View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by ethesis@nitr

Diagnosis of static Topology MANETs in faulty environment

Nayan Jyoti Mishra
Subhadeep Samantaray

Guide
Prof. M.N. Sahoo

Department of Computer Science and Engineering
National Institute of Technology Rourkela
Rourkela-769 008, Orissa, India

Diagnosis of static Topology MANETs in faulty environment

Thesis submitted in

May 2011

to the department of

Computer Science and Engineering

of

National Institute of Technology Rourkela

in partial fulfillment of the requirements

for the degree of

Bachelor of Technology

by

Nayan Jyoti Mishra

(Roll 107CS008)

Subhadeep Samantaray

(Roll 107CS048)

under the supervision of

Prof M.N. Sahoo

Department of Computer Science and Engineering

National Institute of Technology Rourkela

Rourkela 769 008, India

National Institute of Technology
Rourkela

Certificate

This is to certify that the project entitled, '**Diagnosis of Static Topology MANETs in Faulty Environment**' submitted by **Nayan Jyoti Mishra** and **Subhadeep Samantaray**, bearing roll numbers 107CS008 and 107CS048 respectively, is an authentic work carried out by them under my supervision and guidance for the partial fulfillment of the requirements for the award of **Bachelor of Technology Degree** in **Computer Science and Engineering** at **National Institute of Technology, Rourkela**.

To the best of my knowledge, the matter embodied in the project has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date - 10/05/2011
Rourkela

(Prof. M.N. Sahoo)
Dept. of Computer Science and Engineering

Acknowledgement

On the submission of our Thesis report, we would like to extend our gratitude and sincere thanks to our supervisor Prof. M.N. Sahoo, for his constant motivation and support during the course of our work in the last one year. He has been the corner stone of our project and has guided us during periods of doubts and uncertainties. We truly appreciate and value his esteemed guidance and encouragement from the beginning to the end of this thesis. He has been our source of inspiration throughout the thesis work and without his invaluable advice and assistance it would not have been possible for us to complete this thesis.

Nayan Jyoti Mishra

Subhadeep Samantaray

Abstract

Mobile Ad-Hoc Networks (MANETs) are set of mobile nodes that communicates wirelessly without a centralized supporting system. Faulty nodes affect the reliable transmission of messages across the network. In this thesis we deal with the fault identification problem in static topology MANETs. A comparison based approach is used where a set of tasks is given to the nodes and outcomes are compared. Based on these comparisons the nodes are classified either as faulty or fault free. Our new diagnosis model is based on the spanning tree concept in which the testing of the nodes as well as the construction of the spanning tree takes place simultaneously. As a result of which the maintenance and the repairing overhead of the spanning tree is completely avoided thus reducing the number of messages exchanged. We have also developed a simulator which can be applied to a network with large number of nodes. We have carried out the simulation in-order to find out the total number of messages exchanged and the total diagnosis time. On analysing the results we have seen that our model performs better than its previous counterparts. The correctness and complexity proofs are also being provided which also shows that our model performs better from a communication as well as latency viewpoint.

Contents

1 Introduction **1**

1.1 MANETs 1

1.2 Fault Diagnosis 2

1.3 Motivation 3

1.4 Organization 3

2 Basic Concepts **5**

2.1 Notations and Terminologies 6

3 Related Works **8**

3.1 Static distributed self diagnosis protocol 10

3.2 Dynamic distributed self diagnosis protocol 11

3.3 Adaptive distributed self diagnosis protocol 11

4 Proposed Model **13**

4.1 System Model 13

4.2 Fault Model 13

4.3 Diagnostic Model 14

4.3.1 Initiation of diagnosis 14

4.3.2 Reception of test-request 15

4.3.3 Reception of time-out message 15

4.3.4 Reception of local-dissemination message 15

4.3.5 Reception of global-dissemination message 16

4.4 Proposed Algorithm 16

4.5 Protocol Analysis 19

4.5.1 Proof of correctness 19

4.5.2 Complexity Analysis 21
Message Complexity 21
Time Complexity 22
4.6 Comparison with previous works 23
4.7 Simulation Results 23

5 Conclusion and Future work **26**
5.1 Conclusion 26
5.2 Future Work 27
References 28

List of Tables

3.1 Invalidation Rule 10

4.1 Message Complexity 21

4.2 Time Complexity 22

4.3 Comparison with previous works 23

List of Figures

4.1 Comparison between the total number of messages exchanged 24

4.2 Comparison between the diagnosis latency 24

4.3 Comparison among the total number of messages exchanged 25

4.4 Comparison among the diagnosis latency 25

Chapter 1

Introduction

Since the early 1980s wireless cellular systems are quite popular. These cellular systems mainly operate with the help of a centralised supporting system, otherwise known as the access point. It is this access point that helps the users to stay connected in the network. But when it comes to places where there is no fixed access point, this technology has own its limitations. In case of rescue and emergency operations installing a centralised supporting system is time consuming. So in-order to overcome this problem we have mobile ad-hoc networks which can be quickly deployed in places where it is not possible otherwise. MANETs are basically a collection of mobile nodes that communicate wirelessly.

1.1 MANETs

Mobile Ad-Hoc Networks(MANETs) are basically a collection of mobile nodes that communicate wirelessly without any centralised supporting system. Here the users or nodes are free to roam within the transmission range. Mobile ad-hoc networks (MANET) are gaining much popularity in various rescue and emergency operations because of its self-organizable, autonomous and can-be-deployed-anywhere type of characteristics. Nodes in MANER are equipped with a receiver and a transmitter[1]. MANETs are of the following types:

• **Vehicular Ad-Hoc Networks (VANETs)**: This type of MANET is mainly used to communicate between the vehicles and the roadside equipments or just to communicate among the vehicles.

• **Intelligent vehicular ad hoc networks (InVANETs)**: it includes artificial intelligence that aids the vehicles to behave in intelligent manner during drunken driving, collision etc.

• **Internet Based Mobile Ad hoc Networks (iMANET)**: this type of ad-hoc network connects mobile nodes with the fixed internet gateway node. Here the ad-hoc routing algorithms cannot be applied directly.

