ISSN 2277-3061

Detection of Faulty Reading in Wireless Sensor Networks

Neelam, Sandeep Mehla, Matish Garg
Department of Computer Science and Engineering
Kurukshetra University, Kurukshetra, India

Abstract

In this paper, the problem of determining faulty readings in a wireless sensor network without compromising detection of important events is studied. By exploring *correlations* between readings of sensors, a correlation network is built based on similarity between readings of two sensors. By exploring Markov Chain in the network, a mechanism for rating sensors in terms of the correlation, called *Sensor Rank*, is developed. In light of Sensor Rank, an efficient in-network voting algorithm, called *Trust Voting*, is proposed to determine faulty sensor readings. Performance studies are conducted via simulation. Experimental results show that the proposed algorithm outperforms *majority voting* and *distance weighted voting*- two state-of-the-art approaches for in-network faulty reading detection.

General Terms

Algorithms; Design; Reliability.

Keywords

Faulty readings; Wireless sensor networks.

Categories and Subject Descriptors

H.3.4 [Systems and Software]: Distributed Systems.

Council for Innovative Research
Peer Review Research Publishing System
Journal: INTERNATION JOURNAL OF COMPUTERS AND TECHNOLOGY
Vol. 13, No. 9
editorijctonline@gmail.com
www.ijctonline.com, www.cirworld.com

ISSN 2277-3061

1. INTRODUCTION

Sensors are prone to failure in harsh and unreliable environments. Faulty sensors are likely to report arbitrary readings which do not reject the true state of environmental phenomenon or events under monitoring. Meanwhile, sensors may sometimes report noisy readings resulted from interferences [3]. Both arbitrary and noisy readings are viewed as *faulty readings* in this paper. The presence of faulty readings may cause inaccurate query results and hinder their usefulness. Thus, it is critical to identify and filter out faulty readings so as to improve the query accuracy. In this paper, we target at the problem of determining faulty readings in sensor networks. Obviously, a naive approach to this problem is to collect all readings to a sink, where statistical analysis is performed to determine what readings are outliers. However, this centralized approach may not be practical due to limited energy budget in sensor nodes. If readings are sent to the sink all the time, sensor Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Nodes may soon exhaust their energy. Nevertheless, simply filtering out unusual readings at individual sensor nodes may compromise monitoring accuracy of some important events. The goal of this study is to design energy efficient in- network algorithm for determining faulty readings without compromising detection of important events. The fact that data readings of nearby sensors are similar can be captured by *spatial correlation* [6]. Thus, an idea for determining faulty readings is to exploit this spatial correlation. In other words, if a sensor obtains an unusual reading, the sensor could inquire its nearby sensors (referred to as the *witness set*) by sending the suspected reading to them in order to determine whether the reading is faulty or not. We use clustering technique in this paper. By using clustering technique we make clusters, in each cluster we have a cluster head, it is the responsibility of cluster head to find faulty node in its cluster and give response to base station.

Based on the classical *majority voting*, each sensor (e.g., sensor *si*) in the witness set makes a judgment by comparing its own reading with the unusual reading sent by the suspected sensor (e.g., sensor *sj*). If the di®erence between these two readings exceeds a predetermined threshold, *si* considers the reading sent by *sj* as faulty and gives a negative vote to *sj* . Otherwise, *si* claims that *sj* is normal and returns a positive vote to *sj*. After collecting votes from the nearby sensors, *sj* then can conclude whether the reading is faulty or not. If the number of negative votes is smaller than that of positive votes, the unusual reading reported by *sj* is identified as a faulty reading. Otherwise, it is viewed as an observed event. However, this simple majority voting approach does not work well when the number of faulty sensors increases. To address the problem, two *weighted voting* methods have been proposed in the literature [5, 9]. Motivated by an assumption that the closer sensors have more resembled readings, the weighted voting algorithms give more weights to closer neighbors in voting (i.e., the weights are assigned inverse to the distances from a sensor node to its neighbors). In this paper, however, we argue that the distance between two sensors does not fully represent the correlation between readings of those two sensors. Furthermore, if the nearest sensor is faulty, the voting result may be seriously contaminated by this faulty sensor. We refer to this problem as a *domination problem* in the paper. Figure 1 illustrates a sensor network where the neighboring sensor nodes are linked. Each link is labeled by a weight (determined based on heuristics adopted by different voting methods) that will be used in voting. Assume that the weights of sensors *s*2, *s*3 and *s*4 are 0:3*;* 0:4 and 0:9, respectively, and sensor *s*4 is a faulty sensor.

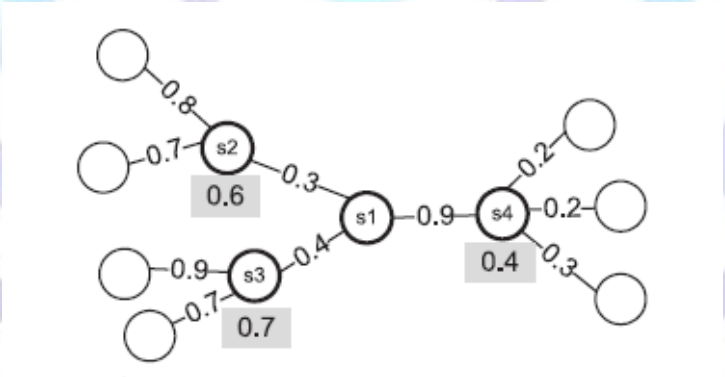

Figure 1: An illustrative topology of a Wireless Sensor Network

Obviously, the reading of sensor *s*1 is identified as a faulty reading when the weighted voting method is performed (i.e., 0.3*1+0.4*1+0.9*(-1)=-0.2) where positive and negative votes are represented by 1 and -1, respectively).As shown above, the distance-based weighted voting method has two primary deficiencies: 1) while the distance between sensor nodes may be a factor in generating similar readings of nearby sensor nodes, it does not precisely capture what we really care about as the *correlation* between sensor readings; 2) while it is a good idea to inquire opinions of neighbors, the *trustworthiness* of neighbors is not considered.

Based on the above observation, in this paper, we propose an innovative in-network voting scheme for determining faulty readings by taking into account the correlation of readings between sensor nodes and the trustworthiness of a node. To obtain the pair-wise correlations of sensor readings, we construct a logical *correlation network* on top of the sensor network. The correlation network can be depicted by a set of vertices and edges, where each vertex represents a sensor node and an edge between two sensor nodes denotes their correlation (i.e., the similarity between their readings).If two nearby sensor nodes do not have any similarity in their readings, these two sensor nodes do not have an edge connected. Therefore, only sensor nodes that are connected by correlation edges are participated in voting.

The weighted voting method actually uses the correlation (modeled as a function of distance) between sensor nodes as weights. However, using the correlation alone may not correctly identify faulty readings due to the domination problem discussed above. Thus, in the proposed algorithm, each sensor node is associated with a trustworthiness value (called *Sensor-Rank*) that will be used in voting. Sensor Rank of a sensor node implicitly represents the number of 'references' (i.e., similar sensor nodes nearby) it has to support its opinions.

A sensor node will obtain a high Sensor Rank if this sensor has many references. The number under each sensor node in Figure 1 is its Sensor Rank. In the figure, *s*4 has a small Sensor Rank because the readings in *s*4 are not very similar to that of its neighbors. By using Sensor Rank, our voting scheme takes the trustworthiness of each sensor into account. A vote with small Sensor Rank has only a small impact on the final voting result.

For example, in Figure 1, when s1 inquires opinions from its neighbors (i.e., *s*2, *s*3 and *s*4), the vote from *s*4 has a small impact due to its lower Sensor Rank.

Our design consists of two parts:

- An algorithm that calculates Sensor Rank for each sensor node.
- An algorithm that use Sensor Rank to determine faulty readings.

Specifically, we first obtain correlations among sensor readings and then model the sensor network as a Markov chain to determine Sensor Rank. In light of Sensor Rank, the *Trust Voting* algorithm we developed will be invoked as needed in operation to effectively determine faulty readings. A preliminary performance evaluation is conducted via simulation. Experimental result shows that the proposed Trust Voting algorithm is able to effectively identify faulty readings and outperforms *majority voting* and *distance weighted voting*, two state-of-the-art voting schemes for in-network faulty reading detection for sensor networks. A significant amount of research effort has been elaborated upon issues of identifying faulty sensor readings [2, 5, 6, 9]. In [6], the authors explored spatial correlation among sensors and proposed a distributed Bayesian algorithm for detecting faulty sensors. By assuming that faulty measurements are either much larger or much smaller than normal measurements, the authors in [2] use a statistical method to detect outlier measurements. Some variations of the weighted voting technique for detecting faulty sensors are proposed in [5] and [9].

2. BACKGROUND AND HISTORY

This Chapter describes relevant background knowledge and related work for readers to easily understand the proposed protocol and the methodology and analysis of our experiments.

