Research Journal of Applied Sciences, Engineering and Technology 4(23): 5097-5107, 2012
ISSN: 2040-7467

Submitted: April 03, 2012 Accepted: April 26, 2012 Published: December 01, 2012

Fault Management for Efficient Data Gathering in Wireless Sensor Networks

M.Y. Mohamed Yacoab and V. Sundaram
Research Scholar, Karpagam University, Coimbatore, India, Asst. Professor, MEASI Institute of Information Technology, 87 Peters Road, Royapettah, Chennai 600014, India
Director, Department of MCA, Karpagam College of Engineering, Coimbatore, India

Abstract: Wireless Sensor Networks (WSNs) are naturally fault-prone owing to the common wireless communication medium, severe developed environments and resources limitation. In data gathering, node and network failures are common in WSNs. It is also essential for the WSN to be able to detect faults early and establish revival actions in order to avoid ruin of service due to faults. In this study we propose a fault management scheme which can efficiently gather data in wireless sensor networks. Our fault management scheme is capable of detecting network faults and node faults along with fault recovery. Initially, we assign some nodes as Reliable nodes (R nodes) in the data aggregation tree, to perform accurate fault discovery and recovery. These R nodes collects the details of residual battery power and signal strength of all intermediate nodes. Node faults are detected by comparing the data values of each node with its neighbor and link failure are detected by estimating the Signal to Noise Ratio (SNR) and Link Quality Indicator (LQI). In case of any link failure in the network, the succeeding R node will send a failure warning message to the previous R node and will then try to forward the packet to the next R node via an alternate path. By simulation results, we show that our proposed technique achieves good packet delivery ratio with reduced energy consumption and delay.

Keywords: Data gathering, fault tolerance, Link Quality Indicator (LQI), Signal to Noise Ratio (SNR)

INTRODUCTION

Wireless sensor networks are promising applications of pervasive computing, consisting of many small, low power and intellectual sensor nodes with one or more base stations. The base station acts as a gateway between sensor nodes and the end user. With recent intensive research in this area, wireless sensor networks have been applied in various areas, such as environment and habitat monitoring, eco-physiology, condition based equipment maintenance, disaster management and emergency response. Sensor nodes work under strict resource constraints such as limited battery power, computing power, memory, wireless bandwidth and communication capability, while the base station has more computational, energy and communication resources. Due to the low cost and the deployment of a large number of sensor nodes in an uncontrolled or even severe environment, it is common for the sensor nodes to become faulty and unreliable. The networks must exclude the faulty sensors to ensure the network quality of service. For many applications, the network is deployed in a harsh environment and some of the nodes may be faulty or may fail during the network's lifetime, thus requiring collaboration to be robust to node failures. Two other constraints in wireless networks of autonomous nodes come from the limited bandwidth and power source of these elements, requiring collaboration to be communication and power efficient (Winnie *et al*., 2007; Thomas *et al*., 2004; Jinran *et al*., 2006).

Need for fault management: Fault tolerance is the ability to retain sensor networks functionalities without any disturbance due to sensor nodes failure. Fault is an incorrect state of hardware or a program as an outcome of a failure of a component. Various faults associated with wireless sensor network are required to be understood in order to design an efficient fault management design for wireless sensor network (Shuo *et al*., 2009). Fault management has been generally considered as a key part of present network management. Recent rapid growth of attention in WSNs has further strengthened the importance of fault management, or in particular, played a crucial role. Faults in WSNs are not exception and tend to occur more frequently. In addition to typical network faults, WSNs have to deal with faults arising out of unreliable hardware, limited energy, connectivity interruption, environmental variation and so on. Thus, in order to guarantee the network quality of service and performance, it is vital for WSNs to be able to detect failures and to perform something analogous to heal and recover the network from events that might cause faults or misbehavior (Asim *et al*., 2010).

Corresponding Author: M.Y. Mohamed Yacoab, Research Scholar, Karpagam University, Coimbatore, India, Asst. Professor, MEASI Institute of Information Technology, 87 Peters Road, Royapettah, Chennai 600014, India

Failed nodes may decrease the Quality of Service (QoS) of the entire WSN. It is important and essential to study the fault detection methods for nodes in WSNs for the following reasons (Peng, 2009):

- Enormous low cost sensor nodes are often deployed in uncontrollable and hostile environments. Therefore, failure in sensor nodes can occur more easily than in other systems.
- The applications of WSNs are being widened. WSNs are also deployed in some circumstances such as monitoring of nuclear reactor, where high security is required. Hence fault detection for sensor nodes in this particular application is of great importance.
- It is hard and not practical to manually examine whether the nodes are functioning normally.
- As failed nodes would generate erroneous data, correct information cannot be obtained by the control center. Moreover, in serious cases it may result in collapse of the whole network.
- Nodes are commonly battery powered and the energy is restricted, so it is common for faults to occur due to battery depletion.