1.2 Fault Diagnosis

As MANETs are mainly used in rescue and emergency operations, having a reliable communication between the mobiles is of utmost importance. Hence the design of dependable MANETs is gaining popularity among the research communities. But the main problem in designing of dependable MANETs is the distributed self-diagnosis problem. Here each fault free mobile has to keep information regarding the state of all the nodes in the neighborhood or in some applications each node should be able to identify the state of all the nodes in the network [1]. Many elegant distributed diagnosis algorithms are available for wired networks and most of them are based either on the invalidation models such as PMC model [2] or comparison model such as the broadcast comparison model [3] and the generalized comparison model [4]. The comparison approach is the most popular diagnosis approach. Here the nodes are given a set of tasks, the tasks are then executed and the outcomes are compared. The comparison outcomes output by the generalized comparison model is summarized below. The comparison outcome is 0, when both the comparator and the compared mobiles are fault-free. If at least one of the compared mobiles is faulty and the comparator is fault-free, the comparison outcome is 1. Finally, the comparison result is unreliable if the comparator mobile is faulty[1].

The earliest works of fault diagnosis in case of MANETs using the comparison approach was proposed by Chessa and Santi in [5]. They have used the shared nature of the communication channel to distribute the diagnosis. In [5], Chessa and Santi have presented with a distributed diagnosis algorithm that allows the fault free mobiles to know the fault status of all the mobiles in the network.

The most recent work to solve the diagnosis problem is presented in [1]. In [1] an adaptive distributed self-diagnosis protocol (Adaptive-DSDP) is proposed to solve the diagnosis problem in fixed-topology MANETs. In case of fixed-topology MANETs it is assumed that the topology of the network is fixed during the diagnosis session. This model uses a spanning tree containing all the fault-free nodes which is maintained, repaired and used to transmit the information about other nodes.

In this report we have proposed a new diagnosis model based on the spanning tree concept in which the testing of the nodes as well as the construction of the spanning tree takes place simultaneously. Here the test request message helps in the construction of the spanning tree. As a result the overhead of maintaining and repairing of the spanning is completely is avoided, thus improving the time as well as the message complexity.

1.3 Motivation

We have analysed the Adaptive-DSDP model and have found that there is an overhead of spanning tree maintenance which occurs all the time even if there is no diagnosis session running. Also the spanning tree is maintained with a particular node as its root i.e. the initiator is fixed. So if the initiator node fails or any other node detects an altered behavior the diagnosis session will not start. Further the spanning tree repairing starts after the testing and gathering phase which increases the diagnosis latency as well as the communication complexity. Spanning tree maintenance and repairing consumes a lot of time, so constructing it in the testing and the gathering phase itself will be more efficient.

1.4 Organization

The report is organized as follows: In chapter 2 we will see some of the basic concepts along with the terminologies used in the thesis. Then we will have a look on the related works in case of fault detection in MANETs. After that we will see our proposed model along with the algorithm, complexity analysis, simulation results and comparison with previous models. The last section will include the conclusion followed by future works.

Chapter 2

Basic Concepts

We know that faults are software or hardware defects in a system that may disrupt the communication or may degrade the performance. There are different types of faults. Some of which are given below:

• **Permanent Fault**: a fault that cannot be repaired during a diagnosis session and has to be removed and/or repaired by some external administrator [1]. It is of two types, hard and soft. Hard and soft faults are discussed in the next section.

• **Transient Fault**: these are the types of faults that disappears after sometime without any intervention.

• **Intermittent Fault**: it is a special type of transient fault. In this the fault recurs from time to time.

• **Dynamic Fault**: A unit diagnosed as fault-free, fails during the diagnosis session itself.

In our work we are mainly focusing on the permanent faults i.e. hard and soft faults. Some of the notations and terminologies used in this report are discussed below.

2.1 Notations and Terminologies

Here we consider n mobile hosts that communicate via radio networks and each mobile host is having a unique identifier. The topology of a network at any particular instant t is represented by a directed graph S(t)=(W,L(t)), where W is the set of nodes or mobiles present in the network and *L(t)* is the set of logical links between two nodes i.e. if two nodes u, v are connected in the network then $L(u, v) \in L(t)$. In other words we can say, u is in the transmission range of v. For the sake of simplicity we can assume the communication graph to be undirected. Hence for any instant of time t, if $L(u, v) \in L(t)$ then $L(v, u) \in L(t)$ i.e. both u and v can exchange information among themselves. Some of the facts that are being considered here are as follows:

- Each mobile has a unique identity, ID, and it knows the IDs of its neighbors.
- Fault free mobiles can correctly identify the sender of a message [1].
- The message sent by a fault free mobile is correctly received by all the fault free mobiles in the vicinity in a particular bounded time.

Now let us see some of the common definitions used in this report:

i. *Neighborhood set N(u,t)*: the nodes that are present in the vicinity of node u, i.e. the nodes that are able to communicate with node u at a particular instant of time t constitute the neighborhood set.

ii. *Connectivity k of graph S(t)*: connectivity of a graph represents the minimum number of nodes which when removed results in a disconnected graph.

iii. *Stable mobile*: a mobile which is assumed to be fixed or moving with a very slow speed so that it is not expected to leave the neighborhood during the diagnosis session is known as a stable mobile.

iv. *Dynamic mobile*: a mobile which is not fixed and may vary rapidly in positions during a diagnosis session is called a dynamic mobile. A dynamic mobile may receive a message from node u at time t and may reply to other neighbor v, given that it has moved away from u.

v. *Permanent fault*: a fault that cannot be repaired during a diagnosis session and has to be removed and/or repaired by some external administrator [1]. It is basically of two type-

- *hard fault*: the nodes that doesnt work at all and do not respond to any kind of stimulus.
- *soft fault*: the nodes that respond to the stimulus and communicate with the network but with an altered behavior.

vi. λ-*diagnosable*: the λ-diagnosability of a MANET tells that, we can identify all the fault-free mobiles in a network given that the number of faulty mobiles is less than or equal to λ. The maximum number of faulty nodes that is permissible in a network to identify all the fault-free nodes is provided by Chessa and Santi in [5]. In [5] it is proved that the maximum number of faulty nodes permissible is k-1. In other words $k \leq k - 1$. From the above context it is clear that if more that k-1 nodes are faulty then the network will be disconnected thus hampering the process of diagnosing the status of all the mobile nodes.

vii. *Self-diagnosable MANET*: When every fault-free node in the MANET participates in the diagnosis session and is able to correctly identify the status of all the nodes. A diagnosis session is said to be correct only if a fault-free mobile is not mistakenly diagnosed as faulty [6]. A diagnosis session is said to be complete when all the nodes in the network have been diagnosed, otherwise it is incomplete.

