DRINA: A Lightweight and Reliable Routing Approach for in-Network Aggregation in Wireless Sensor Networks

Leandro Villas[1,2], Azzedine Boukerche[1], Heitor S. Ramos[1,2], Horacio A. B. F. de Oliveira[3], Regina B. de Araujo[4] and Antonio A. F. Loureiro[2]

[1]Paradise Research Laboratory - SITE, University of Ottawa, Ottawa, Canada
[2]UWL - Research Laboratory, Federal University of Minas Gerais, Belo Horizonte, Brazil
[3]LAMP - Research Laboratory, Federal University of Amazonas, Manaus, Brazil
[4]WINDIS - Research Laboratory, Federal University of São Carlos, São Carlos, Brazil
E-mail: (lvillas, hramos, boukerch)@site.uottawa.ca, loureiro@dcc.ufmg.br, regina@dc.ufscar.br and horacio@dcc.ufam.edu.br

Abstract—Large scale dense wireless sensor networks (WSNs) will be increasingly deployed in different classes of applications for accurate monitoring. Due to the high density of nodes in these networks, it is likely that redundant data will be detected by nearby nodes when sensing an event. Since energy conservation is a key issue in WSNs, data fusion and aggregation should be exploited in order to save energy. In this case, redundant data can be aggregated at intermediate nodes reducing the size and number of exchanged messages and, thus, decreasing communication costs and energy consumption. In this work we propose a novel Data Routing for In-Network Aggregation, called DRINA, that has some key aspects such as a reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission. The proposed DRINA algorithm was extensively compared to two other known solutions: the InFRA and SPT algorithms. Our results indicate clearly that the routing tree built by DRINA provides the best aggregation quality when compared to these other algorithms. The obtained results show that our proposed solution outperforms these solutions in different scenarios and in different key aspects required by WSNs.

Index Terms—Routing Protocol, in-Network Aggregation, Wireless Sensor Networks.

1 Introduction

A Wireless Sensor Network (WSN) consists of spatially distributed autonomous devices that cooperatively sense physical or environmental conditions, such as temperature, sound, vibration, pressure, motion or pollutants at different locations [1], [2]. WSNs have been used in applications such as environmental monitoring, homeland security, critical infrastructure systems, communications, manufacturing and many other applications that can be critical to save lives and assets [3], [4], [5].

Sensor nodes are energy-constrained devices and the energy consumption is generally associated with the amount of gathered data, since communication is often the most expensive activity in terms of energy. For that reason, algorithms and protocols designed for WSNs should consider the energy consumption in their conception [6], [7], [8], [9]. Moreover, WSNs are data-driven networks that usually produce a large amount of information that needs to be routed, often in a multi-hop fashion, toward a sink node, which works as a gateway to a monitoring center (Figure 1). Given this scenario, routing plays an important role in the data gathering process.

A possible strategy to optimize the routing task is to use the available processing capacity provided by the intermediate sensor nodes along the routing paths. This is known as data-centric routing or in-network data aggregation. For more efficient and effective data gathering with a minimum use of the limited resources, sensor nodes should be configured to smartly report data by making local decisions [10], [11], [12], [13]. For this, data aggregation is an effective technique for saving energy in WSNs. Due to the inherent redundancy in raw data gathered by the sensor nodes, in-networking aggregation can often be used to decrease the communication cost by eliminating redundancy and forwarding only smaller aggregated information. Since minimal communication leads directly to energy savings, which extends the network lifetime, in-network data aggregation is a key technology to be supported by WSNs. In this work, the terms information fusion and data aggregation are used as synonyms. In this context, the use of information fusion is twofold [14]: (i) to take advantage of data redundancy and increase data accuracy, and (ii) to reduce communication load and save energy.

One of the main challenges in routing algorithms for WSNs is how to guarantee the delivery of the sensed data even in the presence of nodes failures and interruptions in communications. These failures become even more critical when data aggregation is performed along the routing paths since packets with aggregated data contain information from various sources and, whenever one of these packets is lost a considerable amount of information will also be lost. In the context of WSN, data aggregation aware routing protocols should present some desirable characteristics such as: a reduced number of

Digital Object Indentifier 10.1109/TC.2012.31

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.

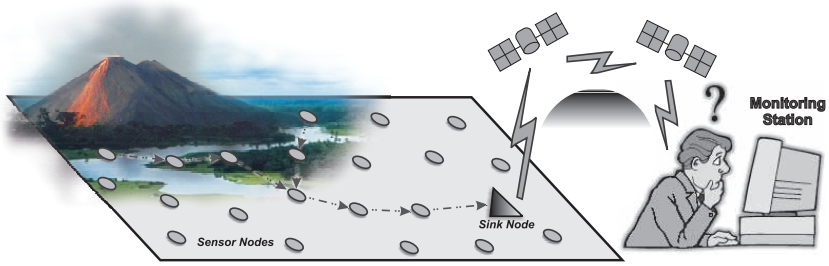


Fig. 1. Data aggregation aware routing, a key algorithm for data-driven WSNs.

messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and also a reliable data transmission. In order to overcome these challenges, in this work we propose a novel Data Routing algorithm for In-Network Aggregation for WSNs, which we refer to as DRINA algorithm. Our proposed algorithm was conceived to maximize information fusion along the communication route in reliable way, through a fault-tolerant routing mechanism. This paper is organized as follows. In Section 2, in-network data aggregation and some related work are discussed. Section 3 introduces our proposed DRINA algorithm. In Section 4, some theoretical bounds of our approach are discussed. Simulation scenarios and experimental results are shown in Section 5. Finally, Section 6 presents our conclusions and future work.

2 In-Network Data Aggregation

In the context of WSNs, in-network data aggregation refers to the different ways intermediate nodes forward data packets toward the sink node while combining the data gathered from different source nodes. A key component for in-network data aggregation is the design of a data aggregation aware routing protocol. Data aggregation requires a forwarding paradigm that is different from the classic routing, which typically involves the shortest path "in relation to some specific metric" to forward data toward the sink node. Differently from the classic approach in data aggregation aware routing algorithms, nodes route packets based on their content and choose the next hop that maximizes the overlap of routes in order to promote in-network data aggregation.

A key aspect of in-network data aggregation is the synchronization of data transmission among the nodes. In these algorithms, a node usually does not send data as soon as it is available since waiting for data from neighboring nodes may lead to better data aggregation opportunities. This in turn, will improve the performance of the algorithm and save energy. Three main timing strategies are found in the literature [15], [16]:

- *Periodic simple aggregation:* requires each node to wait for a pre-defined period of time while aggregating all received data packet and, then, forward a single packet with the result of the aggregation.
- *Periodic per-hop aggregation:* quite similar to the previous approach, but the aggregated data packet is transmitted as soon as the node hears from all of its children. This approach requires each node to know the number of its children. In addition, a timeout may be used for the case of some children's packet being lost.
- *Periodic per-hop adjusted aggregation:* adjusts the transmission time of a node according to this node's position in the gathering tree. Note that the choice of the timing strategy strongly affects the design of the routing protocol as well as its performance.

In-network data aggregation plays an important role in energy constrained WSNs since data correlation is exploited and aggregation is performed at intermediate nodes reducing size and the number of messages exchanged across the network. In data gathering based applications, a considerable number of communication packets can be reduced by in-network aggregation, resulting in a longer network lifetime. In this case, the optimal aggregation problem is equivalent to a Steiner tree problem [17], [18]:

Definition 1 (Steiner Tree): given a network represented by a graph $G = (V, E)$, where $V = \{v_1, v_2, \ldots, v_n\}$ is the set of sensor nodes, E is the set of edges representing the connections among the nodes, i.e., $\langle i, j \rangle \in E$ iff v_i reaches v_j, and $w(e)$ is the cost of edge e, a minimal cost tree is to be built that spans all source nodes $S = \{s_1, s_2, \ldots, s_m\}$, $S \subseteq V$, and the sink node s_0. The cost of the resulting Steiner tree (W) is the sum of the costs of its edges. This problem is a well-known NP-hard problem.

Since it is a NP-hard problem, some heuristics for the Steiner tree problem can be found in the literature. Approximate solutions to the problem are presented in [19] and [20]. However, these solutions are not appropriate for resource-constrained networks, such as WSNs, since their distributed implementation require a large number of messages exchange when setting up the routing tree and, consequently, resulting in high energy consumption.

Thus, various algorithms have been proposed to provide data aggregation during the routing in WSNs. Some of them are *tree-based* algorithms and try to solve some variation of the Steiner tree problem; others are *cluster-based* algorithms while others are simply *structure-less*. In the next sections, these three categories of algorithms and their proposed protocols are briefly discussed.

2.1 Tree-Based Approaches

Protocols in this family [21], [22], [23] are usually based on a hierarchical organization of the nodes in the network. In fact, the simplest way to aggregate data flowing from the sources to the sink node is to elect some special nodes that work

as aggregation points and define a preferred direction to be followed when forwarding data.

In these protocols, a tree structure is constructed first and then used later to either route the gathered data or respond to queries sent by the sink node. Aggregation is performed during the routing when two or more data packets arrive at the same node of the tree. This node then aggregates all received data with its own data and forwards only one packet to its neighbor that is lower in the tree. However, this approach has some drawbacks. For instance, when a packet is lost at a certain level of the tree (e.g., due to channel impairments), data from the whole sub tree will be lost as well. Thus, tree-based approaches require a mechanism for fault tolerance to reliably forward the aggregated data.