- **Wireless Networks**

Today's Internet has been developed for more than thirty years. Recently many network Researchers are studying networks based on new communication techniques, in particular Wireless Communications. [1] Like traditional wired networks, wireless networks are produced by routers and hosts. In a wireless network, the routers are responsible for forwarding packets in the network and hosts may be sources or sinks of data flows. People can deploy a wireless network very easily and speedily .The basic difference between wired and wireless networks is the way that the network components communicate. A wired network depends on physical cables to transfer data. As it is understood that in a wireless network, the communication among different network components can be either wired or wireless. As wireless communication does not have the constraint of physical cables, it allows a explicit freedom for the hosts and/or routers in the wireless network to move. This is one of the advantages of a wireless network [2]. Wireless LANs present the following productivity, convenience, and cost advantages over the wired networks: [3]

- Mobility: Wireless LAN systems can provide LAN users with access to real-time information anywhere in their organization. Mobility supports productivity as well as service opportunities which are not possible with wired networks.
- *There are now thousands of hotels, universities and public places with public wireless connection. These free us from having to be at home or at work to get access the Internet.*
- Reduced Cost-of-Ownership: Though the initial investment required for wireless LAN hardware can be higher than the cost of wired LAN hardware, but overall installation expenses and life-cycle costs can be considerably lower. Long-term cost benefits are greatest in dynamic environments requiring frequent moves and changes.
- Installation Speed and Simplicity: It is very easy and quick process to install a wireless LAN system and thus allows to get rid of the need to pull cable through walls and ceilings.
- Scalability: Wireless LAN systems can be configured in many different types of topologies to fulfil the needs of particular applications and installations. Configurations can be effortlessly changed and range from peer-to-peer networks suitable for a few numbers of users to full infrastructure networks of thousands of users that enable roaming over a broad area.

In a wireless network the network components communicate with each other by the use wireless channels. Different radio frequency (RF) spectrum ranges are used in wireless networks, as for example, 2.5-2.7 GHz for the Multipoint Multichannel Distribution System ,27.5-29.5 GHz for the Local Multipoint Distribution System (LMDS) and 5.15-5.35 GHz and 2.4-2.58 GHz for IEEE 802.11a and 802.11b , correspondingly. The strength of the signal in a wireless medium decreases when the signal travels further [2].

When the signal travels beyond some distance, the strength gets reduced to the point where reception is not possible. The distance that a signal travels when it reaches to this point is called the radio range for the given signal.

To simplify the transmission model regarding this property, people think that the wireless signal is strong enough for the receivers to receive the signal if the receivers are inside of the radio range [2].

Several medium access control (MAC) protocols are used in wireless networks to manage the use of the wireless medium. Few examples like the Bluetooth MAC layer protocol [4] and IEEE 802.11MAClayer protocol [5] .The details of these MAC protocols are outside our scope. Refer to [4] and [5] for more details.

Because radio range is generally limited and the network components may have some mobility, the topology of a wireless network can differ with time. According to the relative mobility of hosts and routers, varying kinds of wireless networks are:

- **Fixed wireless network**

This type of wireless technology helps in communication between the two locations such as two buildings or two offices for the sake of business and for different purposes with the help or involvement of radio waves [6]. Fixed hosts and routers make use of wireless channels to communicate with each other and form a fixed wireless network. On the whole technology of fixed wireless is totally based on the **Wireless LAN** infrastructure phenomenon. Fixed wireless is also helpful economically as it is wireless so it saves the money which are used for the laying the cables between two sites and provide the better results. Chief benefit of fixed wireless is fixed wireless broadband which is a very useful source of communication between different areas. Fixed wireless is also deployable in those areas where, there is very less possibility to have wired networking as for example, in rural areas there is no wired infrastructure technology accessible so far, this problem has been solved by fixed wireless. So, by the use of this application people of rural areas can also compete in the race of modern world. An example is a wireless network formed by fixed network devices using directed antennas, as shown in Figure1.1.a.

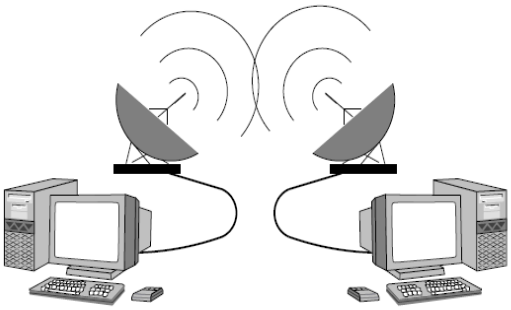

Figure1.1.a: An example of fixed wireless networks [7]

3. Related Works

Malek [20] proposed the first comparison-based diagnosis model. This model assumed that, in a system with *N* units, it is likely to compare outputs generated by task executions from a few or every pair of units. The unit that performed comparisons was called a comparator. A comparison that resulted in a mismatch indicated that one or both units were faulty. It was possible that both units being compared were faulty, and in that case the comparison must indicate a mismatch. Thus this model assumed that:

(1) The Outputs generated by two fault-free units that execute the same task were same always;

(2) The output generated by a faulty unit must be different from the outputs generated by any other unit (faulty or fault free) for the same task.

This model consisted of two activities: fault detection and fault location. The goal of fault detection was only to determine the presence of faulty units in the system, but was not possible to determine which units were faulty. Fault location allowed the identification of faulty units. It also assumed that a central observer exist which collects and maintains information about the task outputs. This central observer also performed the diagnosis of the system based on comparison results, determining the system's faulty units. The central observer was a trustful reliable unit that never fails. When the outputs of two units were compared, the possible outcomes were shown in table3.1 given below. The set of possible comparison outcomes was also called the *invalidation rule*. The outcome *pass* indicated that both units were fault free, while *fail* indicated that at least one of the units was faulty. In that case, more comparisons were required to identify the faulty unit. It was proved that, in a system with *N* units in which comparisons of every pair of units is feasible, the maximum number of faulty units is $N - 2$ for the diagnosis to be correct, that is, the diagnosability would be $N - 2$.

Unit1	Unit2	Comparison outcome
Fault-free	Fault-free	0(pass)
Fault-free	Faulty	1(fail)
Faulty	Fault-free	1(fail)
Faulty	Faulty	1(fail)

Table3.1: Possible Comparison Outcomes of Malek's Model [20]

Chwa and Hakimi [21] proposed a model nearly the same as compared to the diagnosis model by Malek [20].This model also assumed a central observer which performed the complete diagnosis of the system based on comparison outcomes. The only difference between this model and the previous one was that when two faulty units receive the same task to execute, they might produce the same outputs, means, the comparisons of these two tasks outputs might resulted in a match.

Maeng and Malek [22] proposed a model named as MM Model for systems composed of multiprocessor systems consisting of homogeneous processors. The system was represented as a graph G=(V,E)where V was set of units and E was set of communication links. In the MM model, the states of the units were determined by comparing the task output of one unit with the output generated by another unit for the same task. The main difference of the MM model to the previous models [20; 21] was that it allowed the comparisons to be performed by the units themselves, means, units were also comparators. A unit k was a comparator of units i and j only if $(k, i) \in E$ and $(k, j) \in E$; furthermore $k \neq i$ and $k \neq$ j.A diagnosable system under the MM model was represented by a multigraph $M = (V, C)$ defined over the same set of units of graph G. Each edge $(i, j)k \in C$ represented the outputs from units i and j compared by another unit k. M was a multigraph because the outputs from each pair of units might be compared by more than one unit of the system, means, more than one edge might exist between the same pair of vertices. The notation $r((i, j)k)$ was used to represent the comparison result of units i and j by unit k. The result was 0 when the comparison matches and the result was 1 when the comparison indicated a mismatch. If $r((i, j)k) = 1$, at least one of the units i, j or k was faulty. If the result was 0 and the tester k was fault free, then i and j were also fault free. But if the tester k was faulty, the comparison outcomes were not reliable and it was not possible to obtain any conclusion about the state of units i and j. Comparison results were still sent to a central observer that achieved the complete diagnosis. The authors also gave necessary and sufficient conditions for one-step t-diagnosability and also evaluated the diagnostic latency in terms of test cycles. All possible outcomes are shown in table3.3 given below.

SMaeng and Malek [22] also proposed a MM* Model, the special case of the MM Model, the only difference that each unit compared every pair of neighbour units with which they were connected .The comparison outcome was then sent to the central observer for the complete diagnosis of the network.

Ammann and Cin [23] also investigated the diagnosability of comparison based diagnosis and showed that a required condition for a system to be t-diagnosable was that each node in the testing graph had a degree at least t; a minimum degree strictly greater than t was a sufficient condition. The degree—or order—of a node was the number of edges incident on this node. The authors also presented an algorithm for sequential diagnosis of a subset of t-diagnosable systems. The complexity of the proposed algorithm was proved to be $O(N^2)$.

Sallay et al. [24] considered faults affecting the comparator and the central observer. In order to try to diagnose the comparators, the authors proposed a strategy to exhaustively run comparisons of fault-free units and comparators. These tests were performed with different input tasks and it was assumed that a faulty unit always produces the same response for the same input task. The authors applied their proposed approach to wafer-scale circuits, providing a simple money-making wafer design solution.

Pelc [25] performed an algorithmic analysis of both Malek's [20] and Chwa and Hakimi's [26, 21] comparison models, and named them as asymmetric and symmetric Models respectively. In the analysis the author showed the worst-case number of tests for optimal algorithms for t-diagnosis, sequential t-diagnosis, and one-step t-diagnosis for both the models. Also considered adaptive and non adaptive testing and showed that using adaptive testing, the number of tests were often smaller. The minimum number of tests needed for completing t-diagnosis, $t \leq N$, under Malek's model [20] was$\lceil N/2 \rceil$. In case of sequential t-diagnosis (identifies at least one faulty unit), where $t \leq N-2$, the minimum number of tests required is MAX ($N/2*t$) +1 when an adaptive testing strategy is employed and $N - \lfloor N/(t+2) \rfloor$ for non adaptive diagnosis. In case of adaptive one-step t-diagnosis (identifies all faulty units in one step), when $t \leq N - 2$ the minimum number of tests is $\theta(N2/(N-t))$ and when $N \geq 2t + 1$ the number of tests is $\lfloor N/2 \rfloor + 3.5\lceil t/2 \rceil + 3$.For non adaptive one-step t-diagnosis, $t \leq N-2$the minimum number of tests is $\theta(Nt)$. The minimum number of tests for completing t-diagnosis, where $t \leq N - 1$ for Chwa and Hakimi's [26, 21] model is $N - \lfloor N/(t+1) \rfloor$. In case of sequential t-diagnosis, where $t < N/2$ the minimum number of tests required is $N - N/(t+1) + 1$ when an adaptive testing strategy is employed and $N - \lfloor N/(2t+1) \rfloor$ for non adaptive diagnosis. In case of one-step adaptive t-diagnosis, when $t < N/2$ the minimum number of tests is $\theta(N)$. For non adaptive one-step t-diagnosis, if $t < N/2$ the minimum number of tests is $\theta(Nt)$.

