Distributed Fault Detection of Wireless Sensor Networks

Jinran Chen, Shubha Kher, and Arun Somani
Dependable Computing and Networking Lab
Iowa State University
Ames, Iowa 50010
{jrchen, shubha, arun}@iastate.edu

ABSTRACT

Wireless Sensor Networks (WSNs) have become a new information collection and monitoring solution for a variety of applications. Faults occurring to sensor nodes are common due to the sensor device itself and the harsh environment where the sensor nodes are deployed. In order to ensure the network quality of service it is necessary for the WSN to be able to detect the faults and take actions to avoid further degradation of the service. The goal of this paper is to locate the faulty sensors in the wireless sensor networks. We propose and evaluate a localized fault detection algorithm to identify the faulty sensors. The implementation complexity of the algorithm is low and the probability of correct diagnosis is very high even in the existence of large fault sets. Simulation results show the algorithm can clearly identify the faulty sensors with high accuracy.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless Communication

General Terms

Algorithm, Performance, Design

Keywords

Wireless sensor networks, Fault tolerance, Distributed algorithm

1. INTRODUCTION

The dramatic advances in wireless communication and electronics have enabled the development of low cost, low power, and multifunctional wireless sensor nodes which consist of sensing, data processing, and communication components. These tiny sensor nodes can easily be deployed into a designated area to form a wireless network and perform specific functions. With recent intensive research in this area, wireless sensor networks have been applied in various areas, such as environment and habitat monitoring, ecophysiology, condition-based equipment maintenance, disaster management, and emergency response.

Due to the low cost and the deployment of a large number of sensor nodes in an uncontrolled or even harsh or hostile environments, it is not uncommon for the sensor nodes to become faulty and unreliable. The networks must exclude the faulty sensors to ensure the network quality of service. To identify the faulty sensor nodes is not trivial at all because of the existing challenges. Sensor nodes are powered by batteries, which are considered as limited resources. It is very expensive for the base station to collect information from every sensor and identify faulty sensors in a centralized manner. Different applications may require the fault detection to be conducted in a real-time mode with low latency or high throughput. Therefore, a localized and distributed generic algorithm for each node is highly preferred in wireless sensor networks.

Traditional testing and fault detection in computer systems are carried out in the form of built-in self-test (BIST) subsystems or in the form of signature verification subsystems. Built-in self-repair (BISR) techniques are often used to improve the yield in DRAM memories. Our approach adopted the mutual testing method at the processor level where each processing element is capable of testing its neighbors [1]. A processing element tests another processing element and generates a test result based on the success of the test results. The test results may be arbitrary because the tester itself can be faulty. A processor is determined to be good or faulty by diagnosing the collection of all such test results. The algorithm proposed in [1] is applied to regular connected multiprocessor systems. However, the topology of wireless sensor networks is not regular. Therefore, each sensor must maintain a certain number of neighbors, that is the degree of the network must be high. In a densely deployed sensor environment, this condition can be easily realized.

Our major contribution of this paper is the development of a generic localized fault detection algorithm for wireless sensor networks. The paper is organized as follows. We first review the literature in the fault detection area in Section 2. Then, we define the network model and fault model in Section 3. The distributed algorithm for faulty sensor identification is proposed in Section 4. A simple example is also illustrated in Section 4. Performance analysis is presented in Section 5. Simulation results are reported in Section 6. Finally we conclude the paper and suggest future work in Section 7.

2. RELATED WORK

In this section, we briefly review the related works in the area of fault detection in wireless sensor networks.

Fault tolerance in VLSI-based systems and fault tolerance in distributed systems have been studied intensively in the past. In VLSI systems, fault tolerance has been addressed at all levels of abstraction, including circuit level, logic level, register transfer level, program level, and system level.

In [2], a watchdog processor is used for concurrent system-level error detection techniques. A watchdog processor is a small and simple coprocessor that detects errors by monitoring the behavior of a system. They showed that a large number of errors can be detected by monitoring the control flow and memory-access behavior.

In distributed systems, fault detection and identification has long been the subject of active research. A large number of testing connections among units in multiprocessor fault diagnosis, which is too expensive and even not allowed in many applications. In [3], they proposed a general approach to fault diagnosis that is widely applicable and only needs a limited number of connections among units. The algorithm uses a majority vote among the neighbors of a unit to determine the status of the unit.