Despite the potentially high cost of maintaining a hierarchical structure in dynamic networks and the scarce robustness of the system in case of link/device failures, these approaches are still particularly suitable for designing optimal aggregation functions and performing efficient energy management. For instance, there are some proposed solutions [24] where the sink node organizes routing paths to evenly and optimally distribute the energy consumption while still favoring the aggregation of data at the intermediate nodes.

In most cases, tree-based protocols build a traditional shortest path routing tree. For instance, the Shortest Path Tree (SPT) algorithm [17] uses a very simple strategy to build a routing tree in a distributed fashion. In this approach, every node that detects an event reports its collected information by using a shortest path to the sink node. Information fusion occurs whenever paths overlap (opportunistic information fusion). The Directed Diffusion [25] algorithm is one of the earliest solutions to also propose attribute-based routing. In these cases, data can be opportunistically aggregated when they meet at any intermediate node. Based on Directed Diffusion, the Greedy Incremental Tree (GIT) [26] approach was proposed. The GIT algorithm establishes an energy-efficient path and greedily attaches other sources onto the established path. In the GIT strategy, when the first event is detected, nodes send their information as in the SPT algorithm and, for every new event, the information is routed using the shortest path to the current tree. There is a new aggregation point every time a new branch is created. Some practical issues make GIT not appropriate in WSNs [27]. For example, each node needs to know the shortest path to all nodes in the network. The communication cost to create this infrastructure is $O(n^2)$, where n is the number of nodes. Furthermore, the space needed to store this information at each node is $O(D\,n)$, where D is the number of hops in the shortest path connecting the farthest node $v \in V$ to the sink node (network diameter). After the initial phase the algorithm needs $O(m\,n)$ messages to build the routing tree, where m is the number of source nodes.

A different approach is proposed in the Center at Nearest (CNS) algorithm [17]. In this algorithm, every node that detects an event sends its information to a specific node, called aggregator, by using a shortest path. In this case, the aggregator is the closest node to the sink (in hops) that detects an event.

In [28] and [29], data aggregation is implemented in a real world testbed and the Tiny AGgregation Service (TAG) framework is introduced. TAG uses a shortest path tree, and proposes improvements, such as snooping-based and hypothesis testing based optimizations, dynamic parent switching, and the use of a child cache to estimate data loss. In the TAG algorithm, parents notify their children about the waiting time for gathering all the data before transmitting it so the sleeping schedule of the nodes can be adjusted accordingly. However, like most of the cited tree-based data aggregation aware routing algorithms, the TAG algorithm needs a considerable number of message exchange to construct and maintain the tree.

2.2 Cluster-Based Approaches

Similarly to the tree-based approaches, cluster-based schemes [27], [30], [31] also consist of a hierarchical organization of the network. However, in these approaches, nodes are divided into clusters. Moreover, special nodes, referred to as cluster-heads, are elected to aggregate data locally and forward the result of such aggregation to the sink node.

In the Low-Energy Adaptive Clustering Hierarchy (LEACH) algorithm [30], clustered structures are exploited to perform data aggregation. In this algorithm, cluster-heads can act as aggregation points and they communicate directly to the sink node. In order to evenly distribute energy consumption among all nodes, cluster-heads are randomly elected in each round. LEACH-based algorithms assume that the sink can be reached by any node in only one hop, which limits the size of the network for which such protocols can be used.

The Information Fusion-based Role Assignment (InFRA) algorithm [27] builds a cluster for each event including only those nodes that were able to detect it. Then, cluster-heads merge the data within the cluster and send the result toward the sink node. The InFRA algorithm aims at building the shortest path tree that maximizes information fusion. Thus, once clusters are formed, cluster-heads choose the shortest path to the sink node that also maximizes information fusion by using the aggregated coordinators-distance [27]. A disadvantage of the InFRA algorithm is that for each new event that arises in the network, the information about the event must be flooded throughout the network to inform other nodes about its occurrence and to update the aggregated coordinators-distance. This procedure increases the communication cost of the algorithm and, thus, limits its scalability.

Our proposed DRINA algorithm, presented in detail in Section 3, is also a cluster-based approach. In our algorithm, for each new event, it is performed the clustering of the nodes that detected the same event as well as the election of the cluster-head. After that, routes are created by selecting nodes in the shortest path, to the nearest node that is part of an existing routing infrastructure, where this node will be an aggregation point. Our DRINA routing infrastructure tends to maximize the aggregation points and to use fewer control packets to build the routing tree. Also, differently from the InFRA algorithm, DRINA does not flood a message to the whole network whenever a new event occurs.

2.3 Structure-Less Approaches

Few algorithms for routing aware of data aggregation have been proposed that use a structure-less approach. The Data-Aware Anycast (DAA) algorithm [32], a structure-less data aggregation algorithm, uses anycast to forward packets to one-hop neighbors that have packets for aggregation. It has mechanisms for increasing the chance of packets meeting at the same node (spatial aggregation) and at the same time (temporal aggregation). Since the approach does not guarantee aggregation of all packets, the cost of transmitting packets with no aggregation increases in larger networks.

3 DRINA: Data Routing for In-Network Aggregation for WSNs

The main goal of our proposed DRINA algorithm is to build a routing tree with the shortest paths that connect all source nodes to the sink while maximizing data aggregation. The proposed algorithm considers the following roles in the routing infrastructure creation:

- *Collaborator:* a node that detects an event and reports the gathered data to a coordinator node;
- *Coordinator:* a node that also detects an event and is responsible for gathering all the gathered data sent by collaborator nodes, aggregating them and sending the result toward the sink node;
- *Sink:* a node interested in receiving data from a set of coordinator and collaborator nodes;
- *Relay:* a node that forwards data toward the sink.

The DRINA algorithm can be divided into three phases. In Phase 1, the hop tree from the sensor nodes to the sink node is built. In this phase, the sink node starts building the hop tree that will be used by Coordinators for data forwarding purposes. Phase 2 consists of cluster formation and cluster-head election among the nodes that detected the occurrence of a new event in the network. Finally, Phase 3 is responsible for both setting up a new route for the reliable delivering of packets and updating the hop tree.

3.1 Phase 1: Building the Hop Tree

In this phase, the distance from the sink to each node is computed in hops. This phase is started by the sink node sending, by means of a flooding, the Hop Configuration Message (HCM) to all network nodes. The HCM message contains two fields: `ID` and `HopToTree`, where `ID` is node identifier that started or retransmitted the HCM message and `HopToTree` is the distance, in hops, by which an HCM message has passed.

The `HopToTree` value is started with value 1 at the sink, which forwards it to its neighbors (at the beginning, all nodes set the `HopToTree` as infinity). Each node, upon receiving the message HCM, verifies if the value of `HopToTree` in the HCM message is less than the value of `HopToTree` that it has stored and if the value of `FirstSending` is true, as shown in Algorithm 1 - Line 3. If that condition is true then the node updates the value of the `NextHop` variable with the value of the field `ID` of message HCM, as well as the value of the `HopToTree` variable, and the values in the fields `ID` and `HopToTree` of the HCM message. The node also relays the HCM message, as shown in Algorithm 1 - Line 8. Otherwise, if that condition is false, which means that the node already received the HCM by a shorted distance, then the node discards the received HCM message, as shown in Algorithm 1 - Line 12. The steps described above occur repeatedly until the whole network is configured.

Before the first event takes place, there is no established route and the `HopToTree` variable stores the smallest distance to the sink. On the first event occurrence, `HopToTree` will still be the smallest distance; however, a new route will be established. After the first event, the `HopToTree` stores the smaller of two values: the distance to the sink or the distance to the closest already established route.

Algorithm 1: Hop Tree Configuration Phase

```
1  Node sink sends a broadcast of HCM messageswith the value
   of HopToTree = 1;
   // R_u is the set of nodes that received the message
      HCM
2  foreach u ∈ R_u do
3      if HopToTree(u) > HopToTree(HCM) and
       FirstSending(u) then
4          NextHop_u ← ID_HCM;
5          HopToTree_u ← HopToTree_HCM + 1 ;
           // Node u updates the value of the ID field in
              the message HCM
6          ID_HCM ← ID_u ;
           // Node u updates the value of the HopToTree
              field in the message HCM
7          HopToTree_HCM ← HopToTree_u ;
8          Node u sends a broadcast message of the HCM with
           the new values;
9          FirstSending_u ← false ;
10     end
11     else
12         Node u discards the received message HCM;
13     end
14 end
```

3.2 Cluster Formation

When an event is detected by one or more nodes, the leader election algorithm starts and sensing nodes will be running for leadership (group coordinator); this process is described in Algorithm 2. For this election, all sensing nodes are eligible. If this is the first event, the leader node will be the one that is closest to the sink node. Otherwise, the leader will be the node that is closest to an already established route (Algorithm 2, Lines 7 to 9). In the case of a tie, i.e., two or more concurrent nodes have the same distance in hops to the sink (or to an established route), the node with the smallest `ID` maintains eligibility, as shown in Lines 11 to 13 of Algorithm 2. Another possibility is to use the energy level as a tiebreak criterion.

At the end of the election algorithm only one node in the group will be declared as the leader (Coordinator). The remaining nodes that detected the same event will be the Collaborators. The Coordinator gathers the information collected by

Algorithm 2: Cluster formation and leader election

1 **Input**: S // Set of nodes that detected the event
2 **Output**: u // A node of the set S is elected leader of the group
3 **foreach** $u \in \boldsymbol{S}$ **do**
4 $role_u \leftarrow coordinator$;
// Node u sends message MCC in broadcast
5 Announcement of event detection ;
// $\mathcal{N}_u$ is the set of neighbors of node $u \in \boldsymbol{S}$
6 **foreach** $w \in \mathcal{N}_u$ **do**
7 **if** HopToTree(u) > HopToTree(w) **then**
8 $role_u \leftarrow collaborator$;
9 Node u retransmits the MCC message received from node w ;
10 **end**
11 **else if** HopToTree(u) = HopToTree$(w) \wedge$ ID(u) > ID(w) **then**
12 $role_u \leftarrow collaborator$;
13 Node u retransmits the MCC message received from node w;
14 **end**
15 **else**
16 Node u discards the MCC message received from w;
17 **end**
18 **end**
19 **end**

the Collaborators and sends them to the sink. A key advantage of this algorithm is that all of the information gathered by the nodes sensing the same event will be aggregated at a single node (the Coordinator), which is more efficient than other aggregation mechanisms (e.g., opportunistic aggregation).

