International Journal of Computing and Digital Systems
ISSN (2210-142X)
Int. J. Com. Dig. Sys. 7, No.6 (Nov-2018)

http://dx.doi.org/10.12785/ijcds/070601

Power-Efficient Routing Based on Ant-Colony-Optimization and LMST for In-Network Data Aggregation in Event-Based Wireless Sensor Networks

Nadjib BENAOUDA[1] **and Ammar LAHLOUHI**[1]

[1] *L@STIC Laboratory, Department of Computer Science, University of Batna-2, 05000 Batna, Algeria*

Received 22 Aug. 2018, Revised 30 Sep. 2018, Accepted 21 Oct. 2018, Published 1 Nov. 2018

Abstract: Data routing for in-network data aggregation is an important task in event-based wireless sensor networks. Previous works tackling this routing problem are distributed heuristics to the minimal Steiner tree. They aim to find a multihop routing structures that maximize data aggregation with the minimum number of hops. However, the nodes participating in this routing task don't have to use their maximum transmission powers, but instead, they have the potential to minimize them to conserve more energy. In this article we go further and based on the local minimum spanning tree algorithm and Ant-Colony-Optimization metaheuristic, we propose a novel protocol that aims to define a routing structure which maximizes data aggregation and minimizes the total transmission powers while ensuring a symmetric transmission power assignment to reliably deliver data. The proposal was widely compared to two other known protocols. Simulation results show the superiority of our protocol compared to these protocols.

Keywords: Wireless sensor networks; In-network aggregation; Transmission power adjustment; Local minimum spanning tree; Ant-colony-Optimization.

1. INTRODUCTION

A Wireless Sensor Network (WSN) consists of a set of distributed and autonomous devices, called sensor nodes, that have the ability to measure ambient conditions such as temperature, motion, humidity, pressure, noise levels as well as others [1]. These sensor devices cooperate to sense and route, using hop-by-hop communication, the sensed data towards a sink node which represents a collection point. These networks have a wide range of applications such as habitat monitoring, agriculture, home automation, health care, emergency operations and many other applications[1,2]. Generally, two categories of applications can be distinguished [3]: event-based and continuous data collection. In this work, we are interested in even-based applications where the network remains inactive and reacts when an event of interest occurs by building the routing infrastructure which is necessary to deliver the sensed data. The important task here is how to efficiently route event data to the sink. Given that sensor nodes are energy-constrained devices, it is necessary to conduct this routing task in an energy-efficient manner [4].

In fact, the important part of the sensor node's energy is consumed during communications (i.e. transmission and reception) compared to that which is consumed during computations [1,2,3]. For that reason, it is more rational to favor calculations over communications. Basing on this observation and given the spatial correlation that characterizes the data collected by sensor nodes, data aggregation is among the efficient techniques that are used to optimize communication costs [5]. It aims to aggregate the sensed data and eliminate the redundancy that may exist in order to minimize the size and number of messages exchanged across the network. Three components are essential to conduct this aggregation in an efficient manner [3]: (1) routing schema, which defines how data are routed towards the sink by favoring their spatial convergence, (2) the aggregation schedule, which favors the temporal convergence of data by defining the waiting time that should each node wait before aggregating and forwarding the received data, and (3) the aggregation function, which defines how data are aggregated. This paper focuses on the routing schemas by assuming a simple data aggregation function and scheduling algorithm.

The routing problem in such event-based WSNs can be formulated as a minimal Steiner tree problem. Previous works such as InFRA [6], DRINA [7],

E-mail:nadjib.benaouda@gmail.com, ammar.lahlouhi@gmail.com

DDAARP [8] and DST [9] are distributed heuristics to this problem. They aim to find a multihop routing structures that maximize in-network data aggregation with the minimum number of hops. In fact, the connectivity between the different nodes of this structure is defined by their transmission power levels. These nodes are not obliged to use the same transmission power but, on the other hand, have the potential to decrease their powers in order to minimize the total transmission power. This helps to further minimize the energy consumed by these nodes during communication [24]. In this article we go further and formulate the routing problem as a minimum-power Steiner tree problem. We propose a novel Power-Efficient Routing for In-Network Data Aggregation, called PALDA. It aims to: (1) Establish a routing structure that maximizes the number of overlapping routes, and thus, maximizes data aggregation. This structure corresponds to a Steiner tree, (2) Assign to each sensor node of this structure an appropriate transmission power, (3) Minimize the sum of the assigned transmission powers, and (4) Ensure a symmetric transmission power assignment that considers bidirectional links between nodes of the formed routing structure so as to support protocols with ack messages, and thus, delivering data in a reliable manner [10]. This reliable delivery which is very important given that the loss of packets is intolerable when data aggregation is applied [7].

As the works cited earlier, PALDA performs a role assignment process. First, sensor nodes that detect the same event are grouped into a single cluster. Then, a tree that connects the different cluster-heads is formed. However, unlike these works, within each cluster, a sparse and efficient topology is constructed, in a localized manner and based on the local minimum spanning tree algorithm (LMST), over the subgraph which corresponds to the subnetwork containing only the cluster's members. Then, a shortest weighted path tree, which is rooted at the cluster-head and spanning all members is constructed over the edges of the computed sparse topology. For the formation of routes between the cluster-heads, the Ant-Colony-Optimization (ACO) metaheuristic is used to determine the relay nodes and their transmission powers. Previous ant-based minimum Steiner tree algorithms like ACA [11] and MANSI [12] let ant agents explore the network and move in all directions in order to encounter nodes of the existing routing tree, and thus, ensuring routes overlapping. Unlike these works, an underlying geometric structure is established to direct the search of ants towards the necessary parts of the network instead of letting them seeking in all directions. This helps to accelerate the convergence speed of the formed routing tree. According to the way of establishment of the underlying geometric structure, two variations of PALDA are proposed. The first, PALDA-S (Static), where the routing paths are established according to the order of occurrence of events and used during the entire time period of their occurrence. And the second, PALDA-D (Dynamic), where routes are established according to cluster-heads positions and they are reconstructed every time a new event happens in order to improve the quality of the final routing tree. Simulations are carried out to evaluate the performance of the proposed protocol with its two variations by comparing it to ECMANSI [13], an ant-based minimum-power Steiner tree algorithm, and DST [9]. The results demonstrate that the proposal builds routing structures with lower total transmission power while maximizing aggregation rate. They demonstrate also the positive effect of the underlying geometric structure on the convergence time of the solution.

To illustrate our work in detail, the remained of the paper is organized as follows. The section 2 presents the background knowledge supporting this work. In section 3 the problem treated is formulated. Section 4 briefly discusses the related work. Section 5 details our proposed protocol PALDA. The simulation scenarios and evaluation results are presented in section 6 . This article ends with a conclusion in section 7.

2. NETWORK AND EVENT MODEL

Our network consists of a set of sensor nodes and a single sink node. Sensor nodes can adjust their communication ranges by adjusting their transmission power levels, without exceeding a maximum power P_{max} which corresponds to a maximum range R_{max}. Let $G=\{V,E\}$ be the graph that models the topology of our WSN where $V=\{v_1,v_2,\ldots,v_n\}$ is the set of vertices representing the different nodes of the network such as $|V|=n$ and v_1 is the sink node, And E is the set of arcs (links) that connect these nodes. It is defined as follows:

$$E = \{(u,v) \mid u \in V, v \in V \text{ and } \delta_{u,v} \le R_{\max}\}$$

where

$$\delta_{u,v} = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2}$$

represents the Euclidean distance between the two nodes u and v , with $u=(x_u; y_u)$ and $u=(x_v; y_v)$. Static events are considered and each one is described by a region of influence. A binary detection model is assumed. So, each node that is inside the region of influence of an event detects it. Let $S=\{s_1,s_2,\ldots,s_m\}$ be the set of source nodes, i.e. nodes that detect events, such as $|S|=m$ and $S \subseteq (V - v_1)$. A perfect aggregation procedure is assumed. i.e. k packets of size l are aggregated into a single packet of the same size l. This corresponds to the case where simple aggregators such as min, max, average, etc. are used. In a similar way to [26], the required transmission power for node *i* to communicate with node *j* is calculated by assuming the formula:

$$power_{i,j} = \delta_{i,j}^{\alpha} \qquad \textbf{(1)}$$

where α is the path loss and $\delta_{i,j}$ is the distance between *i* and *j*.

