# Enhancement of IoT Applications Dependability Using Bayesian Networks

Yasmine Harbi, Zibouda Aliouat, Sarra Hammoudi

# Enhancement of IoT Applications Dependability using Bayesian Networks

Yasmine Harbi, Zibouda Aliouat, and Sarra Hammoudi

LRSD Laboratory, Computer Science Department
Ferhat Abbas University
Setif, Algeria
harbi_yas@yahoo.com, (zaliouat, hammoudi_sarah)@univ-setif.dz

**Abstract.** Sensors play a vital role in Internet of Things (IoT) monitoring applications. However, the harsh nature of the environment influences negatively on the sensors reliability. They may occasionally generate inaccurate measurements when they are exposed to high level of humidity, temperature, etc. Forwarding incorrect data to the base station may cause disastrous decision-making in critical applications. To address this problem, we proposed a new fault-tolerant mechanism based on Bayesian Networks that carefully solve such a problem. The mechanism is called Reliability of Captured Data (RCD); it enables sensors to both detect and recover faulty data. To show the performance of our algorithm, we compared this latter with another recent algorithm called Fault Detection Scheme (FDS) using Network Simulator 2 (NS2). The results showed that our strategy gives remarkable enhancements in terms of fault detection accuracy and fault alarm rate.

**Keywords:** Internet of Things, Wireless Sensor Networks, Smart sensors, Fault tolerance, Bayesian Networks, Reliability.

## 1 Introduction

The IoT refers to the next generation of the Internet [1]. It cuts across many critical applications. Among of these applications are: process automation applications, infrastructure monitoring networks, process control networks, smart building, patient monitoring, oil and gas industry [1][2][3][4].
The IoT is empowered by the proliferation of a myriad number of miniature sensors and actuators. Sensors can efficiently monitor critical environments in diverse domains.
Fault tolerance is one of the critical problems in IoT applications. The problem of missing sensor node and data are inevitable in Wireless Sensor Networks. Failure problems are due to various factors such as power depletion, environmental impact and dislocation of sensor nodes [5]. In such a case, sensors may generate incorrect measurements, cause false alarms and degrade the reliability of collected data [6]. Therefore, it is extremely important to ensure the accuracy of sensed data before decision-making process. A wide variety of classification-based approaches including Bayesian Networks (BNs), Neural Networks (NNs),

and Hidden Markov Models (HMMs) can be used to detect inaccurate data.
To address such a problem, this paper presents a new fault-tolerant mechanism, called Reliability of Captured Data (RCD) that is able to achieve higher data reliability and accuracy. The RCD makes sensors more intelligent, it enables them to efficiently detect the faulty nodes by analyzing the measured data. The algorithm is also able to recover the unreliable data before forwarding it to the base station. The accuracy of the faulty node detection reflects in the diagnosis of the sensed data using Bayesian Networks.
The Bayesian Network is a probabilistic model which enables the sensors to locally predict their current readings based on their own past readings and the current readings of their neighbors [7][8]. The choice of the Bayesian Networks helps to accurately detect the faulty nodes since its rate of misclassification is very low compared to NN and HMM [9]. In addition, the BN requires fewer parameters than the HMM to represent the same information, and provides the spatial-temporal correlation between the sensors [10][11].
To show the performance of our proposal, we have conducted a several series using Network Simulator 2 (NS2). We compared our strategy to a recent one which is called Fault Detection Scheme (FDS) [13]. The results show that our mechanism achieves high amelioration in terms of fault detection accuracy and false alarm rate.
The rest of this paper is organized as follows. Section 2 presents the related works in the context of fault tolerance. Section 3 describes the concept of Bayesian classifier. Section 4 discusses the proposal and the simulation results are presented in Section 5. Section 6 concludes our work.