3.3 Routing Formation and Hop Tree Updates

The elected group leader, as described in Algorithm 2, starts establishing the new route for the event dissemination. This process is described in Algorithm 3, (Lines 2 to 10). For that, the Coordinator sends a route establishment message to its `NextHop` node. When the `NextHop` node receives a route establishment message, it re-transmits the message to its `NextHop` and starts the hop tree updating process. These steps are repeated until either the sink is reached or a node that is part of an already established route is found. The routes are created by choosing the best neighbor at each hop. The choices for the best neighbor are twofold: (i) when the first event occurs, the node that leads to the shortest path to the sink is chosen (Figure 2(a)); and (ii) after the occurrence of subsequent events, the best neighbor is the one that leads to the closest node that is already part of an established route (Figure 2(c)). This process tends to increase the aggregation points, ensuring that they occur as close as possible to the events.

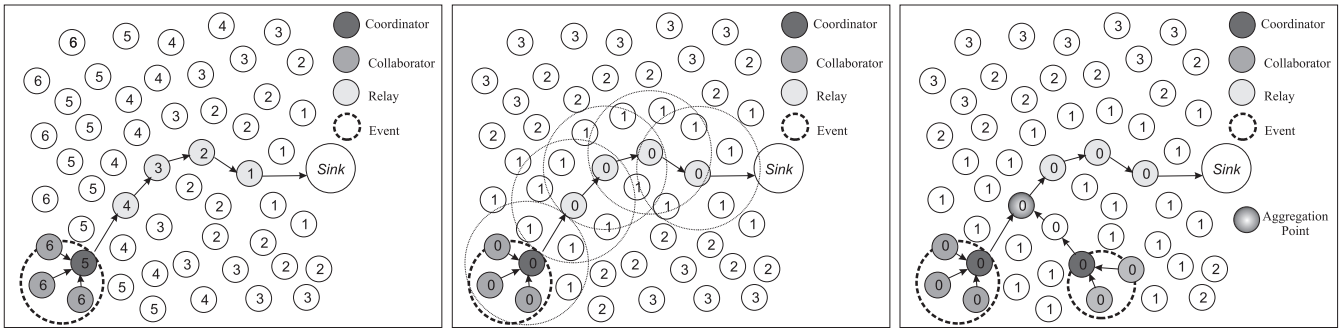
The resulting route is a tree that connects the Coordinator nodes to the sink. When the route is established, the hop tree updating phase is started. The main goal of this phase is to update the `HopToTree` value of all nodes so they can take into consideration the newly established route. This is done by the new relay nodes that are part of an established route. These nodes send an HCM message (by means of a controlled flooding) for the hop updating (Figure 2(b)). The whole cost of this process is the same of a flooding, i.e., each node will send only one packet. This algorithm for the hop updating follows the same principles of the hop tree building algorithm, described in Section 3.1.

Algorithm 3: Route establishment and hop tree update

1 Leader node v of the new event sends a message REM to its $NextHop_v$;
2 **repeat**
// u is the node that received the REM message, that was sent by node v
3 **if** $u = Nextop_v$ **then**
4 $HopToTree_u \leftarrow 0$;
// Node u is part of the new route built
5 $Role_u \leftarrow Relay$;
6 Node u sends the message REM to its $NextHop_u$;
7 Node u broadcasts the message HCM with the value of HopToTree = 1;
8 Nodes that receive the HCM message sent by node u, will run the command Line 2 until the Line 14 of Algorithm 1;
9 **end**
10 **until** *Find out the sink node or a node belonging to the routing structure already established.*;
11 **repeat**
// $sons_u$ is the number of descendants of u
12 **if** $sons_u > 1$ **then**
13 Aggregates all data and sends it to the $nexthop_u$;
14 **if** $Role_u = Relay$ **then**
15 Execute the mechanism of Section 3.4
16 **end**
17 **end**
18 **else**
19 Send data to $nexthop_u$;
20 **if** $Role_u = Relay$ **then**
21 Execute the mechanism of Section 3.4
22 **end**
23 **end**
24 **until** *The node has data to transmit/retransmit*;

The data transmission performed by DRINA uses aggregation techniques that are applied in three different contexts: 1) cluster inside opportunistic aggregation; 2) leader inside aggregation; and 3) cluster outside aggregation. When the routes overlap inside the cluster, the aggregation is performed by the collaborator nodes (cluster inside opportunistic aggregation). Furthermore, the leader node performs data aggregation and sends the results to the sink node (leader inside aggregation). Outside the cluster, aggregation is performed by the relay nodes when two or more events overlap along routing (cluster outside aggregation).

The process of data transmission is described in Algorithm 3 (Lines 11 to 24). While the node has data to transmit, it verifies whether it has more than one descendant that relays its data (Line 12 of Algorithm 3). If it is the case, it waits for a period of time and aggregates all data received and sends the aggregated data to its `NextHop` (Line 13 of Algorithm 3). Otherwise, it forwards the data to its `NextHop`. For every packet transmission with aggregated data, the Route Repair Mechanism is executed as shown in Line 15 of Algorithm 3.



(a) Example of the routing tree to 1 event (b) Update of the hop tree (c) Example of the routing tree to event

Fig. 2. Example of establishing new routes and updating the hop tree

A route repair mechanism is used to send information in a reliable way. Sender nodes wait a pre-defined time period to receive a packet delivery confirmation. When the confirmation is not received by the sender node, a new destination node is selected and the message is retransmitted by that node. This route repair mechanism (Line 15 of Algorithm 3) is described in Section 3.4.

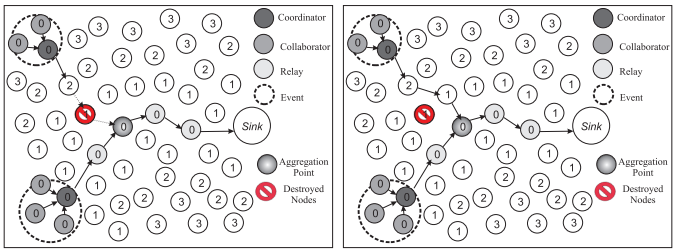
3.4 Route Repair Mechanism

The route created to send the data toward the sink node is unique and efficient since it maximizes the points of aggregation and, consequently, the information fusion. However, because this route is unique, any failure in one of its nodes will cause disruption, preventing the delivery of several gathered event data. Possible causes of failure include low energy, physical destruction, and communication blockage. Some fault tolerant algorithms for WSNs have been proposed in the literature. Some are based on periodic flooding mechanisms [33], [34], and rooted at the sink, to repair broken paths and to discover new routes to forward traffic around faulty nodes. This mechanism is not satisfactory in terms of energy saving because it wastes a lot of energy with repairing messages. Furthermore, during the network flooding period, these algorithms are unable to route data around failed nodes, causing data losses.

Our DRINA algorithm offers a piggybacked, ACK-based, route repair mechanism, which consists of two parts: failure detection at the `NextHop` node, and selection of a new `NextHop`.

When a relay node needs to forward data to its `NextHop` node, it simply sends the data packet, sets a timeout, and waits for the re-transmission of the data packet by its `NextHop`. This re-transmission is also considered an ACK message. If the sender receives its ACK from the `NextHop` node, it can infer that the `NextHop` node is alive and, for now, everything is ok. However, if the sender node does not receive the ACK from the `NextHop` node within the pre-determined timeout, it considers this node as offline and another one should be selected as the new `NextHop` node. For this, the sender chooses the neighbor with the lowest hop-to-tree level to be its new `NextHop`; in case of a tie, it chooses the neighbor with the highest energy level. After that, the sender updates its routing table to facilitate the forwarding of subsequent packets.

As an example, a disrupted route is shown in Figure 3(a). After the repairing mechanism is applied, a newly partial reconstructed path is created as depicted in Figure 3(b).



(a) Region with destroyed nodes (b) Repaired path

Fig. 3. Example of path repair

4 Complexity Analysis

In this section we derive the communication cost bounds for DRINA, InFRA, and SPT algorithms. More specifically, we present the limits for the communication cost of these algorithms to create the routing structure. We also present the best and worst cases, while the average case will be shown in the simulation results (see Section 5).

In the SPT algorithm, there is constant communication cost to build the routing infrastructure. In a reactive fashion operation, it is necessary one flooding started by the nodes that sensed the first event in order to build the routing tree. One more flooding, initiated by the sink node, is also necessary for the other nodes to set up their ancestors in the tree infrastructure. Hence, the constant communication cost for SPT is $2n$, where n is number of nodes.

The InFRA algorithm also presents constant communication costs since it needs one flooding for every elected cluster head, followed by one flooding initiated by the sink node. Each flooding is necessary to set up and update the aggregated coordinators-distance at each node. Also, it is necessary m transmissions to create the cluster, where m is number of transmissions to create the clusters. For this reason, InFRA presents a constant communication cost of $(2kn + m)$, where n is number of nodes, k is number of events. The overhead of the InFRA algorithm can be reduced in some situations by

forcing a delay before each announcement of the cluster-heads is sent by the sink node. Thus, if successive events take place almost simultaneously, only one flooding starting at the sink will be necessary and the communication cost will be lower.

