Soft Fault Detection Using an Improved Mechanism in Wireless Sensor Networks

Mojtaba Montazeri[1] and Rasoul Kiani[1]
[1] Department of Computer Engineering, Islamic Azad University, Iran
[mojtabamontazri2017@yahoo.com; rasoul.kiani87@yahoo.com]
*Corresponding author: Mojtaba Montazeri

Received August 9, 2017; revised November 26, 2017; revised April 25, 2018; accepted May 3, 2018; published October 31, 2018

Abstract

Wireless sensor networks are composed of a large number of inexpensive and tiny sensors used in different areas including military, industry, agriculture, space, and environment. Fault tolerance, which is considered a challenging task in these networks, is defined as the ability of the system to offer an appropriate level of functionality in the event of failures. The present study proposed an intelligent throughput descent and distributed energy-efficient mechanism in order to improve fault tolerance of the system against soft and permanent faults. This mechanism includes determining the intelligent neighborhood radius threshold, the intelligent neighborhood nodes number threshold, customizing the base paper algorithm for distributed systems, redefining the base paper scenarios for failure detection procedure to predict network behavior when running into soft and permanent faults, and some cases have been described for handling failure exception procedures. The experimental results from simulation indicate that the proposed mechanism was able to improve network throughput, fault detection accuracy, reliability, and network lifetime with respect to the base paper.

Keywords: Soft fault, permanent fault, fault-tolerant, wireless sensor networks

1. Introduction

Wireless sensor networks (WSNs) are essentially prone to faults, in which case their reliability is highly affected [1], [2]. Faults are mainly defined as a change or improper, unexpected behavior of a sensor and may not necessarily lead to physical failure or malfunction [3], [4]. Fault detection in WSNs is based on centralized and distributed methods. In the former case, the faults are reported to a single point, i.e. the base station or the nodes close to which, causing the base station to experience a large overhead, energy consumption, and shorter lifetime [5], [6]. In the distributed methods, the decisions are made locally and, therefore, the overhead is decreased since fewer numbers of messages is transmitted to the central node [7]. According to Oh et al. [5], fault types are classified as 1) node fault, 2) network fault, and 3) sink fault. Moreover, Karimi et al. [8] classifies faults in four categories, namely hardware layer, software layer, network communication layer, and application layer. Classification of fault tolerance techniques is based on different criteria, one of which is when the fault tolerance procedure is launched (before or after occurrence of the fault) [9], [10]. Based on this classification, techniques are divided into two categories, namely preventive and curative. Preventive techniques attempt to prevent occurrence of faults through improving the use of current resources or implementing different alternatives which provide similar services. Preventive techniques can be applied in the node level and network level. In the former case, the aim is to extend node lifetime, while the latter deals with the network lifetime. In case the network fails to deliver its assigned task, curative techniques intervene to recover from the fault in order to resume transmission or sensing. To this end, the faulty component is replaced through activating sleep nodes or displacing the nodes [11], [12]. According to Rajeswari et al. [19] a cluster-based fault tolerance technique using genetic algorithm is proposed. The network which presented in this paper, is clustered according to energy-efficient distance-based clustering algorithm. For each cluster head, a set of backup nodes are selected using genetic algorithm based on the sponsored coverage and residual energy parameters. This helps in detecting the faults occurring in cluster members and cluster heads. By simulation results, it has been shown that the proposed technique minimises the energy and packet loss with reduced delay. In the another study presented by Vieira et al. [20] presents a new approach to centralized fault management system for 6LoWPAN WSN. The system is based on two fault detection levels. A first level is performed locally, by all sensors within the network, using statistical methods. The second level is performed by the base station, through an ensemble of Multilayer Perceptron type Artificial Neural Networks (ANN) classifiers. One of them is continuously trained with streaming data, while the other one is used to take actual decisions about fault detection. In another study, the SVM classification method is used for detecting failure based on statistical learning theory. This method has an important adaptation capacity for the nonlinear classification cases as fault detection, by using the kernel functions [21]. According to Abdul-Salaam et al. [22] an energy efficient packet reporting (EPR) scheme proposed to report event packets in an energy-efficient manner. In fact, the application of this method can be used in supporting mobile node navigation in position free hybrid wireless sensor networks (HWSN). This approach, aimed EPR to increase its lifetime due to the fact that sensor energy optimized just for sensing and reporting event packets to mobile nodes. Based on efficient fault detection and routing (EFDR) scheme is proposed by Bonerjee et al. [23] three linear cellular automata (CA) are used to manage transmitter circuit/battery condition/microcontroller fault, receiver circuit fault and sensor circuit fault representation.

Blo

₩On the other hand, L-system rules based data routing scheme is proposed to determine optimal routing path between cluster head and base station. In another case, Wei et al. [24] present an improved Virtual Force Algorithm (VFA) for node deployment in the complex environment. Deploying sensors into a target region is a key issue to be solved in building a wireless sensor network. Moreover, it uses the Google satellite maps to extract practical information such as the land cover and elevation. Then, based on these practical information, it evaluates the proposed algorithm. In accordance with the experimental results, the algorithm could provide 15% higher coverage compared to the traditional VFA.

The present study aimed to present a new mechanism in order to improve the proposed algorithm presented by Sharma et al. [7] in terms of detection of soft and permanent faults.

The remaining sections are organized as follows: Literature review is presented in Section 2. The proposed mechanism is discussed in Section 3. The results of simulations are reported in Section 4 and, ultimately, the paper is concluded in Section 5.

2. Related Work

Based on the k-means clustering method, the K-CFD algorithm was proposed by Yang et al. [6] for fault detection. In this algorithm, the ant colony optimization algorithm was employed to improve the quality of the results obtained from the clustering mechanism. Based on the clustering results, each “good” node propagates the measurements of its neighboring nodes across the network in order to store energy. All the updated data are the locally accumulated based on the ACO algorithm and are again clustered into the located good and located fault clusters. Nitesh et al. [13] proposed an energy-efficient fault-tolerant algorithm known as EEFCA. The proposed distributed algorithm is based on multiple parameters such as remaining energy and distance. The algorithm also guarantees repairing local orphan nodes caused by faults. The main advantage of this algorithm over others lies in its ability to extend the network lifetime, specifically the lifetime of the cluster heads. The NHCRF algorithm proposed by Tang et al. [14] comprises two stages, namely the modeling and monitoring stages. An objective of the modeling stage is to train the NHCRF algorithm using the history data of the nodes. This trained model is then used in the monitoring stage to estimate the probability of unlabeled nodes for being healthy or faulty. Two objectives were considered by Vigneshwari et al. [15] for enhancement of fault-tolerance in WSNs which are based on the hierarchical mechanism. To this end, the number of cluster heads and the communications between the nodes and the cluster head are suggested to be minimized and maximized, respectively. This method attempts to use the general multi-channel technique to increase fault-tolerance. The evaluation function uses the genetic algorithm to achieve these two objectives. Azharuddin et al. [16] used a PSO-based routing algorithm to maximize network lifetime through minimizing the consumed energy. This algorithm mainly focuses on the nodes close to the base station. Note that fault-tolerance was addressed by taking into account the routing direction and detection of permanent faults in the cluster heads. According to Venkataraman et al. [17], faulty nodes result in voids in the network topology, which in turn can cause connection failure and loss of critical data. Therefore, the cluster-based energy-efficient technique was presented to detect faults and recover the structure of the clusters. In summary, the cluster heads, which have more resources compared to other nodes, are responsible for faults in this algorithm when clustering techniques are used for detection of distributed faults. Moreover, special attention is given to detection and repair of faults in the

cluster heads, since their failure leads to limited access to the nodes under their supervision. In order to detect permanent faults, Sharma et al. [7] presented a mechanism based on four scenarios in an attempt to improve the network throughput by focusing on the energy consumption required for fault detection. In this mechanism, each node can assume three states, namely "good", "faulty", and "suspicious". Each node initially checks its status and, in case of a suspicious state, sends a message to the nodes within its predefined neighborhood radius. The neighboring nodes then report their status back to the requesting node, allowing it to reconsider its status. The defined scenarios are as follows:

1. When only one node is faulty within a circular region *D*.
2. When more than one node is faulty within a circular region *D* but faulty nodes are less in number as compare to non-faulty nodes.
3. When many nodes observed a sudden change in their readings within a circular region *D* because of actual occurrence of event.
4. When maximum nodes are faulty in the vicinity of a non-faulty node.

These different scenarios are demonstrated in **Figs. 1-4**.