## 2 Related works

Failure detection and recovery is a crucial task in IoT since sensors are deployed in harsh and unattended environments.
M.Yuvaraja and M.Sabrigiriraj in [12] designed a fault detection and recovery scheme for routing protocols in WSN. The sink periodically broadcasts agent packets to all its neighbors. The non faulty nodes reply with an acknowledgment. The agents form a query path towards the dead or faulty node. To decrease the broadcasting cost overhead, the receiving nodes apply the Random Decision Function (RDF) to be able to take a decision to whether they forward the agent packet or not in order to detect the dead or faulty nodes. Once the faulty nodes is detected, the connectivity is restored by replacing faulty node with block movement. The technique is called Least-Disruptive Topology Repair (LeDiR). The shortcoming of this paper is when applying the RDF function risks the agent packets to not arrive at the final nodes of the network. Therefore, sink may consider it as faulty nodes.
Titouna et al. in [13] proposed Fault Detection Scheme (FDS) which consider both battery power and sensed data to identify faulty nodes. The FDS is performed in two-levels where sensor nodes detect outlier (faulty data) using a Bayesian model, then transmit the decision to Cluster Head (CH) which verifies

only sensor nodes that have a primary decision as faulty. This second verification is based on similarity between primary decisions of cluster's members. The drawback of this approach is with a high number of faulty nodes, the process of detection is very slow and false alarm rate reaches a higher value.
The authors in [14] presented an extension of previous work [13] named Distributed Fault-Tolerant Algorithm (DFTA) in order to recover faulty nodes in WSN. The main idea is to eliminate the faulty node from the network and replace it with a switch-off node which has a higher degree of connectivity and belongs to the same cluster. Sleeping nodes are selected by CH after deployment of the network. The sleeping nodes neighbors' number must be less than the average number of cluster's members. After detecting faulty node, the CH sends a wakeup message to all sleeping nodes. These latter update their routing table and send the number of hops to reach the faulty node to CH. Finally, the CH chooses the appropriate recovery node that has fewer hops to reach the faulty one. The shortcoming of this strategy is that during the recovery phase, the CH drains too much energy since it sends three types of messages. Which overtime may decrease the network's lifetime.
H.Yuan et al. in [15] proposed a Distributed Bayesian Algorithm (DBA) which can efficiently detect faulty readings in WSN using Bayesian networks. The sensor nodes exchange their reading with neighbors and calculate the probability of fault. This latter can be incorrect if the most of the neighbors are faulty. Then, the faulty node will consider itself as good node, or the good node may consider itself as faulty one. To avoid such a case, the probability of fault must be adjusted by border nodes. If a node has two neighbors that have a low probability of fault, but have different status (i.e. one is faulty and other is good), then it will be considered as border node. The drawback of this algorithm is that, in dense networks exchanging with all neighbors will increase the energy consumption and decrease the network's lifetime. Also, the algorithm does not take into account the overhead cost.
A.V.Sutagundar et al. in [16] proposed a fault-tolerant approach based on static and mobile agents which reside in all nodes of the network. This approach is developed at three levels: node level, CH level and Sink level. At node level, the Fault Tolerant Agent (FTA) checks whether the reading is in the reference range. Then, it discards the faulty readings. The Data Transfer Agent (DTA) transmits correct readings to CH. This latter compares received data and computes the average of all corrected data, then sends it to sink through DTA. The Sink Manager Agent (SMA) broadcasts periodically "alive messages" to all nodes and duplicate sink. If the principal sink fails, the duplicate one replaces it. The shortcoming of this mechanism is that in agent based approaches the network experiences a significant amount of delay to retrieve the state of the node, because it needs to wait for an agent to visit the node. Furthermore, agent-based approaches do not perform well in large scale WSNs. Since the number of sensor nodes increases, so do the number of agents.
The authors in [17] proposed a fault-tolerant mechanism for link failure in optical networks since optical fiber cuts is the most frequent failure in metropolitan

areas. The proposed approach is based on pre-configured protection cycles (p-cycles) and Double Rings topology with Dual Attachment (DRDA). The network consists of two rings with the same number of nodes. The nodes of the inner ring and the outer one are connected through bidirectional links. The authors formed two p-cycles that cover all nodes with disjoint links. They defined three recovery strategies in order to recover single and multiple link failure. Also, they provided a formal model of the network using Continuous Time Markov Chains, and they verified dependability properties using probabilistic model checking. The proposed work gives useful information for designing dependable optical networks in metropolitan areas. However, this redundant topology may provide poor availability in a case of multiple link failure by putting an excessive load on the other links. Moreover, it may be very expensive and hard to manage or maintain.

