ENHANCED PACKAGE ROUTING TECHNIQUE FOR MOBILE AD HOC NETWORKS

THESIS

Presented to the Graduate Council
of Texas State University-San Marcos
in Partial Fulfillment
of the Requirements

for the Degree

Master of SCIENCE

by

Anil Maddiboina, B.Tech.

San Marcos, Texas
May 2006

ACKNOWLEDGEMENTS

I would like to thank Dr. Xiao Chen, my thesis and research advisor, for her long supervision and contribution. Without her guidance, my research work and this thesis writing could not have been possible. I owe a huge debt of gratitude to her kindness and patience. My deep appreciation also goes to Professor Davis and Dr. East for their cooperation.

Finally, this thesis would not have been possible without the support of my friends and family. They have been a constant source of inspiration.

This manuscript was submitted on April 24, 2006.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

MANET is an acronym for mobile ad-hoc network. In a MANET the communication does not rely on any existing infrastructure, rather the network is formed by a group of autonomous mobile stations randomly located in a geographic area. In order for this network to work, each node acts as a router to relay packets to the nodes within its communication range. Figure 1.0 illustrates a sample MANET. Applications such as search and rescue, conferences, disaster recovery, battlefield etc, often do not have infrastructure available to set up a communication system. In such situations MANETs can be very easily deployed.

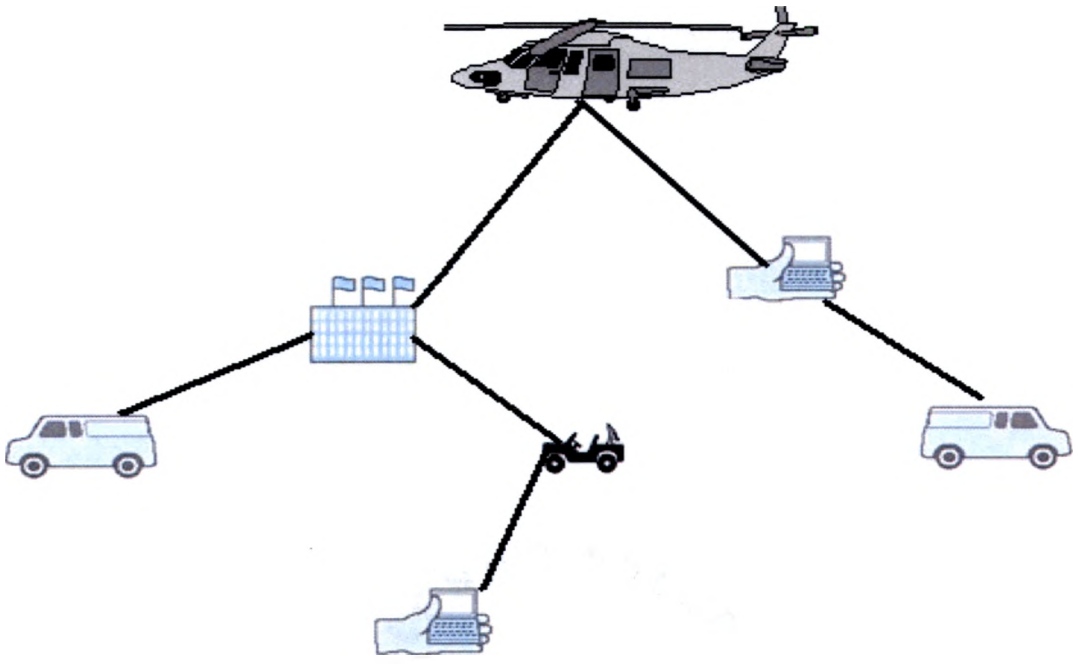


Figure 1.0 : A sample MANET with a stationary and multiple mobile nodes.

All modern mobile computing devices such as laptops, hand held computers, cellular phones etc, are well equipped with network interfaces capable of operating in ad hoc mode, yet there are only a few MANET applications available today. The shortage of applications that can take advantage of the great potential of MANETs is due to the challenges posed by the link layer and network layer protocols. Despite the tremendous amount of recent MANET related research, the protocols for MANET routing are still in their infancy.

Routing in MANETs presents a new set of challenges vis-à-vis traditional wired networks and infrastructure based mobile networks. In a MANET, mobile stations move randomly and unpredictably causing the network topology to change frequently. Each node movement can potentially result in creation of new links as well as loss of previously existing links. The wireless bandwidth available is already limited and sharing this bandwidth among several mobile stations in an area makes it yet more scarce. The multiple access of the same medium by several mobile stations causes signal interference and noise. As each mobile station is powered by a battery, the amount of power available for them to operate is limited. While operating in hostile environment, the security of the mobile stations is challenged. In other words, the mobile stations are not only prone to physical damage, but also to eavesdropping, impersonation, spoofing, and denial of service attacks.

In light of the above described challenging conditions, the choice of routing protocol is very critical. Among all the routing protocols that exist today, there is no clear winner, no single protocol can address all challenges of MANET routing in all scenarios. Also, not every protocol that exists today addresses all of the above mentioned

challenges.

In this thesis, our aim is not to come up with yet another protocol that adds to the myriad of available protocols. We present an enhanced version of Package Routing Technique [X. Chen & X. Jia], which addresses several short falls of Chen's technique.

Chen's Package Routing Technique uses a spanning tree of the network graph for routing purposes. The construction of the spanning tree starts at a randomly chosen root node. The choice of the root node has a great impact on the time required by the algorithm to converge as well as the routing cost. The Enhanced Package Routing Technique (EPRT) developed in this thesis builds the routing network and establishes the center of the spanning tree as the root. Our method builds the tree faster and the routes are relatively shorter. Also Chen's Package Routing Technique does not support node failure and node motion lasting longer than a preset duration. Our technique provides full support for mobility and node failure.

The rest of this thesis is organized as: A brief literature survey in Chapter 2, which classifies the routing protocols and briefly describes how some of them function. Chapter 3 presents a brief description and analysis of Chen's Package Routing Technique presented in [48], followed by a brief description and analysis of two protocols LCMRMG [49] and LCMRMGCS [50], which claim to be better than [48]. Chapter 4 presents a detailed description of the Enhanced Package Routing Technique. Chapter 5 presents the proof of correctness, time complexity analysis, message complexity analysis and a feasibility study of the EPRT technique presented in chapter 4. Chapter 6 concludes this thesis with directions for future work.

CHAPTER 2

LITERATURE SURVEY

This chapter presents a brief survey of the protocols studied during this thesis. For clarity, this literature survey is organized as a classification and a brief overview of the protocols studied. An exhaustive list of existing MANET routing protocols or a detailed description of the protocols studied, is beyond the scope of this document. This chapter presents only a few noteworthy points about protocols of our interest.

The rest of this chapter is organized as follows: Section 2.1 presents a classification of unicast routing protocols. Section 2.2 shows a brief overview of the unicast protocols that operate without location information and a further sub classification along with few sample protocols. Section 2.3 describes an overview of the unicast routing protocols that operate with location information and a few sample protocols of that class.

2.1 Classification of Unicast Routing Protocols

In this section we focus on the classification of the unicast routing protocols for MANETS. Traditionally these protocols are classified as proactive, reactive and hybrid routing protocols, but we decided to classify in a slightly different manner by taking into account any hardware requirements imposed by the protocol on the mobile stations. We broadly classify the unicast routing protocols into two major categories:

- Protocols that operate without location information
- Protocols that operate with location information

Unicast routing Protocols that operate without location information are further classified into three sub categories: Proactive protocols, Reactive protocols and Hybrid protocols. An overview of this class of protocols is presented in section 2.2.

Unicast routing protocols that operate with location information typically require the mobile stations to be capable of finding their geographic location. Although this requirement can be easily met with the widely available GPS technology, the nodes become more susceptible to failure due to the added chance of GPS device malfunction or failure. An overview of this class of protocols is presented in section 2.3. Table 2.1 summarizes the classification of the unicast routing protocols.

<table>
<tr><th colspan="4">Unicast Routing Protocols</th></tr>
<tr><td colspan="3">Protocols That Operate Without Location Information</td><td>Protocols That Operate With Location Information</td></tr>
<tr><td colspan="2">Proactive</td><td rowspan="2">Reactive</td><td rowspan="2">Hybrid</td><td rowspan="2">Eg: DREAM, GPSR, LAR etc.</td></tr>
<tr><td>Event Driven</td><td>Regularly Updated</td></tr>
<tr><td>Eg: DSDV, TORA, WRP, etc.</td><td>Eg: TBRF, OLSR, etc.</td><td>Eg: ABR, AODV, DSR, etc.</td><td>Eg: LANMAR, FSR, ZRP, etc.</td></tr>
</table>

Table 2.1: Classification of Unicast Routing Protocols.

2.2 Unicast Routing Protocols without location Information

Unicast routing protocols that operate without the knowledge of the location information can be categorized into three groups. Based on how these protocols maintain the topology of the network, the groups are:

- Proactive routing protocols
- Reactive routing protocols
- Hybrid routing protocols

From the network graph point of view, each of the above categories of protocols maintains the network information in a very distinct way. Generally the routing overhead is in the order of proactive protocols > Hybrid routing protocols > reactive protocols.

2.2.1 Proactive Routing Protocols

The very first set of protocols proposed for routing in MANETs are the proactive routing protocols, some of these are slightly modified versions of preexisting wire line protocols. Examples of proactive protocols are: TBRF (Topology Based on Reverse path Forwarding [31]), WRP (Wireless Routing Protocol [29]), DSDV (Destination-Sequenced Distance Vector Routing [09]) etc. These protocols typically maintain routes to all or limited set of destinations constantly. Therefore, when a route is required for a particular destination, there is no route acquisition latency.

Proactive routing protocols can be further sub-classified, based on how they transmit routing information, into:

I. Event Driven Protocols

II. Regular Update Protocols

I. Event Driven Protocols

In these set of protocols, nodes transmit routing updates only upon detecting a topological change. TORA, DSDV and WRP are three popular event driven proactive protocols. Following is a brief description of the above mentioned three sample event driven protocols.

(a) TORA-Temporally Ordered Routing Algorithm

This algorithm was introduced by Park and Corson in [28]. It uses a unique single-pass strategy of processing a single event, to perform all route maintenance tasks (deletion of erroneous route, search and establishing new routes).

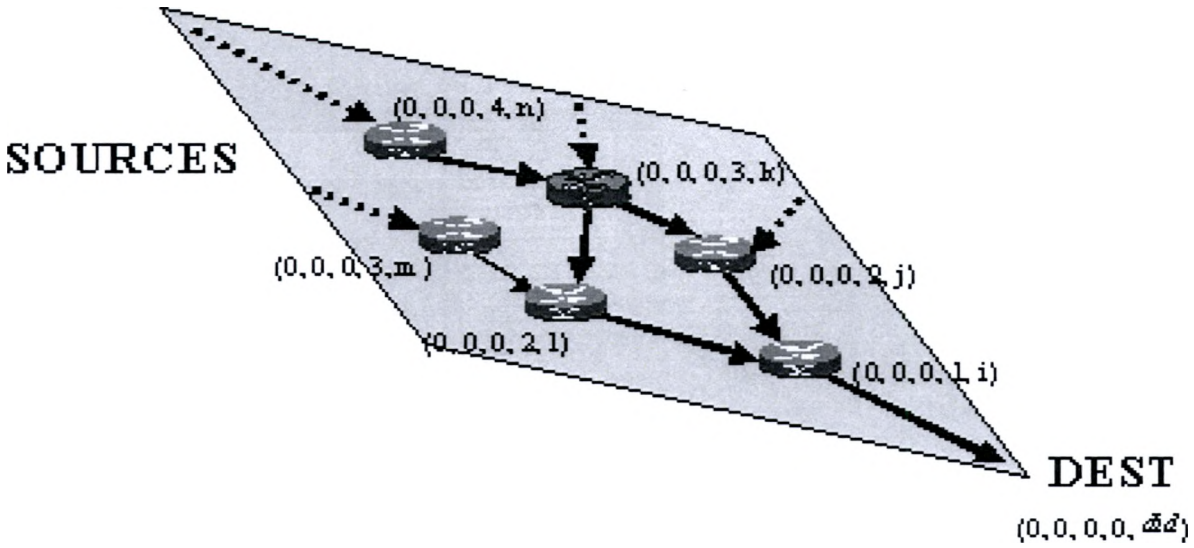


Figure 2.1: Temporarily Ordered Routing.

TORA operates by constructing and maintaining a Directed Acyclic Graph (DAG) rooted at each destination. At any given destination the participating nodes are assigned an arbitrary *height* parameter that is maintained in the DAG as an ordered quintuple. As a result of this multiple routes might be available for a given destination, but none of them are necessarily the shortest route.

An Advantage of TORA is its support for multiple routes between each source and destination pairs. Disadvantages of TORA include the requirement of the nodes to operate with synchronized clocks and reliance on the lower layer protocols to perform some additional functions on behalf of TORA. Figure 2.1 illustrates a sample TORA network.

(b) DSDV- Destination Sequenced Distance Vector Routing Protocol

This is an event driven proactive protocol is a result of adapting an existing distance vector routing algorithm ([13]) to an Ad Hoc network. This protocol introduced the idea of destination sequence numbers to avoid routing loops.

The routing tables in DSDV maintain the information about all destinations with their latest hop count and a sequence number. Whenever a node detects a change in the network topology, it broadcasts the updated routing information to its neighbors. This causes the bi-directional links with most recent sequence numbers to take precedence over older ones. A detailed analysis was given in [24] and [25]. Many comparisons like [26] and [27] show that DSDV is a less efficient protocol than its counterparts.

(c) WRP-Wireless Routing Protocol

WRP is an enhanced version of the Distributed Bellman-Ford protocol (DBF) [06]. It is a table driven proactive routing protocol in which each node within the network maintains four different tables: Distance table, Routing table, Link cost table and Message Retransmission List table (MRL).

The MRL table contains the sequence number of the update messages, a count of how often a message is retransmitted before the connection is lost and a list of updates received in the update message. The routing update messages are sent only to the neighboring set that contains changes in neighborhood links or an update from another node.

Each node is required to confirm the delivery of every update package it received. When there are no update packets to send, the nodes continuously check the link status by sending out "HELLO" messages to its neighbors.

II. Regular Update Protocols

Mobile nodes in regular update protocols transmit the topological information at regular intervals, irrespective of topological changes. TBRPF and OLSR are two popular regular update protocols. Following is a brief description of the two above mentioned regular update protocols.

(a) TBRPF-Topology Broadcast Based on Reverse Path Forwarding

TBPRF is based on the *Extended Reverse Path Forwarding* Algorithm [04]. This is a proactive link-state routing protocol that utilizes hop-by-hop routing, to seek the shortest path to the destination. This algorithm maintains a spanning tree at each node for every other node as the source. The parents of the source node form the spanning tree. The list of all the parents, the full topology table including the cost and sequence number of each link the node is aware of, is stored in the routing tables.

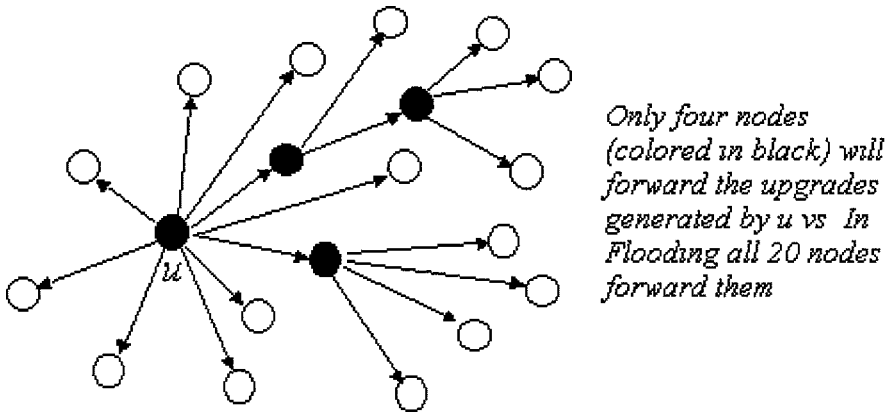


Figure 2.2: A sample TBRPF network

The TBPRF protocol has evolved from the introductory draft undergoing several changes. Recently topology changes were introduced so that partial topology and full topology modes can co-exist. Bidirectional links formed by the nodes are made reliable with a requirement for acknowledgement for every update. HELLO messages are used to check link status and detect new neighbors.

TBRPF is described as being composed of two main components: Neighbor discovery and Routing. For routing each node computes its source tree using a modified version of Dijkstra's algorithm. Figure 2.2 illustrates a sample TBRPF network.

(b) OLSR –Optimized Link State Routing

OLSR is a proactive link state protocol that works best in large dense networks. Each node selects a set of "Multi point Relays" (MPRs) from its neighbors in the 2-hop range and every node knows for which node it acts as an MPR. The OLSR requires bi-directional links and routing through UDP. To reduce channel competition, OLSR requires the avoidance of simultaneous packet emissions among the nodes. Routing in OLSR is based on the shortest path algorithm.

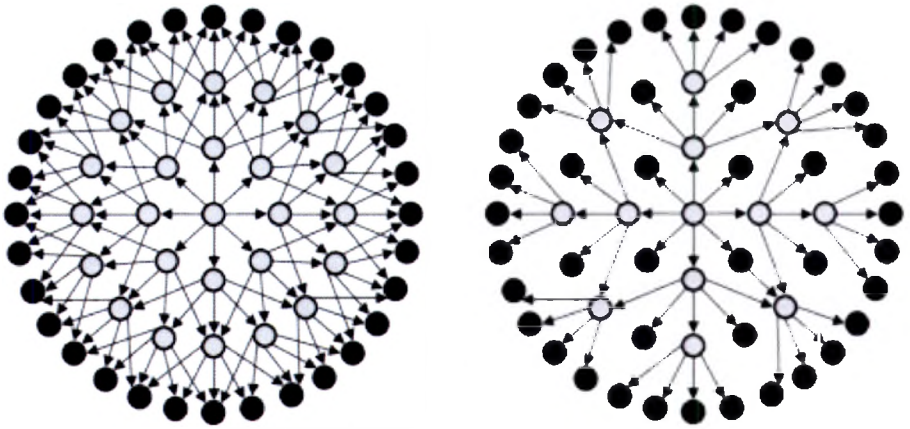


Image on the left is flooding, where every node forwards messages, while in MPR only multi point relays (Grey nodes) can forward the messages

Figure 2.3: MPRs vs. Normal flooding.

The initial draft of OLSR was introduced to MANET working group in 1998; the draft has evolved since to [02]. There were few performance comparisons of OLSR in [01] and [30] to other protocols. Figure 2.3 illustrates a sample OLSR network.

2.2.2 Reactive or On-demand Routing Protocols

In reactive protocols, the routes are created and maintained, on demand, by the sources of communication. When a node needs a route to a destination, it initiates a route discovery process. The route discovery process ends either by finding a route to the

destination or all permutations are exhausted. The route is maintained by route maintenance procedure until the destination becomes inaccessible or route is no longer required. As these protocols are demand driven, there is no effort required to maintain the network topology. Therefore, the routing overhead is considerably lower than their proactive counterparts.

AODV (Ad hoc On-Demand Distance Vector [12]), ABR (Associatively Based Routing [14]) and DSR (Dynamic Source Routing [16]) are three most popular reactive routing protocols. Following is a brief description of the three above mentioned reactive routing protocols.

(a) ABR- Associativity Based Routing

This is an on-demand routing protocol [33]. A mobile station when needs a route to a destination, broadcasts a *Broadcast Query Request*. The destination learns all possible routes and replies along a stable route to the source. Several route reconstruction techniques are available in case of a link break due to the displacement of the source, the destination or any intermediate node.

Every node has a "*degree of associativity*" in form of associativity ticks. Every node maintains a tick value for every one of his neighbors. The tick value is increased every time when a hello message is received from the neighbor. If a neighbor moves out of reach the value is reset to zero. A threshold tick can be chosen and any tick level above this value indicates a stable association between two nodes.

The degree of associativity is chosen as a metric of mobility. On selecting a route the destination does the selection based on the stable routes (i.e., routes with high degree of associativity).

In [34], ABR was compared to other on demand protocols like DBR and DSF by a simulation study. The results are in the favor of ABR considering the throughput, overhead and end-to-end delay. Other criteria like power consumption and memory requirements are pitfalls of ABR.

ABR's initial draft was submitted to MANET working group in 1999, and in 2001, [34] was published proposing an enhancement to ABR. The stability property measured in ticks, in [34], is more advanced and refined. Also, due to an optimized threshold value introduced [34], the route with the highest degree of associativity is no longer a choice.