A distributed diagnosis algorithm to locate faulty PE's in large-scale regular interconnected structures based on the concepts of system-level diagnosis is developed in [1]. This algorithm can either work in a systolic manner or be executed on a supervisory processor to identify the faulty processor. They showed the probability of correct diagnosis is very high even in the presence of a large faulty sets.

Recently, fault tolerance and management in WSNs have drawn attention from the researchers in the area. In [5], a failure detection scheme using a management architecture for WSNs called MANNA, is proposed and evaluated. The scheme creates a manager located externally to the WSN. It has the global vision of the network and can perform complex tasks that would not be possible inside the network. Management activities take place when sensor nodes are collecting and sending temperature data. Every node will check its energy level and send a message to the manager/agent whenever there is a state change. The manager can then obtain the coverage map and energy level of all sensors based upon the collected information. To detect node failure, the manager sends GET operations to retrieve the node state. Without hearing from the nodes, the manager will consult the energy map to check its residual energy. In this way, MANNA architecture is able to locate faulty sensor nodes. However, this approach require an external manager to perform the centralized diagnosis. And the communication between nodes and the manager is too expensive for WSNs.

In [6], a taxonomy for classification of faults in sensor networks and the first on-line model-based testing technique are introduce. The technique considers the impact of readings of a particular sensor on the consistency of multi-sensors fusion. The sensor is most likely to be faulty if the elimination of it significantly improves the consistency of the results. A way to distinguish a random noise is to run a maximum likelihood or Bayesian approach on the multi-sensor fusion measurements.If the accuracy of final results of multi-sensor fusion improve after running these procedure, a random noise should exist. To get a consistent mapping of the sensed phenomena, different sensors' measurements need to be combined in a model. This cross-validation-based technique can be applied to a broad set of fault models. It is generic and can be applied to an arbitrary system of sensors that use an arbitrary type of data fusion. However, this technique is centralized. Sensor node information must be collected and sent to the base station to conduct the on-line fault detection.

An energy efficient fault-tolerant detection scheme is proposed in [7] to introduce the sensor fault probability into the optimal event detection process. The optimal detection error was shown to decrease exponentially with the increase of the neighborhood size. They attempted to disambiguate events from both noise related measurement error and sensor fault and limit the effects of faulty sensor on the event detection accuracy. The measurement noise and sensor faults are likely to be stochastically unrelated, while event measurements are likely to be spatially correlated. The Bayesian detection scheme in [7] selects the minimum neighbors for a given detection error bound such that the communication volume is minimized during the fault correction. Luo et al. in [7] did not explicitly attempt to detect faulty sensors, instead the algorithms they proposed improve the event detection accuracy in the presence of faulty sensors.

In [8], a faulty sensor identification algorithm is developed and analyzed. The algorithm is purely localized and requires low computational overhead, it can be easily scaled to large sensor networks. In the algorithm, the reading at a sensor is compared with its neighbors' median reading. If the difference is large or large but negative, the sensor is very likely to be faulty. Although this algorithm works for large size of sensor networks, the probability of sensor faults needs to be small. If half of the sensor neighbors are faulty and the number of neighbors is even, the algorithm cannot detect the faults as expected. The paper also mentioned the need of sensors' physical location, which require expensive GPS or other techniques to realize. Our localized faulty detection algorithm does not need any physical position and works for large size of faulty sensors. Even when half neighbors fail, it can still successfully identify the faulty sensors.

3. NETWORK MODEL AND FAULT MODEL

We assume sensors are randomly deployed in the interested area and all sensors have a common transmission range. The area is assumed to be entirely covered by the sensors. As shown in Figure 3, the dark circles represent faulty sensors and the light gray circles are good sensors. There could be a failure occurring in a certain area as illustrated in the figure. All sensors in the area go out of service. Since we are depending on majority voting, we assume that each sensor has at least 3 neighboring nodes. Because a large amount of sensors are cast into the interested area to form a wireless network, this condition can be easily obtained. Each sensor node is able to locate the neighbors within its transmission range through a broadcast/acknowledge protocol.