In our report we mainly focus on the fixed topology MANETs where the mobility of the nodes is restricted during the diagnosis. In other words if $v \in N(u,t)$ then for any $t \leq t' \leq t + t_{diag}$, $v \in N(u,t')$, where t_{diag} is the diagnosis time-out period. This assumption doesnt mean that the network is static, it only represents that the topology is fixed only during the diagnosis session. In the next section we briefly discuss some of the previous works that has been done in the case of fixed topology MANETs.

Chapter 3

Related Works

Now-a-days large computer systems consist of many components or units, like a computer with more than one processor, a large software system [8] etc. so it is more likely that once in a while these components will fail and the results obtained from such systems will be unexpected. This may cause trouble to the users. There are many approaches available in-order to identify the faulty behavior. One of the most famous approach is the comparison approach where each node sends its test request to all its neighbors and then compares the results. The set of comparison outcome is known as syndrome. Based on the outcomes of this comparison the nodes are classified as faulty and fault-free. Some of the works related to the comparison approach is summarized below.

i. **Model by Malek [1980] and by Chwa and Hakimi [1981b]**: these two are the earliest known models for the comparison approach. In these models there is a central observer which collects the information about comparisons and then performs the diagnosis based on the comparison syndrome to identify the faulty nodes in the network. The comparison is performed by a node called comparator. The central observer is a reliable one i.e. the observer will never fail. Some of the assumptions made in the diagnosis model proposed by Malek[1980] are:

- two fault-free units produces identical results for the same task.
- the result produced by a faulty node is not at all identical to the results produced by another faulty or fault-free node.

This diagnosis model has two activities: fault detection and fault identification. The

fault detection phase only determines the presence of fault in the network where as the fault identification phase is actually used to locate the faulty nodes.
The model by Chwa and Hakimi [1981b] is almost the same as compared to the diagnosis model by Malek [1980], but the only difference is that here two faulty nodes may give the same result to same test task i.e. the output may match.

ii. **MM Model**: This model was proposed by Maeng and Malek [1981], which is mainly used for multi-processor systems. Here the nodes itself computes the comparisons and only the comparison results are given to the central observer. The central observer then identifies the faulty and the fault-free nodes in the network thus completing the diagnosis. This model assumes that the faults are only permanent and comparison performed by a faulty unit is unreliable. Faulty units performing the same task produce different results.

iii. **MM* Model**: This is a special case of the MM Model. Here a node carries out the comparisons of all its neighbors. The comparison outcome is then sent to the central observer for the complete diagnosis of the network.

iv. **Model proposed my Sengupta and Dahbura [1992]**: They have modified the MM Model by allowing the comparators to be one of the units being compared[8].

v. **Probabilistic Comparison-Based Model**: This model was proposed by Dahbura, Sabnani and King [1987]. Here a probabilistic approach is used which assumes that a node may fail with a certain probability. Thus there is no restriction on the number of faulty units in the network. There are basically two probabilistic approaches. In the first one the diagnosis is restricted to a set of faulty units with high probability. In the second approach the diagnosis of the whole system is performed first and later it is proved to be correct with a high probability.

v. **Broadcast Comparison Model**: This model was proposed by Blough and Brown in [1999]. This model is fully distributed which is based on MM* model for the systems with reliable broadcast. Here a task is assigned to a pair of nodes which performs the comparisons and diagnoses the system. This model diagnoses static as well as dynamic faults in a polynomial time.

The diagnostic models discussed above have gained wide acceptance in various fields like identifying faults in mobile ad-hoc networks, checking the integrity of distributed replicated data, checking the manipulation of job results by malicious nodes in grid computing platforms etc. In this report we focus on identifying faults in mobile ad-hoc networks(MANETs) with fixed topology where the mobiles are allowed to move but cannot migrate outside the transmitting range of their neighbors during the diagnosis. One of the related works in case of fixed topology MANETs includes that of Chessa and Santi in [5]. Their model is generally known as static-DSDP (static distributed self diagnosis protocol). Their work was extended by Elhadef, Boukerche and Elkadiki in [1] and [6]. Elhadef, Boukerche and Elkadiki proposed two models related to the fixed topology MANETs, the first one is known as dynamic-DSDP (dynamic distributed self diagnosis protocol) and the second one is known as adaptive-DSDP (adaptive distributed self diagnosis protocol). All three models mentioned above uses the invalidation rule of the gMM model [7] which is summarized in a tabular form below [5]. The static-DSDP model is also discussed below.

u	v	w	Comparison outcome of v and w generated by u
fault free	fault free	fault free	0
fault free	faulty	fault free	1
fault free	fault free	faulty	1
fault free	faulty	faulty	1
faulty	any	any	x

Table 3.1: Invalidation Rule

3.1 Static distributed self diagnosis protocol

Static DSDP model was proposed by Chessa and Santi in [5]. This model uses the shared nature of the communication channel. Here the initiator starts the diagnosis by sending a test request message to all its neighbors. The neighbors after getting the test request message, generates its own test request if has not done yet, calculates the result and sends a test response. The result is then compared, and the nodes are declared faulty or fault-free based on the invalidation rule summarized in the table above. After a node has identified the status of all its neighbors, it transmits its local view to its neighbors. Likewise all the fault-free nodes are able to get a global view of the network. The diagnosis protocol terminates when all the fault-free nodes identify the status of all the nodes in the network.

3.2 Dynamic distributed self diagnosis protocol

Dynamic-DSDP uses the concept of spanning tree. Here each node sends the test response to at most k+1 neighbor, where k is the connectivity of the network. A comparison approach is used to identify the faulty nodes. Spanning tree is constructed after each fault-free node has identified its neighbors. Once the construction of the spanning tree is over the dissemination phase begins. The leaf nodes send their local views to their parents. The non-leaf nodes then wait until they have received the diagnostic messages from all its children. After that they disseminate their view to their parent and so on. At the end the initiator creates the global view of the network and disseminates it to its children. A node after receiving the global view passes it to its children and so on. The diagnosis session ends when all the nodes in the tree have received the global view.