## 3 Bayesian Classifier

In order to detect sensors that are prone to failure, their captured data should be analyzed and classified to know whether it is correct or not. This classification is based on Bayesian classifier.

A Bayesian classifier is a simple probabilistic classifier based on the Bayes theorem, it allows to classify the samples measured on observations [18][19].

Bayes theorem estimates the probability of an event, based on prior knowledge of conditions that might be related to this events [18][19]. Equation 1 represents the Bayes theorem.

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}, P(B) \neq 0 \tag{1}$$

Where A and B are events.
P(A) and P(B) are the probabilities of observing A and B independently.
$P(B|A)$ a conditional probability, is the probability of B given that A is true.
$P(A|B)$ a conditional probability, is the posterior probability of A given that B is true.

To classify the measured values in the predefined classes using the Bayesian classifier, we will have to define a Bayesian network.

A Bayesian network is an acyclic oriented graph in which nodes present random variables and arcs indicate the dependencies between these variables [7].

Figure 1 represents the structure of a Naive Bayesian Classifier. Node C represents the class $c_i$ and nodes $X = x_1, x_2, ..., x_n$ are the features of the Naive Bayesian Classifier which are independent (i.e. there are strong or naive independence assumptions between the features) [19].

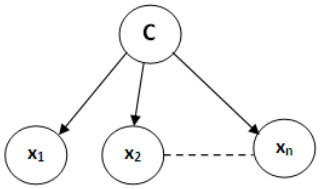


**Fig. 1.** A Bayesian Network corresponding to a Naive Bayesian Classifier

The Bayesian network allows us to calculate the posterior probability $P(C = c_i|X)$ for each possible class $c_i$ by using the Bayes theorem as stated in Equation 2.

$$P(C = c_i|X) = \frac{P(X|C = c_i)P(C = c_i)}{P(X)} \tag{2}$$

The probability $P(X|C = c_i)$ is often impractical to compute without imposing independence assumptions between features X. The Naive Bayesian Classifier assumes that each features $x_j$ is conditionally independent of every other feature [19], this yields Equation 3.

$$P(X|C = c_i) = \prod_{j=1}^{n} P(x_j|C = c_i) \tag{3}$$

Now, we obtain the Equation 4.

$$P(C = c_i|X) = \frac{\prod_{j=1}^{n} P(x_j|C = c_i)P(C = c_i)}{P(X)} \tag{4}$$

The target class $c_i$, output by the Bayesian classifer, can be inferred using the concept of Maximum A Posteriori (MAP) [19] as indicated in Equation 5.

$$c_{MAP} = argmax_{c_i \in C} P(C = c_i|X) = argmax_{c_i \in C} \frac{\prod_{j=1}^{n} P(x_j|C = c_i)P(C = c_i)}{P(X)} \tag{5}$$

Since the denominator is constant and identical for all classes, it does not affect the maximization. After deleting it, we got the Equation 6.

$$c_{MAP} = argmax_{c_i \in C} P(C = c_i|X) = argmax_{c_i \in C} \prod_{j=1}^{n} P(x_j|C = c_i)P(C = c_i) \tag{6}$$

According to Equation 6, we can classify the measured data given various observations.

# 4 Reliability of Captured Data (RCD)

Our contribution is based on Bayesian classifier where every sensor node checks its own sensed data based on his last reading and current readings of their two nearest neighbors. Then, it detects if the captured data is correct or not. Finally, it recovers faulty data by adjusting this latter to avoid transmission of erroneous values. Thus, it ensures the reliability of data before decision-making process.

## 4.1 System assumptions

Our network consists of a large number of stationary and homogeneous sensor nodes organized in clusters. Each cluster has a Cluster Head (CH) that aggregates data transmitted by its cluster's members and forwards the collected data to the Base Station (BS). Detection and recovery of inaccurate sensed data is distributed since every sensor nodes in the cluster checks its current data using the Bayesian Classifier. We also assume that we do not have malicious attacks in the network.