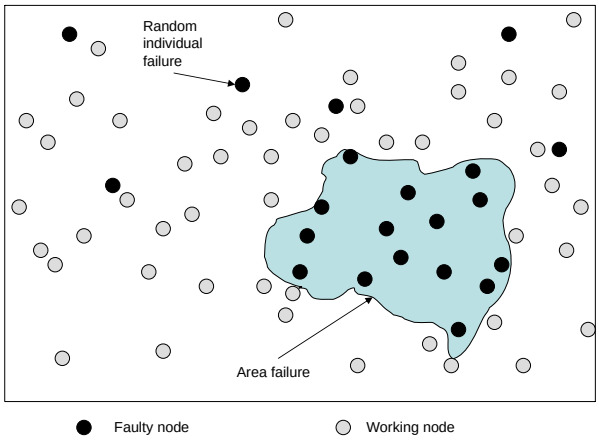

Figure 1: Sensor nodes randomly deployed over an area

Faults may occur at different levels of the sensor network, such as physical layer, hardware, system software, and middleware [4]. In this paper, we focus on hardware level faults by assuming all system software as well as the application software are already fault tolerant. The first of the two groups of components at hardware level consists of a computation engine, storage subsystem and power supply infrastructure, which are very reliable. Another group of components are sensors and actuators which are most prone to malfunctioning. Because in the first group of components the heterogeneous BISR fault tolerant schemes will provide the targeted level of fault tolerance [4], we only consider the sensor faults which include three types of faults: calibration systematic error, random noise error, and complete malfunctioning. Nodes are still capable of receiving, sending, and processing when they are faulty.

4. LOCALIZED FAULTY SENSOR DETECTION

In this section, we first give some definitions for the variables. Then, we present the localized fault detection algorithm.

4.1 Definitions

We list the notations used in our algorithm and analysis below,

- n: total number of sensors;
- p: probability of failure of a sensor;
- k: number of neighbor sensors;
- S: set of all the sensors;
- $N(S_i)$: set of the neighbors of S_i;
- x_i: measurement of S_i;
- d_{ij}^t: measurement difference between S_i and S_j at time t, $d_{ij}^t = x_i^t - x_j^t$;
- $\Delta t_l = t_{l+1} - t_l$;
- $\Delta d_{ij}^{\Delta t_l}$: measurement difference between S_i and S_j from time t_l to t_{l+1}, $\Delta d_{ij}^{\Delta t_l} = d_{ij}^{t_{l+1}} - d_{ij}^{t_l} = (x_i^{t_{l+1}} - x_j^{t_{l+1}}) - (x_i^{t_l} - x_j^{t_l})$;
- c_{ij}: test between S_i and S_j, $c_{ij} \in \{0, 1\}$, $c_{ij} = c_{ji}$;
- $\theta 1$ and $\theta 2$: two predifined threshold values;
- T_i: tendency value of a sensor, $T_i \in \{\mathrm{LG}, \mathrm{LT}, \mathrm{GD}, \mathrm{FT}\}$;
- $Maxd$: an estimate of propagation distance from a set of identified good sensors in the first round of the algorithm iterations. The worst case is n, the best case is $\log n$, and we take a reasonable $\sqrt{n}$.

Sensors are considered as neighboring sensors if they are within the transmission range of each other. Each node regularly sends its measured value to all its neighbors.

We are interested in the history data if more than half of the sensor's neighbors have a significantly different value from it. We can use this $\Delta d_{ij}^{\Delta t_l}$ to find if the current measurement is different from previous measurement. If the measurements change over the time significantly, it is more likely the sensor is faulty.

A test result c_{ij} is generated by sensor S_i based on its neighbor S_j's measurements using two variables, d_{ij} and Δd_{ij}, and two predefined threshold value $\theta 1$ and $\theta 2$. If a sensor is faulty, it can generate arbitrary measurements. If c_{ij} is 0, most likely either both S_i and S_j are good or both are faulty. Otherwise, if c_{ij} is 1, S_i and S_j are most likely in different status.

Sensors can be either LG or LF, determined by using test value from its neighboring sensors. Each sensor sends its tendency value to all its neighbors. The number of the LG sensors with coincident test results determines whether the sensors are GD or FT. That is $\forall S_j \in N(S_i)$ and $T_j = LG$, $\sum(1 - c_{ij}) - \sum c_{ij} = \sum(1 - 2c_{ij})$ must be greater or equal to $\lceil |N(S_i)|/2 \rceil$ to claim S_i is good. In other words, a good S_i will be diagnosed as GD in the first round if it has less than $k/4$ bad neighbors. The probability of a sensor being diagnosed as GD in first round of iteration is:

$$\sum_{i=0}^{i=\lfloor k/4 \rfloor} \binom{k}{i} p^i (1-p)^{k-i} \tag{1}$$

where i is the bad neighboring sensors.