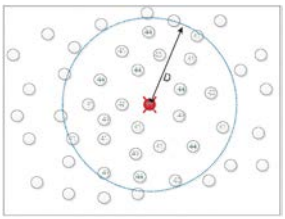

Fig. 1. Single node is faulty [7]

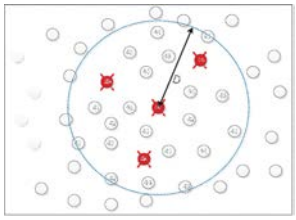

Fig. 2. Multiple nodes are faulty [7]

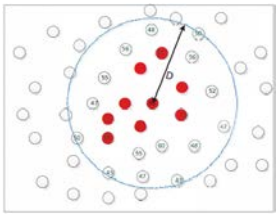

Fig. 3. Actual event is occurred and multiple nodes find themselves suspicious [7]

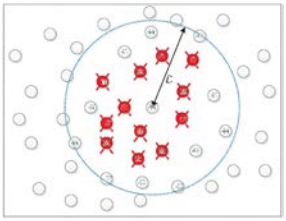

Fig. 4. Maximum nodes are faulty in the neighbor of a non-faulty node [7]

3. Proposed Mechanism

The proposed mechanism presented by Sharma et al. [7] was studied in the previous section and the following challenges were identified:

1. The neighborhood radius is not intelligently determined, meaning that network designer calculates this radius experimentally or through trial and error based on the configuration of nodes in the network. In general, this procedure is not of sufficient accuracy.
2. Detection accuracy of soft faults is decreased, since outlier nodes, i.e. nodes which have insufficient number of nodes in their neighborhood radius, are not taken into consideration.
3. The distribution structure is not investigated in the communication between nodes, hence interfering with the process of detection of soft and permanent faults.

3.1 Definitions

- Network lifetime: The number of alive nodes as a function of time is used to measure the network lifetime.
- Soft fault: Soft faulted units can communicate with its neighbors but with unexpected behaviors and always give unwanted response [10]. In this study, this refers to reading faults both in a partial or global scope.
- Partial fault: Faults in a limited number of sensor nodes. In this study, local faults found within the neighborhood radius of a suspicious node.
- Global fault: Faults in a large number of sensor nodes. In this study, local faults found within the neighborhood radius of a suspicious node.
- Permanent fault: Once it appears in network it remains until it removed and repaired by some external administrator. Permanent faults are simpler to deal [10].
- Alive node: A node with an energy level above zero.
- Dead node: A node which is not alive or has undergone permanent fault.
- Good node: An alive node capable of producing and transmitting correct (reliably) data.
- Suspicious node: An alive node with a pending status until its state is identified as either "good" or "faulty".
- Faulty node: An alive node producing incorrect (unreliably) data.
- MsgB: A message generated by a suspicious node and sent to the CH requesting the neighboring nodes to calculate the difference in readings [7].
- MsgC: A message generated by the CH of the cluster containing the suspicious node [7].
- MsgR: Those nodes receiving the message MsgB reply with an MsgR message [7].
- MsgVM: A message broadcasted across the network by the sink from received message from a cluster head, which containing the reading of a good node.

3.2 Assumptions

The sensors in the Region Of Interest (ROI) the sensors in the environment are denoted by S_i, where $i = (1.2 \ldots . n)$ is the ID of each node and n is the total number of nodes in the network. The sensors in the environment are denoted by S_i, where $i = (1.2 \ldots . n)$ is the id of each node and n is the total number of nodes in the network. The membership of a sensor S_i in a cluster C_k is represented by $C_k S_i$. Moreover, $k = (1.2 \ldots . m)$, where m is the total number of clusters. Each cluster C_k includes a cluster head CH_k, where the CH is responsible for gathering data from the cluster members and aggregating, compressing, and transmitting them to the sink. The only communication channel to the outside of a cluster for a sensor S_i is the CH of the respective cluster, to which the sensor S_i belongs $(C_k S_i)$. The CHs can communicate through the sink. All nodes are equipped with GPS and are aware of their location. Moreover, due to the distributed nature of the network, the CH is in possession of the spatial information of its members for utilization in case necessary. Each alive node S_i is normally either a CH or a cluster member. Each S_i node is either alive or dead due to energy depletion. Each alive node S_i can assume one of the "good", "suspicious", and "faulty" states. In the first iteration of the algorithm, the sink is responsible for calculation of two thresholds, i.e. the neighborhood radius (R_{thr}) and the number of neighboring nodes (N_{thr}), according to the presented algorithm in Section 3.4. The follow procedure (defined by Sharma et al. [7] and handles all the scenarios that illustrated in **Figs. 1-4**) redefined in steps 3 and 4, also in this paper steps 5-9 is called Failure Detection Procedure (FDP).

1. Initially, all nodes set their status variable with "$GOOD$".
2. Every node calculates the variation σ_k^2 of past k readings and whenever the difference of readings of sensor s_i; at time t and $t-1$ is greater than the variation, then it set its status with "$SUSPECIOUS$" and select a random time value in Timer variable form W_t.
3. When node s_i expires the timer then it send message $MsgB(t, COD_i, X_i^t)$ to associated cluster head.
4. All sensor nodes s_j receive the MsgB from a cluster head (As shown in **Fig. 5**) and d_{ij} (Euclidian distance between s_i and s_j) is less than R_{thr}, finds the difference between their reading with s_i as ΔX_{ij}^t and reply by a message $MsgR(t, COD_i, \Delta X_{ij}^t, Statuss_j)$ to their cluster head. Associated cluster head collect messages MsgR from neighbors and reply to s_i.
5. Every node $s_i$maintains an array $RStatus[n_I]$ of size $n_I$where $n_I$is the number of sensors within distance D from node s_i. Initially the array is initialized by sets $RStatus[n_I]$ with default value -1.
6. After receiving the reply of s_j the sensor node s_i update array $RStatus$ with 0, 1, 2 or 3.
7. Set the status of s_i with "$GOOD$" or "$FAULTY$" and in some cases wait for reply from some suspicious node and update $RStatus$ array and restart current step.
8. After $2T_{wait}$ if it receive the reply from n_2 number of sensor node and if maximum of replies have the difference of readings less than $X_{th}$with s_i then s_i sets its status with "$GOOD$" otherwise sets its status with "$FAULTY$".
9. In the status array of s_i if some of s_j finally have their default value -1, then they are considered as permanently failed.

In step 2 above, the condition for a node s_i; to be suspicious is as given in Eq. 1:

$$\left|X_i^t - X_i^{t-1}\right| > min\{\sigma_k^2 + X_{th} + Xd_{rft}\} \tag{1}$$

where X_i^t is the reading of a sensor node s_i at time t, Xd_{rft} is the difference or the possible drift in the reading of two or more sensor nodes for measuring same value of information, σ_k^2 is the variance of past k readings of a sensor node and X_{th} is a threshold value.

In step 4, sensor nodes s_j, which are in the range of D find the difference of their reading with s_i as in Eq. 2:

$$\Delta X_{ij}^t = \min\left\{\left(\frac{D - d_{ij}}{D}\right) \times \left|X_i^t - X_j^t\right| \pm Xd_{rft}\right\} \tag{2}$$

where ΔX_{ij}^t is the difference between the reading of sensor s_i , s_j at time t and d_{ij} is the Euclidian distance between sensor $s_i(x_i, y_i)$ and $s_j(x_j, y_j)$.

In step 6, $RStatus$ assigns as in Eq. 3:

$$RStatus[j] = \begin{cases} 0 \; if \; Statuss_j = \text{"GOOD"} \; \&\& (\Delta X_{ij}^t <= X_{th}) \\ 1 \quad if \; Statuss_j = \text{"GOOD"} \; \&\& (\Delta X_{ij}^t > X_{th}) \\ 2 \qquad\qquad if \; Statuss_j = \text{"}FULTY\text{"} \\ 3 \qquad\qquad\qquad Otherwise \end{cases} \tag{3}$$

In step 7, the status of suspicious node determines through the following pseudo-code:

```
For  l = 0 → n_l − 1
    Cnt0 = Cnt1 = Cnt2 = Cnt3 = Cnt4 = 0 ;
    If  RStatus[l] == 0
        Cnt0 + + ;
    Elseif  RStatus[l] == 1
        Cnt1 + + ;
    Elseif  RStatus[l] == 2
        Cnt2 + + ;
    Elseif  RStatus[l] == 3
        Cnt3 + + ;
    Else
        Cnt4 + + ;
    End
    If  Cnt0 ≥ (n_l − Cnt4)/2
        Status_i = "GOOD" ;
        break ;
    Elseif  Cnt1 > (n_l − Cnt4)/2
        Status_i = "FAULTY" ;
        break ;
    Elseif  Cnt3 > (n_l − Cnt4)/2
        wait for reply from some suspicious node and update RStatus ;
        Goto line 1 ;
    Elseif  Cnt2 > (n_l − Cnt3)/2
        Bradcast MsgVM(t, COD_i, X_ij^t, Statuss_i) ;
        wait for 2T_wait ;
    End
End
```

In a distributed network, a Cluster Head (CH) is responsible for collecting data from normal good nodes, aggregating and extracting them, and finally transmitting to the sink. Therefor CH is one of the main actors in a distributed network, and the failure of CH can lead to the network failure. Hence in the proposed algorithm an alternate CH for each CH has been assigned in each round of the network operation as shown in **Fig. 5**.

