Fault-Tolerant and Energy-Efficient Generic Clustering Protocol for Heterogeneous WSNs

Mandicou Ba, Olivier Flauzac, Rafik Makhloufi, and Florent Nolot
Université de Reims Champagne-Ardenne, France
CReSTIC - SysCom EA 3804
{mandicou.ba, olivier.flauzac, rafik.makhloufi, florent.nolot}@univ-reims.fr

Ibrahima Niang
Université Cheikh Anta Diop, Sénégal
Laboratoire d'Informatique de Dakar (LID)
iniang@ucad.sn

***Abstract*—In the context of Wireless Sensor Networks (WSNs), where sensors have limited energy power, it is necessary to carefully manage this scarce resource by saving communications. Clustering is considered as an effective scheme to increase the scalability and lifetime of wireless sensor networks. Moreover, failures and topological changes are inevitable in sensor networks due to the inhospitable environment, unattended deployment or nodes mobility. Therefore, one of the wanted properties of WSNs is the fault tolerance and adaptivity to topological changes. We propose a fault-tolerant and energy-efficient distributed self-stabilizing clustering protocol based on message-passing for heterogeneous wireless sensor networks. This protocol is adapted to topological changes, optimizes energy consumption and prolongs the network lifetime by minimizing the number of messages involved in the construction of clusters. Our generic clustering protocol can be easily used for constructing clusters according to multiple criteria in the election of cluster-heads, such as nodes' identity, residual energy or degree. We propose to validate our approach under the different election metrics by evaluating its communication cost in terms of messages, energy consumption and number of clusters. Simulation results show that, in terms of of messages, energy consumption and clusters distribution, it is better to use the Highest-ID metric for electing CHs. Furthermore, after faults occurrence, the re-clustering cost is minimal compared to the clustering cost.**

***Keywords*-Self-stabilizing clustering; Wireless Sensor Networks; Energy-efficient; Fault-tolerant; OMNeT++ simulator.**

I. Introduction

A preliminary version of this paper, entitled "Evaluation Study of Self-Stabilizing Cluster-Head Election Criteria in WSNs", is published in CTRQ'2013 [1]. In this paper, we include fault-tolerance and energy-efficiency mechanisms in the context of heterogeneous Wireless Sensors Networks (WNSs) with energy constraint. To the best of our knowledge, there is no paper in the literature where the solutions are fault-tolerant, energy-aware, self-stabilizing and where the same proposed approach is compared in the case of different CH election methods.

Due to their properties and wide applications, WSNs have been gaining growing interest in the last decades. These networks are used in various domains like: medical, scientific, environmental, military, security, agricultural, smart homes, etc. [2].

In WSN, sensors have very limited energy resources due to their small size. This battery power is consumed by three operations: data sensing, communication, and processing. Communication by messages is the activity that needs the most important quantity of energy, while power required by CPU is minimal. For example, Pottie and Kaiser [3] show that the energy cost of transmitting a 1KB message over a distance of 100 meters is approximately equivalent to the execution of 3 million CPU instructions by a 100 MIPS/W processor. Thus, conserving communication power is more important in WSNs than optimizing processing. Consequently, to extend the sensor network lifetime, it is very important to carefully manage the very scarce battery power of sensors by limiting communications. This can be done through notably efficient routing protocols that optimize energy consumption. Many previous studies (e.g., Yu et *al.* [4] and Younis and Fahmy [5]) proved that clustering is an effective scheme in increasing the scalability and lifetime of wireless sensor networks. Clustering consists in partitioning the network into groups called clusters, thus giving a hierarchical structure [6].

On the other hand, nodes in WSNs are prone to be failure due to energy depletion, hardware failure, communication link errors, malicious attack, and so on. Fault tolerance is one of the critical issues in WSNs as proved in many studies like Liu et *at.* [7], Zhang et *al.* [8] and Hao et *al.* [9]. Fault tolerance is defined as the ability of a system to deliver a desired level of functionality in the presence of faults [10]. Therefore, one of the most wanted properties of WSNs is the fault tolerance and adaptivity to topological changes, which consist of the system's ability to react to faults and perturbations. Self-stabilization is an approach to design fault-tolerant and adaptive to topological changes distributed systems [11].

Several self-stabilization clustering approaches are proposed in the literature and used, for example, in the case of a WSN for routing collected information to a base station. However, most of them are based on state model, so they are not realistic compared to message-passing based clustering ones. Moreover, approaches in the last category are not self-stabilizing and they are generally highly costly in terms of messages; while in the case of WSNs, clustering aims at optimizing communications and energy consumption.

In this paper, we propose a fault-tolerant and energy-efficient distributed self-stabilizing clustering protocol based on message-passing for heterogeneous wireless sensor networks. The proposed algorithm is based only on information from neighboring nodes at distance 1 to build k-hops

clusters. It optimizes energy consumption and then prolongs the network lifetime by minimizing the number of messages involved in the construction of clusters. Our clustering protocol offers an optimized structure for routing. It can be easily used for constructing clusters according to multiple criteria in the election of cluster-heads such as: nodes' identity, residual energy, degree or a combination of these criteria. We propose to validate our approach by evaluating its communication cost in terms of messages, energy consumption and percentage of formed clusters. Thus, on one hand, we compare its performance in the case of using different cluster-heads election methods under the same clustering approach and testing framework. On the other hand, we evaluate the fault-tolerance mechanism of proposed approach. Moreover, we compare our algorithm with some of the most referenced self-stabilizing solutions.

The remainder of the paper is organized as follows. Section II illustrates the related work on clustering approaches. Section III describes the proposed clustering approach, clusterhead election methods and the fault-tolerant mechanism. Theoretical validation is discussed in Section IV, where we compare our algorithm with some of most referenced self-stabilizing solutions. Section V presents the validation of the proposed approach through simulation. Finally, Section VI concludes this paper and presents our working perspectives.

II. Related Work

Several proposals of self-stabilizing clustering have been done in the literature [12], [13], [14], [15], [16], [17], [18]. However, self-stabilizing algorithms presented in [14], [15], [16], [17] are 1-hop clusters solutions.

A metric called *density* is used by Mitton *et al.* in [17], in order to minimize the reconstruction of structures for low topology change. Each node calculates its density and broadcasts it to its neighbors located at 1-hop. For the maintenance of clusters, each node periodically calculates its mobility and density.

Flauzac *et al.* [14] have proposed a self-stabilizing clustering algorithm, which is based on the identity of its neighborhood to build clusters. This construction is done using the identities of each node that are assumed unique. The advantage of this algorithm is to combine in the same phase the neighbors discovering and the clusters establishing. Moreover, this deterministic algorithm constructs disjoint clusters, i.e., a node is always in only one cluster.

In [15], Johnen *et al.* have proposed a self-stabilizing protocol designed for the state model to build 1-hop clusters having a bounded size. This algorithm guarantees that the network nodes are partitioned into clusters where each one has at most $SizeBound$ nodes. The clusterheads are chosen according to their $weight$ value. In this case, the node with the highest weight becomes clusterhead. In [16], Johnen *et al.* have extended this proposal from [15]. They have proposed a robust self-stabilizing weight-based clustering algorithm. The robustness property guarantees that, starting from an arbitrary configuration, after one asynchronous round, the network is partitioned into clusters. After that, the network stays partitioned during the convergence phase toward a legitimate configuration where clusters verify the ad hoc clustering properties. These approaches [15], [16], based on state model, are not realistic in the context of wireless sensor networks.

Self-stabilizing algorithms proposed in [12], [13], [18] are k-hops clustering solutions.

In [18], Mitton *et al.* applied self-stabilization principles over a clustering protocol proposed in [17] and they presented properties of robustness. Each node computes its *k-density* value based on its view ($\{k+1\}$-neighborhood) and locally broadcasts it to all its neighbors at distance k. Thus, each node is able to decide by itself whether it wins in its *1-neighborhood* (as usual, the smallest ID will be used to decide between joint winners). Once a clusterhead is elected, the clusterhead ID and its density are locally broadcasted by all nodes that have joined this cluster. A cluster can then extend itself until it reaches a cluster frontier of another clusterhead. The approach proposed in [17], [18] generates a lot of messages. The main reason is due to the fact that each node must know $\{k+1\}$-*neighboring*, computes its *k-density* value and locally broadcasts it to all its *k-neighbors*. This is very expensive in terms of messages and causes an important energy consumption.

In [13], using the criterion of minimal identity, Datta *et al.* have proposed a self-stabilizing distributed algorithm called *MINIMAL*. This approach is designed for the *state model* (also called *shared memory model*) and uses an unfair daemon. Authors consider an arbitrary network G of processes with unique IDs and no designated leader. Each process can read its own registers and those of its neighbors at distance k, but can write only to its own registers. They compute a subset D, a minimal *k-dominating* set of graph G. D is defined as a *k-dominating* set if every process that is not in D is at distance at most k from a member of D. *MINIMAL* converges in $O(n)$ rounds. Using D as the set of *clusterheads*, a partition of G into clusters, each of radius k follows. Authors show that $O(n^2)$ steps are sufficient for the phase clock to stabilize. And after stabilization, *MINIMAL* requires $O(n^2)$ steps to execute n actions. Thus, the system converges to a terminal configuration in $O(n^2)$ steps starting from any configuration and requires $O(\log(n))$ memory space per process, where n is the size of the network.

Caron *et al.* [12], using as metric a unique ID for each process and weighted edges, have proposed a self-stabilizing *k-clustering* algorithm based on a state model. Note that *k-clustering* of a graph consists in partitioning network nodes into disjoints clusters, in which every node is at a distance of at most k from the clusterhead. This solution is partially inspired by Amis *et al.* [19] and finds a *k-dominating* set in a network of processes. It is a combination of several self-stabilizing algorithms and it uses an unfair daemon. Each process can read its own registers and those of its neighbors at distance $k+1$, but can write only to its own registers. This algorithm executes in $O(nk)$ rounds and requires $O(log(n) + log(k))$ memory space per process, where n is the network size.

III. Proposed Clustering approach

A. Basic idea

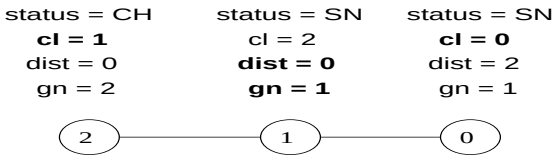
To simplify the description of our approach, we consider the case where the selection criterion to become clusterhead is the node's identity. We will present later the proposed approach using others CHs election criteria.

