Self-organization in Sensor and Actuators Networks: Strategies and their optimizations

Bilel Romdhani, Dominique Barthel, Fabrice Valois

INRIA

INSTITUT NATIONAL DE RECHERCHE EN INFORMATIQUE ET EN AUTOMATIQUE

Self-organization in Sensor and Actuators Networks: Strategies and their optimizations

Bilel Romdhani — Dominique Barthel — Fabrice Valois

N° 7440

October 2010

ISSN 0249-6399 ISRN INRIA/RR--7440--FR+ENG

INSTITUT NATIONAL
DE RECHERCHE
EN INFORMATIQUE
ET EN AUTOMATIQUE

INRIA

centre de recherche
GRENOBLE - RHÔNE-ALPES

Self-organization in Sensor and Actuators Networks: Strategies and their optimizations

Bilel Romdhani*†‡ , Dominique Barthel‡ , Fabrice Valois† §

Thème : 1
Équipes-Projets SWING

Rapport de recherche n° 7440 — October 2010 — 24 pages

Abstract: In Wireless Sensors and Actuators Networks (WSANs), actuator nodes are nodes richer in resources (processing capacity, power transmission and energy storage) and better suited than sensor nodes to process the data, make decisions based on sensed values and perform appropriate actions. In addition, in order to provide timely action, coordination between sensors and actuators is necessary. Thus, in addition to the classical energy constraints of Wireless Sensor Networks (WSNs), WSANs also impose new challenges such as how to support and benefit from the nodes heterogeneity while preserving energy in the self-powered sensor nodes. New communication protocols, specific to WSANs, are needed. In this report, we propose a hybrid self-organizing data-collection protocol in order to provide energy efficiency, low end-to-end delay and high delivery ratio while taking advantage of the resource available on the actuators nodes in the network. This new self-organization protocol constructs its structure from the actuators and other resource-plentiful nodes. The nature of the structure is different inside and outside of transmission range of these resourceful nodes. Two variant of our proposal are detailed also in this report.

Key-words: Self-organization, Wireless Sensor and Actuator Networks, Data-Collection.

§ This work was partially funded by the ANR VERSO ARESA2 project

* Corresponding author
† University of Lyon, INRIA, INSA Lyon, CITI, F-69621, France
‡ Orange Labs Meylan France

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe, 38334 Montbonnot Saint Ismier
Téléphone : +33 4 76 61 52 00 — Télécopie +33 4 76 61 52 52

Stratégies d'auto-organisation dans les réseaux de capteurs et d'actionneurs et leurs optimisations

Résumé : Dans un réseau sans fil de capteurs et actionneurs (Wireless Sensors and Actuators Networks WSAN), les nœuds actionneurs sont des nœuds riches en ressources (capacité de traitement, de puissance de transmission et de stockage en mémoire et en énergie) et sont mieux adaptés que les nœuds capteurs à traiter les données, prendre des décisions fondées sur les observations remontées par les capteurs et effectuer les actions appropriées. En outre, afin d'offrir une action dans les délais, la coordination entre les capteurs et les actionneurs est nécessaire. Ainsi, outre les contraintes de l'énergie comme dans les réseaux de capteurs classiques (Wireless Sensor Networks WSN), les WSANs imposent également des contraintes de temps sous la forme de délai de bout-en-bout. Ainsi des nouvelles propositions de communication spécifiques aux WSAN sont nécessaires, qui fournissent un délai de bout-en-bout contrôlé tout en préservant l'énergie au niveau des noeuds capteurs. Dans ce contexte, nous décrivons dans ce rapport une proposition d'auto-organisation dans les WSANs. Notre proposition crée une structure autour de chaque actionneur. Une phase d'initialisation et de construction de gradient s'appuyant sur des anneaux concentriques autour des actionneurs, ensuite une phase de découverte de voisinage, de construction de structure logique et enfin une phase de remontée des observations des capteurs vers les nœuds actionneurs. Deux variantes de notre proposition sont présentées par la suite pour répondre à quelques points faibles de la proposition de base.

Mots-clés : auto-organisation, Réseau de capteurs et actionneurs, Collection de données

1 Introduction

In recent years, we witnessed the appearance of multi-hop wireless networks. With distinct capabilities, properties, characteristics and target applications, we can identify Wireless Sensor Networks (WSNs) [1], [2], [3] and Wireless Sensor and Actuator Networks (WSANs) [4], [5].

A WSN is generally a network composed of a large number of autonomous and self-organized low power nodes called sensors. These nodes are deployed in specific areas in order to measure or detect events in the physical world and route them to a specific node called sink which is responsible for the monitoring of the field or for interconnecting to a Wide Area Network such as the Internet. WSNs have different constraints compared to other wireless multi-hop networks, namely energy constraints and traffic characteristics. So the design of new protocols and algorithms dedicated for WSN was deemed necessary [6].

Recently, we have observed the extension of WSNs to include new nodes called actuators which are responsible of reacting onto the physical world based on events reported by sensor nodes. This architecture is called WSAN in which a number of actuators (relatively small compared to the number of sensor nodes) is deployed in the sensor field.

In WSANs, actuator nodes are resource-rich devices (higher processing capabilities, transmission power and larger battery capacity) and are better suited than sensor nodes to process the data, make decisions based on the sensed observation and perform appropriate actions. Applications of WSANs include home automation, health care, industrial application, etc ... [5]: a typical example of WSANs is fire detection, where sensors relay the existence and location of the fire to the water sprinkler actuator. The information should arrive fast enough to the actuator so that a decision is made and action taken before the fire spreads beyond easy control. Thus, in addition to energy constraints and self-organization characteristics as in WSNs, WSANs also impose new challenges such as timing constraints and new exploration fields such as node heterogeneity.

This heterogeneity in WSANs opens an opportunity for new algorithms and protocols, considering and taking advantage of this characteristic. Previously proposed self-organization, MAC and routing protocols for WSN have not been designed to support the capabilities of heterogeneous devices [5]. They are not usually designed to exploit resource-rich devices to reduce the communication burden on low power nodes. Consequently, they may not be best suited for several applications of such heterogeneous sensor and actuator networks.

Our idea in this report is to propose a self-organization protocol for WSAN taking advantage of the resource available on the actuator nodes in the network. This new self-organization protocol constructs its structure around the actuators and other resource-plentiful nodes. The nature of the structure is different inside and outside of transmission range of these resourceful nodes. Hence, our proposal will be a combination of two self-organizing protocols. The first one is applied inside the transmission range of the actuator. We propose that the actuator node itself initiates the construction of a structure based on which sensor nodes can send their data. This structure is constructed gradually from the actuators out to every sensor node in its transmission range. The second self-organization protocol is applied outside the transmission range of the actuator. Non-covered sensors should by themselves construct a self-organized structure thereby creating paths to join the actuator node through a covered area. It

should be noted that the problematics of WSANs is different from that of multi-sinks WSNs. Indeed, in the former case the actuator nodes do not only collect data but also react on the physical world based on observations by the sensor nodes. Since actuators are considered as having more generous power supply, they should be used more intensively for construction of the logical topology and for communications. The communication and coordination between actuators is out of scope of this work. As the actuators are usually bigger and resource-richer nodes, they can contain higher-quality antennas and transmit using higher power. Thus, we suppose that the actuators can communicate directly with one another [7].

2 Related Work

The WSNs and WSANs are formed by autonomous and self-organized nodes communicating with one another using radio interfaces. As the radio range of nodes is limited, multi-hop communication is needed, so each node must act either as a terminal or a router based on network needs. Typically, we find in these networks two types of data traffic: "one-to-all" traffic and "all-to-one" traffic.

