ISSN No.: 2454- 2024 (online)

International Journal of Technical Research & Science

Cluster Based Diagnosis Algorithm using Wireless Sensor Network

Er. Anu Devi[1], Er. Meenakshi Sharma[2]
Email: anu.kappor14@gmail.com
GRIMT Radaur, Yamunanagar, Haryana, India

Abstract-The purpose of fault diagnosis in mobile ad hoc network is to have each fault-free node to determine the state of all nodes in the system. In the dissertation work proposes a fault diagnosis algorithm based on the approach for diagnosing nodes in mobile ad hoc network. The proposed diagnosis algorithm is linearly scalable under the assumption that the mobiles may be: (i) crash faulty due to out of range or physical damage and (ii) value faulty due to sending erroneous messages while operating in the field. The parameters such as diagnostic latency and message complexity are used for evaluating the proposed diagnosis algorithm. The purpose of distributed system-level diagnosis is to have each fault-free node determine the state of all nodes of the system. This paper presents a Cluster Based Diagnosis algorithm, which is a fully distributed algorithm that allows every fault-free node to achieve diagnosis in, at most, ($\log_2$ N)2 testing rounds. Nodes are mapped into progressively larger logical clusters, so that tests are run in a hierarchical fashion.

Index Terms – LEACH, WSN, MANET

1. INTRODUCTION

1.1 Wireless sensor networks (WSN)

A wireless sensor network (WSN) is typically designed to extent in a large field for data collection. Data delivery is usually reached with multi-hop transmission through a sequence of nodes. Most of the multi-hop routing protocols have been implemented in sensor networks for data gathering and they usually skilled with special path estimation metrics to select "good" paths for data packets delivery[7].

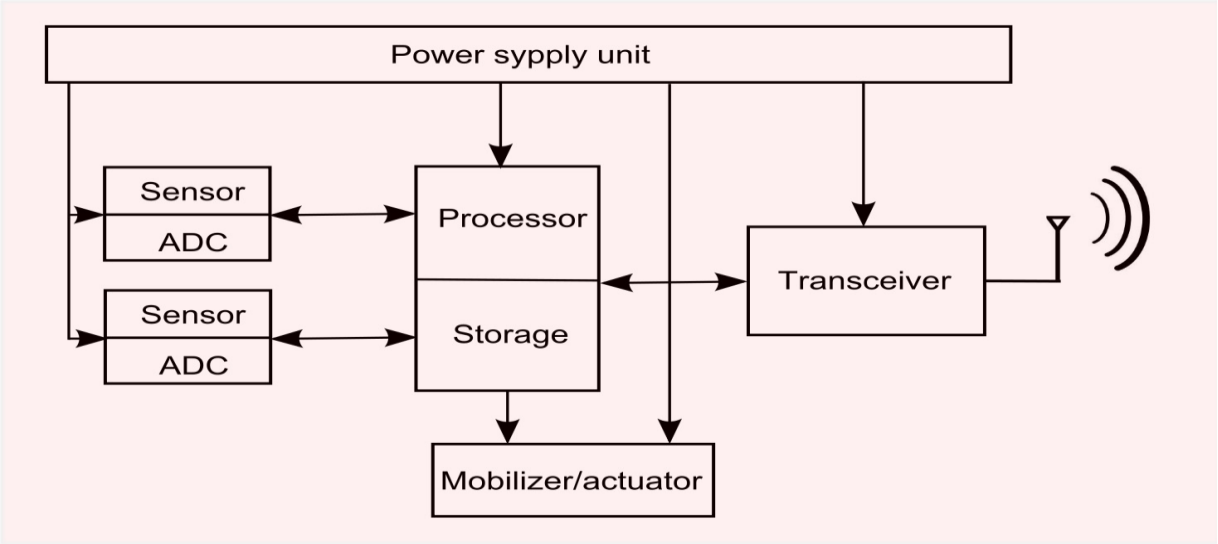


Fig. 1.1 Typical Sensor Node[7]

A sensor node is composed of four major blocks: sensing unit, processing unit, power unit and communication unit. The sensing unit is a sensor that measures a certain physical condition like temperature and pressure. The processing unit is responsible for collecting and processing signals captured from sensors. The wireless communication unit transfers signals from the sensor to the user through the base station (BS). All previous units are maintained by the power unit to supply the required energy in order to perform the mentioned tasks. Typically, WSNs hold hundreds or thousands of sensor nodes, and these sensors have the ability to communicate either among each other or directly to the BS[2].

Variety of applications must need the fault detection to be accompanied in a real-time mode with low latency or high throughput. So, a localized and distributed generic algorithm for each node is highly desired in wireless sensor networks[7].Basically, sensor networks are defined by the combination of miniaturized sensors with communication technology[8].

1.2 MANET

Ad hoc is a Latin Phrase meaning "for this". It mainly signify a solution intended for a specific problem or task, not intended to be able to be adapted to other purposes , non-generalizable[4]. Common examples are organizations, committees, and commissions formed at the national or international level for a particular task. The term ad hoc networking typically refers to a system of network elements that combine to form a network requiring little or no planning.The increasing use wireless portable devices such as phones and laptops is leading to the possibility for spontaneous or ad hoc wireless communication known as Mobile Ad Hoc Networks (MANET)[6].