If a GD sensor is found in the network, its test result can be used to diagnose other sensors' status. The information can be propagated through the whole network to diagnose all other sensors as good or faulty.

If the diagnosis is consistent with the test results, the diagnosis is valid. If there's no sensor being diagnosed, all its neighbors are either not diagnosed or are diagnosed as faulty.

4.2 Algorithm

The localized faulty sensor detection algorithm is summarized in the following:

Algorithm 1 (Localized Fault Detection):

Step 1: Each sensor S_i tests every member of $S_j \in N(S_i)$ to generate test $c_{ij}\{0,1\}$ using the following method:
1: Each sensor S_i, set $c_{ij} = 0$ and compute d_{ij}^t;
2: IF $|d_{ij}^t| > \theta 1$ THEN
3: Calculate $\Delta d_{ij}^{\Delta t_l}$;
4: IF $|\Delta d_{ij}^{\Delta t_l}| > \theta 2$ THEN $c_{ij} = 1$;

Step 2: S_i generates a tendency value T_i based upon its neighboring sensors' test value:
1: IF $\sum_{S_j \in N(S_i)} c_{ij} < \lceil |N(S_i)|/2 \rceil$, where $|N(S_i)|$ is the number of the S_i's neighboring nodes THEN
2: $T_i =$ LG;
3: ELSE $T_i =$ LF;
4: Communicate T_i to neighbors;

Step 3: Compare the number of S_i's LG neighboring nodes with different test results to determine its status:
1: IF $(\sum_{S_j \in N(S_i) \text{and} T_j = LG}(1-2c_{ij}) \geq \lceil |N(S_i)|/2 \rceil$ THEN
2: $T_j =$ GD;
3: Communicate T_i to neighbors;

Step 4: For the remaining undetermined sensors, do the following steps in parallel for $Maxd$ cycles:
1: FOR $i = 1$ to n
2: IF $T_i =$ LG or $T_i =$ LF THEN
3: IF $T_j =$ GD $\forall S_j \in N(S_i)$, THEN
4: IF $c_{ji} = 0$ THEN
5: $T_i =$ GD;
6: ELSE $T_i =$ FT;
7: ELSE repeat
8: Communicate T_i to neighbors;

Step 5: If ambiguity occurs, then the sensor's own tendency value determine its status:
1: FOR each S_i, IF $T_j = T_k = GD$ $\forall S_j, S_k \in N(S_i)$, where $j \neq k$, and IF $c_{ji} \neq c_{ki}$ THEN
2: IF $T_i = LG$ (or LF) THEN
3: $T_i = GD$ (or FT)

End Algorithm 1

Test results c depends on the threshold θ, which can be defined according to various applications at the deployment time. In step 1, we can also set two $\theta 1$ and $\theta 2$ values to be different as we desire. Step 5 is a validation check to make sure the diagnosis is consistent throughout the entire network.

4.3 Example

In this section, we present an example to illustrate our algorithm. Fig.2 shows a partial set of sensor nodes in a wireless sensor network with some faulty nodes. Nodes $S_1 - S_9$ inside the circle area are the nodes we are interested in. If the two nodes are neighbors, they are connected by dotted line. Communication between nodes outside the circle are not shown in the figure.

Each node inside the interested area are tested by its neighbors. Test results are either 0 or 1 depending upon the measurement difference and threshold value θ. Tendency value T_i is finalized at the third iteration. Table 1 lists the analysis results obtained by applying the Localized Fault Detection Algorithm. Four out of nine sensor nodes in the area are faulty. The other five nodes are good and there is no ambiguity occurring in this example. Each node's neighbors with GD tendency value generate the same testing results when they determine the node's status.

