Resiliency in Distributed Sensor Networks for PHM of the Monitoring Targets

J. Bahi[1], W. Elghazel[2], C. Guyeux[1], M. Haddad[3], M. Hakem[1], K. Medjaher[2], and N. Zerhouni[2]

August 23, 2016

Abstract

In condition-based maintenance, real-time observations are crucial for on-line health assessment. When the monitoring system is a wireless sensor network, data loss becomes highly probable and this affects the quality of the remaining useful life prediction. In this paper, we present a fully distributed algorithm that ensures fault tolerance and recovers data loss in wireless sensor networks. We first theoretically analyze the algorithm and give correctness proofs, then provide simulation results and show that the algorithm is (i) able to ensure data recovery with a low failure rate and (ii) preserves the overall energy for dense networks.

1 Introduction

In a monitoring activity, the sensor nodes are placed on/around the monitored target to collect measurements of relevant parameters, such as temperature. These measurements will help evaluate the system's current state of health, diagnose the degree of its severity, and extrapolate the result in the future to estimate when the system is more likely to fail. The goal from this activity is to schedule maintenance in a way that prevents system failure and shutdown. To guarantee the efficiency of this process, the accuracy of on-line measurements is a crucial requirement. Consequently, the Wireless Sensor Network (WSN) used in the monitoring needs to be dependable.

Avizienis [1] defined system dependability as "the ability of a system to avoid failures that are more frequent or more severe, and outage durations that are longer, than is acceptable to the users". A dependable network should be able to deliver a correct service (forwards measurements to the base station) and makes sure that failed components will not lead to a network failure. Dependability of WSNs is a property that integrates the attributes needed for the application to be justifiably trusted. These attributes include availability and reliability.

A network failure can be caused by a number of triggers such as: packet loss, node failure, energy exhaustion, packet interference... The network is considered available if its downtime is very limited, either due to few failures or to quick restarts when a failure takes place [2, 3]. A reliable network is able to continuously deliver a correct service. The reliability can be computed as the probability that a network functions properly during a time interval [3, 4].

Most of the research works solved the problem of network reliability through retransmission and redundancy mechanisms [4].
The acknowledgment mechanism is employed by the receiver to notify the sender of the reception status. If the packet fails to arrive to its destination, the sender keeps on re-sending it until the transmission is successful [5, 6, 7, 8]. Unfortunately, this solution does not respect the energy constraints of WSNs, since packet transfer consumes the highest amount of energy in the network. Reliability can also be introduced via data redundancy mechanisms. A packet is transmitted in multiple copies using different routes as a backup plan in case one of the routes fails [9, 10, 11]. However, this solution results in unnecessary transmissions and therefore does not improve energy consumption WSNs.

In the context of of extending the network's lifetime, a possible solution is to maintain a minimum number of sensor nodes in an active mode [12, 13, 14]. Although this seems to solve the energy problem, other issues arise:

- How can we ensure a minimum coverage rate?
- How can we reduce the loss of data?
- How can we avoid unnecessary packet forwards?

In this study, the number of awake sensor nodes is kept to a minimum; enough to ensure coverage rate. The probability of nodes awakening is updated following two variables: time and failure rate. Data sensed by a sensor is copied on its neighbors, and will only be retrieved when the active node has failed. this mechanism avoids unnecessary packet forwards and therefore preserves the overall energy. The remainder of this paper is organized as follows. Section 2 presents some of the existing work in WSN reliability. In Section 3, we describe our algorithm. The simulation results are shown and discussed in Section 4.

2 Related work

Reliability is an important attribute for WSNs dependability, and it means that the network should be able to continuously deliver a correct service. In order to attain reliability in WSNs, sensing coverage and sensing level need to be considered. The sensing coverage refers to the integrated sensing area which is monitored by at least one sensor node. As for the sensing level, it refers to the number of sensor nodes being able to detect a new event when it takes place [15]. Choi *et al.* argue that existing node scheduling schemes focus on the minimum sensing level for the coverage problem and neglect the fault tolerance issues [15]. In one hand, the minimum sensing level is an NP-complete problem. On the other hand, it cannot be preserved when nodes start to fail. Therefore, the authors propose the Fault-tolerant Adaptive Node Scheduling

(FANS) algorithm, which efficiently handles the degradation of the sensing level. The algorithm designates a set of backup nodes for each active node. If the latter fails, the predesignated set of backup nodes activate themselves to replace it and to restore the lowered sensing level. FANS requires a small number of backup nodes and a small amount of control messages. In [16], Chen *et al.* study fault tolerant out-of-band monitoring for WSNs. They aim at placing a minimum number of monitors in a sensor network in a way that all sensor nodes are monitored by k distinct monitors, and each monitor serves at most w sensor nodes. The authors first prove that this problem is NP-hard and then propose three algorithms providing near optimal solutions.