3. PROBLEM STATEMENT

In this work, the goal is to define, once one or several events occur, a routing structure that maximizes data aggregation with the minimum total transmission power of nodes constituting this structure. In fact, this structure is a Steiner tree that connects all source nodes with the sink node and whose sum of transmission powers of source nodes and relay nodes of the resulting tree is minimal.

Problem: given the graph $G=\{V, E\}$, where V is the set of vertices that correspond to nodes of the network and E the set of arcs that correspond to links between these nodes, with a sink node $v_1 \in V$ and a set $S \subseteq (V - v_1)$ of source nodes, find the subgraph $G' = \{V', E'\}$ of G where $V' = S \cup L \cup v_1$ with L represents the set of relay[1] nodes, and a transmission power assignment $p : V' \rightarrow R^{+}$ such as:

1. G' is a tree that is rooted at the sink node and spanning all source nodes in S.
2. $\sum_{v \in G' and v \neq v_1} p(v)$ is minimized, where $p(v) = \max_{u \in N_v} power_{v,u}$ with $N_v = \{u | u \in V', and, e_{v,u} \in E'\}$ the set of neighbours of v in G'.

4. RELATED WORK

In this section, we briefly discuss some well-known approaches related to our work. Principally, data aggregation techniques are considered by focusing on the underlying routing structure used to perform this aggregation, namely, *tree-based* and *cluster-based* approaches. In addition, works that rely on the ant-colony-optimization metaheuristic for the establishment of this routing structure are also discussed. Moreover, a discussion concerning the limitations of the most considered solutions is presented in a separate section.

A. Tree-based approaches

The most commonly adopted strategy when a tree-based approach is followed is the construction of a shortest path tree [7]. Once constructed, each sensor node reports its data using the shortest path to the sink node and the data are aggregated when the paths overlap. The work presented in [14] is a typical example following this principle. Other approaches adopt other strategies. For example, with the CNS algorithm [14], all source nodes report their data to the closest source node to the sink. This central node then aggregates the received data and sends the result to the sink. The GIT algorithm [15] proceeds differently. With this algorithm, the nodes that detect the first event report their data along the shortest path to the sink. Then, for any new detected event, the shortest path that connects the new sources with the rest of the tree is established. Data aggregation is done at the intersection nodes of the various constructed paths.

Besides these approaches, other works conduct a transmission powers adjustment process. For example, with PEDAP [16], a minimum energy cost tree is used during the collection of data. This tree is constructed in a centralized manner using Prim's spanning tree algorithm. In [17], a localized version of PEDAP is proposed. This new algorithm, which is called L-PEDAP, is proposed to overcome the centralized nature of PEDAP. First, L-PEDAP constructs, in a localized manner, a sparse topology over the visibility graph of the network. Then, a power-efficient tree is computed over the edges of the computed sparse topology. L-PEDAP is based on topologies such as LMST [18] and RNG [19] that can approximate minimum spanning tree. They can be efficiently computed based only on information of one-hop neighbours. For example, LMST [18] is computed as follow: at the beginning, each node *i* determines its one-hop neighbours and calculates its local MST. Then, it considers only the neighbours that are one-hop away from it in the MST formed. The resulting topology is a directed graph. Two ways can be applied to convert it into an undirected graph [18]. The first is to consider any arc between *u* and *v* in the new topology only if the arcs $e_{u,v}$ and $e_{v,u}$ are parts, respectively, of local trees of *u* and *v* ($LMST^{-}$). With the second method, a new arc is included in the final topology only if one of the two arcs $e_{u,v}$ and $e_{v,u}$ exists ($LMST^{+}$).

B. Cluster-based approaches

With such approaches, nodes are organized into clusters. For each one, a special node, called cluster-head, is elected to assume the tasks of aggregation and notification of result to the sink. A cluster-based structure can be done in different ways. It can be done in a proactive manner where it is defined in advance and maintained periodically as in WSN-2-LTS [21] and LEACH [20]. This kind of protocols are dedicated to

---

[1] A relay node is any node in the constructed tree that is not a leaf node nor the root node. In our work, this node can be a source node.

applications where data collection occurs continuously. Differently, this clustering can be established in a reactive way in order to save energy during periods of network inactivity [6]. This case is relevant to event-based applications. InFRA [6] is a reactive protocol, with which, nodes that detect the same event are grouped into the same cluster. The cluster-heads aggregate data collected within their clusters and report the results to the sink node. During this reporting, each cluster-head routes its data along the shortest path to the sink while maximizing the fusion of data. DRINA [7] is another reactive protocol that is similar to InFRA. However, DRINA aims to maximize the aggregation points with the use of a fewer control packets during the formation of routes between cluster-heads. With this algorithm, the routes are formed by considering the shortest paths to the nearest nodes in the existing routing structure. These nodes are then considered as aggregation points. DST [9,22] falls into the same category of InFRA and DRINA, however, DST takes advantage of the coordinates of nodes and establishes an underlying geometrical structure in order to establish the different routes between cluster-heads. Compared to InFRA and DRINA, DST uses a fewer control packets and the created routes do not depend on the order of events, but on the other hand, they are re-established with every new event. This helps to ensure a balance in energy consumption.

C. ACO-based approaches

In fact, an ideal aggregation is ensured when data are routed through a minimal Steiner tree [7]. Since this problem is NP-hard, some works that are based on ant-colony-optimization algorithms exist in the literature. The authors in [11] exploit these algorithms to establish a tree that maximizes data aggregation with a minimum number of hops. The main idea of this work is to let ant agents search around the individual routing paths while reinforcing, each time, the paths that lead to the intersection nodes. The authors in [23] proposes a family of four ant colony algorithms called DAACA. Two main objectives are targeted by these algorithms: (1) minimizing the cost of constructing and maintaining the structures used for aggregation, and (2) balancing energy consumption across the entire network to extend the network lifetime. The four algorithms differ mainly in the way in which the pheromone variables are updated. Other approaches that are dedicated to the multicast problem in AD HOC networks can be considered here since this problem is equivalent to the Steiner tree problem. For example, with MANSI [12], each destination node of a multicast session deploys ant agents that explore the network in order to find a path with a minimum number of hops that allows it to connects with the rest of the tree. In this way, a minimal Steiner tree emerges with the continuous and cooperative work of ants. Based on MANSI, the same authors propose ECMANSI [13,25] where the objective is to minimize, instead of number of hops, the sum of the transmission powers of the non-leaf nodes of the formed multicast tree.

D. Discussion

In this section, we briefly discuss some limitations of the related works. The proposed solutions like PEDAP [16] and L-PEDAP [17] build power-efficient trees to conduct the aggregation process. However, they are dedicated to applications where all the nodes participate in data collection. They do not take full benefit of data aggregation when the multiple-source single-sink communication schema is considered given that they do not maximize routes overlapping. The same remark is applicable to the most shortest path tree based approaches. On the other hand, works like CNS [14], GIT [15], InFRA [6], DRINA [7], DST [9] and ACA [11] aim to maximize routes overlapping. However, their main objective is to minimize the number of hops in the built routing structures so as to minimize to the maximum the number of transmissions needed to perform the data collection. Our proposed protocol aims also to maximize routes overlapping. However, unlike the previous works, the aim is to minimize the total transmission power by assigning to each node of the routing structure an appropriate transmission power. This helps to further conserve the energy of nodes. Concerning the ant-based approaches like ACA [11], the family of algorithms DAACA [23], MANSI [12] and ECMANSI [13], our proposed protocol differ from them principally by the adoption of an underlying geometric structure to direct the search of ants instead of letting them seeking in all directions. This helps to accelerate the convergence speed of the formed routing structure. Moreover, compared to these ant-based approaches, two strategies that define the way in which the paths of the source nodes must be merged are proposed. These two strategies have a direct implication on the quality of the solution obtained in terms of the total transmission power of the final routing tree.