## 4.2 Mechanism Design

In our system model, the sensed data can belong to the interval values $I = [a, b]$. We divide this range into many classes that are mutually exclusive and exhaustive. According to Section 3, the Bayesian Network of our mechanism is represented in Figure 2.

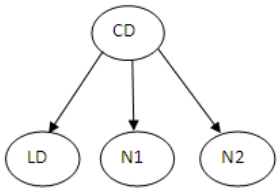


**Fig. 2.** RCD Bayesian Network

CD : the current sensed data that belongs to a class $c_i$
LD : the last sensed data that belongs to a class $c_i$
N1 : the current data of the first neighbor that belongs to a class $c_i$
N2 : the current data of the second neighbor that belongs to a class $c_i$
These variables ensure spatial-temporal correlation in the network.

The RCD's algorihtm is composed of three phases:

**Learning phase** This phase is executed after the deployment of the network. We assume that it is free from faults.
The learning phase aims to calculate the parameters of the classifier which are: the probability of each class $P(CD \in c_i)$ and conditional probabilities $P(LD|CD)$, $P(N1|CD)$ and $P(N2|CD)$.
Algorithm 1 explains the steps of the learning phase executed by sensor nodes.

**Algorithm 1** Learning phase

**Input** : Predefined classes
**Output** : The parameters of the classifier
**BEGIN**
Defines the two nearest neighbors.
Gets current data of the two neighbors.
Calculates the classes' probabilities and the conditionnal probabilities.
**END**

Every sensor node sense 'P' samples and gets the samples (readings) of its two neighbors over a period of time. Then, it calculates the classes' probabilities and the conditional probabilities. The complexity of the algorithm 1 is quadratic since the running time of the first instruction is proportional to the number of the cluster's members while the second one has constant running time. The last instruction has two running times; one is proportional to the number of classes for calculating the classes' probabilities, and the other is proportional to the square of the number of classes for calculating the conditionnal probabilities.

**Inference phase** In this phase, sensor node detects faulty data by calculating the posterior probability of $c_i$ using Bayes theorem. Then, it deduces the most probable class that its sensed value should belong to. The most probable class is calculated using the MAP (See Equation 5).
Each sensor node calculates the posterior probability of $c_i$ according to Equation 6. We replace $x_j$ by three observations; the last sensed data (LD), the current data of the first neighbor (N1), and the current data of the first neighbor (N2). Hence, we obtain the Equation 7.

$$c_{MAP} = argmax_{c_i \in I} P(LD|c_i)P(N1|c_i)P(N2|c_i)P(c_i) \tag{7}$$

After calculating the posterior probability of each predefined or learned class $c_i$, the sensor node compares the class that has the highest probability with the class of the captured value. If they are identical, the sensor considers that the captured value is correct. Otherwise, it identifies it as a faulty one.
The steps of the inference phase are detailed in Algorithm 2. The time complexity of the inference algorithm is linear since the first instruction has constant running time. The second one has a running time proportional to the number of classes, and the running time of if-else statements is constant in the two posibilities (i.e. if sequence and else sequence).

**Algorithm 2** Inference phase

**Input** : CD, LD, N1 and N2
**Output** : The most probable class
**BEGIN**
Exchange with the two neighbors.
Calculate the posterior probability of each class using Equation 7.
**if** (CD $\in$ the most probable class) **then**
  Captured data is correct.
**else**
  Captured data is incorrect.
  Recovery phase.
**end if**
**END**

**Recovery phase** In this phase, sensor nodes adjust faulty data by approximate the data value to the average of the most probable class (the result of the classifer)(See Algorithm 3). The recovery phase aims to connect the regions isolated by faulty nodes. The algorithm of the recovery phase has constant complexity.

**Algorithm 3** Recovery phase

**Input** : Posterior probability of all classes
**Output** : Adjusted data
**BEGIN**
CD = the average of the most probable class (the class with the highest posterior probability)
**END**

We can conclude that the time complexity of our RCD's algorithm is quadratic.