Our proposed DRINA algorithm needs one flooding from the first elected cluster-head and one more flooding initiated by the sink to create the initial tree infrastructure. Successive cluster-heads will make a scope-limited flooding to update the nodes `HopToTree` parameter. Thus, the best case scenario for the communication cost is when successive events take place near the previously established routing tree. The closer the event takes place, the lower the communication cost is; thus, the best case will be achieved when the events happen on the routing tree. In this case, each CH is already attached to the tree. The number of transmissions to establish the initial routing tree is $2n$ plus m transmissions to create the cluster, i.e., the cost is $(2n + m)$. The worst case of the DRINA algorithm happens when successive events take place far from the previously created tree. In this case, the number of transmissions to build the initial tree is $2n$ plus $(k-1)n - \sum_{i=2}^{k} |U_i|$ transmissions for the following events plus m messages to create the cluster, where n is number of nodes, k is number of events, m is number of transmissions to create the clusters and $|U_i|$ is the cardinality of the set of nodes outside the scope-limited flooding for the event i, which will not update their `HopToTree` for this event. Thus, the worst case for the DRINA algorithm is $(2n + ((k-1)n - \sum_{i=2}^{k} |U_i|) + m)$.

Table 1 presents the communication cost of the algorithms assessed in this work.

TABLE 1
Communication complexity of assessed algorithms

Algorithm	Best Case	Worst Case
SPT	$2n$	$2n$
InFRA	$(2n + (k-1)n + m)$	$(2n + (k-1)n + m)$
DRINA	$(2n + m)$	$(2n + ((k-1)n - \sum_{i=2}^{k} \|U_i\|) + m)$

Table 1 shows that DRINA requires more control messages compared to the SPT. However, SPT builds routing trees that are worse than the trees built by our DRINA algorithm, therefore this cost will be recovered by the higher quality of the created tree as we will show in the next section.

5 Performance Evaluation

In this section, we evaluate the proposed DRINA algorithm and compare its performance to two other known routing protocols: the InFRA and SPT algorithms. These two algorithms were chosen for being well known in the literature and have the same goals that the proposed DRINA algorithm. Table 2 shows the basic characteristics of SPT, InFRA and DRINA algorithms. We evaluate the DRINA performance under the following metrics: (i) packet delivery rate; (ii) control overhead; (iii) efficiency (packets per processed data); (iv) routing tree cost; (v) loss of raw data; (vi) loss of aggregated data; and (vii) transmissions number.

5.1 Methodology

The performance evaluation is achieved through simulations using the SinalGo version v.0.75.3 network simulator [35]. In all results, curves represent average values, while error bars represent confidence intervals for 95% of confidence from 33 different instances (seeds). The default simulation parameters are presented in Table 3. For each simulation set, a parameter shown in Table 3 will be varied as described in the evaluated scenario. The first event starts at time $1000\,s$ and all other events start at a uniformly distributed random time between the interval $[1000, 3000]$ seconds. Also, these events occur at random positions. The network density is considered as the relation $n\pi r_c^2/A$, where n is number of nodes, r_c is the communication radius, and A is the area of the sensor field. For each simulation in which the number of nodes is varied, the sensor field dimension is adjusted accordingly in order to maintain the node density at the same value. Sensor nodes are uniformly and randomly deployed.

TABLE 3
Simulation parameters

Parameter	Value
Sink node	1 (top left)
Network size	1024
Communication radius (m)	80
# of events	3
Event radius (m)	50
Event duration ($hours$)	3
Loss probability ($\%$)	0
Simulation duration ($hours$)	4
Notification interval (sec)	60
Sensor field (m^2)	700×700
Node density ($node/m^2$)	21.7

To provide a lower bound to the packet transmissions, an aggregation function was used that receives p data packets and sends only a fixed size merged packet. However, any other aggregation function can be used to take advantage of DRINA features. This function is performed at the aggregation points whenever these nodes send a packet.

The evaluated algorithms used periodic simple aggregation strategy [5] in which the aggregator nodes transmit periodically the received and aggregated information.

The following metrics were used for the performance evaluation:

- *Data packet delivery rate*: number of packets that reach the sink node. This metric indicates the quality of the routing tree built by the algorithms – the lower the packet delivery rate, the greater the aggregation rate of the built tree;
- *Control packet overhead*: number of control messages used to build the routing tree including the overhead to both create the clusters and set up all the routing parameters for each algorithm;
- *Efficiency*: packets per processed data. It is the rate

TABLE 2
Summary of the basic characteristics of assessed algorithms.

Scheme	Route Structure	Objective	Aggregation Nodes	Overhead	Scalability	Drawback
SPT	Tree	Shortest-Path	Opportunistic	Low	High	Data redundancy and static routes
InFRA	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Very High	Low	Low scalability and high cost
DRINA	Tree-based cluster	Maximize overlap routes	Clusterheads and intermediate nodes	Medium	Medium	Static routes

between the total packets transmitted (data and control packets) and the number of data received by the sink;
- *Routing tree cost*: total number of edges in the routing tree structure built by the algorithm;
- *Loss of aggregated data*: number of aggregated data packets lost during the routing. In this metric, if a packet contains X aggregated packets and if this packet is lost, it is accounted the loss of X packets.
- *Number of transmissions*: sum of control overhead and data transmissions, i.e., the total packets transmitted;
- *Number of Steiner nodes*: number of Steiner nodes in the routing structure, i.e., the number of relay nodes;

5.2 Number of Steiner nodes

Since the ideal aggregation is achieved when the information is routed through the minimal Steiner tree [17], in this section we evaluate the number of Steiner tree nodes (i.e., relay nodes) obtained after the construction of the routing tree structure for each of the evaluated algorithms. In this analysis, the network size is varied from 256 to 2048 sensor nodes; the density is varied from 20 to 30; and the number of events were also varied from 1 to 6.

The obtained results are presented in Figure 4. We can clearly see that the number of Steiner nodes in the routing tree built by DRINA algorithm is lower than the ones obtained by the SPT and InFRA algorithms in all studied scenarios. When compared to the MST algorithm, DRINA algorithm results in a slightly larger number of Steiner nodes. However, as mentioned before, the MST algorithm is not suitable for WSNs due to its high overhead and large amount of required memory. We are only comparing our proposed approach to the MST algorithm because the number of Steiner nodes in the routing tree built by the MST algorithm is proved to be at most twice the optimum (i.e., minimum) Steiner tree [36].

The good results obtained by the DRINA algorithm are due to its characteristic of prioritizing nodes that are closer to already existing routes. The InFRA algorithm, on the other hand, prioritizes the distance to the sink node, resulting in lower and/or later aggregations, which increases the number of Steiner nodes.

The Tables 4, 5, 6, and 7 show a different view of the results presented in Figure 4. We can see that in the average case, DRINA is very close to MST, of which, as already mentioned, cost is at most twice the optimal Steiner tree. These results also show that the proposed DRINA algorithm is scalable. For instance, Table 4 shows that, on average, the routing structure built by the InFRA algorithm has 35% more Steiner nodes than DRINA, while Table 6 shows that when increasing the number of nodes to 2048 this difference increases to 42%.

Finally, we can see that in all evaluated scenarios, the minimum routing structures created by the DRINA algorithm have fewer Steiner nodes than the minimum routing structures created by SPT, InFRA, and even the MST algorithm.

TABLE 4
Scenario with 6 events, 256 nodes and density 20

Algorithm	Min	1^{st} Quart.	Median	Mean	3^{rd} Quart.	Max
DRINA	14	18	19	19.30	21	24
INFRA	21	25	26	26.03	28	30
MST	19	20	20	20.39	21	24
SPT	24	28	30	29.79	32	35

TABLE 5
Scenario with 6 events, 256 nodes and density 30

Algorithm	Min	1^{st} Quart.	Median	Mean	3^{rd} Quart.	Max
DRINA	12	14	14	14.45	15	17
INFRA	16	19	20	19.67	20	23
MST	13	15	16	15.94	17	18
SPT	20	22	23	23.61	25	29

TABLE 6
Scenario with 6 events, 2048 nodes and density 20

Algorithm	Min	1^{st} Quart.	Median	Mean	3^{rd} Quart.	Max
DRINA	59	66	68	67.58	70	74
INFRA	87	93	96	96.21	100	104
MST	60	61	62	62.55	64	65
SPT	90	98	104	102.90	106	116

TABLE 7
Scenario with 6 events, 2048 nodes and density 30

Algorithm	Min	1^{st} Quart.	Median	Mean	3^{rd} Quart.	Max
DRINA	44	50	52	51.85	54	58
INFRA	67	73	76	75.48	79	83
MST	47	48	48	48.27	49	50
SPT	72	78	82	81.33	85	92