Sources of faults in WSNs: The design of sensor network is influenced by various faults, but there is no standard description of these faults. Some of the factors that cause fault in wireless sensor networks are as follows:

- **Function faults:** These faults typically results in the collision of individual nodes, packet loss, routing failure or network partition. This type of problem has been extensively studied and addressed by either distributed approaches through neighbor coordination or centralized approaches through status updates.
- **Data faults:** In data faults, a node acts normally in all aspects except for its sensing results, leading to either significant biased or random errors. Several types of data faults exist in wireless sensor networks. Although constant biased errors can be compensated by after deployment calibration methods, random and irregular biased errors cannot be rectified by a simple calibration function (Asim *et al*., 2010).
- **Power faults:** Due to the depletion of batteries or destruction by an external event sensor nodes are breakable and they may not succeed. In addition, because of the environmental influence on their sensing components, nodes may capture and communicate incorrect readings.
- **Link faults:** In ad hoc wireless networks, links are failure-prone, causing network partitions and dynamic changes in network topology. Links may break down when they are blocked by an external object or environmental condition permanently or temporarily. Packets may be corrupted due to the flawed nature of communication. In addition, when nodes are fixed or carried by mobile objects, they can be taken out of the range of communication.
- **Multi hop faults:** All of the above discussed fault scenarios are worsened by the multihop communication nature of sensor networks. It often takes several hops to carry data from a node to the sink. Consequently, failure of a single node or link may lead to missing reports from the entire region of the sensor network. Furthermore, congestion that starts in one local area can spread all the way to the sink and affect data delivery from other regions of the network (Lilia and Qi, 2007).
- **Natural environments:** Sensor nodes are generally used to monitor external environment, due to which sensor nodes are vulnerable to natural phenomenon like rain, fire and fall of trees (Asim *et al*., 2008).

Types of faults: Wireless sensor networks are commonly deployed in severe environment and are subject to faults in several layers of the system. Based on factors affecting sensor networks we can categorize faults into the following categories, (Luciana *et al*., 2007; Muhammad *et al*., 2009):

- **Node faults:** Nodes have several hardware and software components that can produce malfunctions. For instance, the area can suffer impacts and expose the hardware of the sensor node to the extreme conditions of the environment. The sensor readings may become incorrect, when the battery of a node reaches a certain stage. The data aggregation results can suffer deviations from the real value when a node generates incorrect data.
- **Network faults:** Routing is one of the fundamental building blocks in a WSN. Faults on the routing layer can lead to dropped or misguided messages or unacceptable delays. Radio interference can also cause the link between nodes to become faulty.
- **Sink faults:** On a higher level of the network a device (sink) that collects all the data generated in the network and broadcasts it to the back end system is also subject to faults of its components. When this device fails, unless fault tolerant measurements are present, a massive failure of the network happens given that the data from the sensor nodes cannot be accessed.
- **Faults caused by adversaries:** Since WSNs are often deployed for crucial application, attacks by adversaries may cause node faults and consequently leading the network to failure. The need of infrastructure and broadcast nature of wireless medium enable adversaries to intrude into the

network and interrupt the whole functionality (e.g., routing, aggregation etc.) of an individual sensor node.

Fault management and its classification: In order to tackle faults in a WSN, the system should follow two main steps.

Fault detection: The first step is to detect that a specific functionality is faulty and to predict that it will persist to function properly in the near future.

Fault recovery: After the system detects a fault, fault recovery is the second step to enable the system to recover from the faults. Basically, there are two types of detection techniques: self-diagnosis and cooperative diagnosis. Some faults that can be determined by a sensor node itself can adopt self-diagnosis detection. For instance, faults caused by depletion of battery can be detected by a sensor node itself. The remaining battery of the sensor node can be predicted by measuring current battery voltage. Some kinds of faults require cooperative diagnosis among a set of sensor nodes. A large portion of faults in WSNs are in this category (Hai *et al*., 2009).

The proper implementation of a fault management can maintain the network running at an optimum level and minimize the risk of failure and thus making the networks more fault tolerant. Some important functions of a fault management includes:

- Constant monitoring of system status and usage level
- General diagnostics
- Tracing the location of potential and actual failure
- Auto recovery and self-healing in the event of failure

Fault management in WSNs can be classified according to its network management system architecture such as centralized, distributed and hierarchical (Muhammad *et al*., 2009):

Centralized architecture: Base station or the central manager has rich and unlimited resources. Therefore, it performs complex management tasks and controls the whole network.

- **Distributed architecture:** The distributed architecture employs multiple manager stations throughout the whole network instead of having a single central controller. In order to perform management functions, each manager controls a sub-region of the network and may communicate directly with other manager stations in a co-operative manner.
- **Hierarchical architecture:** It is a hybrid between centralized and distributed architectures. The managers are distributed throughout the network in a tree shape hierarchical manner, having lower and higher level of hierarchy. These managers are referred as the intermediate managers which manage a sub-section of a network and perform the management functions. But they do not communicate directly with each other.

Data aggregation in WSNs: Data aggregation is a technique used in WSNs to reduce the amount of messages transported. By aggregating, the data contained in several messages is fused into one single message. When such a message, containing the equivalent of many individual messages, is lost due to transmission errors then this has a harmful effect on the application quality experienced. In many sensor network applications, the quality is severely affected by excessive data loss due to insufficient supply of data. Using data aggregation, several messages transported along the same path can be combined into a single message. Aggregation techniques lower the amount of messages and thus reducing the energy expensive transceiver operation which helps to preserve scarce bandwidth. The data reliability at the sink is altered as aggregation increases the amount of data concentrated in a single message, (Jonathan *et al*., 2007).

There are many types of aggregation techniques, some of them are listed below (Nandini *et al*., 2010).