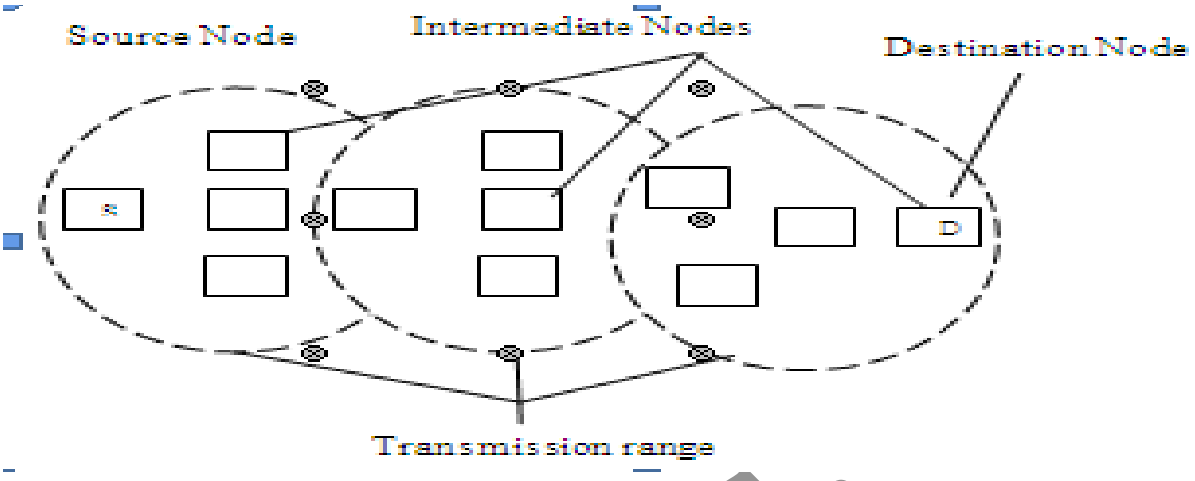


Fig 1.2 MANET[21]

The standard packet format for MANET is RFC5444 which consists of the 54 octets for a single message. An additional octet is proposed and introduced which consists of 6 bits for confidentiality. Among the 2 bits for recovery information 1bit flag is to mention to recover a data or not and the other to search for an alternate recovery manager. This octet is added with RFC5444 standard to provide trust and recoverability[5].

1.3 Clustering

Sensor nodes are randomly distributed within the goal area and form a cluster-based network by Low-energy Adaptive Clustering Hierarchy (LEACH) a protocol architecture for micro sensor networks. The gateway nodes which are responsible for the message exchange between clusters are elected from the overlap-nodes. Each node is assigned with an ID number for identification. After the formation of the network, cluster-heads broadcast a message containing their own ID numbers to their cluster member nodes to notice them the identity of their cluster heads. Cluster member nodes receive the messages and record the ID numbers. Nodes recording only one ID number mark themselves as the ordinary cluster member nodes, whereas nodes at the overlapping regions record more than one ID numbers and exhibit themselves as the overlap-nodes[1].

After receiving the message from the cluster-head, cluster member nodes respond with a message containing their own ID numbers and the ID numbers of their cluster-heads which they can communicate with. Cluster-heads calculate the average values of the skew compensation parameters of intra-cluster effective clocks of nodes within their clusters and the average values of intra-cluster virtual clocks of nodes, and then they update the clock compensation parameters of intra-cluster virtual clocks and simultaneously put out them to the neighboring nodes[1].

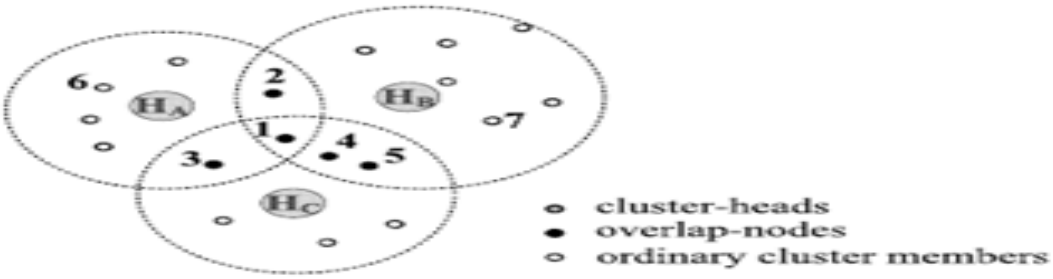


Fig. 1.3 Schematic illustration of the Local Topology of a Network[1].

2. PRESENT WORK

2.1 Sensor Rank

Sensor Rank is to represent the trustworthiness of sensor nodes. By our design, two requirements need to be met in deriving Sensor Rank for each sensor.

Requirement 1: If a sensor has a large number of neighbors with correlated readings, the opinion of this sensor is trustworthy and thus its vote deserves more weight.