(b) AODV- Ad Hoc On-demand Distance Vector Routing Protocol

AODV is the most popular protocol of all MANET routing protocols. It is discussed in a lot of research and also used to compare other routing protocols. The draft of this protocol has reached RFC track as an experimental standard.

The route to a destination, if unknown, is discovered by broadcasting a Route Request packet (RREQ) throughout the network. This broadcast is limited in its impact by sending the RREQ in an expanding ring type technique: the TTL of request packets is set to a small value initially, if no route has been found to the destination the TTL of the request packet is increased and request is resent. The nodes that rebroadcast the request add their addresses to the intermediate-node-list of this packet. The destination responds to the request with a Route Reply (RREP) to the source.

The routing table entries consist of a destination, the next hop towards destination and a sequence number. The updates to the routing table are performed only if the sequence number in a message from a destination is larger than the existing sequence number for that destination, in the routing table. Routing loops and updates with stale

messages are prevented by this destination sequence number scheme. The amount of information stored with a node is limited as in: each node is aware of its neighbors via explicit HELLO messages and/or link-layer notification, the node has information about destinations and the next hop, precursor list for each destination is maintained within the node so that in case of route failure to destination the node is able to notify other nodes. Lastly each routing entry has a lifetime. A comparison of AODV and other ad hoc routing protocols is presented in [35], [36] and [37]. In all comparison studies AODV seems to out perform the rest of the protocols, in most of the cases.

(c) DSR-Dynamic Source Routing

This on-demand protocol uses source routing, in which, each packet carries a complete route to its destination in its header. This protocol was first described in [38].

The routes are discovered on-demand, via a Route Discovery Mechanism similar to AODV. All routes discovered, including the best and worst, are cached. Therefore, unlike AODV, DSR has multiple routes for each destination it discovered. Broken links are countered with a corresponding Route Error message sent throughout the network and an effort is made to patch the links if possible.

DSR is one of the mature ad hoc routing protocols that has been implemented and tested in real time. The tests and results are studied in [39], [40] and [41]. The studies showed that an end-to-end recovery mechanism does not scale as the routing path length increases. DSR is used for performance comparison of other routing protocols and it is also used as a reference protocol to find improvements in Mobile Ad hoc Networks.

2.2.3 Hybrid Routing Protocols

Hybrid routing protocols have the features of both proactive and reactive routing protocols. The proactive component of the hybrid routing protocols maintains the routing

information to select destinations that qualify a preset conditions. The reactive component finds the routes to rest of the destinations dynamically. Few popular hybrid routing protocols are LANMAR (Landmark Routing Protocol [17]), FSR (Fisheye State Routing) and ZRP (Zone Routing Protocol [21] and [22]). Following is a brief description of the above mentioned hybrid routing protocols:

(a) LANMAR – Landmark Routing Protocol

LANMAR can be considered as a combination of FSR and Landmark routing techniques. This protocol combines the characteristics of both link state and distance vector protocols. In each subnet or group of nodes that are likely to move together, a landmark node is chosen and all packets are routed via this landmark node. LANMAR routing updates containing information about the nodes in the scope (within a subnet /group) and the landmarks are used to maintain the subnet. As a packet reaches the vicinity of the destination node, it gets the more accurate route and may not even be required to route the packet through the landmark. Special handling is required when isolated nodes exists which does not belong to any group, but could be their own landmarks.

LANMAR takes edge over AODV and DSDV especially in case of large number of nodes and high mobility. [42] presents an extended version of LANMAR that uses a new landmark election process.

(b) FSR – Fisheye State Routing

Fisheye State Routing, proposed by Mario Gerla et.al, is similar to DREAM. It maintains a topology map at each node and propagates link state updates periodically, by exchanging the entire link state information, only with immediate neighbors. As the link state updates are periodical, frequent updates of broken links can be avoided (especially if

the environment has a unreliable wireless link or overall the nodes are highly mobile). FSR broadcasts link state information periodically and at different frequencies for different nodes, based on hop distance. Update entries to faraway destinations are propagated at low frequency and higher for those nearby.

FSR is able to accurately produce information on distance and path about the neighborhood of a node, but has an imprecise knowledge of the shortest path to distant destination.

(c) ZRP – Zone Routing Protocol

This protocol introduces the idea of *route zone,* which is a set of nodes in the neighborhood. A zone is defined by the maximum number of hops from a node. The ZRP uses an *Intra zone Routing Protocol (IARP),* a proactive protocol to discover routes inside a zone and an *Inter zone Routing Protocol (IERP),* for route discovery in other local zones. ZRP uses a *Border cast protocol (BCP)* to forward the request to peripheral nodes of the zone, which in turn can check if the target is within their own zone or border cast. The border cast process must take care not to forward request back into regions already covered.

ZRP is considered as a reference protocol that utilizes the hybrid approach and should not be used in independent performance comparisons.

2.3 Unicast Routing Protocols with Location Information

In comparison to the protocols that operate without location information, unicast routing protocols that use location information operate with a better knowledge of the network. These protocols generally maintain the location information of all or some of the nodes in the network. Knowing the exact or an estimated location of a destination, nodes can make better routing decisions. Recent development in protocols using location

knowledge is due to the availability of accurate and cheaper GPS hardware. Popular unicast routing protocols that use location information are DREAM (Distance Routing Effect Algorithm for Mobility [18]), GPSR (Greedy Perimeter Stateless Routing Effect Algorithm for Mobility [23]), GRA (Geographical Routing Algorithm [20]), GDR (Geographic Distance routing protocol [19]). In rest of this section we provide a brief description of DREAM and GPSR.

(a) DREAM- Distance Routing Effect Algorithm for Mobility

This algorithm is location-based algorithm that makes use of *distance effect*. The location information for distant nodes need not be updated as accurately as for closer nodes, because nodes appear to move slower with respect to each other with increasing distance.

Each node maintains a routing table with the location information of the rest of the nodes in the network, making it a proactive routing protocol. To send a message, a direction is determined by using the location of the destination node. Then the message is passed to all nodes in that direction.

The frequency of location updates depends on the mobility and distance. While the short lived messages are sent more often to refresh the knowledge of a node's vicinity more frequently, the long-lived messages that reach far away nodes are sent less frequently, making it an energy and bandwidth efficient protocol.

Even though the messages are sent in one particular direction towards destination, the loop-free nature of this protocol can be questioned. In mobile networks the directions change randomly, even back to a node that has already sent a message. Another problem is that the location table entries may become fetid and a close neighbor in the required

direction can not be found. Both of these issues are discussed in [44] and the authors choose to use flooding in their prototype implementation.

(b) GPSR- Greedy Perimeter Stateless Routing

This is a location based protocol in which a node learns about the position of its neighbors from the information in the data packets. Each node forwards messages to a neighbor, which is geographically closest to the destination, in a *greedy way*. If there are no such neighbors in the range of target, it switches to *perimeter mode,* which guides the packet around this void area, using a planar-graph traversal with a right hand rule. This protocol requires the nodes to be aware of their position. Also, each source node, before it can start routing, should know the location of the destination. GPSR is designed to work with both mobile ad hoc networks as well as sensor networks.

This protocol is mentioned in large variety of papers like [45], [46], and [47]. Some simulation studies in the above mention articles, claim to have achieved better results with GPSR than DSR.

CHAPTER 3

RELATED WORK

This chapter presents a brief study of three protocols that are very closely related to our work. First of the three is the Package Routing Algorithm discussed and presented in the 2001 ICPP workshop on Mobile Ad-Hoc Wireless networks by: X. Chen and X. D. Jia, 2001, [48] [1]. This thesis is based on [48] and provides solutions for the identified deficiencies of it. Two other protocols studied in this chapter are Locality Caching Multi-Root Multi-Generation Routing (LCMRMG) (Xin Zhang, 2004) and Locality Caching Multi-Root Multi-Generation Routing with Color Schema Routing (LCMRMGCS) (Li Zhuojing, 2005). Both LCMRMG and LCMRMGCS claim to be improvised versions of [48].

After studying LCMRMG and LCMRMGCS, we found that these improvements claimed by the authors are in fact specious. Upon further analysis it was found that the actual flaw is in the technique used to facilitate the research, that is, the simulator they used to support their research. In this thesis we have attempted to lay down a proper platform for future research of our improved routing technique EPRT, discussed in chapter 4.

Rest of this chapter is organized as follows: each of the sections 3.2, 3.3 and 3.4

[1] In rest of this work reference item [48] of bibliography, Chen's technique, and Chen's package routing technique will be used interchangeably

will introduce, briefly describe and present an analysis of Chen's package routing technique, LCMRMG and LCMRMGCS respectively. For a detailed description of these protocols please refer to the items [48], [49] and [50] of the bibliography.

3.2 Chen's Package Routing Technique For MANETs

Chen and Jia (2001) have put forward a new proactive routing technique that guarantees routes between all source and destination pairs as long as there are no partitions in the network. This routing technique is organized into four algorithms:

I. Algorithm to Construct Spanning Tree and Generation Tables (STGT Algorithm)

II. Algorithm for a Mobile Station to Sign-On to the Network (AMSO Algorithm)

III. Algorithm for a Mobile Station to Sign-Off the Network (AMSOF Algorithm)

IV. Algorithm for a Mobile Station to Move in the Network (AMSM Algorithm)

In the following section we present an overview of Chen's package routing technique.

3.2.1 An Overview of Chen's Technique

The fundamental idea of Chen's routing technique is to build a spanning tree with a generation table at each node of the network. The functionalities of spanning tree construction and generation table creation at each node are performed by the STGT algorithm. A randomly chosen root node starts the STGT algorithm and expands the tree generation by generation until the leaf nodes are reached. Each leaf node will build its generation table and pass that table to its parent node. Its parent will build its own generation table by combining the generation tables of all its children and then pass this table to its parent. This process continues until the generation table of the root node is

constructed.

The distinct features of Chen's protocol are the usage generation tables, generation numbers and child identification numbers. The root node assumes GEN = 0 and CID =0 while rest of the nodes are assigned a generation number (GEN) and a child identity (CID) by their parents. The generation numbers grow from zero at the root to the depth of the tree at the farthest leaves. The child numbers are assigned to the children in an incremental manner and in the order they connect to a parent node. Figure 3.1 from [48], depicts a sample network with generation numbers and the child identification numbers assigned to the destinations. Table 3.1 illustrates a sample generation table for node u0 in the network graph in the Figure 3.1.

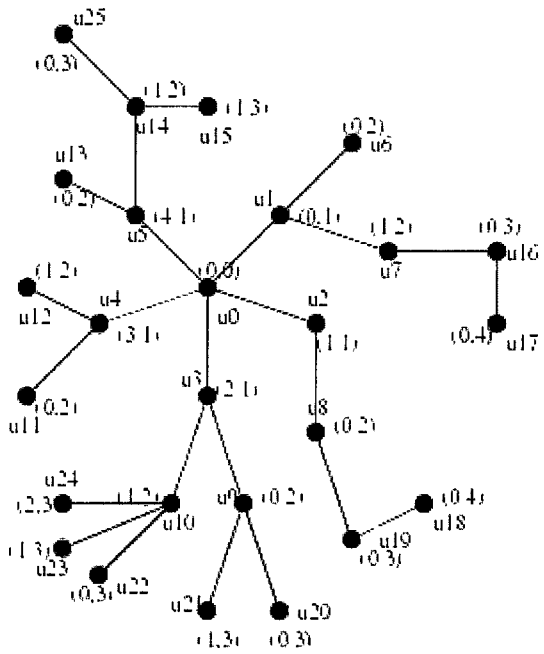


Figure 3.1: Spanning Tree Generated by Package Routing Technique, from [48].

Once the spanning tree and the generation tables are ready, the tasks of a station signing on to the network, signing off of the network and mobility are handled by AMSO, AMSOF and AMSM algorithms respectively. A station signing on to the

network will broadcast its intention to connect, wait until it receives the responses from the stations in its vicinity and then accepts the station with the smallest generation number as its parent. Any station signing off the network will send a sign off message to its parent and its children. The parent deletes the node that is signing off and its entire offspring set from its generation table while the children release their children and sign on to the network as new nodes do.

Each node sends a start of move and an end of move messages right before and right after the travel respectively. If the parent node receives both messages then it sends a within area message back to the displaced node, otherwise the parent node assumes the displaced node moved out of its transmission range and treats it as if it signed off the network. Similarly, if the child nodes receive both start of move and end of move messages, they send a within area message to the displaced node, otherwise they release their children and sign on to the network like new nodes do.

U0's parent's ID None (u0 is root)		
U0 GEN 0 CID 0		
U0's future generations		
CID	IP	Future Generations
0	u1	u6, u7, u16, u17
1	u2	u8, u18, u19
2	u3	u9, u10, u20, u21, u22, u23, u24
3	u4	u11, u12
4	U5	U13, U14, 115, U25

Table 3.1: Generation Tables According to the Package Routing Technique assumes the ID numbers such as u0, u1, etc are the IP addresses.

Routing in Chen's technique is rather simple, each node checks to see if the destination is in its generation table. If the destination is in its generation table, the node forwards the packet to be routed to its offspring which is either the destination itself or has the destination as its offspring. If a node did not find the destination in its generation table, it forwards the packet to be routed to its parent. This process continues until either a node with the destination as its offspring or the root node is reached.

In the following section we identify few drawbacks of Chen's package routing technique.

3.2.2 A Brief Analysis of Chen's Package Routing Technique

(a) Root Selection

In Chen's technique, a random node is chosen as the root of the network. In spite of several topological changes in the network, this node remains as the root for the life of the network. This random choice can potentially show an adverse affect on this routing technique.

For a network graph G of diameter D(G), the worst case diameter of the spanning tree is 2(D(G)). As Chen's technique uses a spanning tree of the network graph, the diameter of the spanning tree can be up to 2(D(G)). Which means the routes in Chen's technique can be as much as twice the actual route length in the network graph. Shown in Figure 3.2 is a classic example of a graph G of diameter D(G) and its spanning tree T with a diameter of D(T) = 2D(G).

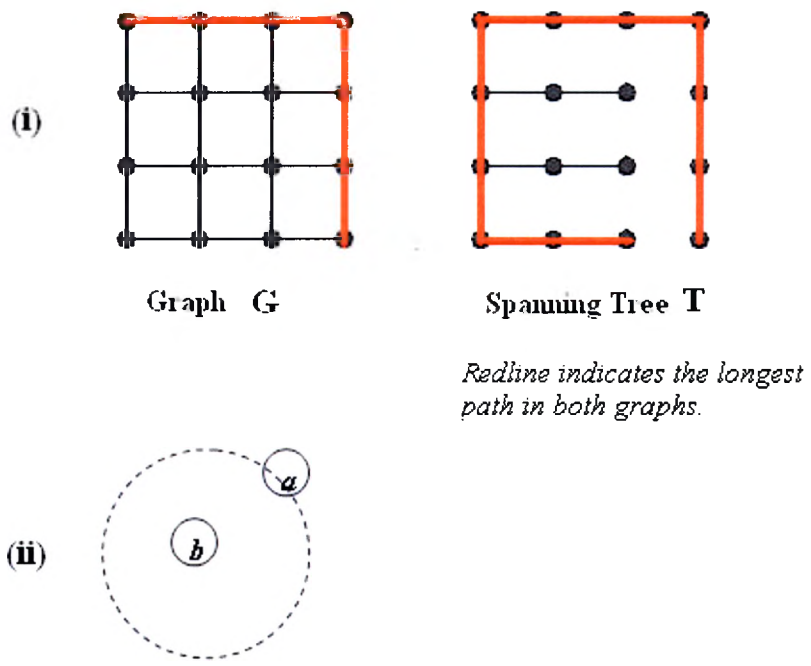


Figure 3.2: (i) A graph G and its Spanning Tree T.
(ii) A node a revolving around node b.

(b) Time Required for the Tree Formation

STGT algorithm in Chen's technique starts at the root node and travels down to the farthest leaf node and then back to the root node. In the worst case if one end of the diameter is chosen as the root, the maximum amount of time required by STGT to finish the tree construction and be ready for routing is equal to twice the diameter of the tree D(T). In other words, the maximum time required by STGT algorithm can be measured as: $2D(T) = 2(2D(G)) = 4D(G)$. This time required can be reduced with a better choice of the root node.

(c) Support for Node Failure

Chen's Package routing technique does not support node failure. In a dynamic situation such as a MANET, node failure can occur more often than one can predict. The

fact that the MANETs are likely to be used in challenging environments such as warfare, disaster recovery, etc. makes the node failure a very common issue. Nodes may also fail due to the exhaustion of the battery.

(d) Support for Mobility

The AMSM algorithm in Chen's Package Routing Technique handles the mobility of the nodes. This algorithm requires the nodes to notify all of their spanning tree neighbors before and after the journey, which seems like the protocol expects the mobile nodes to stop after some travel.

Links in a MANET do not necessarily break due to node mobility, links break when two nodes that were previously in transmission range are no longer in each others transmission range. A node can be in constant motion and still maintain a link with its neighbor. For example: node **a** in section (ii) of Figure 3.2 sends a start of move (SOM) message to node **b** before it starts its motion and then keeps closely revolving around node **b** This results in node **a** not sending an end of move (EOM) message to node **b** In such a situation, although node **a** is within the transmission range of the node **b**, node **b** will assume that the node **a** is no longer in its transmission range. Also, AMSM ignores the fact that two nodes can be both in motion and still be within each others transmission range.

In a very dynamic and mobile environment such as a MANET, it is not a feasible idea to require the mobile nodes to inform its neighbors before the nodes can move. Nodes can come in range and go out of range quite rapidly. A node might not even have time to sign off the network before it moves out of range.

(e) Root Overload and Root Failure

Once a root is selected at the beginning of the protocol, it remains a root for the life of the network, making the entire network dependent on a single node. The root node receives updates for all changes in the network and it is also part of most number of routes in the network. This can consume lot of computing power as well as the battery of the root node within a short duration.

Chen's Package routing technique neither makes any attempt to bypass the traffic through the root nor does to find any alternative root.

3.3 Locality Caching Multi-Root Multi-Generation Routing (LCMRMG)

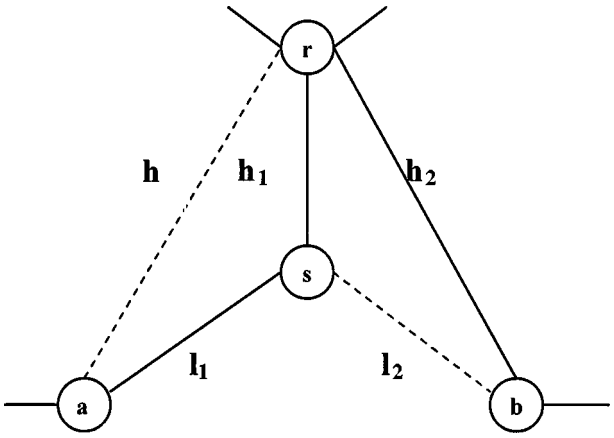


Figure 3.3: Basic idea of new root eligibility in LCMRMG routing.

In [49], Xin Zhang has presented a new protocol that is based on Chen's technique. LCMRMG routing is based on the idea of turning the existing single spanning tree into a multiple rooted graph, routing overhead could be successfully distributed. Each root in the graph is in charge of its own spanning tree resembling the single root in Chen's technique. In this protocol each node belongs to multiple spanning trees at the same time and maintains its generation number with respect to each spanning tree. Similarly each mobile station stores multiple spanning tree information locally, through which multiple routing paths can be found.

In the LCMRMG algorithm, a new technique that takes advantage of traffic locality in MANETs is introduced into the protocol. Figure 3.3 is the example used by the author to illustrate the basic principle behind the algorithm. LCMRMG assumes node **r** as a current root host that covers hosts **s, a,** and **b**. Host **a** is a descendant of host **s**. Host **b** is not in the sub tree rooted at host **s**. Therefore when **a** and **b** communicate with each other, the packets are routed **a←→s←→r←→b.** The total distance of this route is $\mathbf{l_1} + \mathbf{h_1} + \mathbf{h_2}$. In a MANET, larger distance usually implies more hops from a start node to reach a destination.

If host **s** has to function as a router for a high volume of network traffic that flows through it using the routes similar to that of **a←→s←→r←→b**, enabling host **s** to function as a root covers both host **a** and **b** could significantly reduce the network traffic volume. Once **s** functions as such a root host, the previous route between **a** and **b** will be replaced by route **a ←→s ←→b**. The distance for this new route is $\mathbf{l_1} + \mathbf{l_2}$ which in general is smaller than $\mathbf{l_1} + \mathbf{h_1} + \mathbf{h_2}$. For the above technique to work, the author requires each node to be capable of knowing its geographic location coordinates or Global Positioning System (GPS) coordinates. On top of the algorithms used in the Chen's routing technique, LCMRMG uses traffic locality caching algorithm and a different routing packet forwarding algorithm.