First, for one-to-all communication, a specific node (usually a sink or actuator) aims at sending or at disseminating a set of data to all nodes in the sensor network. Among the problems encountered in this type of traffic is flooding. Flooding is commonly used for path discovery or to spread information or queries to sensor nodes. Several protocols for the propagation of such information have been proposed in the literature. In [8], these protocols are classified into four main families: we first distinguish the blind flooding protocols, where each node rebroadcasts all received messages to its neighborhood. The second type is the protocols based on probabilities. A node rebroadcasts the message based on a probability value. Then we find the protocols based on location, where the sender inserts its position in the broadcast message header. Upon receiving a message to retransmit, each node calculates the additional coverage area obtained if it rebroadcasts this message. If the additional area is less than a threshold value, the node does not rebroadcast the received message. Finally, we find the protocols based on knowledge of the neighborhood. This type of protocol requires that each node has knowledge of its one- or/and two-hop neighborhood. The information is obtained through periodic exchanges of `Hello` messages. Each node adds the list of its neighbors in the header of each broadcast message. Based on its neighborhood information, each node decides if it rebroadcasts the message or not. MPR (MultiPoint Relay Protocol) [9] is a protocol belonging to the last family. In this scheme, the transmitting node selects neighboring nodes that should relay the message for broadcast to all nodes in the network. The IDs of selected nodes are stored in the header of the message to be transmitted. A neighbor node that has been selected to retransmit the message also determines its own list of nodes requested to retransmit the message after itself. This process is iterated until broadcast is completed. The selected nodes are called relay nodes and form a subset of neighbors that cover the entire region covered by all neighboring nodes.

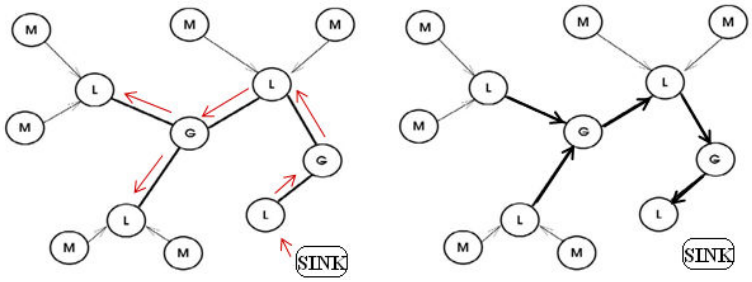
Second, all-to-one communication occurs from sensor nodes to the sink or actuator nodes, forming a traffic called data collection or data gathering. Sev-

eral protocols have been proposed for this type of traffic. We find the BBDD (Backbone Based Data Dissemination) protocol [10] that aims both at reducing the energy required to disseminate information from the sink to all nodes in the network and at facilitating the data collection from sensors to sink node. BBDD uses a two step approach. It first runs Legos [11] which is a self-organizing protocol dedicated to WSN aiming at generating a logic non-directed structure called backbone through local information and local decisions upon nodes' deployment. Secondly, a BBDD sink sends a request through this backbone to create a directed structure in the network. This request is propagated through the logical structure (Figure 1(a)) and directs the Legos backbone. If there is data to report to the sink, once this data arrives at the backbone, it is sent back to the sink node through the path constructed in the first step (Figure 1(b)).

Our proposal is also based on an adaptation of Legos. To support a Legos structure, nodes are in one of the following three states: leader, gateway or member. A network running Legos has a two-level hierarchy: at the higher level, a backbone formed by dominating nodes (leaders and gateways); at the lower level, the dominated nodes (member) where each node is linked to one leader. A leader is in charge of all communications in its 1-hop neighborhood. Leaders are spaced 2 hops apart exactly. A gateway is in charge of interconnecting the leaders. After detecting the leaders in their neighborhood, members attach to one leader. Once a dominated node is attached to a leader, its incoming and out-going communications are controlled by this leader. Instead of collecting physical topology information, nodes take existing local Legos structures into account to make decisions. To join the self-organizing structure, a newly deployed sensor node N starts by listening to the medium to hear a Leader_Broadcast_Msg. If it does hear a leader, then N becomes a member and it will be attached to this leader (Figure 2(a)). If N does not detect a leader in its neighborhood, it initiates a Neighbor Discovery by broadcasting a discovery message. This broadcast is used to see if there is an organization in its one hop neighborhood. If a member node M receives this solicitation, it replies to inform N about the existence of a structure in its neighborhood. Upon receiving this notification, node N becomes a Leader and M becomes a gateway (Figure 2(c)). If the first two steps are unsuccessful, N will consider itself the first node in the network: it will advertize itself as a leader by periodically broadcasting a Leader_Broadcast_Msg (Figure 2(b)).

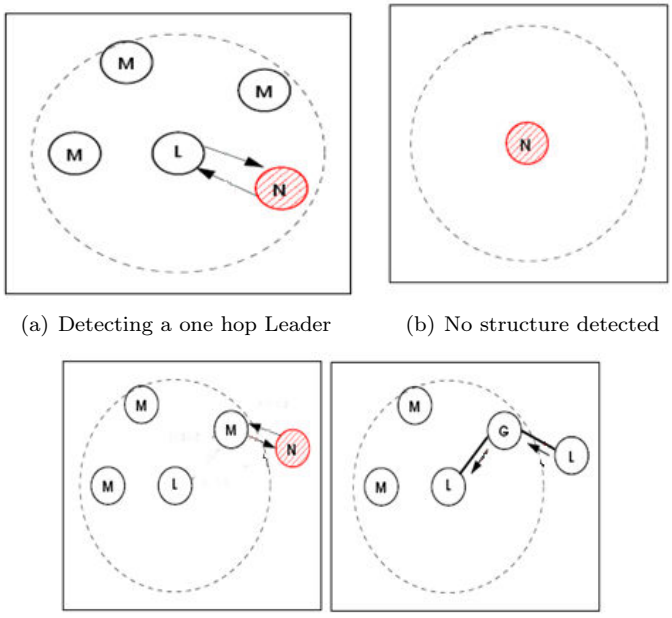
3 FAR-Legos : a hybrid self-organizing data collection protocol for sensor and actuator networks

In this paper, we propose a hybrid self-organizing data-collection protocol to provide energy efficiency, minimum end-to-end delay and high delivery ratio while taking advantage of the resources available on the actuators nodes. In this section, we will describe the hypothesis and our self-organization protocol. Our proposal can be divided into four phases: during the first phase the actuator assigns ranks to the sensor nodes within its transmission range. These ranks will be used as a gradient when there is a data to send to actuator. Sensors in the



(a) Query forwarding (b) Data forwarding

Figure 1: BBDD Backbone Based Data Dissemination



(a) Detecting a one hop Leader (b) No structure detected

(c) Detecting a two hops Leader

Figure 2: Joining the Legos structure

covered area will later use this rank as pseudo-geographic information to route data to the actuator node. Then, sensor nodes launch a neighbor discovery phase. In a third phase, sensors in the non-covered area create a structure derived from the Legos [11] protocol; this structure aims at finding a path to a covered area. Finally the fourth phase is the data-collection phase. In this phase, a sensor detecting an event finds a path to send the data message to the actuator.