5. PALDA: POWER-EFFICIENT ROUTING BASED ON ANT-COLONY-OPTIMIZATION AND LMST FOR IN-NETWORK DATA AGGREGATION

PALDA is a reactive protocol, with which, a role assignment process is triggered once an event is detected. The sensor nodes that detect the same event are organized into the same cluster. Among this set a cluster-head is elected. The latter aggregates the incoming packets from cluster's members and sends the resulting packet to the sink node. The aggregated packet is then routed from one node to another until it reaches the sink node. To perform this process and define the routing structure, our algorithm considers the following roles:

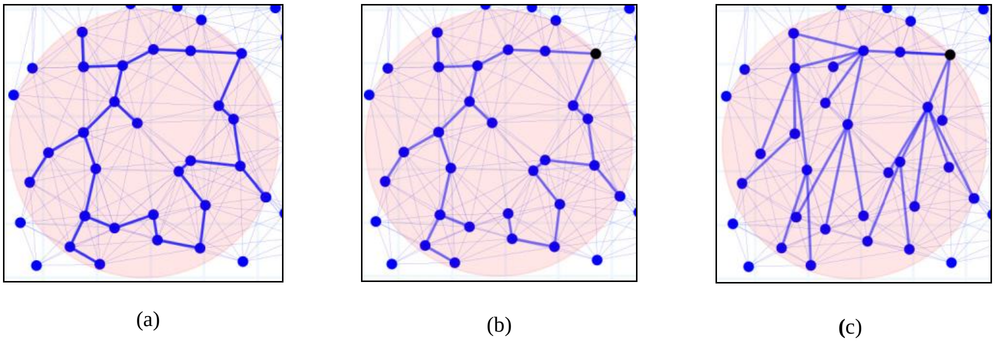

(a) (b) (c)

Figure 1. Two ways of routes set-up within a cluster: the first one, presented in figure (b), where a shortest weighted path tree, which is rooted at the coordinator node (black node) and spanning all collaborators, is computed over the edges of the LMST topology (figure (a)) that is constructed over the subnetwork containing nodes that are within the event region (the pink disk). The second possible way, presented in figure (c), consists in constructing a minimum hop path tree on the original topology.

- **collaborator:** node that detects an event, and thus, represents a cluster-member. It is responsible for reporting the gathered data toward the coordinator node.
- **coordinator:** node that detects an event and represents the cluster-head of all nodes that form the same cluster. It collects the gathered data sent by collaborator nodes, aggregates them and sends the resulting packet toward the sink node.
- **relay:** node that routes the received data from its sons toward the sink node.
- **sink:** node that is interested in the different events that occur.

After an initialization phase, in which, the sink node floods its position to the entire network and each node defines its neighborhood by registering the identifier and the position of each neighbor, the network remains inactive until the occurrence of one or more events. When an event occurs, the following three phases are executed:

1. In Phase 1, the clusters are formed and the coordinator and collaborator roles are assigned to nodes that detect a new event. within each cluster and based on the LMST algorithm, a sparse and efficient topology is constructed. Then, a shortest weighted path tree, which is rooted at the coordinator and spanning all members is constructed over the edges of the computed sparse topology.
2. In Phase 2, each new coordinator is notified of the positions of all coordinators that already exist. These last coordinators are informed also about the position of the new coordinator. These positions allow the coordinators to establish an underlying geometric structure that is exploited by ant agents during routes formation and transmission power assignment.
3. In Phase 3, nodes assuming relay roles are identified and the routing tree that connects the coordinators with the sink node is established. The identification of relay nodes and the establishment of this tree are made following a transmission power assignment process that is based on the ant-colony-optimization metaheuristic.

A. Cluster Formation

When an event occurs, the sensing nodes detecting this event enter in the leader election phase. This process is described in algorithm (1). For this election, all nodes that detect the event are eligible. Different strategies can be applied for the election of coordinator node (e.g. node having the smallest identifier, the one with the highest residual energy, etc). In this work and in a similar way to DST [9], the closest node to the sink is elected as cluster-head (Algorithm (1), Lines 9 and 10). In case of equality, the node with the smallest identifier is chosen (Algorithm (1), Lines 11 and 12). At the end of this phase, a single node is elected as coordinator and the other members assume the collaborator roles.

In order to collect data within the cluster, a tree that is rooted at the coordinator and spanning all collaborators is needed. One possible solution, and as shown in Fig. 1c, consists in establishing a minimum hop path tree over the original topology. However, to decrease the transmission power that is used by each cluster's member we proceed differently. Therefore, we construct, based on the LMST algorithm, within each cluster and as shown in Fig. 1a, a sparse topology over the subgraph which corresponds to the subnetwork containing only the cluster's members. Then and as shown in Fig. 1b a shortest weighted path tree which is rooted at the coordinator node (black node in Fig. 1b) and spanning all collaborators is constructed over the edges of the computed sparse topology.

This process of building the sparse topology and forming the tree within the cluster is done in conjunction

Algorithm 1. cluster formation, leader election and routes set-up inside cluster

```
1: for each $u \in S$ do
2:   $role_u \leftarrow$ coordinator;
3: // Node u announces event detection by broadcasting an Event Announcement Packet (EAP)
4:   Node u broadcasts an EAP;
5:   Node u Collects EAPs for $t_{collect}$;
6: // Node u computes its neighbours on sparse topology considering only its neighbours $w \in S$ ($N_u$),
7:   Node u establishes its LMST and identifies its new neighbours on sparse topology $N_u^{LMST}$;
8:   for each $w \in N_u$ do
9:      if distanceToSink(u) > distanceToSink(w) then
10:       $role_u \leftarrow$ collaborator;
11:     else if distanceToSink(u) = distanceToSink(w) And id(u) > id(w) do
12:       $role_u \leftarrow$ collaborator;
13:     end if
14:  end for
15:  if $role_u$ = coordinator then
16:     CCL.add{id(u), distanceToSink(u)}; //CCL is the list of coordinators candidates
17:     Node u announces coordinator intention to its neighbours $N_u^{LMST}$; //event-scoped flooding
18:  end if
19:  while Coordinator Intention Packet (CIP) received in $t_{discovery}$ do
20:     if update required for CIP then
21:       Node u updates the parent node leading to this candidate coordinator;
22:       Node u updates and broadcasts the received CIP to its neighbours $N_u^{LMST}$;
23:     end if
24:  end while
25: // smallestDTS(CCL) corresponds to the smallest distance among the distances from each
    // candidate coordinator in CCL to the sink
26:  if $role_u$ = coordinator And distanceToSink(u) $\neq$ smallestDTS(CCL) then
27:     $role_u \leftarrow$ collaborator;
28:  end if
29:  if $role_u$ = collaborator then
30:     Node u chooses as parent node the one leading to the closest candidate coordinator to the sink;
31:  end if
32: end for
```

with the leader election process. In the beginning, each node that detects the event broadcasts to its neighbourhood an event announcement message using its maximum transmission power (Algorithm (1), Line 4). It then collects, during the interval $t_{collect}$, the event announcement packets of its neighbours detecting the same event (Algorithm (1), Line 5). After this collection, each node builds its minimum spanning tree (MST) considering only its neighbours from where it received the event announcement messages. It also identifies its new neighbours in the formed topology (Algorithm (1), Line 7). To build a power efficient MST, we consider as weight of an edge separating two nodes, the transmission power allowing the communication between these two nodes. This power is computed according to equation (1). In our work, we choose to use the LMST$^-$ version [17]. For this purpose, member nodes exchange their local MST with their neighbours using the coordinator intention messages during the setting-up of routes inside the cluster.

The establishment of the tree within the cluster is done using the coordinator intention messages that are flooded (event-scoped flooding) by each candidate coordinator (Algorithm (1), Line 17). Each source node