The traffic locality caching algorithm is responsible for determining which node becomes a root and when. Following is the traffic locality caching algorithm used by LCMRMG:

I. Each host maintains a counter variable **ct** that records the number of communication messages it has routed such that these messages carry

three pairs of coordinates namely that of the sender, receiver, and the root.

II. Each host **s** maintains two variables α and β. These two variables will be modified on each message routing through **s** that carries above mentioned three pairs of coordinates.

   a. The variable records the accumulated distance values $l_1 + h_1 + h_2$ as illustrated in Figure 3.3. For each message that goes through **s**, the variable is modified to $\alpha = \alpha + (l_1 + h_1 + h_2)$

   b. The variable records the accumulated distance values $l_1 + l_2$ as illustrated in Figure 3.3. For each message that goes through **s**, the variable is modified to $\beta = \beta + (l_1 + l_2)$.

The host **s** periodically checks the value of its counter variable **ct**. When the value of **ct** becomes sufficiently large, the host calculates the value of $(\alpha - \beta)/\alpha$. If the following two relationships hold, $\alpha > \beta$ and $(\alpha - \beta)/\alpha > \gamma$, the host will declare itself a new root node and invokes the spanning-tree creation operations. In these relationships, γ controls the strictness level of new root host creation.

LCMRMG uses a packet forwarding algorithm that is slightly different from the packet forwarding algorithm used in [48]. This new packet forwarding is due to a node's membership of multiple trees.

For each incoming message, every host **s**:

I. Performs the traffic locality caching algorithm as described above

II. Routes the message accordingly:

   a. If the destination host of the message is in one of its sub trees, host **s** simply forwards the message to the root of that sub tree.

b. Or else forward the message to the parent host that has the lowest generation number.

In the following section we present a brief analysis of the LCMRMG. This analysis focuses primarily on the locality caching mechanism used and the published simulation study of the authors.

3.3.1 Analysis of LCMRMG

Three of the issues identified in [48]: Assumption that nodes do not fail, Nodes have to notify their neighbors before and after the movement and The time required in building the spanning tree, remain part of LCMRMG. The only issue addressed is the root node issue: LCMRMG uses locality caching to identify nodes that can be converted into roots to reduce the network congestion.

(a) Analysis of Locality Caching Used in LCMRMG

The expression used in locality caching algorithm, $(\alpha - \beta)/\ \alpha > \gamma$ indicates whether a node should become a root or not. Analyzing the LCMRMG algorithm and Figure 3.3 it can be deduced that $\alpha = \alpha + (l_1 + h_1 + h_2)$, while the initial value of α is zero.

After the node **s** receives the first message, $\alpha = (l_{1i} + h_{1i} + h_{2i})$ where $i = 1,2,3...ct$. Similarly $\beta = \beta + (l_{1i} + l_{2i})$ where $i = 1, 2, 3...ct$.

Therefore: $\alpha - \beta = h_{1i} + h_{2i} - l_{2i}$

$$(\alpha - \beta) / \alpha = (h_{1i} + h_{2i} - l_{2i}) / (l_{1i} + h_{1i} + h_{2i})$$

We know that the sum of any two sides of a triangle is larger than or equal to the third side, equal when the three vertices of the triangle are on a straight line (Considering a straight line as a special triangle).

The author of LCMRMG found that the number of roots formed were very high (up to 32) when $\gamma > 30\%$ and new roots are rarely created when $\gamma > 50\%$. In the conclusion of the study, Xin Zhang (2005) suggested that in a network of 1000 nodes, about 5 roots are sufficient. Although the authors did not mention what was the exact γ value used in their experiments when they achieved 5 roots, from the above statement it sounds logical to assume the γ value should have been very close to 50%. Therefore: $(\alpha - \beta) / \alpha = (h_{1i} + h_{2i} - l_{2i}) / (l_{1i}+h_{1i}+h_{2i}) > 0.5$

The above expression implies that the positions of the nodes **a, s, r,** and **b** should be such that:

- l_{1i} should be a very small value, which implies the node **s** should be very close to the source node **a**.
- Similarly, l_{2i} should be very small so that the numerator value remains relatively high and when divided by the expression $(l_{1i} + h_{1i} + h_{2i})$ it yields at least 0.5.

Combining the above two statements, we can say: $(h_{1i} + h_{2i} - l_{2i})$ should be at least half of $(h_{1i} + h_{2i})$.

Consider the triangle formed by nodes **s, r** and **b**. If node ***a*** is communicating with node ***b*** through its closest neighbor node ***s***, and this is the only traffic through the node ***s*** at this time. Then:

$$(\alpha - \beta) / \alpha = (h_1 + h_2 - l_2) / (l_1 + h_1 + h_2)$$

$$\approx (h_1 + h_2 - l_2) / (h_1 + h_2), \text{ for a small value of } l_1$$

Below is a brief study of how this expression controls the caching behavior of the LCMRMG algorithm[2]:

---

[2] NOTE: In an expression of the form $(h^2 + j^2) / 2hj$, assuming that $h \geq j$ when both h and j are unknown, $h = (j + k)$, where $k \geq 0$

$(h_1 + h2 - l_2) / (h_1 + h_2) > \gamma$

$(h_1 + h_2 - l_2) > (h_1 + h_2)\ \gamma$

$(1 - \gamma)\ (h_1 + h_2) - l_2 > 0$

$(1 - \gamma)\ (h_1 + h_2) > l_2$

The angle **L** between h_1 & h_2 can be worked out by using the cosine equation:

$l_2^{\,2} = h_1^{\,2} + h_2^{\,2} - 2h_1\, h_2 \cos L$

$L = \arccos\ (\ (h_1^{\,2} + h_2^{\,2} - l_2^{\,2}) / 2h_1\, h_2\ )$

The angle **L** is large when the value of $(h_1^{\,2} + h_2^{\,2} - l_2^{\,2}) / 2h_1h_2$ is the smallest i.e when $h_1 = h_2$ and l_2 is at its maximum. Therefore, when

$h_1 = h_2 = h,$

$l_2 = (1 - \gamma)\ (h_1 + h_2) = 2h\ (1 - \gamma)$ and the value of the expression in the

arc cosine $(h_1^{\,2} + h_2^{\,2} - l_2^{\,2}) / 2h_1\, h_2$ Thus, $((h_1^{\,2} + h_2^{\,2} - l_2^{\,2}) / 2h_1\, h_2\ ) = (\ -1 + 4\gamma - 2\gamma^2)$

So the expression can be stated as
$[(j+k)^2 + j^2] / 2(j+k)j$
$= (2j^2 + 2jk + k^2) / (2j^2 + 2jk)$
$= 1 + [k^2 / (2j^2 + 2jk)]$, this expression being smallest when $k = 0$
Therefore, $(h^2 + j^2) / 2hj$ is smallest when $h = j$

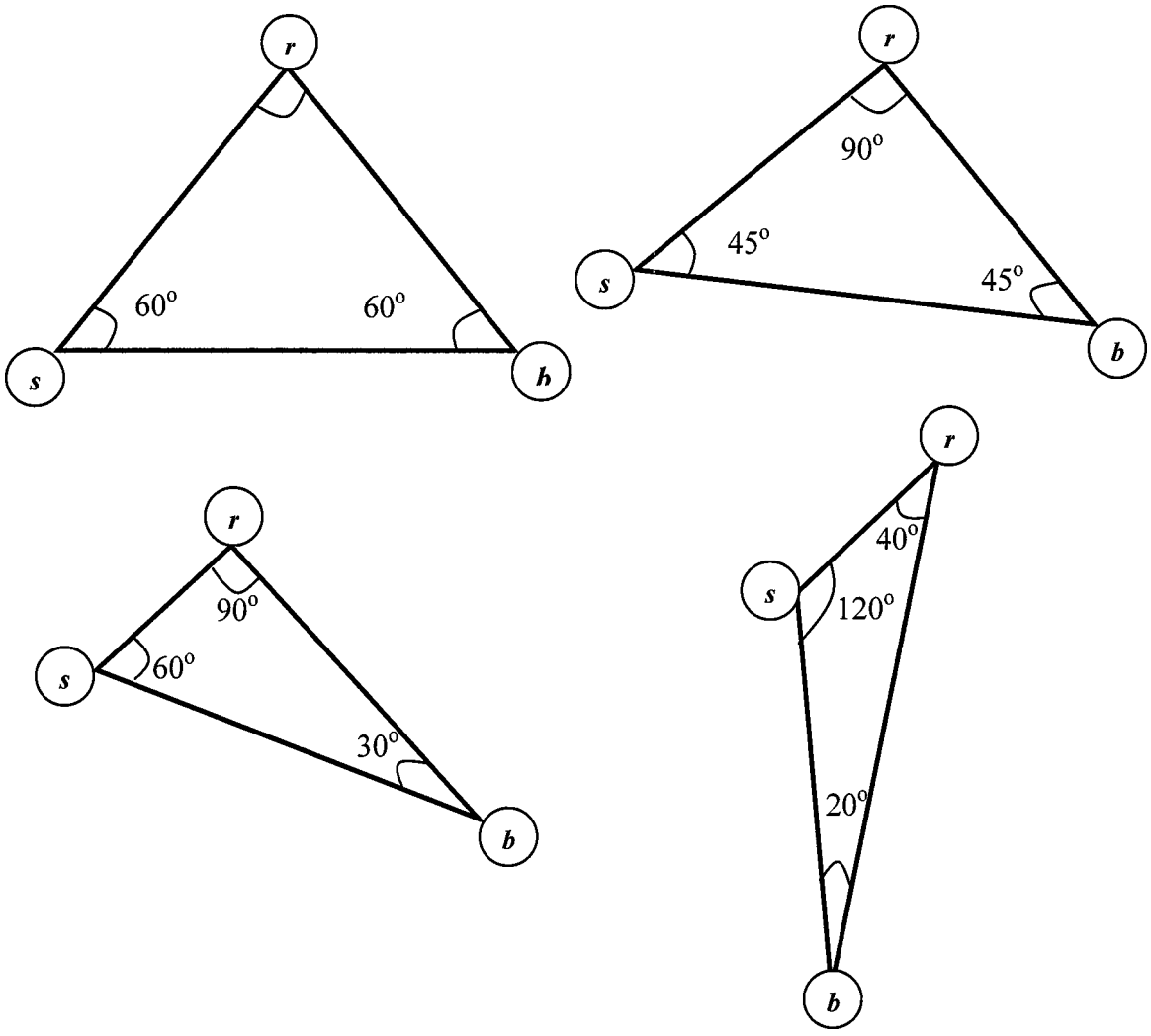


Figure 3.4: Triangles in which, $(\alpha - \beta) / \alpha$ is less than 50%.

For each value of γ there is an upper limit on the angle between $\mathbf{h_1}$ and $\mathbf{h_2}$. For instance,

If $\gamma = \mathbf{50\%}$, then the angle **L** is less than 60^0

If $\gamma = \mathbf{30\%}$, then the angle **L** is less than 83^0

As the angle between the sides represented by $\mathbf{h_1}$ and $\mathbf{h_2}$ increases the length of $\mathbf{l_2}$ increases and the value of $(\mathbf{h_{1i} + h_{2i} - l_{2i}})$ decreases. Shown in Figure 3.4 are several examples for which the value of $\mathbf{(\alpha - \beta) / \alpha < 50\%}$ and **s** in those cases will not become a root, although it could have become a root to help bypass the traffic from the original root.

This suggests that for $\gamma = 50\%$, **s** becomes root only when the angle between $\mathbf{h_{1i}}$, and $\mathbf{h_{2i}}$ is less than 60°. Therefore, LCMRMG algorithm does not necessarily cache all the shorter alternatives nor does it cache all the localized traffic surges in the network, it all depends on the value of γ used and type of localized traffic patterns in the network.

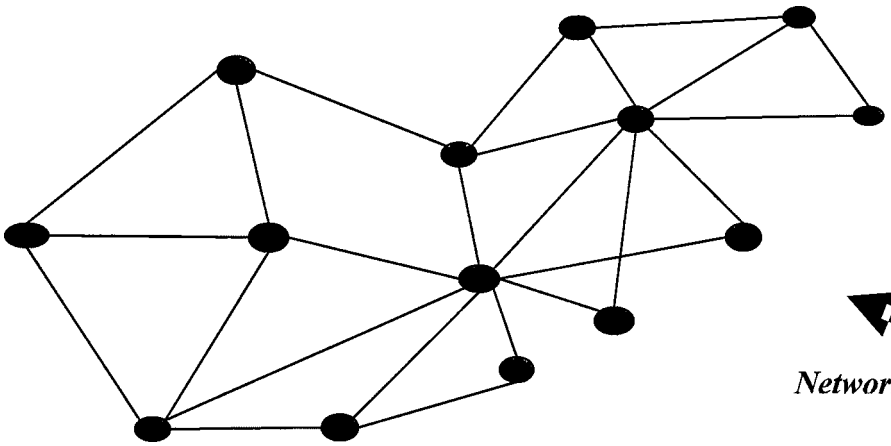
(b) Control Overhead of LCMRMG

As shown in the Figure 3.5, the idea of Chen's technique is to reduce the maintenance overhead by maintaining only a small subset (spanning tree) of the total number links in the network graph. On the other hand, LCMRMG algorithm achieves shorter routes by building new trees. As more and more trees are added, more and more links in the network graph are now maintained by the LCMRMG algorithm. As illustrated in Figure 3.6, we can clearly observe a much bigger set of links is maintained by the LCMRMG technique leading to a significantly higher maintenance overhead.

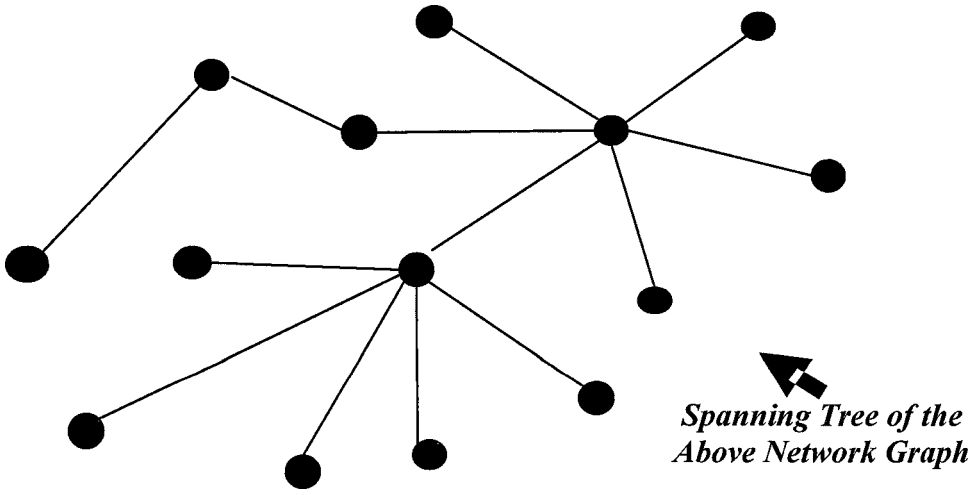
Building a new spanning tree rooted at a different node, not only involves the control overhead for building the spanning tree, but also the control overhead required to keep up with the changes in the network. Using multiple spanning trees for routing can be justified with very heavy local traffic, i.e., the number of messages that are routed in a small region of network is very significantly higher than the control overhead created by the multiple spanning tree maintenance.

MANET scenarios with heavy traffic between few local mobile stations are not suitable candidates for LCMRMG, or any other proactive technique. Reactive protocols should be the choice in such situations.

Another important observation we made is the possibility of LCMRMG adding unnecessary spanning trees.



Number of Nodes	Number of Links
14	24

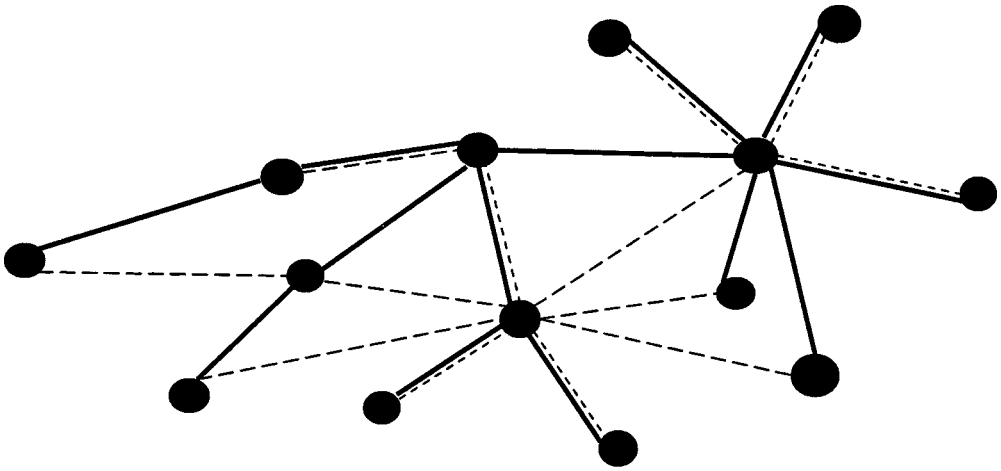


Number of Nodes	Number of Links
14	13

Figure 3.5: A Network graph and its spanning tree.

As described at the beginning of this section, each node in LCMRMG maintains a

counter variable called ***ct*** to keep track of number of packets passing through it.



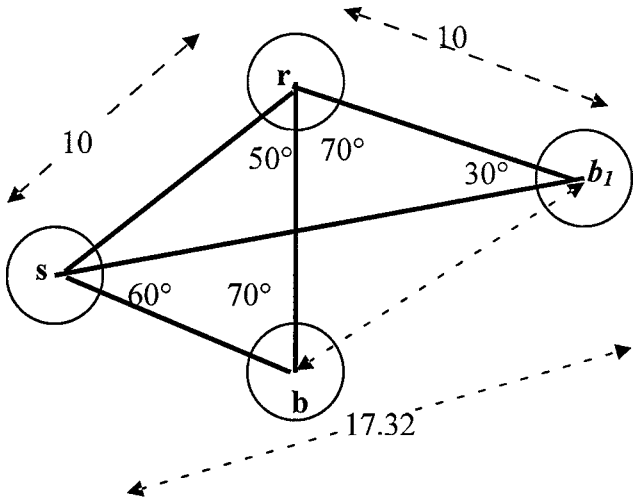
Number of Roots	Number of Links
1 (dotted edges OR solid edges)	13
2(dotted edges AND solid edges)	19
Total number of links in the network graph	24

Figure 3.6 LCMRMG network with two roots.

The value of $(\alpha - \beta) / \alpha$ is calculated when the number of packets reaches ***ct***. The authors of LCMRMG seem to provide no indication of how big or how small the value of **ct** should be. For a small value of **ct** too many spanning trees will be added to the routing protocol, while a large value can lead to situations explained in the following example.

Example: As shown in Figure 3.7, consider node **s** that has already routed **(ct – 1)**

messages to the destination node **b**, through the current root node **r**. If the node **b** moved to a position indicated by node $\mathbf{b_1}$, before the $\mathbf{ct^{th}}$ message, it is routed through **r** to the new location. Although the new location indicated by $\mathbf{b_1}$ is not physically close to **s**, the value of $(\alpha - \beta) / \alpha$ is quite over shadowed by the first $(ct - 1)$ messages. As the value of $(\alpha - \beta) / \alpha$ is greater than γ for the first $(ct - 1)$ messages, it remains greater than γ after the $\mathbf{ct^{th}}$ message.

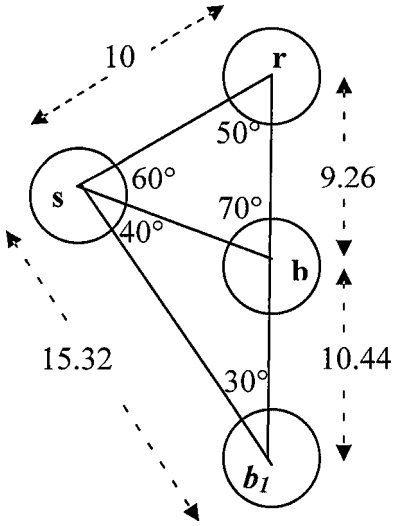


For **Δsrb,**

$$\frac{\alpha-\beta}{\alpha} \approx \frac{h_1+h_2-l_2}{h_1+h_2} = 0.57$$

For **Δsrb_1,**

$$\frac{\alpha-\beta}{\alpha} \approx \frac{h_1+h_2-l_2}{h_1+h_2} = 0.134$$



For **Δsrb,**

$$\frac{\alpha-\beta}{\alpha} \approx \frac{h_1+h_2-l_2}{h_1+h_2} = 0.57$$

For **Δsrb_1,**

$$\frac{\alpha-\beta}{\alpha} \approx \frac{h_1+h_2-l_2}{h_1+h_2} = 0.48$$

Figure 3.7: Analysis of LCMRMG.