Barborak et al. [27] surveyed the first comparison based diagnosis models. This was a key paper in which diagnosis was treated in a unifying framework together with other distributed problems and algorithms, including consensus and the Byzantine Generals problem. The authors presented, a detailed fault classification, including the specification of the incorrect computation fault model, which best defined the faults that can be handled by comparison-based diagnosis. This was relevant because several early diagnosis papers only implicitly presented the assumed fault model, by specifying how faults were detected. The survey also argued that if the frequency in which two units became faulty was low, then there was a low probability that they would be faulty at the same time. Thus two units executing the same tasks should produce identical results unless one, or both, had become faulty.

Kozlowski and Krawczyk [28] extended Chwa and Hakimi's [26] diagnosis model for hybrid fault situations. A hybrid fault situation was defined to be t/m-restricted if the number of faulty units did not exceed t and the number of misleading comparison outcomes was less than m. A misleading comparison outcome was supposed to be occurred when a fault-free unit would evaluate a faulty unit as fault free. The authors also presented an $O(N|C|)$ algorithm for comparison-based diagnosis under a hybrid fault situation.

Chen et al. [29] presented an extension to the MM model. Their model considered both processor and comparator faults separately. Therefore, a processor would either execute tasks or performed comparisons. It was also shown that the system diagnosability was $t \leq \lfloor \delta/2 \rfloor$, where δ was the minimum degree of nodes in the system. Though, they also showed that if the number of faulty comparators was less than the number of faulty processors, the diagnosability would reach $t \leq \delta$. The authors also presented an optimal $O(|E^*|)$ algorithm for the diagnosability $t \leq \lfloor \delta/2 \rfloor$, and an $(|E^*|2)$algorithm for the diagnosability $t \leq \delta$, where E^* was the set of comparators.

Dahabura et al. [30] proposed a Probabilistic Comparison-Based Model in which a pro babilistic approach was used which assumed that a node might be failed with a certain probability. Thus there was no restriction on the number of faulty units in the network. There were basically two probabilistic approaches. In the first one the diagnosis was restricted to a set of faulty units with high probability. In the second approach the diagnosis of the whole system was performed first and later it was proved to be correct with a high probability.

Sengupta and Dahbura [31] proposed a polynomial-time algorithm, with complexity $O(N^5)$, for comparison-based diagnosis under the MM* model. The authors represented a given system by graph $G = (V, E)$ whenever $(i, j), (i, k) \in E$ node i compared the results of nodes j and k. The algorithm adaptively determined the comparisons to be executed on the basis of comparison results. A node i running that algorithm started comparing two nodes $j, k \mid (i, j), (i, k) \in E$, means, node i performed the comparison $(j, k)i$. If the comparison outcome $r((j, k)i) = 1$ (mismatch), then node i would choose another different pair of nodes to compare, if there was such a pair. If the comparison outcome $r((j, k)i) = 0$ (match), then node i would use node j in order to compare *all* its neighbours, means, all comparisons $(j, p)I \mid (p, i) \in E$.

Blough and Brown in [32] proposed Broadcast Comparison Model. This model was fully distributed which was based on MM* model for the systems with reliable broadcast. Here a task was assigned to a pair of nodes which performed the comparisons and diagnosed the system. This model diagnosed static as well as dynamic faults in a polynomial time.

Yang and Tan [33] presented a diagnosis algorithm for the MM* model with time complexity $O(N \times \Delta^3 \times \delta)$, where Δ and δ were respectively the maximum and the minimum degrees of a node. Their algorithm was introduced as an alternative to Sengupta and Dahbura's [31] $O(N^5)$ algorithm. The authors argued that realistic diagnosable systems, such as massive multicomputer, were sparsely interconnected. When $\Delta, \delta \ll N$ then the proposed algorithm would behave better than Sengupta and Dahbura's [31].

Nakajima [34]; Hakimi and Nakajima [35] introduced an important result in system level diagnosis called as *adaptive* diagnosis. Previous models consisted of initially selecting the set of tests to be executed, then executing those selected tests, and finally evaluating the test results in order to identify all faulty units. In adaptive diagnosis, the set of tests which were to be executed were dynamically determined, based on the results of previous tests. The first adaptive diagnosis model was introduced by Nakajima [34]. Assuming a system S of N units with no more than t faulty units, the proposed model adaptively chose and executed tests, repeating the process until a fault-free unit was identified. Then that unit was employed as a tester from which all faulty units were identified. It was proved that $(N - 1) + t(t + 1)$ tests were sufficient to identify all faulty units in such a system. In adaptive diagnosis and all other previous models, test results were collected and processed by an external entity which determined the state of all system units.

Kuhl and Reddy [36, 37] proposed *distributed* system-level diagnosis in which the fault-free nodes of the system themselves diagnosed the state of all nodes. Those nodes executed tests and exchanged test results with each other. The authors proposed the SELF distributed system-level diagnosis algorithm that, although fully distributed, was non adaptive, means, each unit had a fixed testing assignment.

Later Hosseini et al. [38] extended the SELF algorithm, thus presented the NEW-SELF algorithm which allowed all fault-free nodes to independently diagnose the state of all nodes, and provided the total number of failures does not exceed a given bound t.

Bianchini et al. [39] proposed The EVENT-SELF algorithm which used event-driven techniques to reduce the amount of network resources required for diagnosis.

Bianchini and Buskens [40, 41] proposed Adaptive-DSD which was, at the same time, distributed and adaptive. Adaptive-DSD was executed at each node of the system at predefined *testing intervals*. Each node was tested only once per testing interval. A *testing round* was defined as the period of time in which all nodes of the system had executed their assigned tests at least once. All fault-free nodes achieved consistent diagnosis in at most N testing rounds. Up to $N - 1$ nodes might be faulty so that fault-free nodes would still able to diagnose the system. Each time the algorithm was executed on a fault-free node, it performed tests on other nodes until another fault-free node was found, or the tester runs out of nodes to test. Thus the testing graph was a ring connecting fault-free nodes. When the tester executed a successful test, means, the tested node was fault free, the tester obtained diagnostic information from the tested node. The *diagnosis latency* was defined as the number of testing rounds required by all fault-free nodes to complete the diagnosis of the system. Adaptive-DSD had a worst-case latency of N testing rounds. Adaptive-DSD was implemented and practical results were presented that showed the effectiveness of the algorithm when used to monitor a real Ethernet network.

Chessa and Santi [42] considered the problem of identifying faulty mobiles in ad-hoc networks. Current diagnostic models were designed for wired networks; therefore they did not take benefit of the shared nature of communication typical of ad-hoc networks. Here the authors introduced a new comparison-based diagnostic model based on the one-to many communication paradigms. Two implementations of the model were presented. In the first implementation, authors assumed that the network topology does not change during diagnosis, and showed that both hard and soft faults can be detected very easily. Based on this implementation, a diagnosis protocol was presented. The evaluation of the communication and time complexity of the protocol indicated that efficient diagnosis protocols for ad-hoc networks based on our model could be designed. In the second implementation authors allowed the system topology to change during diagnosis. The ability of diagnosing faults under this scenario was significantly reduced with respect to the stationary case.

Elhadef and Boukerche [43] stated that dependable mobile ad-hoc networks were designed to provide reliable and continuous service despite the failure of some of their components. The major building blocks that have been identified for such fault tolerant systems was the failure detection service which aimed at providing some information on which hosts would have crashed. In this paper, the authors presented a new implementation of a failure detection service for wireless ad-hoc and sensor systems that was based on an adaptation of a gossip-style failure detection protocol and the heartbeat failure detector. They showed that their failure detector was eventually perfect− it satisfied both properties: strong completeness and eventual strong accuracy. Strong completeness meant that there was a time after which every faulty mobile was permanently suspected by every fault-free host. While, eventual strong accuracy referred to the fact that no host would be suspected before it crashed. The proposed failure detector was a variant of the heartbeat failure detector and allowed each host to maintain a list of hosts it currently suspected of having crashed. The key features of their failure detector were that it was adaptable and dynamic, that is, it adapted the freshness points to the current network or hosts' load. The distributed failure detection service could be used by distributed applications directly, or support other middleware services such as system management, load balancing and group communication and membership services. As such, failure detection was a valuable extension to current dependable services that a wireless environment was expected to provide.