Requirement 2: A sensor node with a lot of trustworthy neighbors is also trustworthy.

These two requirements ensure that 1) a sensor node which has a large number of similar neighbors to have a high rank; and 2) a sensor node which has a large number of good references to have a high rank. Given a correlation network $G = (V; E)$ derived previously, we determine Sensor Rank for each sensor to meet the above two requirements[8].

Consider an example in Figure 13 In the 1st round,*s*3 has some similarity information from its 1st level neighbors {s2; s4; s9; s10; s11}. Similarly, both *s*2 and *s*4 could ex- change some information with their neighbors. In the second round, *s*3 can obtain similarity information from the second level neighbors {s1; s5g}since its 1st level neighborss2 and *s*4 have explored *s*1 and *s*5 during the 1st round. If *k* is larger, Sensor Ranks will be more accurate since every sensor can explore more neighbors. In sensor networks, the computation cost will be larger when the number of iterations is larger. Therefore, we can limit *k* to a preset bound £. Given a correlation[8].

Table-2.1 Sensor Rank Values

	S1	**S2**	**S3**	**S4**	**S5**	**S6**	**S7**	**S8**	**S9**	**S10**	**S11**
K=0	1	1	1	1	1	1	1	1	1	1	1
K=1	1.13	0.59	1.74	1.11	1.33	0.64	1.14	0.89	0.58	1.3	0.54
K=2	1.17	0.68	1.43	1.05	1.24	0.77	0.91	1.05	0.86	1.04	0.8

2.2 Algorithm 1 Trust Voting

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

Output: Justify whether the reading is faulty or not

1: Set Faculty=False
2: Broadcast $rank_i$ to the neighbor
3: Receive{rankj |sj Є nei(i)} from the neighbors
4: Sort Sensor Rank values received
5: x = ranki's order in the sorted Sensor Rank values
6: n = neighbors of sensor s_i
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

network in Figure 1, we now demonstrate how to calculate Sensor Rank. Initially, sensor s1 sets. Its sensor Rank $rank_i^{(0)}$ to 1. For sensor si, sicalculates the trust relations pi,jto the corresponding neighbor sjand sends $rank_i^{(0)}$. pi,jto sj. For example, s3 sends $rank_3^{(0)}$p3,1 = 1 . 0.5 /3.0 = 0.167 to s1, 0.033 to s2, 0.2/ 3 = 0.067 to s4, and etc. At the same time, s3 receives Sensor Ranks from its neighbors. For example, s3 receives $rank_2^{(0)}$= p2,3 =1.0.1/0.4+0.1+0.7 = 0.083 from s2. Upon receiving all the proportion of Sensor Rank from the neighbors, s3 can update its Sensor Rank to $rank_3^{(1)}$.

$$rank_3^{(1)} = \sum rank_i^{(0)} . P_{j,i}$$

$$= 1.p_{1,3}+1 . p_{2,3} + 1.p_{4,3} +1.p_{9,3+1}.p_{10,3+1}.p_{11,3}$$

$$= 0.5/2.1 + 0.1/1.2 +0.2/1.9 +0.7/1.5 +0.8/2.3 +0.7/1/4$$

$$= 1.74$$

After the 1st round, { $rank_i(1)$ |i= 1, 2, 3, 4 } ={1.13, 0.59, 1.11, 1.33}. In the second round, sensors calculate the values of Sensor Rank with the updated values of Sensor Rank in the 1st round. For example, s1 now sends $rank_i^{(1)}$.p1,3 = 1.13 *0.5 /2.1 = 0.269 to s3. Similarly, when s3 receives all the values from its neighbors, s3 can update its Sensor Rank to $rank_3^{(2)}$[8] .

Assume that £ = 2, s1 will stop updating its Sensor Rank, and {$rank_i^{(2)}$ /i = 1,2, 3, 4 } ={1.17, 0.68, 1.43, 1.05}. As expected, s3 has the highest Sensor Rank 1.43, since s3 has many similar neighbors. Since s1 has fewer similar neighbors than s3, s1 has smaller Sensor Rank than s3. The values of Sensor Rank after the third iteration

are listed in Table 1. From Table 1, s3 has the largest Sensor Rank since more nearby sensors have similar reading behaviors with s3. This meets the requirements we set for design of Sensor Rank as mentioned earlier[8].

2.2. Trustvoting Algorithm

Here we describe our design of the TrustVoting algorithm, which consists of two phases: a) self diagnosis; and b) neigh- bors diagnosis phase. In the self-diagnosis phase, each sensor veries whether the current reading of a sensor is unusual or not. Once the reading of a sensor goes through the self diagnosis phase, this sensor can directly report the reading. Otherwise, the sensor node consults with its neighbors to further validate whether the current reading is faulty or not. If a reading is determined as faulty, it will be altered out. The sensor nodes generating faulty readings will not participate in voting since these sensors are likely to contaminate the voting result. Note that TrustVoting is an in-network algorithm which is executed in a distributed manner. The execution order of algorithm TrustVoting has an impact on faulty reading detection [8].

