International Journal of Scientific Engineering and Science
Volume 2, Issue 11, pp. 56-64, 2018. ISSN (Online): 2456-7361

Wireless Sensor Networks Performance Measurement Approach

Nazar Elfadil Mohamed
College of Computing, Fahad Bin Sultan University, Saudi Arabia

***Abstract*—*The successful of Wireless Sensor Network application monitoring relies on the accuracy and reliability of its nodes operation. Unfortunately, operation deviations of these nodes are regular occurrences not isolated events as in traditional networks. This is due to their special characteristics that reduce network manufacturing and deployment costs and maintains the nodes immunity against internal and external conditions. The goal of this paper is to propose a real-time, distributed, passive, and low resources usage performance-monitoring algorithm that monitors Wireless Sensor Network functionality and isolates the detected deviated nodes from norm operation. Simulation and empirical experiments showed that the proposed algorithm has a slight processing and storage overhead. It is important to mention that these experiments showed that the proposed algorithm has a high reliability in tracking and isolating network nodes problems.*

I. INTRODUCTION

WIRELESS Sensor Networks (WSNs) are expected to be a new revolutionary technology in the manner of the internet due to its characteristics that allow them to have disposal, small size, and unattended maintenance-free nodes. These characteristics arise because the WSNs designers and manufacturers target a cheap system at a fixed performance not as in the traditional network to improve the performance and hold the price constant over the time. This strategy helps to reduce the overall network resources usage. On the other hand, these characteristics, along with the usage of wireless communication, the event-driven nature of the operating system, the harsh environment the nodes work in and the limited usage of fault-tolerant/diagnosis tools, reduce node immunity against internal and external interference, such as software bugs. This reduction increases the probability of deviating network nodes operation from their norm, reduces network overall functionality and degrades network collected data reliability even when the network protocols robustness increases such that it combats against worst-case scenarios. For example, in some practical deployments, such as [1]-[3], an analysis of network collected data showed a reduction in their quality and quantity to an amount of 49% and 55%, respectively. Nevertheless, these analyses showed that these reductions might cause in some cases a failure of the WSN monitoring.

These deviations occur because of two types of errors; i.e. systematic and transient [4]. The systematic type arises because of hardware faults; such as calibration, reduction in the operating power level, and change in the operating condition. It affects the operation continuously until the problem is solved. The transient arises due to temporary external/internal conditions, such as random environment effects, software bugs and channel interference. This type deviates the operation until the effect disappears.

These two deviation types affect the quality and the quantity of the collected data in network [5], [6]. They directly affect individual sensor node measurements and drift them by a constant value; i.e. biased error; change the different between sensor measurement and the actual value; i.e. drift error; and remain sensor measurements constant regardless of changes in the actual value; i.e. complete failure error. In addition, they directly affect network packets communication and drop them. The indirect effect happens when the deviations affect network collaboration function and reduce network performance in terms of the collected data reliability and increase the use of the network resources.

Tracking and detecting the above discussed deviations in WSN is not flexible as in traditional network because of the factors that affect the monitoring analysis and degrade its efficiency such as the:

- Limited finite energy and communication resources,
- Unavailability of a dominant protocol or algorithm that is suitable to work in all applications,
- Unavailability of global measurement variables due to the use of distributed control protocols that reduce the consumption of network resources,
- Network's adaptive nature to frequent changes in connectivity, link failure and node status,
- Imbalance in network traffic as a result of different data characteristics in terms of monitored phenomenon changes, reporting rate duration and the number of sinks in the network,
- Tolerance to data changes and losses as a result of redundancy,
- Collaboration functions used in different tasks to increase the accuracy of collected data and reduce the usage of node resources,
- Direct interaction of nodes with the environment that increases noise level and increases the probability of node failure,

The goal of this paper is to propose Voting Median Based Algorithm for approximate monitoring of Wireless Sensor Network (VMBA) that takes into consideration the above network challenges and reduces their impact on the proposed monitoring tool efficiency. The proposed algorithm works in real-time bases to reduce the required memory and in a distributed scheme so that it will be scalable for any size of network. Moreover, it passively extracts its metrics from network application low and high network levels by utilizing

the overhearing that exists in the neighborhood, as a result of the wireless communication medium. The proposed algorithm reduces its analyses complexity; memory storage and required learning time by calculating the collected parameters uncertainty depend on an interval that relies on the power dissipation model of the monitored phenomenon. This interval gives the imprecision of neighborhood nodes operation and any uncertainty in the monitored phenomenon.

The proposed VMBA algorithm performance was tested under different simulation and empirical experiments scenarios and was approve as uses a slight processing and storage overhead. This insures its easy implementation on the existing constraints resources platform. In addition, these experiments showed the high reliability of the proposed algorithm in tracking and isolating nodes problems in network and nodes levels. This comes with a high resilient to both high neighbors packet loss and random measurements deviations. Unfortunately, this also comes with some limitations due to the simple passive analysis method adapted in the algorithm.

The results outcome from these conducted experiments divided are into two papers. This paper 'part one' deals with our approach node power consumption and its impact on the network lifetime, the proposed algorithm spatial-temporary events tracking, the proposed algorithm detection of neighborhood reliability. Finally it discussed some of the proposed algorithm limitations detected in the experiments. The second part of the paper deals with the proposed algorithm detection at network and node levels, the proposed algorithm released warning messages and the impact of threshold value in algorithm detection and warning messages release.

The rest of the paper layout is as follows: Section two discusses related work; section then provides an explanation of the algorithm different module function. The fourth section discusses some of important experiment results of algorithm resources usage and their impact on network lifetime. Finally, the paper ends with a conclusion, limitations and suggestions for future work.

II. Related Work

Although the researchers' analysis of several real WSN deployments expected there to be an improvement of the deployed network's functionality of up to 51% if a real-time monitoring/measuring tool was used [1], the special characteristics of WSNs, discussed above, lessen the functionality of these tools, reduce their efficiency and increase their impact on network lifetime. As a result, the researchers proposed several methods, such as a tradeoff between the algorithm's detection accuracy, its response time and its resource usage, to reduce the impact of such tools on monitored network. This was achieved by selecting the algorithm's parameters, controlling the required packet exchange, and controlling the level of complexity of the analysis.