In such a situation, according to the LCMRMG routing, the node **s** proceeds to create a

new spanning tree, such spanning trees simply add to the overhead rather than bypassing the traffic through the root node.

3.3.2 Analysis of LCMRMG Simulation Results

In this section we analyze the simulation results presented in [49]. The authors of LCMRMG have simulated their protocol as well as Chen's technique, using their own simulation platform ST-SIM. Details of ST-SIM are available in [49].The graph in Figure 3.8, from [49], depicts the delivery ratio of the single root package routing algorithm and that of the multi root LCMRMG algorithm. ST-SIM does not have the capability to simulate any particular mobility model. Similarly the simulation software used does not simulate any particular traffic pattern. There seems to be no input parameter that could be modeled against time.

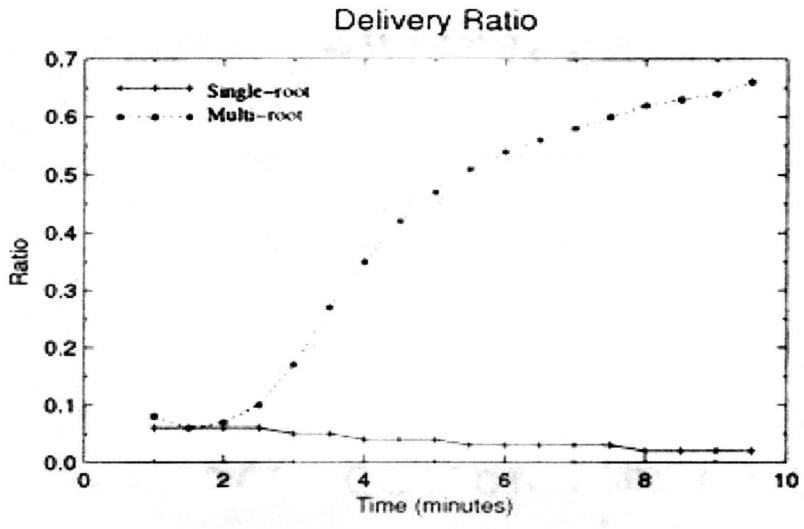


Figure 3.8: LCMRMG, Delivery ratio vs. Time from [49].

Using the graph displayed in Figure 3.8 the author makes two deductions:

I. The delivery ratio of Chen's technique degrades as the simulation time

increases.

II. The delivery ratio of the multiple root LCMRMG algorithm scales up as the

simulation time increases.

But clearly, the routing protocol performance is not function of time. It is in fact other factors such as mobility, traffic pattern and number of roots etc.

Performance of proactive protocols decreases with the increase of number of nodes or mobility in the network. Such degradation of the performance is due to increase in the control overhead. Due to the greater number of links maintained by LCMRMG in comparison to Chen's technique: Not only the links in Chen's technique, but also the link changes in the graph that might have not affected Chen's technique will affect LCMRMG. Therefore, in LCMRMG, as the number of nodes increases, the control overhead will increase and the performance of will decrease.

In contrary to the above argument the authors of LCMRMG, in their simulations claim that the number of nodes does not affect the performance of LCMRMG protocol.

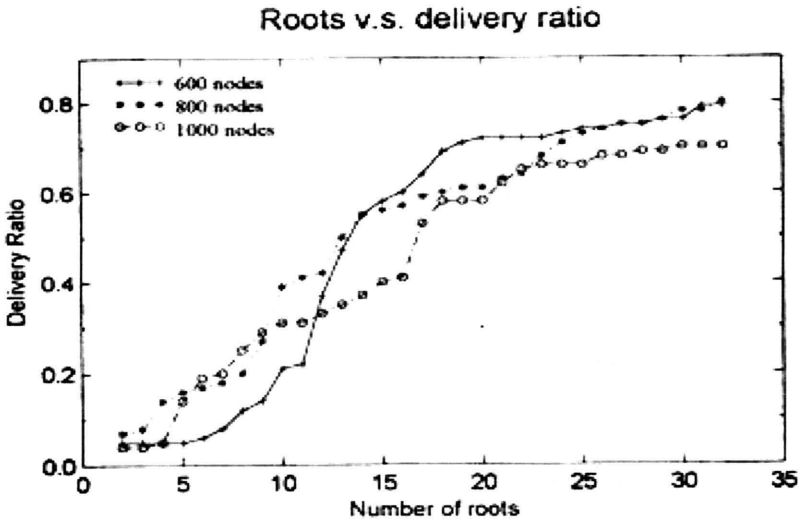


Figure 3.9: LCMRMG, Delivery Ratio vs. Number of Roots from [49].

According to the graph presented in the Figure 3.9 from [49], it appears that the number of nodes does not affect the delivery ratio of LCMRMG protocol. However, the graph suggests that the Delivery ratio will increase as the number of roots increase. This is in clear contradiction of their claim that the performance does not increase beyond a certain number of roots, such as the performance of a 1000 node network does not increase beyond 5 or 6 roots.

Shown in Figure 3.10, is another graph presented by the authors of LCMRMG. The authors have explained this graph as indicative of "no effect of increase in number of roots after a certain threshold".

As quoted by the authors of [49] & [51], "An interesting issue is the number of root hosts needed to support efficient operations in a given MANET. Intuitively this number should be dependent on several factors, including the size of network (number of hosts), host positions relative to each other, their moving speed, the radio transmission strength, and the geographic spread of the hosts. However, once all these factors are fixed, there ought to be a threshold, beyond which adding more roots only increases routing overhead rather than improving the performance. Our simulation confirms this. Figure 3.10, from [49] shows that for a typical MANET under our simulation, the total number of root hosts needed is around five to six."

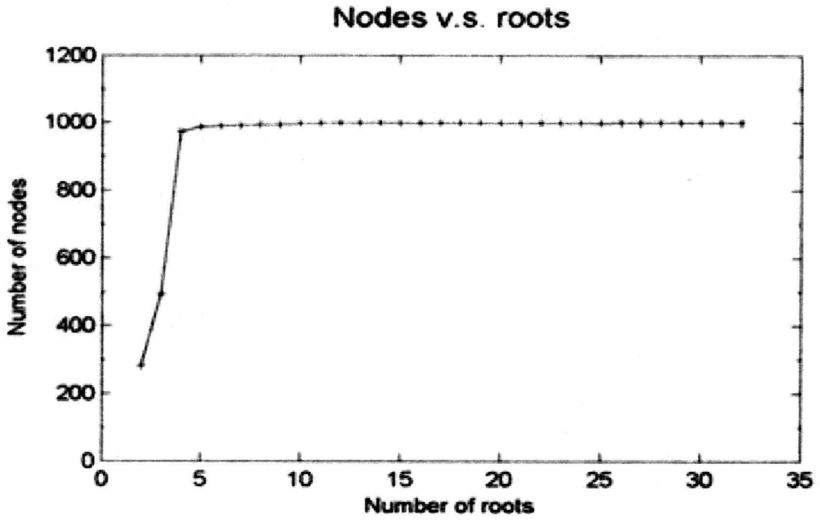


Figure 3.10: LCMRMG, Number of Nodes vs. Number of Roots, from [49].

To the contrary, Figure 3.10 suggests that the number of roots remains around five for networks below 1000 and the number of roots for a 1000 node network can be anywhere between 4 and 32. The graph does not suggest what the authors have claimed that it does.

At the beginning of [49] & [51] the authors state that the larger physical distances are indicative of routes with higher hop count. Therefore, in LCMRMG, due to traffic locality caching and building newer spanning trees the messages are routed over shorter routes. On the contrary the graph in the Figure 3.11 from [49], suggests that the routes in the single root package routing algorithm are significantly smaller than the routes in the LCMRMG algorithm.

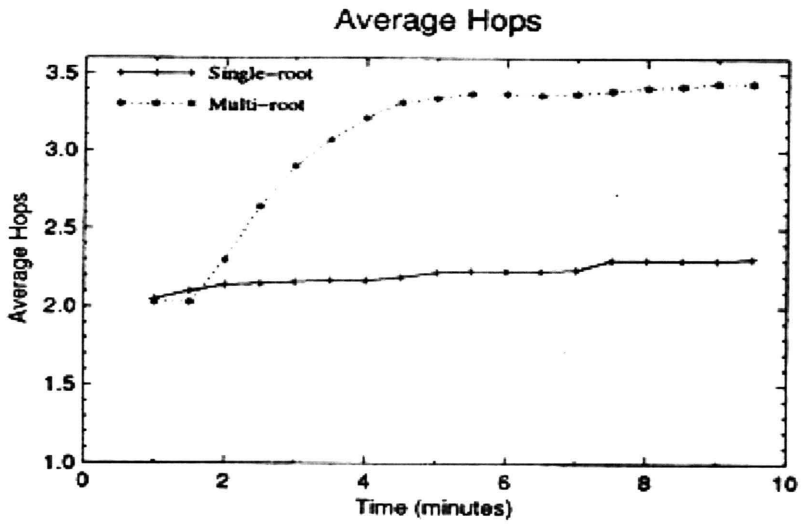


Figure 3.11: LCMRMG, Average number of Hops vs. Time, from [49].

3.4 Locality Caching Multi-Root Multi-Generation Routing With Color Schema Routing (LCMRMGCS)

In [50] & [52], the authors have presented a new routing protocol called LCMRMGCS. This protocol according to the authors extends and significantly enhances the routing algorithm presented in [49], which they achieve by using non-overlapping trees.

At the beginning, the information maintained by the LCMRMGCS routing algorithm is very similar to the spanning tree structure of Chen's technique. More colored trees are created as the routing proceeds and local traffic patterns are discovered in the network. Traffic locality caching technique used in LCMRMGCS is same as the one used by LCMRMG algorithm.

A node, say node **X**, elects to become a new root according to the locality caching algorithm in [49]. Node **X** then sends a message to the node $\mathbf{A_0}$ to apply for a unique color.[3] Upon receiving designated unique color, say black, from the $\mathbf{A_0}$, node **X** performs

[3] NOTE: A0 is the root of the forest and so a special root node. It assigns colors to new roots as they form and maintains

the following actions:

- Inform all its descendant nodes to switch their colors to black[4].
- Inform all nodes on the path from node **X** to the $\mathbf{A_0}$ to add color black as one of their other colors.
- Inform all nodes on the path from node **X** to the $\mathbf{A_0}$ to purge all information about node **X** and its descendant nodes from their generation tables.

An LCMRMGCS node first discovers the color of the destination before it can start sending packets to that destination. This color discovery is handled by the color discovery algorithm described below:

(1) A node, say node X, wishes to find the color of a node Y. X sends a color inquiry message (CIM) which contains its own ID and color, and node Y's ID, starting from itself.

(2) When a CIM message arrives at any node A, it will do the following:

- If node A is a root node and its color matches the requested color, or node A's color matches the inquired color and node A's generation table contains the color information of the inquired node, A sends a positive reply using the routing algorithm (to be presented next). Color discovery completes.
- If node A has other colors, forward the CIM message to each of its child nodes (except the child at the incoming link) that has at least one other color.
- If the CIM message does not come from parent node, and node A has parent node, forward the CIM message to its parent.

---

an entry for each color in the forest

[4]NOTE: at any root node which is a descendant of node X, this color switching request will stop propagating further down

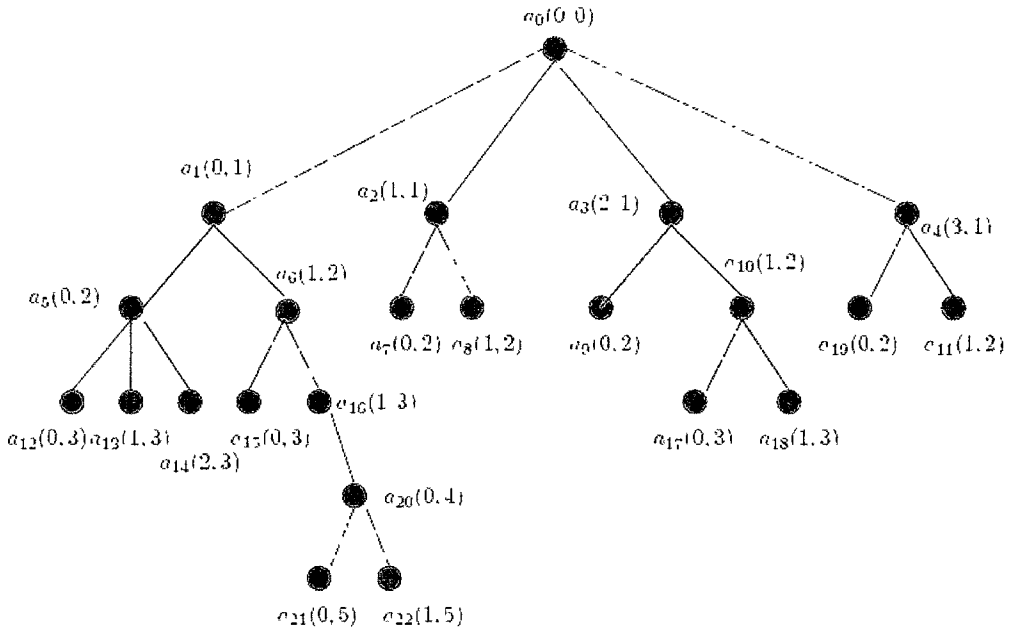


Figure 3.12: Initial state of the forest in LCMRMG from [52].

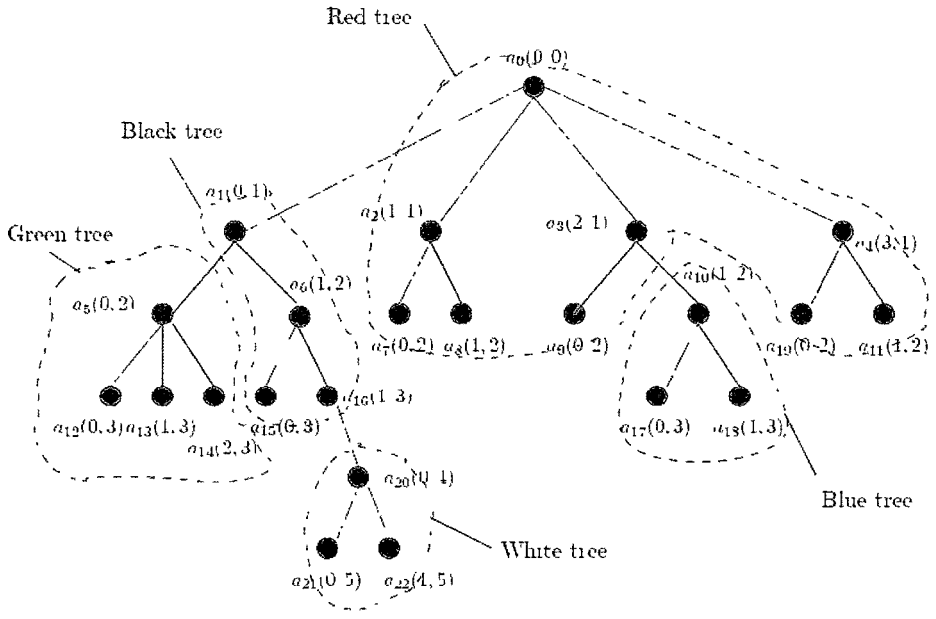


Figure 3.13: An example network of five color trees from [52].

Once the color of the destination node is discovered, nodes in LCMRMGCS use a packet forwarding technique described in the flow chart of Figure 3.14 to forward the packets.

Shown in Figures 3.12 is a sample network as maintained by the LCMRMGCS at the beginning of the routing, which is a spanning tree of the network graph. Shown in Figure 3.13 is a forest of several colored trees. This forest is a consequence of nodes $\mathbf{a_0}$, $\mathbf{a_1}$, $\mathbf{a_5}$, $\mathbf{a_{10}}$ and $\mathbf{a_{20}}$ forming a tree of their own color after detecting local traffic patterns. An interesting observation here is: at any given time, the total number of links maintained by LCMRMGCS is equal to the total number of links maintained by Chen's technique.

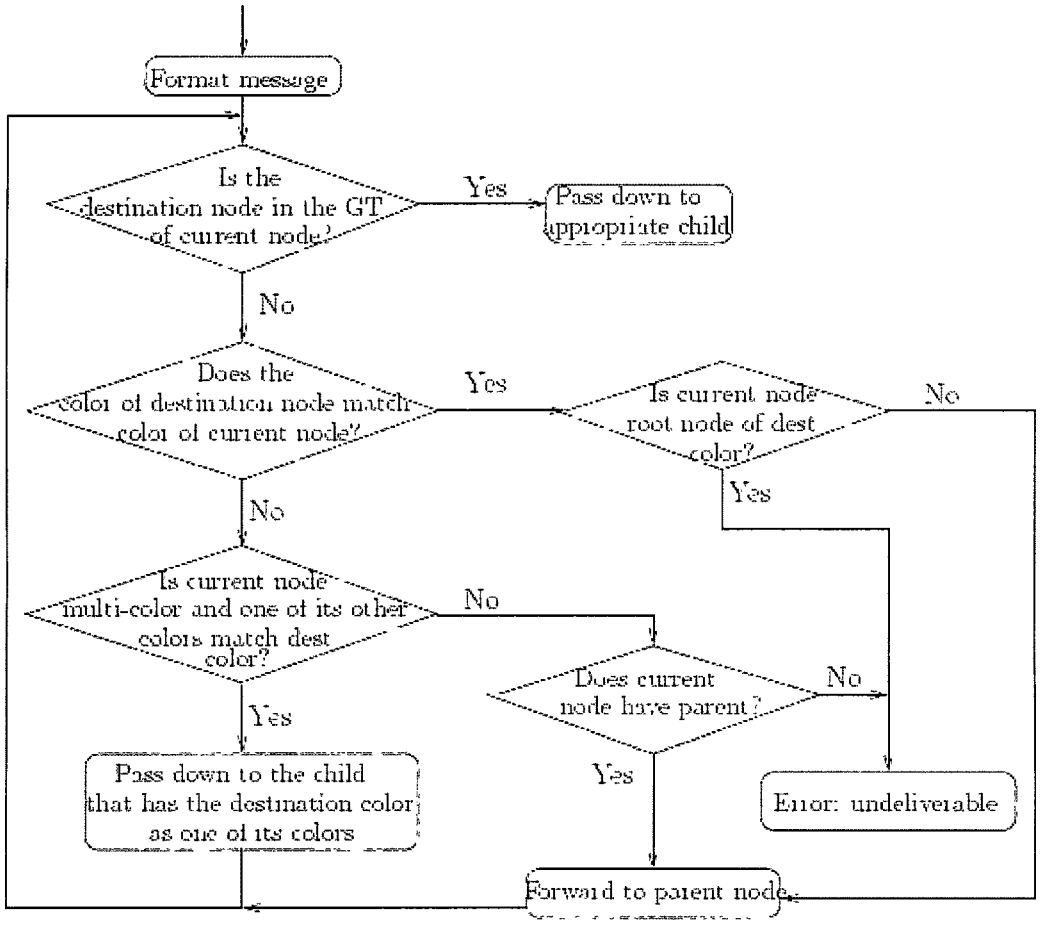


Figure 3.14: LCMRMGCS routing technique from [52].

3.4.1 A brief analysis of the LCMRMGCS algorithm

In this section we present a brief analysis of the LCMRMGCS and its

characteristics as presented in the original literature [52]. Three issues identified in both [48] and [50]: Assumption that nodes do not fail, Nodes have to notify their neighbors before and after the movement and The time required in building the spanning tree, remain part of LCMRMGCS. The only issue addressed is the root node issue; LCMRMGCS uses locality caching to convert the spanning tree of the network graph into a forest of colored spanning trees, supposedly for better routes.

a) Criterion for New Root (Colored Tree) Creation

The authors in their literature have claimed that the LCMRMGCS considers whether there exists a link between the two nodes rather than the real physical positions for creating new roots. But what we can observe in LCMRMGCS is that the nodes make decision of weather to become a root or not solely based on the same scheme presented in LCMRMG algorithm, which is completely based on the real physical positions of the nodes and does not check if there exists a link between the two nodes under consideration. This is completely contrary to their claims.

b) Support for Node Failure

The LCMRMGCS algorithm makes no provision to support a node failure. When the A_0 node fails, the protocol simply comes to a halt.

c) Total Number of Links Maintained By The Routing Algorithm

When we look at the number of links maintained by the routing algorithm in comparison to total number of links in the network graph, we find one thing in common between LCMRMGCS and the single root package routing algorithm in [49]. The total number of links in both routing algorithms at any given instance is equal. Figures 3.15, shows a Network Graph, Spanning Tree Network, LCMRMG Network and

LCMRMGCS Network, for a sample network.