Our proposed algorithm is self-stabilizing and does not require any initialization. Starting from any arbitrary configuration, with only one type of exchanged message, nodes are structured in non-overlapping clusters in a finite number of steps. This message is called *hello message* and it is periodically exchanged between each neighbor nodes. It contains the following four information: node identity, cluster identity, node status and the distance to cluster-head. Note that cluster identity is also the identity of the cluster-head. Thus, the hello message structure is $hello(id_u, cl_u, status_u, dist_{(u,CH_u)})$. Furthermore, each node maintains a neighbor table $StateNeigh_u$ that contains the set of its neighboring nodes states. Whence, $StateNeigh_u[v]$ contains the states of nodes v neighbor of u.

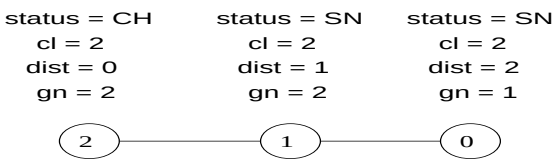
The solution that we propose proceeds as follows:

As soon as a node u receives a hello message, it executes three steps consecutively (see Algorithm 1). The first step is to update neighborhood. The next step is to manage the coherence and the last step is to build the clusters. During the last step, each node u chosen as cluster-head the node that optimizes the criterion and located at most a distance k. At the end of this three steps, u sends a hello message to its neighbors. The details of Algorithm 1 and mathematical proof are describe in Ba et *al.* [20]. Note that we have illustrated this algorithm with the ID criterion. Nevertheless, for the Degree and Energy criteria, we have the same design.

After updating the neighborhood, nodes check their coherency. For example, as a cluster-head has the highest identity, if a node u has CH status, its cluster identity must be equal to its identity. In Fig. 1(a), node 2 is cluster-head. Its identity is 2 and its cluster identity is 1, so node 2 is not a coherent node. Similarly for nodes 1 and 0. Each node detects its incoherence and corrects it during the coherence management step. Fig. 1(b) shows nodes that are coherent.



(a) Incoherent nodes



(b) Coherent nodes

Figure 1. Coherent and incoherent nodes

Algorithm 1: Fault-Tolerant and Energy-Efficient Generic Clustering algorithm for WSNs.

/* Upon receiving message from a neighbor */

Predicates

$P_1(u) \equiv (status_u = CH)$
$P_2(u) \equiv (status_u = SN)$
$P_3(u) \equiv (status_u = GN)$
$P_{10}(u) \equiv (cl_u \neq id_u) \vee (dist_{(u,CH_u)} \neq 0) \vee (gn_u \neq id_u)$
$P_{20}(u) \equiv (cl_u = id_u) \vee (dist_{(u,CH_u)} = 0) \vee (gn_u = id_u)$
$P_{40}(u) \equiv$
$\forall v \in N_u, (id_u > id_v) \wedge (id_u \geq cl_v) \wedge (dist_{(u,v)} \leq k)$
$P_{41}(u) \equiv \exists v \in N_u, (status_v = CH) \wedge (cl_v > cl_u)$
$P_{42}(u) \equiv \exists v \in N_u, (cl_v > cl_u) \wedge (dist_{(v,CH_v)} < k)$
$P_{43}(u) \equiv \forall v \in N_u / (cl_v > cl_u), (dist_{(v,CH_v)} = k)$
$P_{44}(u) \equiv \exists v \in N_u, (cl_v \neq cl_u) \wedge \{(dist_{(u,CH_u)} = k) \vee (dist_{(v,CH_v)} = k)\}$

Rules

/* Update neighborhood */
$StateNeigh_u[v] := (id_v, cl_v, status_v, dist_{(v,CH_v)})$;

/* Cluster-1: Coherent management */
$R_{10}(u) :: P_1(u) \wedge P_{10}(u)$
$\longrightarrow cl_u := id_u; gn_u = id_u; dist_{(u,CH_u)} = 0;$
$R_{20}(u) :: \{P_2(u) \vee P_3(u)\} \wedge P_{20}(u) \longrightarrow$
$status_u := CH; cl_u := id_u; gn_u = id_u; dist_{(u,CH_u)} = 0;$

/* Cluster-2: Clustering */
$R_{11}(u) :: \neg P_1(u) \wedge P_{40}(u) \longrightarrow status_u := CH; cl_u := id_v; dist_{(u,CH_u)} := 0; gn_u := id_u;$
$R_{12}(u) :: \neg P_1(u) \wedge P_{41}(u) \longrightarrow status_u := SN; cl_u := id_v; dist_{(u,v)} := 1; gn_u := NeighCH_u;$
$R_{13}(u) :: \neg P_1(u) \wedge P_{42}(u) \longrightarrow$
$status_u := SN; cl_u := cl_v; dist_{(u,CH_u)} := dist_{(v,CH_v)} + 1; gn_u := NeighMax_u;$
$R_{14}(u) :: \neg P_1(u) \wedge P_{43}(u) \longrightarrow status_u := CH; cl_u := id_v; dist_{(u,CH_u)} := 0; gn_u := id_u;$
$R_{15}(u) :: P_2(u) \wedge P_{44}(u) \longrightarrow status_u := GN;$
$R_{16}(u) :: P_1(u) \wedge P_{41}(u) \longrightarrow status_u := SN; cl_v := id_v; dist_{(u,v)} := 1; gn_u := NeighCH_u;$
$R_{17}(u) :: P_1(u) \wedge P_{42}(u) \longrightarrow$
$status_u := SN; cl_u := cl_v; dist_{(u,CH_u)} := dist_{(v,CH_v)} + 1; gn_u := NeighMax_u;$

/* Sending hello message */
$R_0(u) :: hello(id_u, cl_u, status_u, dist_{(u,CH_u)})$;

B. Cluster-heads election

Existing clustering approaches use one or more criteria for electing cluster-heads, for example: nodes' ID, degree, density, mobility, distance between nodes, service time as a CH, security, information features or a combination of multiple criteria. However, to the best of our knowledge, there is no

paper in the literature where the same proposed approach is compared in the case of different CH election methods. It is important to study the influence of each criterion under the same test conditions and, ideally, under the same clustering approach. To this end, we propose a generic distributed self-stabilizing clustering approach that can be used with any CH election criterion. Then, we compare costs and performance of the proposed solution in the case where several election criteria (Highest-ID, Highest-degree and residual energy of nodes) are used.

1) Highest ID:

Lowest-Identifier based clustering was originally proposed by Baker et *al.* [21]. It has proven that, clustering based on ID criterion is one of the most performant approaches in ad hoc networks [22], [23], [24], [25].

In our approach, each node compares its identity with those of its neighbors a distance 1. A node u elects itself as a clusterhead if it has the highest identity among all nodes of its cluster (in Fig. 2, example of node 9 in cluster V_9). If a node u discovers a neighbor v with a highest identity then it becomes a node of the same cluster as v with SN status (in Fig. 2, example of nodes $1, 3, 4$ and 7 in cluster V_9). If u receives again a hello message from another neighbor which is into another cluster than v, the node u becomes gateway node with GN status (in Fig. 2, example of nodes 5 and 8 in cluster V_{10} and node 2 in cluster V_9). As the hello message contains the distance between each node u and its clusterhead, u knows if the diameter of cluster is reached. So it can choose another cluster.

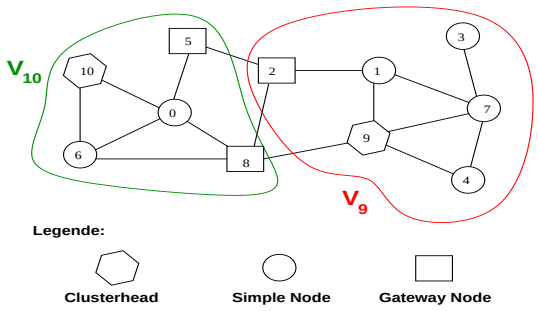


Figure 2. Clusters organization ($k = 2$)

2) Highest or Ideal Degree: In this approach, we determine how well suited a node is for becoming CH according to its degree D (i.e., the number of neighbors). There are two categories of approaches based on nodes' degrees. Some of them propose to limit communications by electing the node having the highest degree as CH. This is an original proposal of Gerla and Tsai [26]. However, each CH can ideally support only ρ (a pre-defined threshold) nodes to ensure an efficient functioning regarding delay and energy consumption. Indeed, at each step of the routing process, when a node has many neighbors it receives as many messages as its degree. This leads to a rapid draining of sensors' battery power. To ensure that a CH handles upto a certain number of nodes in its cluster, some approaches [24], [27], [28] propose to elect as CH the node having the nearest degree to an ideal value ρ. Thus, the best candidate is the one minimizing its distance to this ideal degree $\triangle_d = |D - \rho|$.

For the two cases described above, when more than one node has the maximum (respectively ideal) degree and is candidate to become a CH, the election is done according to a secondary criterion which is the highest ID. As each node of the network has a unique ID, this criterion is discriminating.

3) Residual Energy: In this approach, decision-making concerning the most suitable node to become CH is done according to the residual energy (i.e., remaining battery power level) of each sensor. Indeed, CHs are generally much more solicited during the routing process. So, in order to preserve their energy and to avoid the frequently reconstruction of the clusters, CHs need more important battery levels compared to the others normal nodes.

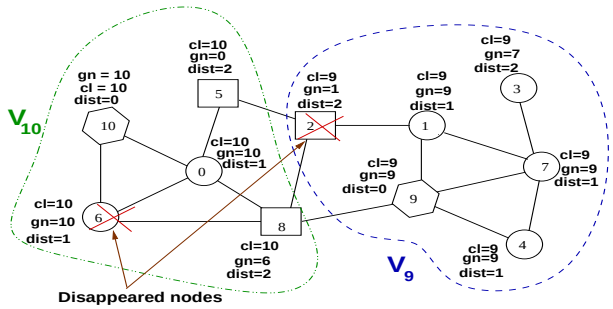
During the clustering procedure, network nodes progressively consume their energy due to the messages exchanges. Thus, after some rounds a node i with initially the maximum battery power level and candidate to become a CH can have later less energy than an another neighbor node j. This can lead to more iterations aiming at electing the other node j with the maximum residual energy. In order to limit the frequently changes of CH candidates for a negligible energy difference, we propose to use an energy gain threshold E_T. Thus, while $\triangle_e = |E_i - E_j|$ is less than E_T, the node i preserves its leadership position. This guarantees more stability of the clustering process and extends the network lifetime by minimizing the energy consumption involved in the clustering procedure.