Battery level, broken links, and communication failures have an impact on the Quality of Service (QoS) of WSNs. This leads to consequences varying from disturbing the traffic in the network to completely interrupting it. Geeta *et al.* [17] propose an Active node-based Fault Tolerance using Battery power and Interference model (AFTBI) to identify the faulty nodes in WSNs. Fault tolerance against low battery power is assured through a hand-off mechanism where the faulty node selects the neighbor with highest battery level and transfers all the services towards it. To reduce interference signal, a dynamic power level mechanism is introduced, where the power of a node is adjusted automatically with regards to its current state (active or asleep). Simultaneous transmissions can be avoided if the nodes are only allowed to transmit data within a time slot. Lee and Choi [18] tackle the same problem by identifying and isolating the faulty sensor nodes in the network. Sensed data is compared among neighbors to determine its accuracy. Once the predetermined fault threshold is reached, the node in question is isolated from the diagnosis process; a faulty node can be included in data transferring but not data sensing. Transient faults in communication and sensor reading are tolerated by using the time redundancy mechanism. The drawback of this solution is that faults are assumed to be only related to the sensing activity, excluding other sources of failure.

Energy in WSNs can also be preserved through lifetime optimization. The authors in [19] leverage prediction to prolong the network lifetime, by exploiting temporal-spatial correlations among the data sensed by different sensor nodes. Based on Gaussian Process, the authors formulate the issue as a minimum weight sub-modular set cover problem and propose a centralized and a distributed truncated greedy algorithms (TGA and DTGA). They prove that these algorithms obtain the same set cover. Lifetime optimization using knowledge about the dynamics of stochastic events has been studied in [20]. The authors presented the interactions between periodic scheduling and coordinated sleep for both synchronous and asynchronous dense static sensor network. They show that the event dynamics can be exploited for significant energy savings, by putting the sensors on a periodic on/off schedule. In [13], the authors design a polynomial-time distributed algorithm for maximizing the lifetime of the network. They proved that the lifetime attained by their algorithm approximates the maximum possible lifetime within a logarithmic approximation factor. Zhang *et al.* [21] presented a stochastic sensing algorithm to reduce energy consumption through node scheduling. They used data correlation between nodes to reduce error rate by adjusting duty cycle of faulty sensors. Their algorithm conserves 60 % of energy as compared to other solutions, while confining sensing error within specified error tolerance. In [22], He *et al.* use actors to allocate spare sensors to sensor-deficient regions or to relocate sensors from sensor-abundant regions to sensor-deficient regions. They introduce a baseline centralized greedy algorithm for sensor allocation, where global sensor information is communicated to obtain the optimal solution. The works cited here focus on a periodic schedule for turning the sensors on and off.

Data collection delay and reliability need to be considered in scheduling algorithms for WSNs. Zhang *et al.* [23] claim that existing algorithms have not solved these two problems effectively. The authors propose the Fault-Tolerant Scheduling (FTS) algorithm, where each sensor node detects the environment and generates some sensing data at regular intervals. The algorithm helps surviving network malfunction by switching the parent of a sensor node to its backup parent. The simulation results show that FTS has a short data collection time and high fault tolerance. Feng *et al.* [24] considered the problem of efficient data aggregation in WSNs by putting in place amendment strategies in case of failures. Their solution needs local information to repair the aggregation tree and automatically reschedules nodes for interference free aggregation after the amendment. Cheng *et al.* [25] present STCDG, an efficient data gathering scheme based on matrix completion. STCDG takes advantage of the low-rank feature instead of sparsity, thereby avoiding the problem of having to be customized for specific sensor networks. They exploit the presence of the short-term stability feature in sensor data, which further narrows down the set of feasible readings and reduces the recovery errors significantly. Furthermore, STCDG avoids the optimization problem involving empty columns by first removing the empty columns and only recovering the non-empty columns, then filling the empty columns using an optimization technique based on temporal stability.

To preserve the overall energy in the network, sensor nodes are on a periodic schedule where they are switched on only when the sensing level is decreased. An optimal schedule needs to take nodes failure rate and the elapsed run-time into consideration. When the failure rate is small, wakening the nodes too often would only waste energy. As we go further in time, nodes start to exhaust their energy supply and this is when they start to fail. A combination a node failure rate and elapsed time would give us a better indication of the optimal nodes wakening schedule.
Maintaining the sensing level considerably reduces the amount of packet loss, yet it does not completely prevent its occurrence. A sudden node failure will result in the permanent loss of the held data packet, unless a redundancy mechanism is put in place. In the context of reducing energy consumption, the redundancy solution should be avoided and replaced by other solutions which do not include unnecessary packet transmission.

In this paper, we present a fault-tolerant data collection algorithm. This algorithm preserves energy consumption by only maintaining the necessary set of nodes in the active mode to ensure the minimum coverage level, while considering nodes failure rate. It is also able to recover data loss when a node fails before forwarding the data towards the base station. This algorithm is described in the next section.

3 The proposed algorithm

To cope with fault tolerance and data survivability, a fully distributed algorithm is presented and theoretically analyzed. Our algorithm seeks to cover data loss by maintaining a necessary set of working nodes and recovering failed ones when needed. We suppose that we are in the case of high density networks, and not all nodes participate in the network's service. Some nodes are in an idle state because their targets are actually covered by working sensors. We consider that these idle sensors wake up periodically to check for eventual node failures and therefore ensure their targets' coverage. In case of failures, they decide to switch to active mode and therefore initiate the recovery process to retrieve the data of the failed nodes. However, during the network's service, how can we handle the case where two (or more) sleeping nodes, would realize at the same time that the working neighbor is down?