The researchers selected the algorithm's parameters so that the required extraction resources had a low impact on the network's lifetime. This was done by using common parameters that are available at low or high network levels and attributed them to network status. For example, at low network levels, they tracked packet losses between neighbor nodes and related them to network congestion, environmental effects, node battery depletion, and other hardware problems, as discussed in [7]. While at high network levels, they tracked the changes of node measurements and share them to sensor node software/hardware problems, as discussed in [4]. Moreover, some of them were extracted at particular parameters that would track certain goals designed in the network, such as power consumption [3], node coverage and connectivity [8].

Also, they control the rate of exchange of the algorithm's parameters in order to reduce the impact of monitoring tool on the network's lifetime. Some of them use techniques continuously to collect parameters and analyze them centrally, such as [3]; others distribute these analyses to reduce the number of exchanged packets and the reply time, as discussed in [9]. The third group uses predicted models that exchange the packets if there is a large discrepancy between the actual and the predicted values, such as in [10].

Finally, these researchers reduce the impact of the algorithm on the network's lifetime by controlling the complexity of the algorithm's analysis and the resources it uses. This is achieved by generating the residual of the monitored parameters in terms of physical or analytical redundancy [5], [6]. Physical redundancy generates an estimate of the actual value of a quantity based on the available redundant information; this is accomplished either by statistical methods (such as descriptive or inferential statistics), or by data fusion. The main advantage of this type of redundancy is that it is relatively easy to implement and provide a high degree of certainty; its reliability relies on the accuracy of the collected measurements. Analytical redundancy methods, on the other hand, provide values other than direct measurements from the parameters and variables of interest using a process model such as a Kalman filter, parity relations, Principal Component Analysis (PCA), and Artificial Neural Networks (ANN). These methods are not easy to implement and they depend on the reliability of the process model.

All the above methods for extracting, exchanging and analysis were gathered together to ensure the reliability of data collected in the network by using four main techniques: data cleaning, fault-tolerance, diagnosis, and performance measurement. Data cleaning techniques work at a high network level and consider reading impacts from a deviated sensor on multi-sensor aggregation/fusion such as in [11],[12]. Such research proposed several methods that isolate deviated readings by tracking or predicting correlation between neighbor nodes measurements. The proposed methods that adapt statistical methods in their analysis did not considered the impact of the packet loss which reduced its detection accuracy and the others that adapt the analytical method in their analysis used complex methods or models that need a high resource usage to detect and predict sensor measurements. In addition, these techniques rectify deviated data after detecting them without checking their cause and their impact on network functionality.

Fault-tolerance techniques are important in embedded networks where it is difficult to access them physically. The advantage of these techniques is their ability to address all network levels; such as circuit level, logical level, memory level, program level and system level. But due to WSNs scare recourses these techniques have a limited usage. In general WSNs fault-tolerant techniques detect faults in fusion and aggregation operation, network deployment and collaboration, coverage and connectivity, energy consumption, energy event fault tolerance, reporting rate, network detection, and many others [13]-[20]. Faults are detected using logical decision predicates computed in individual sensors, faulty node detection, or event region and event boundary detection. These methods detect metrics either at high or low network level without relating them to each other and without checking their impact on network functionality. The main problem of these techniques is the impact of deviation on network functionality and collected data accuracy before it is detected and isolated. This is because the techniques operate when there is an effect on functionality that the network cannot tolerate with it and need to readjust itself to solve it.

Diagnosis techniques use active or proactive monitoring to trace, visualize, simulate and debug historical network log files in real and non real time as discussed in [9]. These techniques are used to detect faults at high or low network levels after testing their cause. For example Nithya in [7], proposed a debugging system that debugs low network level statistical changes by drawing correlations between seemingly unrelated, distributed events and produces graphs that highlight those correlations. Most of these diagnosis techniques are complex and use iteration tests for their detection. This is because they assume a minimal cost associated with continuously transmitting of debug information to centralized or distributed monitor nodes and send/receive test packets to confirm the detection of a fault.

Finally, performance techniques are similar to diagnosis techniques but without iteration tests. Unfortunately, there is little literature and research on systematic measurement and monitoring in wireless sensor networks. Zhao in [3] studied the effect of low network level and impact on network stability and network processing. He studied the effect of the environment conditions, traffic load, network dynamic, collaboration behavior, and constraint recourse on packet delivery performance using empirical experiments and simulations. Although packet delivery is important in wireless communication and can predict network performance, it can give wrong indications of network performance level due to collaboration behavior, and measurement redundancy which makes a network able to tolerate a certain degree of changes. Also, Zhao proposed an energy map aggregation based approach that sends messages recording significant energy level drops to the sink. Although energy consumption is very important in WSNs and all network levels affected by it, several researchers such as [21], showed in their analysis, that there can be a sudden drop in node and network functionality which is not possible to detect by measuring voltage level. This sudden drop causes network instability due to sudden route change, and more energy consumption due to usage of non-optimal routes to the destination.

Although the above discussed methods and tools help in monitoring and measuring WSN functionality, their limits are more than expected, especially at large network scale. As consequence, this paper was motivated by such drawbacks and limitations. The main contribution will be to find a real-time monitoring tool that efficiently detects the degradation of WSN performance before it happens, isolates the detected deviated nodes and uses low network resources.

III. Algorithm Approach

The range of simulation/empirical experiments that were conducted, and different researches which were discussed in section II, demonstrates the goal of the proposed algorithm, the algorithm needs to be divided into four modules; namely: (A) listening and filtering, (B) data analysis and threshold tests, (C) decision confidence control, and (D) warning packet exchange. This is because the main resource usages that affect the lifetime of the monitored network and the detection confidence of the algorithm come from the:

- Algorithm's collected parameters and the method used to collect them from the monitored network;
- Complexity of the analysis method used;
- Uncertainty level of the algorithm's analysis;
- Algorithm's packet exchange.