3.3 Network Configuration

The PLEACH algorithm proposed by Jianfeng [18] was used to configure the network. By taking into account the remaining energy and distance between nodes, PLEACH employs the PSO algorithm to optimize the clustering process. The improved PSO algorithm can be advantageous in calculating the optimal number of CHs, specifically in the first phase. In the first step, the CH is generated by the sink and the number of optimal CHs in a rectangular of AxB dimensions is calculated according to the information provided by problem. This process guarantees uniform distribution of the cluster in the initial phase. In the second step, the selected CH sends an announcement message to the other nodes and the non-CH nodes select and join an appropriate CH depending on their distance. After receiving the messages from the nodes, the CH then generates a TDMA schedule message and informs the nodes. In the third step, the nodes start transmitting information based on the aforementioned schedule. In the fourth step, in case the remaining energy of the CH becomes lower than the threshold, a

message is sent to the other cluster nodes by the CH. Upon reception, the nodes send their remaining energy level and location to other nodes. Therefore, each node stores the geometric information of other nodes. The objective is to present a fitness function in order to select a CH. The fitness values not only should reflect the energy level of each node, but also their distance from the other nodes. The fitness function is expressed according to Eq. 4.

$$f(k) = \eta e_k - \lambda d. where\ .\ e_k \geq \bar{e} \tag{4}$$

where η is the energy impact factor, $\bar{e}$ is the average energy of nodes in the cluster, k is the number of current nodes, λ is the distance impact factor, and d is the mean distance of each node based on Euclidean distance and speed characteristic in the Particle Swarm Optimization (PSO) algorithm. Moreover, $\eta+\lambda=1$ and $\eta, \lambda \in [0,1]$. Ultimately, the node with the highest fitness value announces itself as the CH.

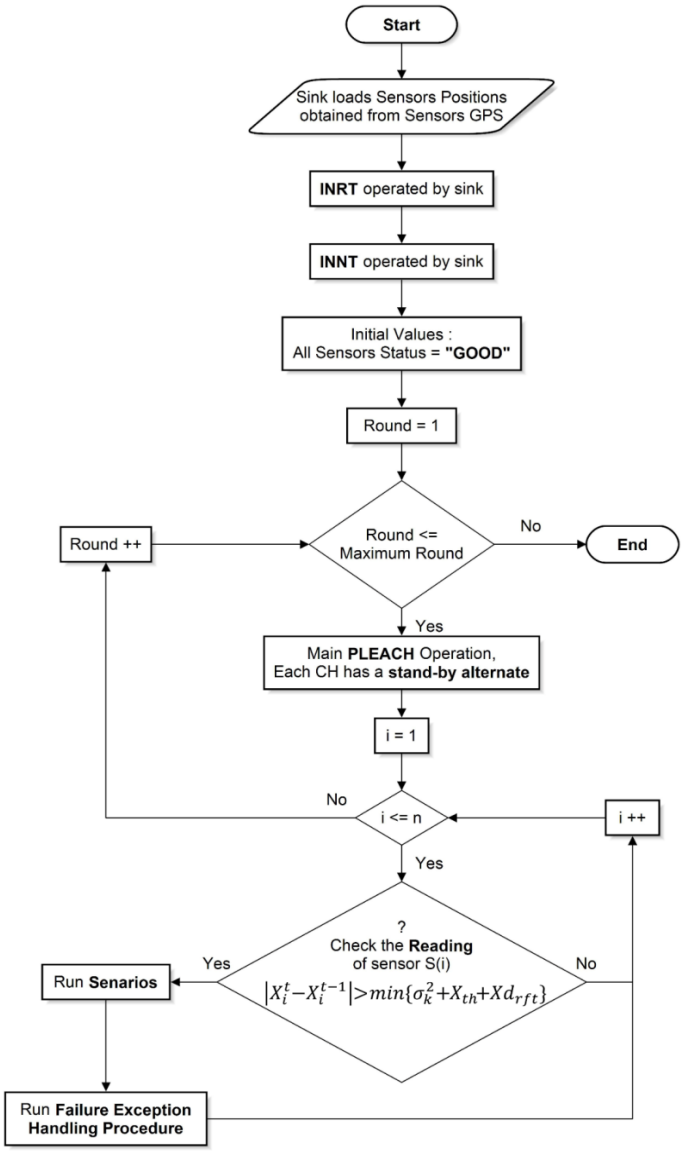

Fig. 5. Main failure procedure

3.4 Intelligent Neighborhood Radius Threshold (INRT)

In order to determine the neighborhood radius threshold for each node according to the flowchart shown in **Fig. 6**, the coefficient of variation for each node is calculated using the matrix of distances between nodes and forming standard deviation and mean matrixes. Finally, the coefficient of variation and standard variation are combined to determine the neighborhood radius threshold and then the number of neighboring nodes. This procedure is described in the followings.

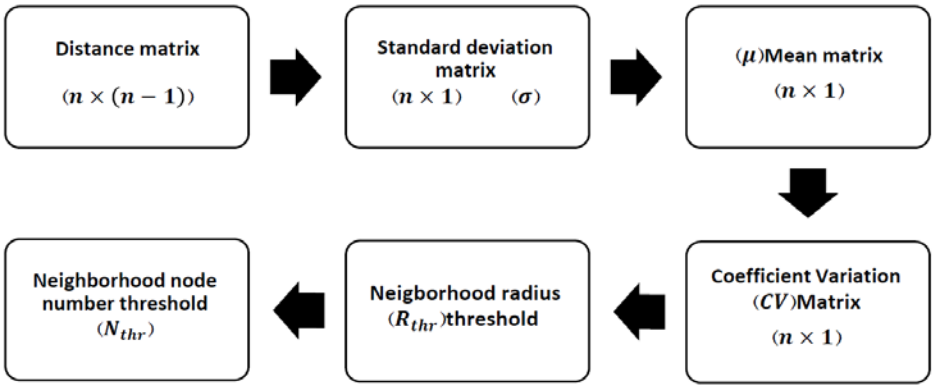

Fig. 6. Intelligent neighborhood radius threshold (INRT) procedure

3.4.1 Distance Matrix

The matrix of Euclidean distance is simply formed from Eq. 5 using the distances between nodes (S_i).

$$d_{ij} = \sqrt{\Delta Sx_{ij}{}^2 + \Delta Sy_{ij}{}^2} \tag{5}$$

where ΔSx_{ij} and ΔSy_{ij} are obtained from the following equations:

$$\Delta Sx_{ij} = \left|Sx_j - Sx_i\right| \tag{6}$$

$$\Delta Sy_{ij} = \left|Sy_j - Sy_i\right| \tag{7}$$

which is derived by combining Eqs. 5-7:

$$d_{ij} = \sqrt{(Sx_j - Sx_i)^2 + (Sy_j - Sy_i)^2} \tag{8}$$

where d_{ij} is the Euclidian distance between nodes S_i and S_j. The distance matrix $\mathrm{D}_{(\mathrm{n}\times(\mathrm{n}-1))}$ is calculated through the following pseudo-code:

```
1    For  i = 1 → n
2         j = 0 ;
3         For  a = 1 → n
4              If  i ≠ a
```

```
5                       j++;
6                       D_(i,a) = d_ij ;
7                   End
8               End
9           End
```

3.4.2 Standard Deviation Matrix

Standard deviation σ is the square root of variance σ^2 expressed using Eq. 9.

$$\sigma_i = \sqrt{\sigma_i^2} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n-1}(D_{ij} - \bar{D}_\iota)^2} \tag{9}$$

where D_{ij} is the distance between nodes i and j, and $\bar{D}_\iota$ denotes the mean distance of nodes from node i. Finally, σ_i is the standard deviation of node i.