Indeed, two neighboring sensor nodes may be elected at the same time step, and the recovery process of two neighboring nodes may be the same. This paper aims at filling this gap by proposing an efficient node failure recovery scheme in order to allow sensor networks to gracefully degrade in performance instead of failing unpredictably.

In the following, we first focus on the legitimate state formulation and next, we present the algorithm which consists in only three rules and give the correctness proofs.

3.1 Problem formulization

Let $G = (V, E)$ be the graph modeling the sensor network, with $|V| = n$ and $|E| = m$. We assume sensor node identifiers to be unique. We recall that sensor node identifier is unique if and only if $i.Id \neq j.Id$ holds for each $i, j \in V (i \neq j)$. A sensor node can be in one of these three states: *failed, working,* or *probing.* Every node i in the network has to maintain the following data structure:

- D_i: the sensed data by node i. Each time a node updates D_i, it sends/replicates the newly sensed data to/on its neighbors.

- P_i: the parity information on node i. It is the result of the combination of the replicated information of its neighbors.

We considered two different scenarios for the parity information. In the first scenario, there are no memory constraints. Each new data is saved on a different memory register, and we used the SUM function for data collection. In the contrary, all information must be saved on the same memory register when it comes to the second scenario. So, we used the XOR function to preserve memory space.

Let $T = t_1, t_2, ..., t_k$ be the set of monitoring targets to be covered and $S = 1, 2, ..., n$ the set of sensor nodes. Each target in T has to be covered by at least one sensor node in S. We call Γ_u the set of neighbor- sensors of target $t_u, 1 \leq u \leq k$. Each neighbor-sensor $j \in \Gamma_u$ is capable of monitoring the target t_u, formally:

$$\forall j \in \Gamma_u : ds(t_u, j) \leq R_s, \Gamma_u \subseteq S, t_u \in T,$$

where $ds(t_u, j)$ denotes the distance between points t_u and sensor j.

Let N_i be the initial set of neighbors of node i and $d_i = |N_i \setminus \Gamma_u(i) \setminus i|$, the number of its working neighbors. As the number of failures goes up with time, we let d_i^* be the dynamic number of alive neighbor nodes. We denote by D^k the set of $d_i + 1$ replicas of data D_i. Also, we denote by $s\ (D^k)$ the sensor node to which data-replica D^k is assigned, for $1 \leq k \leq d_i + 1$ and by $\hat{s}\ (D^k)$ the elected sensor node who recovers D_i if node i fails. The data are replicated on different nodes (space exclusion, see Lemma 1) since the goal is to achieve data survivability even if some node failures occur in the network.
We say that a sensor node i is independent if
$i.state = working \wedge (\forall j \in \Gamma_u(i))(j.state = sleeping \vee probing \vee failed)$
and that i is dominated if
$(i.state = sleeping \vee probing) \wedge (\exists j \in \Gamma_u(i))(j.state = working)$
The legitimate state (let us denote it Σ) of the network is then expressed as follows:
$\forall i \in V : i.state = failed$
$\Rightarrow ((\exists \hat{s}, \hat{s}' \in \Gamma_u(i))(\hat{s}.state = \hat{s}'.state = working) \Rightarrow (\hat{s}(D_i^k) = \hat{s}'(D_i^k)))$

In other words, each data loss is recovered by at most one working sensor node.

3.2 The algorithm

When a sleeping node wakes up, it sends a *probe-request* message to check if there exist working nodes in its vicinity. If no working nodes, it recovers the lost data of the failed node and starts to operate in the active mode; otherwise, it sleeps again. Nodes are initially in the sleeping mode. Each node sleeps for an exponentially distributed time generated according to a probability density function (PDF) $f(t) = \lambda e^{-\lambda t}$, where λ is the probing rate of the sensor node and t denotes its sleeping time duration.

Upon detecting an eventual failure, a probing node i updates its actual probing rate λ_i by taking into account the dynamic number of alive neighbors $d_i^* : \lambda_i^{new} \leftarrow \lambda_i \cdot \frac{d_i}{d_i^*}$. Then, a new sleeping period is generated by using the new computed parameter λ_i^{new} according to the PDF function: $f(t) = \lambda^{new} e^{-\lambda^{new} t}$. The following notations are also given for the predicates of node i

- $W(i)$: working neighbor: $\exists j \in \Gamma_u(i), i.state = working$

- $W^*(i)$: working neighbor with lower Id: $\exists j \in \Gamma_u(i), j.state = probing \wedge i.Id > j.Id$
- $F(i)$: failed neighbor: $\exists j \in \Gamma_u(i), j.state = failed$
- $P^*(i)$: probing neighbor with lower Id: $\exists j \in \Gamma_u(i), j.state = probing \wedge i.Id > j.Id$

The proposed algorithm uses the following three rules:
$r1$:

```
if ($i.state = probing \wedge (P^*(i) \vee W(i))$) then

    if $P^*(i)$ then
        $\lambda_i^{new} \leftarrow \lambda_i.\frac{d_i}{d_i^*}$
    end if
    $i.state \leftarrow sleeping$
end if
```

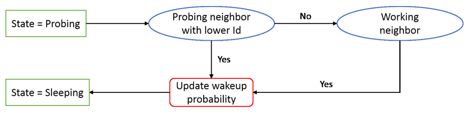


Figure 1: Algorithm rule 1.