Elhadef et al. [44, 45] presented comparison-based diagnosis protocols for mobile ad hoc networks. Two protocols were presented: the Adaptive Distributed Self-Diagnosis Protocol (Adaptive-DSDP) for fixed topology networks, and the Mobile Distributed Self-Diagnosis Protocol (Mobile-DSDP) for time-varying topology networks. The basic idea of both protocols was that a node, when replying to a test request, should also provide the test task along with its output for that test. So any receiver would be able to diagnose its state by simply comparing that output to similar outputs for the same test, or even by comparing the received result to its own output after performing the same test. Besides the fact that nodes forward tasks with test responses, the fixed-topology diagnosis model on which Adaptive-DSDP was based also differed from Chessa and Santi's model [42] in their dissemination strategies. In Chessa and Santi's [42] model, once a node collected all its neighbours' responses, it forwarded its local view to all other nodes in the MANET using a flooding-based dissemination phase. In contrast, Adaptive-DSDP used a spanning tree and a gossip-style dissemination strategy [44].

Again Elhadef et al. [46] presented another distributed comparison-based self-diagnosis Protocol for wireless ad hoc networks based on Chessa and Santi's [42] model. The proposed protocol was called Dynamic-DSDP which also identified hard and soft faults.

Xu et al. [47] proposed a new algorithm for routing in mobile survivable networks, based on the grouping of position-based routing concepts and fault tolerant routing techniques in computer networks. Taking the combination of these two concepts, they employed an easy way of localizing routing overhead though at the same time they improved the operational effectiveness of the position-based routing approaches by alleviating some of the drawbacks related with them, such as deadlock occurrences during routing, and thus creating a robust and fault tolerant routing strategy. The algorithm proposed here was based on the Position guided Sliding-window Routing (PSR) protocol. This protocol provided a single-tier routing organization scheme by employing an easy way of localizing routing overhead. In this paper the authors have enhanced this approach by adding an additional level of hierarchy (on the cluster level which is of much smaller scale) so as to improve the operational effectiveness of this scheme and alleviate some of the drawbacks related with the position-based protocols (such as routing deadlock occurrences).To overcome the drawbacks i.e. Deadlock and loop that are inherent to the position based routing schemes, gateways were used as intermediate hops along the path to the destination. When a packet arrived to a gateway, a few calculations were performed at the gateway to see if there exists a path between the local nodes to another gateway of local cluster that was closer to the destination. If deadlock/loop was found during that operation, then a request was sent to the gateway of the previous cluster to change the path to another cluster. The grid-clustered PSR was used to avoid deadlock/loop creation.

Although the authors adopted the concept of cluster creation, but avoided the use of cluster-heads and thus achieved high level of fault-tolerance. Also, because of the introduction of cluster level, less service information was required to be transmitted, so it was easier to use more elaborate adaptive routing techniques, further improving network performance. The authors performed a complete and in-depth comparative evaluation of the basic PSR approach with the grid-clustered PSR scheme, in terms of many performance parameters such as, average packet delivery time, maximal throughput, number of hops, packet delivery ratio, etc.

Duarte and Nanya [48] Considered a system composed of N nodes that can be faulty or fault-free. The aim of distributed system-level diagnosis was to have each fault-free node determine the state of all nodes of the system. This paper presented a Hierarchical Adaptive Distributed System-level Diagnosis (Hi-ADSD) algorithm, which was a fully distributed algorithm that allowed every fault-free node to achieve diagnosis in, at most, $(\log_2 N)^2$ testing rounds. Nodes were mapped into progressively larger logical clusters, so that tests were run in a hierarchical fashion. Each node executed its tests independently of the other nodes, i.e., tests were run asynchronously. All the information that nodes exchanged was diagnostic information. The algorithm assumed no link faults, a fully-connected network and imposed no bounds on the number of faults. The worst-case diagnosis latency and correctness of the algorithm were formally proved. The algorithm was implemented on a 37-node Ethernet LAN, integrated to a network management system based on SNMP (Simple Network Management Protocol) as an example application and then Experimental results of fault and repair diagnosis were also presented.

Moallemi and Moghaddam [49] described that Resource reservation and mutual exclusion are challenging problems in mobile ad-hoc networks (MANET). Because of the dynamic characteristics of nodes in these networks, yet, few algorithms have been proposed. Another problem in these networks is link or node failure due to many reasons (e.g. hardware software crash, running out of battery, getting out of transmission range due to high mobility). So fault tolerance for these algorithms is another necessity which hasn't been completely accomplished. In this paper authors proposed an algorithm which was totally fault tolerant (covers temporary and permanent faults). It also had the mutual exclusion property for critical resource reservations. The authors used three recovery processes in their proposed algorithm to maintain the stable state for whole system. At last the authors proved the proposed algorithm's Safety and Aliveness properties to show its integrity.

Khilar et al. [50] proposed a scalable failure detection service for large scale ad hoc networks using an efficient cluster based communication architecture. Their failure detection service adapted the detection parameter to the current load of the wireless ad-hoc network. The proposed approach used a heartbeat based testing mechanism to detect failure in each cluster and took the advantage of cluster based architecture to forward the failure report to other cluster and their respective members. The failure detection algorithm together with suitable clustering algorithm made a very efficient failure detection service for wireless ad-hoc networks. Clustering divided entire network into two level communication architecture namely intra-cluster and inter-cluster. Two types of message overheads were required to maintain such as intra-cluster and inter-cluster. The disadvantage of the clustering approach was that CH itself might fail, hence it became necessary that the presence of leader was also needed to be monitored and in case of its failure another node might took over the CH. Author used the concept of deputy cluster head or backup cluster head to solve this problem.The simulation results showed that this approach was linearly scalable in terms of message complexity and consensus time.

Rangarajan and Dahbura [51] described a distributed algorithm for detecting and diagnosing faulty processors in an arbitrary network. Fault-free processors performed simple periodic tests on one another; when a fault was detected or a newly-repaired processor joined the network, that new information was disseminated in *parallel* throughout the network to minimize the information latency in the network. It was formally proven that the algorithm was correct; and it was also shown that the algorithm was optimal in terms of the time required for all of the fault-free processors in the network to learn of a new event. Simulations of the algorithm using the process-oriented simulation language **CSIM** showed that parallelizing the dissemination stage also allows for nodes that are local to the event to, in general, learn about the event before more distant nodes. Further, in their algorithm, a newly repaired node could rejoin the system without relying on other nodes to first detect that it had been repaired; equivalently, faulty nodes did not have to be periodically tested. The algorithm provided an option through which ***dead messages*** could be removed at the cost of increasing the information latency; means, a tradeoff could be made between message overhead and latency.

Bharath et al. [52] described that reliable distributed systems provide high availability for an important class of applications through a combination of software and hardware support. Redundancy and replication were key features of these systems but both came with a high cost. One trend that promised to provide more intelligence to the allocation of resources in that environment was adaptation. Adaptive fault tolerance was the idea of adaptively configuring system resources to respond to environmental changes (i.e. faults). They presented an overview of several adaptive fault tolerant systems, and described the challenges involved in their implementation .They described a unified model highlighting fundamental components in the design of an adaptive fault tolerant system. They used their model to describe a selection of recent representative systems and exposed the design decisions made during their construction. Adaptive fault tolerance could increase availability, reliability and decrease cost in a distributed computing environment. Current AFT systems are mature in their use of redundancy, communication and synchronization but to further the goal of reliability other directions need to be explored. Environment awareness and other proactive measures were features of AFT that they believed future systems would attempt to leverage.

Vanaja and Umarani [53] stated that the increasing popularity in wireless communication devices and the advancements in wireless technology make the communication in an effective and efficient manner. Mobile ad hoc Network (MANET) is a kind of wireless network, having collection of mobile nodes communicating through wireless links without using any infrastructure. Routing Protocols are essential for forwarding of data packets to have effective communication. The performance of MANET routing protocols reduce/hampers the network performance when there is a link break. This paper mainly dealt with the fault management to resolve the mobility induced link break. The proposed protocol was the adaptive fault tolerant multipath routing (AFTMR) protocol which reduced the packet loss due to mobility induced link break. In that fault tolerant protocol, battery power and residual energy were taken into account to determine multiple disjoint routes to every active destination. When there was link break in the existing path, AFTMR initiated Local Route Recovery Process. Network Simulator NS-2 was used for implementation and performance was analyzed using the quantitative metrics such as packet delivery ratio, end to end delay, control overhead, throughput and packet drop. Simulation results showed that the proposed protocol achieved better packet drop and energy, better throughput and packet delivery ratio with reduced delay,.

Albini et al. [54] proposed the generalized distributed comparison-based model: a hierarchical, adaptive and distributed model based on Sengupta and Dahbura's model- *Hi-Comp* diagnosis algorithm: required at most $O(N^3)$ comparisons and had worst-case latency of $O(log_2 N)$ rounds.

Albini et al. [55] presented an adaptive distributed system-level diagnosis algorithm, called as *Hi-ADSD with Detours* having latency at most $log_2^2 N$, but required less tests and less diagnostic information than other hierarchical diagnosis algorithms. Nodes running the new algorithm were grouped in clusters. If a tested node was faulty, instead of executing more tests, the tester tried to obtain information about the rest of the cluster from nodes tested fault free outside the cluster, such that the diagnosis of the system wouldn't delay. Each such alternative path to a cluster was called a *detour*. An extra test was executed on a given cluster only when no detour was available. Hi-ADSD with Detours was a practical algorithm that could be used to monitor real local area networks. Considering the number of tests required, the impact on network performance was lower than that of previous algorithms with the same latency. The worst case of the algorithm's latency was formally proved.