3.1 Hypothesis

We consider a WSAN consisting of a large number of sensors nodes and a much smaller number of actuators nodes. We assume that at t = 0 all sensor and ac-

tuator nodes are deployed. We assume that the transmission power of actuator nodes is greater than that of sensor nodes and that it is adjustable. Also, we suppose that there is area which will be not covered by any actuator node in the network. Furthermore, we assume that all sensors have the same immutable transmit power, that is known to the actuators. No geographic information is available for any network node. Finally, we suppose that the links are symmetric and that the network is stable and static: no mobility and no deployment of new nodes in the network. The latter assumption will be reconsidered later in this paper.

3.2 FAR-Legos Description

The basic idea of our proposal is to take advantage of the abundant energy and larger transmission range of actuator nodes. We propose to use the adjustable transmit power to assign ranks which serve as a pseudo-geographical information for structuring the network area covered by the actuator. These ranks are used to direct data packets back to actuator nodes. The four phases of FAR-Legos are:

3.2.1 Rank Assignment Phase

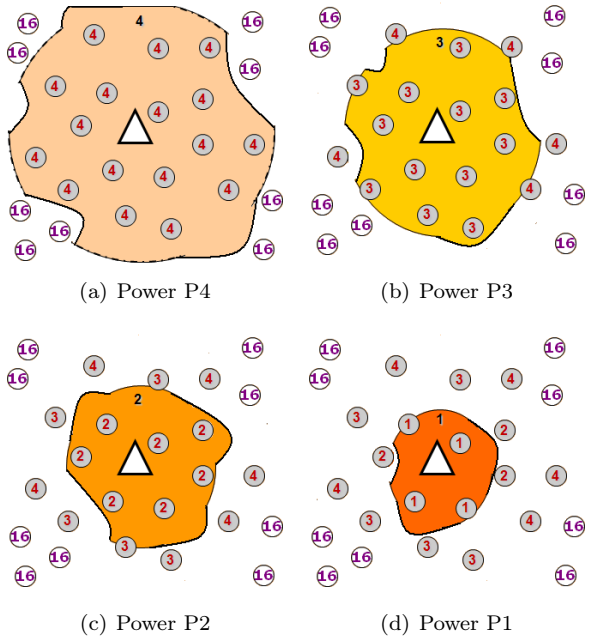
We propose that each actuator node assigns ranks to sensor nodes within its transmission range by broadcasting `Rank Assignment` messages of decreasing transmission power. Thus sensor nodes nearest the actuator node will have a smaller rank.

At each step, the actuator assigns a rank to nodes in a ring around it. The thickness of this ring is equal to the radio range of sensor nodes. Thus a sensor belonging to the ring N can communicate with sensors belonging to the (N-1) and (N+1) rings. In addition, we used a decremented transmission power because it is more energy efficient than incremented one. Indeed, when the nodes which are farther from the actuator has received a rank, they can switch to energy conservation mode (turn off their radio for example).

In the example in Figure 3, the actuator begins advertising rank 4 with its maximum power P4. A sensor receiving this message will assign itself this rank value (Figure 3(a)). Then the actuator node broadcasts a second message with a power $P3 < P4$. The sensors that do hear this message and that currently have a higher rank will update their rank to 3 (Figure 3(b)). This process is repeated until the actuator reaches its minimum transmission power P1 (Figure 3(d)). P1 is set to the transmission power of sensor nodes in the network. Sensor nodes that are not reached by the actuator keep their default rank value (16 in this example).

3.2.2 Neighbor Discovery Phase

After the phase of rank assignment, each sensor node sends a `Hello` message to discover its neighborhood. This phase is started by the actuator nodes. Each actuator sends a `Hello` message using its minimum power transmission (which is equal to the power transmission of sensors nodes). In this discovery message, each node puts its ID and its rank (which is equal to 0 for actuator nodes and



(a) Power P4 (b) Power P3

(c) Power P2 (d) Power P1

Figure 3: Example of allocation of 4 ranks around an actuator node. The nodes that are not covered by the actuator keep the default rank value, equal to 16 in this example

default value for non-covered sensors nodes). At the end of this phase, each node will have a list of its neighbors and their ranks. Two cases arise. First, in the covered area, each sensor node can classify its neighbors into three types: sensors closer than self to the actuator (having a smaller rank than self), sensor further out from the actuator than self (having a higher rank) and sensor in the same ring (having the same rank). Second, in the non-covered area, sensors that detect nodes with a non-default rank conclude that they are connected to a covered area. They memorize that fact and will use it to be the first nodes which launch the third phase detailed hereunder.

3.2.3 Filling The Voids Phase

In this phase, a logic structure is built in the area that is not covered by the actuators. This set of sensor nodes start the construction of the structure at $t = T_{start\_Legos}$ calculated as in equation (1). As described in (1), each sensor node waits for a $Timeout_{neighbor\_discovery}$ which corresponds to the time during which a sensor node can receive `Hello` messages from its neighborhood nodes. Then, it waits for a random time calculated with the function $Rand(X)$, where X here refers to the situation of the node (near the covered area or not). This function returns a random waiting time depending on the situation (X) of a node. Hence, nodes near the covered area have a chance to start the construction of the structure first. The other non-covered nodes start the construction and join the structure later. We introduce this random time for two reasons: first, to introduce a priority between nodes constructing the structure and those

joining at as leaf-nodes. Second, to reduce packet collisions and to avoid that many nodes try to start the construction of the structure at the same time. Finally, each non-covered node should wait for a $Timeout_{Leader\_detection}$ which correspond to the period separating two leader broadcast messages in Legos structure.

$$\begin{aligned} T_{start\_Legos} =& Timeout_{neighbor\_discovery} \\ &+ Rand(X) \\ &+ Timeout_{Leader\_detection} \end{aligned} \tag{1}$$

3.2.4 Data-Collection Phase

Each sensor having data to send to the actuator node will apply this data-collection algorithm. This phase depends on the status of the sensor node (i.e. whether in a covered or a non-covered area).
In a covered area, a sensor node that has data to send to the actuator picks the next hop from its neighbors list: if the destination (the actuator) is not a direct neighbor, the next hop will be a neighbor having a smaller rank than self. Hence, the data messages will run down the rank gradient through the covered area until they reach the actuator. If the source node does not have a neighbor with a smaller rank, it understands that there is a void in the network and forwards its data packet to a neighbor having the same rank or, at worst, to a node having a higher rank, in an attempt at routing around voids. Using neighbors having a same or a higher rank helps increasing the delivery ratio [12]. However, this also creates a risk of loops in the data-collection route. Hence, when a sensor node sends a message back to its sender, it remembers the ID of this message (sequence number). Thus when a node receives a message, it first checks if this message has to go through itself. If so, it refuses to participate in the delivery of this message and explicitly requests the sender to find another node to be next hop.
In the non-covered area, a sensor having a data destined to an actuator first verifies if it is directly connected to a covered area: if it is, it sends the data to the nearest covered node regardless of its role in the Legos structure. If it is not, a node that is a member or gateway node sends its data to its Leader node. A Leader node, either having a data from itself or receiving one from its neighborhood nodes, first checks if it has a member or a gateway node that is connected to a covered area. If it has such one, it sends the data packet to that node which will forward it to a covered area. Else, it will choose a leader to which it will forward the data, which will then travel along the Legos backbone until a Leader finds a connection to a covered area, and thereby a path to an actuator.