$r2$:

```
if $i.state = probing \wedge (\neg W(i) \wedge \neg P^*(i) \vee F(i))$ then
    if $F(i)$ then
        if memory constraint then
            $D_i \leftarrow P_z \bigoplus_{k \in N_z \backslash \Gamma_u(i), k \neq j} D_k$ (*$F(i) = F(z) = j$*)
        else
            $D_i \leftarrow D_k, k \in N_z \backslash \Gamma_u(i), k \neq j$ (* $k$ is chosen randomly*)
        end if
    end if
    $i.state \leftarrow working$
end if
```

$r3$:

```
if ($i.state = working \wedge W^*(i)$) then
    $i.state \leftarrow sleeping$
end if
```

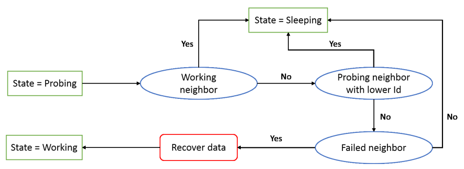


Figure 2: Algorithm rule 2.

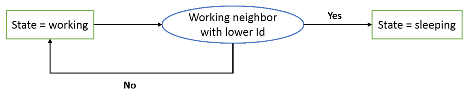


Figure 3: Algorithm rule 3.

3.3 Correctness proofs

In this section, we detail properties of our fault tolerant algorithm, and express its validity/convergence. We assume that links are trustworthy/flawless and lossless.

Lemma 1. *A sensed data D_i is guaranteed to survive in the presence of d_i permanent faults if and only if $s(D_i^k) \neq s(D_i^{k'})$, for $1 \leq k,\ k' \leq d_i + 1$.*

Proof. If d_i nodes fail, then there is $s(D_i^u), 1 \leq u \leq d_i + 1$ which did not fail, and therefore D_i^u will be recovered successfully from $s(D_i^u)$ since there are $d_i + 1$ copies of D_i assigned to $d_i + 1$ different nodes. However, if there is a sensor node $s(D_i^k), 1 \leq k \leq d_i + 1$, such that $s(D_i^k) = s(t_i^u) = s^*$ and s^* fails, then neither D_i^k nor D_i^u can be recovered successfully. □

Lemma 2. *If at most d_i neighbors crash down for any sensor node $i \in V$ in the network, then the algorithm is valid and resists to eventual node failures.*

Proof. The proposed algorithm is based on replication scheme with space exclusion. Thus, according to Lemma 1, each data is replicated $d_i + 1$ times onto $d_i + 1$ distinct sensor nodes. We have at most d_i node failures at the same time. So at least one copy of each data is recovered from a fault free node. □

Lemma 3. *If a node changes to the working state by $r2$, then it remains in its state and will never execute a rule again until an eventual failure takes place.*

Proof. Let i be a sensor node that executes $r2$. According to the preconditions of all rules, node i can execute only rule $r3$ in the next round. However, in order to do so, one of its neighbors would have to switch to *working* state following $r2$. This is impossible as long as node i is in the *working* state. Thus, node i will never execute a rule again. If node i is down, it remains in its state (fail-stop failure). □

Lemma 4. *If a sensor node is enabled by rule r2, then each one of its neighbors will execute at most one more rule until their next wake-up/probing, and this rule will be r1.*

Proof. Let i be a node that executes $r2$. When node i changes to *working* state, all its neighbors are either in *sleeping*, *probing*, or $failed$ state. So, we have three possible scenarios: i) neighbors in *sleeping* state: there is no conflict in this case. ii) neighbors with *probing* state: these neighbors have a higher Id than i. iii) failed neighbors will remain in their state until their recovery. □

Lemma 5. *Every sensor node is either independent, dominant, or failed.*

Proof. From the point of view of node i, we have three scenarios:

- if node i is in the *working* state and is not *independent*, then i may execute rule $r3$.
- if node i is in the *sleeping* $\vee$ *probing* state and is not *dominated*, then node i may execute rule $r2$.
- if node i is in the $failed$ state, then node i will remain in its state until its recovery.

□

Lemma 6. *When a node is not failed $\vee$ sleeping, it can make at most two moves.*

Proof. By Lemma 3 and Lemma 4, each rule can be executed at most once by a node. Hence, the only case a node makes two moves is when it executes $r3$ then $r2$ with a *working* state. □

Theorem 1. *With respect to the legitimate state Σ of the network, the proposed algorithm converges within 2n moves.*

Proof. This follows from Lemma 1 to Lemma 6. □

3.4 Message complexity analysis

In the following, we give an Upper-Bound of the actual number of probe/reply messages exchange during the network's lifetime task.

Theorem 2. *The number of probe/reply messages involved by the algorithm is at most:*

$$O\left(n\ m \times \max_{i} \frac{t_i^{R_i}}{\Delta_i}\right), 1 \leq i \leq n$$

where, n is the number of nodes, m is the number of virtual communication links, $t_i^{R_i}$ is the reliable lifetime of node i and Δ_i is the smallest sleeping period time of node i.
This bound is attainable.