The number of edges in a connected graph with no cycle (tree) is equal to the number of vertices minus one. Chen's technique maintains a spanning tree of the network graph. The LCMRMGCS algorithm on the other hand maintains a connected component of the graph, which is a forest of trees with no cycle. Therefore the total number of links in both LCMRMGCS and Chen's technique is: ((total number of nodes) – 1).

d) Impact of Node Mobility

As a node moves from one colored tree to other colored tree, the only nodes that are aware of such transition are the roots of the former and the later trees. Rest of the nodes in the network, if they have previously communicated with the node that has now changed its color, they now hold stale information. This can cause forwarding of messages to wrong neighbors. For instance in the Figure 3.13, if node $\mathbf{a_{22}}$ left its parent $\mathbf{a_{20}}$ and joined node $\mathbf{a_{18}}$, then the only nodes aware of this are nodes $\mathbf{a_{20}}$ and $\mathbf{a_{10}}$. If node $\mathbf{a_1}$ (or any other node that is not a white or blue node), has previously communicated with node $\mathbf{a_{22}}$, still assumes that $\mathbf{a_{22}}$ is a white node and forwards all the messages routed to $\mathbf{a_{22}}$ towards $\mathbf{a_{20}}$, which is a spurious route. If this happens at the beginning of the communication session, then the communication session has to wait until other trees are explored for the desired destination. In the worst case this may lead to a lookup in the entire tree present. Similarly, when the above described scenario arises while the communication session is in progress, this can potentially lead to a very long wait time for the route repair.

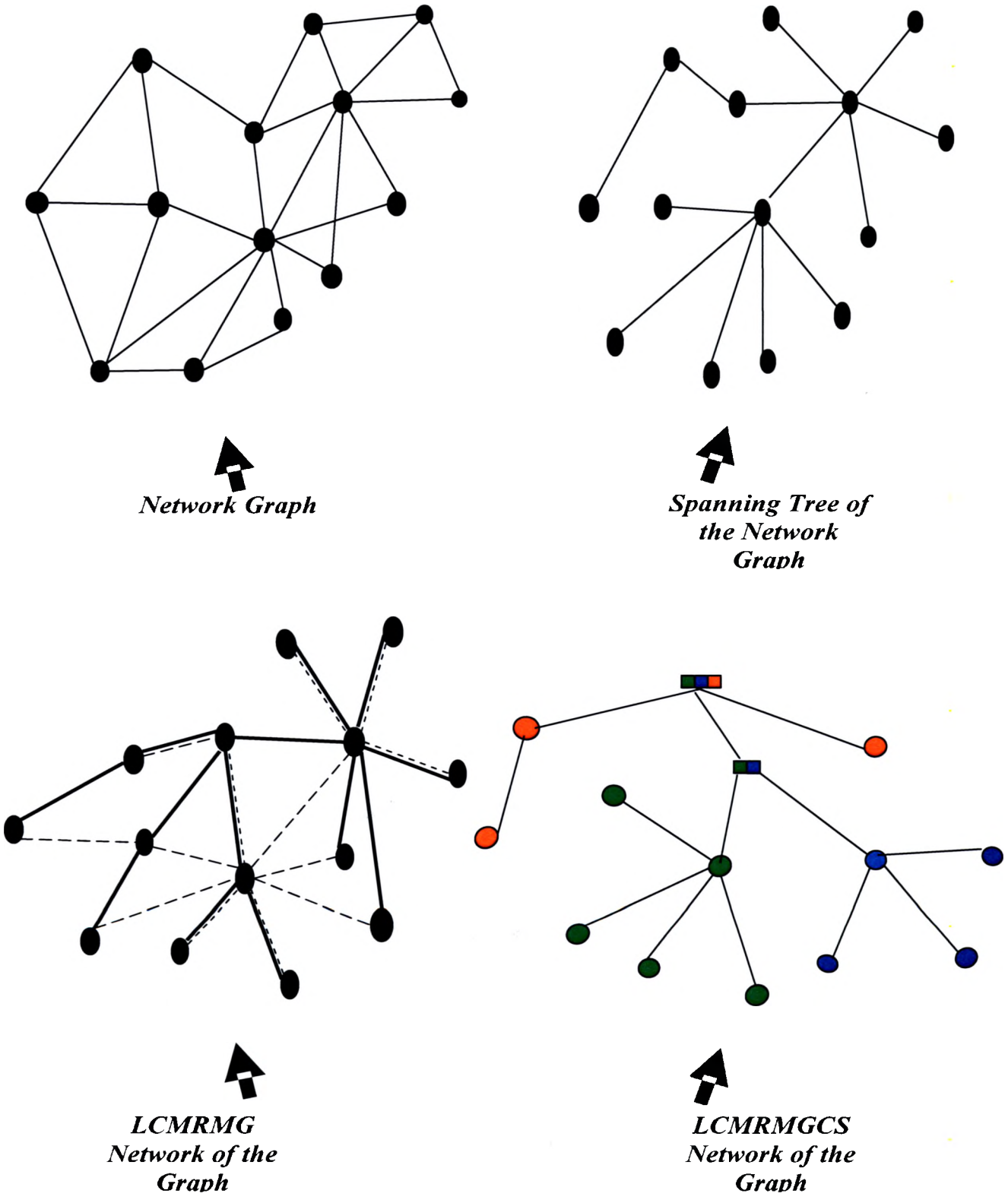


Figure 3.15: Network Graph, Spanning Tree Network, LCMRMG Network and LCMRMGCS Network.

e) Proactive and Reactive Components of the Routing Protocol

Of the three protocols, Chen's technique, LCMRMG and LCMRMGCS, the only protocol with a reactive or dynamic component in the routing algorithm is LCMRMGCS. Although there is a dynamic spanning tree creation in the LCMRMG algorithm, once such trees are created they are maintained statically. In LCMRMG, at any given time, at least the root nodes in the network have knowledge of rest of the nodes in the network.

In LCMRMGCS, each root only maintains the nodes that are of its own color. Roots do not maintain the lower generations (generations further away from root and have higher generation numbers), unless, such nodes are of the same color as the root. Therefore, each source before it can communicate with a destination should know the color of the destination, unless, the destination is in its generation table. What appears from [49] & [52] is: a node can proceed with its knowledge of the color of the destination due to a prior communication session or the node shall find the color of the destination using Color Discovery Algorithm presented in [52].

The only dynamic part of the LCMRMGCS algorithm is the Color discovery algorithm, i.e., finding which tree in the forest, does the destination belong to. The information of the path towards the root of a certain tree is available at the most at A_0.

In dynamic protocols, for instance AODV, DSR etc, during a communication session, if the path to the destination breaks due to the mobility of the source, destination or any intermediate node, the routing algorithm tries to repair such routes in AODV, or the redundant routes maintained by DSR (promiscuous mode), help in minimizing the interruptions to communication session. However, in LCMRMGCS, if a destination changes its color during a communication session there is no route repair nor there are

any alternative cached routes and what makes things worse is there is no single node like the root in LCMRMG or package routing algorithm where all the updates converge. Therefore, in such situations, the communication session has to halt until all the trees in the network are searched for the destination and it gets worse as the number of trees increases in the forest.

f) Routing Protocols Ability to Bypass A_0

In the above Figure 3.16 from [52], what we can observe is, the generation table of node A_0 maintains the color of the tree it belongs to as well as the other colors, indicating the other color trees within its descendants. Compare the number of colors in the Figure 3.13 with all the colors in the generation table of A_0, they are equal.

ID	A_0	
Parent ID	NIL	
Generation no.	0	
A_0's child ID	0	
A_0's own color	red	
A_0's other colors	green,blue,white,black	
Future generation		
Child ID	ID	Future g.
0	A_1 (black-green-white)	NIL
1	A_2 (red)	A_7, A_8
2	A_3 (red-blue)	A_9
3	A_4 (red)	A_{10}, A_{11}

Figure 3.16: Generation Table of A0, from [52].

Any node that is on the path between two root nodes, forming two different colored trees, is always involved in all of the communications between any pair nodes, one from each tree. Node A_0 like the root node in the package routing algorithm remains part of the communication sessions between the nodes in the sub trees rooted at the one hop neighborhood of it.

g) Analysis of LCMRMGCS Simulation Study

In [52] the authors have presented a simulation study of LCMRMGCS algorithm. The simulator used: ST-SIM, is developed by them. The simulation software used, does not have the capability to simulate any particular mobility model. Similarly the simulation software used does not simulate any particular traffic pattern. It appears like the mobility and communication are part of the node itself. Mobility pattern or the communication sessions can not be input into the simulator. The simulator does not have all layers of the MANET communication protocol stack, it only implements the routing protocol. Missing layer are: Physical layer (Which is useful for study of radio and battery related data), Link layer (All the collisions and contentions due to heavy traffic (both data and control traffic) in the network are encountered in this layer) and Application layer (There are no protocols like http, ftp, telnet, etc, to study the protocol behavior for reliable and unreliable services).

(i) Received Messages vs. Maintenance

The authors have defined Maintenance[5] as: the measure of accumulated number of changes in the generation tables of all nodes caused by sign-on and sign-off procedures in updating the network topology.

[5] NOTE: It is very essential to note that the so called maintenance here is not a measure of control traffic.

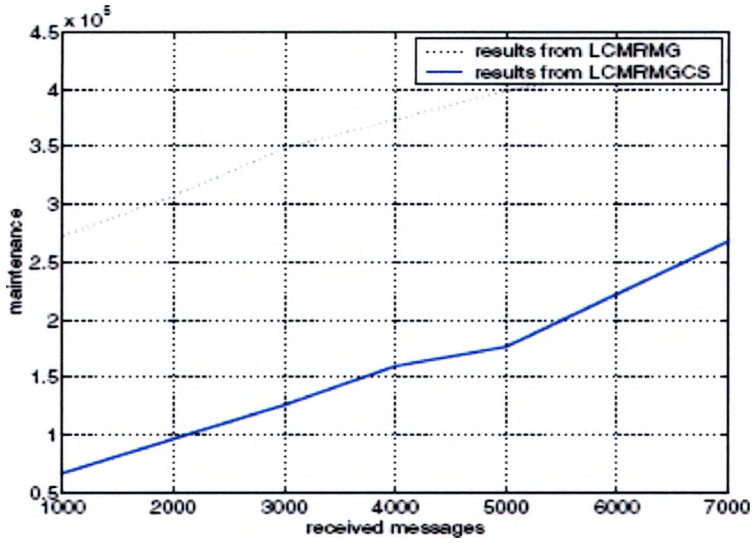


Figure 3.17: Received Messages vs. Maintenance, from [52].

From our study of LCMRMG in the previous section and the study of LCMRMGCS, we know that there is no dynamic component in the LCMRMG. Therefore the number of changes in the generation tables is only a function of number of roots (keeping transmission range constant) as long as the impact of mobility is the same as time progresses. But it is hard to arrive at such a conclusion, because the mobility used in the simulation is random movement. In any case both LCMRMG and LCMRMGCS start with one root and then they proceed with the creation of more roots based on the same criterion presented in LCMRMG. More roots are only created after there has been a considerable amount of data flow within relatively small regions of the network. Therefore the maintenance of both protocols should be quite comparable at the beginning of the simulation, i.e., when the number of packets received is close to zero, both protocols should show very similar maintenance values.

Looking at the slopes of the two curves (almost lines) in the Figure 3.17, it appears that the curve for LCMRMG can only meet with the curve for LCMRMGCS, if the LCMRMG curve took a steep upsurge between 0 and 1000 received messages.

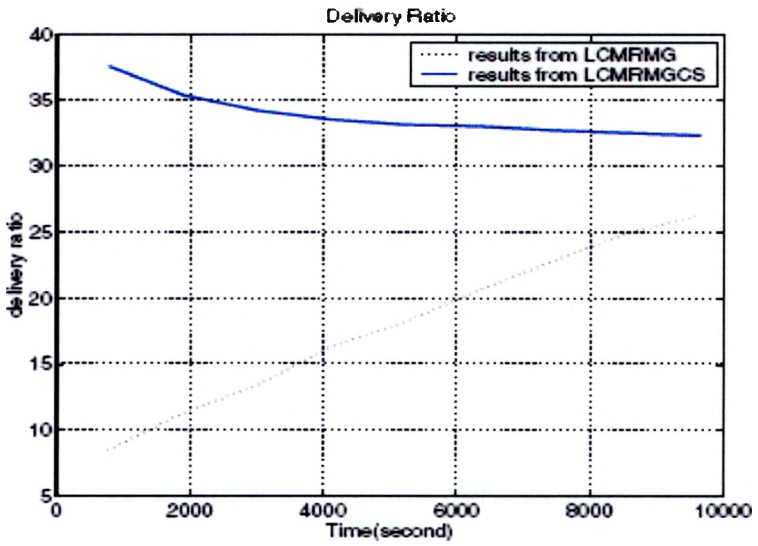


Figure 3.18: Delivery Ratio vs. Time from [52].

(ii) Delivery Ratio vs. Time

Once again like the authors of LCMRMG, the authors of LCMRMGCS have presented a graph to show, how the delivery ratio of their protocol scales up over time. This claim does not sound very logical. If in case the performance of the LCMRMGCS increases with the number of roots, one can say the performance increased over time as more and more roots are created. The authors also claim that the performance drops beyond a certain number of roots. Now as the time progresses, the only thing that can change in their simulation is the number of roots (mobility, number of nodes etc, doesn't quite change). But if the number of roots reaches the threshold, the delivery ratio can not increase, it can only decrease.

(iii) Delivery Ratio vs. Number of Roots

In [52], the authors claim that the delivery ratio is a function of number of roots of a given MANET. They claim under a certain threshold, the more the number of roots the better the delivery ratio.

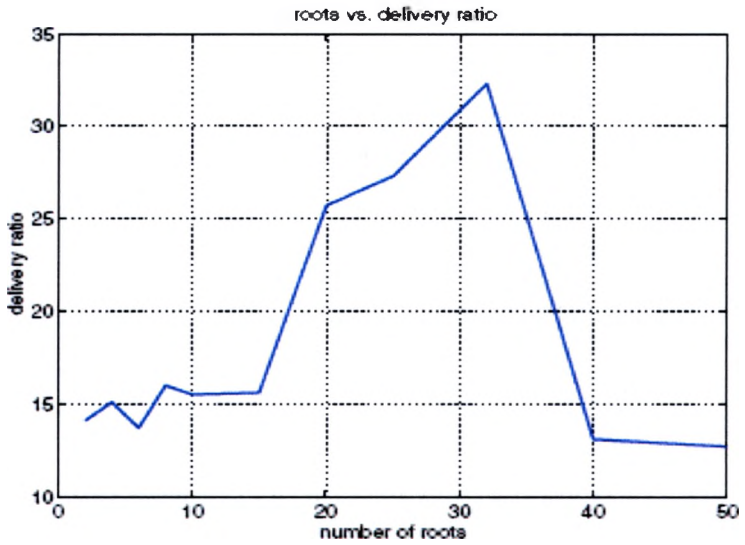


Figure 3.19: Delivery Ratio vs. Number of Roots from [52].

The LCMRMGCS protocol starts with one colored tree, which is equivalent to package routing presented in [48]. When there are traffic patterns in the network, which require a significant message flow between two nodes that are relatively close (physical distance) compared to the sum of distances between the root, source and root, destination. When new roots are formed, LCMRMG adds more links to the protocol, but the LCMRMGCS does not add any new links and just separates the newly formed tree rooted at the new root from the root of the forest and rest of the forest. As the number of roots increases, the dynamic control overhead increases and the static control overhead decreases. There seems to be no convincing reason for higher delivery ratio.

(iv) Hop Count vs. Number of Roots

Once again upon creation of new roots the LCMRMGCS does not add any new links to the initial tree structure and therefore LCMRMGCS route lengths do not increase or decrease with a change in the number of roots in the network.

3.4.2 Observations

Few important observations, we made, from the discussion of LCMRMGCS protocol in the above section:

- The total number of links maintained by the protocol is equal to the total number of links maintained by the package routing technique
- LCMRMGCS stores spurious routes (due to stale color information)
- There is route acquisition latency
- Routes are not shorter than package routing technique

In light of the above four properties of LCMRMGCS, the claim that it can achieve, performance comparable to LCMRMG is highly questionable.

CHAPTER 4

ENHANCED PACKAGE ROUTING TECHNIQUE

In this Chapter we present an enhanced version of Chen's technique called Enhanced Package Routing Technique (EPRT). EPRT overcomes the issues identified in all three protocols discussed in Chapter 3. EPRT uses a dynamic root and therefore different nodes become the root as the topology changes. To address the root over load issue EPRT uses promiscuous mode operation to bypass the root node in many cases. Except the center node, each node in EPRT maintains the routing information of only a small subset of mobile stations that are further away from the center. In a network with no partitions, this protocol guarantees a route between every pair of mobile stations.

What started of as a pursuit of heuristic method to find a node that best fits as the root node, resulted in the discovery of EPRT. The initial idea of our work was to improve Chen's technique by making a better root node choice than a random node. An obvious solution was to use the tree center as the root. But Chen's technique runs on a static root, once a node is designated as root it remains the root for the life of the network. After a few topological changes the root node will no longer be center and the whole idea of using the center node as root will be nullified.

Overcoming the above drawback will involve using a dynamic technique that can keep track of the graph center. Keeping track of the graph center will be possible either with the knowledge of the entire network (which would be very easy if there is an external agent that can constantly keep track of the entire network topology and informs a node, multiple nodes or even designate a node/s as the center), or by gathering the

knowledge by communicating with all or some nodes in the network.

In light of the above, we decided to deduce a technique that neither requires an external agent nor adds complexity to Chen's technique. The authors of [53] presented a distributed algorithm to find the center of a tree graph. But the authors of [54] found a flaw in [53] and presented a decentralized version of it with a remedy. Our initial idea was to use [54] to find the root of the spanning tree generated by Chen's Package Routing Technique and use the same to keep track of the center as the spanning tree changes. This adds the overhead of finding the center each time the spanning tree changes. This solution is not desirable in situations as dynamic as a MANET, where the spanning tree topology can change quite rapidly. Therefore, we devised our technique EPRT in such a way that the center node is found, chosen as root and tracked with an overhead less than or equal to the overhead involved in Chen's technique.

Rest of this chapter is organized as data structures, message types and conventions used in EPRT in section 4.1, a detailed description of EPRT in section 4.2, Promiscuous mode operation of EPRT to bypass traffic through the root in section 4.3 and support for node mobility in section 4.4.

4.1 Data Structures, Message Types and Conventions Used in EPRT

In this section we present a detailed description of the EPRT technique. This technique is an enhanced version of Chen's technique which over comes all the issues we identified in all three protocols discussed in Chapter 3. For the purpose of clarity, we begin the description of EPRT with an introduction to the data structures, message types and any conventions used by EPRT.

In contrast to the three routing techniques discussed in Chapter 3, EPRT uses Routing Tables and Neighbor Tables instead of the generation tables. Figures 4.1 and 4.2

illustrate the routing table and neighbor table formats used by the nodes in EPRT.

DESTINATION	*HOP_COUNT*	*NEXT_HOP*	*EXPIRATION_TIME*	*FLAG*

Figure 4.1: Routing Table Format Used By EPRT.

NEIGHBOR	*EXPIRATION_TIME*	*FLAG*

Figure 4.2: Neighbor Table Format Used by EPRT.

Described below are few conventions used by EPRT:

(a) Each node does maintain in its routing table, an entry for itself as a destination (its own address)with HOP_COUNT = 0 and NEXT_HOP = 0. This will be the first entry in the routing table.

(b) NEXT_HOP = 0 and HOP_COUNT = 1 implies the next hop is a parent. Therefore, every time a node needs to mark an entry in the routing table, that is one hop away as a parent; changes the value of NEXT_HOP for this entry to 0. A node can have only one parent, therefore: every time a node marks any entry as parent, it makes sure to convert any other valid entry marked as a parent is now updated as a child.

(c) NEXT_HOP = DESTINATION implies the next hop is a child. Therefore,

every time a node needs to mark an entry in the routing table, that is one hop away as a child; changes the value of NEXT_HOP for this entry to the value in the DESTINATION field.

EPRT uses its own message scheme that is different from its predecessors. Listed below are the message types used by EPRT and the necessary fields in those messages. The fields specified in each message type are minimum required set.