C. Fault-tolerance mechanism

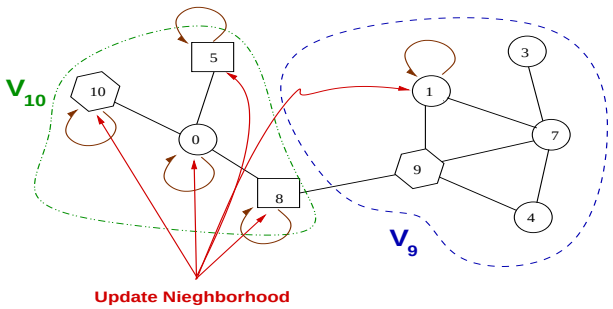
In this section, we study the fault-tolerance mechanism of proposed approach. Our algorithm is fault-tolerant and adapted to topological changes. To the best of our knowledge, there is no paper in the literature where the solutions are fault-tolerant, energy-aware, self-stabilizing and where the same proposed approach is compared in the case of different CH election methods.

A system failure occurs when the delivered service deviates from the specified service [10]. Hardware and software faults affect the system state and the operational behavior, such as memory or register content, program control flow, and communication links, etc. Communication faults can be caused due to hardware failure or energy depletion. Communication can be disrupted due to environmental conditions like wind or rain. Hardware faults can also disrupt radio communication, ending all the communication.

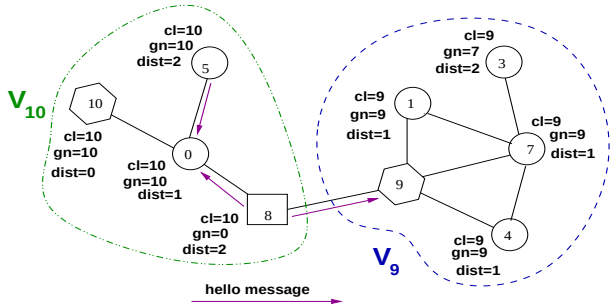
In the following, we consider that after the occurrence of a fault, the concerned node disappears from the network and the graph remains connected. We also assume that faults can occur after stabilization (i.e., after clusters formation). As soon as a node detects the disappearance of a neighbor, it considers this as an occurred fault. Thus, it triggers the fault-tolerance mechanism called *re-clustering*. Let u the disappeared node. According the status of node u, two cases are possible:



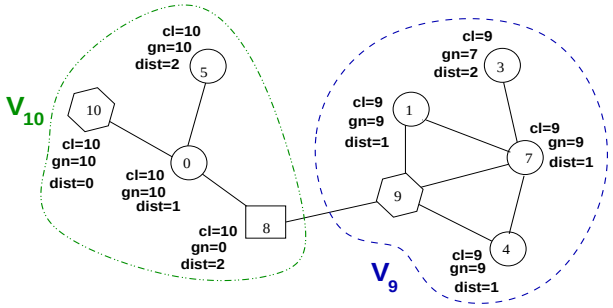
(a) Clusters structure before the disappearance of node's status SN or GN. In this example, node 6 in cluster of node 10 with SN status and node 2 in cluster of node 9 with GN status will disappear from the network.



(b) After the disappearance of node's status SN or GN, nodes 10, 0 and node 8 member of cluster of node 10 that detect the disappearance of node 6 update its neighborhood table. Similarly, node 5 and 1 update their neighborhood tables after the disappearance of node 2 with GN status and member of cluster of node 9.



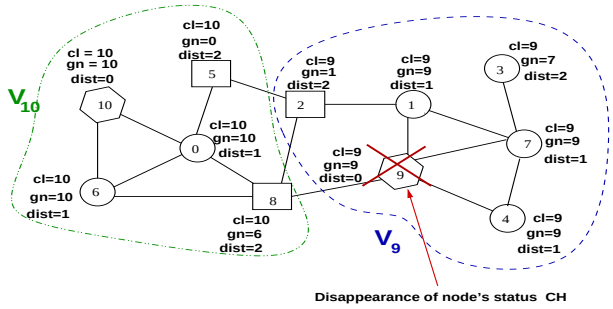
(c) After updating the neighborhood, nodes impacted by the disappearance of a neighbor send a hello message to its neighbors. In this example, node 8 is impacted by the disappearance of node 6. In fact, as shows in Fig. 3(a), the gateway of node 8 to reach its CH (node 10) was node 6. Thus, node 8 selects node 0 as the new gateway to reach its CH. It sends a hello message to its all neighbors at distance 1 to notify of its gateway change. Similarly to node 5, it becomes simple node with SN status. In fact, node 2 was the only one to be a member of another cluster in the neighborhood of node 5.



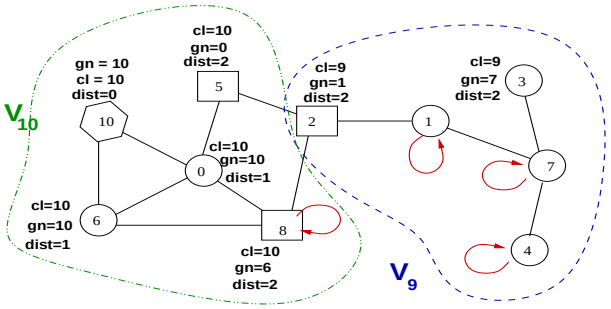
(d) Clusters structure at the end of re-clustering process caused by the disappearance of node's status SN or GN. Thus, the disappearance of node's status SN or GW does not lead to clusters change but only one updating neighborhood table.

Figure 3. Disappearance of node's status SN or GW (in this example $k = 2$)

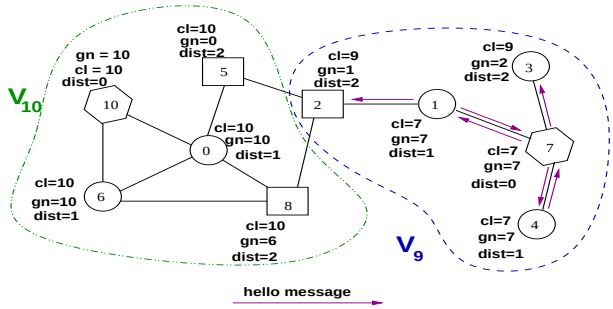
- **Case 1**, $status_u \in \{SN, GN\}$: the disappeared node is a simple or gateway node. In this case, all node v that detects the disappearance of node u removes all information about u from its neighborhood table. As illustrated in Fig. 3, the disappearance of node's status SN or GW does not lead to clusters change but only one updating neighborhood table. However, if the node u has been chosen by a node v as gateway (i.e., $gn_v = id_u$) through which it can reach its CH, then v chooses another node w in its neighborhood table as new gateway to reach its CH. Furthermore, if node u was the only one to be a member of another cluster in the neighborhood of node v, thus v becomes simple node with status SN. After updating the neighborhood table, all node v that is impacted by the disappearance of node u sends a hello message to its all neighbors distance 1.
- **Case 2**, $status_u = CH$: if a node u with CH status disappears from de the network, the fault-tolerance mechanism proceeds as follows (example of the disappearance of node 9 as illustrated in Fig. 4):
  1) For all node v at distance 1 of u that is member of another cluster (i.e., $(cl_v \neq id_u) \wedge (dist_{(v,CH_v)} = k)$), the only requirement action is to remove all information about this CH by updating its neighborhood table. This is the case case of node 8 as illustrated in Fig. 4(a) and Fig. 4(b)
  2) For all node v at distance 1 of u that is member of cluster of node u (i.e., $v \in N_u$ such that $status_v \in \{SN, GN\} \wedge (cl_v = id_u)$) as illustrated in Fig. 4(b) and Fig. 4(c), v executes three actions. Firstly, it removes all information about this CH by updating its neighborhood table. Secondly, it triggers the re-clustering process in order to choose another cluster-head. Thirdly, after having chosen another cluster-head, each node v sends to its neighbors at distance 1 a hello message in order to inform their cluster-head change. Therefore, all node w at distance 2 to u such that $w \in N_v \backslash \{v\} \wedge cl_w = id_u$ receives information about the disappearance of node u.
  3) Thus, in our process of re-clustering, we have the following induction assumption: each node at distance i of the disappeared CH, executes process re-



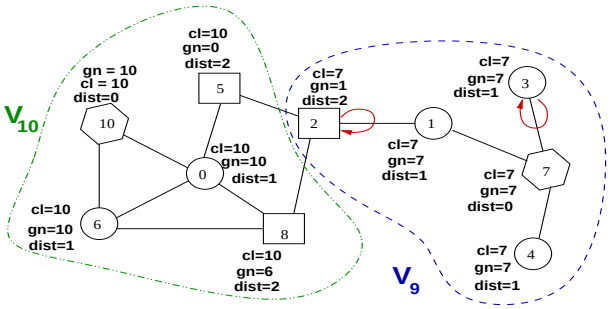
(a) Structure of clusters before the disappearance of node 9 with CH status.



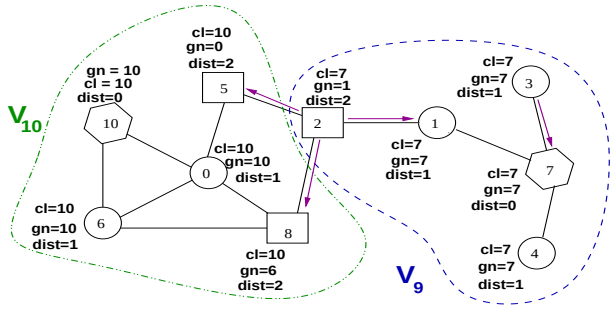
(b) After the disappearance of node 9 with CH status, node 8 removes all information about node 9 in its neighborhood table. In fact, node 8 is member of cluster of node 10 and it is at distance 2 of node 10. Thus, it is not affected by the disappearance of node 9. Nevertheless, nodes 1, 4 and 7 have as cluster-head node 9. Thus, they will trigger the process of re-clustering after updating their neighborhood table.



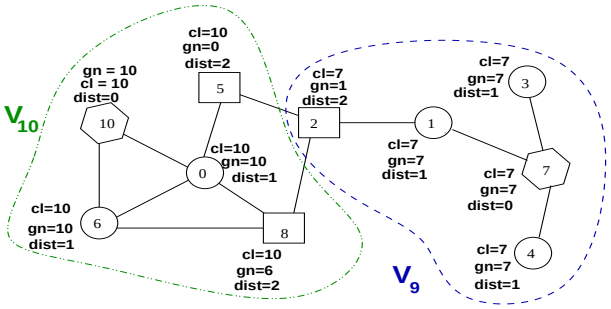
(c) After updating neighborhood, each node impacted by the disappearance of node 9 chooses another cluster-head. To do this, each node select from its neighborhood table the node with highest ID. Node 7, as it have the highest ID in its neighborhood, it becomes cluster-head. Nodes 1 and 4 choose node 7 as cluster-head. In fact, node 7 represents node with highest ID at distance 1 in the neighborhood of nodes 1 and 4. Therefore, each node with state change sends a hello message to its neighbors at distance 1.