Proof. (a)- The reliable lifetime $t_i^{R_i}$, of the node $i, 1 \leq i \leq n$ for a specified reliability R_i, starting the mission at age 0, is computed as follows:
$R_i = 1 - F(t_i^{R_i}) = e^{-\lambda t_i^{R_i}} \Rightarrow ln\ R_i = -\lambda t_i^{R_i} \Rightarrow t_i^{R_i} = -\frac{1}{\lambda}\ ln\ R_i$
This is the lifetime during which the sensor node i will be functioning successfully with a reliability of R_i.
According to node's sleeping periods subdivisions of the time, we have:
$0 = t_o < t_1 < t_2 < ... < t_k = t$. Let $\Delta_p = [t_{p-1}, t_p[, 1 \leq k$ denote the p^{th} sleeping period time. Since the number of failures goes up, the sleeping time period decreases with time. This implies that the probing process of node i costs at most $O\left(\frac{t_i^{R_i}}{\Delta_i}\right), 1 \leq i \leq n, \Delta_i = min \Delta_p, 1 \leq p \leq k$. In addition, for each probing message issued from node i, we may have the corresponding reply messages from its working neighbors. This cost is at most $O(|N_i|)$. Therefore, from the point of view of node i, the number of probe/reply messages is at most $O\left(|N_i| \times \frac{t_i^{R_i}}{\Delta_i}\right)$.
Finally, summing up for the whole n sensor nodes, the algorithm's message cost is at most $O\left(\Sigma_{i=1}^{n} |N_i| \times \frac{t_i^{R_i}}{\Delta_i}\right) \leq O\left(n\ m \times max_i \frac{t_i^{R_i}}{\Delta_i}\right), 1 \leq i \leq n$

(b)- To see that this bound is really attainable, consider a linear chain graph of only two sensor nodes s_1 and $s_2 (n = 2)$. We need to orchestrate the involved communications between these nodes in time. Assume that s_1 is working and s_2 is the passive state. If $t_1^{R_1} = t_2^{R_2}$ (s_1 and s_2 start functioning and fail at the same time), then the whole number of probe-message issued from s_2 is $\frac{t_2^{R_2}}{\Delta_2}$, where Δ_2 is the constant sleeping time period of s_2. Since both s_1 and s_2 have the same life for which nodes will be functioning successfully, node s_1 will reply for each probing message issued from s_2. As a result, the whole number of involved probe-request/reply message before the failure of s_1 and s_2 is $n\ m \times \frac{max_i t_i^R}{min_i min_j \Delta_{ji}} = 2 \times \frac{t_2^{R_2}}{\Delta_2}$

□

4 Simulation results

In this section, we discuss some results through simulations. We consider a flat grid topology of 10 by 10 *i.e.* 100 monitoring zones. We vary the number of sensors between 200 and 1600 nodes. Since sensors are uniformly distributed in the monitoring area, the density of sensors at each zone varies between 2 and 16.

The performance evaluation considers five aspects: (i) Network lifetime evolution; (ii) Failure rate: that is the ratio of information recovery attempts that did not succeed; (iii) Effective monitoring time: this measure is related to the time between the death of the active node in a monitoring zone and its replacement; it is expressed in (%); (iv) Total number of messages; and (v) Number of awakenings per inactive sensors.

Our simulations are performed in two different settings: the first setting sets a low wake-up rate but enough to keep the monitoring time ratio higher than 50% in almost all configurations (see Figure 4); while the second setting considers a wake-up rate four times higher than the rate in the previous setting. This allowed us to reach monitoring time ratios up to 90% (see Figure 5).

These two settings were put in place in order to test and compare two different solutions for data collection. In the first scenario, we assume that there is no constraint related to memory capacity of sensor nodes. Therefore, each sensor is able to save data received from all nodes in its neighborhood on a different memory register (this method is called SUM). As for the second scenario, we aim for preserving the memory space. Thus, we suppose that each node has one memory register that is available to save all information received from all its neighboring nodes; every new data packet is added in the register using the function XOR. We can notice that the second method for data saving (XOR) is highly sensitive to any neighboring node failure. In fact, the failure of a neighbor induces a corruption of the calculation of the data needed for the recovery process. For this reason, the coverage rate needs to be high enough for this solution to work. Consequently, the XOR method is only implemented in the 4x version of our simulation settings.

4.1 Wake-up rate = 1x

In this section, only the SUM method is implemented. The nodes in the network are designed to fail randomly. This failure rate varies from 0% to 8% by a pitch of 2%.