3.4.3 Mean Matrix

The mean value μ for each node is calculated from Eq. 10.

$$\mu_i = \bar{D}_\iota = \frac{\sum_{j=1}^{n-1} D_{ij}}{n-1} \tag{10}$$

3.4.4 Coefficient Variation Matrix

Coefficient of variation for each node is obtained from the standard variation of the respective node σ_i divided by its mean distances μ_i according to Eq. 11.

$$CV_i = \frac{\sigma_i}{\mu_i} \tag{11}$$

In fact, coefficient of variation represents the density of data around the mean value. In simpler terms, the smaller the difference between node distances, the lower the coefficient of variation. In other words, the more uniform the distribution of nodes in the network, the smaller the coefficient of variation, so that in case the node distances from a given node are identical, then $\sigma_i = 0$ and consequently $CV_i = 0$.

3.4.5 Neighborhood Radius Threshold

In this section, the coefficient of variation and the standard deviation should be combined such that the most appropriate neighborhood radius threshold R_{thr} is achieved. To this end, it is suggested that the average of coefficient of variation is multiplied by the standard deviation according to Eq. 12.

$$R_{thr} = \text{mean}(\sigma_i) \times \text{mean}(CV) \tag{12}$$

3.4.6 Intelligent Neighborhood Node Number Threshold (INNT)

The number of neighboring nodes for node i within the neighborhood radius R is represented by Nbr_r^R. In order to determine the neighborhood node number threshold N_{thr}, the minimum

number of neighboring nodes in the neighborhood radius is initially calculated, then the minimum mean number of neighbors is obtained as the neighborhood node number threshold as expressed in Eq. 13.

$$N_{thr} = \frac{1}{n}\sum_{i=1}^{n} Nbr_i^{R_{thr}} \tag{13}$$

where $Nbr_i^{R_{thr}}$ is the number of neighboring nodes within the neighborhood radius R_{thr} for node i.

The INNT is calculated through the following pseudo-code:

```
%  Table " S " is created from received sensors positions :
10    For  i = 1 → n
11         S(i.1) = Sx_i ;
12         S(i.2) = Sy_i ;
13    End
14    Sink creates a Table of Distances (D_{n×(n−1)}) as described in Distance Matrix ;
15    Sink calculates INRT (Intelligent Neighborhood Radius Threshold) ;
16    Nbr_i = 0 ;%  i = {1.2.3. ….n}  Nbr_i = Number of Neighbors of Sensor S_i
17    For  i = 1 → n
18         For  j = 1 → n-1
19              If  D_ij ≤ R_thr
20                   Nbr_i + + ;
21              End
22         End
23    End
24    N_thr = mean(Nbr) ;
25    RNbr_i = R_thr ;%  i = {1.2.3. ….n}RNbr_i= Neighborhood Radius of Sensor S_i
26    For  i = 1 → n
27         While  Nbr_i < N_thr
28              RNbr_i + + ;
29              Do Step 9 to 13 ;
30         End
31    End
32    Sink extracts RNbr_{n×1} and broadcasts it over the network ;
33    All sensors receive an extracted RNbr_{n×1} and store it in the memory ;
```

3.5 Detection of Suspicious Nodes and Soft Faults

In this section, the proposed algorithm for exploitation was used to detect suspicious nodes and soft faults (global and partial). To this end, the proposed method proposed by Sharma et al. [7] and the following figure were used to detect suspicious nodes and soft faults, respectively.

In order to find suspicious reading, every node calculates the variation (σ_k^2) of past k readings and whenever the difference of readings of sensors; at time t and t-1 is greater than the variation, then node consider its reading suspicious. The condition of for a node s; to be suspicious is as given in Eq. 14.

$$|X_i^t - X_i^{t-1}| > min\{\sigma_k^2 + X_{th} \pm Xd_{rft}\} \tag{14}$$

where,

- X_i^t: Reading of a sensor node $s_i$at time t.
- Xd_{rft}: The difference or the possible drift in the reading of two or more sensor nodes for measuring same value of information.
- σ_k^2: The variance of past k readings of a sensor node.
- X_{th}: A threshold value.

The variance of past k readings can be easily calculated as in Eq. 15.

$$\sigma_k^2 = \frac{\sum_{j=0}^{k}(X_i^j - \mu_k)}{k} \tag{15}$$

where, the mean of past k value is:

$$\mu_k = \frac{\sum_{j=0}^{k-1}(X_i^{t-j})}{k}$$

3.6 Scenarios

In this section the four scenarios of proposed by Sharma et al. [7](BP) presented in **Figs. 1-4**, were redefined and illustrated in **Figs. 7-11**, in order to predict system behavior in the event of soft and permanent faults (global and partial).

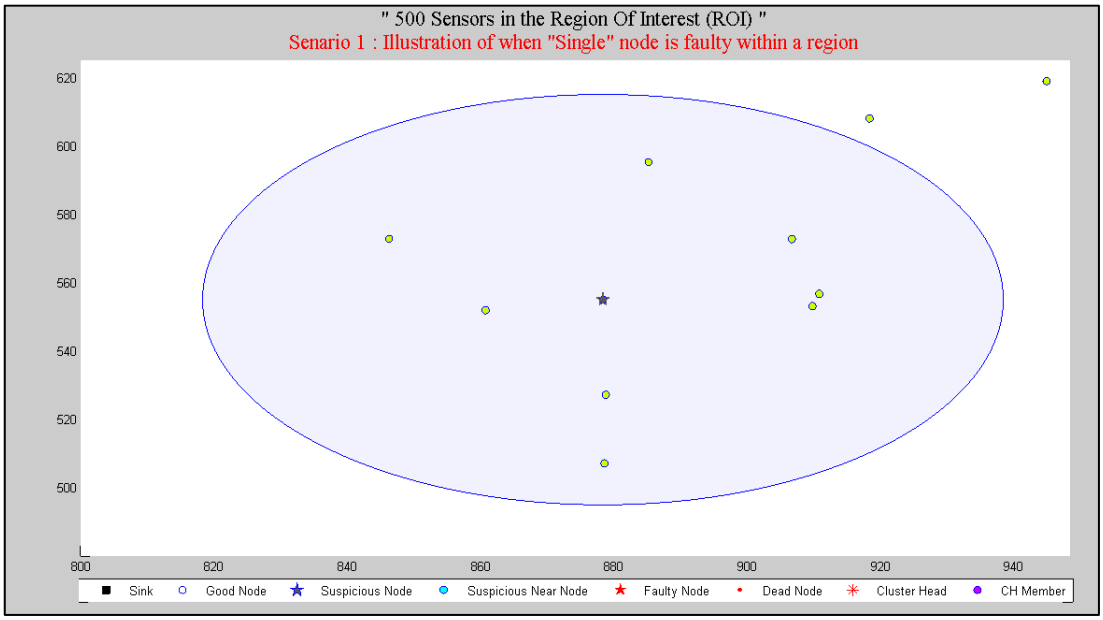

Fig. 7. One faulty node is present in the region D with intra-cluster neighbors (partial fault).

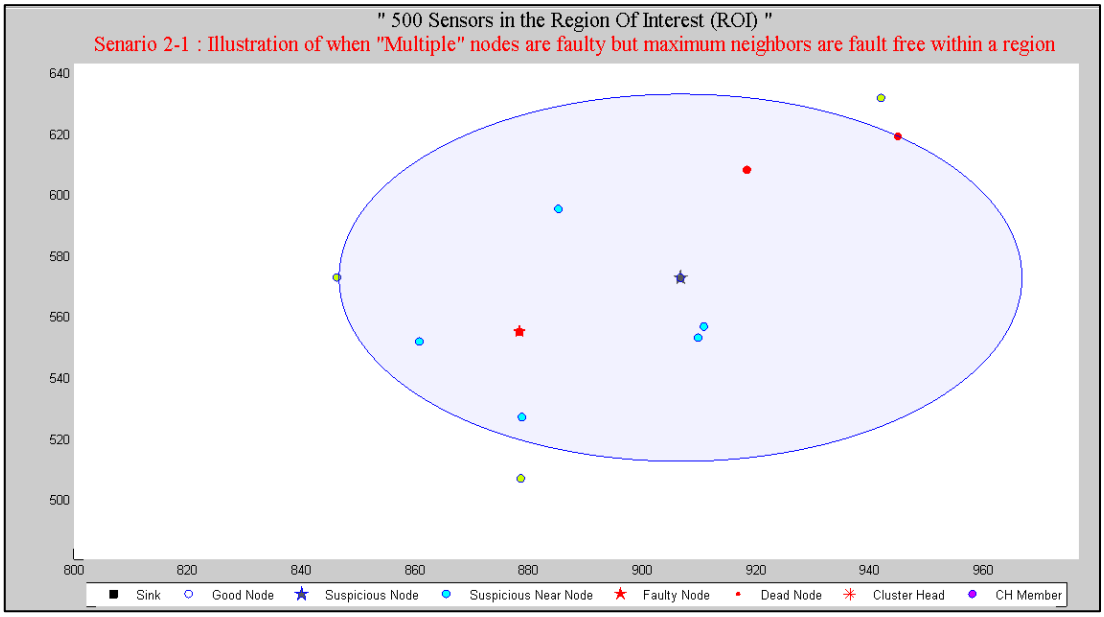

Fig. 8. Multiple faulty nodes fewer than the number of functional nodes Intra-cluster neighbors (partial fault).