Qin et al. [56] described that the hierarchical routing protocols had been proposed to deal with the path search in wireless multi hop networks in so many different research works. Most of the existing designs of ad hoc network routing protocols are based on the assumption of non-adversarial environment, that every node in the network is cooperative and well behaved. Though, such assumption usually does not hold in realistic environments. The performance of current routing protocols degrades significantly when misbehaving nodes exist in the network. An efficient and effective hierarchical algorithm for MANET, which was called Fault-tolerance Cluster Head based (FTCH) routing protocol had been proposed to provide a certain packet delivery fraction guarantee and low routing overhead in the presence of faulty nodes. The FTCH Routing protocol was evaluated through both analysis and simulations compared with Max-Min Multi-Hop routing protocol (MMMH), AODV and DSR. The results showed that FTCH greatly improved the ad hoc routing performance in the presence of misbehaving nodes.

Liu and Payton [57] described that recent technological advancements had led to the popularity of mobile devices, which could dynamically form wireless networks. Unfortunately, mobile devices were vulnerable to failure because of many factors, like for example physical damage due to deployment in harsh environmental conditions, limited energy. Detecting node failure was an important problem that had been widely studied; recent attention had focused on determining failure when nodes were mobile. Detection of node failure required additional messages to be sent across the network, which was costly in terms of energy consumption. The authors contended that fault detection algorithms should be designed with consideration of the tradeoffs between cost and accuracy of fault detection. In this paper, they presented two approaches to dynamically adapting a fault detection algorithm. They compared their adaptive approaches to existing approaches and evaluated the tradeoffs between cost and accuracy. They used a cluster based probe-and-ack algorithm to illustrate how a) application specific requirements could be used to drive the adaptation of the rate at which failure detection probes were issued and b) how failure detection history could be used to drive adaptation of the interrogation period. The use of either of these approaches could result in the reduction of network load and message overhead, which could extend the lifetime of the network.

Yan and Lv [58] described that in modern computer networks, fault diagnosis has been a spotlight of research activity. This paper described the history of fault diagnosis in networks and discussed the main methods in information collecting section, information analyzing section, diagnosing section and at last the revolving section of fault diagnosis in networks. Emphasis was placed upon knowledge-based methods with discussing the advantages and shortcomings of the different methods. The survey was concluded with a description of some open problems. Modern fault Diagnosis methods in computer networks, focused on the contributions which they think close to the modern theory and might gain some relevance for the future research and practical applications. Fault diagnosis in networks had made great progress in common fault detecting and localization. Each method of fault diagnosis in networks relied on one or more theories, which determinate the application of method.

Vashist et al. [59] described that fault detection and localization is a well-studied problem in communication networks. The inherent variability, limited component reliability, and constrained resources of MANETs (Mobile Ad hoc Networks) make the problem not just more important, but also significant. Practical development imply that fault detection and localization methods must a) avoid relying on excessively detailed models of network protocols and traffic assumptions and instead rely on actual cross-layer observations, and b) be applicable across varying network scales and topologies with minimum adjustments. The authors proposed an important and as yet unexplored approach to fault management in MANETs: network-invariant fault detection, localization and diagnosis with limited knowledge of the underlying network and traffic models. They showed how fault management methods can be derived by observing statistical network/traffic measurements in one network, and afterward applied to other networks with satisfactory performance. The authors demonstrated that a carefully designed but widely applicable set of local and weak global indicators of faults can be efficiently aggregated to produce highly sensitive and specific methods that perform well when applied to MANETs with varying sizes, topologies, and traffic matrices.

Ziwich et al. [60] proposed generalized distributed comparison-based model assuming the comparison of faulty units outputs may match - *Hi-Dif* diagnosis algorithm that required at

Most $O(N^2)$ comparisons and had worst-case latency of $O(log_2 N)$ latency.

Further Ziwich et al. [61] presented a survey that integrated the vast amount of research efforts that have been produced in the field of fault diagnosis, from the earliest theoretical models to new promising applications. Key results also included the quantitative evaluation of a relevant reliability metric—the diagnosability—of several popular interconnection network topologies. This work presented a comprehensive and integrated view of comparison-based diagnosis results including models, algorithms, diagnosability bounds, and applications. In comparison-based system-level diagnosis tasks were assigned to and executed by pairs of units. The task outputs were returned and then compared. Depending on the comparison outcomes, units were classified as faulty or fault free. This survey described how the several models for comparison-based diagnosis differed, that is, in terms of assumptions, on how tasks were assigned, how outcomes were returned, where task outputs were compared, and how results were interpreted. Models either assumed that only the task execution is distributed, or, alternatively, that also comparisons and the diagnosis itself were distributed among the system units. Some models worked under probabilistic assumptions. The diagnosability of several popular interconnection network topologies under comparison-based models was also presented. The objective was to describe not only models but also algorithms in a way to help readers to understand each contribution and how it relates to the field as a whole. A range of applications have been described, including the detection of unauthorized modifications for replicated data, monitoring task outcomes in grid systems, and the diagnosis of mobile ad hoc networks. Besides integrating and clarifying comparison-based diagnosis results, the main objective of the survey was to ignite the potential of these models, methods, and technology, which could bring novel contributions to diverse fields. In security for instance, comparison-based diagnosis could be used for checking the integrity of data and Services

The literatures surveys are presented that have been done during the research work and the related work that has been proposed by so many different researchers. The research work related to fault and fault diagnosis from 1980 to 2012 has been shown which discussed about different methods and algorithms to diagnose the fault in the system.

Several approaches have been proposed for failure detection, including the heartbeat [14], probe and-ack [57], comparison strategies [42]. However, those approaches are not suitable when nodes are mobile. Faulty nodes cannot communicate with the other mobiles or behave unexpectedly and send unexpected results. In this way it unnecessarily cause inconsistency and consumes energy. So many different protocols presented by researchers to identify the fault in ad- hoc network were for static diagnosis, where node cannot change their status during diagnosis session. The fault (crash and value) identification in dynamic diagnosis is more complex than static diagnosis; during the diagnosis fault-free node can be faulty. After the deep study, we found that faulty node's presence affects the efficiency and throughput of the network, which makes the network inconsistent. Also the above approaches lack scalability and are not applicable to the large scale MANETs.

Problem Formulation

As we have discussed in the introduction chapter that MANETs have many types of faults like transmission error, node or link failure, route breakage and congested node or links. Previous work was done on finding the nodes that are faulty or fault- free in the wired network. In wired network the diagnosis was done on large networks of 64 nodes and 512 nodes. The network was integrated to an SNMP- based network management system on a 37- node Ethernet LAN. This network did not give the information about which type of faults have occurred and also didn't not tell that how much time nodes in network take to send or receive the messages .

In this dissertation, the clustering concept is used. The nodes store the information of all the nodes of cluster in the local diagnostic information, the complete information of all the nodes of network is stored in global diagnostic information. Use of heartbeat messages is shown which tells how the messages are sending or received among the nodes. The concept of timeout is used which is the maximum waiting time by the initiator nodes to diagnose faulty mobile nodes. The global diagnostic message stores the complete information of nodes of complete network. The threshold value tells how much percentage of nodes is faulty or fault-free.

4. Proposed Work

System and Fault Model

We assume that the wireless ad hoc network is a large connected network in which there are totally N sensor nodes denoted by 1, 2, 3… N. The nodes are distributed randomly in some physical domain and become stationary after deployment. The transmission range for each node is fixed and link between two hosts is bi-directional. If host u is in the transmission range of another host v, then there must be a link between the two. The system can be modeled as a communication graph G = {V, E}, where V= {1, 2….N}, and E= {(v1, v2): v1 is in transmission range of v2 and vice versa}.

A cluster is a unit disk with a radius equal to the center node's transmission range. As a result, any non-center nodes in a cluster are one-hop neighbors of the center node. The center node is called the cluster head (CH), while a node that is a one hop neighbor of the CHs of two different clusters can become the gateway (GW) node (see Figure 1). After the autonomous cluster formation, only CH and GW node, which are elected in a fully distributed fashion, participate in the inter-cluster communication (see Figure 1(b)), while ordinary members (OMs) in each cluster talk only to their CHs (and to other members when necessary).

The proposed system is not fully distributed. The total number of nodes is equally divided into a number of clusters. Each cluster has a CH and there is a GW node between two clusters to forward the message from one cluster to another. The cluster is controlled by the CH. The fault is detected by the CH in each cluster and the message is forwarded to all nodes of the cluster and also forwarded to other CH . All the clusters are operating simultaneously.

Intra-Cluster and Inter-Cluster Communication
In the fault detection of wireless sensor networks, we assume that all the sensor nodes have the same transmission range. Sensor nodes can be randomly deployed or placed in predetermined locations. Nodes with faulty sensors and permanent communication faults are to be identified. Sensor nodes which generate incorrect sensing data or fail in communication intermittently are treated as usable nodes, and thus are diagnosed as fault-free. Sensor nodes with malfunctioning sensors could participate in the network operation since they are still capable of routing information. Only those sensor nodes with a permanent fault in communication (including lack of power) are detected and this information is disseminated throughout the network and removed from the network.

Algorithm for cluster formation

This section describes the algorithm for cluster formation in the proposed system model. The algorithm is given in a table [4.1].