(d) Upon receiving a hello message from node 1 notifying its cluster-head change, node 2 knows that its cluster-head (node 9 as illustrated in Fig. 4(a)) has disappeared. Thus, it triggers the process of re-clustering after updating its neighborhood table. As node node 7 is the node with the highest ID in at most at distance 2 (node 10 is at distance 3 and in this example $k = 2$), it is selected as a cluster-head by node 2. Similarly, node 3 applies the same principle and becomes member of cluster of node 7.



(e) Nodes 2 and 3 send a hello message to its neighbors at distance 1 in order to inform about its state change. All nodes that receive message from node 2 and node 3 update information about these nodes in its neighborhood table. As $k = 2$ in this example, the re-clustering process ends at this step. All clusters become stable as showed in Fig. 4(f).



(f) Structure of clusters after the disappearance of node 9 with CH status

Figure 4. Disappearance of node with CH status (in this example $k = 2$)

clustering and informs its neighbors at distance $i+1$. So on until the whole network becomes stable again. As illustrated in Fig. 4(f), nodes 2 and 3 apply this induction assumption to correct the disappearance of its CH.

IV. Theoretical Validation

In [20], we have provided a formal proof of our clustering approach. Table I illustrates a comparison of stabilizing time and memory space between our proposal algorithm and other approach designed for the state model. We note that our stabilization time does not depend on the parameter k contrary to approach proposed by Caron et *al.* [12]. We have a unique

TABLE I
THEORETICAL COMPARISON OF STABILIZING TIME AND MEMORY SPACE

	Stabilization Time	Memory space per node	neighborhood
Our approach	$n+2$	$\log(2n+k+3)$	1 hop
Datta et al. [13]	$O(n)$, $O(n^2)$	$O(\log(n))$	k hops
Caron et al. [12]	$O(n*k)$	$O(log(n)+log(k))$	k+1 hops

phase to discover the neighborhood and build *k-hops* clusters and an unique stabilization time contrary to approach describes in [13]. Furthermore, we consider a 1-hop neighborhood at opposed to Datta et *al.* [13] and Caron et *al.* [12].

Furthermore, in Ba et *al.* [29], we have compared our proposed algorithm with one of most referenced papers on self-stabilizing solutions based on message-passing model [18]. This shows that we reduce communication cost and energy consumption by a factor of at least 2.

V. VALIDATION FRAMEWORK

In this section, we present the evaluation study that we carried out using *ONMeT++* [30] simulator to compare the performance of the previously described clustering approach when utilizing different CH election methods. For generating random graphs, we have used the SNAP [31] library. All simulations were carried out using *Grid'5000* [32] platform.

A. Models

In order to implement our clustering approach in a realistic way, we use standard models for representing both the energy consumption and the network structure.

1) Energetic model: To model the energy consumption for a node when it sends/receives a message, we use the first order radio model proposed by Heinzelman et *al.* [33] and used in many other studies [4], [34], [35]. A sensor node consumes E_{Tx} amount of energy to transmit one l-bits message over a distance d (in meters). As shown in equation (1), when the distance is higher than a certain threshold d_0, a node consumes more energy according to a different energetic consumption model.

$$E_{Tx}(l,d) = \begin{cases} l * E_{elec} + l * \varepsilon_{fs} * d^2, & \text{if d} < d_0; \\ l * E_{elec} + l * \varepsilon_{mp} * d^4, & \text{if d} \geq d_0. \end{cases} \quad (1)$$

Each sensor node will consume E_{Rx} amount energy when receiving a message, as shown in equation (2).

$$E_{Rx}(l) = l * E_{elec} \quad (2)$$

Parameters values used in equations (1) and (2) to model energy are summarized in Table II.

TABLE II
RADIO MODELING PARAMETERS

Parameter	definition	Value
E_{elec}	Energy dissipation rate to run radio	$50nJ/bit$
ε_{fs}	Free space model of transmitter amplifier	$10pJ/bit/m^2$
ε_{mp}	Multi-path model of transmitter amplifier	$0.0013pJ/bit/m^4$
d_0	Distance threshold	$\sqrt{\varepsilon_{fs}/\varepsilon_{mp}}$

2) Network model: In our experimental studies, we consider networks represented by an arbitrary random graph based on a Poisson process with $\lambda > 1$ for all network sizes. In fact, random graphs based on a Poisson process provide a better representation for WSNs. It is used in many studies like [18], [36], [37], [38], [39]. Nodes in the network are distributed uniformly at random as per a homogeneous spatial Poisson process of intensity λ in two-dimensional plane. We model our network by an undirected graph $G = (V, E)$ following standard models for distributed systems given in [40], [41]. $V = n$ is the set of network nodes and E represents all existing connections between nodes. Each node u of the network has a unique identifier noted id_u such that $0 \leq id_u \leq n-1$. An edge exists if and only if the distance between two nodes is less or equal than a fixed radius $r \leq d_0$. This r represents the radio transmission range, which depends on wireless channel characteristics including transmission power. Accordingly, the neighborhood of a node u is defined by the set of nodes that are inside a circle with center at u and radius r and it is denoted by $N_r(u) = N_u = \{\forall v \in V \setminus \{u\} \mid d_{(u,v)} \leq r\}$. The degree of a node u in G is the number of edges that are connected to u, and it is equal to $deg(u) = |N_r(u)|$.

B. Testbed

The parameters used in our simulations are summarized in Table III. In all simulations, a 99% confidence interval I_c is computed for each average value represented in the curves. These intervals are plotted as error bars and computed according to this equation: $I_c = [\overline{x} - t_\alpha \frac{\delta}{\sqrt{n}}; \overline{x} + t_\alpha \frac{\delta}{\sqrt{n}}]$, where n is the population length, $\overline{x}$ is the average value, δ is the standard deviation, and finally, t_α has a fixed value of 2.58 in the case of 99% interval.

TABLE III
SIMULATION PARAMETERS

Parameter	**Value**
Message size	2000 bits
distance between 2 nodes	100 meters
Initial Energy $\mathcal{E}_i^{init}$	{1,2,3} Joules
Ideal degree	{6,10,12,20}
Energy threshold	{0.1,0.05} %
Number of nodes	[100,1000]
Random graph model	Poisson process
λ parameter	[2,11]
k parameter	[1,10]
Number of simulations for each network size	100

C. Simulation results: evaluation of cluster-head election criteria

In this section, we present a performance evaluation of cluster-head election criteria. For each cluster-head election criterion, the following performance parameters are assessed.

- Total exchanged messages ($\mathcal{M}_{total}$): It is defined as the total number of exchanged messages in the whole network until the formation of stable clusters.

$$\mathcal{M}_{total} = \sum_{i=0}^{n-1} \mathcal{M}_i^{Send}$$

  Where $\mathcal{M}_i^{Send}$ is the total number of messages send by sensor node i and n represents the network size.
- Total energy consumption ($\mathcal{E}_{total}$) : It is defined as the energy consumption necessary to the clusters formation.

$$\mathcal{E}_{total} = \sum_{i=0}^{n-1} (\mathcal{E}_i^{init} - \mathcal{E}_i^{av})$$

  Where $\mathcal{E}_i^{init}$ is the initial energy of sensor node i and $\mathcal{E}_i^{av}$ is the available energy of node i at the end of clustering.
- Number of clusters: It is defined as the percentage of formed clusters according to the network size.

Theses performances are evaluated according λ and k parameters.

1) Communication cost (messages): We start the evaluation of our protocol by measuring the necessary communication cost in terms of exchanged messages to achieve the clustering procedure.

In the set of experiments described in Fig. 7, we calculate the communication cost according λ (Fig. 7(a), Fig. 7(c) and Fig. 7(e)) and k (Fig. 7(b), Fig. 7(d) and Fig. 7(f)) parameters for each cluster-head election criterion. These simulations are based on the same network topology for each value of λ and k parameters.

As illustrated in $3D$ curves show in Fig. 7(a), Fig. 7(b), Fig. 7(c), Fig. 7(d), Fig. 7(e) and Fig. 7(f), we observe that, for each cluster-head election criterion, the total number of exchanged messages increases linearly together with the number of nodes in the network. Indeed, the increase in network size entails more communications. However, Fig. 7 shows that our protocol is scalable. Furthermore, λ and k parameters do not affect the amount of generated messages by our protocol. The main reason is that our algorithm is based only on information from neighboring nodes at distance 1 to build k-hops clusters.

Experiments in Fig. 7 show that the clustering based on the criterion of ID generates less messages. Fig. 5 shows the gain of the ID criterion compared to Degree and Energy criteria according λ parameter and $k = 2$. The criterion of ID reduces the communication cost between 7.5% and 10.2% compared to Degree criterion and between 22.6% and 32.1% compared to Energy criterion. The main reason is that the ID criterion brings greater stability during the clustering phase. In addition, the ID criterion is simpler and deterministic

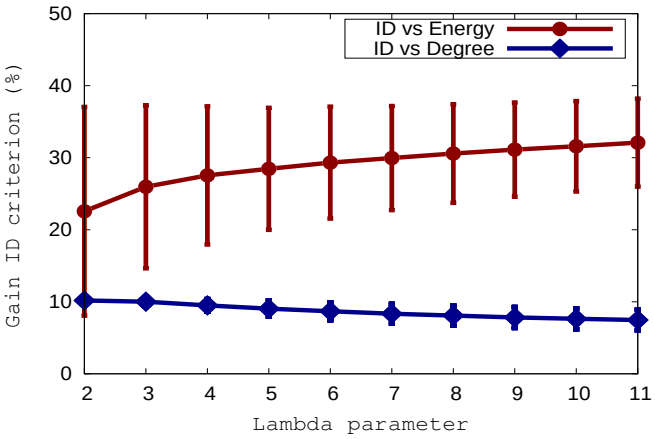


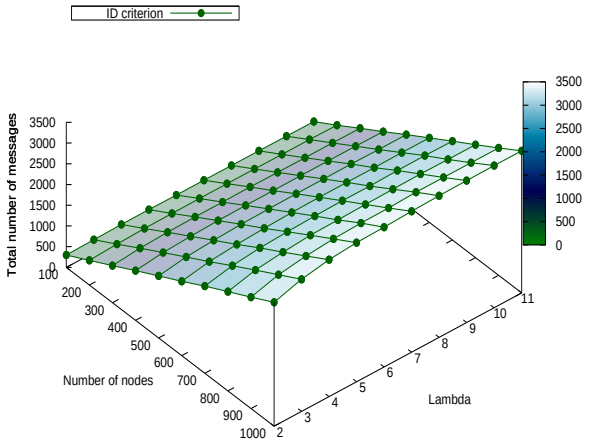
Figure 5. Communication cost reduction of ID criterion (%)

compared to the criteria of degree or energy. Indeed, for the Degree criterion, it is necessary for nodes to receive a message from their neighbors to calculate their degree. Then, the degree is sent by broadcast and after that, clustering phase begins. This is expensive in terms of messages. Also, the residual energy criterion generates more messages compared to the ID and Degree criteria. As energy level is a parameter which decreases during the clustering phase, it provides less stability and requires more messages to reach a stable state in the entire network. Note that we observe the same gain in terms of energy consumption of the ID criterion compared to Degree and Energy criteria.