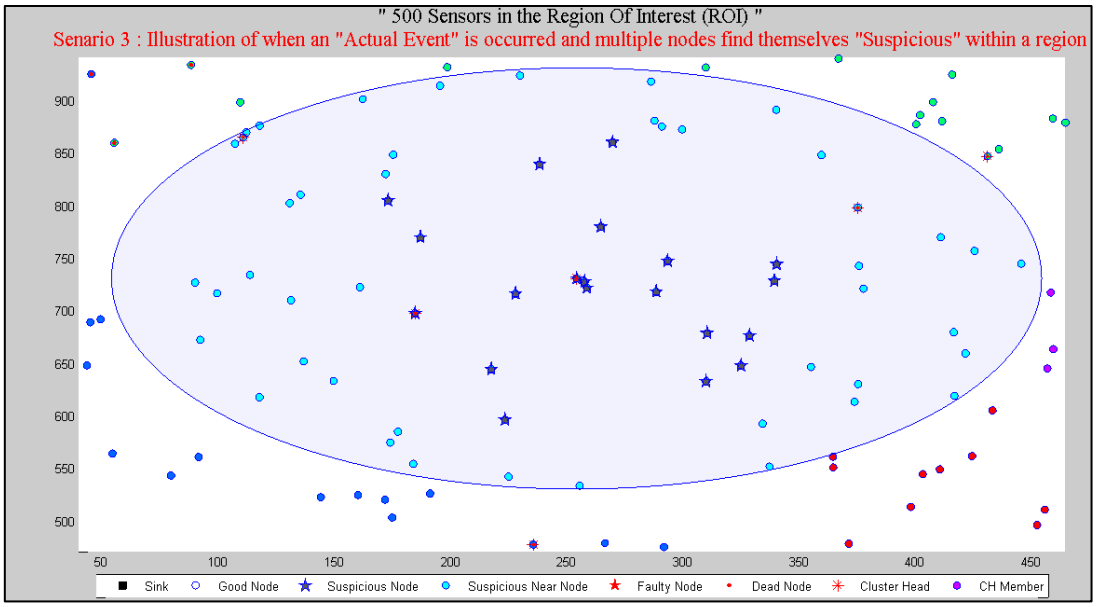

Fig. 9. A large number of nodes observe sudden variations in their sensed data (global fault).

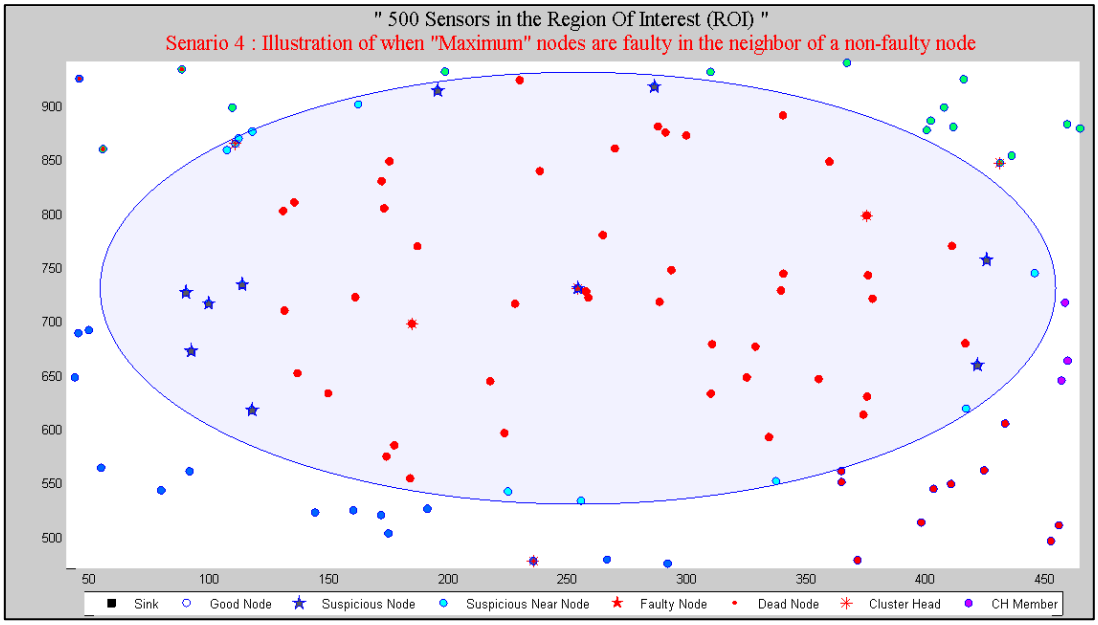

Fig. 10. Maximum number of faulty nodes around a functional node (global fault).

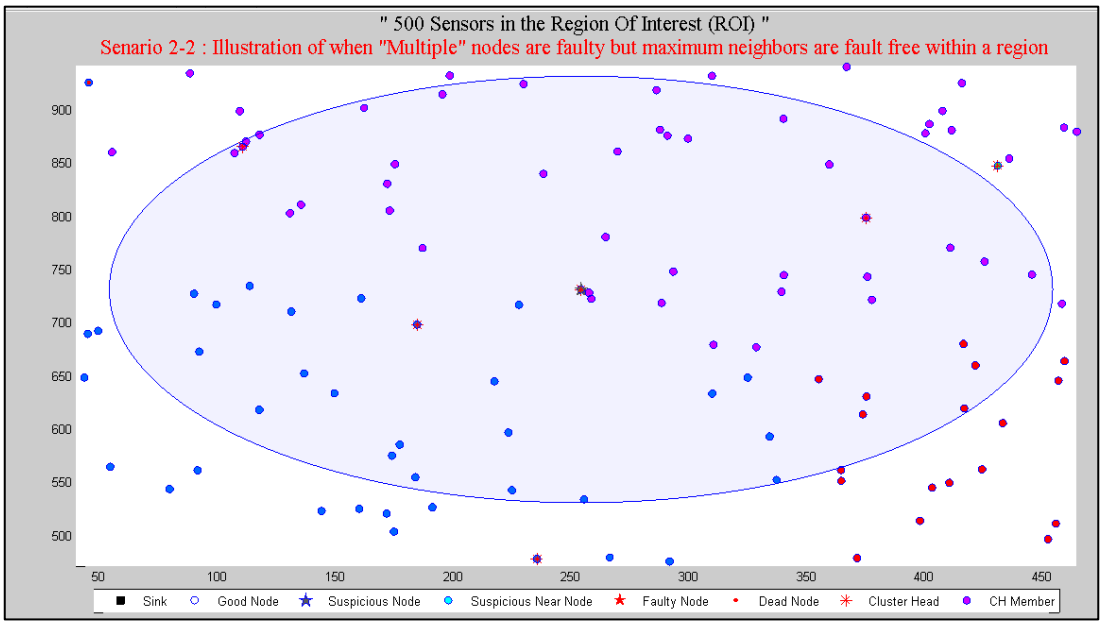

Fig. 11. Multiple faulty nodes fewer than the number of functional nodes inter-cluster neighbors (partial fault).

There are challenges in distributed network which if not considered, may lead the network function to break down. These challenges are described in this paper as exceptional cases which has described in section 3.7. An example of fault detection exception which occurs at scenario 2 (in BP) above has been shown in **Fig. 11**, which shows multiple faulty nodes with inter-cluster neighbors in the following section some exceptional cases have been described in detail.

3.7 Failure Exception Handling Procedure

In this section some exceptional cases have been described, which may affect the network function. There are two main exceptional cases. The first is the inter-cluster neighbors and the second one is the fault of a cluster head.

3.7.1 Case 1

The first exceptional state is the inter-cluster neighbors. A normal good node since in a distributed network is just connected to its associated cluster head and some neighbors may be located at the other clusters, so a new mechanism is provided as a Normal Failure Procedure (NFP), which is shown in **Fig. 12**.