In Figure 4(a), we can observe that the network's lifetime increases in an almost linear manner. For a small network, the number of times a sensor node receives a wake-up message is higher comparing to the same number when the network is larger. Therefore, the more nodes are participating in the coverage process, the less energy is consumed per node and the overall energy in the network is preserved. The overall energy level has an impact on the recovery process. In fact, when the energy level start to go down, the number of nodes able to cover a given area is reduced. The remaining nodes will receive an increasing number of wake-up messages, and when they fail the number of replacements is continuously decreased. Eventually, some zones will no longer be covered. Thus, data recovery rate increases when the network is larger. Even though failures are less impacting for a dense network, the failure rate is low even for a small network (see Figure 4(b)). The coverage rate (illustrated in Figure 4(c)) is more successful when the density of the network grows. Nevertheless, it remains between the values of 45% and 55% depending on the settings. Indeed, as discussed above, the number of wake-up messages in the network is higher with the growth of nodes number and therefore more energy is dissipated. On the other hand, when the number of nodes is small, even though the number of wake-up messages is reduced, nodes fail faster as the number of node replacements is small. Consequently, there is no huge difference in the coverage rate. Still, a dense network guarantees a better coverage rate. The total number of exchanged messages in the network will only grow with the increased number of nodes in the network as shown in Figure 4(d). In fact, each node will copy its sensed data onto each one of its neighbors. So when the number of nodes increases, the number of messages increases accordingly. The number of wake-ups per node illustrated in Figure 4(e) follows a logarithmic form. Sensor nodes periodically wake up to verify if there zone is being covered by an active node. The wake-up rate follows a probability function that is updated considering node failure. So, the number of these messages highly depends on the number of nodes failure. When the probability of node failure increases, the number of nodes in the network is decreased and thus the total number of wake-ups.

4.2 Wake-up rate = 4x

In this section, both of SUM and XOR methods are tested. Since all the curves are similar, except for failure of the recovery process, only the figure corresponding to the latter illustrates the comparison between both methods. Nodes failure rate are fixed to 0% and 8% only (the two extreme cases from the previous configuration).

Comparing to the previous configuration, the network's overall lifetime has decreased. Considering that the wake-up rate here is 4 times more frequent, it is normal that network consumes more energy in this setting (see Figure 5(a)). Nevertheless, the different zones coverage rate illustrated in Figure 5(c) was considerably and understandably improved. The failure of the recovery process in Figure 5(b) remains very low with the absence of memory constraints, and even lower comparing to the previous configuration. In the contrary, the XOR function appears to be highly sensitive to node failure. When the failure rate reaches 8%, the recovery failure jumps by 30% for a small network and 15% when the network is dense. The total number of exchanged messages is considerably higher than the number in the previous configuration, and this is due to the increased number of wake-up messages. The algorithm also improves the overall energy consumption by only maintaining a necessary set of nodes in the active mode. the rest of the node wake up randomly to check their area and ensure that coverage is performed. This random function is optimized by updating it accordingly to the nodes failure rate.

5 Conclusion

In this paper, we proposed a fully distributed algorithm that seeks to cover data loss by maintaining a necessary set of working nodes and recovering failed ones when needed. Each sensor node copies its data onto neighbors using two different assumptions: (i) in the first one, we suggest that there is no memory constraint and each new information is copied on a different register, and (ii) in the second one we put in place a memory constraint and use the XOR function to add a new data to the common memory register for all data. We also tested two different configurations, where the wake-up rate is 4 times more frequent from one configuration to the other. The performed simulations showed that a more frequent wake-up rate helps improve the quality of the recovery process. Even though the absence of memory constraints facilitates the recovery process, this rate was maintained below 35% for a small network and around 15% for a dense one even in the presence of memory constraints. This algorithm also helps preserving the energy in the network by only maintaining a necessary set of sensor nodes in the active mode. The rest of the nodes wake up randomly to ensure that their area is covered by a sensor node. This random function is optimized by updating it according to the nodes failure rate.

References

[1] Avizienis, A., Lapire, J.-C., and Randell, B. (2000) Fundamental concepts of dependability. Technical Report CS-TR-739. University of Newcastle, England.

[2] Knight, J. (2004) An introduction to computing system dependability. *Proceedings of the 26th International Conference on Software Engineering (ICSE'04)*, Edinburgh, United Kingdom, May 23-28, pp. 730–731. IEEE Computer Society, Washington DC, USA.

[3] Taherkordi, A., Taleghan, M., and Sharifi, M. (2006) Dependability consideration in wireless sensor networks applications. *Journal of Networks*, **1**, 28–35.

[4] Silva, I., Guedes, L., Portugal, P., and Vasques, F. (2012) Reliability and availability of wireless sensor networks for industrial applications. *Sensors*, **12**, 806–838.

[5] Akan, O. B. and Akyildiz, I. F. (2005) Event-to-sink reliable transport in wireless sensor networks. *IEEE/ACM Transactions on Networking*, **13**, 1003–1016.

[6] Zhou, Y., Lyu, M. R., Liu, J., and Wang, H. (2005) Port: A price-oriented reliable transport protocol for wireless sensor networks. *Proceedings of the 16th IEEE International Symposium on Software Reliability Engineering (ISSRE'05)*, Chicago IL, November 8-11, pp. 117–126. IEEE Computer Society, Los Alamitos CA, USA.

[7] Gungor, V. C. and Ozgur B. Akan, O. B. (2006) Dst: Delay sensitive transport in wireless sensor networks. *Proceedings of the 7th IEEE International Symposium on Computer Networks (ISCN'06)*, Istanbul, Turkey, June 16-18, pp. 116–122. IEEE Communications Society, New York, USA.