## 5 Simulation Results

In order to evaluate our contribution, we have conducted several series of simulation using Network Simulator 2 (NS2). The simulation results are compared to FDS detailed in Section 2. To illustrate the performance of our proposal, we injected some faulty nodes in the network.

### 5.1 Simulation Parameters

The simulation parameters are summarized in Table 1.

### 5.2 Performance Metrics

For comparaison purposes, we have considered the following performance metrics.

**Table 1.** Simulation parameters

| Parameter | Value |
|---|---|
| Network size | 100m*100m |
| Number of sensor nodes | 100 |
| Initial energy | 2J |
| Number of clusters | 4 |
| Round duration | 20s |
| Slot duration | 0,02s |

**Fault Detection Accuracy (FDA)** is the ratio of the number of faulty sensor nodes detected to the total number of faulty sensor nodes. Ideally, this ratio should be equal to 1.

**False alarm rate (FAR)** is defined as the ratio of the number of non-faulty nodes diagnosed as faulty to the total number of non-faulty nodes. The FAR should be very small to achieve a reliable detection.

### 5.3 Results and Discussions

As we mentionned above, to evaluate the performance of our proposal, we injected 15 faulty nodes in the network. Figure 3 shows the fault detection accuracy in both of FDS and RCD.

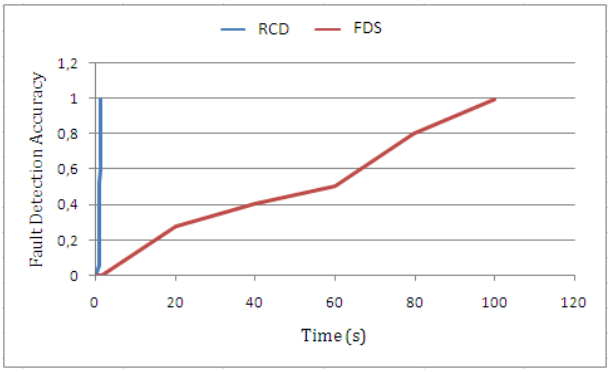


**Fig. 3.** Fault Detection Accuracy in FDS and RCD with 15 faulty nodes

According to Figure 3, we notice that the curve of our protocol evolves very quickly to reach 1 during 1.5s. This is due to the distribution of the detection which is performed by nodes before data transmission. The data transmission is performed in rounds. Sensor nodes transmit data during its time slot of the

round. Unlike the RCD protocol, the FDS reaches the value 1 after 100s because the detection of faulty sensed data is performed in two-levels: sensor node's level and CH's level. We conclude that both of RCD and FDS reach the ratio 1 which is the ideal case. However the fault detection accuracy using the RCD algorithm is faster than when using the FDS one.

Figure 4 represents the false alarm rate for both FDS and RCD.

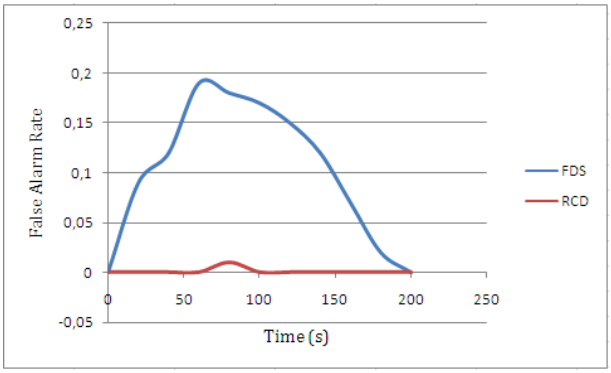


**Fig. 4.** False Alarm Rate in FDS and RCD with 15 faulty nodes

According to Figure 4, we notice that the maximum rate of false alarm in the FDS is 0,18 at 60s, then, it decreased to 0 at 200s. While in the RCD protocol, it reaches 1 at 80s. This proves that our proposal can achieve an accurate and reliable detection.