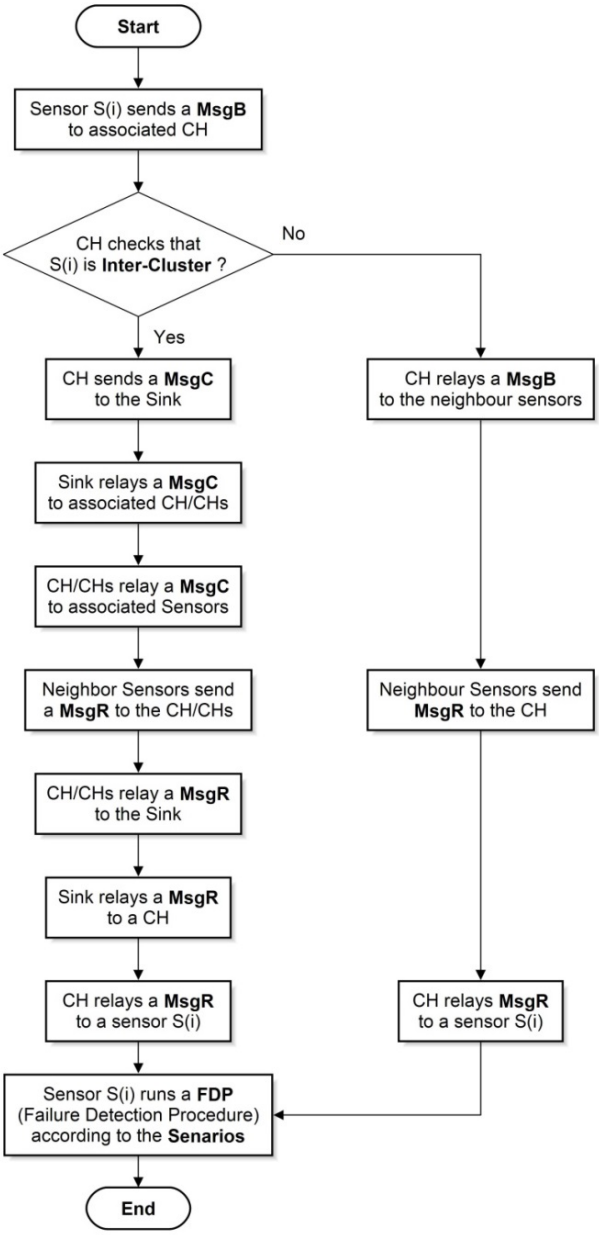

Fig. 12. Normal failure procedure (NFP)

3.7.2 Case 2

As the status of a cluster head changed to “SUSPICIOUS”, an alternate cluster head has been replaced to the current cluster head temporarily until end of fault detection procedure which has been shown in **Fig. 13**. Then if the status changed to “FAULTY”, an alternate cluster head has been replaced to the faulty cluster head, and vice versa.

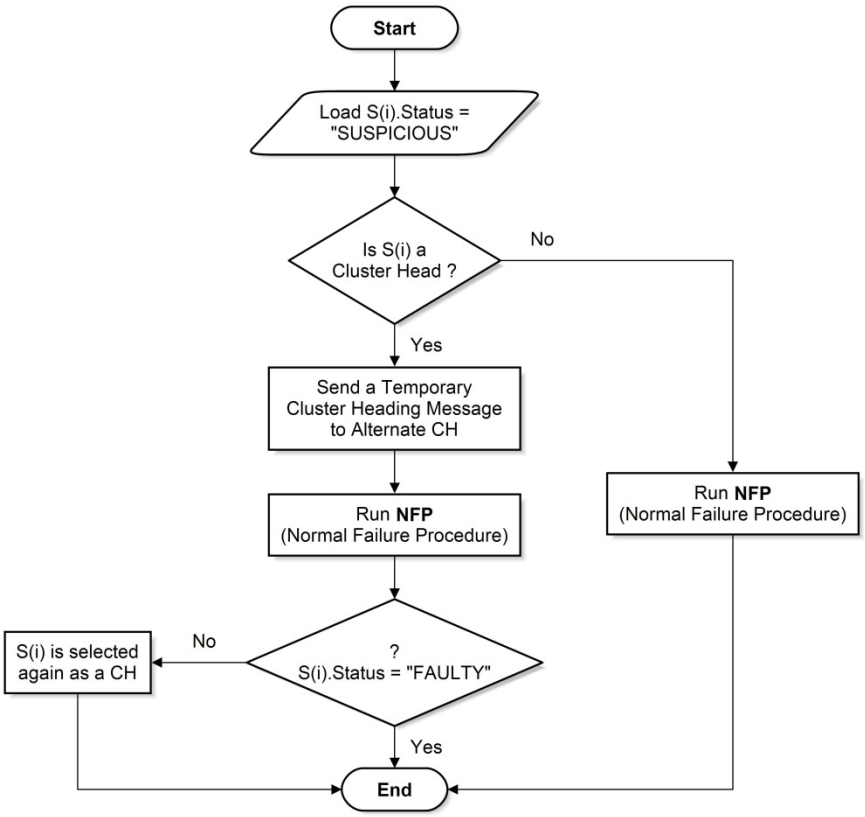

Fig. 13. Normal failure procedure (NFP)

4. Simulation Results

This section applies the proposed mechanism in Section 3 on a randomly generated data set and investigates the results. To this end, two simulations were conducted with 40 and then 70 randomly distributed nodes in a 2-D environment (based on BP). The results for BP and INRT were then compared in terms of different simulation parameters. The simulation was programmed in MATLAB. Moreover, the scalability of the INRT algorithm was simulated with different number of nodes (e.g. 500 nodes).

4.1 Simulation Assumptions

Calculating the energy required for transmission and receiving by the nodes from Eqs. 16 and 17:

$$E_{T,x}(l,d)=\begin{cases} l\times E_{elec}+l\times u_{fs}d^2,\ d<d_0 \\ l\times E_{elec}+l\times u_{mp}d^4,\ d\geq d_0 \end{cases} \tag{16}$$

$$E_{R,x}=l \times E_{elec} \tag{17}$$

where $E_{T,x}(l,d)$ is the transmission energy, $E_{R,x}$ is the receiving energy, d is the distance between two nodes or a node and a sink, E_{elec} is the consumed energy to run the transmission/receiving circuit, u_{fs} and u_{mp} are dependent on the transmitter amplifier, d is the distance threshold for the transmitter, and l is the length of transmitted data. Moreover, u_{fs} and u_{mp} are substituted with Eqs. 18 and 19, respectively.

$$u_{fs}=10\times10^{-12} \tag{18}$$

$$u_{mp}=0.0013\times10^{-12} \tag{19}$$

The initial parameters are given in **Table 1**.

Table 1. Initial parameters

Parameter	Value
Network Size	(1000 m * 1000 m)
Sink Position	(500 m, 500 m)
Node Number	500-40-70
Data Packet Size	4000 bit
Control Packet Size	100 bit
E_{elec}	50 nJ/bit
Sensor Type	mica mote2 with temperature
Initial Node Energy	0.5 J

4.2 Program Execution

The network included *n=40,70, and 500* number of alive nodes. The energy of each individual node and ultimately the overall energy of the network decrease overtime as the distributed algorithm rounds and sensed, control, and aggregation messages are transmitted. Each round of the algorithm for a distributed network respectively involves the following stages: sensing the parameter values, transmission of data to the CH based on the TDMA algorithm, aggregation of the data sensed by the members in the CH, and, finally, transmission of data to the sink. Each alive node may assume "good", "suspicious", or "faulty" states. Each node in the network belongs to only one cluster at a time, and each cluster is designated by a unique color. The number of colors is similar to the number of cluster (*Cluster No.*). Network time (*Time*) is based on real-time execution of the network under real conditions, from which the internal calculation time for the simulation is subtracted. In the network setup phase, the sink intelligently calculates the neighborhood radius (*R(threshold)*) and the neighborhood node number threshold (*N(threshold*)) based on the stable location of sensor nodes. Then, when a given node assumes a "suspicious" state once it suspects its read data, the CH identifies the respective node based on the neighborhood radius and neighborhood node number threshold and, if necessary, makes appropriate arrangements with the CHs in the adjacent clusters. In case the threshold for the number of suspicious nodes is not satisfied within the neighborhood radius of the suspicious node, the respective node is considered outlier in the network configuration and the CH extends the neighborhood radius as long as sufficient number of neighboring nodes are provided. As shown in **Fig. 14**, the total number of nodes with "good", "suspicious", and

"faulty" states are equal to the number of alive nodes, and the sum of alive and dead nodes are equal to the total number of nodes.