2.2.1 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 sionly 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 b_i(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[8].

2.2.2 Algorithm 2: Trust Voting Neighbor Diagnosis

Input: a sensor s_i,its current reading behaviour b_i (t), and a threshold ϭ

Output : The variable faulty

1: set $dec_i = 0$
2: broadcast $b_i(t)$ to the neighbors
3: **for all** s_j€ nei(i) **do**
4: **if** sim (bi (t) ,bj (t)) ≥ ϭ **then**
5: $Vote_j$ (i) = $rank_j$
6: **else**
7: Vote j (i) = - $rank_j$
8: $dec_i = dec_i + tr_{ij} * vote_j(i)$
9**: if** $dec_i \geq 0$ **then**
10: return false
11: **else**
12: return false

Table-2.2 Faulty Detection Under Different Orders

Order	Faculty	Not Faculty
S1,S2,S3,S1,S5	S5	S1,S2,S3,1
S5,S1,S2,S3,S4	S1,S5	S1,S2,S3

From Table-2.2, not all faulty readings reported by faulty sensors (i.e., *s*2, *s*4 and *s*5) are detected and difference executions orders have an impact on the faulty rading detection. As such, how to determine an appropriate order to perform self-diagnosis and neighbor-diagnosis in algorithm Trust Voting will have an impact on the final result. Since algorithm Trust Voting is executed in a distributed manner, we could use a timer to control the execution order of procedures self-diagnosis and neighbor-diagnosis [8].

2.3 CBD Algorithm

CBD maps nodes to cluster, which are set of nodes and employs a divide and conquer testing strategy to permit nodes to independently achieve consistent diagnosis. In this, nodes are grouped into clusters for the purpose of testing. The number of nodes in a cluster, its size, is always a power of 2 and system itself is a cluster of N nodes[6].

A hierarchical approach to test cluster is shown. In the first testing interval, each node performs tests on node of a cluster that has one node. In the second testing interval, on nodes of cluster that has two nodes, in the third testing interval, on nodes of cluster that has four nodes and so on, until the cluster of 2logN-1 nodes is tested. After that, the cluster of size one is tested again and the process is repeated until all the nodes are tested by every other node in the network. For the system in Fig.3, for all i and s, Ci,s is listed in Table 1[6]

2.3.1 The proposed diagnosis algorithm

In proposed diagnosis algorithm, an initiation heartbeat message goes from each initiator node simultaneously to a node of a cluster of size Ci, s of one and waits for a time out period of Tout. If the node in this cluster is fault free, the initiator node will receive a response heartbeat message from this fault free node and collects the diagnosis information about the entire network from this node. If the node in the cluster is faulty, the initiator node either will not receive a response heartbeat message (crash fault) or it may receive an erroneous message (value fault)[6].

2.3.2 Algorithm 3 CBD

Step 1: Create a cluster by u computing the formula $C_{1,s}$having N number of nodes

For all i=0,1,2,…….,N-1

S=1,2,……,log N

Step 2: Let us Assume that all the nodes in the network can initiate the diagnosis and they all are faulty free at the time of initiation

Step 3: Start Diagnosis

Repeat

For s=1 to logN Do

Send i_hb(I,j,Dj,init_hb_msg)

Set _timeout(T_{out})

Step 4: response r_hb(j,I,Dj,res_hb_msg)

If Dj=D'_j

status _Table[i]=fault free

ff=ff U[j]

else

diagnosed as faulty

f=N(initnode_id)-ff

if(f=N(initnode_id))then

complete

terminate=true

Step 5: Timeout

Step 6: Receive_local_diagmsg(I,fi)

F=f U f_i

D=D U {i}

D=N(init_node_id)-f

Step 7: Now all initator node will exchange local diagnostic message with each other and send it to every other node in the network.

Irrespective of crash or value fault, the initiator will detect this fault maximum within Tout. The initiator then sends another initiation heartbeat message to another node in the cluster of size 2 and repeats same process. In the worst case, an initiator node has to send an initiation heartbeat message to all the nodes of all the clusters of size Ci, s consisting of only faulty nodes. Thus, the total time elapsed to test every node in every cluster for a network size of N by an initiator node is ($\log_2$N. Ci, s) Tout [6].

3. RESULT

3.1 Performance of TrustVoting Algorithm

The length of reading vectors for a sensor node is set to 10 and the similarity threshold is set to 0.6. For Trust Voting, the number of iterations for calculating Sensor Rank is set to 5 for 150 nodes. Fig. 3.1 shows the faulty nodes, and Fig.3.2 shows the Number of nodes which are faulty.