As a result, controlling individually each of the factors listed above ensures a reduction in the effect of the proposed algorithm on the network's lifetime and increases the algorithm's detection confidence. The proposed algorithm designed such that it works in a distributed manner to ensure the scalability at any large-size network. It collects its metrics online from network application low and high network levels to ensure the reduction of the required overhead usage; such as memory and processing. VMBA algorithm relies in its analysis on a simple voting technique that calculates the median of network protocols parameters. It provides an operation interval for neighborhood nodes operation that depends on network designed goals to quantify the uncertainty in the monitored phenomenon and nodes operation. This simple analysis reduces the complexity of the algorithm and makes it without a need for learning time like most of the uncertainty models. In addition, the algorithm relates the two network levels parameters to test the impact of the change on collected data reliability and network functionality. This is done by checking the percentage of deviated nodes from neighborhood median operation and the level of their weighted residual.

A. Listening and Filtering Module

The first module is considered to be the most important module in the algorithm because other algorithm modules analysis depends on its collected parameters. The module integrated in network application flow process such that it reuses the same application protocols parameters, and their used memory. This reduce the VBMA algorithm required resources usage for metrics calculation, follow the application process such that it is not affected by used power save

techniques and there will not be a need for a special synchronization timer for its functionality.

The module starts its functionality by waiting for a period of time after application triggers a report of sensor. This time period is controlled by the period between the two measurements and/or the sleeping node period. In addition, it depends on the network's data delivery period, node functionality in the network collaboration function, and node location. Any un-received measurements within this assigned period are considered to be a loss. If more than one measurement packet is received from a neighbor during this period, the monitoring node will take the first at that time interval. After that, the module examines all neighborhood available measurements to determine whether they fall in the range of normal sensor operation (i.e. depend on sensor nodes characteristics and collaboration as discussed in [22]). Measurements that are out of this range filter and the counter of node deviation increment; as shown in the pseudo-code in Appendix B and its notation definitions in Appendix A. This measurements filtering prevents the node from wasting resources on obvious deviated measurements.

After that the module constructs neighbor readings tables; i.e. depend on the application requirements. Finally, the median of available measurements is calculated and the available measurements and calculated median send to algorithm module 2.

The proposed algorithm uses statistical method for its uncertainty analysis to estimate phenomenon readings because it is the lowest resources usage method compared to the fusion and the analytical methods. It uses median because it's robust to noise and error borne where it tolerate to small error in a large fraction of measurements, small calibration error or noise and tolerate large error in small fraction of measurements such as it tolerate with 50% error. Wagner in [23] showed that median error is , its resilience is if and its break point; i.e. the point at which the method calculation will be affected; is 50% of nodes; where n denote the total number of nodes, k denote the number of error nodes and the standard deviations of nodes measurements. In addition, the proposed algorithm uses divide-and-conquer method for measurement sort in order to reduce analysis growth of the function to [24].

B. Data Analysis and Threshold Test Module

The second module starts its function after receiving the calculated median value and time instant neighbors collected measurements from module 1. It calculates the difference between the last stored calculated neighborhood median and the new received calculated median to validate the accuracy of the new calculated median value. If the difference is larger than a threshold that represent the maximum expected change in the measured phenomenon within that period of monitoring time, the module increment neighborhood accuracy degradation counters and replaces the new calculated median with the stored median value; as shown in Appendix C. Else, the old stored median value replaced by the new value. (This is done to reduce the affect of the high neighbor packets losses on median calculation). After that, the module evaluates the measurements received from module 1 at that time instance by comparing their residual difference from the median with a threshold value that relies on the power dissipation model of the monitored phenomenon. If any of the calculated residual is greater than the threshold value, the module increment deviation detection counters of that node. Else, the algorithm is going to stop its function up to the new request from module 1.

The algorithm uses this threshold to accommodate the uncertainty in wireless sensor node operation. This threshold works as interval that contains the expected correct operation value and represents the imprecision of expected correct operation. This can be defined as where denoted the precision of used sensor and is the expected change of the measured phenomenon characteristic at the end of monitoring node sensing range. This required some knowledge of the monitored phenomenon, the specification of used sensor node, and network designed goals between neighbor nodes taken from the deployed network designed goals; as discussed in [25].

The used method allows the proposed algorithm to follow a straightforward approach that calculates any faulty deviations in sensor node operation. Which reduces the required analysis complexity, required memory storage and with it there will be no learning time interval that all prediction models have. This is because the approach assumes that true measurements of a phenomenon's characteristics, following a Gaussian pdf, centre on a calculated median of neighborhood readings with variation that controlled by the correlation between the neighbor nodes at the end of monitoring node sensing range (The assumption is based on the fact that the change within node sensor range is govern by deployed network designed goals and the imprecision of sensor node). Any external impact or monitored phenomenon characteristic change will affect all neighbors at the same time but to a different degree depending on its location from the nodes and the position of the nodes from each other. If this affect exceeds the designed goal between neighbor nodes the deployment goal fails and the network should reconfigure itself again to achieve it.

The second module also tests the effect of any loss on the reliability of the collected data by calculating the degree of distortion in the neighborhood data that has occurred because of its affect on the collected data accuracy and network functionality. This is done by calculating the ratio of the number of healthy readings to the total number readings as shown in Appendix C step 8.

C. Decision and Confidence Control Module

The third module of the proposed algorithm is concerned with a decision-making framework to analyze the detected changes in the operation of neighbor nodes and network functionality. This is done at the proposed algorithm by monitoring window method and collected data validity method. In monitoring window method the algorithm detection control by varying the period of monitoring window and its detection threshold; i.e. similar to what discussed in [26]. While in collected data validation method the algorithm detection controlled by changing the allowed range of sensed regions; i.e. similar to what discussed in [1]; degree of

neighbor measurement closeness or conflict, and redundancy of measurements. Both methods of configuration settings depend on the characteristics of network application and its tolerance to changes (as discussed in [6]); the required algorithm respond time and the required accuracy of collected data. The function of this module is shown in Appendix D. Else, the module initializes the counters.