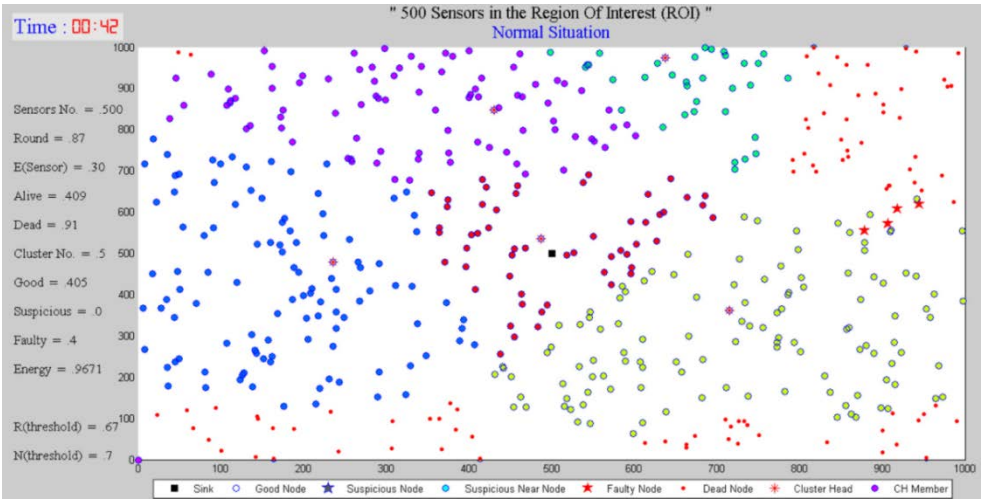

Fig. 14. Normal failure procedure (NFP)

Note that in the proposed method, only alive nodes with a "good" status contribute to the formation of clusters and transmission of sensed data. Suspicious nodes do contribute to cluster formation, but their sensed data are neglected by the CH until their status is fully determined.

4.3 Results

The proposed mechanism in this study and that of BP were compared in this section.

4.3.1 Network Throughput Diagram

Network throughput was initially assessed using the AODV protocol without applying the error mechanism. The results were then used to analyze the effect of the proposed method on the network throughput. The BP method as the reference method and the INRT method were applied to a similar network as shown in **Fig. 15**. The system automatically calculated the data with a neighborhood radius (D) of 100 m for the BP and INRT methods. A transmission range of 75 meter was considered for all sensors.

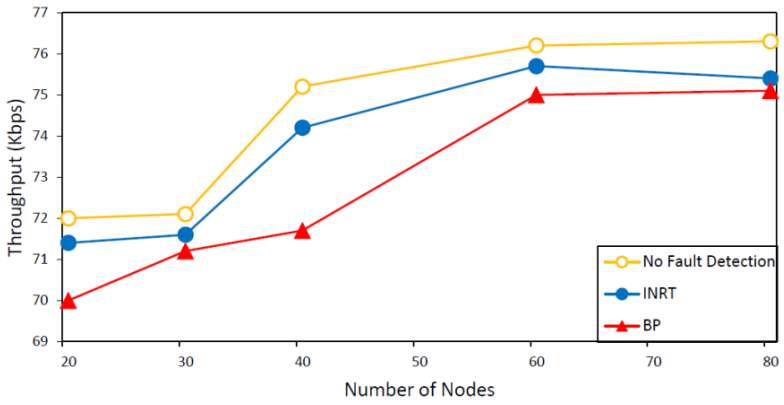

Fig. 15. Diagram of number of nodes with respect network throughput

As shown, the network throughput is considerably higher in this method compared to the BP method, since the fault coefficient declined and throughput was increased as the number of nodes increased. The overhead in this method is significantly lower than the BP method.

4.3.2 Fault Detection Accuracy Diagram

A higher fault detection accuracy in WSNs leads to higher fault-tolerance and consequently higher reliability. Diagram of fault detection accuracy with respect to the rate of failure is demonstrated in **Fig. 16**.

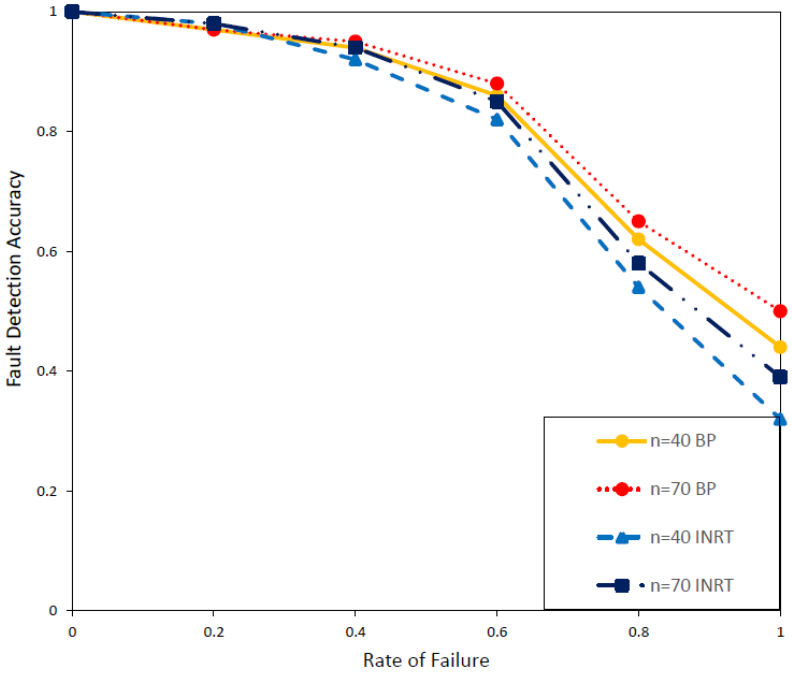

Fig. 16. Diagram of fault detection accuracy with respect to the rate of failure

As shown, the proposed method has a higher fault detection accuracy in both networks with 40 and 70 sensors.

4.3.3 Total Energy Consumption Diagram

In this section, the total energy of the network in INRT is investigated and compared with the BP method. The total remaining energy of the network over time for 40 and 70 sensors is compared in **Figs. 17,18**. As indicated, the total energy consumption is significantly decreased, which is due to the adaptation of the reference algorithm with the distributed structure. This is considered an advantage of the distributed algorithm which yields energy-efficient results. The assessment results suggest an increased network lifetime as one of the most important factors in the efficiency of WSNs.

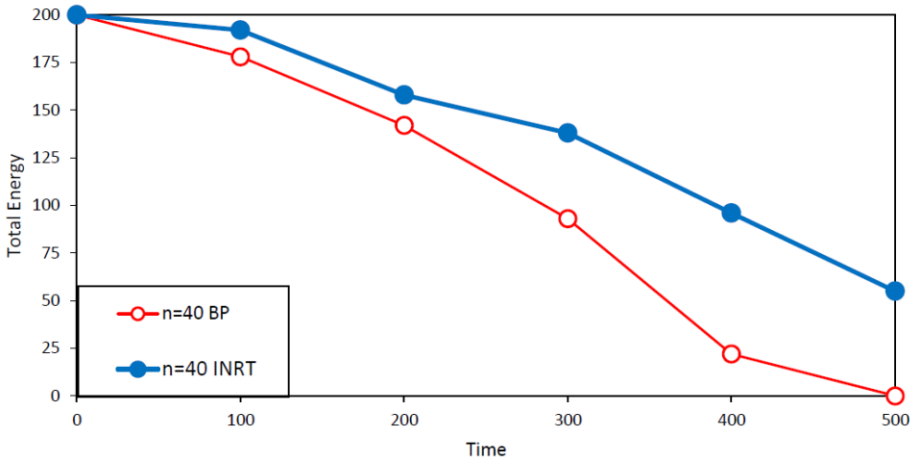

Fig. 17. The total energy consumption of the network with 40 nodes

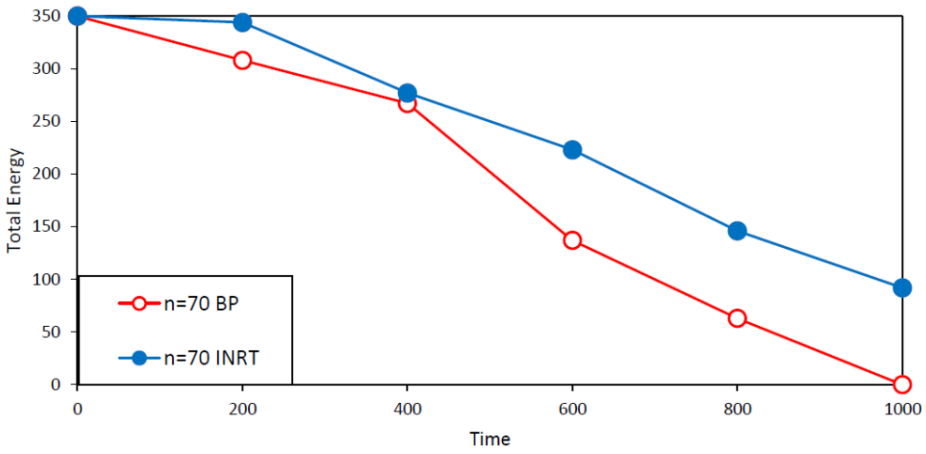

Fig. 18. The total energy consumption of the network with 70 nodes

5. Conclusion

In this paper, a new mechanism was used to improve fault tolerance of the system against soft (partial and global) and permanent faults in WSNs. This mechanism includes determining the intelligent neighborhood radius threshold (INRT), the threshold number of neighboring nodes, customizing the base paper algorithm (BP) for distributed systems, and redefining the BP scenarios to predict network behavior when running into soft and permanent faults. The experimental results indicate that the proposed mechanism was able to improve network throughput, fault detection accuracy, reliability, and network lifetime with respect to the BP.