The system model uses an existing method FIND (Faulty Node Detection) to detect nodes with data faults [11]. After the nodes in a network detect a natural event, FIND ranks the nodes based on their sensing readings as well as their physical distances from the event. A node is considered faulty if there is a significant mismatch between the sensor data rank and the distance rank.

```
For any unselected node v
{
If ((node v is an indispensable node) || (node v is the only node with highest quality Qv
        among unselected neighbor) || (among unselected neighbor with same quality
        node v is with the smallest ID)
{
Update status to selected;
Regard itself as a CH;
Send an invite packet, invite (v) to all neighbors ;
}
On receiving an invite packet from neighboring node v
If (node u is an indispensable node)
Discard this packet;
Else
{
Regards itself as an ordinary node;
Updates status to selected;
Sends a join packet, join (u,v) to join the cluster constructed by v;
If (more than one such packets are received)
Join the one with smallest ID;
Else
Joins sender with largest logical degree;
Regards itself as a gateway node;
}
On receiving a join packet sent from neighboring node u decreases the logical degree
        by 1;
}
}
```

Table 4.1: Algorithm for cluster formation

Self-diagnosis Phase

When a set of sensor nodes is queried, each sensor in the queried set performs a self-diagnosis procedure to verify whether its current reading vector is faulty or not. Once the reading vector of a sensor node is determined as normal, the sensor node does not need to enter the neighbor-diagnosis phase. To execute a self-diagnosis, each sensor s_i only maintains two reading vectors: i) the current reading vector at the current time t (denoted as $b_i(t)$); and ii) the last correct reading vector at a previous time tp (expressed by $bi(tp)$). $b_i(tp)$ records a series of readings occurred in the previous time and is used for checking whether the current reading behavior is faulty or not. If these two reading vectors are not similar, $b_i(t)$ is viewed as an unusual reading vector. Once a sensor node is detected an unusual reading vector, this sensor node will enter the neighbor-diagnosis phase to further decide whether the unusual reading behavior is faulty or not. Note that when $b_i(t)$ is identified as a normal vector through the neighbor-diagnosis, $b_i(tp)$ is updated so as to affect the current monitoring state.

Clustering

The fault diagnosis algorithm coupled with suitable clustering concepts make a very efficient fault diagnosis service for wireless ad hoc networks. In MANETs, clustering [19] can be defined as a notional arrangement of the dynamic nodes into various groups. These virtual collections of nodes are grouped together regarding their relative transmission range proximity to each other that allows them to establish a bidirectional link. The diameter size of the clusters determines the control architectures as single-hop clustering and multi-hop (K-hop) clustering. In single-hop clustering every member node is never more than 1-hop from a central coordinator - the cluster head. Therefore all the member nodes remain at most two hops n distance away from each other within a logical cluster. In multi-hop clustering, the limitation or restriction of an immediate proximity to member nodes from the head is removed, allowing them to be present in serial k-hop distance to form a cluster.

Ordinary nodes (cluster member): As the name suggests, ordinary nodes do not perform any other function beyond a normal node role. They are members of an exclusive cluster independent of neighbours residing in a different cluster.

Cluster Gateway Nodes: Is a node that works as the common or distributed access point for two cluster heads .When a node remains within the transmission range of two cluster heads.

Cluster head nodes: for any efficient cluster (subsets of nodes in a network satisfying a particular property) operation there must be a support or backbone to sustain all necessary control functions such as bandwidth allocation, power control and virtual-circuit support channel access, routing, calculation of the routes for longer-distance messages, and forwarding inter-cluster packets,. This support or backbone takes the form of connected cluster heads, in managerial role; linked either directly or via gateway nodes and they will have the subordinate nodes of that cluster linked to them. Another function of cluster heads is internal node communication, to forward inter-cluster messages. To send a packet an ordinary node must first direct it to its ‘superior’ its directly connected cluster head.

ISSN 2277-3061

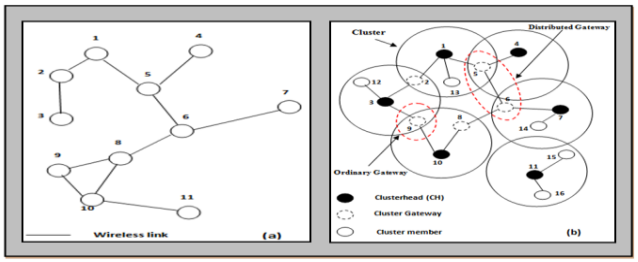

Figure2.8: Nodes in flat and cluster structure. (a) Flat structure. (b) Cluster structure [19]

Neighbor-diagnosis Phase

If a sensor node s_i sends b_i (t) to a neighbor s_j , s_j will compare b_i (t) with its own current adding vector b_j (t) and then give its vote with respect to *bi* (t). From the votes from neighbors, s_i has to determine whether *bi* (t) is faulty or not. Notice that some votes are from sensors with high Sensor Rank. A sensor node with high Sensor Rank has more similar neighbors to consult with and thus is more trust worthy. Therefore, the votes from the neighbors with high Sensor Rank are more authoritative, whereas the votes from the neighbors with low Sensor Rank should cast less weight.

When sensor s_i sends *bi* (t) to all its neighbors for the neighbor-diagnosis, each neighbor should return its vote after determining whether b_i (t) is faulty or not. If a neighbor *sj* considers b_i (t) is not faulty by comparing the similarity of the two reading vectors (i.e., $corri_{i,j}$), *sj* will send a positive vote, denoted $vote_j$ (i), to s_i. therwise, the vote will be negative. In addition, the vote from s_j will be weighted by its sensor Rank.

$$votej\ (i) \qquad rank_j\ , \quad corri_{i,j} \geq \sigma$$

$$rankj\ , \quad \text{otherwise.}$$

After collecting all the votes from the neighbors, s_i has two classes of votes: one is positive class (*bi* (t) is normal) and the other is negative class (b_i (t) is faulty). If the weight of the former is larger than the weight of the later, the most neighbors will view b_i (t) as normal. Note that the weight of a vote represents how authoritative a vote is. It is possible that a neighbor *sj* of *si* with a large Sensor Rank has a small correlation with s_i. In this case, these two sensor nodes may not provide good judgments for each other. Therefore, each vote (i.e., *votej* (i)) has to be multiplied by the corresponding correlation, $corri_{i,j}$. Thus, we use the following formula to determine whether the reading is faulty or not.

$$deci = \quad \Sigma\ corri_{i,j}.vote_j(i)$$

Purposed Algorithm

Input: a sensor s_i, Sensor Rank $rank_i$ and time interval t

Output: justify whether the reading is faulty or not (i.e. *faulty* = *true* or not)

1 set *faulty* = *false*

2 broadcast $rank_i$ to the neighbors

3 receive $rank_j$ from the neighbors

4 sort Sensor Rank values received

5 x = $rank_i$ order in the sorted Sensor Rank values

6 n = neighbors of sensor *si*

7 *timer* = x * (t/ n+1)

8 while *time* == *timer* do

9 *faulty* = Procedure *Self-Diagnosis*

10 if *faulty* == *true* then

11 *faulty* = Procedure *Neighbor-Diagnosis*

12 return *faulty*

5. Result and Analysis

We simulate a synthetic environment, where sensors are deployed in a 500 by 500 to monitor temperatures. The temperature reading range is [¡25; 275]. Moreover, events with unusual readings are randomly generated in the monitored ¯held. The models of generating events are the same as in [5, 6]. The faulty sensor rate (abbreviated as faulty rate) is the ratio of the number of faulty sensors and the total number of sensors deployed. Each sensor will report noisy readings according to the parameter *noise_ prob*. A faulty sensor always report faulty readings and thus *noise probe* is set to 1 for faulty sensors. On the other hand, a normal sensor is still likely to report noise or faulty readings. Thus, for normal sensors, we set the *noise_ prob* to 0:1. A noise reading (referred to as a faulty reading) is randomly biased from the normal reading generated and the amount of bias is within the range of [¡50; 50]. A query is submitted to wireless sensor networks with its query region as a rectangle and query region size varied from 80 by 80 up to 160 by 160. To evaluate the simulation result, two performance metrics are employed: *faulty detection rate* and *false positive rate*. Specially, a query is issued to a query region B to obtain the current readings sensed by the sensors, where the set of these current readings is

denoted as XB. Assume that Y_B is a set of faulty readings in X_B. After executing purposed algorithm, we can ¯filter out a set of faulty readings denoted as Y_B, and obtain a subset of current readings $X0_B \mu X_B$ without faulty readings.

5.1 Performance of Purposed Algorithm

First, we evaluate the performance of these three algorithms. The length of reading vectors for a sensor node is set to 5 and the similarity threshold is set to 0.5. For purposed algorithm, the number of iterations for calculating Sensor Rank is set to 3. Figure 4 shows the faulty detection rates of these three algorithms with various faulty rates. It can be seen that purposed algorthim can detect almost 90% faulty readings while Major Voting and Weight Voting can only identify 40% faulty readings. However, since faulty readings in our faulty model are biased from normal readings, it is hard to indented faulty readings for Major Voting and Weight Voting. By exploring Sensor Rank, purposed algorithm outperforms other two voting algorithms. Figure 5 shows the false positive rate of the three algorithms. As the faulty rate increases, false positive rates of three algorithms tend to increase due to a larger number of faulty sensors (i.e., it is hard to correctly detect faulty sensors when the number is large).