D. Warning Packet Exchange Module

When module four receives a send request, it checks its neighbors warning exchange memory to ensure that none of the neighbor nodes have reported the same fault in that monitoring window period. If none of the neighbors has a report, it sends a message or it cancels the request. In addition, this module tests warning messages received from its neighbors with statistics from module three. If the suspected node flags up a counter indication smaller than a threshold, (i.e. below the 30% set for the experiments), a message will be released indicating 'NO_EVIDENCE_OF_FAULT'. On the other hand; if the threshold is higher or equal to the threshold, then the node cancels any similar warning message request from module three during that monitoring period. This is to ensure the reliability of the warning message detection and to correct any incorrect detection that may occur because of the loss or other network circumstances. Moreover, module four reduces the algorithm warning packets released by checking if any of its neighbors sent the same message at that time interval. If it has been sent, the algorithm will discard module three requests as shown in Appendix E part 3, and to reduce algorithm positive detection when the algorithm detect 'NO_EVIDENCE_OF_FAULT' messages more than one time at that time interval of the same reported fault, it will drop send warning message to the sink. This will save the multi-hop routing message consumption and reduce it to consumption of neighbor broad cast at neighborhood.

IV. Experimental Results

We used various empirical and simulation experiments scenarios to evaluate the multiple aspects of the proposed algorithm and to test its efficiency. Also, these experiments tested algorithm analysis resilient to network dynamic and the high packet losses, its adaptability and responsiveness to different network conditions and its scalability. At this stage, paper discusses some of the important results on algorithm resources usage, algorithm spatial-temporary events tracking and algorithm neighborhood reliability detection.

A. Algorithm Resources Usage and Power Consumption

Several simulation experiments conducted in MATLAB and PowerTossim [27] test the proposed algorithm resources usage in different random network deployment sizes with different scenarios of faults rate, packet loss, and random measurement deviations. The outcomes of these experiments evaluated based on two metrics. The first algorithm concern is measuring the impact of VMBA algorithm on node lifetime. This is approach used the average extra CPU power consumption metric, which is the difference between CPU power consumption of the TinyOS 'Surge' application with and without algorithm use under the same network and nodes arrangements. This metric estimates the increase in CPU computation when the algorithm used. The second metrics was the average node lifetime reduction, which is the average decrease of node lifetime for both algorithm best and worst case (that is Best when none of the nodes having faulty deviation and worst when 50% of readings above predefined algorithm threshold). This metric point out the proposed algorithm influence on node lifetime.

The conducted experiments on PowerTossim showed a little increase in node power consumption when the TinyOS 'Surge' application node programmed with the proposed algorithm. The experiments included neighbors ranging from 2 to 16 single hop communicate with the sink each of 50 meter transceiver range and a Mode 1 Mote duty cycle; i.e. 1% to 99% operation. This detected increase raised as a result of the extra CPU consumption; as shown in Figure 6; and the power used to release algorithm warning packets after detecting the fault; as shown in Figure 2. The PowerTossim simulations showed that the average CPU power consumption increase was in the rage of 4.4±1.5 mJ with a maximum 5.9 mJ (that is 0.8% added to CPU consumption). In addition, these simulations showed a varying of CPU power consumption along with the change of the number of neighbor nodes where it was at the maximum when the number of nodes at neighborhood was fewer than four nodes; as shown in Figure 1. This is because of the increase in node wakeup period as the number of neighbor's increases. This makes the consumed CPU power almost negligible when compared with other at the same period, such as processing and storing received packets. Moreover, this low consumption in CPU is due the start of algorithm process when only deviation from neighborhood estimated value detected.

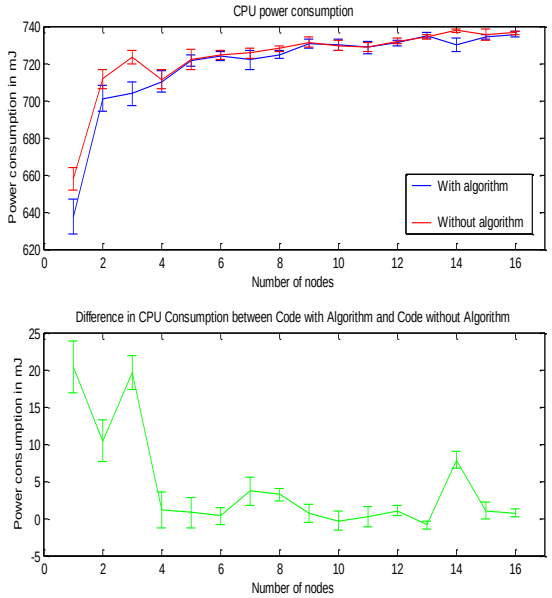

Fig. 1. CPU Power Consumption between Codes with and without the Algorithm in the 'Surge' Application

When the same PowerTossim power consumption model used to calculate the impact of the proposed algorithm released warning messages on network lifetime in MATLAB

International Journal of Scientific Engineering and Science
Volume 2, Issue 11, pp. 56-64, 2018. ISSN (Online): 2456-7361

code based experiments of the same network scenarios, the experiments results showed that at the worst case (that is when the number of faulty node is 50% and all warning messages reached to algorithm stop sending condition), the maximum impact of the proposed algorithm affects the original expected network lifetime before algorithm implementation by 0.03 days (out off 224 days); which is equal to 0.01% of network lifetime as shown in Figure 2.