-EXPLORE (<SOURCE> <NET_ID> <TIME_STAMP> <TTL>)

-PUP (<SOURCE> <DESTINATION> <TTL>)

-RETURN (<SOURCE> <DESTINATION> <FLAGS><DESTINATION, HOP_COUNT>)

-CENTER (<SOURCE> <DESTINATION> <FLAGS><DESTINATION, HOP_COUNT>)

-INVALID_DESTINATION (<DESTINATION> <INVALID-DESTINATION> <FLAGS>)

-JOIN (<SOURCE> <TTL>)

-JOIN_APPROVED (<SOURCE> <DESTINATION> <DEPTH> <TTL>)

-RELEASE (<SOURCE> <DESTINATION> <TTL> <FLAGS>)

-HELLO (<SOURCE> <DESTINATION> <TTL>)

-SUBTREE (<SOURCE> <DESTINATION> <CNTER> <SIZE> <FLAGS>)

4.2 Enhanced Package Routing Technique (EPRT)

The Enhanced Package Routing Algorithm discussed in this section consists of four major components:

1) Spanning Tree Generation, Center Finding & Center Maintenance

2) Neighbor Maintenance

3) Handling Link Failures

4) Routing Data

In this protocol we assume there is an external authority called STARTER_PICK_AUTHORITY that will pick the random STARTER_NODE. The first part is a modified version of [54]. It is responsible for generating a spanning tree and finding a center of that tree. It also helps the center node to determine if it is still a center following the updates it received. Second part handles the neighbor maintenance tasks. The Neighbor maintenance algorithm helps to detect link failures among the tree neighbors due to mobility or even system failure. Any node that detects a link failure uses the third item of EPRT to carry out appropriate actions. The final part helps the mobile stations to make packet forwarding decisions.

4.2.1 Spanning Tree Generation and Center Finding Algorithm (STGCFA)

The STGCFA is the core of the enhanced package routing technique presented in this chapter. The STGCFA algorithm is responsible for building the spanning tree, populating the routing tables and establishing the center of the spanning tree as the root. Unlike its predecessors [48], [49] & [51] which start with a root node and then build a spanning tree, STGCFA builds the spanning tree and then establishes the center of the spanning tree as the root.

Before the STGCFA starts all nodes remain in IDLE state. Each node has only one entry in its routing table, an entry for itself as a destination (its own address) with HOP_COUNT = 0 and NEXT_HOP = 0. Each node also maintains its own neighbor table. Now the STGCFA starts and proceeds as described below:

1) The STARTER_NODE changes to ACTIVE state and broadcasts an EXPLORE message with its address as the sender's address (SENDER field), TIME_STAMP with

the value of current time and a NET_ID. If the STARTER_NODE did not receive at least one PUP message within EXPLORE_TIMEOUT time units, it retries by sending a new EXPLORE message (with the same data and with new time stamp). After EXPLORE_MAX attempts without any response, the STARTER_NODE notifies the STARTER_PICK_AUTHORITY, up on which, the STARTER_PICK_AUTHORITY can make a decision of, weather it should continue with the same node as the STARTER_NODE or it should pick a new node as the STARTER_NODE.

2) If the STARTER_NODE received only one PUP message as a response to its EXPLORE message (which means the STARTER_NODE is a leaf node, i.e.), then the STARTER_NODE sends a RETURN message to its only neighbor the PUP message sender, changes its state to RETURNED and updates its routing table entry for the only neighbor with NEXT_HOP = 0 (implies it is a parent node).

The contents of the RETURN message would be the destinations in its routing table (minus recipient of the RETURN message) with their corresponding hop counts.

3) A node that is in IDLE state and received an EXPLORE message:

- Changes to ACTIVE state
- Updates its routing table by adding appropriate information in the EXPLORE message to the routing table. That is, a new entry is added to the routing table with the sender of the EXPLORE message as the DESTINATION, HOP_COUNT = 1 and the NEXT_HOP = DESTINATION
- Sends a PUP message to the EXPLORE message sender
- Broadcasts a new EXPLORE message with sender's address (SENDER field) as its address

4) A non STARTER_NODE node that is in ACTIVE state, and did not receive any PUP

messages in response to it's EXPLORE message, is a leaf node. Such a node sends a RETURN message to its only neighbor (its parent), changes its state to RETURNED and updates its routing table entry for the only neighbor with NEXT_HOP = 0.

5) A node that is in ACTIVE or RETURNED state if receives any EXPLORE message adds the EXPLORE message sender to its neighbor table.

6) A node that is in ACTIVE state and receives a RETURN message:

- Processes the RETURN message by adding all the DESTINATION items in the RETURN message to the routing table. HOP_COUNT for such items is increased by 1 and the NEXT_HOP for each such entry is the RETURN message sender (Source field in the RETURN message).
- If the RETURN messages have been received from all the neighbors in the routing table, then the node changes its state to TERMINAL state.
- If the RETURN messages have been received from all but one neighbor in the routing table, then the node sends a RETURN message to that neighbor node and changes its state to RETURNED. It also updates its routing table entry for that node by changing the NEXT_HOP field to 0 (implies that is a parent entry).

Two adjacent nodes that received RETURN messages from all of their corresponding neighbors but each other may end up transmitting a RETURN message simultaneously. Therefore, when a node sends a RETURN message to a neighbor and receives a RETURN message from the same neighbor, the node changes its state to TERMINAL if it's IP address is bigger than that of the neighbor, else the node remains in RETURNED state.

7) A node that is in TERMINAL state has the knowledge of a spanning tree of the network graph and therefore can check if it is the center or forward a CENTER message

in an appropriate direction:

- Find the two nodes say n1 and n2 in the routing table such that 'HOP_COUNT of n1' = h1 is the largest hop count and 'HOP_COUNT of n2' = h2 is the second largest hop count.

  (i) If h1 - h2 = 0 then the current node is the center and the node changes its state to CENTER.

  (ii) If h1 - h2 = 1 then the current node is the center and the node changes its state to CENTER.

  (iii) If h1 – h2 > 1, The node changes its state to RETURNED and sends the CENTER message to the node 'n1' and marks 'n1' as its parent.

The CENTER message consists of all the destinations in the routing table of the current node minus the destinations with the recipient of the center message as the NEXT_HOP (sub tree rooted at the recipient of the CENTER message).

8) A node in RETURNED state when receives a CENTER message from a parent node:

- Marks the CENTER message sender as a child.
- Adds all the contents of the CENTER message to its routing table, with the CENTER message sender as the NEXT_HOP and the HOP_COUNT equal to the HOP_COUNT in the CENTER message plus 1.
- Changes its state to TERMINAL State.

9) A node that is in CENTER state, upon every update to its routing table:

- Find the two nodes say n1 and n2 in the routing table such that HOP_COUNT of n1' = h1 is the largest hop count and 'HOP_COUNT of n2' = h2 is the second largest hop count.

  (i) If h1 - h2 = 0 then the current node is the center and the node changes

its state to CENTER.

(ii) If h1 - h2 = 1 then the current node is the center and the node changes its state to CENTER.

(iii) If h1 – h2 > 1, The node changes its state to RETURNED and sends the CENTER message to the node n1 and marks n1 as its parent.

Shown in the Figures 4.3 and 4.4 are spanning tree generated by STGCFA, Routing table and Neighbor table for a sample network.

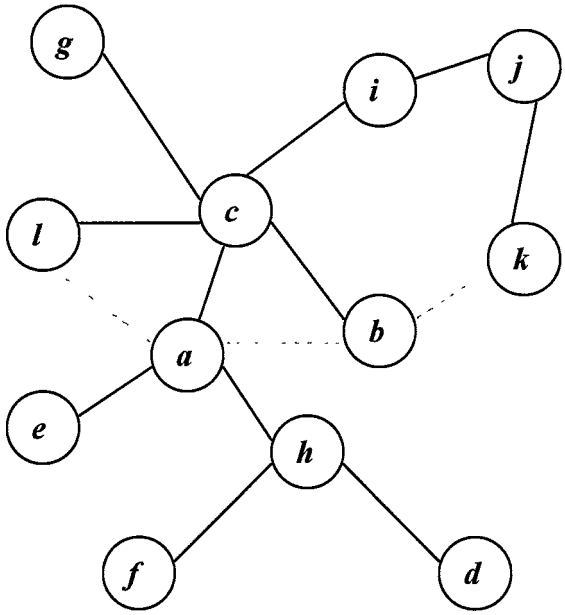


Figure 4.3: Spanning tree Generated by STGCFA.

4.2.2 Neighbor Maintenance

Neighbor Maintenance Algorithm (NMA) at a node is responsible for tracking any changes in its neighborhood that can affect the spanning tree structure built by the STGCFA. Each time a node receives a data or a control packet from a neighbor, it resets the EXPIRATION_TIME for that routing table or neighbor table entry to current time plus LINK_TIMEOUT. In the past LINK_TIMEOUT duration, if a node did not receive

DESTINATION	*HOP_COUNT*	NEXT_HOP	EXPIRATION_TIME	FLAGS		
				G		
a	*0*	*0*	*t*	0		
c	*1*	*0*	*t*	0		
e	*1*	*E*	*t*	0		
h	*1*	*H*	*t*	0		
f	*2*	*H*	*t*	0		
d	*2*	*H*	*t*	0		
k	*NULL*	*B*	t_b	1		

DESTINATION	EXPIRATION_TIME	FLAGS		
		G		
l	*t*	0		
b	t_b	1		

Figure 4.4: Routing table and Neighbor table of node a in Figure 4.3.

any packet from a neighbor in the routing table (very rarely for a neighbor in the neighbor table), then the node is not sure weather that particular neighbor is still within its transmission range. In essence, NMA will be triggered at any node due to two reasons.

(a) If the Routing table entry for a neighbor, has EXPIRATION_TIME equal to CURRENT_TIME. (This implies that the current node neither received a control packet nor a data packet from that neighbor in the past LINK_TIMEOUT duration.)

(b) A node might choose to maintain its connectivity with one or more of its non-

tree neighbors. A flag set for a neighbor table entry can indicate that. In such cases, the NMA is triggered when the EXPIRATION_TIME for a neighbor entry reaches CURRENT_TIME.

NMA at a node sends a small packet called HELLO_MESSAGE to particular neighbor and waits for HELLO_TIMEOUT time units for a HELLO_RESPONSE from that neighbor.

1) If the node does receive a HELLO_RESPONSE within HELLO_TIMEOUT time, then the node (resets the EXPIRATION_TIME for that routing table or neighbor table entry to CURRENT_TIME plus LINK_TIMEOUT) resets the EXPIATION_TIME for that link.

2) If the node did not receive a HELLO_RESPONSE within HELLO_TIMEOUT duration the link is considered to be broken.

(a) No RETURN message sent on the failed link yet

If the failed link is neither a parent nor a child, and that neighbor is present in the neighbor table; then that entry is removed from the neighbor table.

Consider two nodes A and B in the network, such that they are in each other's communication range. If A sent an EXPLORE message to B and B sent a PUP message to A; but neither one of them sent any RETURN messages yet (If any one of them had sent a RETURN message, the other should have received such a message and hence both become aware that a RETURN message was sent on that link). In such a scenario:

- The parent node (node A), will remove the child node (node B) from the routing table and proceeds as though that neighbor never existed.
- When the neighbor maintenance algorithm (described in section 3) at the child node (node B,) detects the loss of link with the parent node (node A); then the child node (node B), changes its state to RELEASED and sends a RELEASE

message to all its children.

Each node that receives a RELEASE message changes its state to RELEASED, sends a RELEASE message to its children and purges its routing table.

(b) A RETURN message was sent on the failed link

If A and B are two nodes in the network, such that node A is the parent of node B and earlier node B has already sent a RETURN message A, and the link between A and B failed. In such scenario:

- If the neighbor is a child node, then the node assumes it lost the connectivity with that neighbor and notifies the loss of sub tree rooted at that node, to the center. This is done by sending a RETURN message to its parent with the entries from the routing table for all destinations in the sub tree rooted at the child with whom the node lost its connectivity. i.e., all the entries from the routing table with the NEXT_HOP value equal to the node with which the current node has lost the connection. All such entries are marked as invalid or expired.
- If the neighbor is a parent node, then the node assumes it lost the connectivity with that neighbor and changes its state to RELEASED and sends a RELEASE message to all its children.

Each node that receives a RELEASE message changes its state to RELEASED, sends a RELEASE message to its children and purges its routing table.

4.2.3 Handling Link Failure

A Released node is a consequence of a node losing a link with its spanning tree parent or one of its earlier generation nodes losing its respective parent. This section presents a procedure followed by EPRT to rejoin the released nodes to the spanning tree.

(a) Released Nodes and JOIN Messages

A node in RELEASED state can join the spanning tree by pairing up with any node around it, by receiving an EXPLORE or by joining a node in RETURNED or CENTER state by sending JOIN messages. A RELEASED node can receive an EXPLORE message only if the RELEASED node lost its link with its parent before the spanning tree construction is complete.

(b) Nodes Released Before the Formation of the Spanning Tree

A node that is in RELEASED state if receives an EXPLORE message mimics the behavior of a node in IDEAL state. It will change its state to ACTIVE, proceeds pairing up with the EXPLORE message sender with a PUP message and then broadcasts an EXPLORE message with its address as the sender's address.

(c) Nodes Released After the Spanning Tree Formation

A node that is in RELEASED state is free to join any node that is in RETURNED or CENTER state. A RELEASED node broadcasts a JOIN message (With TTL = 1) and waits for a response for JOIN_EXPIRE units.

If a RELEASED node receives more than one JOIN_APPROVED message then:

- The node adds all the JOIN_APPROVED message senders to its neighbor table.
- The node picks the one message with the highest value of DEPTH and adds that node as a parent to its routing table.
- Sends a RETURN message to the parent node and changes its state to RETURNED.

Within JOIN_EXPIRE time units if the RELEASED node does not receive any JOIN_APPROVED message, then the node retries until it gets a response.

(d) Processing JOIN Messages

A node in RETURNED state when receives a JOIN message:

- Adds the JOIN message sender to its NEIGHBOR_TABLE
- Sends a JOIN_APPROVED message to the JOIN message sender. The JOIN_APPROVED message consists of the sender's address and the highest HOP_COUNT value from its routing table as DEPTH.

4.2.4 Updates to the Routing Table Due to Topological Changes

A node that is in RETURNED or CENTER state, when receives a RETURN message from a child node, resets the EXPIRATION_TIME for that child node's entry in the routing table, processes the content of the RETURN message, creates a new RETURN message with the updates to its routing table and forwards the new RETURN message to its parent (CENTER node does not have a parent, hence it doesn't have to further relay such messages). If no changes are made to the routing table after processing the RETURN message, no further generation or transmission of RETURN messages is necessary.

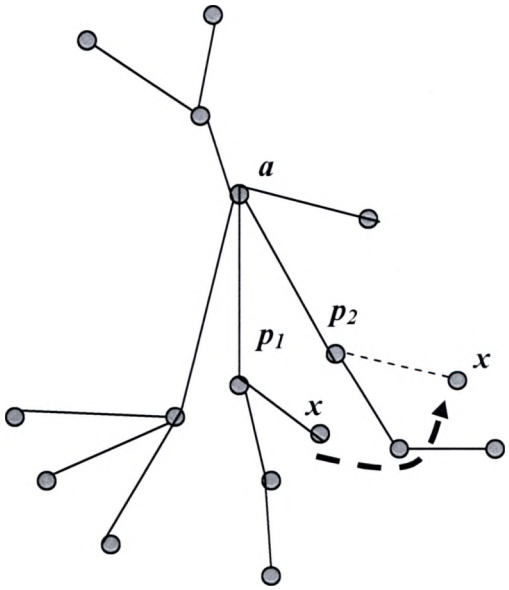
The following are the steps involved in processing RETURN Messages at nodes in either RETURNED or CENTER State:

Each RETURNED or CENTER node as it receives such RETURN messages

- Increments the HOP_COUNT field of all the entries in the RETURN message by one unit.
- Add a destination in the RETURN message to its routing table, if that particular destination is not already in the routing table. This entry should be included in the RETURN message that the current node sends to its parent.
- Updates the routing table entry for a particular destination, if that particular destination is already in the routing table, has a different HOP_COUNT (compared to the HOP_COUNT in the RETURN message) and the NEXT_HOP

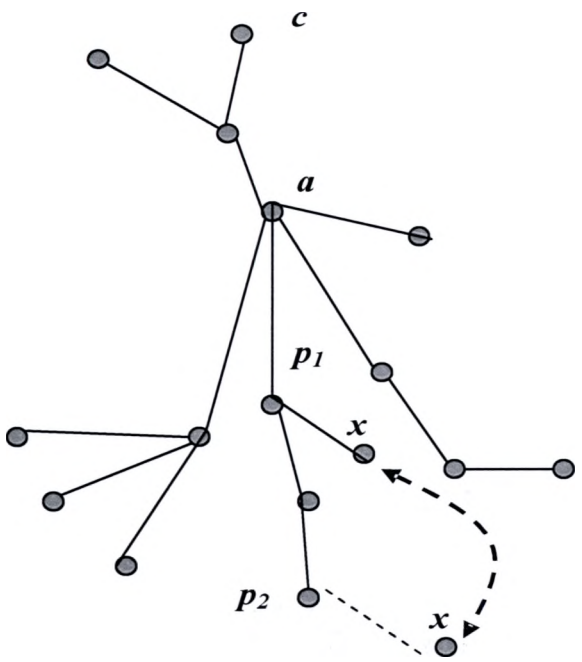
value in the routing table is the RETURN message sender. Updated field is HOP_COUNT for that particular entry. This entry should be included in the RETURN message that the current node will send to its parent, if this update changed the value of the current node's last known largest HOP_COUNT.

- Updates the routing table entry for a particular destination, if that particular destination is already in the routing table, has the same HOP_COUNT (compared to the HOP_COUNT in the RETURN message) and the NEXT_HOP value in the routing table is not the RETURN message sender. Updated field is NEXT_HOP for that particular entry. This entry should be included in the RETURN message that the current node will send to its parent, if this update changed the value of the current node's last known largest HOP_COUNT.



If node a received an update from p_2 before node a notified about loss of link failure with x. Updates are limited to a, p_1 and p_2.as the last known largest HOP_COUNT of node a did not change.

Figure 4.5: Updates to the routing table scenario I



In the adjacent sub tree, x leaving p_1 and joining p_2 or vice versa, can potentially displace the center. p_1 or p_2 should update node x's displacement to it's parent only if the last known largest HOP_COUNT in it's routing table is changed due to node x's movement.

Figure 4.6: Updates to the routing table scenario II

- Removes a destination in the RETURN message from its routing table, if that particular destination is already in the routing table, the HOP_COUNT value in the RETURN message is equal to the HOP_COUNT value in the routing table and NEXT_HOP value in the routing table is equal to the RETURN message sender. This entry should be included in the RETURN message the current node sends to its parent.

Figures 4.5 and 4.6 illustrate the updates to the routing table as explained above.

4.2.5 Routing Data Packets

In the following section, an entry is considered to be found in the routing or neighbor table if that entry's EXPIRATION_TIME is greater than the current time.

A node when needs to communicate with a remote host:

1) Look up the neighbor table for remote host.

- If the remote host is found in the neighbor table, forward the packet to the neighbor.
- Else, proceed to step 2.

2) Look up the routing table for an entry for that particular remote host.

- If the remote host is found in the routing table, and the HOP_COUNT = 1, then the remote host is either an immediate child of this node or a parent. In both cases weather the remote host is a child or a parent of the current node, the current node sends the data packets to that particular destination.
- If the remote host is found in the routing table, and the HOP_COUNT > 1, then the remote host is child of the current node (but not an immediate child). In this case the node forwards the packet to the NEXT_HOP node found for this entry in the routing table.

3) If the remote host is found neither in the neighbor table nor in the routing table.

- If the current node is not a CENTER node then, forward the packet to the parent node.
- If the current node is a CENTER node then, notify the sender of the packet with an INVALID_DESTINATION message.

4.3 EPRT Operation in Promiscuous Mode

In this section we present a technique that capitalizes on the inherent capabilities of wireless nodes, to bypass traffic from the root node. Most common feature among the modern LAN hardware for broadcast media such as wireless is the capability to operate the network interface in "promiscuous" receive mode. This feature allows the hardware to pass every packet that it receives to the network driver software without filtering the packets based on the data-link-layer destination address. Promiscuous mode operation

has two major side effects that could cause some adverse effect on a MANET node:

- Promiscuous mode operation increases the software overhead on the CPU.
- Promiscuous mode operation increases the power consumption of the network interface hardware.