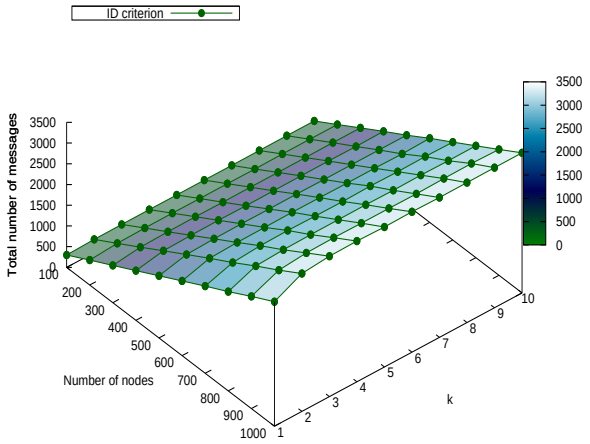
2) Energy consumption: In the second set of experiments shown in Fig. 8, we have measured the energy consumption required for building clusters in the entire network according network size and λ or k parameters.

As illustrated in $3D$ curves described in Fig. 8(a), Fig. 8(b), Fig. 8(c), Fig. 8(d), Fig. 8(e) and Fig. 8(f), we note that for each cluster-head election criterion, the energy consumption increases linearly together with the number of nodes in the network. The main reason is that the energy consumption is a linear function following the communication cost. However, λ and k parameters do not affect the amount the energy consumption required for building clusters. In fact, as illustrated in experiments show in Fig. 7, the communication cost does not depending on λ and k parameters.

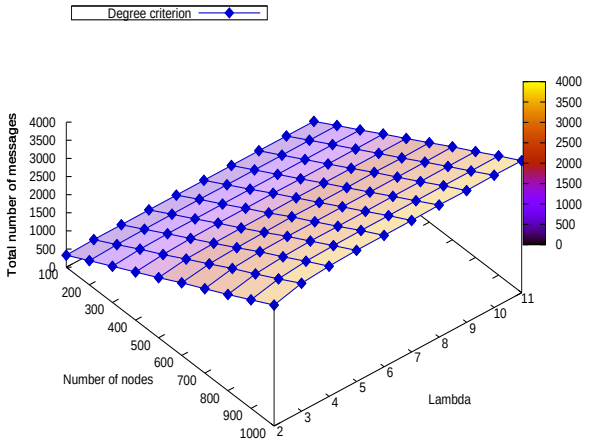
Experiments illustrated in Fig. 8 show that the clustering based on the ID criterion requires less energy consumption during the clustering phase. Indeed, results illustrated in Fig. 7 show that both Degree and Energy criteria generate more messages than ID criterion during the clusters formation. However, communications are the major source of energy consumption in WSNs. Moreover, ID criterion reduces energy consumption required to the clusters formation. Fig. 6 shows the gain of ID criterion compared to Degree and Energy criteria according k parameter and $\lambda = 6$. The ID criterion reduces the energy consumption between 7.3% and 9.7% compared to Degree criterion and between 18.1% and 35.1% compared to Energy



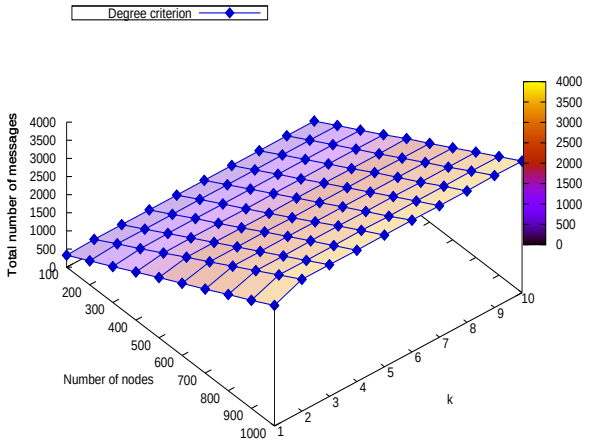
(a) ID criterion according λ parameter



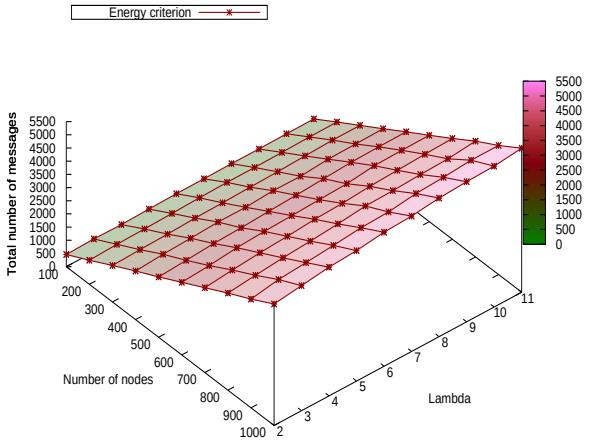
(b) ID criterion according k parameter



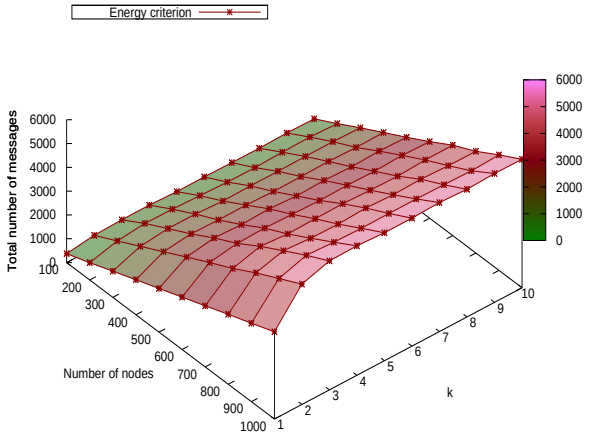
(c) Degree criterion according λ parameter



(d) Degree criterion according k parameter

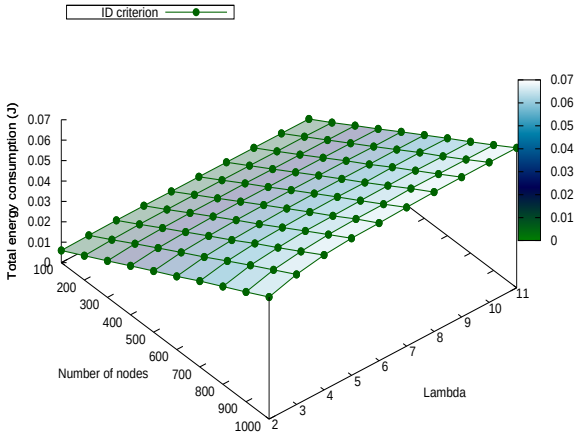


(e) Energy criterion according λ parameter

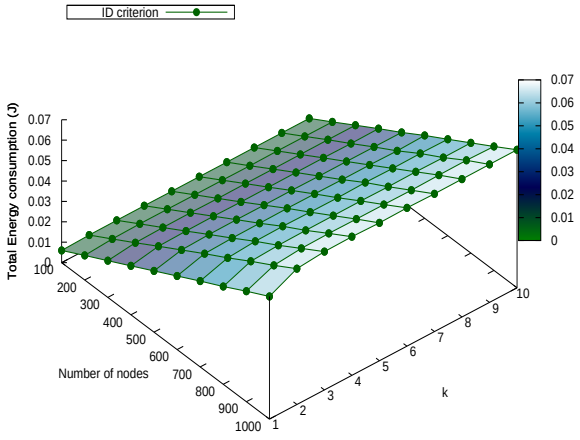


(f) Energy criterion according k parameter

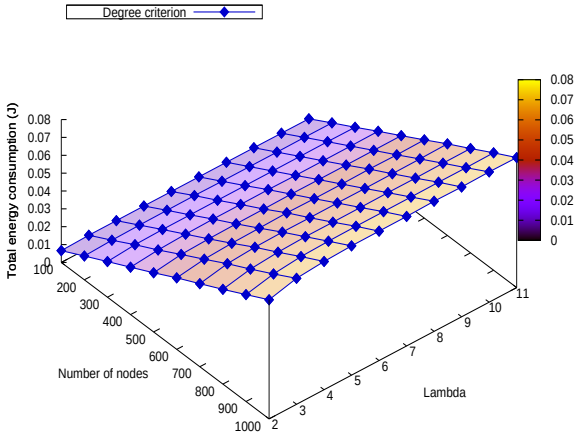
Figure 7. Total exchanged messages according λ and k parameters