3.3 Presence of many actuators in the network

When there are many actuators on the network, each actuator randomly starts the first phase of rank assignment. Each sensor node in the covered areas saves the rank assigned by each actuator heard. In the discovery phase, each of the sensor nodes sends in the `Hello` messages the list of the actuators that it heard. Hence, when a node has a data to send, it looks at its list of actuators and

picks the destination actuator: the destination can simply be the actuator with which it has the smallest rank or a specific actuator for a given application data. The algorithm for the data-collection phase remains as described previously. It should be noted that in non-covered areas, nodes don't have a list of actuator nodes, so the choice of the destination is made by the first sensor node in a covered area.

3.4 Deployment of new nodes in the network

Our proposal assumes that the network is stable and static, but we can provide mechanisms to take into account the dynamics caused by deployment of new sensors or actuators in the network. When a new actuator node is deployed in the network, it begins by inviting the sensor nodes in its neighborhood to a rank assignment phase. Thus, the new actuator broadcasts its `Rank Assignment` messages to its neighborhood. Hence, the sensor nodes create a new entry in their list of actuators. Conversely, when a new sensor node is deployed in the network, it first discovers its neighborhood by sending a `Hello` packet. Nodes already present in the network and receiving this `Hello` message answer with their rank. Three cases arise: First, when this newly deployed node only detects nodes in a non-covered area, it joins the Legos structure as described above and keeps the default value for its rank. Second case, when this new node only detects covered nodes (having a rank different from the default one), it simply calculates its rank, either as the average or the maximum of its neighbor ranks. After calculating its rank, the newly deployed node sends a second `Hello` message to inform its neighbors of its new rank. Neighbors add this sensor as a new entry in their neighborhood table. Thus, this node is integrated into the structure of the covered area without affecting it. Finally, in the case where the new node detects both covered and non-covered nodes, it saves the list of the covered nodes and their ranks. Then it starts joining the Legos structure in the non-covered area by announcing to the Legos structure its border situation, while storing this information. We chose that this new node should join the structure of the non-covered area in order to increase the number of nodes that can reach the covered area.

3.5 Performance evaluation

In this section, we describe the parameters we used in simulation to evaluate the performance of our proposal. Then we present the main simulation results.

3.5.1 Simulation Parameters

We assume a network in the form of a grid of 120m * 120m. We deploy 169 nodes in the network. For single actuator networks, we assume that the actuator node is placed at the center of the area studied. We trigger a random number of events in the network; each of those events will be detected by a sensor node which will report it to the actuator node. Table 1 summarizes the main characteristics of the network.

Parameter	Value
Topology	Grid
Sensor Nodes Cardinality	169
Event Number	1 .. 19
Network Average Degree	8 Nodes
PHY Layer	Ideal
Propagation Model	Unit Disk Graph
MAC Protocol	802.11 DCF
Hop Limit	16 hops
Confidence Interval	95%
Simulator	WSNet [13]

Table 1: Simulation Parameters

3.5.2 Evaluation of the energy consumption of the rank assignment phase

We start the performance evaluation of our proposal by studying the energy consumption of the rank assignment phase. We compare our proposal based on successively decreasing transmission powers to a prior technique based on MPR [9]. We assume a uniform deployment with a predefined geographical density. We also assume that the sensor transmission energy cost is equal to that of reception.

We have chosen to look at the energy consumed in transmission and reception, and to not take into account the energy consumed by passive listening. Intuitively, we can deduce that the energy consumed in passive listening by our proposal is lower than that consumed by MPR. Indeed, with FAR-Legos, sensor nodes that are far from the actuator node are reached quicker by FAR-Legos `Rank Assignment` messages than in the case of MPR. Indeed, in the MPR, before moving from one level to another, the discovery phase neighborhood extends the period of passive listening of farthest nodes.

We chose to compare our FAR-Legos proposal with two variants of MPR protocol with and without `Hello` packet. Indeed, in MPR, nodes have to exchange `Hello` messages to select those who will relay the broadcast messages. We have chosen, in the second variant of MPR, to ignore the cost of these packets because in our FAR-Legos proposal, we will also use the `Hello` packets during the discovery phase of the neighborhood (as described previously).

Figure 4 shows, for a constant geographical node density of 8 neighbors per node, the total amount of energy consumed in the network as the number of ranks is varied (the actuator reaches out further, assigning more rings of constant thickness to an expanding space). First, it is clear in Figure 4 that our proposal consumes less power than MPR. This is due to the fact that nodes with FAR-Legos proposal receive only the `Rank Assignment` messages broadcasted by the actuator node. By contrast, with MPR, each node receives `Hello` messages from all its neighborhood and a rank assignment from the relay node and sends a `Hello` message and a rank assignment if it is a relay node. Second, we note also in Figure 4 that, even without considering the cost of the `Hello` message with MPR and for a small number of ranks assigned around an actuator node, our FAR-Legos proposal consumes less power. For a large number of ranks, the

energy consumption in FAR-Legos is larger than that of MPR without `Hello` message. Indeed, to assign a rank N, FAR-Legos implies the reception of this `Rank Assignment` message by all nodes of the lower ranks. While for MPR, the assignment of a new rank N only generates a reception of this message at the nodes of rank (N-1).

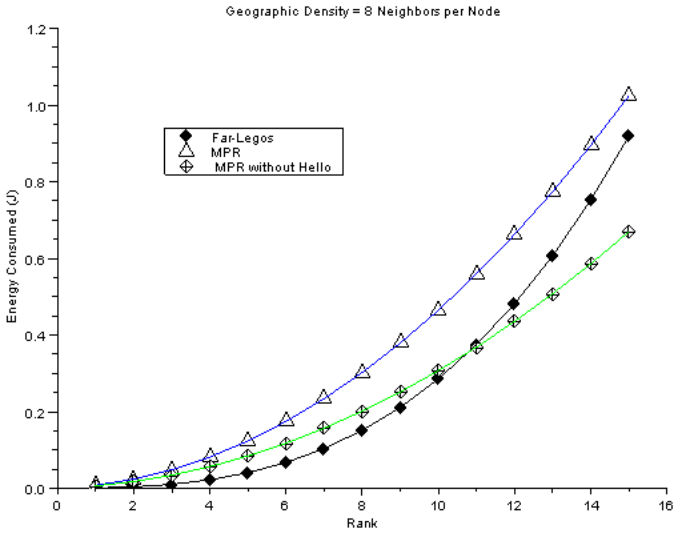


Figure 4: Comparison of energy consumption of the rank assignment phase for FAR-Legos vs MPR

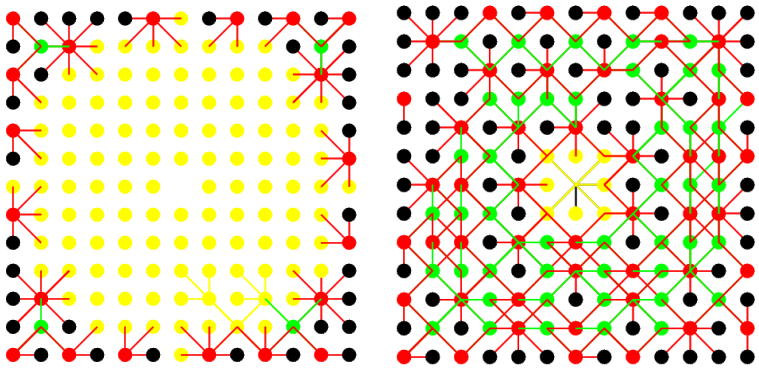
3.5.3 End-to-end delay and delivery ratio evaluation

In this section, we are interested in a grid topology with a single actuator placed at the center of a fixed-sized, constant density network. We vary the size of the non-covered area in the network by varying the range of the actuator node. We are interested in the end-to-end delay and the delivery ratio when we increase the number of events detected by sensor nodes in the network. We compare the performance of our FAR-Legos proposal with the performance obtained when using BBDD protocol [10].