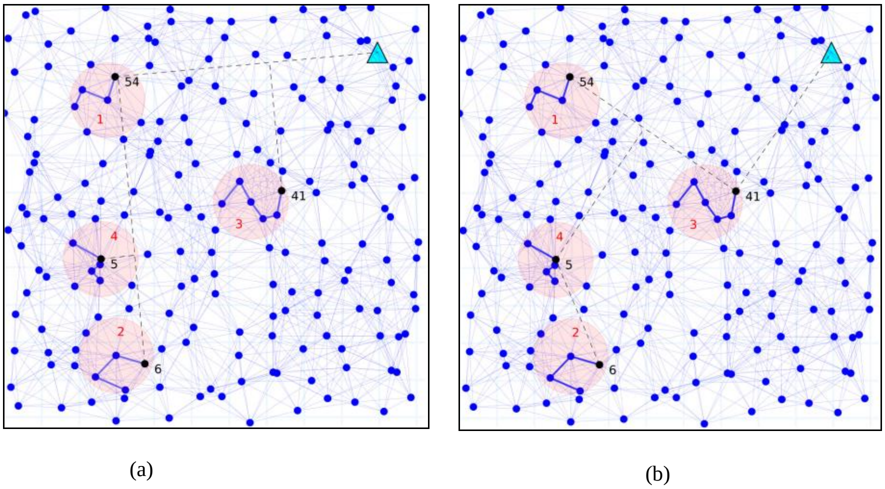


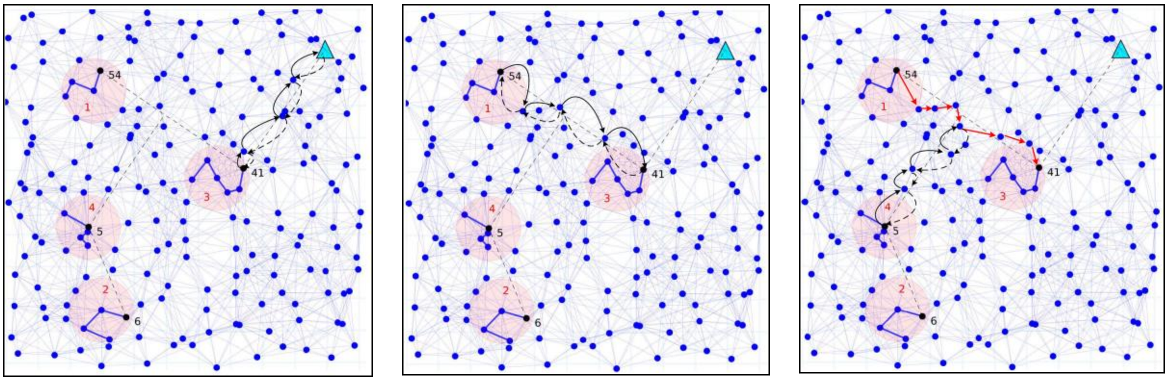
Figure 2. Two variations of PALDA according to the adopted approach to establish the underlying geometric structure: (a) PALDA-S where coordinator nodes establish their straight line segments according to the order of occurrence of events (the number noted in red within the region of influence of each event); (b)) PALDA-D where coordinator nodes establish their straight line segments according to their distances from the sink independently of the order of occurrence of events.

that receives this message rebroadcasts it, after updating, to its neighbours in the sparse topology if it finds that it has not yet received a message from the candidate coordinator specified in this message. It also rebroadcasts it in the case where the path which leads to this candidate coordinator via the sender of the message is more advantageous in terms of the total transmission power (Algorithm (1), Line 19 to 24). After the interval $t_{discovery}$, each collaborator chooses as parent node the neighbour which leads to the closest candidate coordinator to the sink node (Algorithm (1), Line 30).

B. Establishment of the underlying geometric structure

In a similar way to DST [9] and YEAST [22], PALDA is based on the establishment of an underlyinggeometric structure for the formation of the routing tree that connects the coordinator nodes with the sink node. This structure is exploited by ant agents to guide their searches and identify the relay nodes and their transmission powers. Each source node that is elected as a coordinator informs the sink node of its position. The sink sends to it then the positions of all coordinators that already exist. The sink node also informs these coordinators of the position of the new coordinator. Each coordinator and based on these positions, its own coordinates and those of the sink calculates its own straight line segment. This results in a geometric structure that is formed by the connexion of a set of straight line segments that start at each coordinator. These straight line segments can be calculated using different approaches [22]. In this work, we consider two variations of PALDA following the adopted approach to establish this underlying geometric structure:

1. **PALDA-S (Static):** with this approach, the coordinators establish their straight line segments following the order of occurrence of events. Therefore, the coordinator related to the first event that occurs is the first to creates its own straight line segment to the sink. Then comes the turn of the coordinator related to the second event which creates its straight line segment to the nearest point of the straight line segments that already exist. These steps are repeated until all coordinators establish their straight line segments. The structure which is formed by following this approach is illustrated in Fig. 2a where the order of occurrence of event is noted in red within the region of influence of each event. With this approach, a straight line segment that is initially created remains unchanged until the end of occurrence of the event. This results in a static routing structure where the same routes (formed during Phase 3) are used during the entire time period of occurrence of events. The algorithmic complexity of this process is *O(e)* where *e* is the number of events.
2. **PALDA-D (Dynamic):** with this approach, the closest coordinator to the sink node is the first to create its straight line segment to the sink. Then the second closest coordinator forms its own straight line segment to the nearest point of the first created straight line segment. The other coordinators follow the same steps and create their straight line segments to the nearest point of the straight line segments that already exist.



(a) Case 1. (b) Case 2. (c) Case 3.

Figure 3. Potential destination nodes of a given Forward Ant. A Forward Ant released from a given coordinator node moves from one node to another, according to the straight line segment of this coordinator, until arriving at its destination node or encountering a relay node of this destination node. At that moment and as second step, this Forward Ant turns into a Backward Ant and takes the reverse path to its corresponding coordinator while depositing pheromone along the way.

The structure which is formed by following this approach is illustrated in Fig. 2b. With this approach, the straight line segments are reconstructed every time a new event happens. This helps to improve the quality of the final formed tree. The algorithmic complexity of this process is $O(e)$ where e is the number of events.

Any coordinator that wants to create its straight line segment, calculates first, and according to the adopted approach, the straight line segments of the other coordinators that have to create their straight line segments first. Once created, it creates its own straight line segment to the nearest point of these straight line segments.

C. Routes formation

After the establishment of the underlying geometric structure, the coordinators proceed to the process of determining the routes that connect them with the sink. During this phase, the set of relay nodes and their transmission powers are determined. This process is based on the ant-colony-optimization metaheuristic. Therefore, each coordinator releases, every ANT_INTERVAL, a Forward Ant with the objective of finding a minimal cost path that connects this coordinator with the rest of the routing tree. Depending on the endpoint of the straight line segment of each coordinator, the latter is allowed to connect with one of the following nodes:

1. The sink node (e.g. the coordinator 41 in Fig. 3a). In this case, a Forward Ant carries, as destination and position of destination, the identifier of the sink and its coordinates, respectively.
2. Another coordinator (e.g. the coordinator 54 in Fig. 3b is allowed to connect with the coordinator 41). In this case, a Forward Ant carries, as destination and position of destination, the identifier of this coordinator and its coordinates, respectively.
3. A relay node that is responsible for the forwarding of another coordinator's data. This last coordinator must be the starting point of the straight line segment to which the straight line segment of the first coordinator is connected (e.g. the coordinator 5 in Fig. 3c) is authorized to connect with one of the relay nodes, on which, the coordinator 54 relies for the routing of its data). For this reason, each node that is selected as a relay registers the identifier of the coordinator that relies on it for the forwarding of data. In addition and in this case, each Forward Ant carries, as destination and position of destination, the identifier of this coordinator and the ending point of the extended[2] straight line segment of the coordinator source of this Forward Ant, respectively.

A sensor node i uses the following three tables during the execution of the algorithm:

1. Pheromone table: an entry of this table defines the pheromone intensity $\tau(i, j, d)$ that node i maintains on the link to its neighbour j, with respect to the destination d. The pheromone value is between 0 and 1.

---

[2] In this third case, the straight line segment of the coordinator source of a given Forward Ant is extended to ensure the meeting of a relay node.

2. Best-cost Table: this table is used to keep track of the currently known best path *cost(i,d)* from node *i* to destination *d* or a relay node of *d*.
3. Tracking table: this table is used mainly to track the paths taken by Forward Ants.

An ant can take one of the following two types: deterministic and non-deterministic. A deterministic ant always follows the next hop node that is part of the currently known best path. It is mainly used to define the relay nodes with their children and parent nodes and to reinforce the optimum path. On the other hand, a non-deterministic ant determines its path probabilistically in order to discover new paths with lower costs.

1) Heuristic information

The equation (2) represents the heuristic information that is used to direct the choices of ants toward the closest neighboring nodes. It also allows to favor the neighboring nodes that are not very far from the straight line segment of the coordinator source of this ant.

$$\eta(i,j)=\frac{1}{power_{i,j}+\delta_{j,proj_j}^{\varphi}} \tag{2}$$

In equation (2), $power_{i,j}$ represents the transmission power allowing the communication between node *i* and its neighbour *j*. This power is calculated according to equation (1). The incorporation of this power favors the choice of a closer neighbour of node *i* to be the next hop for the Forward Ant. $\delta_{j,proj_j}$ represents the perpendicular distance separating the neighbour *j* from its projection $proj_j$ on the straight line segment of the coordinator source of the considered Forward Ant. φ is a parameter that controls the importance of $\delta_{j,proj_j}$. The idea behind incorporating this distance is to make ants look around the straight line segments separating their source coordinators from their destinations and avoid orienting to neighbouring nodes which are very far from these straight line segments. In this way, their search scope is reduced by privileging, each time, the neighbouring nodes that can ensure good paths. This helps to speed up the convergence speed.

2) Next hop selection

Each Forward Ant which is at node *i* and wants to choose its next hop *n* to connect with *d* or a relay node of *d* defines first the set of candidate next hop nodes CN_i. This set consists of the set of neighboring nodes of *i* that are closer to the position of destination of this Forward Ant than node *i*.