- **Centralized approach:** This is an address centric approach where each node sends data to a central node using a multihop wireless protocol, via the shortest possible route. The sensor nodes just send the data packets to a leader, which is the powerful node. The leader aggregates the data which can be queried. Each intermediate node has to send the data packets addressed to leader from the child nodes. Hence a large number of messages have to be transmitted for a query in the best case equal to the sum of external path lengths for each node.
- **In-network aggregation:** In-network aggregation is the global process of gathering and routing information through a multi hop network. Also it processes the data at intermediate nodes with the intention of reducing resource consumption, in that way increasing network lifetime. There are two approaches for in-network aggregation specifically with size reduction and without size reduction. In-network aggregation with size reduction refers to the process of combining and compressing the data packets received by a node from its neighbors consecutively reducing the packet length to be transmitted or forwarded towards sink. In-network aggregation without size reduction refers to the process merging data packets received from diverse neighbors in to a single data packet but without processing the value of data.

- **Tree based approach:** The tree based approach aggregation is performed by constructing an aggregation tree, which could be a minimum spanning tree, rooted at sink with source nodes considered as leaves. In order to forward data each node has a parent node and the flow of data start from leaves nodes up to the sink, where the aggregation is done by parent nodes.
- **Cluster based approach:** In cluster based approach, the entire network is divided in to several clusters. Each cluster has a cluster head which is selected among cluster members. Cluster heads do the role of aggregator which aggregate data received from cluster members locally and then transmit the result to sink.

Proposed solution: In our earlier work (Mohamed and Sundaram, 2011), we have developed a multiple sink based data aggregation technique, assuming the sinks are static. In this technique, initially a sink oriented tree is determined for each sink. If the amount of data in the network becomes large, the data is transmitted in the slots allocated for the specific part of the data such that interference is avoided in the data transmission. As data gets aggregated at the nodes which are nearer to the sink it will be compressed and then forwarded to the next level. This way data is efficiently transmitted to the sink without any loss and interference.

Though data aggregation is performed efficiently with the help of multiple sinks, there are chances of faults during aggregation. It may be node level or network (explained in further section) level which affects the performance very much. As discussed in further section 1.4, fault management techniques are required for efficient data aggregation. So our objective is to detect various faults and provide a recovery technique, so that the aggregation becomes reliable.

As an extension to our previous work, we propose a new reliable and accurate fault discovery and recovery mechanism for both node and network. It consists of detecting network faults and fault Recovery. Initially we designate some nodes as reliable nodes (R nodes) in the data aggregation tree, to perform accurate fault discovery and recovery. The R nodes collect the details of residual battery power and signal strength of all intermediate nodes. If the residual energy and signal strength of any intermediate node becomes less than a minimum threshold value at time t, then the succeeding R node will send a failure warning message to the previous R node. When an active node receives a failure warning message from other R nodes, it begins to cache a copy of each packet and try to forward the packet to the next R node with more residual energy and power. After receiving packets, each R node will send an acknowledgement to the previous R node within a time tack. If acknowledgment is not received within a specified time interval, then the R node is considered as faulty and next R node is selected.

LITERATURE REVIEW

Peng (2009) has proposed an improved Distributed Fault Detection (DFD) scheme by defining new detection criteria to improve the shortcomings of existing DFD. He has overcome some of the shortcomings of DFD such as the fault detection accuracy which decreased rapidly when the number of neighbor nodes to be diagnosed was small and the node's failure ratio was high. Also by applying his improved DFD to the sensor network, he has reached high fault detection accuracy with many neighbors of nodes to be diagnosed.

Mohammad *et al*. (2011) have proposed the in-network data aggregation approach which achieved ideal energy consumption by limiting a number of redundant and unnecessary responses from the sensor nodes. Also their approach could increase the chance of receiving data packets at the destination and caused more accurate results. Their schemes make use of the available path redundancy in the network to deliver a correct aggregate result to the data sink, which resulted in a high probability of completeness of responses, while still realizing significant power savings.

Sanam and Majid (2011) have proposed a cluster based method for fault detection and network connectivity recovery which used some of the nodes as gateway nodes in the network for implementing of voting mechanism. Due to the large impact of the permanent faults in the cluster head side, they have explored the fault tolerant mechanism for the permanent faults in the cluster head side. Their results have showed that their proposed CHFM for fault detection and network connectivity recovery was energy efficient and have used the new energy efficient method to fault recovery that prolonged the network lifetime.

Paulo *et al*. (2010) have extended their previous work on averaging by Flow Updating, in static settings, which have already introduced fault tolerance, achieving up to an order of magnitude improvement in convergence speed without increasing the message load. They have also brought attention to the vulnerabilities of state of the art averaging techniques when faced with failures and dynamic environments. In order to address the demanding scenarios, they have believed that the actual mass exchange might give way to idempotent flow management. So they have introduced a dynamic version of Flow Updating where entries for neighbor nodes were added or removed according to the current participants. Their simple design adapted to abrupt changes of network membership and tracked the continuous variations of network size with a good level of accuracy.

Arunanshu and Pabitra (2011) have presented a distributed self fault diagnosis model for Wireless Image Sensor Network (WISN) where fault diagnosis was achieved by disseminating decision made at each node. Their distributed self fault diagnosis model would detect both hard and soft faults in the network and each sensor

node make a decision based on comparison between its own reading and readings of its one hop neighbors. They have also presented architecture of fault tolerant wireless image sensor node. As a whole their model was simple and would detect faulty sensor nodes with high accuracy for a wide range of fault probabilities, while maintaining low message overhead.