We define the end-to-end delay as the average time required by a message originating at a sensor node to reach the final destination (the actuator node).

Figure 5(a) and Figure 5(b) represent the topologies obtained with our FAR-Legos proposal for a small non-covered area and a large non-covered area, respectively.

First, for a small non-covered area, we note that our hybrid FAR-Legos proposal offers a better end-to-end delay than BBDD. This is because the messages go through fewer hops when they travel through the covered area (Figure 6). Second, for a large non-covered area, we compute the delivery ratio of FAR-Legos and BBDD. In Figure 7, we represent the delivery ratio when the number of the event detected in the network is increased. We note that, when applying our proposal, the delivery ratio decreases. Because all sensor nodes have the same rank value in the non-covered area, the path is extended forcing the nodes



(a) Small non-covered area (b) Large non-covered area

Figure 5: Topologies obtained with FAR-Legos proposal: yellow nodes are in the covered area, red nodes are Leader, green nodes are Gateways and black nodes are Member.

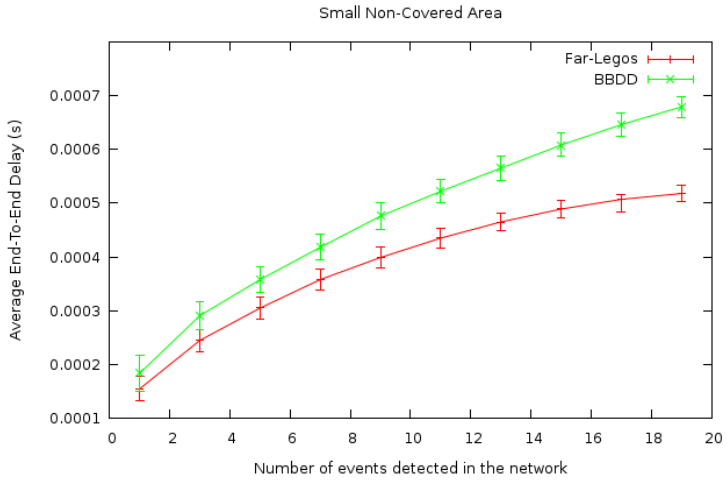


Figure 6: Comparison of end-to-end delay for FAR-Legos and BBDD in a small non-covered area

to drop messages when reaching the maximum allowed hop count. But it is important to note that the delivery ratio of our proposal FAR-Legos remains high, around 80%.

We have therefore verified that the hybrid FAR-Legos proposal offers a minimum end-to-end delay and a high delivery ratio in both small and large non-covered area.

3.5.4 Comparison of the Number of hops performed

In this section, we compare the average number of hops a packet needs with FAR-Legos and BBDD. In Figure 8, we compare the resulting average number of hops with each proposal to the optimal number of hops, in the case of a

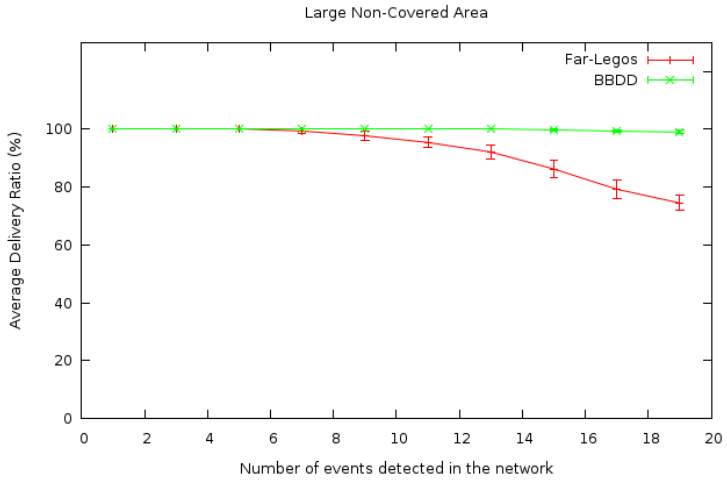


Figure 7: Comparison of delivery ratio for FAR-Legos and BBDD in a large non-covered area

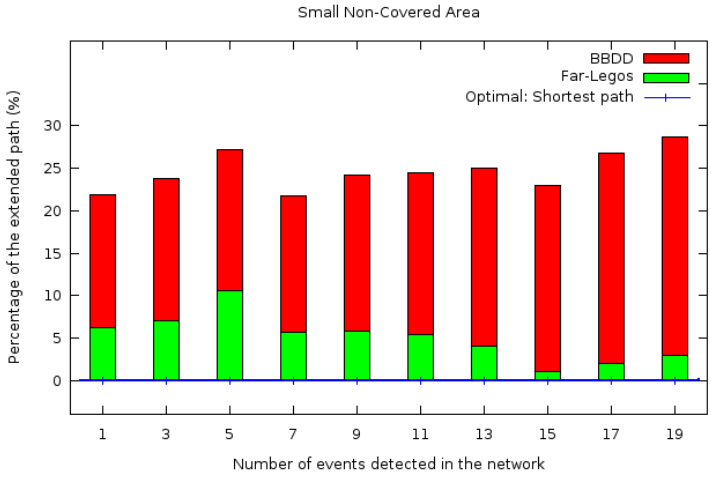


Figure 8: Compared path stretch between FAR-Legos and BBDD in a small non-covered area

network with a small non-covered area. The optimal number of hops correspond to the number of hops with the shortest path algorithm.

The number of hops performed by FAR-Legos is in all cases less than that done by BBDD. This is due to the fact that, in the covered area, FAR-Legos offer less hops than BBDD.

In fact, in the covered area, the number of hops done by FAR-Legos is optimal: indeed, since there is no void and the radio range of sensor nodes is equal to the rings thickness around the actuator, each sensor in the covered area has at least one neighbor with a lower rank. So, in the covered area, the message makes only one hop per rank until it reaches the destination (the actuator node).

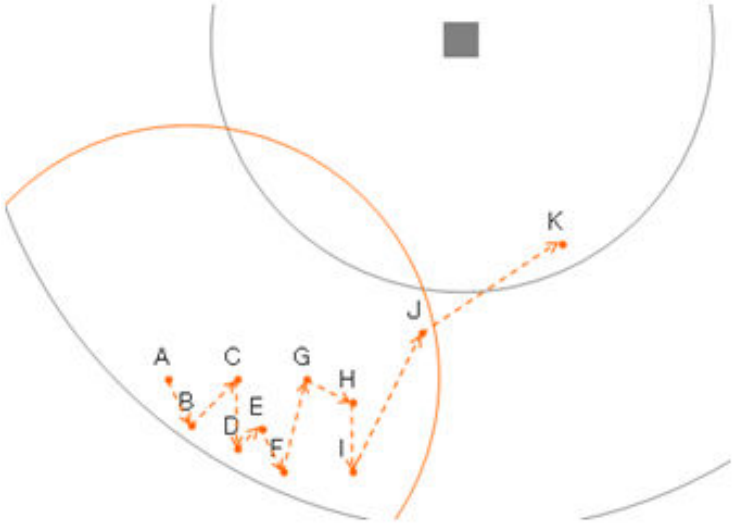


Figure 9: Example of blind/random intra-layer forwarding

Whereas with BBDD, each member node has to send its data packet to its leader even if it can directly communicate with the final destination. So by sending data packets through the BBDD backbone in the covered area, BBDD needs more hops to reach the actuator compared to FAR-Legos .