3.3 Adaptive distributed self diagnosis protocol

In Adaptive-DSDP the spanning tree is constructed previously and maintained thereafter. When the network enters into the diagnosis phase the maintenance of the spanning tree stops. During the testing phase each node responds to all the test requests it has received in order to ensure that after this phase it has the fault status of its children as well as parent. After the testing phase is over, the repairing of the spanning tree starts. Here a node removes the faulty nodes from its children. If its parent is faulty, it sends a reconnect message to all its neighbors seeking a fault-free parent. Once the repairing phase is over the dissemination phase starts in which all the leaf nodes send their local views to their parent. The non-leaf nodes then collect the local views from their children and send that to their parent. The initiator after receiving the local views from its children creates the global view of the network and passes them down the tree. A node after receiving the global view passes it to its children. The diagnosis session ends when all the nodes in the spanning tree have received the global view.

Chapter 4

Proposed Model

4.1 System Model

Here we assume the MANET to be a k-connected graph with vertices representing the mobile nodes. The set of nodes which are one hop away from a node u are its neighbors. So a node can directly communicate with its neighbors. Here every node has at-least k neighbors. If more than k-1 nodes fail, the graph will be disconnected because in the worst case a fault-free node will have all its neighbors faulty making it unable to communicate with the rest of the nodes in the network. So we assume that the maximum number of allowable faults is k-1. Hence each node has at-least one fault-free neighbor. Every node is represented by a unique identifier. The topology of the MANET is assumed to be static during the diagnosis, i.e. the mobiles are allowed to move but they cannot migrate outside the transmission range of their neighbors. So N(u,t)=N(u,t'), where $t \leqslant t' \leqslant t + T_{diag}$ and T_{diag} is the time taken for the completion of diagnosis. We assume all the links between the nodes to be fault-free.

4.2 Fault Model

We assume that all the faults in the network to be permanent and a fault-free mobile always produces correct result for a test-task. The result of a fault-free and that of a faulty mobile are always different for the same test-task. When the diagnosis begins a node generates a test-task and sends to its neighbors. It then compares their result with its own to find out their fault status. So a comparison approach is used to identify the fault status of the nodes. There is no restriction on the matching of the results produced by two faulty units i.e. they may produce the same result.

4.3 Diagnostic Model

When a node detects an altered behavior in the system, it can itself initiate a diagnosis or delegate the task to any other node. The node which begins the diagnosis is known as the initiator. We assume that the initiator is always fault-free.
The format of the test-request message sent by any node n is given below:
$TQ_n =< n, TT_n, R_n, parent_n, depth_n >$, where
n = Id of the node sending the message.
TT_n = test task generated by the n.
R_n = result of n for its own test-task TT_n.
$parent_n$ = Id of the parent of n in the spanning tree.
$depth_n$ = depth of n in the spanning tree.

The format of the local dissemination sent by any node n is given below:
$LDM_n =< n, F_n, FF_n >$, where
n = Id of the node sending the message.
F_n = fault set of n
FF_n = fault-free set of n

The format of the global dissemination sent by any node n is given below:
$GDM_n =< n, F_n, FF_n >$, where
n = Id of the node sending the message.
F_n = fault set of n
FF_n = fault-free set of n

4.3.1 Initiation of diagnosis

The initiator i triggers the diagnosis by generating a test task TT_i along with the result R_i. Since the initiator i is the root of the spanning tree, it has no parent and its depth is zero i.e. $parent_i$ = -1 and $depth_i$ = 0.

So the test-request message for i is

$TQ_i = < i, TT_i, R_i, -1, 0 >$

i then transmits TQ_i to all its neighbors and sets its timer to a time-out period of T_{out}.

4.3.2 Reception of test-request

A node m after receiving a test-request TQ_n from the node n, generates its own result $R_{n,m}$ for the test-task TT_n. If the result $R_{n,m}$ is not equal to that of R_n, it identifies n as faulty and adds the Id of n into its faulty set.

Otherwise m identifies n as fault-free. If m has not yet sent its own test-request, it selects n as its parent, sets $parent_m = n$ and $depth_m = depth_n + 1$. It then generates in own test task TQ_m and broadcasts it to its neighbors followed by setting its timer to T_{out}.

$TQ_m = < m, TT_m, R_m, n, depth_n + 1 >$

If m has already generated its own test request and $parent_n = m$, m adds n as one of its children. In this way the spanning tree is constructed during the testing phase itself.

4.3.3 Reception of time-out message

When a node m receives the time-out message from the timer, it marks the status of all the undiagnosed neighbors as faulty. If m is a leaf node then it sends its local diagnostic message LDM_m to its parent.

4.3.4 Reception of local-dissemination message

A non-leaf node waits until all of its children send their local views. It then appends its own view to those from its children and sends it up the hierarchy in the spanning tree. If the node is the initiator it creates a global view of the network and passes it to its children. Otherwise it sends its local view to its parent.

4.3.5 Reception of global-dissemination message

A node m after receiving the global-dissemination message, transmits the same to its children. In this way every fault-free node gets the global view of the network. The diagnosis terminates when every fault-free node in the network has the global view.

4.4 Proposed Algorithm

We have proposed a new algorithm to identify the faults in the fixed topology MANETs. The various terms used in the procedure are as follows:

F_i= the set containing the Ids of the faulty nodes.

FF_i= the set containing the Ids of the fault free nodes.

testGenerated= it is a booloean variable which is initialised to true if the node has generated its test-request, else false.

TT_i= test-task generated by i.

TQ_i= test-request message generated by i.

R_i= result generated by i for its own test-task TT_i.

$R_{u,i}$= result generated by i for the test-task TT_u.

$parent_i$= Id of the parent node of i in the spanning tree.

$depth_i$= depth of i in the spanning tree.

1RB(.)= one hop reliable broadcast protocol in the MAC layer.

$Children_i$= Ids of the children of i in the spanning tree.

CH_i= Ids of the children of i from which it has received the local diagnostic message.

LDM_i= local dissemination message of node i.

GDM_i= global dissemination message of node i.