PROPOSED SOLUTION

System model: In this section we will present the system model used in our study. First we state our major assumptions.

- We assume that there are multiple sink nodes in the wireless sensor networks, each of which has an infinite amount of energy.
- Also every sensor, whose location is randomly distributed, has the same initial energy and radio range while both the sensor nodes and the sink nodes are stationary.
- A perfect MAC layer with error free communication links is assumed and once the energy of a sensor node has been depleted, the communication is not possible.
- A transceiver exhibits first order radio model characteristics in free space i.e., energy spent in transmitting a bit over a distance d is proportional to its square (d^2).
- A Wireless Sensor Network (WSN) is modeled as a graph G (S, L), as show in Fig. 1, where S is the set of all sensor nodes and all sink nodes, L is the set of all links:

$$S = S_{sink} \cup S_{sensor}, L = \{(i, j) \mid i, j \in S\}$$

Every sensor's initial energy is E_{init} and its residual energy is E_{RES}. The path is defined as:

$$\{S_1, S_2 \ldots S_i, S_j \ldots S\}, S_i, S_j\, S_{sensor}, S\, S_{sink}$$

The cost is defined as the cost of one link $<S_i,S_j>$:

$$Cost_{ij} = \rho * d^2 + \sigma \quad (1)$$

Now we define the path cost as follows:

$$\text{Path Cost} = \Sigma Cost_{ij} * E_{RES}^{\lambda} \quad (2)$$

where, ρ is the energy/bit consumed by the transmitter electronics, σ is energy/bit consumed by the transmitting and receiving signal operation overhead for amplifying and d is the distance between two sensor nodes. λ is the coefficient of residual energy and it is a none-zero negative value.

From formula (1) and (2), it is clear that the longer the transmitting distance or the larger the overhead, the higher the cost. So the increase in the hop count between the sensor nodes and sink node will results in the increase of path cost. Also, if the residual energy for each sensor decreases, the path cost will increase. Hence, after a path has been used excessively, the residual energy of the sensors in the path will decrease, driving up the path cost and triggering the path switching process.

As shown in Fig. 1 let S_1, S_2 and S_3 be the three sinks, R denotes the R nodes and the remaining are normal sensor nodes. Each of the R nodes under each sinks share acknowledgements between them in order to ensure their conditions. If there is no acknowledgement from any of the R node, then the corresponding R node is identified as faulty.

Improved fault detection scheme: Our node fault detection scheme determines the status of node by testing among neighbor nodes mutually. For any two neighbor nodes N_i and N_j, a test result R_{ij} is produced by the data sensed by each of them. The data at the moment t_1 should be very close to each other because they are near and the difference (δd_{t1}) between this data should not exceed a certain threshold θ_1. In addition at another moment t_2 let the difference of the data of the two neighbor nodes be (δd_{t2}). The difference of δd_{t1} and δd_{t2} is $_{ij}$ which should not exceed a certain threshold θ_2. Let N be the total number of neighbor nodes for source S_i.

- For two neighbor nodes N_i and N_j, set R_{ij} as 0 and calculate δd_{t1}
  - If $\delta d_{t1} > \theta_1$, set R_{ij} as 1 and turn to the next node in the neighborhood.
  - If $\delta d_{t1} <= \theta_1$, calculate ϕ_{ij}. If $\phi_{ij} > \theta_2$, Set R_{ij} as 1 and turn to the next node in neighborhood.
- Repeat above steps until the test results of each node in neighbor node with S_i are all obtained.
- If $\sum_{S_j \in ZN} R_{ij} < N/2$, set initial detection status D_i of S_i as Possibly Normal (P_N), otherwise D_i is Possibly Faulty (P_F).
- Let N (P_N) be the number of neighbor nodes of S_i, whose initial detection status is P_N.
- If $\sum_{S_j \in N and D_j = P_N} R_{ij} < N(P_N)/2$, set the status of S_i as Normal (N), otherwise it's Faulty (FT).
- If there are no neighbor nodes of S_i whose initial detection status is P_N and if the initial detection status D_i of S_i is P_N, then set the status of S_i as Normal (N), otherwise as Faulty (FT).
- Check whether detection of the status of all nodes in network is completed or not. If it has been completed, then exit. Otherwise, repeat the above steps 1, 3, 4 and 5.

Estimation of link quality: In our approach, we also estimate the link quality based on two main parameters