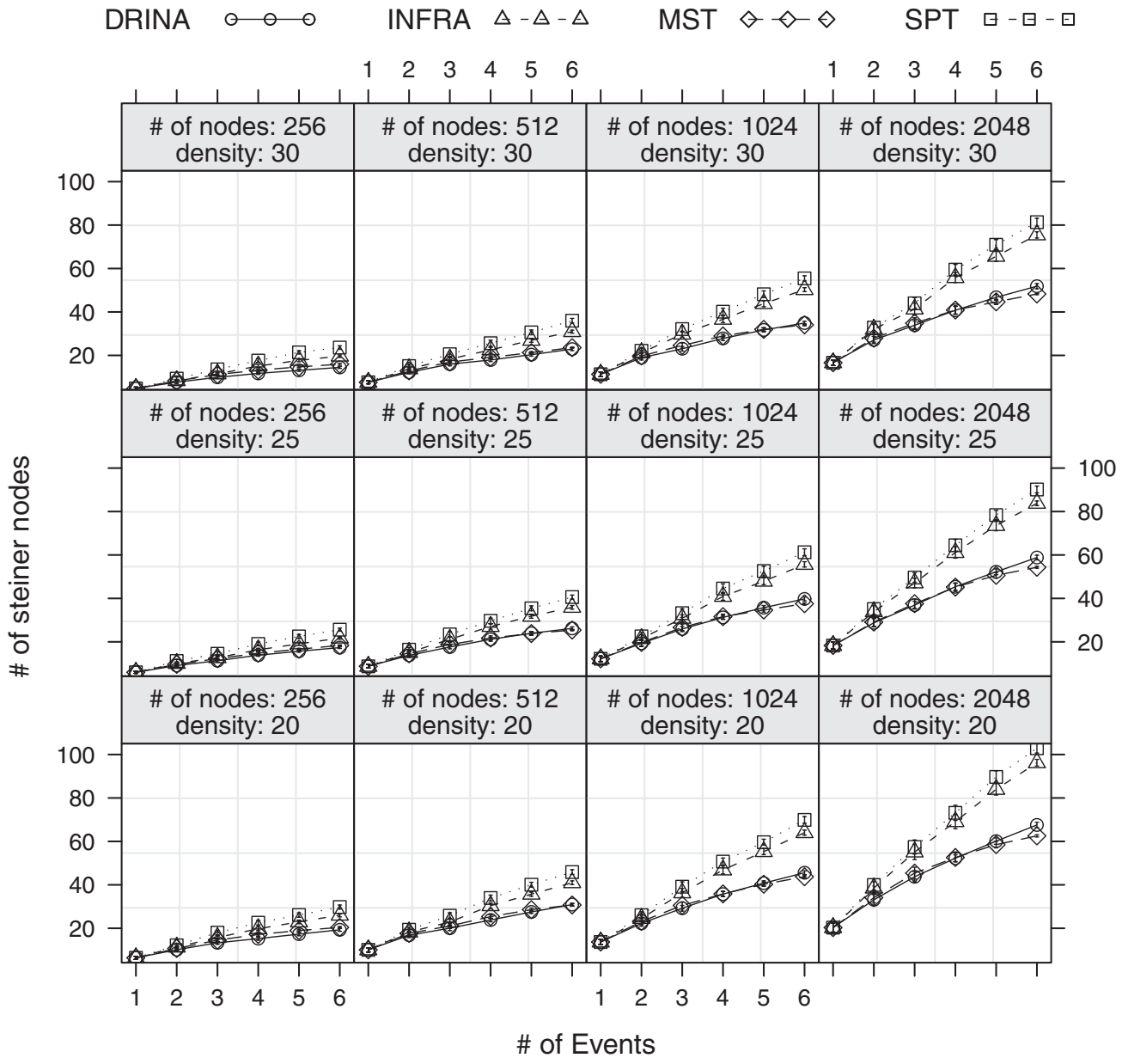
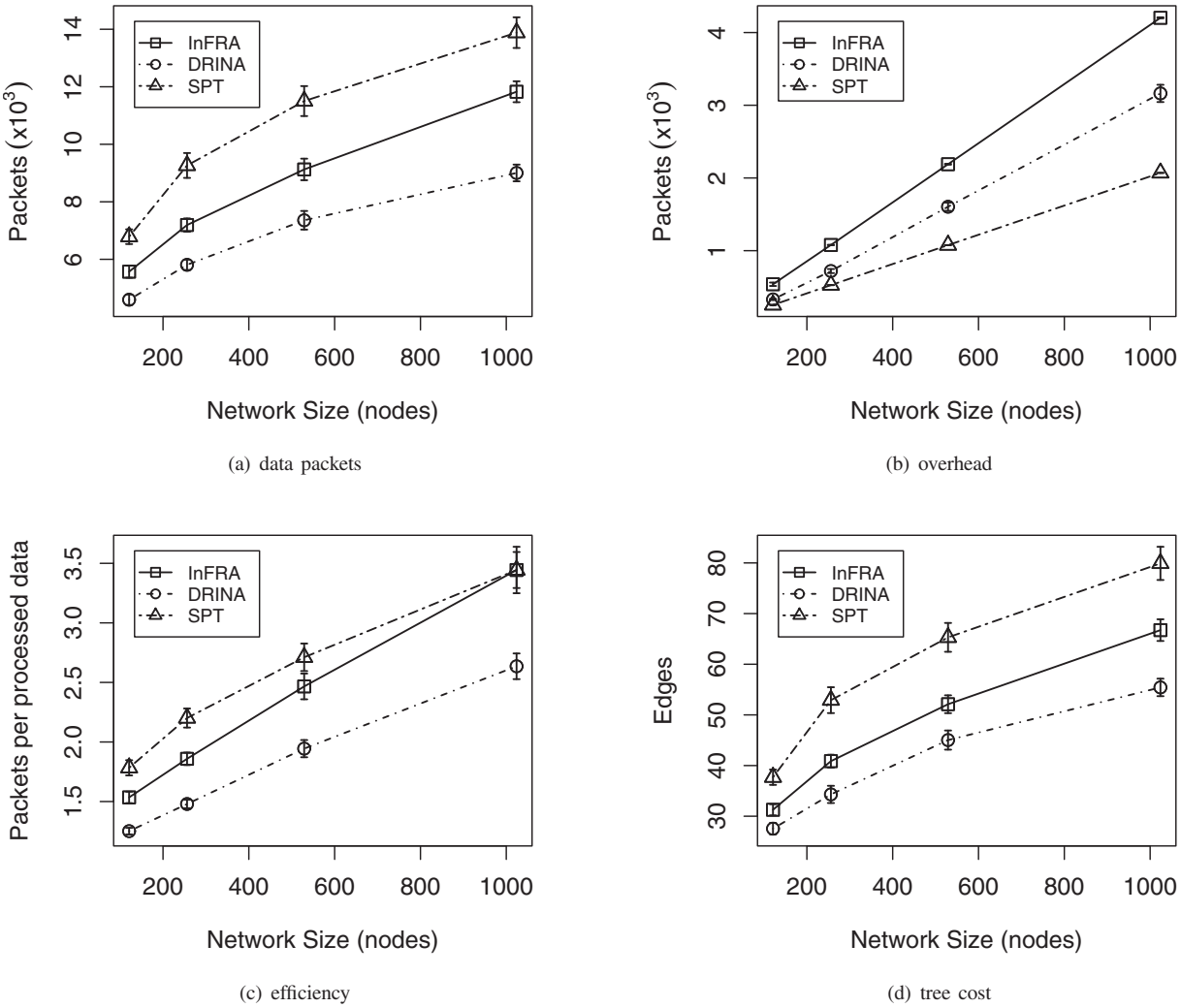


Fig. 4. Number of Steiner nodes in the routing tree built by the DRINA, InFRA, MST, and SPT algorithms.

5.3 Impact of the Network Size

In this simulation scenario, the network size was varied from 128 to 1024 to evaluate the algorithms' scalability. Figure 5 presents the results. Since we are not evaluating the number of Steiner nodes, the MST algorithm is not included in the results. In Figure 5(a) we can see that our DRINA algorithm sends only 77% of the data packets sent by InFRA and about 65% of the data packets sent by SPT. This result clearly indicates that DRINA maintains the quality of the routing tree even when the number of nodes increases. Furthermore, Figure 5(b) shows that DRINA is more scalable than the InFRA algorithm since our algorithm needs 30% less control messages to build the routing structure. On the other hand, the DRINA algorithm requires, on average, 25% more control messages than the SPT algorithm. However, the routing trees built by SPT results in 30% less efficiency than the trees built by DRINA algorithm, as depicted in Figure 5(d). At last, Figure 5(c) shows that DRINA is 20% and 28% more efficient than the InFRA and SPT algorithms, respectively. This occurs because DRINA algorithm needs less control messages to build the routing tree when compared to InFRA. Also, the routing tree built by DRINA has a better data aggregation quality than InFRA and