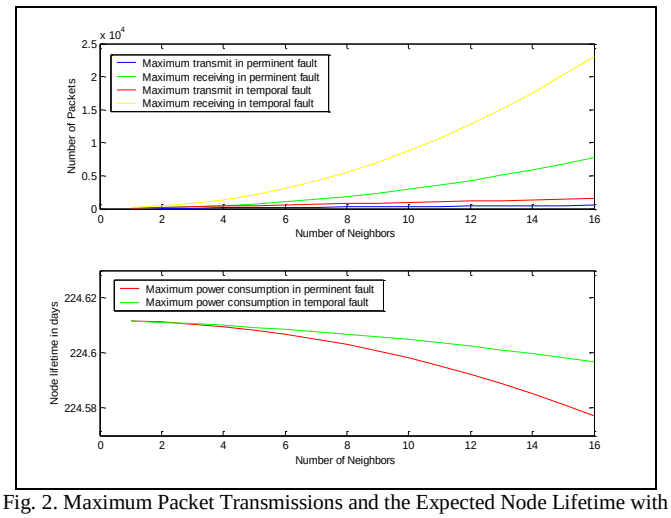

Fig. 2. Maximum Packet Transmissions and the Expected Node Lifetime with Temporary and Permanent Neighbor Faults

B. *Algorithm Spatial-temporal Events Tracking*

To evaluate the performance of algorithm spatial-temporal events analysis and tracking, and test the effect of removing the confirmed deviated node on the accuracy algorithm analysis, real world data sets were used; i.e. Intel Lab data set [28]. The data set consist of temperature, humidity and light intensity measurements, and was collected by 54 sensor nodes deployed in the Intel lab from February 28th until April 5th 2004 for 720 hours with a scheduled communication approach used with a waking period of four seconds and a 13% duty cycle. The data set had a lot of missing data, noise, and failed sensors, especially when the battery levels were low at the end of the experiment. This is due to the deployment goal of this experiment which was concerned to test the behavior of a sensor network with different conditions of battery power depletion, traffic generation, and multi-hop aspects.

Four metrics were chosen to analyze the results of the experiments. The first metric is the residual value of deviated neighbors, which is the difference between the neighborhood median and the data at a given time instance. This metric computes the diversity level of individual readings from other neighborhood nodes. In addition, it shows the behavior of the fault. The second metric was the weight residual metrics: i.e. the difference between a reading and the median calculated at a time instance multiplied by the ratio of similar correlated readings compared to the total number of readings at the same time instance. This metric shows the weight of each deviation on the neighborhood node collected data. The third metric was network performance, which is the ratio of healthy readings as opposed to the total number of nodes in the neighborhood. This metric computes the effect of the losses on the network's functionality.

Figure 3 shows the functionality of the network when the Intel Lab experiment data sets employed MATLAB as a tool in simulating the proposed algorithm functionality on node 1 (as monitoring node), without removing suspected nodes. The figure shows fluctuation in the accuracy of the collected data accuracy and the network's performance as a result of the residual impact of deviated data on the neighborhood data accuracy, as shown in Figures 3.1 and 3.4. This fluctuation continues up to the time where it becomes very heavy due to the effect of losses and the preponderance of unhealthy readings that become the majority; as shown in Figure 3.1. Afterwards, this heavy fluctuation becomes constant when the number of permanently deviated nodes is greater than the healthy nodes at the end of the experiment (i.e. from event 680000 upwards). Normally, this heavy fluctuation does not occur in WSNs due to the redundancy that schedules the function of nodes, which makes the probability of failure occurring at the same time low. Even if this happens, it can be detected by the dramatic change in data accuracy and the increase in the weighted residual which moves up to a constant level for a long period.

The figure also shows that even there is heavy fluctuation in neighbor readings accuracy due to losses. The calculated median that the algorithm depends on in its analysis does not deviate from the correct phenomenon value until permanently faulty nodes become the majority (as shown in Figure 3.3). This is due to fact that, the algorithm beside its median calculation to time interval neighborhood measurements compares the difference between the new calculated median and the old stored median values with the application's permitted degree of change that depend on the measured phenomenon characteristics. This is used as a filter to remove median values. That is value obviously deviate from the normal as a result of the loss impact of neighborhood packet.

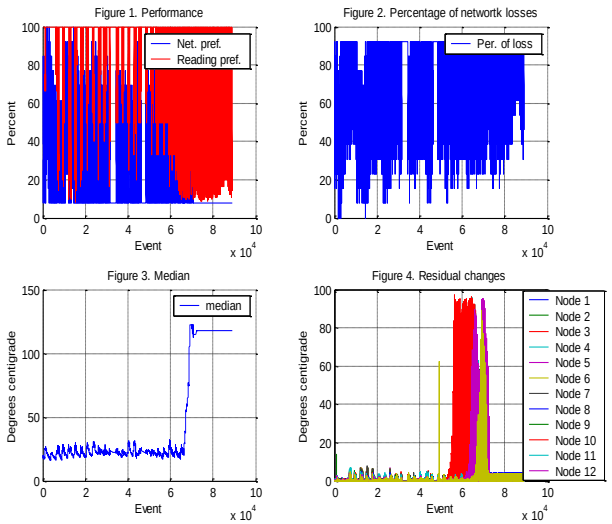

Fig. 3. Simulation of VBMA function at Intel data set without Isolation of Faulty Nodes

International Journal of Scientific Engineering and Science
Volume 2, Issue 11, pp. 56-64, 2018. ISSN (Online): 2456-7361

Figure 4 illustrates the proposed algorithm's calculation of the weighted residual for individual node measurements with respect to the neighborhood median. If this figure is compared with Figure 3, almost the same changes in detection can be seen but with different residual values; these depend on the number of deviated readings at each time interval.

If the algorithm is allowed to isolate faulty deviated nodes in a specified monitoring window, the neighborhood's performance is improved but with any new deviated node making a higher impact, as shown in Figure 5. This happens because of the increase in the impact of the residual on the collected data as a result of reducing the number of data samples at each time interval. On the other hand, not removing the deviated reading affects the accuracy of the collected data for the period it occurs, as shown in Figure 3.