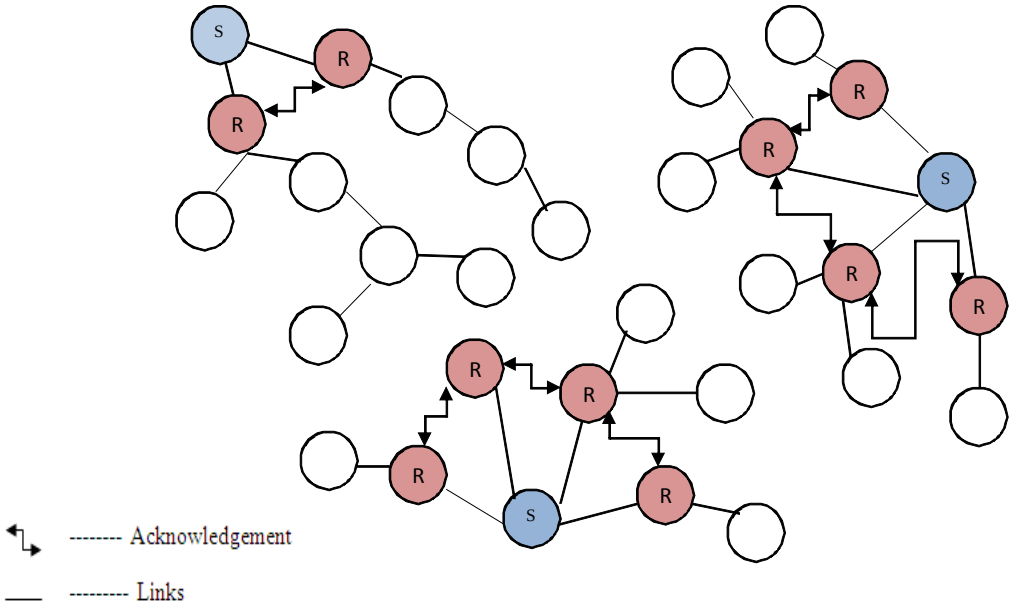

Fig. 1: Sensor network with R nodes

namely Signal to Noise Ratio (SNR) and Link Quality Indicator (LQI) (Carlo *et al*., 2010).

Signal to Noise Ratio (SNR): In general, SNR can differentiate between very good links and the rest. A link with a mean SNR above 20 dB can be safely considered as a very good link, but links with average SNRs between 5 and 10 dB are hardly distinguishable between bad, average and good. Hence, SNR cannot accurately classify the entire spectrum of link qualities independently but combining their information with another parameter could improve the classification process.

Link Quality Indicator (LQI): LQI presents a saturation that makes it incapable to distinguish between good and very good links. On the other hand, LQI shows a smoother decay that enables a better classification of bad, average and good links.

Let us assume that N packets are used to sample the channel and S of those packets are successfully received (0<S<= N). The LQI and SNR of each successfully received packet i are denoted by LQI_i and SNR_i. Upon reception of the sampling packets, the receiver then calculates the mean SNR and LQI in the following way:

$$SNR = \frac{\sum_{k=1}^{S} SNR_k}{N} \quad (3)$$

$$LQI = \frac{\sum_{k=1}^{S} LQI_k}{N} \quad (4)$$

When both the SNR and LQI calculated by using Eq. (3) and (4) falls above a minimum threshold value of SNR_{thr} and LQI_{thr}, respectively, then the link quality is good. On the other hand if it falls below both the thresholds then the link quality is bad and is not suitable for transmission.

Multipath routing: Our objective is to find an efficient approach to construct the appropriate set of K paths, such that the energy consumption for most of the sensor nodes is minimized. Let $E_{i,j}$ as the energy consumption for the sensor node S_i when the path Pj is followed by the sink, we have (Dakai *et al*., 2009):

$$E_{i,j} = PWR(d_{i,j}) * t \quad (5)$$

where, $d_{i,j}$ is the shortest distance from the path P_j to the location of sensor node S_i and PWR ($d_{i,j}$) is the corresponding power level. During one round of data collection the average energy consumption for sensor node S_i is defined as:

$$E_t = \frac{\sum_{j=1}^{K} E_{i,j}}{K} \quad (6)$$

For the WSN with N static sensor nodes being deployed in the field at known locations, construct the set of K paths $P_S = \{P_1 \ldots P_K\}$ for the sink, subject to:

$$L(P_j) \leq L \quad (7)$$

$$d_{ij} \leq d_{max} \quad (8)$$

where, L (P_j) stands for the length of path P_j . Here, the Eq. (7) states that, the length for any constructed path should be within the path length limit L. When followed the paths constructed Eq. (8) guarantees the distance from any sensor node to any path is within the maximum distance limit d_{max} for the sink to collect data directly from all sensor nodes.

The lifetime of WSNs under consideration depends on not only the number of paths (i.e., K) to be constructed, but also the track of each path. Due to the path length limitation, we assume that any constructed travel path within the path length limit cannot enable the sink to pass by the location of each and every sensor node. Therefore, to maximize the lifetime of a WSN, the planned paths should aim to balance the energy consumption on the sensor nodes. If a sensor node is far away from one path, it should be on or close to other travel paths of the sink. A feasible travel path for the sink is the one that satisfies the constraints represented by Eq. (7) and (8). That is, its length should be no more than the path length limit L and the distance from the path to any sensor node should be no more than the maximum distance limit d_{max}. Moreover, without exceeding the path length limit L, the constructed travel path should be as close to the sensor nodes that are not on the path as possible.

Following these guidelines, our multi path routing is based on the idea of R nodes, where a partial path is first constructed based on a subset of sensor nodes which is defined as R nodes and then expanded to consider other non R nodes. We first deal with construction of a single feasible travel path for the sink that will pass by a subset of R nodes. The algorithm for constructing such a single feasible travel path is given in algorithm-1.

Algorithm-1:

1. Get the path P_j from $_j$ using Dijkstra's shortest path routing algorithm.
2. Extend the path to ensure d_{max}
3. While ($S_i \in _j$, such that $d_{ij} > d_{max}$) do
   3.1 Expand path P_j to ensure $d_{ij} \leq d_{max}$

End while.

4. If (L (P_j) >L) then
   4.1. The construction of P_j fails and exits.

End if.