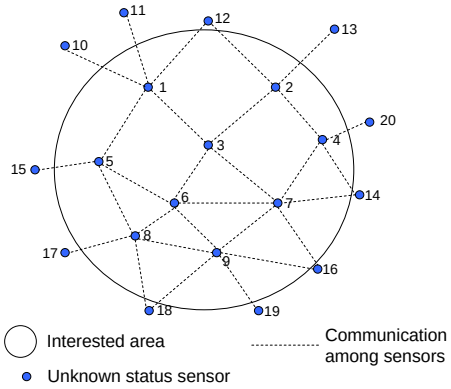

Figure 2: A partial set of sensor nodes in a wireless sensor networks with faulty sensors

First, each of $S_1 - S_9$ generates c_{ij} test results for all their neighbors in the way as specified in step 1 of our algorithm. The results are shown under the 2nd and 3rd columns of Table 1.

Secondly, $S_1 - S_9$ decide their own tendency value, $T_1 - T_9$. If the summation of test results is less than half of the number of its neighbors, the sensor is likely good. Otherwise, it's likely faulty. For example, for S_1, $\sum_{s_j \in N(S_1)} c_{1j} = 1 < \lceil |N(S_1)|/2 \rceil = 3 \Rightarrow T_1 =$ LG. The same test is done for all other nodes. For S_2, $\sum_{s_j \in N(S_2)} c_{2j} = 3 > \lceil |N(S_2)|/2 \rceil = 2 \Rightarrow T_2 =$ LF. We assume that sensors outside the circle can decide their tendency value in the same way.

Then, we need to find GD sensors from all the sensors. Look at S_1, as specified in step 3 of our localized fault detection algorithm, $\sum_{s_j \in N(S_1) \text{and} T_j = \text{LG}}(1-2c_{1j}) = 3 > \lceil |N(S_1)|/2 \rceil \Rightarrow T_1 = GD$. We obtained all the values under the Iteration 1 column in Table 1 from this step.

Finally, by using the GD sensors, we can test other non-GD sensors to find out their status base upon the test results. The values under Iteration 2 column in Table 1 are generated from this step. The last step is to check if there is any ambiguity between any neighbors test results. All test results are consistent in this example.

From this localized fault detection algorithm and the above example, we make the following observations:
1. A sensor node S_i's tendency value can be LG if S_i is good and has $\lceil N(S_i)/2 \rceil$ or more good neighbors. It can also be LG if S_i is faulty and has over $\lceil N(S_i)/2 \rceil$ faulty neighbors.

Table 1: Analysis of Faults in Fig.2

S_i	S_j with $c_{ij}=0$	S_j with $c_{ij}=1$	T_i in Iterations		
			0	1	2
1	3,5,11,12	10	LG	GD	GD
2	4	3,12,13	LF	LF	FT
3	1,7	2,6	LG	GD	GD
4	2	7,14,20	LF	LF	FT
5	1,15	6,8	LG	GD	GD
6	8	3,5,7,9	LF	LF	FT
7	3,9,14	4,6,16	LG	GD	GD
8	6,17	5,9,18	LF	LF	FT
9	7,18,19	6,8,16	LG	GD	GD

2. If no GD sensor is determined at step 3, the network can not determine the sensor nodes' status at all and the algorithm will exit at this step. Only at least one sensor node is diagnosed as GD in step 4, can this algorithm continue to execute and determine more sensors' status.
3. A faulty sensor can only be diagnosed as good sensor in step 3 because the system will not use any incorrect information in step 4.
4. A good sensor can be diagnosed as a faulty sensor only when some faulty sensors are diagnosed as good nodes in step 3. By using these observations, we can further analyze the algorithm.

5. ANALYSIS OF THE ALGORITHM

We define that the diagnosis of faulty sensors is correct if no good sensor is diagnosed as faulty and no faulty sensor is diagnosed as good. The diagnosis is complete if all sensor nodes are identified as faulty or good in a predefined time. An incorrect diagnosis happens when a good sensor is labeled as faulty or a faulty sensor is labeled as good. A diagnosis is incomplete if any node's status cannot be diagnosed in the network. A complete and correct diagnosis is desired. Generally, an incorrect diagnosis is unacceptable because the error information may be propagated to the base station or users. However, an incomplete diagnosis is acceptable under certain circumstances. Our proposed algorithm minimizes the likelihood of incorrect diagnosis.