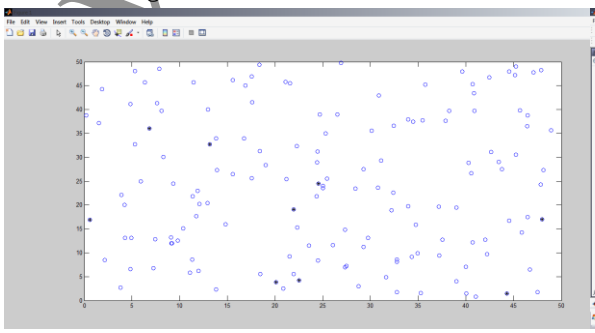


Fig 3.1Faulty Nodes in WSN

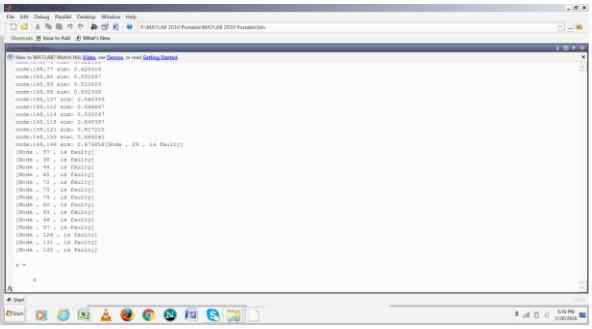


Fig 3.2 Number of Nodes which are Faulty In WSN

Fig. 3.4 shows the faulty per round in purposed algorithm. Fig. 3.3 shows that the comparison between existing and purposed algorithm Faulty per round in WSN .In Fig. 3.5 shows comparison between existing and purposed algorithm Energy Consumption Per round in WSN.

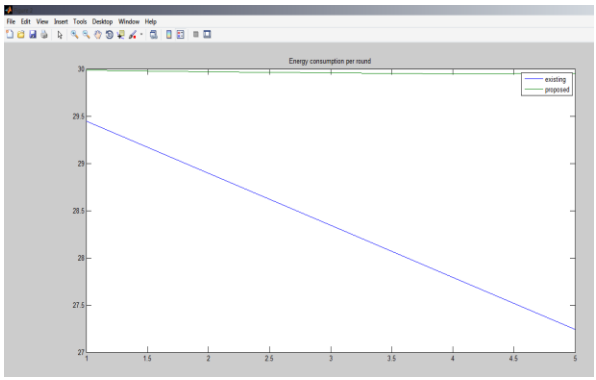


Fig 3.3Comparison Between Existing and Purposed Algorithm Faulty Per Round in WSN

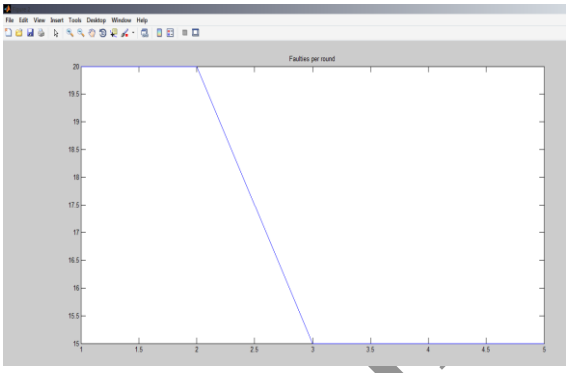


Fig 3.4 Faulty Per Round in Purposed Algorithm in WSN

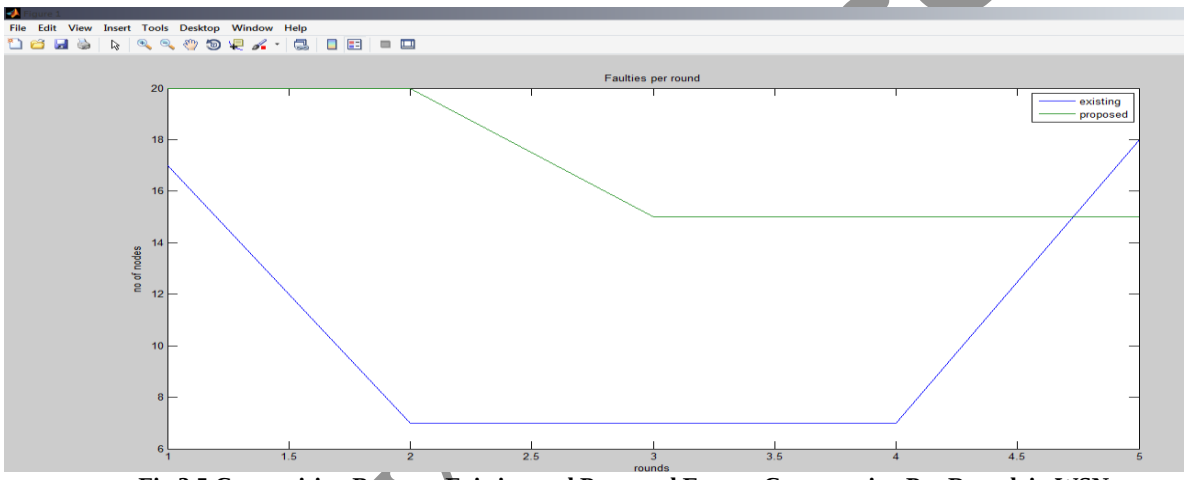


Fig 3.5 Comparision Between Existing and Purposed Energy Consumption Per Round in WSN.