(a) data packets

(b) overhead

(c) efficiency

(d) tree cost

Fig. 5. Network Size

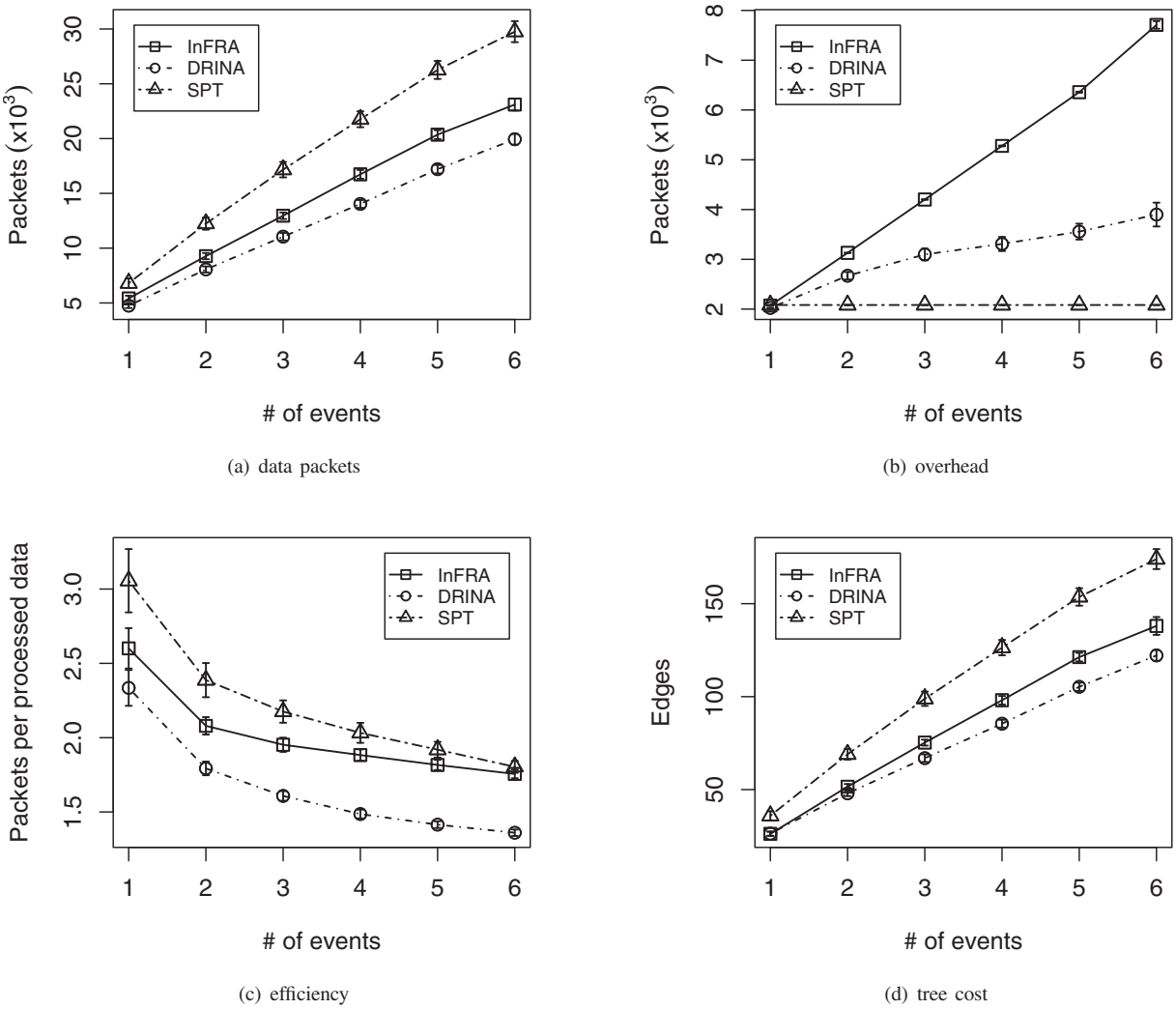
SPT, as shown in Figure 5(d).

5.4 Impact of the Number of Events

In this simulation scenario, the number of events was varied to evaluate the behavior of the proposed algorithms in networks with $1, 2, 3, 4, 5,$ and 6 events occurring simultaneously. The results are presented in Figure 6. As depicted in Figure 6(a), DRINA sends less data packets than the InFRA and SPT algorithms. For instance, DRINA sends approximately 81% and 67% of the data packets sent by InFRA and SPT, respectively. This result indicates one of the main advantages of our DRINA algorithm: by varying the number of events, DRINA builds routing trees more likely to have higher data aggregation rates. Also, Figure 6(b) shows that DRINA needs only 50% of the control messages used by InFRA in the occurrence of 6 events and, on average, only 29% of the control messages used by InFRA to build the routing structure. Thus, for more than one event, DRINA is more efficient than SPT and InFRA, as shown in Figure 6(c). Finally, the cost of the routing tree built by DRINA is 10% smaller than in the InFRA algorithm, and 30% smaller than in the SPT, as we can see in Figure 6(d).

5.5 Impact of the Event Duration

In this simulation scenario, the event duration was varied from 1 to 5 hours. The results are presented in Figure 7. As we can see in Figure 7(a), our proposed DRINA algorithm sends less data packets than the other evaluated algorithms. More specifically, DRINA sends approximately 84% and 64% of the data packets sent by InFRA, and SPT respectively. This indicates that by varying the time of an event duration, DRINA obtains a data aggregation rate greater than InFRA and SPT. Also, Figure 7(b) shows that DRINA requires less control messages to create the routing structure than InFRA but it requires more control messages than the SPT algorithm. Although DRINA requires 33% more control messages than SPT, SPT does not build a good data aggregation routing



(a) data packets

(b) overhead

(c) efficiency

(d) tree cost

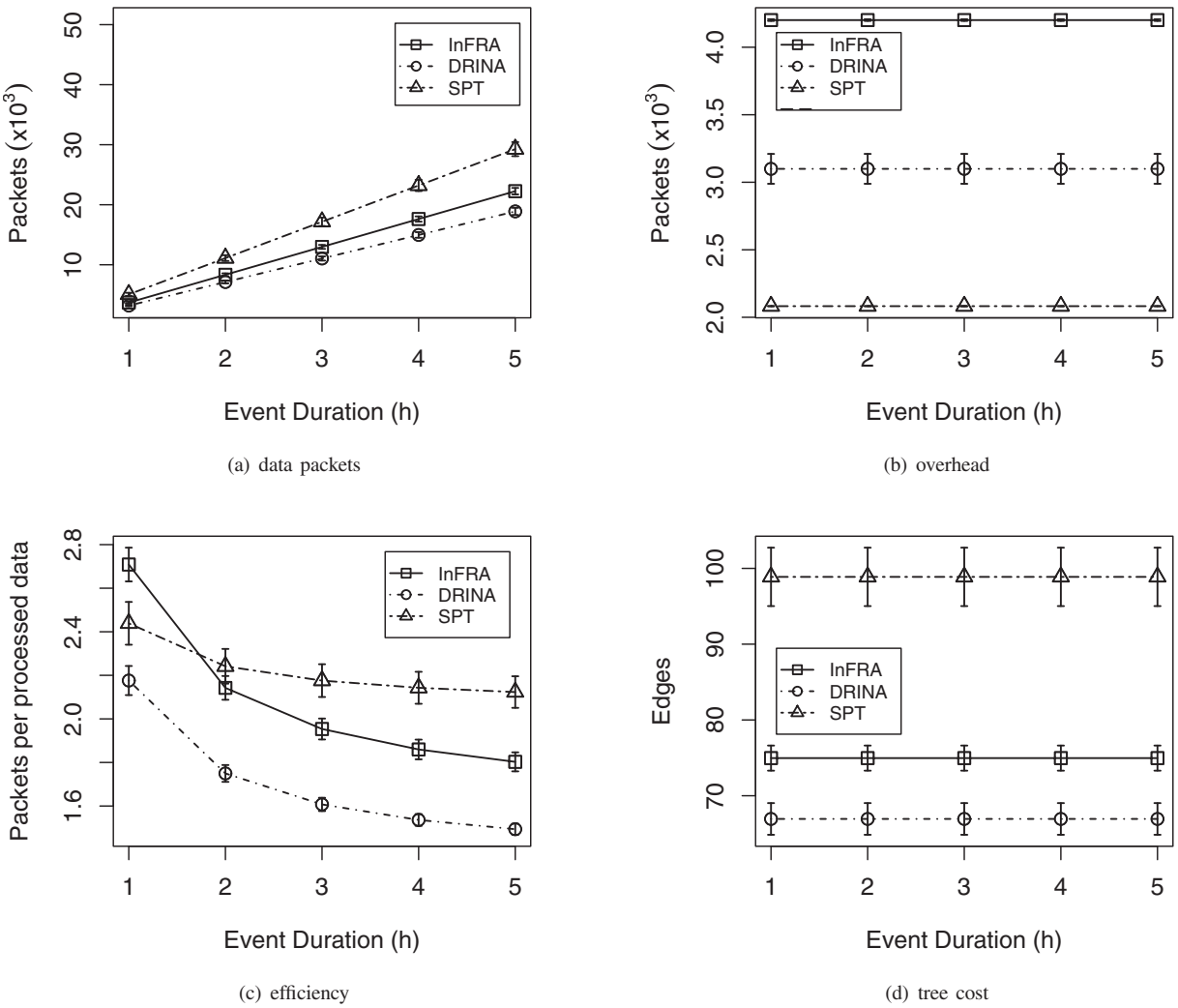
Fig. 6. Number of Events

infrastructure, as shown in previous results. At last, Figure 7(c) shows that DRINA is more efficient than InFRA and SPT. Our proposed algorithm outperforms the other evaluated algorithms even in scenarios of short-term events while InFRA exceeds the SPT only in scenarios where the event duration is longer (typically more than 2 hours).

5.6 Impact of Communication Failures

In this scenario, we evaluate the reliability of our proposed DRINA algorithm. For this, the communication failure probability parameter was varied from 0% to 20%. The results are presented in Figure 8. This simulation also aims to evaluate the cost of DRINA path repair mechanism. As we can see in Figure 8(a), in the DRINA algorithm, data packet transmission increases when the probability of communication failure increases. This is due to the fact that lost packets with aggregated data are retransmitted. On the other hand, SPT and InFRA protocols send less data packets when the communication failures probability increases. This happens because when a packet is lost due to communication failures the packets are not retransmitted and do not reach the sink, as shown in Figure 8(b). In this last figure, we can also see that in a scenario with 20% communication failure, the delivery rate of InFRA is only 30%, while DRINA delivers all aggregated data that have been sent. In summary, DRINA delivers aggregated data reliably with the best performance when compared to SPT and InFRA, as shown in Figure 8(c).