$N(u)$= The set of nodes which are one hop away from the node u

W= set of all nodes in the network.

checkHeader(.)= boolean function which checks the consistency of the header of a message.

initiator= Id of the node which initiates the diagnosis.

```
Procedure diagnosis /* executed at node i */
{
    do {
```

```
switch (msg)
{
 case Test Request: /* i.e. $TQ_u =< u, TT_u, R_u, parent_u, depth_u >$ from a
                    node u */
  if ($checkHeader(TQ_u)= false$)
     $F_i = F_i \cup \{u\}$;
  else {
     if($R_{i,u} \neq R_u$)
       $F_i = F_i \cup \{u\}$;
     else { /* i.e. $R_{i,u} = R_u * /$
       $FF_i = FF_i \cup \{u\}$;
         if( ! testGenerated) {
           $parent_i = u$;
           $depth_i = depth_u + 1$;
           generate test task $TT_i$ to test the neighbors and find the
           response $R_i$ for it;
           Prepare the test request message -
           $TQ_i =< i, TT_i, R_i, parent_i, depth_i >$;
           Start the timer;
           $1RB(TQ_i)$; /* broadcast the $TQ_i$ */
           testGenerated=true;
         }
         else if ($i = parent_u$)
           $Children_i = Children_i \cup u$;
       }
     }
  if ($F_i \cup FF_i = N(u)$){
     stop the timer;
     if($children = \phi$){
       prepare local dissemination message;
       $LDM_i =< i, F_i, FF_i >$;
       $1RB(LDM_i)$;/*broadcast $LDM_i$*/
```

```
        }
    }
  case Time-Out:
    stop the timer;
    for each node $v \in N(u)$
        if $v \notin F_i$ and $v \notin FF_i$
            $F_i = F_i \cup \{v\}$;
    if($children = \phi$){
        prepare local dissemination massage;
        $LDM_i =< i, F_i, FF_i >$;
        $1RB(LDM_i)$; /*broadcast $LDM_i$*/
    }
  case Local Dissemination Message:/* $LDM_u =< u, F_u, FF_u >$ from u */
    if($u \in Children_i$) {
        $CH_i = CH_i \cup u$;
        $F_i = F_i \cup F_u$;
        $FF_i = FF_i \cup FF_u$;
        if($Children_i = CH_i$) {
            if(i=initiator){
                for each $v \in node$
                  if ($v \notin F_i$ and $v \notin FF_i$)
                      $F_i = F_i \cup \{v\}$
                prepare global dissemination message-
                $GDM_i =< i, F_i, FF_i >$;
                $1RB(GDM_i)$;/*broadcast $GDM_i$*/
            }
            else {
                prepare local dissemination message;
                $LDM_i =< i, F_i, FF_i >$;
                $1RB(LDM_i)$;/*broadcast $LDM_i$*/
            }
    }
```

```
            case Global Dissemination Message:/* $GDM_u =< u, F_u, FF_u >$ from u */
              if($u = parent_i$) {
                $F_i = F_u$;
                $FF_i = FF_u$;
                if($children \neq \phi$) {
                  prepare the global dissemination message-
                  $GDM_i =< i, F_i, FF_i >$;
                  $1RB(GDM_i)$;/*broadcast $GDM_i$*/
                }
              }
      }while($F_i \cup FF_i \neq W$)
}
```

4.5 Protocol Analysis

4.5.1 Proof of correctness

A protocol is correct if every fault-free node is able to correctly identify the fault status of every other node in the network at the end of a diagnosis session. This can be proved by two properties:

i. **Local Correctness**: This tells that a fault-free node correctly identifies the fault status of every other node in its vicinity.

ii. **Dissemination Correctness**: This tells that the dissemination message sent by a fault-free node is correctly received by every other fault-free node in the network.

Proof of Local Correctness

In a k-connected network every node has at-least k neighbors and the maximum number of allowable faults is k-1. So in the worst case a fault-free node n will have at-least one fault-free neighbor m. Since the network is connected there exists at-least one path from m to the initiator containing only fault-free nodes. Hence it is guaranteed that m will generate its own test request. So n gets test request from m. Since n is fault-free it will generate its own test request TQ_n containing its test task as well as the result(a fault-free node generates the test request only if it has received test request from another fault-free node). Consequently TQ_n will be received by m, which identifies n as fault-free. As the test request contains the test task as well as the result, n can identify the status of every responsive node in its neighborhood. If n doesn't receive any test request from a node in its neighborhood within time T_{out}, it identifies that node as faulty. So every fault-free node identifies the fault status of its neighbors correctly.

Proof of Dissemination Correctness

Here we have to show that the spanning tree contains only fault-free nodes and all the fault-free nodes are present in it.

i. According to our model a node will generate its test request only when it has received a test-request from a node which has been identified as fault-free node by it. The test-request sent by a faulty node m is be detected by other fault-free nodes. So m cannot connect to the spanning tree. As a result the spanning tree contains only the fault-free nodes.

ii. Every fault-free node n has at-least one fault-free neighbor m which is connected to the spanning tree(as shown in the proof of local correctness). So after getting test request from m, n sends its own test request requesting m as its parent. m then identifies n as fault-free and adds to its $children_m$ set. Hence n gets connected to the spanning tree.

In the worst case after time T_{out} from the generation of test request each leaf node starts sending its local view to its parent. The parent then collects the diagnostic views of its children, appends those to its own view and sends it to its parent. Finally, the initiator will have the global view of the network which is then disseminated down in the spanning tree. Hence all the nodes in the network will have the global view of the system in a finite time.

4.5.2 Complexity Analysis

Message Complexity

Let n be the number of nodes in the network.

Message type	# Messages	Case
Test Request	n	In the worst case scenario every node generates a test request
Local Dissemination	n-1	Every node except the initiator sends exactly one local dissemination
Global Dissemination	n-1	In the worst case the maximum depth depth of the spanning tree is n-1. Hence n-1 number of messages are needed for the completion of diagnosis

Table 4.1: Message Complexity

From the above table we see that the total number of messages exchanged is **3n-2**. Hence the message complexity is $\boldsymbol{O}(\mathbf{3n}) \backsimeq \boldsymbol{O}(\mathbf{n})$.

Time Complexity

Here we have considered the following terms :

i. d_{ST}**:** Depth of the spanning tree.

ii. T_{gen}**:** Upper bound to the time elapsed by a node between the reception of a test request and the generation of its own test request.

iii. T_f**:** Upper bound to the one hop propagation delay of a message.

