our algorithm, we used IEEE 802.11 2.4 GHz band with the following radio-modes:

- *radio-mode 0 (rm_0)* on channel 1, tx range 40m , DSSS rate 11Mbps
- *radio-mode 1 (rm_1)* on channel 6, tx range 101m, DSSS rate 5.5Mbps
- *radio-mode 2 (rm_2)* on channel 11, tx range 151m, DSSS rate 1Mbps.

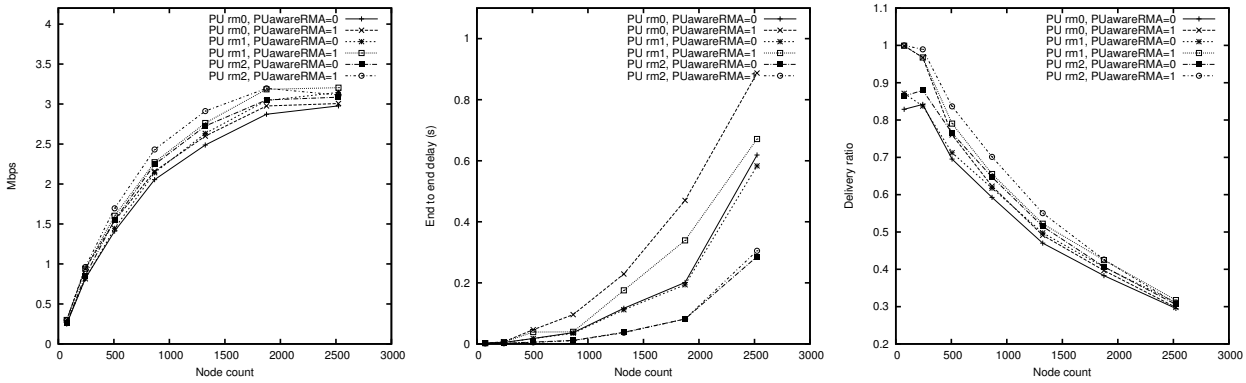
Our algorithm starts from the assumption that the network is connected when all nodes use rm_0, that is the transmission range tx_0. For sensor deployment purpose, we divide the square area into a number of virtual grids with size $tx_0/\sqrt{5}$ and then deploy one sensor randomly in each grid. We deploy the rest of the sensors randomly in the whole square area. The deployment parameters are specified in Figure 4. The sink S is placed in the middle of the deployment area.

Data gathering is performed along the shortest-paths. Each sensor node has a parameter p - the probability the node generates a data message in each interval. We consider two cases: $p = 100\%$ and $p = 30\%$. The size of data messages is 500 bytes.



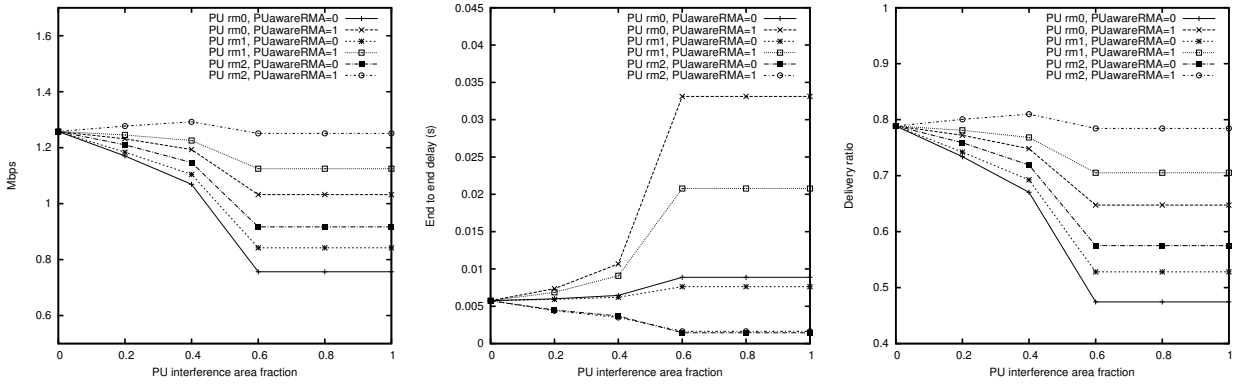
(a) (b) (c)

Fig. 5. Comparisons between RMA and PUawareRMA algorithms when $p = 30\%$. (a)Data throughput. (b)End-to-end delay. (c)Delivery ratio.



(a) (b) (c)

Fig. 6. Comparisons between RMA and PUawareRMA algorithms when $p = 100\%$. (a)Data throughput. (b)End-to-end delay. (c)Delivery ratio.



(a) (b) (c)

Fig. 7. Comparisons between RMA and PUawareRMA algorithms when PU interference area varies. (a)Data throughput. (b)End-to-end delay. (c)Delivery ratio.

We test the behavior of the network in the presence of a PU. The percentage of the area affected by the PU varies between 0 (no PU) to 1 (PU is affecting the whole area). The monitoring area is a square with side L. The area affected by the PU is taken to be the rectangle with height L and width $PU_{fraction} \times L$, starting from the origin. The values that we used in simulations are $PU_{fraction} = 0, 0.2, 0.4, 0.6, 0.8, 1$.