When a data packet needs less hops to reach the destination, this also means we use less energy at the nodes in the network and so we increase the lifetime of the network.

3.6 Discussions and optimizations for basic FAR-Legos

Considering the data-collection phase, the path stretch may be important due to the blind selection of the next hop in the case of intra-layer forwarding. Indeed, if the routing process appears to be *oriented* for inter-layer relaying because the next hop is necessarily a node with a lower rank, the routing process for intra-layer is not optimized and the next hop is a node of the neighborhood and in the same layer: this random selection appears to be blind. A comprehensive overview of this problem is depicted in Figure 9.

We assume that there are 11 nodes in the network. As shown is the figure 9, nodes A, B, C, D, E, F, G, H, I, J have the same rank value of 2 whereas the rank value of node K equals to 1, and only node J is a neighbor of a node in the next layer (node K, rank value equals to 1). When node A has a message to send to the actuator, because A does not have any neighbor in the next layer, it should forward the packet using intra-layer process: from its point of view it is like a local hole. To bypass this situation, A forwards the packet to a non-determined neighbor which relays the packet and so on. In the worst case, 9 hops are needed to reach node J which has a neighbor with smaller rank value although node J is a neighbor of node A. In the next section, we propose to local optimizations to improve the behavior of FAR-LEGOS in this kind of situation.

4 Solution for blind data collection phase: Oriented-FAR and Clustreded-FAR

In this section, we present two variants of FAR-Legos to overcome the problem of blind data-collection described in the previously: Oriented-FAR is discussed in section 4.1 and Clustered-FAR is presented in section 4.2.

4.1 Oriented-FAR

Ranks affection used in FAR-Legos is a static process: ranks value do not evolve despite the location of the nodes in the layer and the evolution of the network. Nodes know their rank according to the `Rank Assignment` messages broadcasted by the actuator. The basic idea of oriented-Far is to adapt the rank value using local information, *i.e.* to take into account the knowledge of lower and/or upper layer(s) given by the neighborhood.

4.1.1 Solution

We propose to adapt the ranks of the sensor nodes in the network and to assign new roles to the nodes belonging to the same layer. Thus, during the data collection phase, to send data message will be oriented and will no longer be blindly. In this proposal a node can have three states:

- Near_Next: the sensor node detects at least another node in the next layer (having a smaller rank than itself)
- Near_Previous: the sensor node detects at least another node in the previous layer (having a larger rank than itself)
- Near_Center: all the nodes in the neighborhood have the same rank value.

4.1.2 Detailed description of Oriented-FAR algorithm

Because Oriented-FAR is based on FAR, the operation of Oriented-FAR can be also divided into four phases as we summarized previously: rank assignment phase, neighbor discovery phase, structure construction phase and data-collection phase. The aim of Oriented-FAR is to decrease the probability of blind relay in the third phase by involving a dynamic assignment rank. In Oriented-FAR, the second phase (neighbor discovery) is modified to adapt dynamically the rank values. The other three phases remain unchanged.

The new neighbor discovery phase can be divided into three sub-phases:

- Neighbor discovery: As it was described previously (section 3.2.2), each sensor node sends a `Hello` message to discover its neighbors. In this `Hello` message each node puts its identification ID and its rank. By receiving this `Hello` message, each sensor node put this information in its neighbor table.
- New rank calculation: After collecting information about its neighborhood, each sensor node calculates a new rank based on the information collected in the previous sub-phase. The new rank will be calculated as

in the equation 2. In 2, α represents the number of sensors having a rank larger then itself and β represents the number of sensor nodes belonging to the next layer. Note that α and β should not exceed 9, to conserve the gradient between layers.

$$New\_Rank = Rank\_Assigned + \alpha * 0.1 - \beta * 0.1 \quad (2)$$

- New rank announcement: After calculating the new rank value, each sensor node verify if the new rank is different from the rank assigned in the first phase. If it is so, it announces this new rank to its neighborhood. By receiving this new announcement, each node updates the rank if necessary.

The last two phases (structure construction in the non-covered area and the data collection phase) are not modified. During the data-collection, each sensor source node sends its data to the neighbor having the lowest rank in its neighbor table. Hence the data collection phase will be oriented: a data message will not be send in blind in the same layer. It will sent to nodes which have a neighbor with a smaller rank so a node nearest to the actuator node. Performance evaluation of Oriented-FAR will be discussed in 4.3.

4.2 Clustered-FAR

To minimize the path stretch between a source node and the actuator, the forwarders selection is a key issue. For inter-layer routing process, we proposed to select a node with a lower rank value, it means in the next layer. For intra-layer routing and according to the discussion in section 3.6, we need to choose a forwarder in the same layer which is as close as possible to the next layer. Intuitively, such forwarder is a node which have the most important number of neighbors which are in the next layer. Clearly, we propose to build local clusters in each layer, where the clusterhead is always the node which is the more closer to the next layer.

4.2.1 Proposition

We propose to distinguish roles of nodes with same rank value is a feasible way to solve the problem of blind data-collection phase into a layer. We introduce:

- Upstream node: node that has at least one neighbor with smaller rank value.
- Downstream node: node that has at least one neighbor with bigger rank value.
- Ordinary node: node that has only neighbors with same rank value.

All upstream nodes may be clusterhead where the cluster contains only both downstream nodes and ordinary nodes of the same layer. An example is given in figure 10. In this case, nodes are classed into 3 roles:

- Upstream node in lay 2: node J
- Downstream node in lay 2: node D

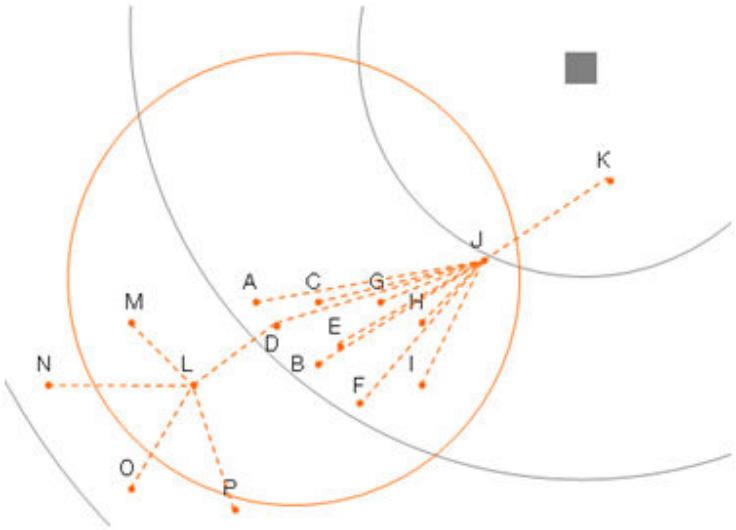


Figure 10: Example of clustered intra-layer transmission with Clustered-FAR

- Upstream node in lay 3: node L

Using this simple, but efficient, mechanism, the path stretch will be reduced.

4.2.2 Detailed description of Clustered-FAR algorithm

The algorithm of Clustered-FAR takes benefit from the different phases of FAR-Legos: the construction process is also divided into four phases as for FAR-Legos: rank assignment phase, neighbor discovery phase, structure construction phase and data-collection phase (cf. section 3.2). The basic idea is to decrease the probability of blind relay in data-collection phase by using a cluster structure provided in construction phase. Clustered-FAR should be more efficient in data-collection phase but as price for that: new algorithm needs more messages, more energy and time to form cluster in structure construction phase. Clustered-FAR does not require modification in the first two phases (rank assignment phase, neighbor discovery phase). After the first phase, each node has 1-hop neighborhood information, and then, structure construction phase is processed both in covered area and non-covered one. Note that in the original proposition of FAR, structure construction was dedicated only for non-covered area.