The probability of correct and complete faulty sensor diagnosis is computed and analyzed in the following. Let p be the probability of failure of node S_i. Let $k = |N(S_i)|$. A sensor node with tendency value LG can be either good or faulty. A sensor node with tendency value LF can also be either good or faulty. The probability of a good sensor node having a likely good tendency value is:

$$P_{\text{glg}} = (1-p) \sum_{i=0}^{\lceil k/2 \rceil - 1} \binom{k}{i} p^i (1-p)^{k-i} \tag{2}$$

The probability of a good sensor with a likely faulty tendency value is:

$$P_{\text{glf}} = (1-p) \sum_{i=0}^{\lceil k/2 \rceil - 1} \binom{k}{i} (1-p)^i p^{k-i} \tag{3}$$

The probability that a sensor is faulty and has a likely good tendency value is:

$$P_{\text{flg}} = p \sum_{i=0}^{\lceil k/2 \rceil - 1} \binom{k}{i} (1-p)^i p^{k-i} \tag{4}$$

The probability that a sensor is faulty and has a likely faulty tendency value is:

$$P_{\text{flf}} = p \sum_{i=0}^{\lceil k/2 \rceil - 1} \binom{k}{i} (1-p)^{k-i} p^i \tag{5}$$

From the observations in Section 4, a faulty sensor's LG value is determined by its neighboring nodes. Only when at least half of its neighbors are faulty and have LG tendency value, the faulty sensor will be diagnosed as good. The probability of a faulty sensor being diagnosed as good is:

$$P_{\text{FG}} = p \sum_{a=\lceil \frac{k}{2} \rceil}^{k} \{ \binom{k}{a} P_{\text{flg}}^a \sum_{b=0}^{a-\lceil \frac{k}{2} \rceil} [P_{\text{glg}}^b \sum_{c=0}^{k-a-b} (P_{\text{flf}}^{k-a-b-c} P_{\text{glf}}^c)] \} \tag{6}$$

The probability that none of the faulty sensors is diagnosed as good in the entire network is:

$$P_{\text{NFG}} = \sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i} (1-P_{\text{FG}})^i \tag{7}$$

A good sensor is not diagnosed as a good sensor only when the difference between LG faulty sensors and LG good sensors is less than half of its neighbors. The probability of a good sensor not being diagnosed as good node is:

$$P_{\text{G}\overline{\text{G}}} = (1-p) \sum_{a=0}^{\lceil \frac{k}{2} \rceil - 1} \{ \binom{k}{a} P_{\text{flg}}^a \sum_{b=0}^{a} [P_{\text{glg}}^b \sum_{c=0}^{k-a-b} (P_{\text{flf}}^{k-a-b-c} P_{\text{glf}}^c)] \} \tag{8}$$

The probability that no good sensor nodes are diagnosed as good in the entire network is:

$$P_{\text{N}\overline{\text{G}}} = \sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i} P_{\text{G}\overline{\text{G}}}^{n-i} \tag{9}$$

From Equation 7 and Equation 9, P_{NFG} and $P_{\text{N}\overline{\text{G}}}$ approach 1 and 0 respectively when network size goes to exponentially large. Since the number of sensors in WSN can be hundreds and network size is relatively large, the probability that none of the faulty sensors is diagnosed as good approaches one. The probability that all good sensors are not being diagnosed as good is very small, which is approaching 0.

Table 2: Probability of No Faulty Sensor Diagnosed as Good

p	Average Number of Sensors				
	4	6	10	15	20
0.05	1	1	1	1	1
0.10	1	1	1	1	1
0.15	1	1	1	1	1
0.20	0.9999	1	1	1	1
0.25	0.9994	1	1	1	1

Table 3: Probability of No Good Sensor Diagnosed as Good

p	Average Number of Sensors				
	4	6	10	15	20
0.05	8.93E-66	8.88E-66	0	0	0
0.10	1.04E-50	1.00E-50	0	0	0
0.15	7.17E-42	6.39E-42	0	0	0
0.20	1.43E-35	1.14E-35	0	0	0
0.25	1.16E-30	8.14E-31	0	0	0

Tables 2 and 3 show the probabilities P_{NFG} and $P_{\text{NG}\overline{\text{G}}}$ under different probabilities of failure of sensors and with different average number of neighbors. The results demonstrate that with large network size almost all the good and faulty sensors will be diagnosed correctly. The probabilities calculated in the tables show that our algorithm performs well. However, in real situation, when sensors are deployed in the field randomly, they may not have enough number of neighbors for the correct and complete analysis. This can cause the incorrect diagnosis, resulting faulty sensors being diagnosed as good or good sensor being diagnosed as faulty.