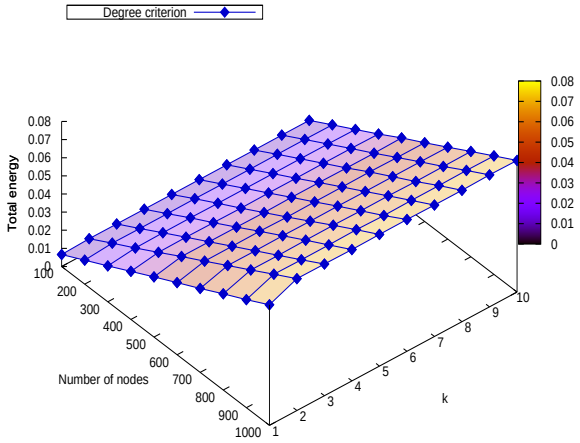
(a) ID criterion according λ parameter



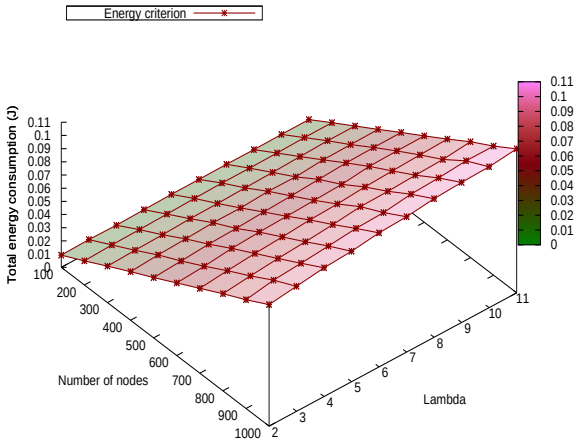
(b) ID criterion according k parameter



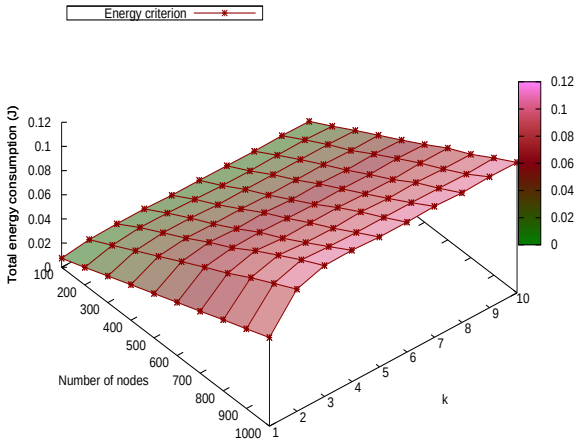
(c) Degree criterion according λ parameter



(d) Degree criterion according k parameter

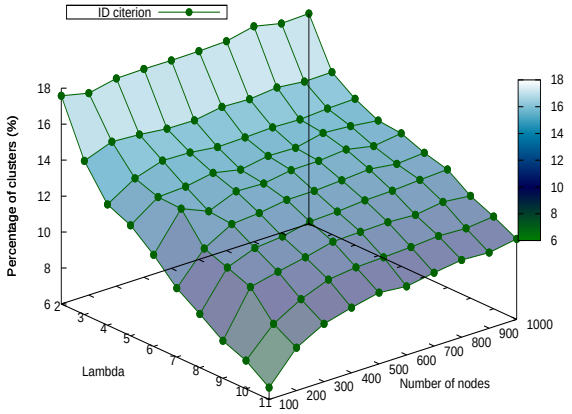


(e) Energy criterion according λ parameter

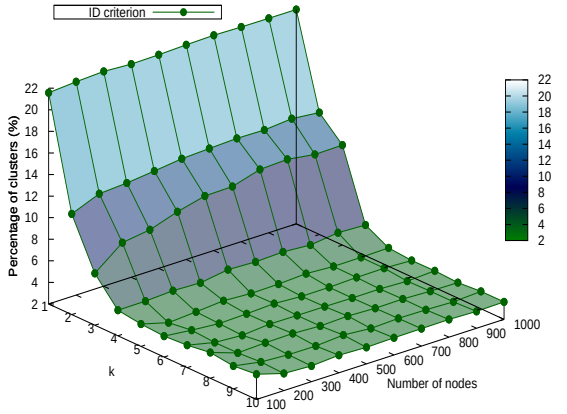


(f) Energy criterion according k parameter

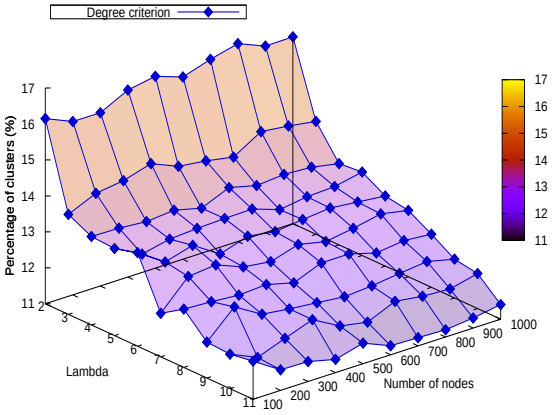
Figure 8. Total energy consumption according λ and k parameters



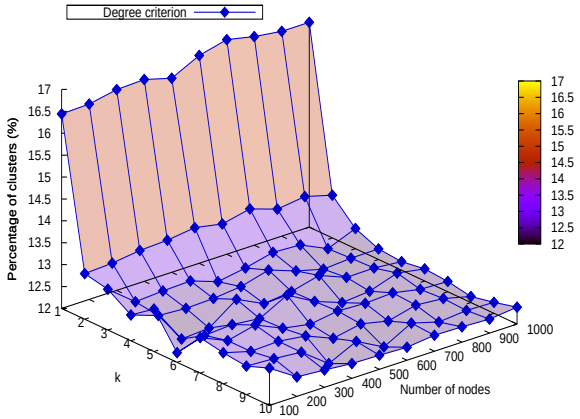
(a) ID criterion according λ parameter



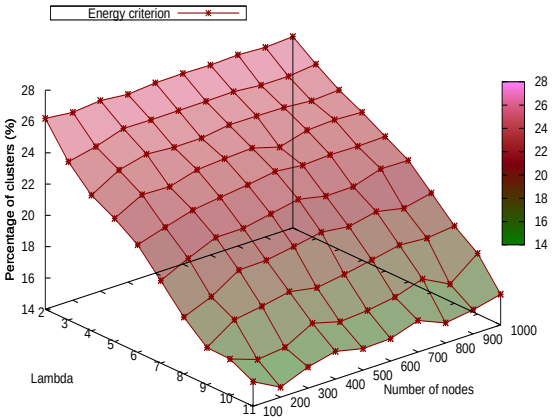
(b) ID criterion according k parameter



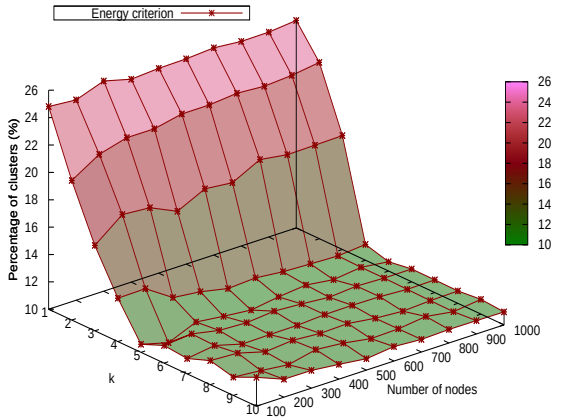
(c) Degree criterion according λ parameter



(d) Degree criterion according k parameter



(e) Energy criterion according λ parameter



(f) Energy criterion according k parameter

Figure 9. Percentage of number of clusters according λ and k parameters

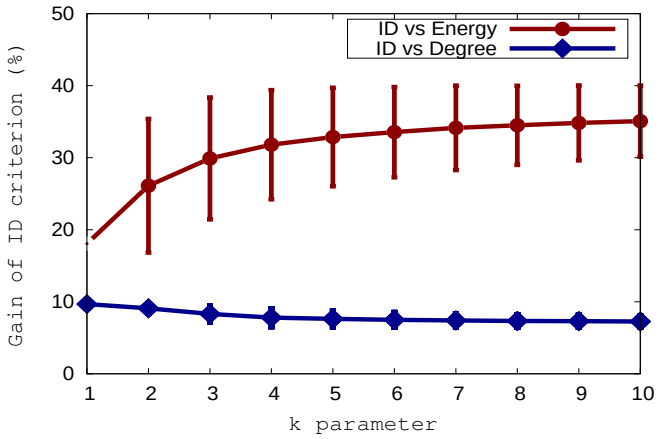


Figure 6. Energy consumption reduction of ID criterion (%)

criterion. Note that we observe the same gain in terms of communication cost for the ID criterion compared to Degree and Energy criteria.

3) Number of clusters : The number of clusters build by our protocol for each cluster-head election criterion is illustrated by the set of experiments described in Fig. 9. These $3D$ curves reflect the percentage of formed clusters according network and λ or k parameters.

In Fig. 9(a), Fig. 9(c) and Fig. 9(e), we set $k = 2$ and we vary arbitrary the value of λ parameter between 2 and 11. Firstly, we observe that for each cluster-head election criterion, the percentage of clusters build does not significantly vary according the network size for each fixed value of λ parameter. Therefore, our approach is scalable in term number of clusters. Secondly, for each fixed network size, the percentage of clusters decreases as the value of λ parameter increases. In fact, the λ parameter represents the average number of neighbors. Thus, network density increases as the λ parameter increases. Therefore, clusters size increases, implying a reduction of the number of clusters. Note that the ID criterion provides a better distribution of clusters (between 6% and 18%) compared to Degree and Energy criteria. The main reason is that ID criterion provides more stability.

In Fig. 9(b), Fig. 9(d) and Fig. 9(f), we set $\lambda = 6$ and we arbitrary vary the value of k parameter between 1 and 10. Firstly, we observe that for each cluster-head election criterion, the percentage of clusters build does not significantly vary according the network size for each fixed value of k parameter. Therefore, our approach is scalable in term number of clusters. Secondly, for each fixed network size, the percentage of clusters decreases significantly as the value of k increases. In fact, if the k parameter increases, clusters of larger diameter are constructed. This implies that clusters size is larger. Thus, a decrease in the percentage of clusters built. Note that, values of k parameter that provide the better distribution of clusters are comprised between 2 and 4. Beyond, we obtain large clusters that will not be easy to manage by the cluster-head.

4) Impact of highest and Ideal degree: To evaluate the impact of highest and Ideal degree as studied in Section III-B2, we arbitrary fix $\triangle_d$ to 6, 10, 12, and 20 and then we evaluate energy consumption. Note that in the set of experiments shows in Fig. 10, we fix $k = 2$ and $\lambda = 6$. We observe a slight decrease in the energy consumption for ideal degree fixed to 6 compared to highest degree as illustrated in Fig. 10. In fact, the Ideal degree fixed is equal to the λ parameter. As the λ parameter represents the average number of neighbors in the whole network, a Ideal degree equal to the λ parameter reduces communications required during the clusters formation implying slight decrease in energy consumption.

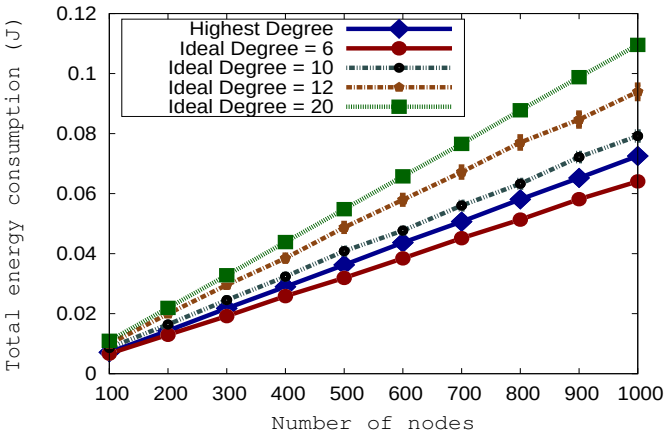


Figure 10. Energy consumption under highest and ideal degrees

On the other hand, we observe an increase in the energy consumption for ideal degree fixed at 12, 15 and 20. The main reason is that nodes attempt to join the cluster-head that is the node minimizing its distance to this ideal degree ρ ($\triangle_d = |D - \rho|$). This leads an increase of communications required during the clusters formation, implying at the same time an increase of energy consumption. The major advantage of this method is to allow the setting of the number nodes managed by cluster-head.