However the limiting factors of the MANETs are network speed and most importantly the availability. We propose in the following section an optimization that can increase the speed and availability of the network. The proposed optimization takes advantage of the network interface's ability to operate in promiscuous mode. Promiscuous mode operation of the nodes is not necessary for the functioning of this protocol and it is completely voluntary. A node can switch back and forth between promiscuous and non-promiscuous mode based on its available battery life, network traffic or any other relevant factor.

4.3.1 Promiscuous Mode for Neighbor Maintenance

A node operating in promiscuous mode, if detects any messages transmitted by any of its neighbors resets the EXPIRATION_TIME value of that entry in the neighbor table or routing table (where ever the entry is available) to current time plus LINK_TIMEOUT. A promiscuously operating node when detects a message from a destination that is neither in the routing table nor in the neighbor table adds that destination to its neighbor table. A sample scenario is illustrated in Figure 4.7.

4.3.2 Promiscuous Mode for Shorter Routes

Apart from the neighbor maintenance, promiscuous mode helps the mobile stations to obtain shorter routes to destinations that are not their children. This also helps in reducing the traffic through the center node if there are any nodes that are closer to the originating node than the center and know of a route to the destination. A sample scenario

is illustrated in figure 4.8.

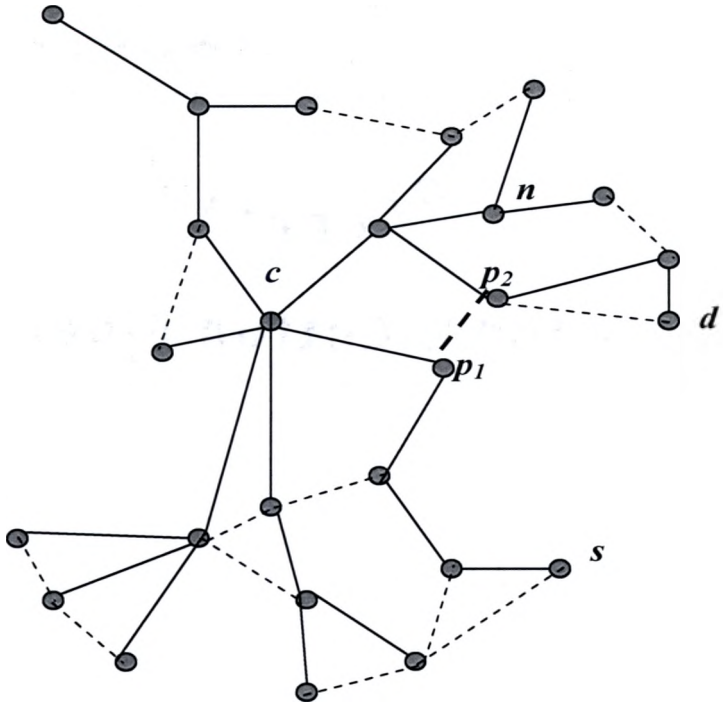


Figure 4.7: Promiscuous mode for Neighbor Maintenance.

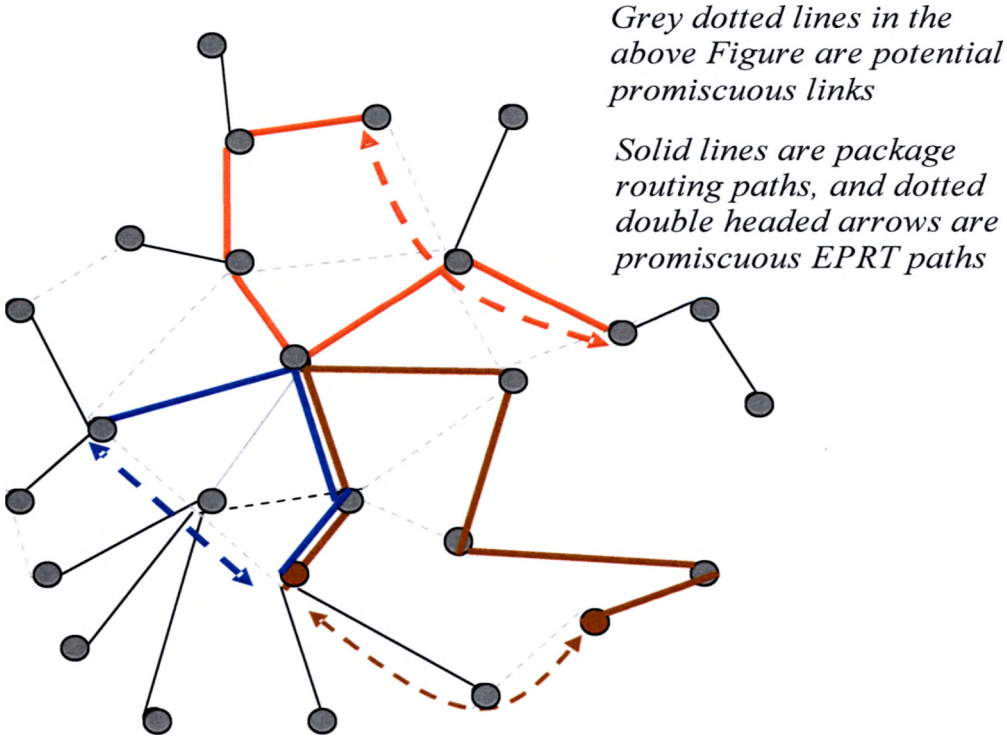


Figure 4.8: Promiscuous Mode for Shorter Routes.

(a) Operation of Gratuitous Return Message Sender

(i) Sending Gratuitous RETURN Messages

If a node operating in promiscuous mode detects packets en route to a destination that is either its child or a neighbor, the promiscuous node sends a RETURN message with a 'G' bit set ('G' for Gratuitous) to the sender of the message. Such RETURN message contains the DESTINATION and HOP_COUNT values from the routing table or neighbor table, depending upon weather the DESTINATION is present in the routing table or the neighbor table. The gratuitous RETURN message sender also designates such destination entries in the routing table or neighbor table as GRATUITOUS entries (Using the G flag in the routing table or neighbor table).

Figure 4.6 illustrates the gratuitous operation of node p_2. p_2 when detects that p_1 is trying to find a route to destinations ***d*** or ***n***, p_2 sends a gratuitous RETURN message to p_1. The same applies regardless of weather p_1 is routing the packets of a node say ***s*** or even if p_1 is trying to find the destination for itself.

(ii) Sending Loss of Gratuitous Link Notice

A node when detects an entry in the routing table marked as GRATUITOUS and if that entry were to be removed from the routing table due to an update (RETURN message from one of the current nodes children); then the node sends a RETURN message with an 'L' flag set ('L' for LOST), to all its neighbors. Any packets waiting in the queue for this lost destination are dropped (or, they could be forwarded to its parent and further to the root).

A node when detects an entry in the neighbor table marked as GRATUITOUS and if that entry were to be removed from the neighbor table due to loss of link with that neighbor; then the node sends a RETURN message with an 'L' flag set ('L' for LOST), to

all its neighbors. Any packets waiting in the queue for this lost destination are dropped (or, they could be forwarded to its parent).

(b) Operation of Gratuitous Return Message Receiver

(i) Processing Gratuitous RETURN Messages

The recipient of the gratuitous RETURN message (RETUN messages with 'G' flag set) stores the destination in the RETURN message in its routing table with the NEXT_HOP as the RETURN message sender, HOP_COUNT as NULL and the EXPIRATION_TIME as the expiration time of the RETURN message sender in the neighbor table or the routing table.

Any node operating in promiscuous mode when over hears a RETURN message with a 'G' bit set can add the destination in such RETURN message to its routing table as discussed above.

(ii) Processing Loss of Gratuitous Link Messages

When a node receives a RETURN message with an 'L' flag set, for each destination in the RETURN message, it removes any entries in the routing table with the Destination and NEXT_HOP equal to that of the RETURN message sender.

4.4 SUPPORT FOR NODE FAILURE

In this section we present the mechanism used by EPRT to overcome a node failure. Given the challenging conditions under which the MANETs operate, support for node failure is a crucial item of any MANET routing protocol. A key strength of EPRT is its ability to overcome node failures. A general node failure in EPRT is identified by the Neighbor maintenance algorithm and is handled as described in the above section.

In an instance of CENTER node failure, the proposed routing protocol leads to RELEASE of all the nodes in the network. In such a situation, the STARTER_NODE

will not know weather it was released due to CENTER failure or due to the failure of top most parent of the STARTER_NODE that is one hop closer to it than the CENTER node. This scenario is illustrated in Figure 4.9; STARTER_NODE up on its release can not distinguish between loss of link between ***(c, t)*** and loss of all dotted links due to center failure. Described below are two solutions that can effectively handle the failure of the center node.

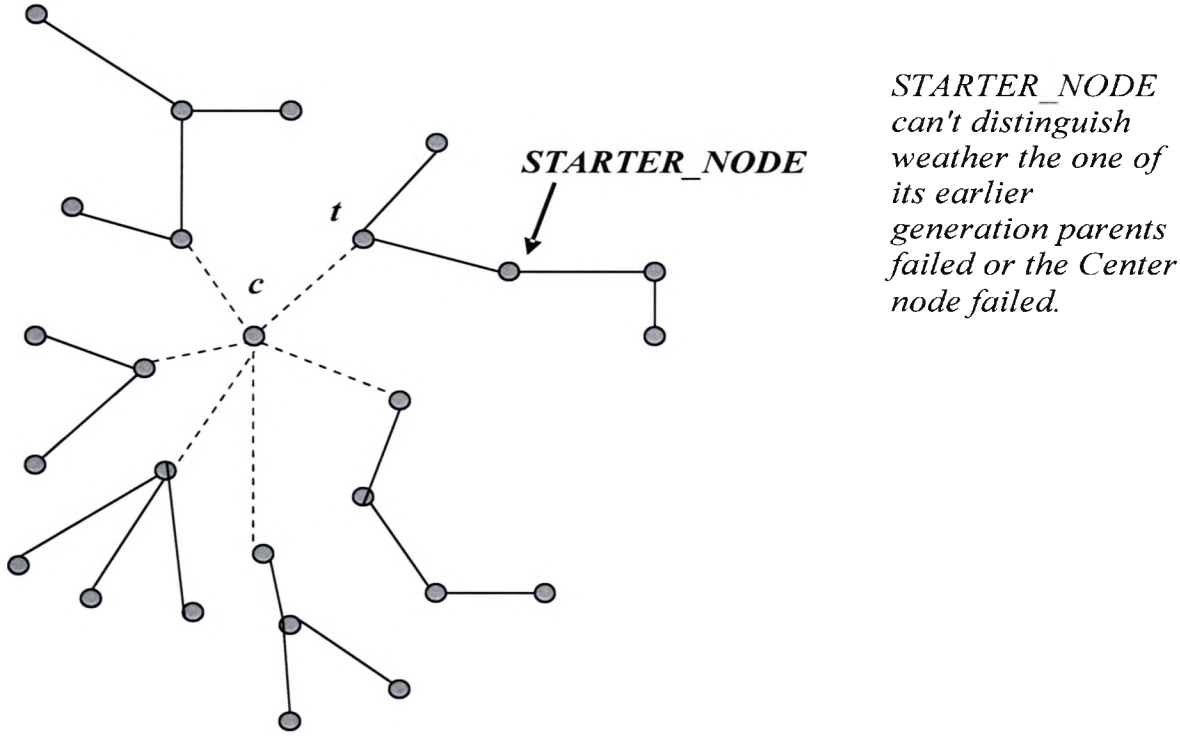


Figure 4.9: Support for node failure.

4.4.1 Release All Method

A simple solution for this problem can be the CENTER node can RELEASE all its neighbors if it detects the failure of its link with the root of the sub tree in which the STARTER_NODE is present. That is, say if the node ***c*** lost its connection with the node ***t*** then the node ***c*** will release all of its children. This will result in release of all the nodes in the network and a new spanning tree creation can be initiated by the STARTER_NODE.

Although this solution eliminates the confusion between loss of link between (*c*, *t)* and center node failure, it has a very high potential of dramatically increasing the control overhead of the network.

4.4.2 Greedy Tree Merger Method

An alternative to the above solution would be making all the nodes on the network aware of the sub tree break away and let the sub trees merge if there is no network partition. Presented below is the detailed operation of this technique.

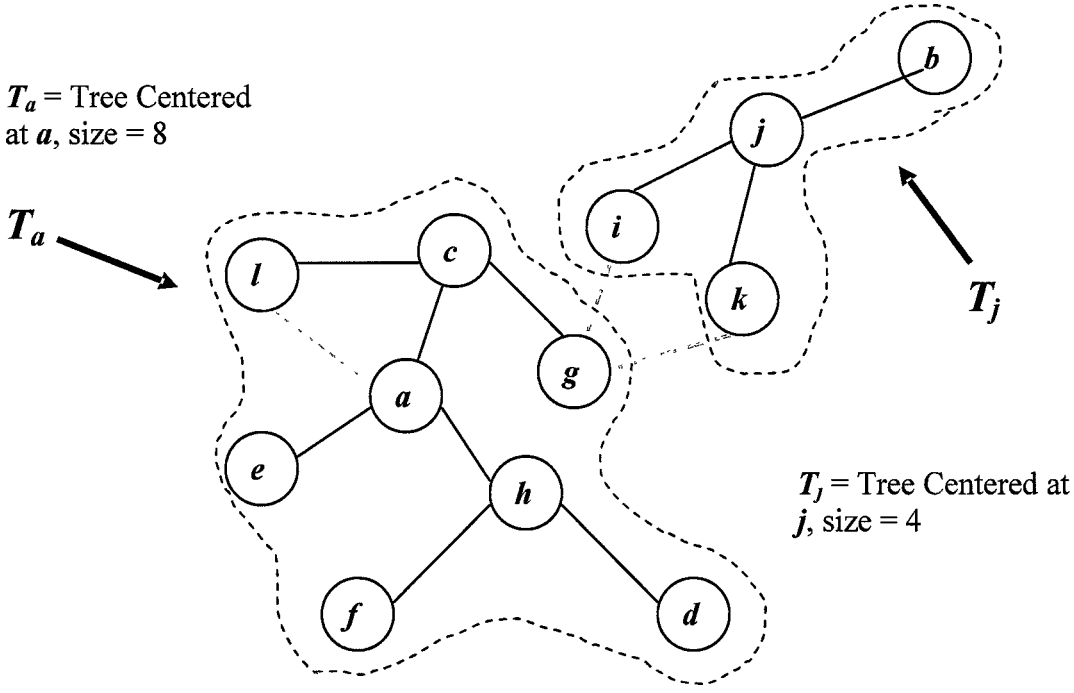


Figure 4.10: Greedy Sub Tree Merger.

An offspring of the CENTER node when loses a link with the CENTER node, changes its state to TERMINAL state and proceeds with finding the center of the sub tree it belongs to. To notify other nodes in the sub tree, it sends all its control messages with the 'S' flag set (S to indicate sub tree only). The new CENTER node within this sub tree now knows that it is a CENTER of a sub tree and hence broadcast's a control message called SUBTREE message consisting of the its identity (IP address) and size of the sub

tree (number of nodes in the sub tree) to all its children periodically.

Similarly the CENTER node in case of link failure with one of its children changes its state to TERMINAL state and proceeds with the task of finding the center of the sub tree it belongs to. However to notify other nodes in the sub tree, it sends all its control messages with the 'S' flag set (S to indicate sub tree only).

The new CENTER node within each sub tree now knows that it is a CENTER of a sub tree and hence broadcast's a SUBTREE message consisting of its identity (IP address) and size of the sub tree (number of nodes in the sub tree) to its children periodically.

The purpose of SUBTREE message is to make sure that each node knows how big a sub tree it belongs to and the identity of the CENTER. Starting at the CENTER, as this message flows down the tree, each node now knows that it is a member of a sub tree, rooted at the CENTER and how many other nodes are members of that sub tree.

Figure 4.10 illustrates the greedy sub tree merger technique. Nodes i and k belong to T_j, a tree of size = 4, and node g belongs to T_a, a tree of size 8. Nodes i and k encountered a bigger sub tree, via the node g. Both k and i will now leave T_j to join T_a via g. T_j slowly dissolves into T_a.

(a) Processing of SUBTREE Messages

SUBTREE Messages Received From The Parent Node:

- Update the CENTER node's address and size of the sub tree to the values received in the SUBTREE message.
- Create a new SUBTREE message with its knowledge of CENTER node's identity, sub tree size and broadcast it with TTL = 1.

SUBTREE Messages Received From A Neighbor That Is Neither A Parent Nor A Child:

If the CENTER in the SUBTREE message received from a neighbor is different from the CENTER in the most recent SUBTREE message from the parent, and the sub tree size of the neighbor is bigger than the size of the sub tree it belongs to, then:

- Forward the SUBTREE message received from the neighbor to the parent.
- Change its state to RELEASED, send RELEASE message down to its children.
- Send a RETURN message to the neighbor, which is a part of a bigger sub tree and add that neighbor to the routing table as the parent node.
- Change the CENTER identity and sub tree size values to that of the values it received in the SUBTREE message from the neighbor.

SUBTREE Messages Received From The Child Node:

When a node receives a SUBTREE message from a child node, it implies that the child node has encountered a bigger sub tree and therefore is no longer a child of the current node.

- Mark the child as a neighbor (Move the routing table entry for that child from the routing table into neighbor table).
- Proceed as if the SUBTREE message was received from a neighbor.

(b) Merger of Two Equal Size Sub trees

The technique presented above works by releasing the nodes in smaller sub trees so that they can join bigger sub trees. As a result smaller sub trees grow smaller and smaller until they are completely destroyed and the biggest sub tree grows bigger and bigger until all the nodes in the network are part of the tree (as long as there are no partitions in the network). However, the above section does not address the scenario where the network has two sub trees of equal size. Either one of them can grow by

merging and dissolving the other sub tree into it. In such a situation we recommend that the sub tree whose CENTER node's IP address is smaller (or bigger) of the both will dissolve into the other sub tree.

CHAPTER 5

ANALYSIS AND FEASIBILITY STUDY OF EPRT

In this chapter we present a complete analysis of the EPRT technique discussed in Chapter 4. We start of by proving some important theorems in section 5.1. These theorems are necessary to establish a proof of correctness of the STGCFA. In sections 5.2 and 5.3 we establish an upper limit on the time and message complexity of EPRT. In section 5.4 we take a slightly different approach and establish an upper limit on the time required by EPRT based on the simulation area.

In light of the pitfalls in the simulation study of LCMRMG and LCMRMGCS (discussed in chapter 3), we decided to explore the impact of mobility on the MANETs rather than the MANET routing protocols. Therefore, in section 5.5 we presented a brief study of various realistic mobility models and their impact on the MANETs under different conditions. We have used some simulations to justify the conclusions we made in this section. Section 5.6 presents a feasibility study of EPRT in which we identify the MANET scenarios suitable and not suitable for EPRT.

5.1 PROPERTIES OF EPRT

The STGCFA is the core of the enhanced package routing technique presented in chapter 4. The STGCFA algorithm is responsible for building the spanning tree, populating the routing tables and establishing the center of the spanning tree as the root. Unlike its predecessors [48], [49] & [51] which start with a root node and then build a spanning tree, STGCFA builds the spanning tree and then establishes the center of the

spanning tree as the root. Following any topological changes in the network, STGCFA is also used in finding the new center of the spanning tree which then becomes the new root.

To establish a proof of correctness for EPRT we start of by proving that the STGCFA of EPRT yields a spanning tree and does so in finite time for finite trees. We use theorems 2 and 3 to prove that the STGCFA finds the center of the spanning tree it built. The final theorem in this section, theorem 4 will be used to aid our arguments in sections 5.2 and 5.3.

Theorem 1

The STGCFA algorithm of EPRT forms a spanning tree of the network graph and does so in finite time for finite network graphs

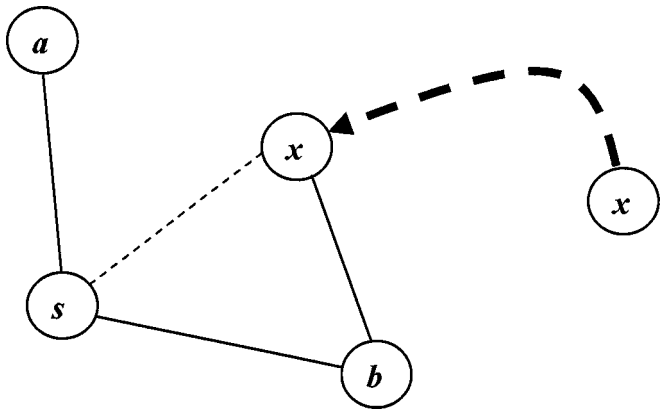


Figure 5.1: No cycles are formed in STGCFA.

Proof:

As illustrated in the Figure 5.1 let *s* be a starter node, *a* and *b* be the nodes in the transmission range of *s* and *x* be a node in the transmission range of *b*.