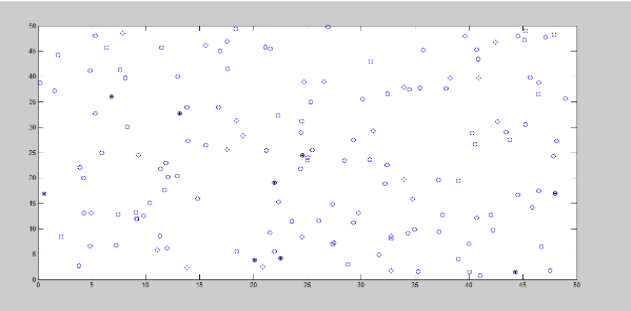

Fig5.1: Fault Detection Rate

5.2 Simulation Parameters

As mentioned before, Sensor Rank is calculated iteratively. We now examine the impact of the number of iterations (i.e., the parameter ±) to purposed algorithm. Specially, faulty rates are set to 0:4, 0:5 and 0:6. The length of reading behaviors is set to 5 and the similarity threshold is set to 0.5. The experimental results are shown in Figure 5.2 and Figure 5.3. It can be seen in Figure 5.2, when ± increases, the faulty detection rate will increase. This is due to that with a larger number of iterations; Sensor Rank is able to have more neighboring information. Therefore, Trust Voting is able to precisely identify faulty readings. Furthermore, with the number of iterations increases, the false positive rate declines. However, increasing the number of iterations for Sensor Rank will incur message transmissions among sensors. In addition, from Figure 5.2 and Figure 5.3, it can be seen that after 3 iterations, the improvements in the faulty detection rate and the false positive rate are not very sign cant. Therefore, in the following experiments, we set to number of iterations for Sensor Rank to be 3. Clearly, the number of iteration for Sensor Rank will be dependent upon the sensing data and can be empirically determined.

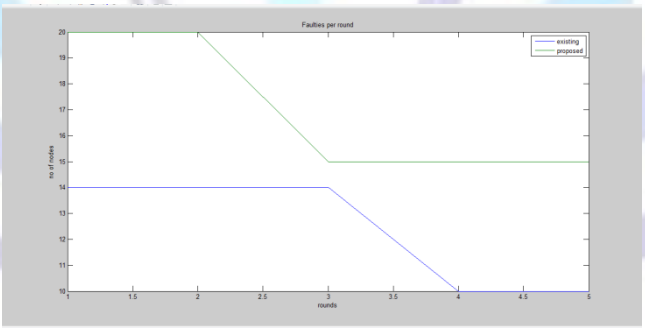

Fig 5.2: Faulty per round

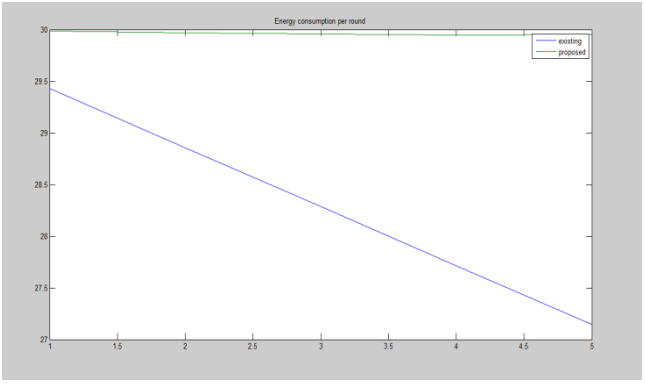

Fig 5.3: Energy Consumption per Round

6. Conclusion

With the presence of faulty readings, the accuracy of query results in wireless sensor networks may be greatly affected. In this thesis, we first formulated the correlation among readings of sensors nodes. Given correlations among sensor nodes, a correlation network is built to facilitate derivation of Sensor Rank for sensor nodes in the network. We have use clustering technique to make clusters. In each cluster we have a cluster head; the responsibility of cluster head is to determine the faulty node in the cluster. In light of Sensor Rank, an in-network algorithm Purposed algorithm is developed to determine faulty readings. Performance evaluation shows that by exploiting Sensor Rank, algorithm Trust Voting is able to efficiently identify faulty readings and outperforms majority voting and distance weighted voting, two state-of-the-art approaches for in-network faulty reading detection.

7. Future Work

Future work includes the following:

1) To make a trusted network or central network, by making the central network the burden of splitting the information to the number of nodes may reduce. The complete information is stored in this trusted network so that every cluster accesses the information from this trusted network, instead of storing the information of its own nodes in separate table. The complete information is stored at one place, all the nodes of the clusters access the information from this central network and may reduce the burden of nodes.

2) Taking a set of nodes that may be used only for diagnosis of the information, but not to test the nodes of a network or a cluster. Nodes may be tested only by their own network nodes or cluster head or it may be tested by initiator node. The set of nodes which store the diagnosis information should be used only for transferring information to the rest of the nodes in the network.
3) Making the clustering algorithm dynamically by the use of communication of nodes on the basis of ideal state of nodes or busy state of nodes. When the nodes will be in the ideal state the number of messages to be send and the information to be collected may be more than the busy state. When the nodes will be in the busy state the number messages to be sent and the information to be collected may be less.

8. References

[1]. Dr. Goel. A & Sharma. 334 “Performance Analysis of Mobile Ad-hoc Network Using AODV Protocol”. International Journal of Computer Science and Security (IJCSS), Volume (3): Issue (5), pp334-343.

[2]. Nayak. T.R. *et al.* “Implementation of adaptive zone routing protocol for wireless networks”. International Journal of Engineering Science and Technology Vol.2 (12), 2010, pp.7273-7288.

[3]. http://kbserver.netgear.com/kb_web_files/N100688.asp

[4]. IEEE LAN MAN Standards Committee, *Wireless LAN Medium Access Control Layer (PHY) Specifications, IEEE Std. 802.11a-1999*. The Institute of Electrical and Electronics Engineers, New York, Dec. 1999.

[5]. IEEE LAN MAN Standards Committee, *Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, IEEE Std. 802.11-1997*, The Institute of Electrical and Electronics Engineers, New York, New York, 1997.

[6]. http://www.wifinotes.com/fixed-wireless.html

[7]. Lin, T. “Mobile Ad Hoc Network Routing Protocols: Methodologies and Applications” (PhD thesis).

[8]. http://en.wikipedia.org/wiki/Ad_hoc

[9]. Bakshi, A.; Sharma, A.K. and Mishra, A. “Significance of Mobile AD-HOC Networks (MANETS)” International Journal of Innovative Technology and Exploring Engineering (IJITEE) ISSN: 2278-3075, Volume-2, Issue-4, March 2013.

[10]. Sun, J. Mobile Ad-Hoc Networking: “An Essential Technology for Pervasive Computing” appeared in Info-Tech and Info-Net, Vol.3, pp.316-321, Proceedings ICII 2001-Beijing.

[11]. Mueller S., Tsang R. and Ghosal D. “Multipath Routing in Mobile Ad Hoc Networks: Issues and Challenges” Performance Tools And applications To Networked Systems Lecture Notes in Computer Science,Vol.2965,pp.209-234,(2004).

[12]. Elhadef M., Boukerche A. and Elkadiki H. “A Dynamic Distributed Diagnosis Protocol For wireless and mobile Ad-Hoc Networks” *reviewed at the directions of IEEE Communications Society subject matter experts for publication in the IEEE GLOBECOM 2006 proceedings.*pp.165-172, (2006).

[13]. Sharma A. and Richariya V. “Intermittent Fault Diagnosis on WSNs Using an Energy Efficient Clustering Technique” International Journal of Emerging Technology and Advanced Engineering Website: www.ijetae.com (ISSN 2250-2459, Volume 1, pp.162-170, Issue 2, December 2011).

[14]. Elhadef, M. and Boukerche, A., “A Failure Detection Service for Large-Scale Dependable Wireless Ad-Hoc and Sensor Networks”, The Second international Conference on Availability Reliability, and Security, pp. 182-189, 2007.

[15]. Elhadef, M; Boukerche, A and Elkadiki, H. “Diagnosing mobile ad hoc networks: two distributed comparison-based self-diagnosis protocols,” in: Proceedings of the 4th ACM International Workshop on Mobility Management and Wireless Access Protocols, Terremolinos, Spain, October 2006.

[16]. Chessa,S. and Santi,P. “Comparison-Based System-Level Fault Diagnosis in Ad Hoc Networks.” In Proc. of the 20th Symp. on Reliable Distributed Systems, pages 257-266, LA, USA, 2001.

[17]. M. Elhadef, A. Boukerche, H. Elkadiki, A distributed fault identification protocol for wireless and mobile ad hoc networks. J. Parallel Distrib. Comput.vol. 68 (2008) pp.321 - 335.