Phase	Time Taken	Explanation
Testing and Gathering	$d_{ST}T_{gen} + T_{out}$	The last mobile to generate its test message will do so at most in T_{gen} since the first diagnostic message was received. It follows that in at least $d_{ST}T_{gen}$ all non-faulty mobiles generate their test requests. Since the test request message propgates along the direction of construction of the spanning tree. Every fault-free mobile diagnoses its neighbors and gets connected to the spanning tree in at most T_{out} time. So the time Complexity of this phase becomes $d_{ST}T_{gen} + T_{out}$.
Local Dissemination	$d_{ST}T_f$	The local dissemintaion message passes from the leaf nodes to the initiator along the edges of the spanning tree. So the time complexity of this phase becomes $d_{ST}T_f$.
Global Dissemination	$d_{ST}T_f$	The global dissemintaion message passes from the initiator to the leaf nodes along the edges of the spanning tree. So the time complexity of this phase becomes $d_{ST}T_f$.

Table 4.2: Time Complexity

From the above table we see that the total time taken is $d_{ST}T_{gen} + 2d_{ST}T_f + T_{out}$. Hence the time complexity is $O(d_{ST}T_{gen} + 2d_{ST}T_f + T_{out})$.

Algorithm	Message Complexity	Time Complexity
Static-DSDP	$O(n^2)$	$O(\triangle T_{gen} + \triangle T_f + T_{out})$
Dynamic-DSDP	$O(nk)$	$O(\triangle T_{gen} + d_{ST}T_f + T_{out})$
Adaptive-DSDP	$O(nl)$	$O(\triangle T_{gen} + d_{ST}T_f + T_{out})$
Proposed Model	$O(n)$	$O(d_{ST}T_{gen} + d_{ST}T_f + T_{out})$

Table 4.3: Comparison with previous works

4.6 Comparison with previous works

Previous algorithms along with their message and time complexity is shown in table-4.3.

l = maximum degree of the network

$\triangle$ = diameter of the network

Other terms are defined in the previous section

From the above table we see that the communication and time complexity of our proposed model is better than the previous models. This is because the there is no overhead of maintenance and repairing of the spanning tree which reduces the number of messages exchanged as well as the diagnosis latency.

4.7 Simulation Results

We have designed a simulator in Java 1.6.11 and have simulated the Static-DSDP model, Adaptive-DSDP model and our proposed model for networks of different sizes. Here we are taking the network graph and the Id of the initiator as input. The output contains the total time taken for diagnosis and the number of each type of message exchanged along with the spanning tree. Some of the sample outputs for networks of different sizes is shown in a graphical format (fig.1 and fig.2)

Fig.1 and Fig.2 shows the comparison of the Adaptive-DSDP model with our work. Fig.3 and Fig.4 shows the comparison between the Static-DSDP model, Adaptive-DSDP model and our proposed model. Thus we see that our proposed model works better when compared to the Static and Adaptive DSDP models.