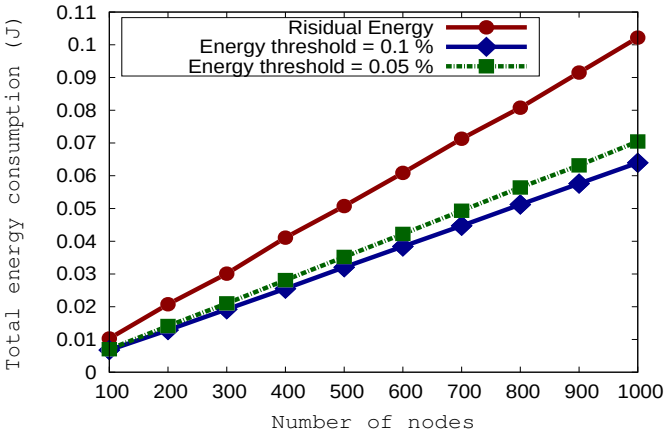
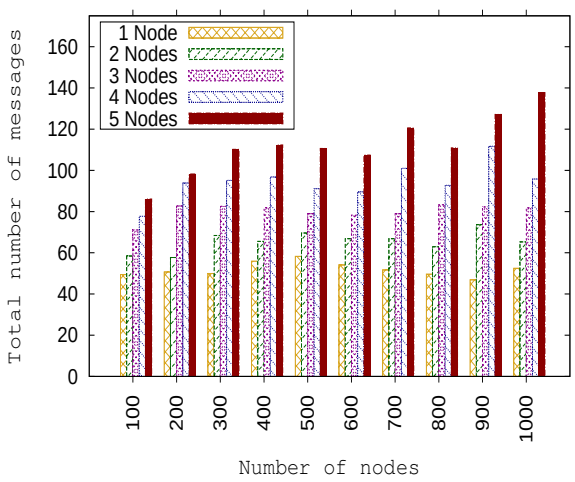
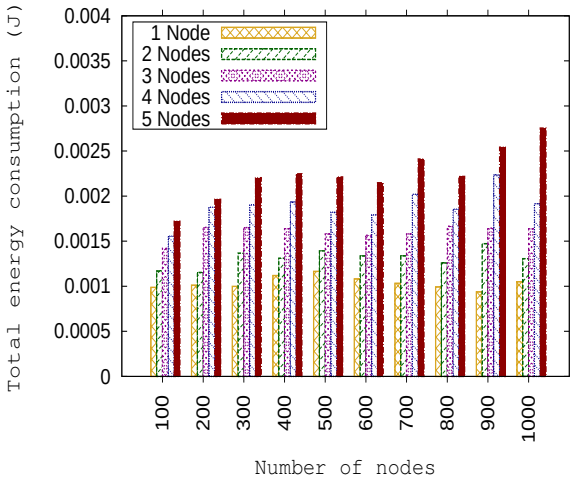


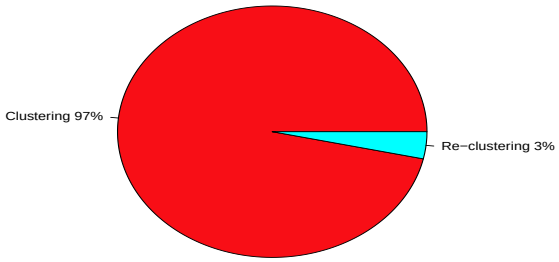
Figure 11. Residual energy vs Energy threshold



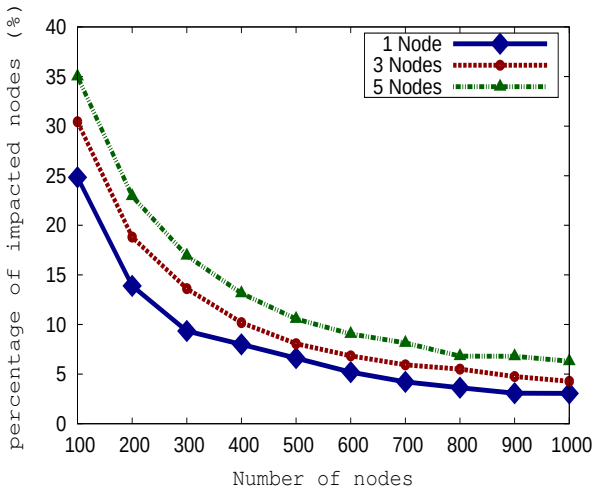
(a) Supplementary communication cost



(b) Supplementary energy consumption



(c) Impact of 5 nodes: percentage of supplementary cost



(d) Percentage of impacted nodes

Figure 12. Fault-tolerant in the case of 1, 2, 3, 4 and 5 disappeared nodes

5) Impact of residual energy or energy threshold: As the main problem with the criterion of energy is its volatility, we fix energy threshold to limit abrupt changes of nodes when their energy CHs decreases substantially. We fixed the energy threshold to 0.1% and 0.05% and we evaluate both energy consumption. Fig. 11 shows that energy threshold reduces energy consumption during the clustering phase. Indeed, nodes no longer change after a slight decrease of their energy CHs. This entails less messages exchanged and less energy consumption.

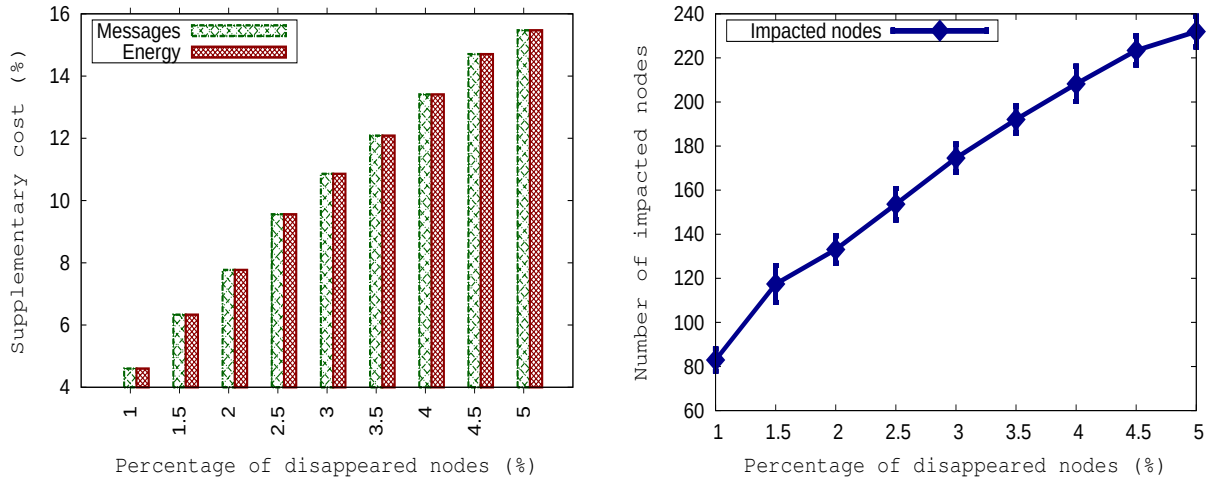
D. Simulation results: fault-tolerant evaluation

In this section, we study by simulation the robustness or our approach again nodes failure. To do this, we consider only the ID criteria of our protocol.

Firstly, we vary the network size between 100 and 1000 nodes. For each network size, after stabilization (i.e., formation of stable clusters in whole network), we randomly disappear 1, 2, 3, 4 and 5 nodes. Thus, the fault-tolerance mechanism is triggered by starting the re-clustering process. At the end of the re-clustering process, we evaluate the supplementary communication cost, energy consumption and percentage of impacted nodes. For each network size, we compute for each metric the average as the average of all values corresponding to 100 simulations results with 99% fixed as confidence interval.

Fig. 12(a) shows the supplementary communication cost at the end of the re-clustering process according the network size. We observe that, the disappearance of 1 until 5 nodes and according network size, generates on average between 50 and 150 supplementary messages in whole network. Fig. 12(b), shows a supplementary energy consumption between 1 mJ and 4 mJ. We remark that the energy consumption follows the same pattern as the communication cost. The main reason is that the energy consumption is a linear function following the communication cost.

In order to evaluate the impact of re-clustering, in Fig. 12(c) we calculate the percentage of re-clustering cost compared to



(a) Supplementary cost: messages and energy

(b) Number of impacted nodes in a network with 1000 nodes

Figure 13. Fault-tolerant according the percentage of disappeared nodes between 1% and 5%

clustering cost. We note that, the re-clustering cost (in terms of communication cost and energy consumption) represents 3% of resource consumption compared to the clustering cost. In fact, with our fault-tolerance mechanism, as illustrated in examples shown in Fig. 3 and Fig. 4 in the Section III-C, the occurrence of fault impacts generally the cluster where the fault has occurred and eventually adjacent clusters. This result is consolidated through Fig. 12(d), where we have estimated the percentage of impacted nodes compared to the network size. We say that a node u is impacted if only if, one of its local variables (cl_u, $statut_u$ or $dist_{(u,CH_u)}$) undergoes a modification caused by the disappearance of a node v. Fig. 12(d) shows that the disappearance of 5 node in the network size 1000 impacts around 6% of nodes.

To better observe the impact of re-clustering as illustrated in Fig. 13, we set a network size at 1000 nodes and we randomly disappear between 1% and 5% of nodes in the network. At the end of re-clustering process, we evaluate the supplementary communication cost, the energy consumption and the percentage of impacted nodes. Fig. 13(a) shows the supplementary re-clustering cost in terms of exchanged messages and energy consumption compared to clustering cost. We observe that the disappearance until 5% of nodes leads an additional cost in terms of exchanged messages and energy consumption of 15.5%. The main reason is due to the fact that the re-clustering caused by disappearance of 5% nodes does not impact the entire network. In fact, as illustrated in Fig. 13(b), disappearance of 5% nodes impacts around $1/4$ of total number of nodes in the network.