After defining the next hop candidate neighbors, this Forward Ant selects its next hop node *n*. If it is non-deterministic, it chooses *n* using a parameter q_0 which lies between 0 and 1 $(0\le q_0\le 1)$ and another parameter *q* that is a random number uniformly distributed in $[0,1]$. If $q\ge q_0$, the exploration rule is used. Otherwise (i.e. $q<q_0$), the intensification rule is used. If the Forward Ant is deterministic, the choice of *n* is always done by the application of the intensification rule.

In the case of exploration, a Forward Ant selects its next hop by calculating the probability of choosing each neighbor $j\in CN_i$ as follows:

$$p_{i,j}^{d}=\frac{[\tau(i,j,d)]^{\gamma}\cdot[\eta(i,j)]^{\beta}}{\sum_{k\in CN_i}[\tau(i,k,d)]^{\gamma}\cdot[\eta(i,k)]^{\beta}} \tag{3}$$

In equation (3), $\tau(i,j,d)$ is the pheromone intensity that node *i* maintains on the link to its neighbor *j*, with respect to *d*. A base pheromone value is added to neighbors that don't have any corresponding pheromone entry in order to increase the probability of their selection; $\eta(i,j)$ represents the heuristic information that is defined according to the equation (2); γ and β are two parameters that control the importance of $\tau(i,j,d)$ and $\eta(i,j)$, respectively.

On the other hand and in the case of intensification, this Forward Ant chooses the neighboring node of *i* which offers the highest pheromone intensity with respect to *d*. According to these two rules, a Forward Ant that is at node *i* chooses its next hop node *n* among the candidates CN_i using the following rule:

$$n=\begin{cases}\arg\max_{j\in CN_i}\tau(i,j,d) & \textit{if } q<q_0 \\ & \textit{or deterministic Ant} \\ J & \textit{otherwise}\end{cases}$$

In this equation, *J* is the neighboring node which is selected according to the probabilities calculated by equation (3).

3) Pheromone and cost updating

Once a Forward Ant arrives at its destination node *d* or a relay node of *d*, it turns into a Backward Ant. Then, it returns to its coordinator source based on the entries added in the tracking tables of nodes that are traversed during its outward trip. In a similar way to [13], a Backward Ant uses two fields to measure the cost of a path. A first *accCost* which is initialized to zero and that serves to measure the cost during the return trip. A second *localCost* which allows other nodes to calculate the additional cost that must be added to the sender of the ant in the case where this sender is chosen as relay. By considering this latter field, we ensure a transmission power assignment that considers the bidirectionality of links between the nodes of the formed routing tree. Given that the loss of packets is intolerable when data aggregation is applied [5], this helps to ensure the reliable delivery of the data collected [10].

When node *i* receives or hears a Backward Ant from its neighbor *j*, it updates the two fields *accCost* and *localCost* of this received Backward Ant as follow [13]:

$$accCost' = accCost + linkCost + extraCost$$

$$localCost' = linkCost$$

Where *linkCost* and *extraCost* are defined as follow:

$$linkCost = power_{i,j} / P_{\max}$$

$$extraCost = \max(linkCost - localCost, 0)$$

After this operation, node *i* updates its pheromone and best-cost tables according to the type of this Backward Ant. To update these tables, we follow the same updating process proposed in [25]. Two reasons make us adopt such process. First, regarding the updating of pheromone, there is an emphasize on the best-so-far path. Therefore, the search is more directed around this path. Second, pheromone intensity is constrained within the range [0,1]. This helps to avoid the situation of stagnation and promote the diversification in the selection of paths.

Algorithm 2. The updating of node *i* to the entries in its pheromone and best-cost tables that correspond, respectively, to its neighbour *j* and destination *d* using *accCost'*.

```
1: if deterministic Ant then
2:     $cost(i,d) \leftarrow accCost'$;
3:     $\tau(i,j,d) \leftarrow \tau(i,j,d) + 1/accCost'$;
4: else
5:     if $accCost' < cost(i,d)$ then
6:         $cost(i,d) \leftarrow accCost'$;
7:         $\tau(i,j,d) \leftarrow 1$;
8:     else
9:         $\tau(i,j,d) \leftarrow \tau(i,j,d) + \frac{cost(i,d)}{\zeta \cdot accCost'}$;
10:    end if
11: end if
12: $\tau(i,j,d) \leftarrow \min\{\tau(i,j,d), 1\}$;
```

The algorithm (2) illustrates this updating process. In the case of a non-deterministic Backward Ant, node *i* updates the known best cost to the new cost and sets the pheromone intensity to the maximum allowed value, which is 1, if a better path is found. Otherwise and as shown in line 9 of algorithm (2), a small amount of pheromone is added. In this line 9, ζ is a parameter that control the amount of pheromone added and which is set in our experiments to 20.

On the other hand and in the case of a deterministic Backward Ant, node *i* updates the best-cost table entry that correspond to *d* to the new cost and a certain amount of pheromone that is inversely proportional to this new cost is added. Furthermore, node *i* becomes relay. It also updates its dependency set which consists of its parent and children nodes and adjusts its transmission power so as to reach the farthest neighbour among this set. Note that this dependency set is flushed at interval in order to adapt to routes dynamics.

During this updating process, the broadcast nature of wireless communication is exploited and the overheard Backward Ants are used to update the different tables. This helps to speed up the dissemination of information, and therefore, speed up the search for the best path. However, we note that an overheard Backward Ant is always considered as a non-deterministic ant when updating the different tables.

Besides this updating process, each node *i* decreases, each EVAPORATION_INTERVAL, the pheromone intensity in all entries of its pheromone table according to equation (4). Pheromone evaporation gives more significance to the most updated information by making past information less important.

$$\tau(i,j,d)' \leftarrow (1-\rho) \cdot \tau(i,j,d) \qquad \textbf{(4)}$$

In equation (4), ρ stands for the fraction of pheromone that is evaporated.

D. *Data aggregation*

When data are transmitted along the routing paths, PALDA performs data aggregation at two levels:

1. **Intra-cluster aggregation:** each collaborator that is considered as an intra-cluster relay has the potential to aggregates the packets being relayed. Then, the coordinator node aggregates the received packets from its members and sends the result to the sink node.
2. **Inter-cluster aggregation:** when the routes overlap outside the cluster, relay nodes that know these overlaps aggregate the received packets and send the results to their parents in order to be routed to the sink node.

6. PERFORMANCE EVALUATION

In this section, we evaluate the performance of the proposed PALDA algorithm and its two variations. For this, we compare their performance to two other known routing protocols: ECMANSI [13,25] and DST [9,22].

ECMANSI is used to evaluate the performance of PALDA with respect to the phase of establishment of the routing tree that connects the coordinator nodes with the sink. In fact, ECMANSI is an ant-based protocol that is dedicated to the multicast problem, and therefore to the Steiner tree problem. It also aims to minimize the sum of the transmission powers of non-leaf nodes of the constructed multicast tree. Given that we are interested, instead of multicast communication, in collecting data from a set of source sensor nodes and delivering them to the sink. Therefore and with ECMANSI, the core node (data source) in the multicast session is considered as sink node and the destination nodes are considered as source nodes.

A. *Methodology*

Using J-sim simulator [27], a series of simulations were conducted to evaluate the performance of the proposed protocol PALDA. For this evaluation, a methodology that is similar to the one followed in [7,22] was adopted. Table 1 presents the default simulation parameters. For some simulations, these parameters will be changed and the changes will be mentioned in the corresponding section. 33 different networks (different seeds) were generated with nodes uniformly and randomly deployed. For each network, different events were generated at random positions. The first event starts at time 1000 s and the other events start at an uniformly distributed random time between [1000, 3000] seconds. Network density was defined according to the following 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 collecting field. With PALDA-D, PALDA-S and ECMANSI, each coordinator was allowed to carry out a maximum number of searches which is equal to 150. However, with PALDA-D, a coordinator reset this number to 0 when it changed its straight line segment. The figures presented in the following have been made by averaging the simulation results obtained from the 33 generated topologies and will be shown with a 95% confidence interval. All evaluated algorithms used periodic simple aggregation strategy [5], with which, the aggregator nodes transmit periodically the received and aggregated data. For ECMANSI, the same parameter values that were adopted in [13] were used. The different protocols were evaluated according to the following metrics:

Tableau 1. Simulation parameters.