5. While L (P_j) <L) do
   5. 1. S_m = Select node for further expansion.
   5. 2. Calculate path increase δ for P_j to reach S_m.
   5. 3. If ($\delta \leq L_j$) then
      5. 3. 1. Expand P_j to reach S_m, such that $d_{mj} = d_{min}$;
   Else
      5. 3. 2. Expand P_j with length increase as Lj

   End if.

End while.

For a given subset of R nodes $_j$ (j = 1, 2 … K).

By exploiting the existing Dijkstra's shortest path routing algorithm, an initial partial travel path can be constructed by solving the corresponding shortest paths with the R nodes. The initial path will pass by all the R nodes, which enables the R nodes to transmit their data to the sink at the minimum power level PWR_{min}.

Then the initial partial path will be modestly expanded. That is, the sensor node can transmit its data directly to the sink with a higher power level if the distance from a non R node to the path is no more than d_{max}, without the need of any path expansion. On the other hand, for the non R nodes that have the distance more than d_{max}, then the path will be expanded iteratively to ensure that the distance from the path to all sensor nodes is not more than d_{max}. This will guarantee that the sink can accumulate the data from all sensor nodes directly while it follows the path. A feasible path is obtained if the length of the expanded path is not more than the path length limits L unless the shortest path algorithm fails to construct a feasible path with the selected R nodes. In such case, the algorithm can announce the failure or choose another set of R nodes. The path will be further expanded in the last step to reach as many non R nodes as possible when a feasible path is obtained with the path length being less than L. Specifically by following a certain strategy called fixed K multipath scheme, some non R nodes that are far away from the path will be selected and the path is expanded to pass by them one by one until the length of the path reaches the limit L.

Fixed K multipath scheme: From Algorithm 1, it is seen that the selection of R nodes is essential since these R nodes are able to transmit their data at the minimum power level PWR_{min}, when the sink follows the corresponding generated travel path. In our fixed K multipath scheme, the R nodes are statically determined for each path to be constructed. In order to guarantee that each sensor node serve as a R node and on at least one of the generated travel paths. When the j^{th} path is constructed using Algorithm 1, our fixed K multipath scheme first statically divides the sensor nodes into K subsets (with N/K R nodes in each set), where the j^{th} subset will serve as the set of R nodes. For our fixed K multipath scheme, the paths will be constructed independently (i.e.) the construction of one path does not depend on other paths. More specifically, during the further path expansion after a feasible path is created, the non R nodes that have the largest distance to the path will be selected, regardless of how much energy it consumes while the sink follows other (K-1) paths.

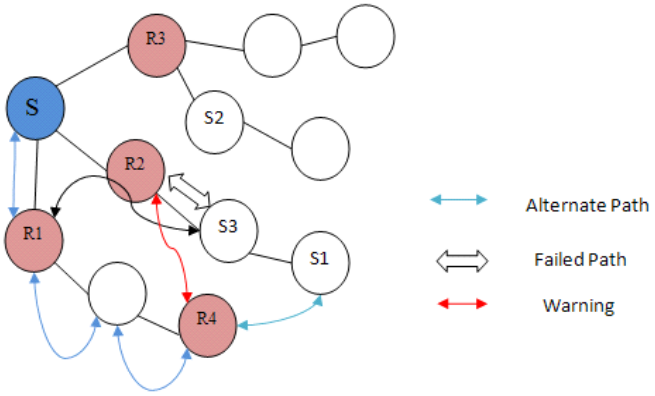

Fig. 2: Scenario briefing link failure

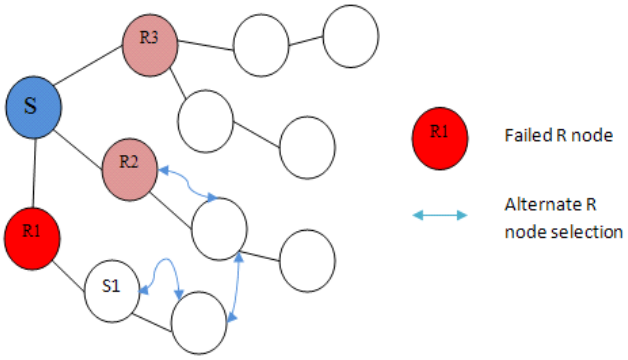

Fig. 3: Scenario briefing R node failure

Multipath based fault tolerance data aggregation: Our proposed multipath based reliable and accurate fault discovery and recovery mechanism is capable of detecting network faults and fault Recovery. Initially we designate some nodes as Reliable nodes (R nodes) in the data aggregation tree, to perform accurate fault discovery and recovery. These R nodes collect the details of residual battery power and signal strength of all intermediate nodes. If the residual energy and signal strength of any intermediate node becomes less than a minimum threshold value at time t, then the succeeding active node send a failure warning message to the previous active node. When an R node receives a failure warning message from other R nodes, it begins to cache a copy of each packet and try to forward the packet to the next reliable node with more residual energy and power. After receiving packets, each R node will send an acknowledgement to the previous R node within a time tack. If acknowledgment is not received within a specified time interval, then the R node is considered as faulty and next R node is selected.

Algorithm-2:

1. Select nodes nearer to sink as R nodes.
2. A feasible path from R node to the corresponding sink is determined by follow the steps in Algorithm-1.
3. It is then extended to determine multiple paths by Fixed K Multipath Scheme.
4. When R node receives data from sensor node S_i
   - 4. 1. If the initial detection status D_i of S_i is P_N, then
   - 4. 2. Set the status of S_i as Normal (N).