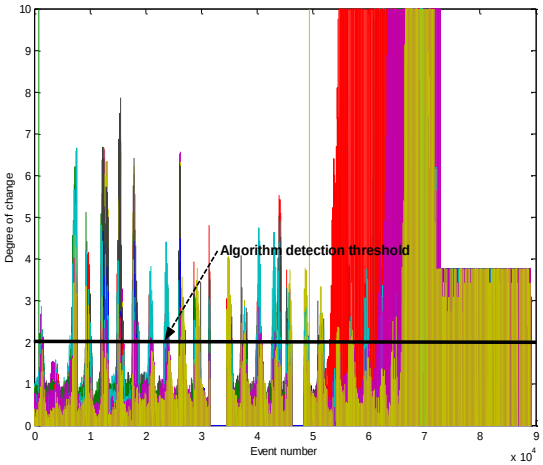

Fig. 4. VMBA Algorithm Detection with Weighted Residual Changes

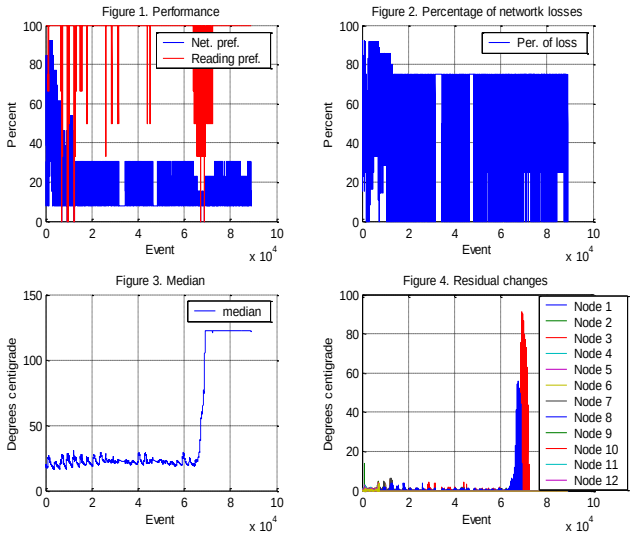

Fig. 5. Simulation of VBMA function at Intel data set without Isolation of Faulty Nodes

The above experiments were repeated so that the analysis could be carried out on Node 2 acting as the monitoring node. Table 1 shows the detection interval and the number of messages sent by Nodes 1 and 2 before isolating faulty nodes. As can be seen from the table, some of the nodes detected faulty nodes at the same event while others detected them at different times. The table shows that more algorithm warning messages were sent by Node 2. This was as a result of the different neighbor packet losses each node faced.

TABLE 1. Event number of removed detected faulty nodes

	1	3	4	33	35	37	39
Node1	--	**66240**	**60960**	**15840**	**53760**	**62880**	**66720**
Times	--	**17**	**15**	**15**	**37**	**20**	**18**
Node2	--	**66240**	**60960**	**18240**	**24000**	**24000**	**66720**
Times	--	**13**	**6**	**29**	**21**	**16**	**24**

C. Algorithm Detection

Since there is no ground truth for the measured phenomenon, statistical methods were used to check the algorithm's detection of the location of faults. This was done by using the Box-Whisker method [29] (i.e. a box plot) which quantifies changes in the measurements of neighbor sensor nodes. With this method, the box represents the middle of the data while the median is the line around it at a range known as the inter quartile range. The analysis shows that 97% of the faults detected by the proposed algorithm lie within the same outlier regions as those detected by the Box-Whisker method. The algorithm detected 108133 changes of value for all nodes, and conformed 83891 to be faulty deviations for a data set with 65% loss. Other 3% were measurements deviations which happened at the same time instance and have the same residual weight.

D. Algorithm Detection of Neighbourhood Reliability

The proposed algorithm combine in its analysis the two network levels parameters to analyze events not only gives the impact of the two levels on network performance and data reliability but increase the confidence of the algorithm detection. It detects neighborhood malfunctions by testing the affect of losses on the calculated neighborhood median, as shown in Figure 6 (for node 1 analysis neighborhood temperature measurements of Intel data set [28]). When a change detected between two consecutive median calculations that is larger than the expected change in the phenomenon, the algorithm detects neighborhood's functionality malfunctions. This change causes communication and application protocols in the network to recalculate their tables often and change data gathering points that change data collection accuracy, and communication paths. Such frequent changes cause instability in the network. If this instability detected for a specified period, the algorithm sends a message to point out a neighborhood problem. In addition, the relation between high and low network levels predicts the impact on neighborhood Intel data set collected data accuracy as shown in Figure 7. The figure shows the simulation of the same data set and the impact of the losses on degrading neighborhood collected data reliability.

E. Impact of Packet Losses on Algorithm Detection

Figure 8 illustrates the effect of increasing the percentage of packet losses on the proposed algorithm's detection with 20% of faulty deviated nodes. (The experiments for this data set employed MATLAB as a tool in simulation scenarios using 1000 sensor nodes randomly distributed over 1000

International Journal of Scientific Engineering and Science
Volume 2, Issue 11, pp. 56-64, 2018. ISSN (Online): 2456-7361

square meters, each of them with a 50 meter transceiver range). The figure shows that, as losses increased, the algorithm's detection decreased linearly and reached 40% at 60% loss. This occurred along with a gradual increase in the algorithm's positive detection which reached 20% with 60% losses. If a monitoring window of a sample size of 10 and a 6 sample threshold was used, the positive detection would be reduced to almost 0%.

F. *Detection of Deviated Faulty Nodes*

Figure 9 illustrates the effect of the increased percentages of faulty deviated nodes on the proposed algorithm's detection per event in 100 runs with 1,000 nodes had a 50 meter transceiver range; these are randomly deployed over the 1,000 square meters with 0.1% packet losses, 0.1% deviated node measurements, and 0.1% dead nodes. The figure shows that, as the percentage of deviated faulty nodes increases the proposed algorithm's detection of faulty nodes decreases exponentially and reaches around 20% when faulty deviated nodes reach a level of 80%. The algorithm's positive false detection increases linearly as the number of deviated faulty nodes increases, reaching around 80% when 80% of the network's nodes are faulty.