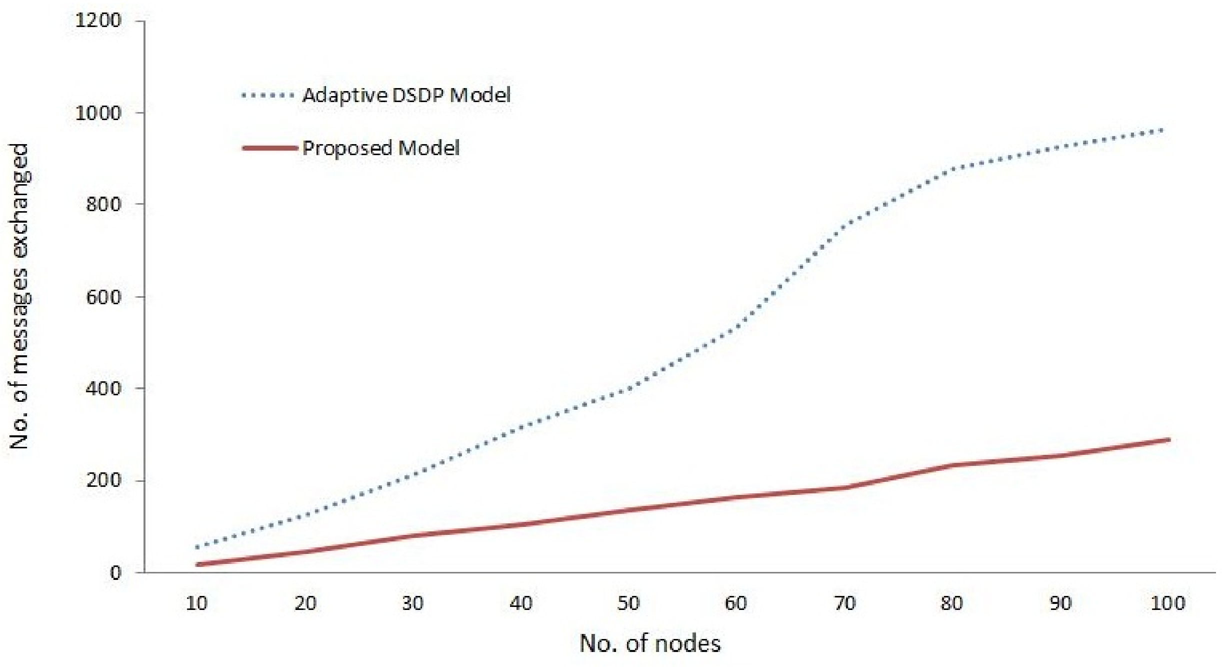

Figure 4.1: Comparison between the total number of messages exchanged

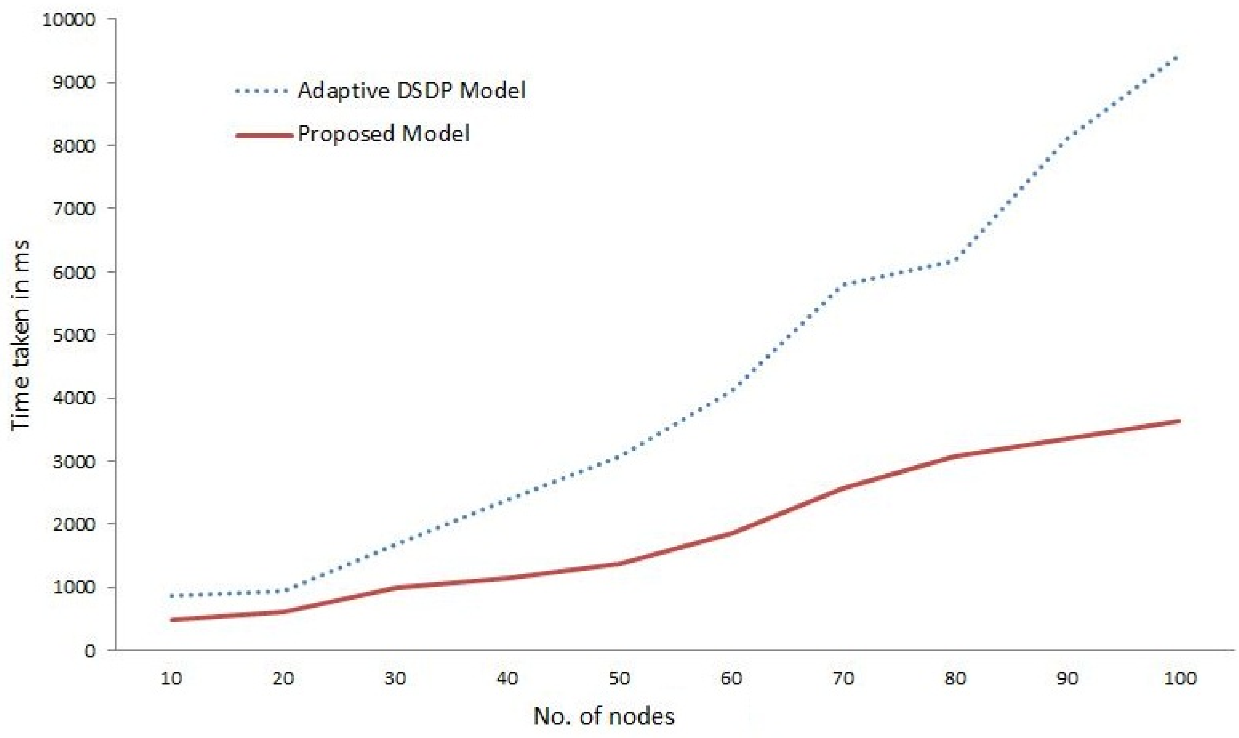

Figure 4.2: Comparison between the diagnosis latency

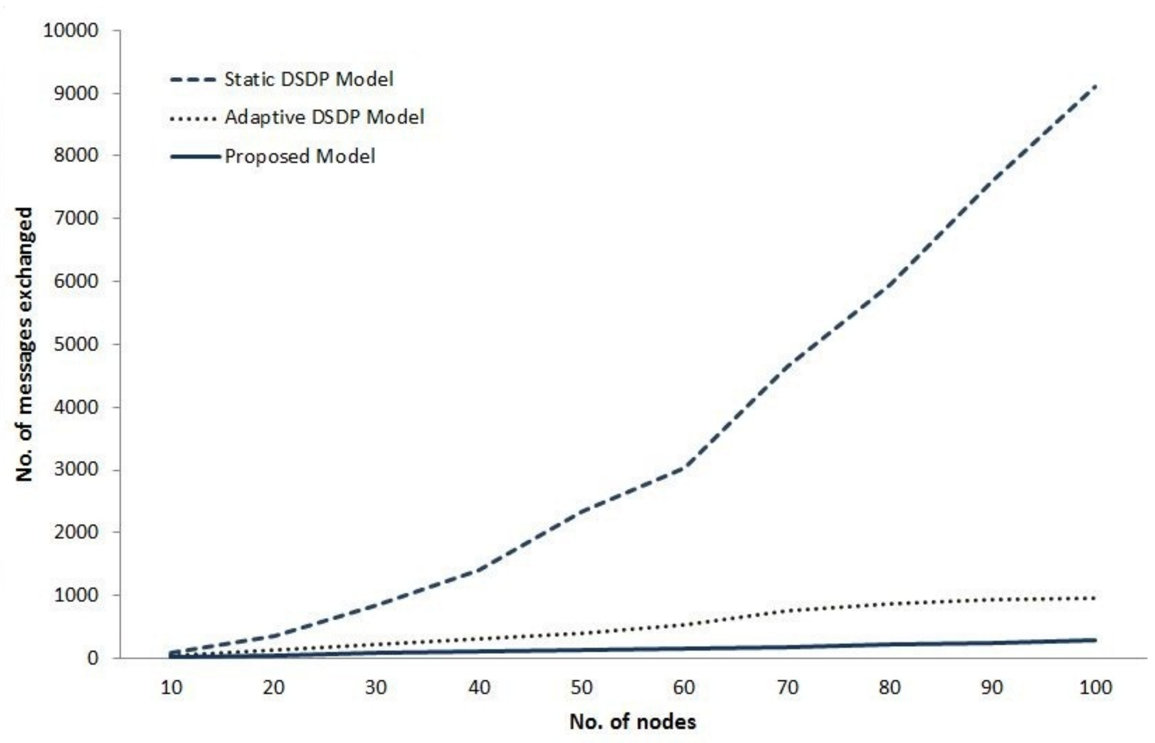

Figure 4.3: Comparison among the total number of messages exchanged

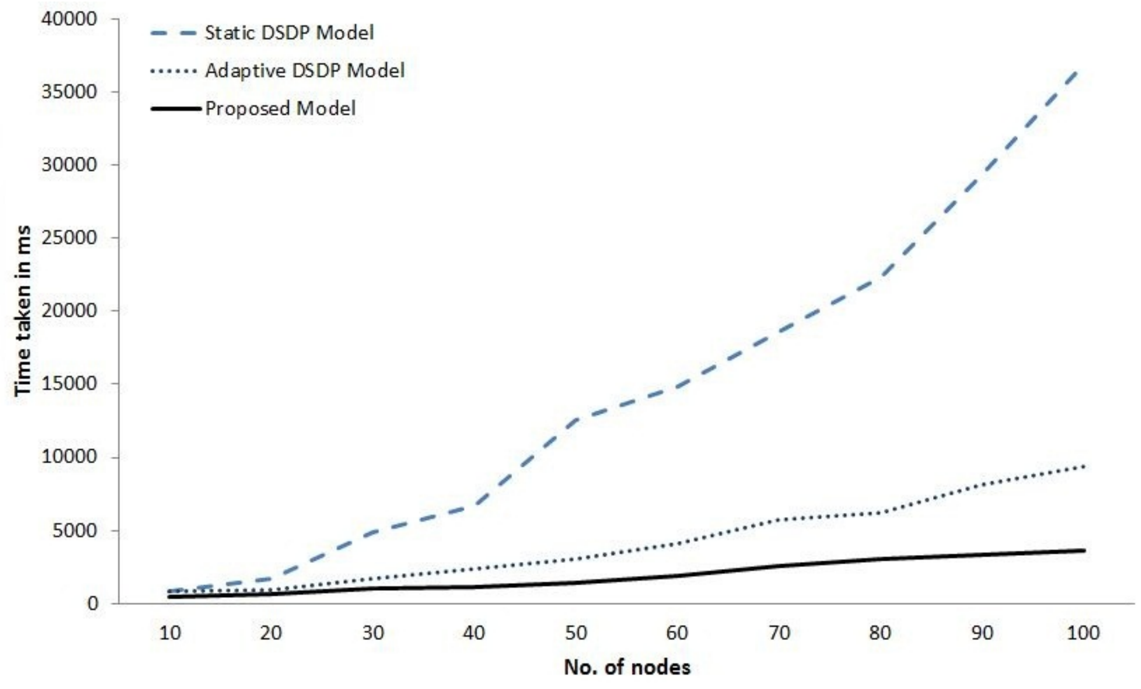

Figure 4.4: Comparison among the diagnosis latency

Chapter 5

Conclusion and Future work

5.1 Conclusion

The earliest work on fault identification in case of mobile ad-hoc networks was carried out by Chessa and Santi in there work in [5]. There model, known as Static-DSDP, considers a comparison based approach and the network topology is assumed to be fixed during the diagnosis session. Such type of network is known as fixed topology network. Another work in case of fixed topology MANETs was carried out by Elhadef et all. in [6]. The model also known as Dynamic-DSDP uses a spanning tree approach to disseminate the local diagnostic messages collected during the testing phase. Here the spanning tree is constructed after the fault status of the nodes has been identified. Adaptive-DSDP [1] also considers a fixed topology environment and a spanning tree approach which is a improvement over the Dynamic-DSDP model. In case of Adaptive-DSDP the spanning tree is initially configured with the MANET and the protocol enables the maintenance as well as the reconfiguration of the spanning tree while the hosts are moving or they are diagnosed by their neighbor.
In this thesis we have proposed a new model for fixed topology environment. A spanning tree approach has also been considered here. In this model the testing of the nodes, gathering of information about neighbors and building of the spanning tree takes place simultaneously. As a result of which the maintenance and the repairing overhead of the spanning tree is completely avoided thus reducing the number of messages exchanged. We have also developed a simulator which can be applied to a network with large number of nodes. We have carried out the simulation in-order to

find out the total number of messages exchanged and the total diagnosis time. On analysing the results we have seen that our model performs better than its previous counterparts. The correctness and complexity proofs are also being provided. From the message and time complexity thus derived we see that our model performs better from a communication as well as latency viewpoint.

5.2 Future Work