3.2 Performance Evaluation in MANET

3.2.1 Simulation Model

A simulator is designed in MATLAB language where we present experimental results of diagnosis on large network using Cluster Based Diagnosis Algorithm, obtain through exhaustive simulation. The experiments were conducted for the network of varying sizes of 8, 16,32, 64,128 nodes. Tests were scheduled for each node at each 30 ± σ units of time, where σ is 6 random number between 0 and 3. During each test,the status of nodes are tested and if the node is fault free, diagnosis information regarding the cluster is copied to testing node. If the tested node is faulty,The testing nodes continue testing as in the algorithm. The parameters from diagnosis literature area assumed for executing the diagnosis tasks, send initiation time and propagation time of the messages in the MANET. The values of these parameters are given in the following Table .Table Values of different parameters used in the simulation.The parameters to evaluate the diagnosis algorithm are given in the following section.

3.2.2 Simulation parameters

There are three different parameters are used in the literature. These parameters are usually used to evaluate the proposed fault diagnosis algorithm.

- **Diagnostic latency**: It is the time elapsed by the initiator node to determine the status of the node in the network.
- **Message complexity**: It is the number of messages exchanged among nodes in the network to determine the status of nodes.

- **Hop count ratio**: It is the ratio of the euclidian distance between the source and destination node to the number of nodes in between the source and destination node.

3.2.3 Result

The length of reading vectors for a sensor node is set to 8 and the similarity threshold is set to 0.6 .Figure 6 shows faulty nodes in Manet.In figure 5.7 shows Diagnostic latency Vs. Network size. Figure 5.8 shows Message complexity Vs. Network size and Figure5.9 shows Hop-count Vs. Network size.

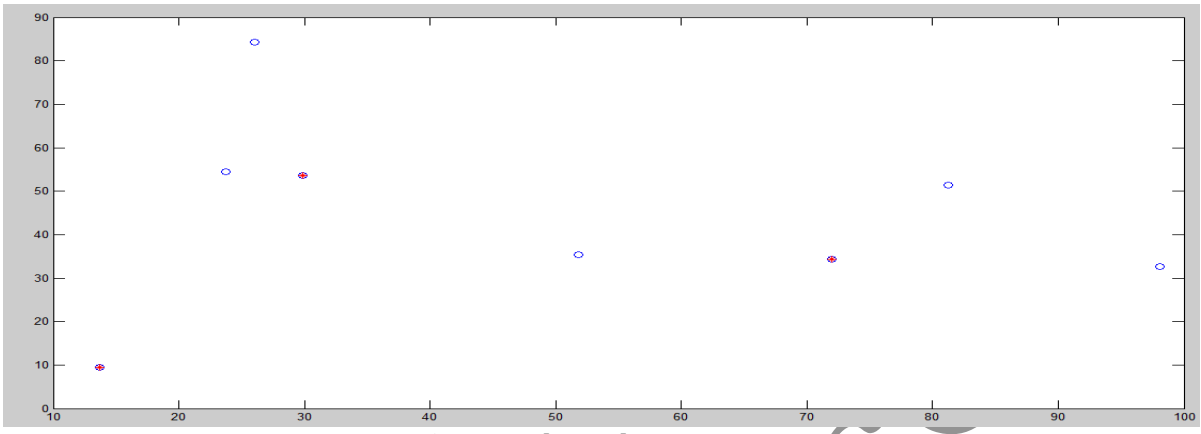
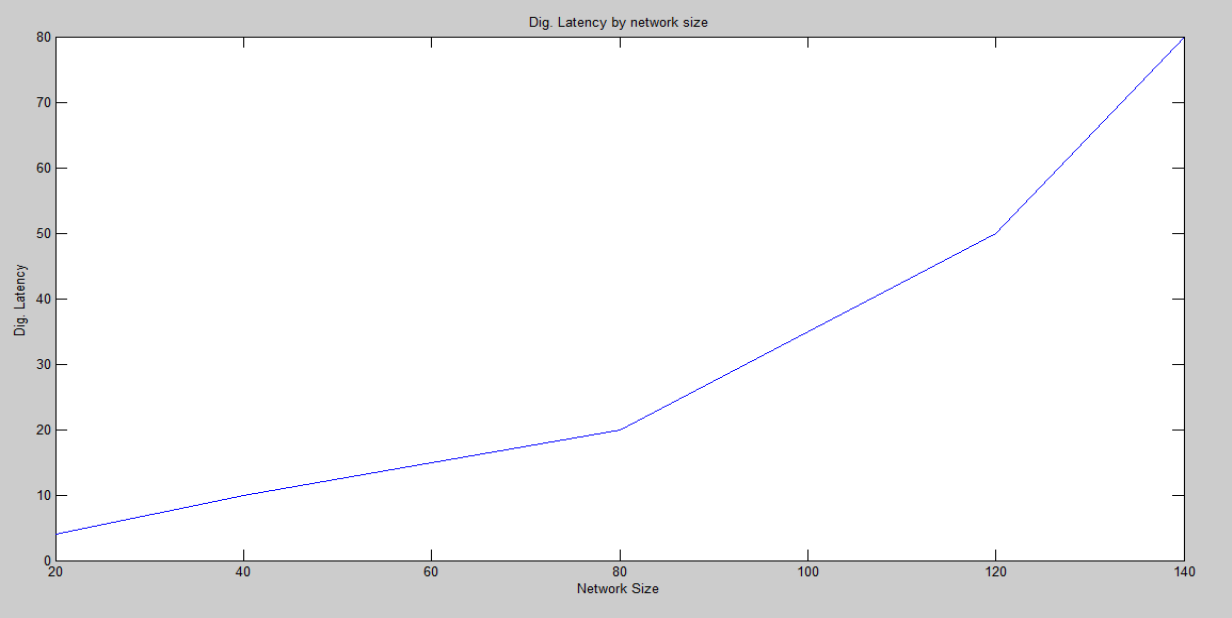