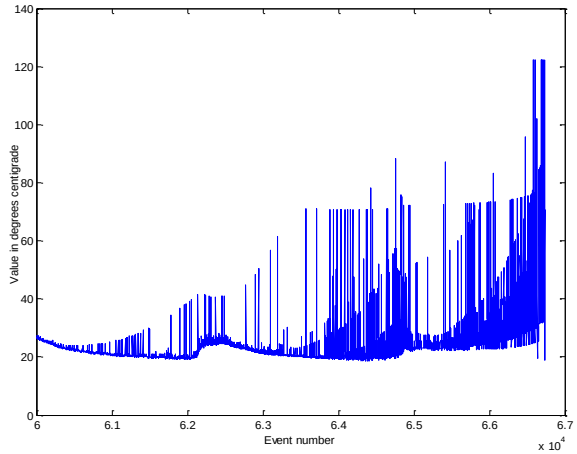

Fig. 6. Neighborhood Calculated Median Value Between Events 60000 and 67000 at Intel Lab Data Set.

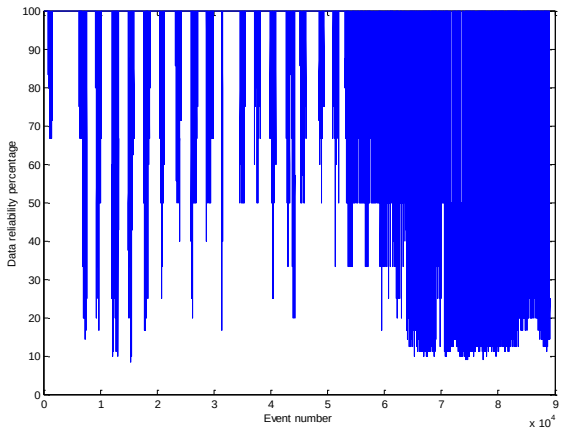

Fig. 7. Neighborhood Collected data reliability at Intel Lab data set experiment

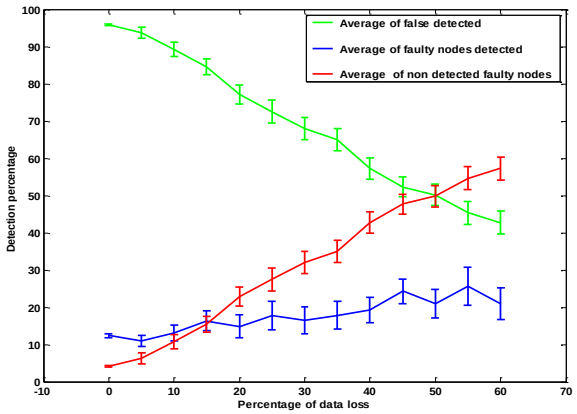

Fig. 8. The Algorithm Detection of Faulty Deviated Nodes with different Data Loss Percentages, 0.1% Packet Loss and 0.1% Dead Nodes.

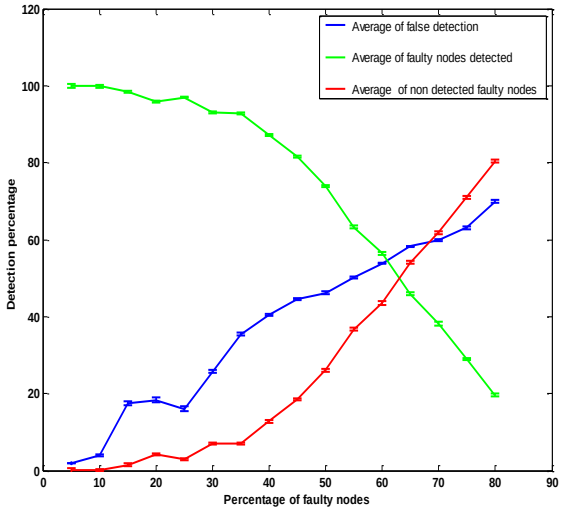

Fig. 9. The Algorithm Detection versus the Percentage of Faulty Nodes.

V. LIMITATIONS OF THE ALGORITHM

The proposed algorithm uses passive voting analysis and assumes that readings different from the majority constitute a change due to either a fault in a sensor node, the depletion of a sensor battery, a calibration problem, or a coverage problem. Because of this simple method, the algorithm functionality and its detection confidence relies on the number of neighbors, the number of deviated neighbor readings at each time interval, loss percentages and the degree of measurement deviation from the others.

The number of neighbor nodes is very important since it is concerned with the confidence of algorithm detection. The conducted experiments showed that as the number of received measurements from neighborhood increased, the algorithm detection error percentage decrease.

Moreover, these experiments showed that detection error value depends on the ratio of un-healthy neighbors to the total number of neighbors. If this ratio is more than 50% then the algorithm calculated median may drift from the real neighborhood phenomenon measurements. This drift is sensitive to packet losses spatially if the losses make the faulty deviated nodes the majority at that monitoring time interval.

This drawback was solved by adding historical readings from the last medium calculation and comparing this with the new median and with a threshold of allowable phenomenon changes as discussed in section III-B.

VI. Conclusion and Fiture Work

The algorithm proposed in this paper enables each sensor node in a sensor network to detect the Wireless Sensor Network's performance in a distributed manner. It sends a warning packet to the sink reporting any detection of degradation in nodes or neighborhood. This is done by tracking the changes in the status of the nodes and compares them with an estimated norm value of the neighborhood.

Simulation experiments showed that 97% of faults detected by the Box-Whisker method were detected by the algorithm. These experiments showed a similar level of detection of deviations by neighbor nodes that used the algorithm, with slight changes in detection time due to losses that each node faced in receiving its neighbors' measurements.

Although the experiments showed a good level of detection for the proposed algorithm and limited impact on network lifetime, there are some limitations due to the simple adapted method that may degrade the algorithm performance especially at small size networks or very high packet loss. Numerous aspects can be considered in the future in order to extend this work and improve the algorithm's functionality, such as checking the impact of the mobility of sensor nodes on the algorithm's functionality. Also, we are planning to use the influence diagram to replace the existence window and data validity tests and check the algorithm's time response and the level of deviation affect on the network functionality.