In the case of covered area, after neighbor discovery phase, each upstream node waits for a timeout before starting to broadcast it clusterhead (CH) status in its neighborhood. This timeout is used to be sure that the nodes nearest the actuator start first the construction of the cluster structure. Ordinary nodes which received this CH status will be member nodes attached to this clusterhead. In the data collection phase each node, according to its role, will sends the data to its clusterhead if it is an ordinary node, or to a node belonging to the next layer in the direction of the actuator.

4.3 Performance evaluation of Oriented-FAR and Clustered-FAR

In this section, we describe the parameters we used in simulation to evaluate the performance of Oriented-FAR and Clustered-FAR proposals. Then we present the main simulation results.

4.3.1 Simulation parameters

We consider a wireless sensor network with 169 nodes with a network grid topology. Without losing generalization, we assume the actuator is located in the center of the sensing region and all the nodes in this filed can be covered by the actuator. Since the modifications done on Oriented-FAR and Clustered-FAR are only suitable for the covered area, we suppose in this section that the actuator can cover the whole network. The layer thickness equals to three times the maximum transmission range of sensors. We trigger a random number of events in the network; each of those events will be detected by a sensor node which will report it to the actuator node. Table 2 summarizes the main characteristics of the network.

Parameter	Value
Topology	Grid
Layer thickness	3 * power transmission of sensor node
Sensor Nodes Cardinality	169
Event Number	1 .. 90
Network Average Degree	8
PHY Layer	Ideal
Propagation Model	Unit Disk Graph
MAC Protocol	CSMA/CA
Hop Count Limit	16 hops
Confidence Interval	95%
Simulator	WSNet [13]

Table 2: Simulation Parameters

4.3.2 Delivery ratio

In Figure 11, we represent the delivery ratio when the number of the event detected in the network increases. Note that the delivery ratio of FAR-Legos proposal decreases when the number of event increases, while the delivery ratio of Oriented-FAR and Clustered-FAR are close to 100%. This is due to the FAR-Legos blind forwarding: the path lentght increases and then, data message will be dropped once it reaches the maximal hop count.

4.3.3 End-to-end delay

In this section we compare the end-to-end delay of the two variants Oriented-FAR and Clustered-FAR. Figure 12 represents the average end-to-end delay when the number of event increases in the network: the two variants exhibit almost the same end-to-end delay.

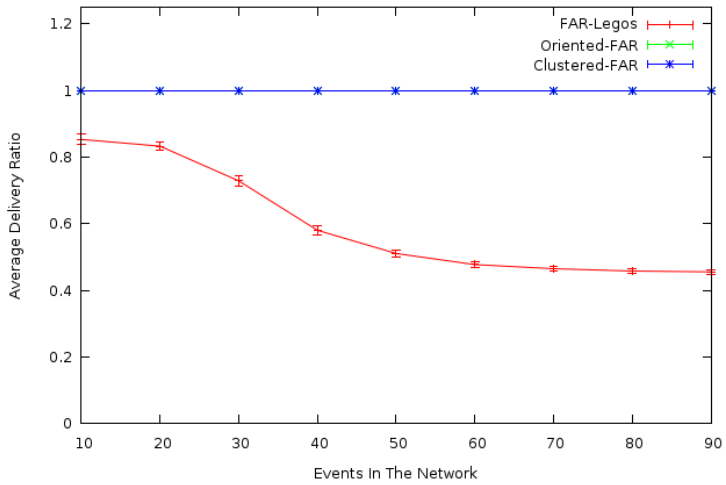


Figure 11: Comparison of delivery ratio for FAR-Legos, Oriented-Far and Clustered-FAR

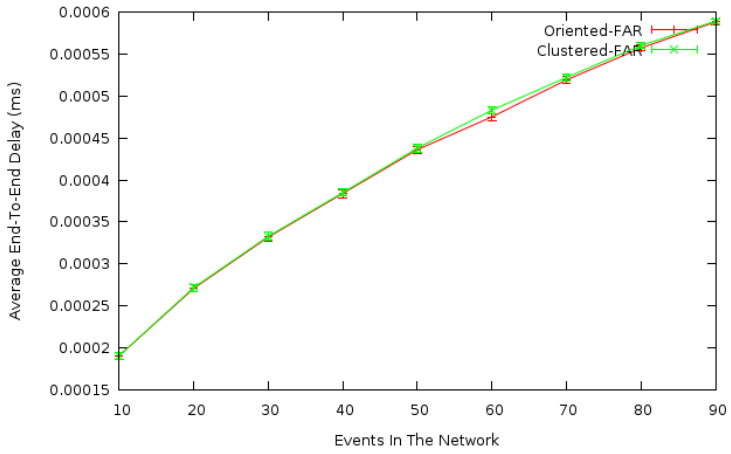


Figure 12: Comparison of end-to-end delay for Oriented-Far and Clustered-Far

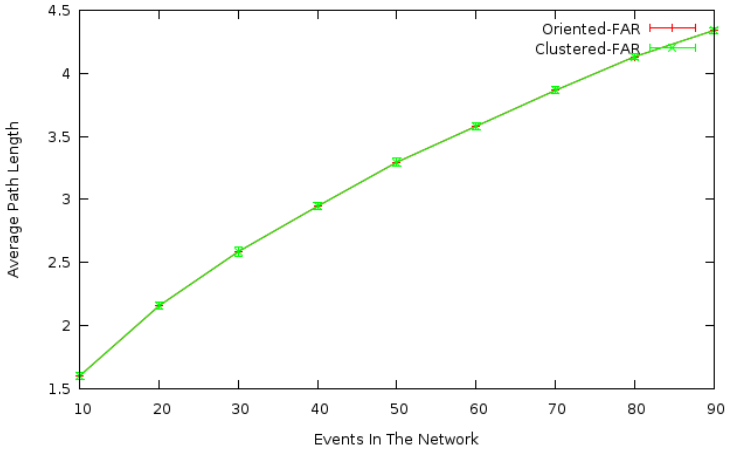


Figure 13: Comparison of hop number count for Oriented-Far and Clustered-Far

4.3.4 Hop number count

In this section we compare the hop number count for the two variants Oriented-FAR and Clustered-FAR. Figure 13 represents the average number of hops when the number of event increase in the network. One more time, the two variants exhibit a very close behavior because in both Oriented-FAR and Clustered-FAR, the next forwarder is chosen as close as possible to the next layer.

4.3.5 Overhead

Here we compare the number of control messages used by the two variants Oriented-FAR and Clustered-FAR. Figure 14 represents the overhead performed by Oriented-FAR and Clustered-FAR when the number of event increases in the network. As it is depicted in Figure 14, the number of control packets sent by the two optimizations of FAR-Legos decreases according to the number of events in the network. The reason is because these control messages are used only once a node change its rank (for Oriented-Far) or it become a clusterhead (for Clustered-Far). In Figure 14, the overhead of Clustered-Far is less than Oriented-Far, because in Clustered-Far only clusterhead nodes send a `CH Notification` message. Whereas with Oriented-Far, every node in the network sends its new rank after the new rank calculation phase.