[18]. Duarte E.P and Nanya T., "A Hierarchical Adaptive Distributed System-Level diagnosis algorithm", IEEE Transactions on computers, Vol. 47, issue. 1, pp. 34 - 45 January (1998).
[19]. Shayeb, I.G; Hussein, A.R.H; Nasoura, A.B "A Survey of Clustering Schemes for Mobile Ad-Hoc Network (MANET) American Journal of Scientific Research .ISSN 1450-223X Issue 20(2011), pp.135-151.
[20]. Malek,M. "A comparison connection assignment for diagnosis of multiprocessor Systems." In *Proceedings of the 7th Annual International Symposium on Computer Architecture,* pp.31–36, 1980.
[21]. Chwa, K. Y. and Hakimi, S. L. "Schemes for fault-tolerant computing: A comparison of modularly redundant and t-diagnosable systems."*Inf. Control 49*, pp. 212–238, 1981.
[22]. Maeng, J. and Malek, M. "A comparison connection assignment for self-diagnosis of multiprocessor systems." In *Proceedings of the 11th IEEE Fault-Tolerant Computing Symposium*. Pp.173–175, 1981.
[23]. Ammann, E. and Dal Cin, M."Efficient algorithms for comparison-based self diagnosis" *Self-Diagnosis and Fault Tolerance, Werkhefte der Universitat Ttibingen, 4 Attempto-Verlag, Tubingen, vol. 1, issue.18*, 1981.
[24]. Sallay, B.; Maestrini, P., and Santi, P." Wafer-scale diagnosis tolerating comparator faults. *IEE Proc. Comput. Digital Techn." Vol.146, issue.* 4, pp.211–215, 1999.
[25]. Pelc, A. "Optimal fault diagnosis in comparison models." *IEEE Trans. Computer Vol.41, issue.*6, pp. 779–786, 1992.
[26]. Chwa, K. Y. and Hakimi, S. L. "On fault identification in diagnosable systems." *IEEE Trans. Comput.vol. C-36, issue.*6, pp.414–422, 1981.
[27]. Barborak, M.; Dahbura, A., and Malek, M. "The consensus problem in fault-tolerant computing." *ACM Computer Survey Vol.25, issue.*2, pp.171–220, 1993.
[28]. Kozlowski, W. E. and Krawczyk, H. "A comparison-based approach in multicomputer system diagnosis in hybrid fault situations." *IEEE Trans. Computer Vol.40, issue .*11, pp. 1283–1286. 1991.
[29]. Chen, Y.; Bucken, W. and Echtle, K." Efficient algorithms for system diagnosis with both processor and comparator faults." *IEEE Trans. Parallel Distributed System. Vol.4, issue.* 4, pp.371–381, 1993.
[30]. Dahbura, A. T.; Sabnani, K. K., and King, L. L. "The comparison approach to multiprocessor fault diagnosis." *IEEE Trans. Comput. Vol. -36, issue.*3, pp.373–378, 1987.
[31]. Sengupta, A. and Dahbura, A. T. "On self-diagnosable multiprocessor systems: Diagnosis by comparison approach." *IEEE Trans. Comput. Vol.41, issue.*11, pp.1386–1396, 1992.
[32]. Blough, D. M. and Brown, H. W. "The broadcast comparison model for on-line fault diagnosis in multicomputer systems: Theory and implementation." *IEEE Trans. Comput.vol. 48*, pp.470–493, 1999.
[33]. Yang, X. and Tang, Y. Y. "Efficient fault identification of diagnosable systems under the comparison model". *IEEE Trans. Comput. Vol.56, issue.*12, pp.1612–1618, 2007.
[34]. Nakajima, K. "A new approach to system diagnosis." In *Proceedings of the 19th Allerton Conference on Communication, Control and Computing*, pp.697–706, 1981.
[35]. Hakimi, S. L. and Nakajima, K. "On adaptive system diagnosis." *IEEE Trans. Comput.vol. C-33, issue.* 3, pp.234–240, 1984.
[36]. Kuhl, J. G .and Reddy, S.M. "Distributed fault-tolerance for large multiprocessor systems." In *Proceedings of the 7th Annual International Symposium on Computer Architecture*, Pp.23–30, 1980.
[37]. Kuhl, J. G. and Reddy, S. M. "Fault-diagnosis in fully distributed systems." In *Proceedings of the 11th IEEE Fault-Tolerant Computing Symposium*, pp.100– 105, 1981.
[38]. Hosseini, S. H., Kuhl, J. G., and Reddy, S. M." A diagnosis algorithm for distribute computing systems with dynamic failure and repair. " *IEEE Trans. Comput, Vol. C-33, issue.* 3, pp.223–233, 1984.
[39]. Bianchini, R. P.; Goodwin, K., and Nydick,D. S. " Practical application and implementation of system-level diagnosis theory." In *Proceedings of the 16th IEEE Fault-Tolerance Computing Symposium*, pp.332–339, 1990.
[40]. Bianchini, R. P. and Buskens, R. "An adaptive distributed system-level diagnosis algorithm and its implementation." In *Proceedings of the 21th IEEE Fault-Tolerance Computing Symposium*. 1991.
[41]. Bianchini, R. P. and Buskens, R. "Implementation of on-line distributed system- level diagnosis theory." *IEEE Trans. Comput. Vol.41*, pp.616–626, 1992.
[42]. Chessa, S. and Santi,P. "Comparison-Based System-Level Fault Diagnosis in Ad Hoc Networks" appeared in reliable distributed Systems,2001.procedings.20th IEEE symposium, pp.257-266, 2001.
[43]. Elhadef, M. and Boukerche, A., "A Failure Detection Service for Large-Scale Dependable Wireless Ad-Hoc and Sensor Networks", The Second international Conference on Availability Reliability, and Security, pp. 182-189, 2007.
[44]. Elhadef, M., Boukerche, A., And Elkadiki, H." Self-diagnosing wireless mesh and adhoc networks using an adaptable comparison-based approach."In *Proceedings of the 2nd International Conference on Availability, Reliability and Security*, pp.983–990,2007.
[45]. Elhadef, M., Boukerche, A., And Elkadiki, H. 2006a. Diagnosing mobile ad hoc networks: Two distributed comparison-based self-diagnosis protocols. In *Proceedings of the 4th ACM International Workshop on Mobility Management and Wireless Access,* pp18–27, 2006.
[46]. Elhadef, M., Boukerche, A., And Elkadiki, H." Performance analysis of a distributed comparison based self-diagnosis protocol for wireless ad hoc networks." In *Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems*. 165–172, 2006.
[47]. Xu , S.; Papavassiliou,S. and Zakrevski,L." Fault tolerant cluster-based approach in wireless mobile ad hoc networks" *supported in part by New Jersey Institute of technology under Grant no.421050and in part by the New jersey Commission on Higher Education.,*pp.2613-2617,2001.
[48]. Duarte E.P and Nanya T., "A Hierarchical Adaptive Distributed System-Level diagnosis algorithm", IEEE Transactions on computers, Vol. 47, no. 1,pp. 34 - 45January (1998).
[49]. Moallemi, M.; Mghaddam, M .H.Y. and Naghibzadeh. , N. "A Fault-Tolerant Mutual Exclusion Resource Reservation Protocol for Clustered Mobile Ad hoc Networks." Eighth ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing., pp.528-533,Vol.2, 2007.
[50]. Khilar, P.M; Singh, J .K. and Mahapatra, S. "Design and Evaluation of a Failure Detection Algorithm for Large Scale Ad Hoc Networks Using Cluster Based Approach", in proceeding of IEEE International Conference on Information Technology, Bhubaneswar, India, pp.153-158 (2008).

[51]. Rangarajan, S.; Dahbura, A.T. and Ziegler, E. A. "A Distributed System-Level Diagnosis Algorithm for Arbitrary Network Topologies" IEEE Transactions on Computers, Vol. 44, issue. 2, pp.312-334, 1995.
[52]. Roger, B.; Dumas, M. and Kurul, M. E. "Adaptive Fault Tolerance in Distributed Systems" Department of Computer Science University of California, San Diego La Jolla, pp.1-10, March 2001.
[53]. Vanaja, K. and Umarani, R." An Adaptive Fault Tolerant Multipath Routing (AFTMR) Protocol for Wireless Ad Hoc Networks" European Journal of Scientific Research ISSN 1450-216X Vol.79, No. 2, pp.180-190, 2012.
[54]. Albini, L. C. P.; Duarte JR., E. P., And Ziwich, R. P. "A generalized model for distributed comparison-based system-level diagnosis." *J. Brazil. Comput. Soc. Vol.10, issue.*3, pp.44–56, 2005.
[55]. Duarte, E.P.; Albini, C.P.; Brawerman, A. and Guedes, A.L.P. "A Hierarquical Distributed Fault Diagnosis Algorithm Based on Clusters with Detours". Appeared in network operations and management symposium 2009 pp.1-6, 2009.
[56]. Qin, Y. and Pang, K. L. "A Fault-tolerance Cluster Head Based Routing Protocol for Ad Hoc Networks." Appeared in vehicular technology conference, 2008.VTC spring 2008, IEEE, pp-2472-2476.
[57]. Liu, D. and Payton, J." Adaptive Fault Detection Approaches for Dynamic Mobile Networks" The 8th Annual IEEE Consumer Communications and Networking Conference - Smart Spaces and Personal Area Networks, pp.735-739, (2011).
[58]. Yang, Z.; Wang, Y. and Lv, J. "Survey of Modern Fault Diagnosis Methods in Networks. 2012 International Conference on Systems and Informatics (ICSAI 2012), pp.1640-1643, 2012.
[59]. Vashist, A.; Izmailov, R.; Manousakis, K; Chadha, R. Chiang, C.J. and Serban, C. "Towards Dependable Networks of Mobile Arbitrary Devices"- Diagnosis and Scalability. In *Future Directions in Distributed Computing*, Lecture Notes in Computer Science, pages 191–196. Springer, 2012.
[60]. Ziwich, R. P., Duarte JR., E. P., and Albini, L. C. P. "Distributed integrity checking for system with replicated data." In *Proceedings of the 11th IEEE International Conference on Parallel and Distributed Systems*, pp. 363–369, 2005.
[61]. Ziwich, R. P., Duarte JR., E. P., and Albini, L. C. P. "A Survey of Comparison-Based System-Level Diagnosis", Federal University of Parana ACM Computing Surveys, Vol. 43, No. 3, Article 22, Publication date: April 2011.
[62]. Available at http://cimss.ssec.wisc.edu/wxwise/class/aos340/spr00/whatismatlab.htm
[63]. http://en.wikipedia.org/wiki/MATLAB
[64]. http://www.mccormick.northwestern.edu/docs/efirst/matlab.pdfmtlb