VI. Conclusion and Perspectives

In this paper, we proposed a self-stabilizing distributed energy-efficient and fault-tolerant clustering protocol for heterogeneous wireless sensor networks. This protocol prolongs the network lifetime by minimizing the energy consumption involved in the exchanged of messages. It can be used under different CHs election methods like those investigated in this work. Moreover, our proposed protocol is fault tolerance and adapted to topological changes. We have also compared our algorithm with some of most referenced self-stabilizing solutions.

Simulation results show that in terms of number of messages, energy consumption and clusters distribution, it is better to use the Highest-ID metric for electing CHs. Furthermore, after the occurrence of faults, the re-clustering cost is minimal compared to the clustering cost and faults do not affect the entire network.

As future work, we plan to propose a routing process based on our clustering approach.

Acknowledgment

This research was supported by Regional Council of Champagne-Ardenne and European Regional Development Fund through the CPER CapSec ROFICA project. Simulations are executed on Grid'5000 experimental testbed hosted at the ROMEO HPC Center. We express our thanks to Regional Council of Champagne-Ardenne, European Regional Development Fund, Crid'5000 and ROMEO HPC Center. And finally, we wish to thank the reviewers and editors for their valuable suggestions and expert comments that help improve the contents of this paper.

References

[1] M. Ba, O. Flauzac, R. Makhloufi, F. Nolot, and I. Niang, "Evaluation Study of Self-Stabilizing Cluster-Head Election Criteria in WSNs," in *Proceedings of the 6th International Conference on Communication Theory, Reliability, and Quality of Service*, CTRQ'13, pp. 64–69, 2013.

[2] I. Akyildiz, W. Su, Y. Sankarasubramaniam, and E. Cayirci, "Wireless sensor networks: a survey," *Computer Networks: The International Journal of Computer and Telecommunications Networking*, vol. 38, no. 4, pp. 393 – 422, 2002.

[3] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Communications of the ACM*, vol. 43, no. 5, pp. 51–58, 2000.

[4] J. Yu, Y. Qi, G. Wang, and X. Gu, "A cluster-based routing protocol for wireless sensor networks with nonuniform node distribution," *AEU - International Journal of Electronics and Communications*, vol. 66, no. 1, pp. 54 – 61, 2012.

[5] O. Younis and S. Fahmy, "HEED: a Hybrid, Energy-Efficient, Distributed clustering approach for ad hoc sensor networks," *IEEE Transactions on Mobile Computing*, vol. 3, no. 4, pp. 366–379, 2004.

[6] C. Johnen and L. Nguyen, "Self-stabilizing weight-based clustering algorithm for ad hoc sensor networks," in *Proceedings of the 2nd International Conference on Algorithmic Aspects of Wireless Sensor Networks*, ALGOSENSORS'06, pp. 83–94, 2006.

[7] H. Liu, P.-J. Wan, and X. Jia, "Fault-tolerant relay node placement in wireless sensor networks," in *Computing and Combinatorics*, Lecture Notes in Computer Science, vol. 3595, pp. 230–239, 2005.

[8] W. Zhang, G. Xue, and S. Misra, "Fault-tolerant relay node placement in wireless sensor networks: Problems and algorithms," in *Proceedings of the 26th IEEE International Conference on Computer Communications*, INFOCOM'07, pp. 1649–1657, 2007.

[9] B. Hao, J. Tang, and G. Xue, "Fault-tolerant relay node placement in wireless sensor networks: formulation and approximation," in *Proceedings of the Workshop on High Performance Switching and Routing*, HPSR'04, pp. 246–250, 2004.

[10] H. Liu, A. Nayak, and I. Stojmenovi, "Fault-tolerant algorithms/protocols in wireless sensor networks," in *Guide to Wireless Sensor Networks*, Computer Communications and Networks, pp. 261–291, 2009.

[11] C. Johnen and F. Mekhaldi, "Self-stabilization versus robust self-stabilization for clustering in ad-hoc network," in *Proceedings of the 17th international conference on Parallel processing*, Euro-Par'11, pp. 117–129, 2011.

[12] E. Caron, A. K. Datta, B. Depardon, and L. L. Larmore, "A self-stabilizing k-clustering algorithm for weighted graphs," *Journal of Parallel and Distributed Computing*, vol. 70, no. 11, pp. 1159–1173, 2010.

[13] A. K. Datta, S. Devismes, and L. L. Larmore, "A Self-Stabilizing O(n)-Round k-Clustering Algorithm," in *Proceedings of the 28th IEEE International Symposium on Reliable Distributed Systems*, SRDS'09, pp. 147–155, 2009.

[14] O. Flauzac, B. S. Haggar, and F. Nolot, "Self-stabilizing clustering algorithm for ad hoc networks," in *Proceedings of the 5th International Conference on Wireless and Mobile Communications*, ICWMC '09, pp. 24–29, 2009.

[15] C. Johnen and L. Nguyen, "Self-stabilizing construction of bounded size clusters," *International Symposium on Parallel and Distributed Processing with Applications*, pp. 43–50, 2008.

[16] C. Johnen and L. H. Nguyen, "Robust self-stabilizing weight-based clustering algorithm," *Theoretical Computer Science*, vol. 410, no. 6-7, pp. 581 – 594, 2009.

[17] N. Mitton, A. Busson, and E. Fleury, "Self-organization in large scale ad hoc networks," in *Proceedings of the 3rd Annual Workshop Mediterranean Ad Hoc Networking*, MED-HOC-NET, 2004.

[18] N. Mitton, E. Fleury, I. Guerin Lassous, and S. Tixeuil, "Self-stabilization in self-organized multihop wireless networks," in *Proceedings of the 2nd International Workshop on Wireless Ad Hoc Networking*, ICDCSW '05, pp. 909–915, 2005.

[19] A. Amis, R. Prakash, T. Vuong, and D. Huynh, "Max-min d-cluster formation in wireless ad hoc networks," in *Proceedings of the 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, INFOCOM'00, pp. 32–41, 2000.

[20] M. Ba, O. Flauzac, B. S. Haggar, F. Nolot, and I. Niang, "Self-Stabilizing k-hops Clustering Algorithm for Wireless Ad Hoc Networks," in *Proceedings of the 7th ACM International Conference on Ubiquitous Information Management and Communication*, IMCOM'13, pp. 38:1–38:10, 2013.

[21] D. J. Baker and A. Ephremides, "The architectural organization of a mobile radio network via a distributed algorithm," *IEEE Transactions on Communications*, vol. 29, no. 11, pp. 1694–1701, 1981.

[22] Y.-F. Wen, T. A. F. Anderson, and D. M. W. Powers, "On energy-efficient aggregation routing and scheduling in IEEE 802.15.4-based wireless sensor networks," *Wireless Communications and Mobile Computing*, 2012.

[23] I. G. Shayeb, A. H. Hussein, and A. B. Nasoura, "A survey of clustering schemes for mobile ad-hoc network (MANET)," *American Journal of Scientific Research*, pp. 135–151, 2011.

[24] M. Chatterjee, S. K. Das, and D. Turgut, "WCA: A Weighted Clustering Algorithm for Mobile Ad Hoc Networks," *Cluster Computing*, vol. 5, no. 2, pp. 193–204, 2002.

[25] C.-C. Chiang, M. Gerla, and L. Zhang, "Forwarding Group Multicast Protocol (FGMP) for multihop, mobile wireless networks," *Cluster Computing*, vol. 1, no. 2, pp. 187–196, 1998.

[26] M. Gerla and J. T.-C. Tsai, "Multicluster, mobile, multimedia radio network," *Wireless Networks*, vol. 1, no. 3, pp. 255–265, 1995.

[27] W. Choi and M. Woo, "A Distributed Weighted Clustering Algorithm for Mobile Ad Hoc Network',' in *Porceedings of the International Conference on Internet and Web Applications and Services/Advanced*, AICT-ICIW '06, 2006.

[28] M. R. Brust, A. Andronache, and S. Rothkugel, "WACA: A hierarchical weighted clustering algorithm optimized for mobile hybrid networks," in *Proceedings of the 3rd International Conference on Wireless and Mobile Communications*, ICWMC'07, pp. 27–37, 2007.

[29] M. Ba, O. Flauzac, R. Makhloufi, F. Nolot, and I. Niang, "Comparison between self-stabilizing clustering algorithms in message-passing model," in *Proceedings of the 9th International Conference on Autonomic and Autonomous Systems*, ICAS'13, pp. 27–32, 2013.

[30] A. Varga and R. Hornig, "An overview of the OMNeT++ simulation environment," in *Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems*, Simutools '08, pp. 60:1–60:10, 2008.

[31] SNAP: Stanford Network Analysis Platform. [Online]. Available: http://snap.stanford.edu, 2013.

[32] F. Cappello and et *al.*, "Grid'5000: A large scale and highly reconfigurable experimental grid testbed," *International Journal of High Performance Computing Applications*, vol. 20, no. 4, pp. 481–494, 2006.

[33] W. R. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, 2000.

[34] J. Wang, J.-U. Kim, L. Shu, Y. Niu, and S. Lee, "A distance-based energy aware routing algorithm for wireless sensor networks," *Sensors*, vol. 10, no. 10, 2010.

[35] J. Wang, J. Cho, S. Lee, K. C. Chen, and Y. K. Lee, "Hop-based energy aware routing algorithm for wireless sensor networks," *IEICE Transactions on Communications*, vol. 93-B, no. 2, pp. 305–316, 2010.

[36] J. C. Hou, N. Li, and I. Stojmenovic, *Topology construction and maintenance in wireless sensor networks*, pp. 311–341, 2005.

[37] S. Bandyopadhyay and E. Coyle, "An energy efficient hierarchical clustering algorithm for wireless sensor networks," in *Proceedings of the 32nd Annual Joint Conference of the IEEE Computer and Communications*, INFOCOM'03, pp. 1713–1723, 2003.

[38] M. Nasim, S. Qaisar, and S. Lee, "An energy efficient cooperative hierarchical mimo clustering scheme for wireless sensor networks," *Sensors*, vol. 12, no. 1, pp. 92–114, 2011.

[39] J. Chen, C.-S. Kim, and F. Song, "A distributed clustering algorithm for voronoi cell-based large scale wireless sensor network," in *Proceedings of the 2010 International Conference on Communications and Mobile Computing*, CMC '10, pp. 209–213, 2010.

[40] H. Attiya and J. Welch, *Distributed computing: fundamentals, simulations, and advanced topics*, Wiley series on parallel and distributed computing, 2004.

[41] G. Tel, *Introduction to Distributed Algorithms*. Cambridge University Press, 2000.