Parameter	Value
Sink	1(top right)
Number of sensor nodes	300
Communication radius (m)	80
events	3
Event duration (hours)	3
Simulation duration (hours)	4
Notification interval (sec)	10
Density	20
(path loss exponent)	2
E_{elec}	50 nJ/bit
ε_{amp}	10 pJ/bit/m^2
φ	2
γ, β	1, 1
q_0	0.5
ρ	0.1
Ant_Interval (Sec)	3
Evaporation_Interval (Sec)	2

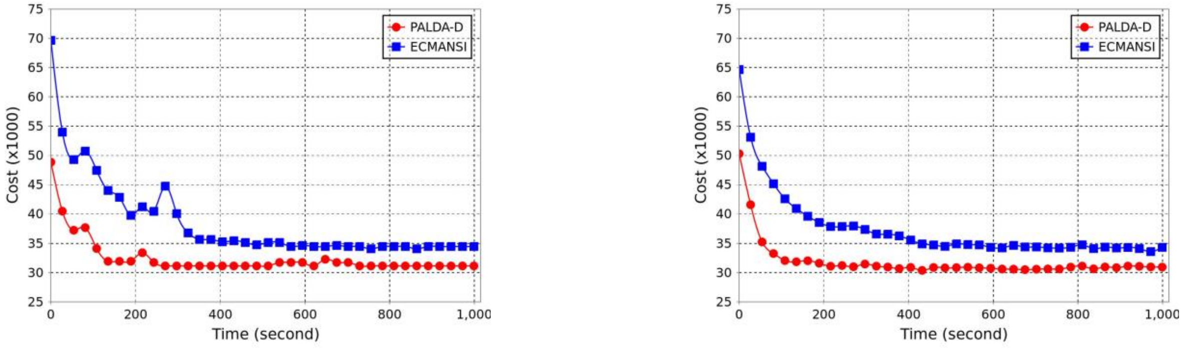
1. **Routing tree cost:** this cost represents the total transmission power of the final routing tree constructed by each protocol.
2. **Control packet overhead:** represents the number of control packets that are used to establish each routing structure and assign the transmission powers to nodes.
3. **Aggregation rate:** represents the ratio between the number of all data packets sent and the number of data packets received by the sink node.
4. **Total energy consumption:** represents the total energy that is spent by each protocol. It includes the energy spent on the transmission of data packets plus the energy consumed by control packets.

For energy consumption, the model proposed in [20] was adopted. Thus the energy dissipated during the transmission of a packet of size l over a distance δ is given by the formula:

$$E_{TX}(l,\delta) = lE_{elect} + l\varepsilon_{amp}\delta^2$$

And the energy dissipated when receiving a packet of size l is given by the formula:

$$E_{RX\,(l)} = lE_{elec}$$



(a)

(b)

Figure 4. Routing tree cost versus time (a) the same network. (b) Average cost for 33 networks.

Where E_{elec} is the base energy required to run the transmitter or receiver circuity and ε_{amp} is the unit energy required for the transmitter amplifier. In our model, the energy dissipated during the perception and processing were ignored because it is negligible compared to the energy consumed in the transmission.

B. Convergence speed

Here, the convergence speed of the proposed protocol is evaluated. For this purpose, PALDA-D was compared to ECMANSI by considering the evolution of the cost of the established routing tree over 1000 seconds of simulation (Fig. 4a and Fig. 4b). In this simulation scenario, a network of 100 nodes with 6 events occurring simultaneously was considered. Moreover, it was assumed that each event could be detected by a single sensor node. Therefore, the scenarios evaluated here do not consider any clustering and thus no collaborator nodes. The maximum number of searches was set to ∞.

As shown in Fig. 4a and Fig. 4b, we notice for each protocol that the cost decreases continuously during the first seconds and stabilizes gradually whenever the time advances. However, we notice that PALDA-D converges rapidly compared to ECMANSI. In fact, ants in PALDA-D base their searches on an underlying geometric structure. Initially, each coordinator knows the coordinator that is supposed to connect with and ants are oriented following the straight line segments of their respective coordinators. This is not the case with ECMANSI where an initial routing tree which resembles a shortest path tree connecting the coordinators with the sink is first built. Then, ants that are released by each coordinator, search in all directions in order to find a better path that connects this coordinator with the rest of the routing tree. We notice also from these two figures that PALDA-D builds routing trees with lower cost compared to ECMANSI. This superiority is due to the strategy followed by PALDA-D to combine the paths of coordinators. In fact, with ECMANSI each node, relay or coordinator, is associated with a height which corresponds to the highest identifier of coordinators relying on this node to connect with the sink. The height of the latter is equal to ∞. Thus, a coordinator is allowed to connect directly either with the sink or with a relay node having a height greater than its identifier. According to this approach, the final routing tree that is built by ECMANSI depends on the identifiers of coordinator nodes and their positions. However, with PALDA-D, the closest coordinators to the sink are the first to establish their paths, which favors the construction of low cost routing trees.

C. Event size effect

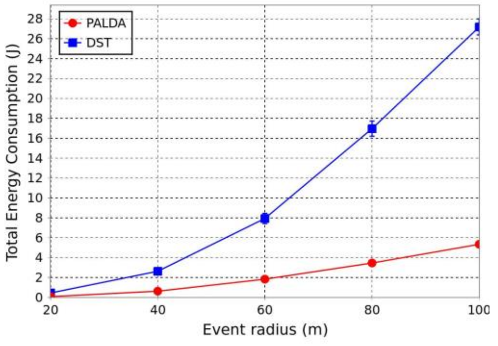
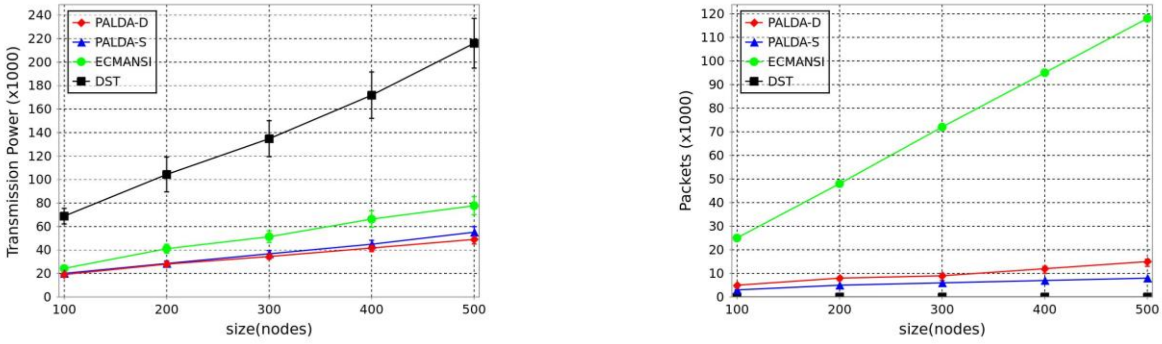
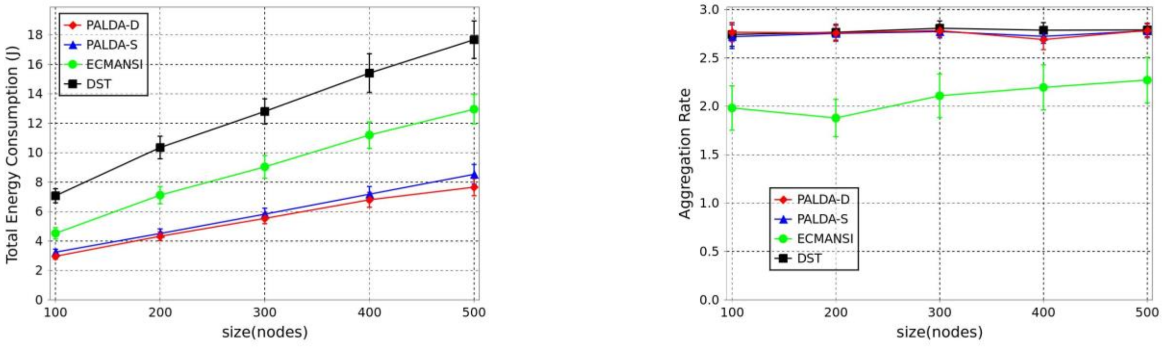


Figure 5. Impact of event size.



(a) Routing tree cost.

(b) Control overhead.



(c) Total energy consumption.

(d) Aggregation Rate.