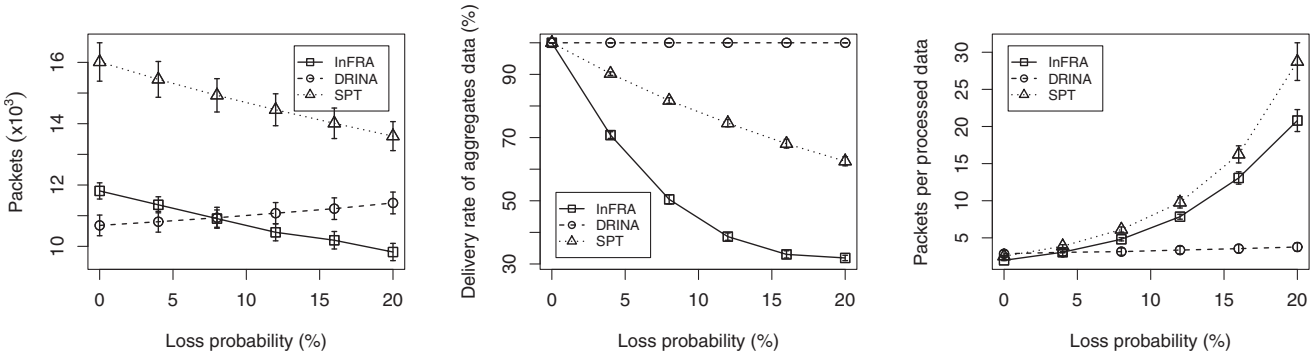
The results presented in this section clearly show that our proposed DRINA algorithm is more scalable than InFRA and SPT in all considered scenarios in terms of network size, events number, event duration time and communication failure probability. In the Notification Rate, DRINA is more efficient when the notification rate is greater than 0.15 notifications per seconds.



(a) data packets (b) overhead

(c) efficiency (d) tree cost

Fig. 7. Event Duration



(a) data packets (b) Delivery rate of aggregated data packets (c) Efficiency

Fig. 8. Communication Failures

6 Conclusion and Future Work

Aggregation aware routing algorithms play an important role in event based WSNs. In this work we presented the DRINA

algorithm, a novel and reliable Data Aggregation Aware Routing Protocol for WSNs. Our proposed DRINA algorithm was extensively compared to two other known routing algorithms, the InFRA and SPT, regarding scalability, communication costs, delivery efficiency, aggregation rate and aggregated data delivery rate. By maximizing the aggregation points and offering a fault tolerant mechanism to improve delivery rate, the obtained results clearly show that DRINA outperformed the InFRA and SPT algorithms for all evaluated scenarios. Also, we show that our proposed algorithm has some key aspects required by WSNs aggregation aware routing algorithms such as a reduced number of messages for setting up a routing tree, maximized number of overlapping routes, high aggregation rate, and reliable data aggregation and transmission.

As future work, spatial and temporal correlation of the aggregated data will also be taken into consideration as well as the construction of a routing tree that meets application needs. We also plan to modify DRINA algorithm to stochastically select nodes that will be part of the communication structure. The goal is to find a balance between the overhead and the quality of the routing tree. In addition, new strategies will be devised to control the waiting time for aggregator nodes based on two criteria: average distance of the event coordinators, and spatial and semantics event correlation.

ACKNOWLEDGMENT

This work was partially supported by the Canada Research Chairs program, NSERC Strategic Project Program and NSERC DIVA Network Research Program, and ORF/MRI Program. The authors also wish to thank FAPEMIG, CNPq, FAPESP, CAPES (INCT-SEC Project, processes 573963/2008-8 and 08/57870-9).

REFERENCES

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cyirci, "Wireless sensor networks: A survey," *Computer Networks*, vol. 38, no. 4, pp. 393–422, March 2002.

[2] K. Romer and F. Mattern, "The design space of wireless sensor networks," *IEEE Wireless Communications*, vol. 11, no. 6, pp. 54–61, December 2004.

[3] G. Anastasi, M. Conti, M. Francesco, and A. Passarella, "Energy conservation in wireless sensor networks: A survey," *Ad Hoc Networks*, vol. 7, no. 3, pp. 537–568, May 2009. [Online]. Available: http://dx.doi.org/10.1016/j.adhoc.2008.06.003

[4] A. Boukerche, R. B. Araujo, and L. Villas, "Optimal route selection for highly dynamic wireless sensor and actor networks environment," in *MSWiM '07: Proceedings of the 10th ACM Symposium on Modeling, analysis, and simulation of wireless and mobile systems*. New York, NY, USA: ACM, 2007, pp. 21–27.

[5] O. Younis, M. Krunz, and S. Ramasubramanina, "Node clustering in wireless sensor networks: Recent developments and deployment challenges," *IEEE Network*, vol. 20, no. 3, pp. 20–25, December 2006.

[6] S. Olariu, Q. Xu, and A. Zomaya, "An energy-efficient self-organization protocol for wireless sensor networks," in *Intelligent Sensors, Sensor Networks and Information Processing Conference (ISSNIP)*. Melbourne, Australia: IEEE, December 2004, pp. 55–60.

[7] H. S. AbdelSalam and S. Olariu, "A lightweight skeleton construction algorithm for self-organizing sensor networks." in *ICC*. IEEE, 2009, pp. 1–5. [Online]. Available: http://dblp.uni-trier.de/db/conf/icc/icc2009.html#AbdelSalamO09

[8] L. Villas, A. Boukerche, R. B. de Araujo, and A. A. F. Loureiro, "Highly dynamic routing protocol for data aggregation in sensor networks," in *Proceedings of the The IEEE symposium on Computers and Communications*, ser. ISCC '10. Washington, DC, USA: IEEE Computer Society, 2010, pp. 496–502. [Online]. Available: http://dx.doi.org/10.1109/ISCC.2010.5546580

[9] L. A. Villas, A. Boukerche, H. A. de Oliveira, R. B. de Araujo, and A. A. Loureiro, "A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks," *Ad Hoc Networks*, no. 0, pp. –, 2011. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1570870511001892

[10] I. Chatzigiannakis, T. Dimitriou, S. E. Nikoletseas, and P. G. Spirakis, "A probabilistic algorithm for efficient and robust data propagation in wireless sensor networks," *Ad Hoc Networks*, vol. 4, no. 5, pp. 621–635, 2006.

[11] I. Chatzigiannakis, S. Nikoletseas, and P. G. Spirakis, "Efficient and robust protocols for local detection and propagation in smart dust networks," *Mob. Netw. Appl.*, vol. 10, no. 1-2, pp. 133–149, 2005.

[12] C. Efthymiou, S. Nikoletseas, and J. Rolim, "Energy balanced data propagation in wireless sensor networks," *Wirel. Netw.*, vol. 12, no. 6, pp. 691–707, 2006.

[13] L. A. Villas, D. L. Guidoni, R. B. Araújo, A. Boukerche, and A. A. Loureiro, "A scalable and dynamic data aggregation aware routing protocol for wireless sensor networks," in *Proceedings of the 13th ACM international conference on Modeling, analysis, and simulation of wireless and mobile systems*, ser. MSWIM '10. New York, NY, USA: ACM, 2010, pp. 110–117. [Online]. Available: http://doi.acm.org/10.1145/1868521.1868540

[14] E. F. Nakamura, A. A. F. Loureiro, and A. C. Frery, "Information fusion for wireless sensor networks: Methods, models, and classifications," *ACM Computing Surveys*, vol. 39, no. 3, pp. 9–1/9–55, 2007.

[15] F. Hu, X. Cao, and C. May, "Optimized scheduling for data aggregation in wireless sensor networks," in *ITCC '05: Proceedings of the International Conference on Information Technology: Coding and Computing (ITCC'05) - Volume II*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 557–561.

[16] I. Solis and K. Obraczka, "The impact of timing in data aggregation for sensor networks," in *Communications, 2004 IEEE International Conference on*, vol. 6, June 2004, pp. 3640–3645 Vol.6.

[17] B. Krishnamachari, D. Estrin, and S. B. Wicker, "The impact of data aggregation in wireless sensor networks," in *ICDCSW '02: Proceedings of the 22nd International Conference on Distributed Computing Systems*. Washington, DC, USA: IEEE Computer Society, 2002, pp. 575–578.

[18] J. Al-Karaki, R. Ul-Mustafa, and A. Kamal, "Data aggregation in wireless sensor networks - exact and approximate algorithms," in *High Performance Switching and Routing, 2004. HPSR. 2004 Workshop on*, 2004, pp. 241–245.

[19] S. Hougardy and H. J. Prömel, "A 1.598 approximation algorithm for the steiner problem in graphs," in *SODA '99: Proceedings of the 10th annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 1999, pp. 448–453.

[20] G. Robins and A. Zelikovsky, "Improved steiner tree approximation in graphs," in *SODA '00: Proceedings of the 11th annual ACM-SIAM symposium on Discrete algorithms*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics, 2000, pp. 770–779.

[21] A. Boukerche, B. Turgut, N. Aydin, M. Z. Ahmad, L. Bölöni, and D. Turgut, "Survey paper: Routing protocols in ad hoc networks: A survey," *Comput. Netw.*, vol. 55, pp. 3032–3080, September 2011. [Online]. Available: http://dx.doi.org/10.1016/j.comnet.2011.05.010

[22] J. Al-Karaki and A. Kamal, "Routing techniques in wireless sensor networks: a survey," *Wireless Communications, IEEE*, vol. 11, no. 6, pp. 6–28, Dec. 2004.

[23] E. Fasolo, M. Rossi, J. Widmer, and M. Zorzi, "In-network aggregation techniques for wireless sensor networks: a survey," *Wireless Communications, IEEE*, vol. 14, no. 2, pp. 70–87, April 2007.

[24] A. Boukerche, *Algorithms and Protocols for Wireless Sensor Networks*. Wiley-IEEE Press, 2008.