Fig 3.6 Faulty Nodes In MANET

3.2.3.1 Diagnostic latency Vs. Network Size

Fig. 5.7 compares the diagnostic latency for the proposed algorithm and Cluster Based algorithm. As the network size increases, the diagnostic latency for both proposed algorithm and Cluster Based algorithm increases. This shows that the proposed algorithm is fit for large MANETs deployed in hostile and harsh environments. It can be observed that the diagnostic latency depends on number of messages exchanged and the network parameters such as transmission and propagation delay and directly proportional to the number of messages exchanged in the network to achieve diagnosis.



.

Fig3.7 Diagnostic Latency Vs Network Size

3.2.3.2 Message complexity Vs. Network size

Fig. 3.8 shows the number of messages exchanged during the execution of proposed fault diagnosis algorithm. Message complexity i.e. total number of messages exchanged increases linearly with the number of nodes and found to be O (N. Ci,s). Whereas the

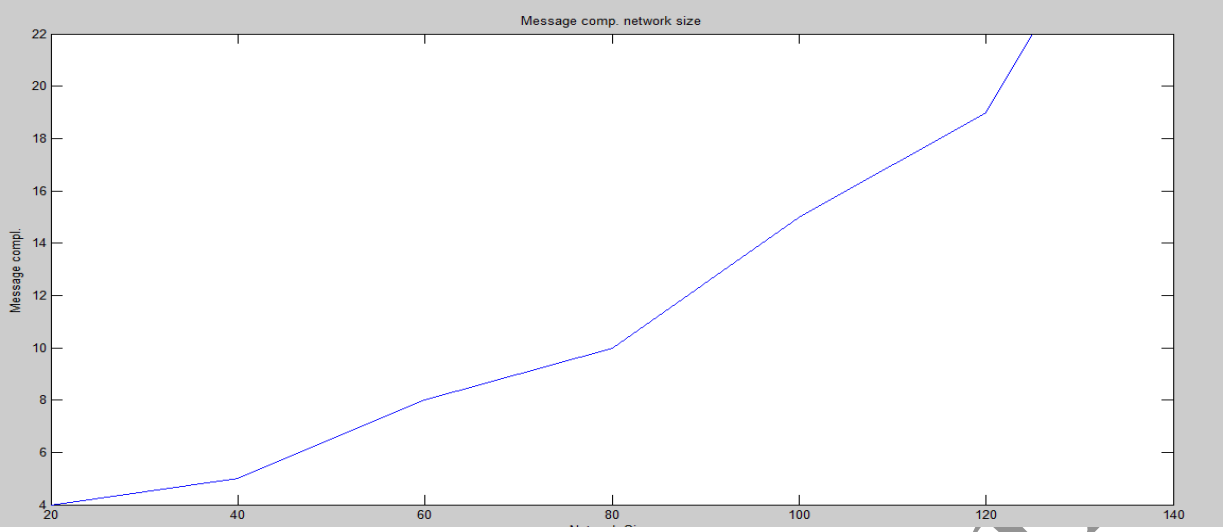


Fig 3.8 Message Complexity Vs Network Size

message complexity Cluster Based is O (n2, mn, l) where n is the number of cluster heads, m is the total number of gateways and l is the number of cluster members and is very high as compared to proposed diagnosis algorithm. This shows the proposed diagnosis algorithm is linearly scalable.

3.2.3.3 Hop-count Vs Network Size

Fig. 3.9 shows the number of hop counts for the proposed diagnosis algorithm to complete fault diagnosis for network of different sizes. Hop count on an average is being calculated as the ratio of the Euclidian distance between the source and destination node in the simulation and the number of nodes in between the source and destination node.