References

[1] N. Ramanathan, T. Schoellhammer, D. Estrin, M. Hansen, T. Harmon, E. Kohler, and M. Srivastava, "The final frontier: Embedding networked sensors in the soil," CENS Technical Report #68, Center for Embedded Networked Sensing, UCLA, USA, November 2006.

[2] G. Tolle, J. Polastre, R. Szewczyk, D. Culler, N. Turner, K. Tu, S. Burgess, T. Dawson, P. Buonadonna, D. Gay, and W. Hong, "A macroscope in the redwoods," *ACM Conference on Embedded Networked Sensor Systems (SenSys'05)*, 2005, pp. 51-63.

[3] Zhao Yonggang, "Measurement and monitoring in wireless sensor networks," PhD Thesis, Computer Science Department, University of Southern California, USA, June. 2004.

[4] E. Eiman and N. Badri, "Cleaning and querying noisy sensors," in *The First ACM Conference on Embedded Networked Sensor Systems (SenSys'03)*, pp. 78-87, 2003.

[5] Nasir Mehranbod, "A probabilistic approach for sensor fault detection and identification," Doctor Thesis, Drexel University, November 2002.

[6] Laura K. Balzano, "Addressing fault and calibration in wireless sensor networks," Master Thesis, University of California, Los Angeles, 2007.

[7] N. Ramanathan, K. Chang, R. Kapur, L. Girod, E. Kohler, and D. Estrin, "Sympathy for the sensor network debugger," in *The 3rd ACM Conf. Embedded Networked Sensor Systems (SenSys 2005)*, pp. 255-267, 2005.

[8] C. M. Vuran, B. O. Akan, and F. I. Akyildiz, "Spatio-Temporal correlation: Theory and applications for wireless sensor networks," *Computer Networks Journal (Elsevier)*, vol. 45, pp. 245-261, June 2004.

[9] C. Jaikaeo, C. Srisathapornphat, and C. Shen, "Diagnosis of sensor networks," in *IEEE International Conference Communications, ICC 2001*, pp. 1627-1632, 2001.

[10] A. Raquel Mini, A. Antonio Loureiro, and N. Badri, "Prediction-based energy map for wireless sensor networks," in Simpsio Brasilerio De Redes De Computadores, pp. 165-169, 2003.

[11] W. Yao-jung , M. Alice Agogine and G. Kai, "Fuzzy validation and fusion for wireless sensor networks," in *ASME International Mechanical Engineering Congress and RD&D Expo (IMECE2004)*, Anaheim, California, USA, 2004.

[12] Caimu Tang and Cauligi S. Raghavendra, "Correlation analysis and applications in wireless microsensor networks," in *Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS 2004)*, pp. 184-193, 2004.

[13] G. Indranil, V. Robbert Renesse, and P. Kenneth Birman, "Scalable fault-tolerant aggregation in large process groups," in *The 2001 International Conference on Dependable Systems and Networks*, pp. 433-442, 2001.

[14] T. Clouqueur, K. K. Saluja, and P. Ramanathan, "Fault tolerance in collaborative sensor networks for target detection," *IEEE Transactions on Computers*, vol. 53, issue 3, pp. 320-333, March 2004.

[15] C. M. Vuran, B. O. Akan and F. I. Akyildiz, "Spatio-Temporal correlation: Theory and applications for wireless sensor networks," *Computer Networks Journal (Elsevier)*, vol. 45, pp. 245-261, June. 2004.

[16] H. Song and C. Edward, "Continuous residual energy monitoring in wireless sensor networks," in *International Symposium on Parallel and Distributed Processing and Applications (ISPA 2004)*, pp. 169-177, 2004.

[17] M. Ding, D. Chen, K. Xing, and X. Cheng, "Localized fault-tolerant event boundary detection in sensor networks," in *IEEE INFOCOM 2005*, pp. 902-913, 2005.

[18] K. Bhaskar and S. S. Iyengar, "Distributes bayesian algorithms for fult-tolerant event region detection in wireless sensor networks," *IEEE Transaction on Computers*, vol. 53, pp. 421-250, March 2004.

[19] F. Koushanfar, M. Potkonjak and A. Sangiovanni-Vincentelli, "On-line fault detection of sensor measurements," in *Sensors, 2003. Proceedings of IEEE*, pp. 974-979, 2003.

[20] X. Luo, M. Dong, and Y. Huang, "On distributed fault-tolerant detection in wireless sensor networks," *IEEE Transactions on Computers*, vol. 55, pp. 58-70, June. 2006.

[21] R. Szewczyk, A. Mainwaring, J. Polastre, J. Anderson, and D. Culler, "An analysis of a large scale habitat monitoring application," in *SenSys '04: Proceedings of the 2nd International Conference on Embedded Networked Sensor Systems*, ACM Press, pp. 214-226, 2004.

[22] Ricardo Gutierrez-Osuna, Wright state university, "Sensor Characteristics," March 2007. http://courses.cs.tamu.edu/rgutier/ceg499_s02/l2.pdf.

[23] David Wagner, "Resilient aggregation in sensor networks," in *SASNA'04: Proceedings of the Security of ad hoc and Sensor Networks*, ACM press, 2004, pp. 78-87.

[24] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein, *Introduction to Algorithms*, Second Edition, MIT Press and McGraw-Hill, pp. 5-90, 2004.

[25] Caimu Tang and Cauligi S. Raghavendra, "Correlation analysis and applications in wireless microsensor networks," in *Mobile and Ubiquitous Systems: Networking and Services (MOBIQUITOUS 2004)*, pp. 184-193, 2004.

[26] M. Hempstead, V. Shnayder, and B. rong Chen, "Power TOSSIM: efficient power simulation for TinyOS applications," Report No. CS263, March 2007, http://www.eecs.harvard.edu/~shnayder/ptossim/.

[27] Intel Lab "Intel Lab Experiment Data Set," March 2007, http://berkeley.intel-research.net/labdata/.

[28] David Hand, Heikki Mannila, Padhraic Smyth, *Data Mining*, The MIT Press Cambridge, England, pp. 546, 2001.