## 6 Conclusion

In IoT, the application reliability is strongly correlated with the accuracy of data captured by sensors. When a sensor captures wrong data, this may lead to wrong decision-making, which is highly undesirable in critical applications. To overcome such a problem, we proposed in this paper a new fault-tolerant mechanism called RCD (Reliability of Captured Data). The RCD enables sensors to detect faulty data using the Bayesian Network. This mechanism is also able to recover these innacurate and unreliable data before forwarding them to the BS. To show the efficiency of our work, we compared the RCD strategy to the FDS one using NS2. The results show a remarkable improvement in terms of false alarm rate and fault detection accuracy.
As a future perspective of our current work, we aim to integrate safety into data communication of IoT applications.

## References


1. Li, S., Xu, L.D., and Zhao, S. (2015). The internet of things: a survey. Information Systems Frontiers, 17(2), 243-259.
2. Silva, I., Leandro, R., Macedo, D., and Guedes, L.A. (2013). A dependability evaluation tool for the Internet of Things. Computers and Electrical Engineering, 39(7), 2005-2018.
3. Lee, I., and Lee, K. (2015). The Internet of Things (IoT): applications, investments, and challenges for enterprises. Business Horizons, 58(4), 431-440.
4. Cauchi, N., Hoque, K.A., Abate, A., Stoelinga, M. (2018). Efficient Probabilistic Model Checking of Smart Building Maintenance using Fault Maintenance Trees. arXiv:1801.04263.
5. Kakamanshadi, G., Gupta, S., and Singh, S. (2015). A Survey on Fault Tolerance Techniques in Wireless Sensor Networks. In International Conference on Green Computing and Internet of Things (pp. 168-173).
6. Samson Arun Raj, A., Ramalakshmi, K., and Priyadharsini, C. (2014). A Survey on Classification of Fault Tolerance Techniques Available in Wireless Sensor Network. International Journal of Engineering Research and Technology (IJERT), 3(1), 688-691.
7. Ben-Gal, I. (2008). Bayesian Networks. Encyclopedia of Statistics in Quality and Reliability.
8. Zhang, Y., Meratnia, N., and Havinga, P. (2010). Outlier Detection Techniques for Wireless Sensor Networks: A Survey. IEEE communications surveys and tutorials, 12(2), 159-170.
9. Ripley, B. D. (1996). Pattern recognition and neural networks. Cambridge, UK: Cambridge University Press.
10. Ghahramani, Z. (2001). An introduction to Hidden Markov Models and Bayesian Networks. International Journal of Pattern Recognition and Artificial Intelligence, 15(1), 9-42.
11. Ayadi, A., Ghorbel, O., Obeida, A.M., and Abida, M. (2017). Outlier detection approaches for wireless sensor networks: A survey. Computer Networks, 129(1), 319-333.
12. Yuvaraja, M., and Sabrigiriraj, M. (2017). Fault detection and recovery scheme for routing and lifetime enhancement in WSN. Wireless Networks, 23(1), 267277.
13. Titouna, C., Aliouat, M., and Gueroui, M. (2016). FDS: Fault Detection Scheme for Wireless Sensor Networks. Wireless Personal Communication, 86(2), 549-562.
14. Titouna, C., Gueroui, M., Aliouat, M., Ari, A.A.A., and Amine, A. (2017). Distributed Fault-Tolerant Algorithm for Wireless Sensor Network. International Journal of Communication Networks and Information Security (IJCNIS), 9(2), 241-246.
15. Yuan, H., Zhao, X., and Yu, L. (2015). A Distributed Bayesian Algorithm for Data Fault Detection in Wireless Sensor Networks. In International Conference on Information Networking (pp. 63-68).
16. Sutagundar, A.V., Bennur, Vidya S., and Bhanu, K.N.Anusha A.M. (2016). Agent Based Fault Tolerance in Wireless Sensor Networks. In International Conference on Inventive Computation Technologies (pp. 1-6).
17. Siddique, U., Hoque, K.A., Johnson, T.T. (2017). Formal specification and dependability analysis of optical communication networks. In 2017 Design, Automation and Test in Europe Conference and Exhibition (DATE) (pp. 1564-1569).
18. Zhang, H. (2004). The optimality of naive Bayes. In Seventeenth Florida articial intelligence research society conference.

19. Murphy, Kevin P. (2012). Machine learning : a probabilistic perspective. Cambridge, MA: MIT Press.