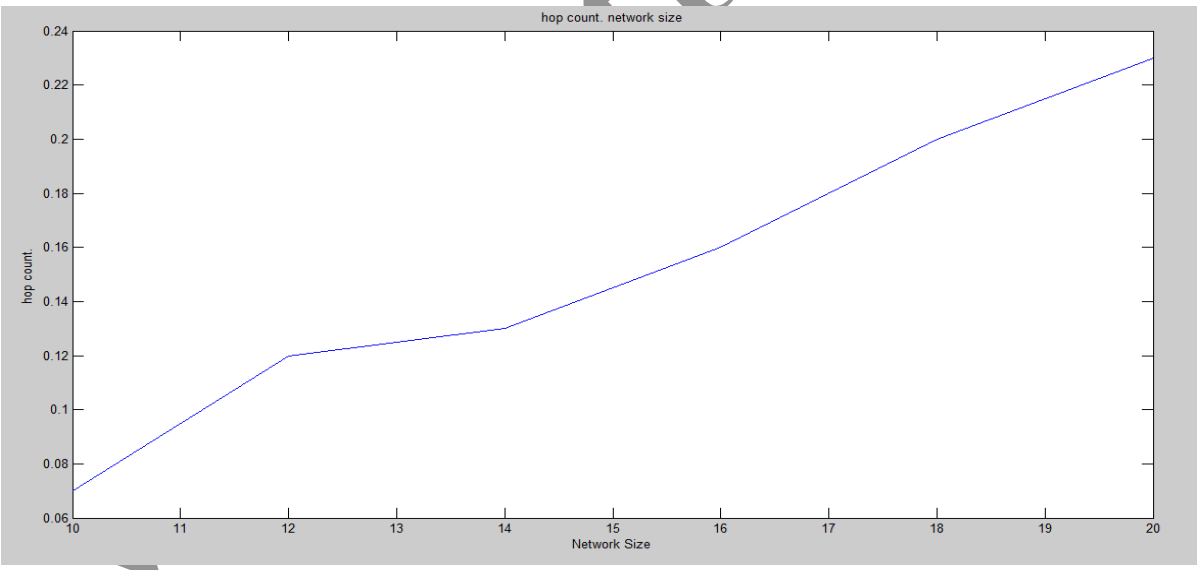


Fig 3.9 Hop-Count Vs Network Size

Here, The length of reading vectors for a sensor node is set to 8 and the similarity threshold is set to 0.6. In fig. 3.7 shows Diagnostic latency Vs. Network size. Fig. 3.8 shows Message complexity Vs. Network size and Fig. 3.9 shows Hop-count Vs. Network size.

CONCLUSION

In WSN with the presence of faulty readings, the accuracy of query results in wireless sensor networks may be greatly affected. first formulated the correlation among readings of sensors nodes. Given correlations among sensor nodes, a correlation network is built to simplify derivation of Sensor Rank for sensor nodes in the network. In light of Sensor Rank, an in-network algorithm Trust Voting is developed to determine faulty readings.

In MANET we proposed a hierarchically adaptive distributed diagnosis algorithm for diagnosing crash and value faulty nodes in MANET based on CBD algorithm. CBD algorithm maps nodes to cluster and uses a divide-and-conquer testing strategy to achieve diagnosis. The proposed algorithm has been replicated using MATLAB and has been evaluated analytically using the standard performance measures such as diagnostic latency and message complexity. The result shows that the proposed algorithm is linearly scalable in terms of diagnostic latency and message complexity as compared to cluster based diagnosis algorithm(CBD).

FUTURE SCOPE

Performance of the system may be improved and further various approaches are used such as pinging approach, in present work we work on heartbeat approach. Our future work includes fault diagnosis in MANET using other clustering techniques.

REFERENCE

[1] C. Intanagonwiwat, R. Govindan, D. Estrin, J. Heidemann, and F. Silva, "Directed diffusion for wireless sensor networking", IEEE/ACM Trans. Netw., vol. 11, no. 1, pp. 2–16, Feb. 2003.

[2] Maher N. Elshakankiri, "Energy Efficient Routing Protocol for Wireless Sensor Networks" , 978-1-4244-2957-8/08/$25.00 © 2008 IEEE.

[3] Nishi Yadav, P.M. Khilar "Hierarchically Adaptive Distributed Fault Diagnosis in Mobile Ad hoc Networks Using Clustering" , 2010 5th International Conference on Industrial and Information Systems, ICIIS 2010, Jul 29 - Aug 01, 2010, India.

[4] Pabitra Mohan Khilar, "Fault Diagnosis in Wireless Sensor Networks", IEEE communications surveys & tutorials, vol. 15, no. 4, fourth quarter 2013.

[5] Shanmugam Udhayakumar ,"Trusted Fault Tolerant Model of MANET with Data Recovery" **,** 2011 Fourth International Conference on Intelligent Networks and Intelligent Systems.

[6] Walee Al Mamun and Hanan Lutfiyya,"Fault Detection in MANETs", 978-1-4673-0269-2/12/$31.00_c 2012 IEEE.

[7] Weigang Wu ,"Eventual Clusterer: A Modular Approach to Designing Hierarchical Consensus Protocols in MANETs" , IEEE transactions on parallel and distributed systems, vol. 20.

[8] Wang-Chien Lee, "Using SensorRanks for In-Network Detection of Faulty Readings in Wireless Sensor Networks",MobiDE'07, June 10, 2007, Beijing, China.

.