Figure 6. Impact of network size

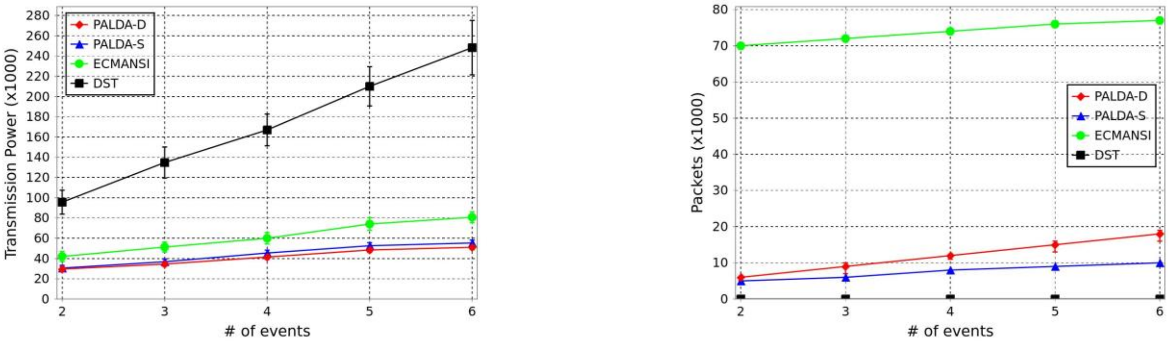
In order to show the superiority of the proposed protocol with regard to the adopted clustering approach, we evaluate here the impact of event size. For this purpose, PALDA was compared to DST by considering a single event and by increasing its radius from 20 m to 100 m. The results which show the total energy consumed by source nodes are shown in Fig. 5.

As shown in this figure, we notice that the total energy consumption increases with the increase of event radius, because, more source nodes are, each time, considered. However, we notice that PALDA consumes less energy than DST. This superiority is due to the fact that collaborator nodes with PALDA report their data using an energy-efficient tree. This is not the case of DST where a shortest hop path tree is used within a cluster.

D. Network size effect

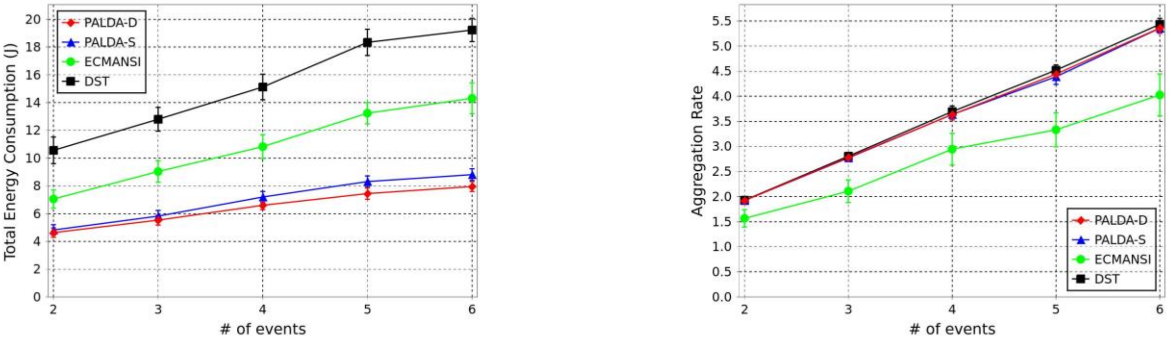
In this simulation scenario, the impact of network size on the algorithms performance is evaluated. To achieve this, the network size was increased from 100 to 500 nodes. In this scenario and with the two scenarios that are simulated in the remainder of this performance evaluation section, it was assumed that each event was detected by a single sensor node. Therefore, no clustering was considered.

As shown in Fig. 6a, the routing tree cost increases each time the number of nodes increases. This is due to the fact that routes are longer when the number of nodes increases. Fig. 6a shows also that DST constructs the highest cost routing trees given it does not perform any transmission power assignment process while ECMANSI builds the worst. We notice also that PALDA-D is slightly better compared to PALDA-S, because, the routes that are formed by PALDA-D do not depend on the order of occurrence of events. In turn, they are each time re-established to improve the final routing tree's cost. However, this superiority of PALDA-D compared to PALDA-S is accompanied by a greater expenditure of control packets (Fig. 6b). In this sense, DST spends the least number of control packets and ECMANSI spends the most. The higher overhead of ECMANSI is due in particular to the use, in addition to Forward Ants and Backward Ants messages, of other control packets which are periodically flooded over the entire network.



(a) Routing tree cost.

(b) Control overhead.



(c) Total energy consumption.

(d) Aggregation Rate.

Figure 7. Impact of the number of events

The shows the aggregation rate of each protocol. ECMANSI has the lowest aggregation rate because the other protocols have the certainty to find overlapping routes view the strategies that are followed to combine the routes of coordinator nodes. In terms of the total energy consumption, Fig. 6c illustrates results which are, in particular, in correspondence with the costs of the final constructed routing trees. This energy is also influenced by the number of control packets expended by each protocol.

E. Events number effect

In this simulation scenario, the behaviour of the different protocols when the number of events increases is evaluated. To do this, the number of events was increased from 2 to 6. The results are shown in Fig. 7.

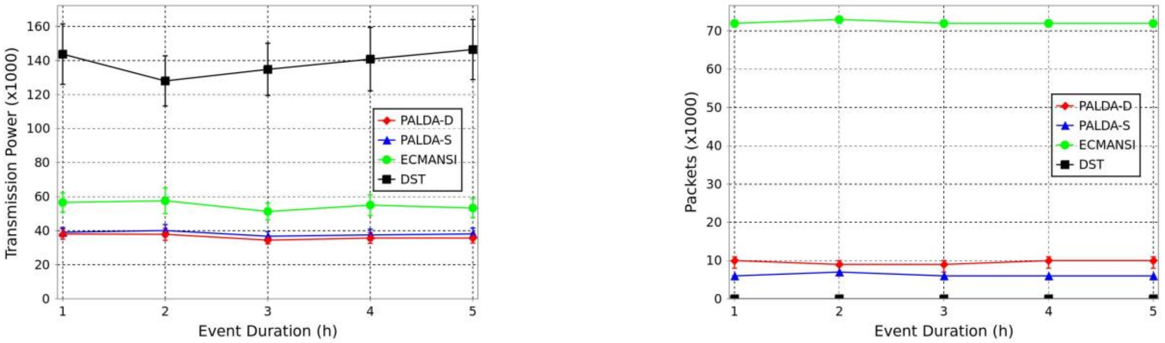
The Fig. 7a shows that the costs of the constructed routing trees increases when the number of events increases, because, more routes are each time created. This is also true with regard to the number of control packets spent (Fig. 7b). In terms of aggregation rate and as shown in Fig. 7d, we notice that this rate increases as the number of events increases, because more data are aggregated. Fig. 7c shows the total energy consumed by each protocol. it shows a correspondence between this energy and the total transmission powers of the routing trees formed.

F. Event duration effect

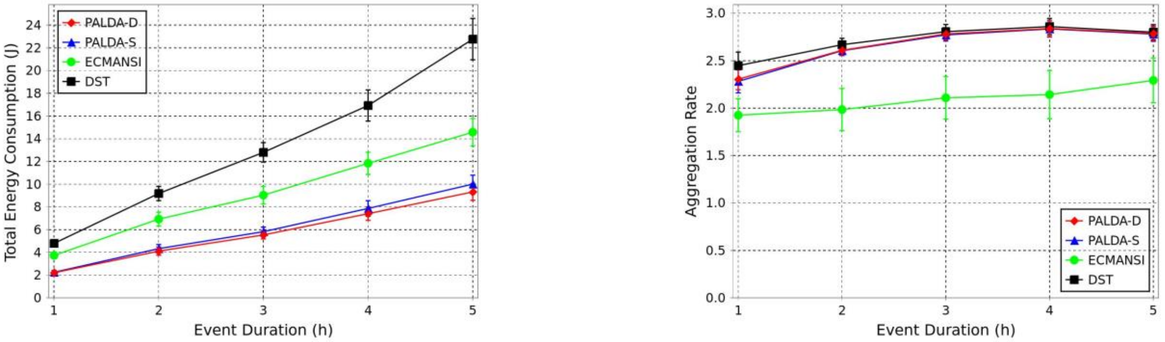
In this simulation scenario, the impact of event duration is evaluated by varying this duration from 1 to 6 hours. The results are shown in Fig. 8. Obviously, the costs of the constructed routing trees (Fig. 8a), the number of control packets (Fig. 8b) and the aggregation rates (Fig. 8d) remain relatively at the same level, because, the same configuration is considered (3 events and a network of 300 nodes). However, and as shown in Fig. 8c, the difference between the total energies that are consumed by the different protocols increases and becomes greater each time the duration of events increases. This is due to the difference between the costs of the routing trees that are established by each protocol.