References

[1] S. Chouikhi, I. El Korbi, Y. Ghamri-Doudane, and L. A. Saidane, “A survey on fault tolerance in small and large scale wireless sensor networks,” *Comput. Commun.*, vol. 69, pp. 22–37, 2015. Article (CrossLef Link).

[2] H. Alwan and A. Agarwal, "A survey on fault tolerant routing techniques in wireless sensor networks," in *Proc. of Proceedings of the 2009 Third International Conference on Sensor Technologies and Applications, SENSORCOMM'09*, 18-23 June, Athens, Glyfada, Greece: IEEE, pp. 366–371, 2009. Article (CrossLef Link).

[3] M. Zhao and T. W. Chow, "Wireless sensor network fault detection via semi-supervised local kernel density estimation," in *Proc. of Proceedings of the 2015 IEEE International Conference on Industrial Technology (ICIT)*, 17-19 March, Seville, Spain: IEEE, pp. 1495–1500, 2015. Article (CrossLef Link).

[4] Z. Gao, C. Cecati, and S. X. Ding, "A survey of fault diagnosis and fault-tolerant techniques—Part I: Fault diagnosis with model-based and signal-based approaches," *IEEE Trans. Ind. Electron.*, vol. 62, no. 6, pp. 3757–3767, 2015. Article (CrossLef Link).

[5] S. H. Oh, C. O. Hong, and Y. H. Choi, "A malicious and malfunctioning node detection scheme for wireless sensor networks," *Wirel. Sens. Netw.*, vol. 4, no. 03, p. 84, 2012. Article (CrossLef Link).

[6] Y. Yang, Q. Liu, Z. Gao, X. Qiu, and L. Rui, "Data clustering-based fault detection in WSNs," in *Proc. of Proceedings of the 2015 Seventh International Conference on Advanced Computational Intelligence (ICACI)*, 27-29 March, Wuyi, China: IEEE, pp. 334–339, 2015. Article (CrossLef Link).

[7] K. P. Sharma and T. P. Sharma, "A throughput descent and energy efficient mechanism for fault detection in WSNs," in *Proc. of Proceedings of the 2015 International Conference on Industrial Instrumentation and Control (ICIC)*, 28-30 May, Pune, India: IEEE, pp. 311–316, 2015. Article (CrossLef Link).

[8] H. Karimi *et al.*, "Implementing a reliable, fault tolerance and secure framework in the wireless sensor-actuator networks for events reporting," *Procedia Comput. Sci.*, vol. 73, pp. 384–394, 2015. Article (CrossLef Link).

[9] A. A. Abbasi and M. Younis, "A survey on clustering algorithms for wireless sensor networks," *Comput. Commun.*, vol. 30, no. 14, pp. 2826–2841, 2007. Article (CrossLef Link).

[10] A. Mahapatroa and P. M. Khilar, "Transient fault tolerant wireless sensor networks," *Procedia Technol.*, vol. 4, pp. 97–101, 2012. Article (CrossLef Link).

[11] N. Alrajei and H. Fu, "A survey on fault tolerance in wireless sensor networks," in *Proc. of Proceedings of the 2014 American Society for Engineering Education (ASEE) North Central Section Conference*, 4-5 April, Oakland University, Rochester Hills, USA, pp 366-371, 2014. Article (CrossLef Link).

[12] L. M. S. De Souza, H. Vogt, and M. Beigl, "A survey on fault tolerance in wireless sensor networks," *Sap Res. Braunschw. Ger.*, 2007. Article (Crosslef Link).

[13] K. Nitesh, M. Azharuddin, and P. K. Jana, "Energy efficient fault-tolerant clustering algorithm for wireless sensor networks," in Proc. of Proceedings of the 2015 *International Conference on Green Computing and Internet of Things (ICGCIoT)*, 8-10 Oct, Noida, India: IEEE, pp. 234–239. Article (CrossLef Link).

[14] P. Tang and T. W. Chow, "Wireless sensor network faulty scenes diagnosis using high dimensional Neighborhood Hidden Conditional Random Field," in *Proc. of Proceedings of the 2015 13th International Conference on Industrial Informatics (INDIN)*, 22-24 July, Cambridge, UK: IEEE, pp. 1130–1135, 2015. Article (CrossLef Link).

[15] S. Vigneshwari and S. Devi, "Fault Diagnosis inWSN Using Optimized Neighborhood Hidden Conditional Random Field," *International Journal of Modern Trends in Engineering and Science*, Vol 4, pp. 4-6, 2017. Article (CrossLef Link).

[16] M. Azharuddin and P. K. Jana, "A PSO based fault tolerant routing algorithm for wireless sensor networks," *Information systems design and intelligent applications*, Springer, pp. 329–336, 2015. Article (CrossLef Link).

[17] G. Venkataraman, S. Emmanuel, and S. Thambipillai, "Energy-efficient cluster-based scheme for failure management in sensor networks," *IET Commun.*, vol. 2, no. 4, pp. 528–537, 2008. Article (CrossLef Link).

[18] J. Jianfeng, "Research on Hierarchical Routing Algorithm of Wireless Sensor Networks," in *Proc. of Proceedings of the 2015 2nd International Conference on Information Science and Control Engineering (ICISCE)*, 24-26 April, Shanghai, China: IEEE, pp. 429–432, 2015. Article (CrossLef Link).
[19] K. Rajeswari and S. Neduncheliyan, "Genetic algorithm based fault tolerant clustering in wireless sensor network," *IET Commun.*, vol. 11, no. 12, pp. 1927–1932, 2017. Article (CrossLef Link).
[20] T. P. Vieira, P. E. Almeida, and M. R. Meireles, "Intelligent fault management system for wireless sensor networks with reduction of power consumption," in *Proc. of Proceedings of the 2017 IEEE 26th International Symposium on Industrial Electronics (ISIE)*, 19-21 June, Edinburgh, UK: IEEE, pp. 1521–1527, 2017. Article (CrossLef Link).
[21] Zidi, S., Moulahi, T. and Alaya, B., "Fault Detection in Wireless Sensor Networks Through SVM Classifier," *IEEE Sensors Journal*, 18(1), pp.340-347, 2018. Article (CrossLef Link)
[22] Abdul-Salaam, G., Abdullah, A.H. and Anisi, M.H., 2017. "Energy-efficient data reporting for navigation in position-free hybrid wireless sensor networks," IEEE Sensors Journal, 17(7), pp.2289-2297. Article (CrossLef Link)
[23] Banerjee, I., Chanak, P., Rahaman, H., & Samanta, T., "Effective fault detection and routing scheme for wireless sensor networks," Computers & Electrical Engineering, 40(2), 291-306, 2014. Article (CrossLef Link)
[24] Lu Wei, Yang Yuwang, Zhao Wei and Wang, "Practical Node Deployment Scheme Based on Virtual Force for Wireless Sensor Networks in Complex Environment," KSII Transactions on Internet and Information Systems, vol. 9, no. 3, pp. 990-1013, 2015. Article (CrossLef Link)

Mojtaba Montazeri is an instructor in Islamic Azad University, Iran. He received the B.Sc degree in Computer Engineering-Software from Islamic Azad University of Larestan Branch, Larestan, Iran in 2014 and M.Sc. degree in Computer Engineering-Software from Islamic Azad University of Fars Science and Research Branch, Tehran, Iran, in 2016. His research interests are WSNs, Data Mining and Image Processing.

Rasoul Kiani is an instructor in Islamic Azad University, Iran. He received the B.Sc degree in Computer Engineering-Software from Torbat-e-Heydarieh University, Torbat-e-Heydarieh, Iran in 2012 and M.Sc. degree in Computer Engineering-Software from Islamic Azad University of Fars Science and Research Branch, Tehran, Iran, in 2016. His research interests are WSNs, Data Mining and Image Processing.