   Else
   - 4. 3. Set the status of S_i as Faulty (FT).

5. If status of S_i is N.
   - 5. 1. R node then calculates SNR and LQI of its preceding nodes by using Eq. (3) and (4).
   - 5. 2. If ($SNR>SNR_{thr}$ && $LQI>LQI_{thr}$) then
     - 5. 2. 1. The link is in good condition.

   Else
     - 5. 2. 2. The link is not suitable for transmission.

End if.

6. If the link condition of S_i is good then
   - 6. 1. The current path is suitable for transmission.

Else
   - 6. 2. A failure warning message is send to the nearby R node.
   - 6. 3. The new R node will cache a copy of each packet.

6. 4. The new R node would the select an alternate path to the sink and forward the data.
7. If R_Ack is not received by each R nodes then
    7. 1. The corresponding R node is faulty.
    7. 2. Select an alternate R node.
End if.

For instance now consider the following scenario as in Fig. 2, the link between node S3 and R2 fails the succeeding active node (R2) sends a failure warning message to the nearby R node (R4). When an R node (R4) receives a failure warning message from other R nodes (R2) and try to forward the packet to the next reliable node with more residual energy and power (i.e.,) in our case new R node (R4) is selected by S1.

Also if the R node (R1) of S1 fails to send an acknowledgement, now node S1 will select an alternate R node (R2) as shown in Fig. 3.

SIMULATION RESULTS

Simulation setup: Fault Management for Efficient Data Gathering technique is evaluated through NS2 [Network Simulator, http://www.isi.edu/nsnam/ns] simulation. A random network deployed in an area of 500X500 m is considered. We vary the number of nodes as 20, 40….100. Initially the nodes are placed randomly in the specified area. The base station is assumed to be situated 100 m away from the above specified area. The initial energy of all the nodes is assumed as 10.1 joules. In the simulation, the channel capacity of mobile hosts is set to the same value: 2 Mbps. The Distributed Coordination Function (DCF) of IEEE 802.11 is used for wireless LANs as the MAC layer protocol. The simulated traffic is CBR with UDP source and sink.
Table 1 Summarizes the simulation parameters used

Performance metrics: The performance of Fault Management for Efficient Data Gathering (FMDEG) technique is compared with our previous Cluster-based and Hierarchical Fault Management (CHFM) Sanam and Majid (2011) protocol. The performance is evaluated mainly, according to the following metrics.

Average end-to-end delay: The end-to-end-delay is averaged over all surviving data packets from the sources to the destinations.

Average packet delivery ratio: It is the ratio of the number of packets received successfully and the total number of packets transmitted.

Energy consumption: It is the average energy consumed by all the nodes in sending, receiving and forwarding operations.

Table 1: Simulation parameters

No. of nodes	20, 40, ….100
Area size	500X500
Mac	802.11
Simulation time	50 sec
Traffic source	CBR
Packet size	512
Transmit power	0.660 w
Receiving power	0.395 w
Idle power	0.035 w
Initial energy	10.1 J
Transmission range	75 m
No. of sinks	2
No. of sources	4
No. of aggregator nodes	4

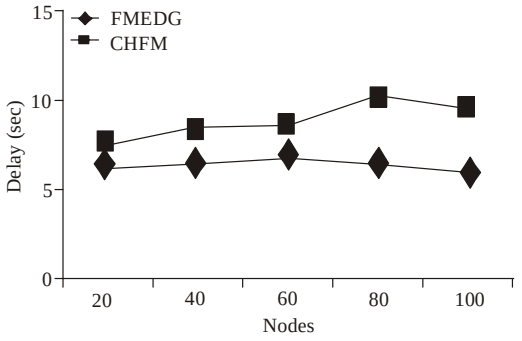

Fig. 4: Nodes vs delay

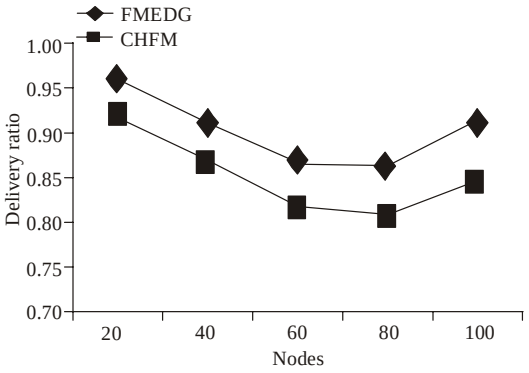

Fig. 5: Nodes vs delivery ratio

The simulation results are presented in the next section.

Simulation results: In our experiment, we vary the number of nodes as 20, 40, 60, 80 and 10 in which the sources are sparsely deployed.

Figure 4 gives the average end-to-end delay when the number of nodes is increased. From the Fig. 4, it can be seen that the average end-to-end delay of the proposed FMEDG technique is less when compared with CHFM.

Figure 5 presents the packet delivery ratio when the number of nodes is increased. FMEDG achieves good delivery ratio, compared to CHFM.