[25] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking," *IEEE/ACM Trans. Netw.*, vol. 11, no. 1, pp. 2–16, 2003.

[26] C. Intanagonwiwat, D. Estrin, R. Govindan, and J. Heidemann, "Impact of network density on data aggregation in wireless sensor networks," in *Distributed Computing Systems, 2002. Proceedings. 22nd International Conference on*, 2002, pp. 457–458.

[27] E. F. Nakamura, H. A. B. F. de Oliveira, L. F. Pontello, and A. A. F. Loureiro, "On demand role assignment for event-detection in sensor networks," in *ISCC '06: Proceedings of the 11th IEEE Symposium on Computers and Communications*. Washington, DC, USA: IEEE Computer Society, 2006, pp. 941–947.

[28] S. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tag: a tiny aggregation service for ad-hoc sensor networks," *SIGOPS Oper. Syst. Rev.*, vol. 36, no. SI, pp. 131–146, 2002.

[29] S. Madden, R. Szewczyk, M. J. Franklin, and D. Culler, "Supporting aggregate queries over ad-hoc wireless sensor networks," in *WMCSA '02: Proceedings of the Fourth IEEE Workshop on Mobile Computing Systems and Applications*. Washington, DC, USA: IEEE Computer Society, 2002, p. 49.

[30] A. P. Chandrakasan, A. C. Smith, and W. B. Heinzelman, "An application-specific protocol architecture for wireless microsensor networks," *IEEE Transactions on Wireless Communications*, vol. 1, pp. 660–670, 2002.

[31] L. A. Villas, A. Boukerche, R. B. Araujo, and A. A. Loureiro, "A reliable and data aggregation aware routing protocol for wireless sensor networks," in *Proceedings of the 12th ACM international conference on Modeling, analysis and simulation of wireless and mobile systems*, ser. MSWiM '09. New York, NY, USA: ACM, 2009, pp. 245–252. [Online]. Available: http://doi.acm.org/10.1145/1641804.1641846

[32] K.-W. Fan, S. Liu, and P. Sinha, "On the potential of structure-free data aggregation in sensor networks," in *INFOCOM 2006. 25th IEEE International Conference on Computer Communications. Proceedings*, April 2006, pp. 1–12.

[33] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*. New York, NY, USA: ACM, 2000, pp. 56–67.

[34] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. Culler, and K. Pister, "System architecture directions for networked sensors," *SIGPLAN Not.*, vol. 35, no. 11, pp. 93–104, 2000.

[35] Sinalgo, "Simulator for network algorithms," 2008, Distributed Computing Group - ETH-Zurich, last visited in October, 2008. [Online]. Available: http://dcg.ethz.ch/projects/sinalgo

[36] M. A. Takahashi, H., "An approximate solution for the steiner problem in graphs," *Math. Jap.*, pp. 573–577, 1980.

BIOGRAPHY

Azzedine Boukerche is a full professor and holds a Canada Research Chair position at the University of Ottawa (uOttawa). He is a fellow of the Canadian Academy of Engineering and the founding director of the PARADISE Research Laboratory, School of Information Technology and Engineering (SITE), Ottawa. Prior to this, he held a faculty position at the University of North Texas, and he was a senior scientist at the Simulation Sciences Division, Metron Corp., San Diego. He was also employed as a faculty member in the School of Computer Science, McGill University, and taught at the Polytechnic of Montreal. He spent a year at the JPL/NASA-California Institute of Technology, where he contributed to a project centered about the specification and verification of the software used to control interplanetary spacecraft operated by JPL/NASA Laboratory. His current research interests include wireless ad hoc and sensor networks, wireless networks, mobile and pervasive computing, wireless multimedia, QoS service provisioning, performance evaluation and modeling of large-scale distributed systems, distributed computing, large-scale distributed interactive simulation, and parallel discrete-event simulation. He has published several research papers in these areas. He served as a guest editor for the Journal of Parallel and Distributed Computing (special issue for routing for mobile ad hoc, special issue for wireless communication and mobile computing, and special issue for mobile ad hoc networking and computing), ACM/Kluwer Wireless Networks, ACM/Kluwer Mobile Networks Applications, and Journal of Wireless Communication and Mobile Computing. He serves as an Associate Editor of IEEE Transactions on Parallel and Distributed systems, IEEE Transactions on Vehicular Technology, Elsevier Ad Hoc Networks, Wiley International Journal of Wireless Communication and Mobile Computing, Wiley's Security and Communication Network Journal, Elsevier Pervasive and Mobile Computing Journal, IEEE Wireless Communication Magazine, Elsevier's Journal of Parallel and Distributed Computing, and SCS Transactions on Simulation. He was the recipient of the Best Research Paper Award at IEEE/ACM PADS 1997, ACM MobiWac 2006, ICC 2008, ICC 2009 and IWCMC 2009, Globecom 2012, and the recipient of the Third National Award for Telecommunication Software in 1999 for his work on a distributed security systems on mobile phone operations. He has been nominated for the Best Paper Award at the IEEE/ACM PADS 1999 and ACM MSWiM 2001. He is a recipient of an Ontario Early Research Excellence Award (previously known as Premier of Ontario Research Excellence Award), Ontario Distinguished Researcher Award, and Glinski Research Excellence Award. He is a cofounder of the QShine International Conference on Quality of Service for Wireless/Wired Heterogeneous Networks (QShine 2004). He served as the general chair for the Eighth ACM/IEEE Symposium on Modeling, Analysis and Simulation of Wireless and Mobile Systems, and the Ninth ACM/IEEE Symposium on Distributed Simulation and Real-Time Application (DS-RT), the program chair for the ACM Workshop on QoS and Security for Wireless and Mobile Networks, ACM/IFIPS Europar 2002 Conference, IEEE/SCS Annual Simulation Symposium (ANNS 2002), ACM WWW 2002, IEEE MWCN 2002, IEEE/ACM MASCOTS 2002, IEEE Wireless Local Networks WLN 03-04; IEEE WMAN 04-05, and ACM MSWiM 98-99, and a TPC member of numerous IEEE and ACM sponsored conferences. He served as the vice general chair for the Third IEEE Distributed Computing for Sensor Networks (DCOSS) Conference in 2007, as the program cochair for GLOBECOM 2007-2008 Symposium on Wireless Ad Hoc and Sensor Networks, and for the 14th IEEE ISCC 2009 Symposium on Computer and Commmunication Symposium, and as the finance chair for ACM Multimedia 2008. He also serves as a Steering Committee chair for the ACM Modeling, Analysis and Simulation for Wireless and Mobile Systems Conference, the ACM Symposium on Performance Evaluation of Wireless Ad Hoc, Sensor, and Ubiquitous Networks, and IEEE/ACM DS-RT.

Antonio Alfredo Ferreira Loureiro received his B.Sc. and M.Sc. degrees in computer science from the Federal University of Minas Gerais (UFMG), Brazil, and the Ph.D. degree in computer science from the University of British Columbia, Canada. Currently, he is a professor of computer science at UFMG, where he leads the research group in wireless sensor networks. His main research areas are wireless sensor networks, mobile computing, and distributed algorithms. In the last 10 years he has published regularly in international conferences and journals related to those areas, and also presented tutorials at international conferences.

Leandro Aparecido Villas is a Ph.D. student of Computer Science at the Federal University of Minas Gerais, Brazil. He received his Master Degree in Computer Science from the Federal University of Sao Carlos, Brazil, in 2007. His research interests include context interpretation, distributed algorithms, routing algorithms, wireless actor and sensor networks, quality of Service. He has published papers in the area of wireless sensor networks and received the prize for best article of the XXV Brazilian Symposium on Computer Networks and Distributed Systems, SBRC. He was a Visiting PhD Student for the Academic year 2010-2011 at PARADISE Research Laboratory - University of Ottawa, Canada.

Horacio A.B.F. Oliveira is an Associate Professor of Computer Science at Federal University of Amazonas (UFAM), Brazil. He received his Ph.D. in Computer Science from the Federal University of Minas Gerais, Brazil with a partial doctoral fellowship at University of Ottawa, Canada, in 2008. His research interests include localization and synchronization algorithms, distributed algorithms, and wireless ad hoc, vehicular, and sensor networks. He is author of several papers in the different areas of his research interests.

Regina B. Araujo is an Associate Professor at the Computer Science Department of Federal University of Sao Carlos, SP, Brazil. Prof. Araujo research interests are context aware physical environments for emergency preparedness where wireless sensor networks, distributed simulations, context aware computing and collaborative virtual environments areas are integrated for accurate monitoring and visualization of physical environments subject to emergency conditions. Prof. Araujo was a Program Co-Chair for the 1st ACM Workshop on QoS and Security for Wireless Networks (Q2SWinet 2005). In 2005, she was a Visiting Professor at the PARADISE Research Laboratory, University of Ottawa. She serves as the program Co-Chair for the 2nd ACM Intl Workshop on Wireless Multimedia Networking and Performance Modeling.

Heitor S. Ramos received his bacharel degree in Electrical Engineering from the Federal University of Campina Grande (UFCG), Brazil, and his master in Computing Modeling from the Federal University of Alagoas (UFAL), Brazil. His research interests rely on wireless networks, sensors networks, and mobile and ad hoc networks. In 2010, he received the Microsoft Latin American PhD Fellowship award, and spent the 2010's summer as an intern at the Networked Embedded Computing group (NEC) at Microsoft Research, Redmond, USA. He is currently a PhD candidate in Computer Science at the Federal University of Minas Gerais (UFMG), Brazil. He was a Visiting PhD Student for the Academic year 2010-2011 at PARADISE Research Laboratory - University of Ottawa, Canada.