Each simulation scenario is run for 20 seconds and nodes generate data each 1 second interval with probability p. We run each simulation scenario 5 times using different seed values and report the average values in the graphs.

B. Simulation Results

In the first experiment in Figure 5 we compare RMA (PUawareRMA=0) and PUawareRMA (PUawareRMA=1) algorithms when $p = 30\%$ and $PU_{fraction} = 0.4$ (PU affects 40% of the area). We measured the cases when PU is on rm_0, rm_1, and rm_2. Figure 5a shows that PUawareRMA obtains a better throughput (or received data rate) at the sink compared to RMA for each radio mode. The best throughput is obtained when PU is on rm_2, which has the lowest data rate (so the radio-modes with higher data rate are available for data transmission), and the lowest throughput is obtained when PU is on rm_0 which has the highest data rate.

Figure 5b displays the average end-to-end delay of the packets received by the sink. We observe that PUawareRMA algorithm has a higher end-to-end delay than RMA for all radio-modes. This is because when nodes reassign their radio to a different radio-mode, the path length increases compared to the shortest-path. Figure 5c measures the packet delivery ratio. The results are consistent with those in Figure 5a.

In the second experiment in Figure 6 we kept the same settings except $p = 100\%$, that means all sensor nodes send a data packet in each interval. Results are consistent with those in Figure 5. We observe a higher throughput at the sink, since more packets are being generated. At the same time the packet delivery ratio is smaller, since more packets are being dropped due to collisions. The network becomes overloaded and large queues in sensor nodes trigger a larger delay.

In the third experiment in Figure 7 we show the benefit of using PUawareRMA when the area affected by the PU varies between 0 (no PU) and 1 (PU is affecting the whole area), $p = 30\%$, and the number of sensors is 1323. In Figure 7a, PUawareRMA has a better throughput than RMA for each radio mode. The difference between the two increases as the area affected by the PU increases. It should be noted that when $PU_{fraction} \geq 0.6$ the throughput is the same. The reason is that the sink is in the area affected by the PU and it cannot receive data on that specific radio-mode. In the PUawareRMA algorithm, all sensor nodes on the same radio-mode as the sink will switch to a different radio-mode. Similar to previous graphs, PU on rm_2 obtains the highest throughput and PU on rm_0 has the lowest throughput.

In Figure 7b, PUawareRMA has a slightly larger delay as explained previously. The delay increases as the area affected by the PU increases, and it stays the same once the sink is the area affected by the PU. The data delivery ratio results in Figure 7c shows the benefit of using PUawareRMA and they are consistent with the data throughput represented in Figure 7a.

V. Conclusions

This paper presents PUawareRMA, a low-overhead distributed algorithm to be used in the presence of a PU. Sensor nodes affected by the PU switch their primary radio-mode to a backup radio-mode such that the resulting topology remains connected and communication does not interfere with the PU. Network performance is analyzed using ns-3 simulations.

ACKNOWLEDGMENT

This work was supported in part by the NSF grant IIP 0934339.

References

[1] I. Akyildiz, W.-Y. Lee, M. C. Vuran, and S. Mohanty, NeXt generation/dynamic spectrum access/cognitive radio wireless networks: A survey, *Computer Networks* 50, pp. 2127-2159, 2006.

[2] M. Cardei and Y. Wu, Using reconfigurable radios to increase throughput in wireless sensor networks, *IEEE International Conference on Mobile Ad-hoc and Sensor Networks (MSN 2014)*.

[3] Crossbow Technology, *MICA2 wireless measurement system*, Mica2 Datasheet, http://www.eol.ucar.edu/isf/facilities/isa/internal/CrossBow/DataSheets/mica2.pdf, last accessed August 2014.

[4] M. Derakhshani, T. Le-Ngoc, Learning-based opportunistic spectrum access with adaptive hopping transmission strategy, *IEEE Transactions on Wireless Communications*, pp.3957-3967, Nov. 2012.

[5] A. Gupta, C. Gui, and P. Mohapatra, Exploiting multi-channel clustering for power efficiency in sensor networks, *International Conference on Communication System Software and Middleware*, pp. 1-10, 2006.

[6] J.N. Al-Karaki and A.E. Kamal, Routing techniques in wireless sensor networks: a survey, *IEEE Wireless Communications*, pp.6-28, Dec. 2004.

[7] MEMSIC Inc, *MICAz, wireless measurement system*, Micaz Datasheet, http://www.memsic.com/userfiles/files/Datasheets/WSN/micaz_datasheet-t.pdf, last accessed August 2014.

[8] ns-3 network simulator, http://www.nsnam.org, 2014.

[9] A. Pal and A. Nasipuri, DRCS: A Distributed routing and channel selection scheme for multi-channel wireless sensor networks, *PERCOM Workshop*, 2013.

[10] Y. Wu, J. Stankovic, T. He, S. Lin, Realistic and efficient multi-channel communications in dense wireless sensor networks, *IEEE INFOCOM*, 2007.