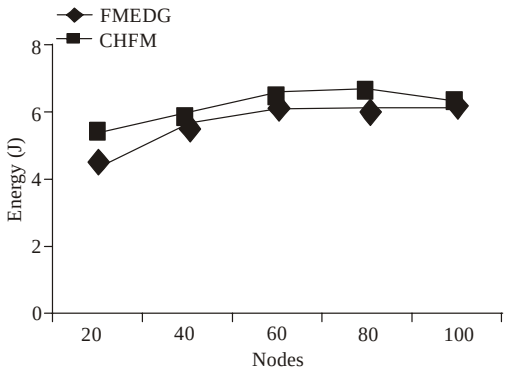

Fig. 6: Nodes vs energy

When the number of nodes is increased from 20 to 100, the route length increases, resulting in more packets exchanged. Hence the energy consumption increases for both the protocols. Figure 6 shows the results of energy consumption when the number of nodes is increased. We can see that FMEDG technique has less energy consumption when compared with CHFM.

CONCLUSION

In this study, we have developed an improved fault management scheme which can efficiently gather data in wireless sensor networks. Our node fault detection scheme determines the status of each node by testing the data values among its neighbor nodes. Initially, we assign some nodes as Reliable nodes (R nodes) in the data aggregation tree, to perform accurate fault discovery and recovery. These R nodes estimate the link quality based on two main parameters namely Signal to Noise Ratio (SNR) and Link Quality Indicator (LQI). Multiple paths are constructed, such that the energy consumption for most of the sensor nodes is minimized. When an R node receives a failure warning message from other R nodes, it tries to forward the packet to the next reliable node with more residual energy and power. By simulation results, we have shown that our proposed technique achieves good packet delivery ratio with reduced energy consumption and delay.

REFERENCES

Arunanshu, M. and M.K. Pabitra, 2011. Detection of Node Failure in Wireless Image Sensor Networks. ISRN Sensor Networks, 21 December.

Asim, M., H. Mokhtar and M. Merabti, 2008. A fault management architecture for wireless sensor networks. IEEE Conference (WCMC'08), pp: 779-785.

Asim, M., M. Hala and M. Madjid, 2010. A self-managing fault management mechanism for wireless sensor networks. Int. J. Wireless Mob. Network, 2(4).

Carlo, A.B., A.Z. Marco, V. Thiemo, W. Andreas and R. Kay, 2010. The triangle metric: Fast link quality estimation for mobile wireless sensor networks. Proceedings of 19th International Conference on Computer Communications and Networks (ICCCN), Zurich, 2-5 Aug.

Dakai, Z., G. Yifeng and S.T. Ali, 2009. Multi-Path planning for mobile element to prolong the lifetime of wireless sensor networks. 15th IEEE International Conference on Embedded and Real Time Computing Systems and Applications (RTCSA '09), Beijing, 24-26 Aug.

Hai, L., N. Amiya and S. Ivan, 2009. Fault-Tolerant Algorithms/Protocols in Wireless Sensor Networks. In: Sudip, M., W. Isaac and C.M. Subhas (Eds.), Handbook of Wireless Ad Hoc and Sensor Networks. Springer-Verlag, London.

Jinran, C., K. Shubha and S. Arun, 2006. Distributed fault detection of wireless sensor networks. Proceedings of the 2006 workshop on Dependability issues in wireless ad hoc networks and sensor networks DIWANS 06.

Jonathan, P.B., O.D. Tony and J.S. Cormac, 2007. Reliability control for aggregation in wireless sensor networks. 32nd IEEE Conference on Local Computer Networks (LCN 2007), Dublin, 15-18 Oct.

Lilia, P. and H. Qi, 2007. A survey of fault management in wireless sensor networks. J. Network Syst. Manage., 15(2).

Luciana, M.S.D.S., V. Harald and B. Michael, 2007. A Survey on Fault Tolerance in Wireless Sensor Networks. SAP Research, Karlsruhe University, Germany.

Mohamed, Y.M.Y. and V. Sundaram, 2011. Multiple sink based compressive data aggregation technique for wireless sensor network. Int. J. Wireless Mob. Network., 3(2).

Mohammad, H.A., H.A. Abdul and A.R. Shukor, 2011. Efficient data aggregation in wireless sensor networks. International Conference on Future Information Technology (IPCSIT), Singapore, Vol. 13.

Muhammad, Z.K., M. Madjid and A. Bob, 2009. Design Considerations for Fault Management in Wireless Sensor Networks. PG Net 2009, Liverpool.

Nandini, S.P. and P.R. Patil, 2010. Data aggregation in wireless sensor network. IEEE International Conference on Computational Intelligence and Computing Research.

Paulo, J., B. Carlos and S.A. Paulo, 2010. Fault-tolerant aggregation for dynamic networks. 29th IEEE Symposium on Reliable Distributed Systems, New Delhi, Oct. 31-Nov.

Peng, J., 2009. A new method for node fault detection in wireless sensor networks. Sensors, 9(2): 1282-1294, DOI: 10.3390/s90201282.

Sanam, H. and H. Majid, 2011. CHFM: Cluster-based and hierarchical fault management to fault detection and network connectivity recovery in wireless sensor networks. Aust. J. Basic Appl. Sci., 5(7): 243-248, ISSN: 1991-8178.

Shuo, G., Z. Ziguo and H. Tian, 2009. FIND: Faulty node detection for wireless sensor networks. Proceedings of the 7th ACM Conference on Embedded Networked Sensor Systems (SenSys '09), New York, USA.

Thomas, C., K.S. Kewal and R. Parameswaran, 2004. Fault tolerance in collaborative sensor networks for target detection. IEEE T. Comput., 53(3).

Winnie, L.L., D. Amitava and C.O. Rachel, 2007. Network management in wireless sensor networks. Management, 4(6): 1859-1873.