7. CONCLUSION

In this article, we proposed the protocol PALDA that is dedicated for event-based WSNs. the goal of PALDA is to define, once one or several events occur, a routing structure that maximizes data aggregation with the



(a) Routing tree cost. **(b) Control overhead.**



(c) Total energy consumption. **(d) Aggregation Rate.**

Figure 8. Impact of event duration.

minimum sum of transmission powers of nodes constituting this structure.

To achieve this objective, we followed two main steps. During the first step where the clustering is performed, we formed, within each cluster, an energy-efficient tree that is used for the collection of data of cluster's members. With regard to the second step, we used the ant-colony-optimization metaheuristic to form the routes between the different cluster-heads, thus, defining the relay nodes and their transmission powers.

Our proposed protocol PALDA was widely compared to two other known protocols, ECMANSI and DST, regarding different factors: convergence time, network size, number of events, event duration and event size. With the consideration of an underlying geometric structure and the adoption of a better strategy for the formation of routes between the various coordinators, the obtained results showed that PALDA is better than ECMANSI. Moreover, even if PALDA require more control packets compared to DST, the simulation results showed that PALDA outperformed DST due to the advantage of adjusting the transmission powers of nodes participating in the collection of data.

As future work, we aim to include the real-time constraint. In fact, an aggressive assignment of transmission powers generates routes with a large number of hops, and thus, an important latency time. For this, it is necessary to ensure a trade-off between the process of transmission power assignment and the timely delivery of the collected data.

REFERENCES

[1] Akyildiz IF, Su W, Sankarasubramaniam Y, Cayirci E. Wireless sensor networks: a survey. Comput. Netw., 2002; 38: 393-422.

[2] Dressler F. Self-Organization in Sensor and Actor Networks. Wiley; 2007.

[3] Bagaa M, Challal Y, Ksentini A, Derhab A, Badache N. Data Aggregation Scheduling Algorithms in Wireless Sensor Networks: Solutions and Challenges. IEEE Commun. Surv. Tutor., 2014; 16: 1339-1368.

[4] Amer A, Fawzy AE, Shokair M, Saad W, El-Halafawy S, Elkorany A. Balanced Energy Efficient Grid Based Clustering Protocol for Wireless Sensor Networks. Int. J. Com. Dig. Sys., 2017.

[5] Solis I, Obraczka K. In-network aggregation trade-offs for data collection in wireless sensor networks. Int. J. Sens. Netw., 2006; 38: 200-212.

[6] Nakamura EF, Ramos HS, Villas LA, de Oliveira HABF, de Aquino ALL, Loureiro AAF. A reactive role assignment for data routing in event-based wireless sensor networks. Comput. Netw., 2009; 53: 1980-1996.

[7] Villas LA, Boukerche A, Ramos HS, de Oliveira HABF, de Araujo RB, Loureiro AAF. DRINA: A Lightweight and Reliable Routing Approach for In-Network Aggregation in Wireless Sensor Networks. IEEE Trans. Comput., 2013; 62: 676-689

[8] Villas L, Boukerche A, de Araujo RB, Loureiro AAF. Highly Dynamic Routing Protocol for data aggregation in sensor networks. The IEEE symposium on Computers and Communications, Riccione, Italy; 2010; 496-502.

[9] Villas LA, Guidoni DL, Araujo RB, Boukerche A, Loureiro AAF. A Scalable and Dynamic Data Aggregation Aware Routing Protocol for Wireless Sensor Networks. ACM MSWiM '10, Bodrum, Turkey; 2010; 110-117.

[10] Santi P. Topology Control in Wireless Ad Hoc and Sensor Networks. ACM Comput. Surv., 2005; 37: 164-194.

[11] Liao WH, Kao Y, Fan, CM. Data aggregation in wireless sensor networks using ant colony algorithm. Journal of Network and Computer Applications, 2008; 31: 387-401.

[12] Shen CC, Jaikaeo C. Ad Hoc Multicast Routing Algorithm with Swarm Intelligence. Mobile Networks and Applications, 2005; 10: 47-59.

[13] Jaikaeo C, Sridhara V, Shen CC. Energy conserving multicast for MANET with swarm intelligence. IEEE MASS 2005, Washington, DC, USA; 2005.

[14] Krishnamachari L, Estrin D, Wicker S. The impact of data aggregation in wireless sensor networks. IEEE ICDCS 2002 Workshop, Vienna, Austria; 2002; 575-578.

[15] Intanagonwiwat C, Estrin D, Govindan R, Heidemann J. Impact of network density on data aggregation in wireless sensor networks. Proceedings 22nd International Conference on Distributed Computing Systems, Vienna, Austria; 2002; 457-458.

[16] Tan HO, Korpeoglu I. Power Efficient Data Gathering and Aggregation in Wireless Sensor Networks. SIGMOD Rec, 2003; 32: 66-71.

[17] Tan HO, Korpeoglu I, Stojmenovi I. Computing Localized Power-Efficient Data Aggregation Trees for Sensor Networks. IEEE Trans. Parallel Distrib. Syst., 2011; 22: 489-500.

[18] Li N, Hou JC, Sha L. Design and analysis of an MST-based topology control algorithm. IEEE Trans. Wireless Commun., 2005; 4: 1195-1206.

[19] Toussaint GT. The relative neighbourhood graph of a finite planar set. Pattern Recognit., 1980; 12: 261-268.

[20] Heinzelman WB, Chandrakasan AP, Balakrishnan H. An application-specific protocol architecture for wireless microsensor networks. IEEE Trans. Wireless Commun., 2002; 1: 660-670.

[21] Benaouda N, Mostefai M. A New Two-Level Clustering Scheme for Partitioning in Distributed Wireless Sensor Networks. International Journal of Distributed Sensor Net-works, 2015; 11: 435048.

[22] Villas LA, Boukerche A, de Oliveira HABF, de Araujo RB, Loureiro, AAF. A spatial correlation aware algorithm to perform efficient data collection in wireless sensor networks. Ad Hoc Netw, 2014; 12: 69-85.

[23] Lin C, Wu G, Xia F, Li M, Yao L, Pei Z. Energy efficient ant colony algorithms for data aggregation in wireless sensor networks. J. Comput. Syst. Sci., 2012; 78: 1686-1702.

[24] Fawzy AE, Amer A, Shokair M, Saad W. An Efficient Design of Cluster Size for Maximizing Outcomes of Direct Transmission in Cluster-Based Wireless Sensor Networks. Int. J. Com. Dig. Sys., 2016.

[25] Shen CC, Li K, Jaikaeo C, Sridhara V. Ant-based Distributed Constrained Steiner Tree Algorithm for Jointly Conserving Energy and Bounding Delay in Ad Hoc Multicast Routing. ACM Trans. Auton. Adapt. Syst., 2008; 3: 1-27.

[26] Bagci H, Korpeoglu I, Yaz c A. A Distributed Fault-Tolerant Topology Control Algorithm for Heterogeneous Wireless Sensor Networks. IEEE Trans. Parallel Distrib. Syst., 2015; 26: 914-923.

[27] Sobeih A, Hou JC, Kung LC, Li N, Zhang H, Chen WP, Tyan HY, Lim H. J-Sim: a simulation and emulation environment for wireless sensor networks. IEEE Wireless Communications, 2006; 13: 104-119.

Nadjib Benaouda Nadjib Benaouda received his master-degree in 2012, in computer science from the university of Setif-1, Setif, Algeria. He is currently a Ph.D. Student at the university of Batna-2, Batna, Algeria. His research interests focus on wireless sensor/ad hoc networks, cooperating objects and Multi-agent systems.

Ammar Lahlouhi Ammar Lahlouhi is an Associate Professor at the Department of Computer Science, University of Batna-2, Batna, Algeria. In 2006, he received his Ph.D. degree from the University of Constantine. Currently, he is interested in pervasive multi-agent systems. He is the author of MASA-Method Methodology for the Development of Heterogeneous Multi-agent Systems, Head of projects financed by the Algerian government including MASp (Pervasive Multi-agent Systems), and Member of the European Association EASST and of CRISTEL (a European Union Common Program TEMPUS project). He is Associate Editor of Bonfring International Journal of Software Engineering and Soft Computing and a member of program committees of several conference series (IMETI, ICEME, IDREC, IMCIC, CCISE, WMSCI, etc.).