[8] Iyer, Y. G., Gandham, S., and Venkatesan, S. (2005) Stcp: A generic transport layer protocol for wireless sensor networks. *Proceedings of the 14th International Conference on Computer Communications and Networks (IC-*

CCN), San Diego, California, USA, October 17-19, pp. 449–454. IEEE Computer Society, Washington DC, USA.

[9] Al-Wakeel, S. S. and Al-Swailem, S. A. (2007) Prsa: A path redundancy based security algorithm for wireless sensor networks. *IEEE Wireless Communications and Networking Conference, WCNC*, Hong Kong, China, March 11-15, pp. 4159–4163. IEEE Communications Society, New York, USA.

[10] Dijkstra, E. W. (1974) Self-stabilizing systems in spite of distributed control. *Communications of the ACM*, **17**, 643–644.

[11] Mojoodi, A., Mehrani, M., Forootan, F., and Farshidi, R. (2011) Redundancy effect on fault tolerance in wireless sensor networks. *Global Journal of Computer Science & Technology*, **11**, 34–39.

[12] He, S., Chen, J., Li, X., Shen, X. S., and Sun, Y. (2012) Leveraging prediction to improve the coverage of wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, **23**, 701–712.

[13] He, S., Chen, J., Yau, D. K. Y., Shao, H., and Sun, Y. (2012) Energy-efficient capture of stochastic events under periodic network coverage and coordinated sleep. *IEEE Transactions on Parallel and Distributed Systems*, **23**, 1090–1102.

[14] Kasbekar, G. S., Bejerano, Y., and Sarkar, S. (2011) Lifetime and coverage guarantees through distributed coordinate-free sensor activation. *IEEE/ACM Transactions on Networking*, **19**, 470–483.

[15] Choi, J., Hahn, J., and Ha, R. (2009) A fault-tolerant adaptive node scheduling scheme for wireless sensor networks. *Journal of Information Science and Engineering*, **25**, 237–287.

[16] Chen, X., Kim, Y.-A., Wei, W., Shi, Z. J., and Song, Y. (2009) Fault-tolerant monitor placement for out-of-band wireless sensor network monitoring. *Journal of Information Science and Engineering*, **25**, 237–287.

[17] Geeta, D., Nalini, N., and Biradar, R. (2013) Fault tolerance in wireless sensor networks using hand-off and dynamic power adjustment approach. *Journal of Network and computer Applications*, **36**, 1174–1185.

[18] Lee, M.-H. and Choi, Y.-H. (2008) Fault detection of wireless sensor networks. *Computer communications*, **31**, 3469–3475.

[19] Kasbekar, G. S., Bejerano, Y., and Sarkar, S. (2011) Lifetime and coverage guarantees through distributed coordinate-free sensor activation. *IEEE/ACM Transactions on Networking*, **19**, 470–483.

[20] He, S., Chen, J., Li, X., Shen, X. S., and Sun, Y. (2012) Leveraging prediction to improve the coverage of wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, **23**, 701–712.

[21] Zhang, Q., Fu, L., Gu, Y., Gu, L., Cao, Q., Chen, J., and He, T. (2014) Collaborative scheduling in highly dynamic environments using error inference. *IEEE Transactions on Parallel and Distributed Systems*, **25**, 591–601.

[22] He, S., Chen, J., Cheng, P., Gu, Y., He, T., and Sun, Y. (2012) Maintaining quality of sensing with actors in wireless sensor networks. *IEEE Transactions on Parallel and Distributed Systems*, **23**, 1657–1667.

[23] Zhang, L., Ye, Q., Cheng, J., Jiang, H., Wang, Y., Zhou, R., and Zhao, P. (2012) Fault-tolerant scheduling for data collection in wireless sensor networks. *Proceedings of IEEE GLOBECOM*, Anaheim, CA, December 3-7, pp. 5345–5349. IEEE Communications Society, New York USA.

[24] Feng, Y., Tang, S., and Dai, G. (2011) Fault tolerant data aggregation scheduling with local information in wireless sensor networks. *Science and Technology*, **16**, 451–463.

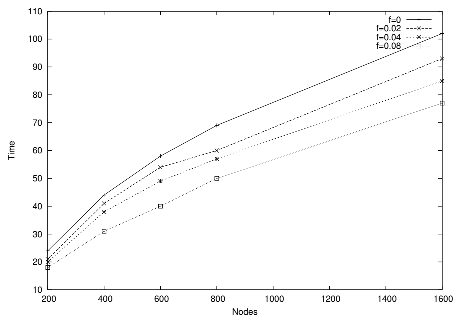
[25] Cheng, J., Ye, Q., Jiang, H., Wang, D., and Wang, C. (2013) Stcdg: An efficient data gathering algorithm based on matrix completion for wireless sensor networks. *IEEE Transactions on Wireless Communications*, **12**, 850–861.

[26] Peng, Y., Dong, M., and Zuo, M. (2010) Current status of machinery prognostics in condition-based maintenance:a review. *International Journal of of Advanced Manufacturing Technology*, **50**, 297–313.