The model we have proposed is for a fixed topology MANET. Moreover this model identifies only the permanent faults. In future it is possible to extend this model for dynamic topology MANETs and for identifying intermittent as well as dynamic faults. One of the approach for identifying intermittent faults is to repeat our proposed algorithm for a fixed number of times.

Bibliography

[1] M. Elhadef, A. Boukerche, H. Elkadiki, 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.

[2] M. Barborak, M. Malek, and A. Dahbura. The Consensus Problem in Fault-Tolerant Computing. ACM Computing Surveys, 25(2):171-220, June 1993.

[3]] D. Blough and H. Brown. The Broadcast Comparison Model for On-Line Fault Diagnosis in Multiprocessor Systems: Theory and Implementation. IEEE Trans. on Computers, 48(5):470-493, May 1999.

[4] A. Sengupta and A. Dahbura. On Self-Diagnosable Multiprocessor Systems: Diagnosis by the Comparison Approach. IEEE Trans. on Comput., 41(11):1386-1395, 1992.

[5] S. Chessa and P. Santi. 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.

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

[7] J. Maeng, and M. Malek, A Comparison Connection Assignment for Self-Diagnosis of Multiprocessor Systems, Proc. FTCS-11, pp. 173 - 175, 1981.

[8] Elias P. Duarte Jr., Roverli P. Ziwich, Luiz C. P. Albini, A Survey of Comparison-Based System-Level Diagnosis, ACM Computing Surveys, Pages 1-65.

[9] M. Malek, A comparison connection assignment for diagnosis of multiprocessor systems, in: Proceedings of the 7th International Symposium on Computer Architecture, New York, Association for Computing Machinery Publishers, New York, 1980, pp. 31-35.

[10] S.L. Hakimi, K.Y. Chwa, Schemes for fault tolerant computing: a comparison of modularly redundant and t-diagnosable systems, Inform. and Control 49 (1981) 212-238.

[11] Jinghao Xu and Leszek Lilien A Survey of Methods For System-Level Fault Diagnosis, ACM '87 Proceedings of the 1987 Fall Joint Computer Conference on Exploring technology: today and tomorrow.

[12] A. Bagchi, S.L. Hakimi, An optimal algorithm for distributed system level diagnosis, in: FTCS, 1991, pp. 214-221.

[13] S.H. Hosseini, J.G. Kuhl, S.M. Reddy, A diagnosis algorithm for distributed computing systems with dynamic failure and repair, IEEE Trans. Comput. 33 (3) (1984) 223-233.

[14] E.P. Duarte Jr., T. Nanya, A hierarchical adaptive distributed system level diagnosis algorithm, IEEE Trans. Comput. (1) (1998) 34-45.

[15] A. Subbiah, D.M. Blough, Distributed in diagnosis dynamic fault environments, IEEE Trans. Comput. 15 (5) (2004) 453467.