6. SIMULATION RESULTS

For simulation we used C++ as the tool. An example simulation scenario composed of total 1024 sensor nodes are randomly deployed in a region of size 32×32 units as shown in Figure 3. The measurement parameter x_i is considered to be temperature. We set the values of x_i as good and faulty with ranges as follows, "Good" = 70-75 degrees and "Faulty" as 100-105 degrees. Also transmission range was chosen to ensure that sensors have the average number of neighbors in simulation runs. In step 1 of the algorithm, a threshold value $\theta 1$ and $\theta 2$ are needed to determine the testing value. We set both $\theta 1$ and $\theta 2$ to be 15 for the simulation.

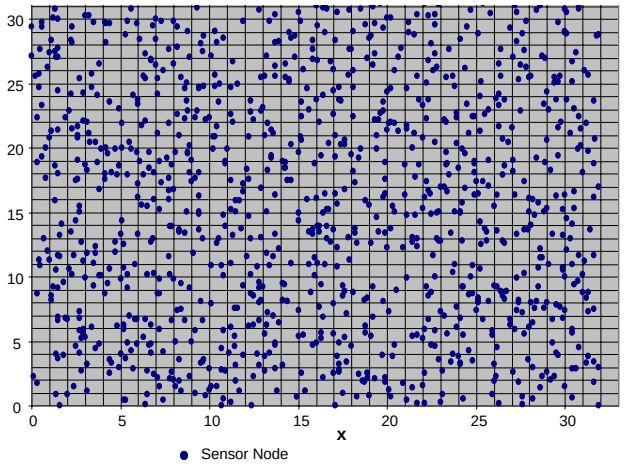

Figure 3: A 32×32 region with 1024 sensor nodes randomly deployed in it

Faulty sensor detection accuracy (FSDA) and false alarm rate (FAR) are the two metrics used to evaluate our algorithm performance. FSDA is defined as the ratio of the number faulty sensor detected to the total number of faulty sensors in the field. The FAR is the ratio of the number of non-faulty sensor diagnosed as faulty to the total number of non-faulty sensors.

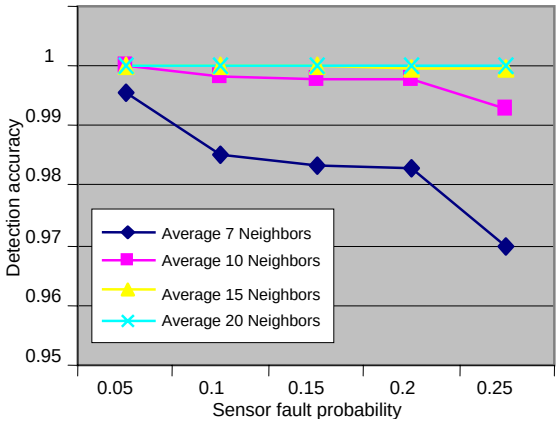

Figure 4: Faulty Sensor Detection Accuracy

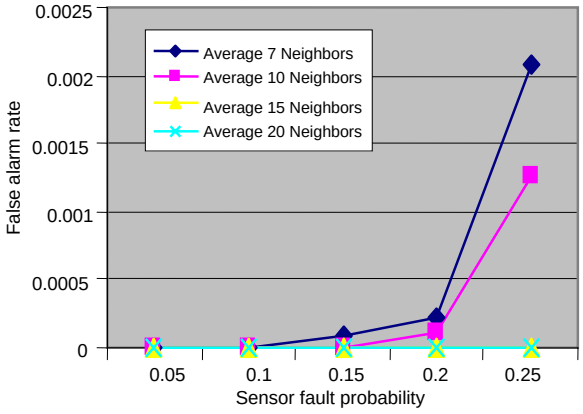

Figure 5: False Alarm Rate in Faulty Sensor Detection

In the simulation, sensors are randomly chosen to be faulty with the probabilities of 0.05, 0.10, 0.15, 0.20, and 0.25 respectively under different average number of neighbors for each sensor. Average number of neighbors/sensor is chosen to be 7, 10, 15, and 20 respectively.