4.3.6 Energy Consumption

The energy consumption is illustrated in Figure 15. We compute here the total energy consumption by all the sensor nodes in of the network. Figure 15(a) (resp. 15(b)) represents the energy consumption of sensors nodes in the network when using Oriented-Far proposal (resp. Clustered-Far proposal).

As we can see in both figures, the energy consumption map of the two proposal is the same in two cases. Sensor nodes near the actuator node (in the center of the network) consume more energy than far nodes, because they relay more messages to the actuator node.

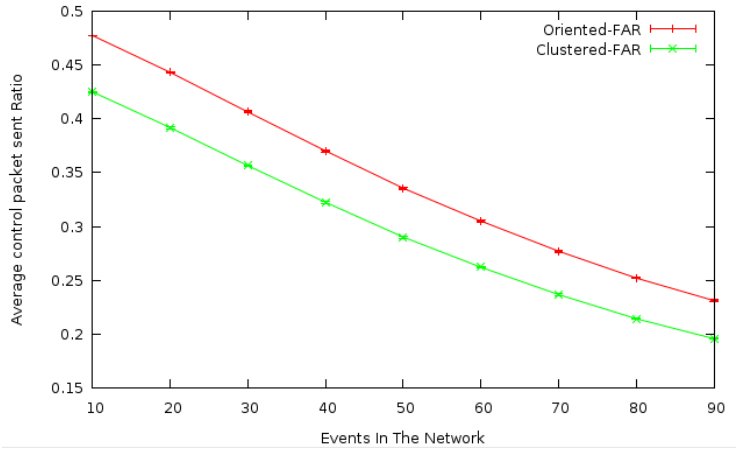
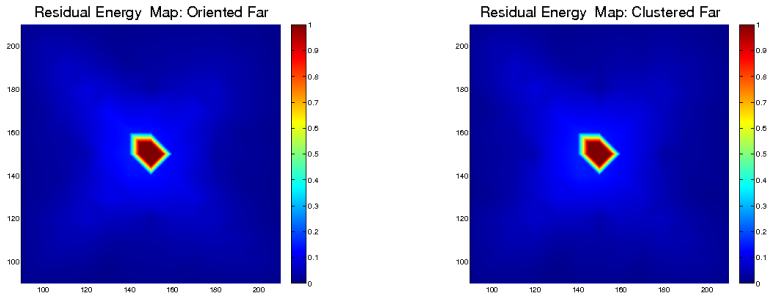


Figure 14: Comparison of overhead of Oriented-Far and Clustered-Far



(a) Oriented-Far (b) Clustered-Far

Figure 15: Consumption Energy Map: (a) Oriented-Far (b) Clustered-Far

5 Conclusion

We proposed in this report a hybrid self-organizing data-collection protocol suitable for a wireless sensors and actuator network. Our proposal is based on two self-organizing protocols in order to provide energy efficiency, low end-to-end delay and high delivery ratio while taking advantage of the resource available on the actuators nodes. We have shown through simulations that our proposal offers an end-to-end delay as good as that of other published algorithms and a high delivery ratio in both large and small non-covered area in a stable network. Then we presented two variants of our proposal to remind with the problem of blind forwarding in the data collection phase: Clustered-Far and Oriented-Far. The two variants present improved performances. The two variants lead to close performances in end-to-end delay, the average delivery ratio and the number of hop count performed. Clustered-Far provides an overhead lower the the overhead performed by Oriented-Far.

References

[1] I. F. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci. Wireless sensor networks: a survey. *Comput. Netw.*, 38(4):393–422, 2002.

[2] F. Martincic and L. Schwiebert. *CHAPTER1 Introduction to Wireless Sensor Networking*. Wiley Series on Parallel and Distributed Computing, 2005.

[3] D. Culler, D. Estrin, and M. Srivastava. Guest editors' introduction: Overview of sensor networks. *Computer*, 37(8):41–49, 2004.

[4] I. F. Akyildiz and I. H. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks*, 2(4):351–367, 2004.

[5] R. Verdone, D. Dardari, G. Mazzini, and A. Conti. *Wireless Sensor and Actuator Networks: Technologies, Analysis and Design*. Academic Press, 2008.

[6] P. Levis, A. Tavakoli, and S. Dawson-Haggerty. Overview of existing routing protocols for low power and lossy networks. In *draft ietf roll protocols survey -07-*, 2009.

[7] T. Melodia, D. Pompili, V. C. Gungor, and I. F. Akyildiz. Communication and coordination in wireless sensor and actor networks. *IEEE Transactions on Mobile Computing*, 6(10):1116–1129, 2007.

[8] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *MobiHoc'02: Proceedings of the 3rd ACM international symposium on Mobile ad hoc networking & computing*, pages 194–205, New York, NY, USA, 2002. ACM.

[9] A. Qayyum, L. Viennot, and A. Laouiti. Multipoint relaying for flooding broadcast messages in mobile wireless networks. In *HICSS'02: Proceedings of the 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9*, page 298, Washington, DC, USA, 2002. IEEE Computer Society.

[10] J. Lu and F. Valois. On the data dissemination in wsns. In *WiMob'07: Proceedings of the 3rd Int'l Conference on Wireless and Mobile Computing, Networking and Communications*, White Plains, USA, 2007.

[11] J. Lu, F. Valois, D. Barthel, and M. Dohler. Low-energy self-organization sceme for wireless ad hoc sensor networks. In *WonS'07: Proceedings Of the 4th Annual Conference on Wireless On demand Network Systems and Services*, Hongkong, China, 2007.

[12] Q. Lampin, D. Barthel, and F. Valois. Efficient route redundancy in dag-based wireless sensor networks. In *WCNC'10: IEEE Wireless Communications & Networking Conference*, Sydney, Australia, 2010.

[13] E. Ben Hamida, G. Chelius, and J. M. Gorce. Scalable versus accurate physical layer modeling in wireless network simulations. In *PADS'08: Proceedings of the 22nd Workshop on Principles of Advanced and Distributed*

Simulation, pages 127–134, Washington, DC, USA, 2008. IEEE Computer Society.

Centre de recherche INRIA Grenoble – Rhône-Alpes
655, avenue de l'Europe - 38334 Montbonnot Saint-Ismier (France)

Centre de recherche INRIA Bordeaux – Sud Ouest : Domaine Universitaire - 351, cours de la Libération - 33405 Talence Cedex
Centre de recherche INRIA Lille – Nord Europe : Parc Scientifique de la Haute Borne - 40, avenue Halley - 59650 Villeneuve d'Ascq
Centre de recherche INRIA Nancy – Grand Est : LORIA, Technopôle de Nancy-Brabois - Campus scientifique
615, rue du Jardin Botanique - BP 101 - 54602 Villers-lès-Nancy Cedex
Centre de recherche INRIA Paris – Rocquencourt : Domaine de Voluceau - Rocquencourt - BP 105 - 78153 Le Chesnay Cedex
Centre de recherche INRIA Rennes – Bretagne Atlantique : IRISA, Campus universitaire de Beaulieu - 35042 Rennes Cedex
Centre de recherche INRIA Saclay – Île-de-France : Parc Orsay Université - ZAC des Vignes : 4, rue Jacques Monod - 91893 Orsay Cedex
Centre de recherche INRIA Sophia Antipolis – Méditerranée : 2004, route des Lucioles - BP 93 - 06902 Sophia Antipolis Cedex

Éditeur
INRIA - Domaine de Voluceau - Rocquencourt, BP 105 - 78153 Le Chesnay Cedex (France)
http://www.inria.fr
ISSN 0249-6399