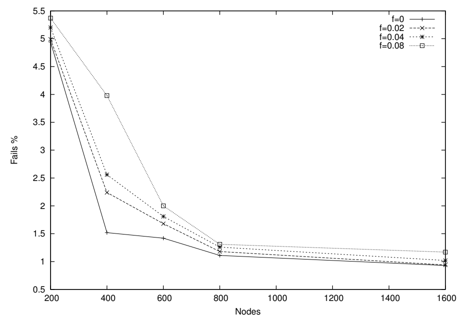
[27] Hu, C., Youn, B., Wang, P., and Yoon, J. (2012) Ensemble of data-driven prognostic algorithms for robust prediction of remaining useful life. *Reliability Engineering and System Safety*, **103**, 120–135.

[28] Heng, A., Zhang, S., Tan, A., and Mathew, J. (2009) Rotating machinery prognostic: State of the art, challenges ahead and opportunities. *Mechanical Systems and Signal Processing*, **23**, 724–739.

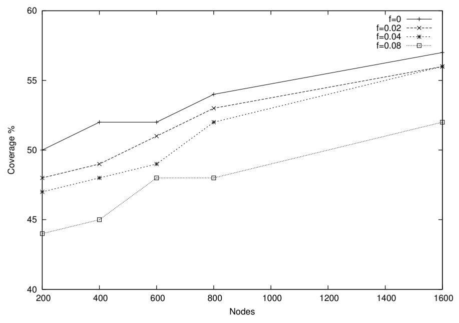
[29] Elghazel, W., Bahi, J., Guyeux, C., Hakem, M., Medjaher, K., and Zerhouni, N. (2015) Dependability of wireless sensor networks for industrial prognostics and health managements. *Computers in Industry*, **68**, 1–15.



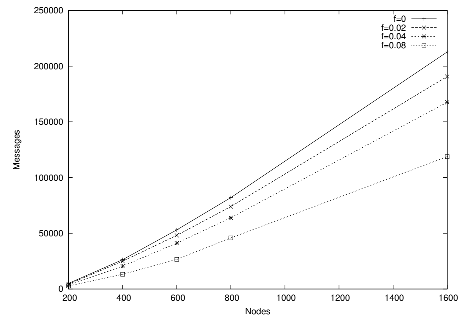
(a) Network's lifetime



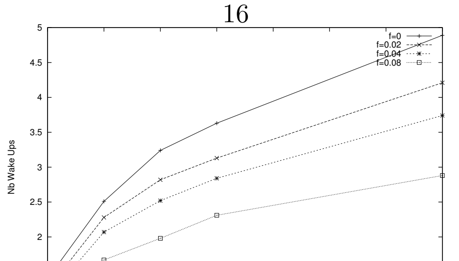
(b) Failures of the recovery process

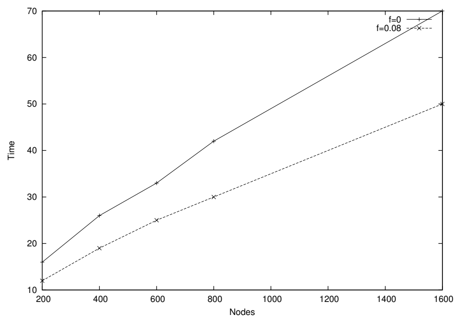


(c) Coverage rate

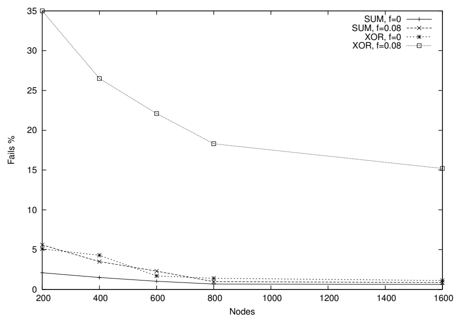


(d) Number of total messages in the network

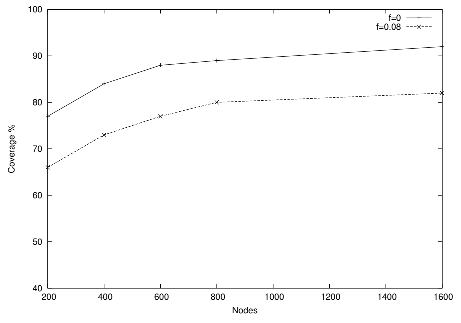




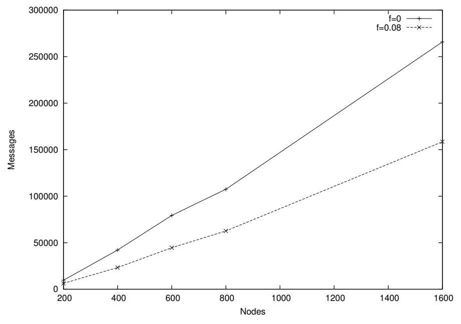
(a) Network's lifetime



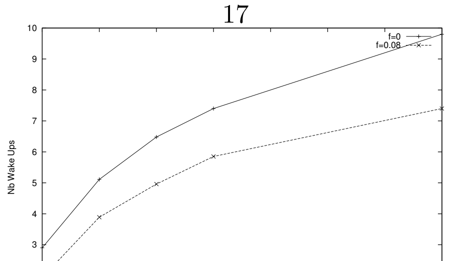
(b) Failures of the recovery process



(c) Coverage rate



(d) Number of total messages in the network


f=0
f=0.08
Nb Wake Ups