Figures 4 and 5 show the faulty sensor detection accuracy and false alarm rate against the sensor fault probability for different average number of neighbors. In Figure 4, the detection accuracy for 7 neighbors and 10 neighbors decreases when the fault probability becomes larger. But the fault detection accuracy is still about 97% when there are about 25% of the sensors being faulty. There are several faulty sensors not being diagnosed as faulty because the randomly deployment of the sensors in the network results in very few neighbors for those sensors. When the average of number neighbors is greater than 15, the fault detection is very high and almost all the faulty sensors can be detected even under a high faulty sensor probability. This result is consistent with our probability analysis in Section 5.

In Figure 5, for the 7 neighbors and 10 neighbors, the higher the fault probability, the higher false alarm rate. This is be-

cause the large number of faulty sensor test good sensors to be Likely Faulty and these good sensors are then diagnosed as faulty sensors. For 15 and 20 neighbors, the false alarm rate is as low as 0. Again, this is consistent with our probability analysis.

Overall, our algorithm outperforms previous fault detection algorithm proposed in [8] in terms of the faulty sensor detection accuracy and false alarm rate. Our localized fault detection algorithm achieves high detection accuracy and low false alarm rate even with a large set of faulty sensors.

7. CONCLUSION

We proposed a distributed localized faulty sensor (DLFS) detection algorithm where each sensor identifies its own status to be either "good" or "faulty" and the claim is then supported or reverted by its neighbors as they also evaluate the node behavior. The proposed algorithm is analyzed using a probabilistic approach. In our probabilistic analysis, the probabilities of faulty sensors being diagnosed as "good" and good sensors not being diagnosed as "good" in the entire sensor network are very low.

Finally, the algorithm is tested using a simulation for an example case under different number of faulty sensors in the same area. Our simulation results show that the FSDA is over 97% even when 25% nodes are faulty. The FAR is very accurate when the sensor fault probability is low. Simulation results support and demonstrate that our proposed algorithm can have a high fault detection accuracy and low false alarm rate with a large number of faulty sensors existing in the network.

At this time there may be issues related to scalability and overhead due to exchange of information between neighbors. However, the aim of this work is to detect a faulty sensor as "faulty" in a distributed environment. The success of doing so is promising and we would like to extend it to see how it behaves in extremely large deployments. We are further working on developing an algorithm to find the event edge by partially using the algorithm proposed in this paper. Future work should include the implementation of the algorithms on NS2 sensor network simulators.

8. REFERENCES

[1] A. K. Somani and V. K. Agarwal. *Distributed Diagnosis Algorithms for Regular Interconnected Structures*. IEEE Transaction of Computers, Vol.41, No.7: 899-906, July 1992.

[2] A. Mahmood and E. J. McCluskey. *Concurrent error detection using watchdog processors-a survey*. IEEE Transactions on Computers, Vol.37, No.2: 160-174, Feb.1988.

[3] D. Blough, S. Sullivan, and G. Masson. *Fault diagnosis for sparsely interconnected multiporcessor systems*. In Proc. of FTCS-19, 1989, pp.62-69.

[4] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli. *Fault-Tolerance in Sensor Networks*. Handbook of Sensor Networks, I. Mahgoub and M. Ilyas (eds.), CRC press, Section VIII, no. 36, 2004.

[5] L. B. Ruiz, I.G.Siqueira, L. B. Oliveira, H. C. Wong, J. M. S. Nogueira, and A. A. F. Loureiro. *Fault management in event-driven wireless sensor networks*. MSWiM'04, October 4-6, 2004, Venezia, Italy.

[6] F. Koushanfar, M. Potkonjak, and A. Sangiovanni-Vincentelli. *On-line Fault Detection of Sensor Measurements*. Sensors, 2003. Proceedings of IEEE Volume 2, 22-24, Oct. 2003, pp.974-979.

[7] X. Luo, M. Dong, and Y. Huang. *On distributed fault-tolerant detection in wireless sensor networks*. IEEE Transactions on Computers, Vol.55, No.1: 58-70, Jan. 2006.

[8] M. Ding, D. Chen, K. Xing, and X. Cheng. *Localized fault-tolerant event boundary detection in sensor networks*. Proceedings of IEEE INFOCOM 2005, Miami, March 2005.