According to STGCFA, all nodes are in IDLE state before it starts. The starter node then changes its state to ACTIVE and broadcasts an EXPLORE message. The nodes *a* and *b*, will pair-up with the node *s* by sending a PUP message and change their state to

ACTIVE.

Each node after changing to ACTIVE state now sends an EXPLORE message with its address as the sender's address. Therefore, the EXPLORE message sent by *b* will be received by the node *x.* Let us assume that the node *x* after pairing up with node *b* has wandered into the transmission range of *s*. If the node x, now in ACTIVE state sent an EXPLORE message. According to the STGCFA, only nodes that respond to an EXPLORE message are the nodes in IDEAL state. Therefore, even if the node *s* received an EPLORE message from the node *x,* it will simply drop the message.

Similarly, every node in STGCFA once finishes exploring its neighbors, it will not participate in any more pairing up activity. Therefore, the links added by the STGCFA algorithm will not form a cycle. -------> (1)

Each node in STGCFA sends or receives a RETURN or a CENTER messages only to the nodes it had paired-up with. ----------> (2)

In STGCFA the only links added to the spanning tree are the ones on which either a PUP message is received from a non leaf node or a RETURN messages is received from a leaf node. ---------> (3)

Combining this with (1), (2) & (3), we can conclude that the graph formed STGCFA is a tree. This is a tree derived from the actual network graph. *Therefore, the graph yielded by STGCFA is a spanning tree of the network graph.*

In a finite graph the distance between the STARTER_NODE and any leaf node is finite, STGCFA is loop free, first node to reach TERMINAL state does so in finite time and the time required to find the center node after the first node reaches TERMINAL state is finite. Therefore, STGCFA will build the spanning tree and find the center of the spanning tree in finite time.

Theorem 2

In STGCFA exactly one node in the topology will reach the TERMINAL state at a given time.

Proof:

We establish a proof for this theorem by using a proof by contradiction method. First we prove that three or more nodes can not reach TERMINAL state simultaneously and then we prove neither two adjacent nodes nor two non adjacent nodes can reach TERMINAL state simultaneously.

(a) Three or more nodes can not reach TERMINAL state simultaneously in STGCFA

Here we use a proof by contradiction method. As illustrated in Figure 5.2, imagine three nodes *a*, *b* and *c* that are not necessarily neighbors reached TERMINAL state simultaneously.

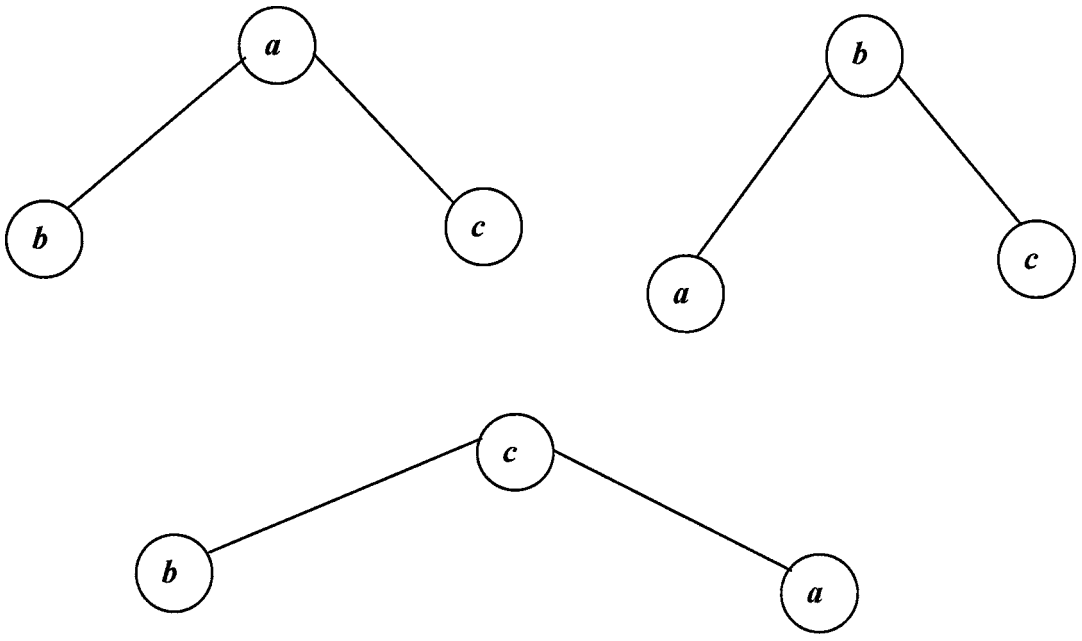


Figure 5.2: Three or more nodes can not reach TERMINAL state simultaneously.

According to the STGCFA, a node reaches TERMINAL state if an only if it received a

RETURN message from all of its neighbors. ----------(1)

Therefore, node *a* received RETURN message from nodes *b* and *c* (or from the nodes that are in the neighborhood of *a* and parents of *b* and *c*). Similarly node *b* should have received RETURN messages from the nodes *a* and *c*, and node *c* should have received RETURN messages from nodes *a* and *b* ----------(2)

According to the STGCFA:

A node can send a RETURN message only after it has received a RETURN message from all but one of its neighbors. ----------(3)

We can clearly see that (2) is impossible under the conditions (1) and (3). *Hence 3 nodes can not simultaneously reach TERMINAL state in the STGCFA.*

Applying the above logic we can prove that four or more nodes can not simultaneously reach the TERMINAL state in STGCFA. In other words "*The number of nodes that can reach TERMINAL state is less than 3.*"

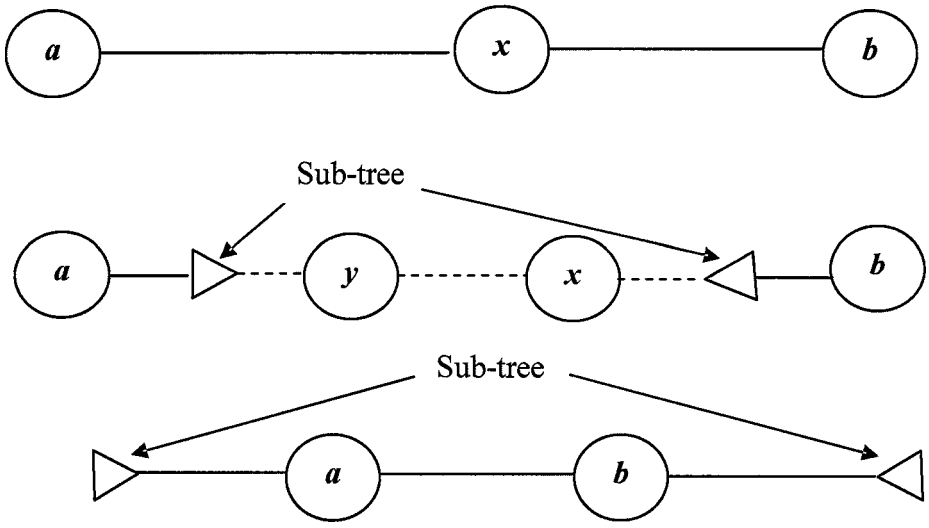


Figure 5.3: Two non adjacent nodes can not reach TERMINAL state in STGCFA.

(b) Two non adjacent nodes can not reach TERMINAL state simultaneously in STGCFA.

Consider three nodes *a, x* and *b.* Let *x* be a node in between *a* and *b,* as shown in the above Figure 5.3. Let us assume both nodes *a* and *b* have reached TERMINAL state simultaneously. Which is only possible if node *x* has sent a RETURN message to both *a* and *b.*

According to STGCFA, before the center is found each node sends only one RETURN message. Also, if the node x sent a RETURN message to both *a* and *b*, it implies that both nodes *a* as well as *b* are the parent nodes of *x,* which is impossible in STGCFA.

Similarly, even when there are more nodes between *a* and *b,* the above argument applies and two non adjacent nodes can not reach TERMINAL state simultaneously.

(c) Two adjacent nodes can not reach TERMINAL state simultaneously in STGCFA

Two nodes In STGCFA can reach TERMINAL state if and only if two neighboring nodes sent each other a RETURN message simultaneously. In STGCFA, if two nodes exchange a RETURN message simultaneously, then both nodes become aware of this. In such situations the node with smaller IP address among the two will reach TERMINAL state. (As each node knows that, it and its neighbor have received RETURN messages from all of their neighbors, the node that is closer to the center can reach the TERMINAL state and the other node can decide to remain in RETURNED state).

Combining (a), (b), and (C) we can conclude that *at a given time only one node can be in TERMINAL state in STGCFA.*

Theorem 3

STGCFA of EPRT will find the center of the spanning tree it constructed.

Proof:

We use the Figure in 5.4 to illustrate this proof. We know that the STGCFA forms

a tree and at any given time there is only one node in TERMINAL state. In this algorithm a node that received a RETURN message from all of its neighbors (In other words, a node which reached TERMINAL state), is a center if $h_1 = h_2$ or $(h_1 - 1) = h_2$.

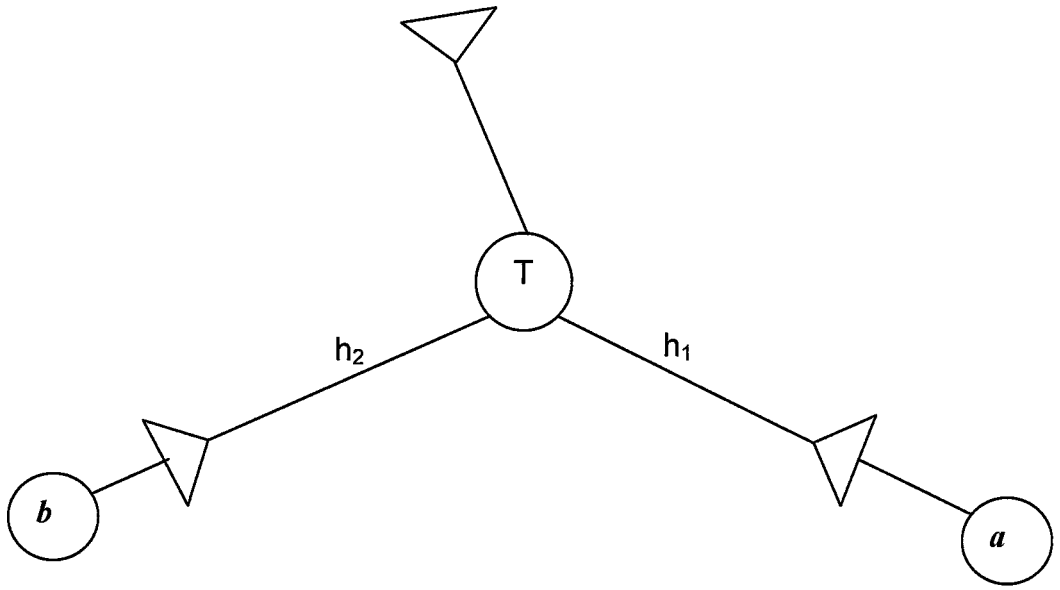


Figure 5.4: STGCFA finds center correctly.

Assume a non-center node *t* reached TERMINAL state. Let *a* and *b* be two nodes in the routing table of the node *t* such that their corresponding HOP_COUNT values are h_1 and h_2 (where h_1 is the largest in the routing table and h_2 is the 2^{nd} largest). Let *c* be the center node. If d(i, j) is the distance in hops between two nodes i and j Then we have:

$h_1 = d(a, t)$ and $h_2 = d(t, b)$

Node *t* is not a center node, $h_1 - h_2 > 1$ and $h_1 > h_2$, therefore, node *t* sends changes its state to RETURNED and sends a CENTER message to its neighbor, in the direction of the node *a* (in other words node *t* sends the CENTER message to the next hop node on the routing table entry for node *a*). Hence node *t* sends the CENTER message towards the actual center.

The above process continues until the CENTER message reaches the node *c,* which is the real center of the tree. Once the center node changes its state to TERMINAL, it will find itself as the center (because, either $h_1 = h_2$ or $h_1 - h_2 = 1$). This process does stop in finite time in a finite tree. *Hence the STGCFA finds one of the center nodes, of the spanning tree it generated*

Theorem 4

The farthest node from any node in the tree formed by the STGCFA is an extreme of a diameter path

Proof:

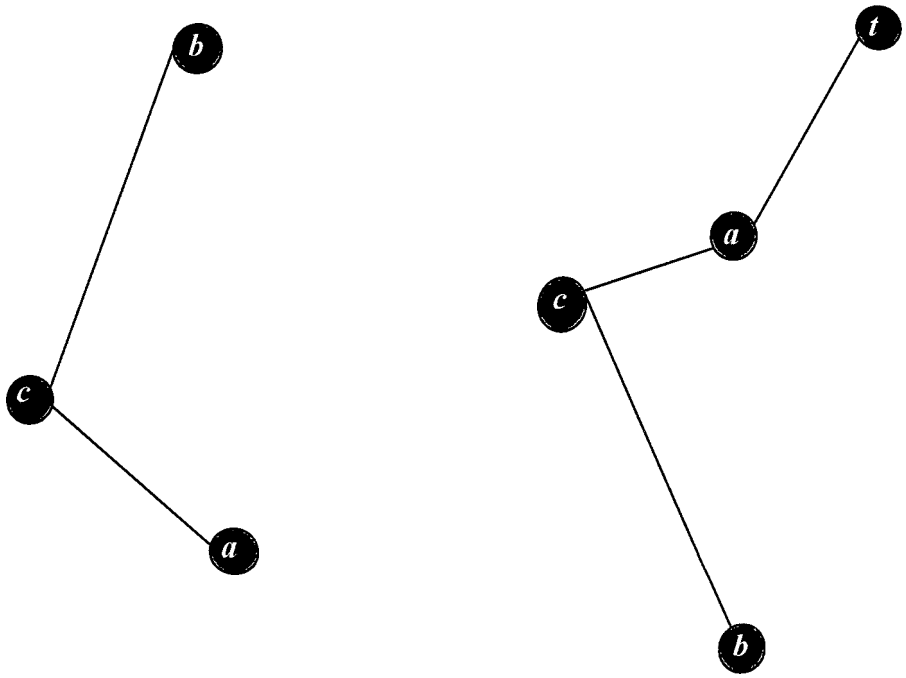


Figure 5.5: Farthest node from a destination is an extreme of a diameter path.

Let *a, b* and *c* be three nodes in the tree formed by STGCFA such that node *c* is the center and *b* is an extreme of a diameter path as shown in Figure 5.5. Any node *a,* that is farthest from *b* would be the other end of the diameter path.

That is, d(*a, b*) = D(T), where D(T) is the diameter of the tree formed by the

STGCFA. We also know that for any node '*x*' that is farther than D(T)/2 (radius of the tree) from a given node *y*, path from *x* to *y* represented by <*x, y*>:

$$<x, y> = <x, c> + <c, y>$$

Similarly: d(a, b) = D(T)

= D(T)/2 + D(T)/2

= d(a, c) + d(c, b) since a and b are the two extremes of a diameter

= d(a, c) + D(T)/2

Any node farthest from the center *c*, is at the most D(T)/2 hops away.

Therefore, as node *a* in consideration moves from the leaf node on one extreme of a diameter path to the center node, the farthest node from *a* will be *b,* where *b* is the other extreme of a diameter path.

5.2 Maximum Time Required by STGCFA

In this section we establish an upper bound on the maximum time required by STGCFA to build the spanning tree, find the center and be ready for routing. We assume that the processing time at each node is negligible and all messages are delivered in unit time.

Theorem 5

For a network graph G(V, E), STGCFA forms the spanning tree and finds the center of the spanning tree at most in 2 D(G) time units. (In other words the Enhanced Package Routing Technique presented in this paper is ready for routing in 2 D(G) time units). Where D(G) is the diameter of the network graph G

Proof:

As illustrated in Figure 5.6, let T(V, E') be the spanning tree created by STGCFA, *s* be the STARTER_NODE, node *f* be the farthest node from *s.* The tree T(V, E')

generated by STGCFA is similar to the tree generated by a flooding technique, therefore the mobile stations from *s* through *f* will change their state from IDLE to ACTIVE in the order of hop count from *s*

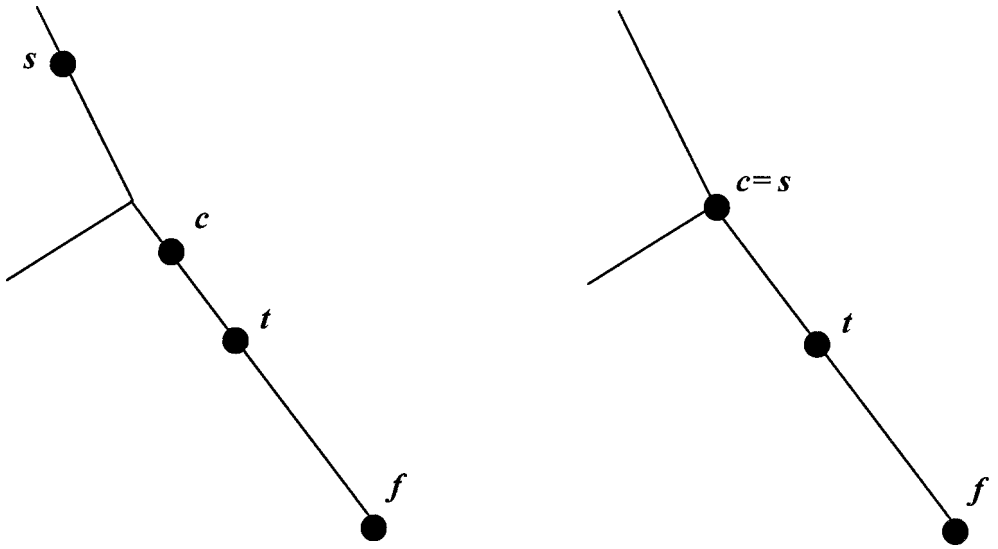


Figure 5.6: Maximum time required by STGCFA.

The center node *c* lies on the path between *s* and *f* (unless there are two nodes that satisfy the conditions of *f*, in which case the node *c* is same as the node *s*). This follows from the above property 4 and the properties of a center node.

Let *t* be a node that first reached the TERMINAL state. The RETURN messages flow from *f* to *t* in a bottom up manner. Therefore the most time it takes for the first node to reach TERMINAL state is: d(*s*, *f*) + d(*f*, *t*). It takes another d(*t*, *c*) time units until the center node reaches TERMINAL state and changes its state to CENTER.

Therefore the total time t = d(*s*, *f*) + d(*f*, *t*) + d(*t*, *c*)

= d(*s*, *f*) + d(*f*, *c*)

=d(*s*, *f*) + D(T)/2

The tree T(V, E') is generated from the graph G(V, E). Therefore, the farthest node from node *s* will be at the most D(G) hops from it, in which case *s* and *f* are the two

extremes of a diameter path. Therefore:

$$d(s, f) + D(T)/2 = D(G) + D(T)/2 \text{ ----------->(i)}$$

In [53] the authors have shown that the radius (denoted by R(T)) of an arbitrary spanning tree T of a graph G is at most as big as the diameter of the graph.

$$\text{i.e., } R(T) = D(G)$$

$$\text{Therefore, } D(T) = 2R(T) = 2D(G) \text{ ----------->(ii)}$$

Combining (i) & (ii), we have:

$$d(s, f) + D(T)/2 = D(G) + 2D(G)/2 = 2\,D(G)$$

Hence the maximum time required for the STGCFA to form the spanning tree, find the center and be ready for routing is at most 2 D(G).

Also when two nodes that satisfy the conditions for *f* exist, the maximum time taken for the first node to reach TERMINAL state = $d(s, f) + d(f, t)$. By the nature of STGCFA, if *s* is the center, s will be the first node to reach TERMUNAL state. Therefore, the time taken by STGCFA = $d(s, f) + d(f, t) = D(T)/2 + D(T)/2$

$$= D(T) = 2D(G)$$

Hence, the time required by the STGCFA to build the spanning tree, find the center and be ready for routing is at the most twice the diameter of the network graph.

5.3 Maximum Number of Messages Used by STGCFA

In this section we establish an upper bound on the maximum number of messages required by STGCFA. Since we are more interested in finding the total number of messages, regardless of the message type, we assume each message to be unit size.

Theorem 6

Given a network graph G(V, E), STGCFA forms the tree and finds the center of the tree using no more than 3V(G) + D(G) - 2 messages.

Proof:

Various messages used by the STGCFA before the center is found are EXPLORE, PUP, RETURN and CENTER.

(i) Total number of EXPLORE messages

Only nodes that are in ACTIVE state send EXPLORE messages and they do it only once. Therefore, the total number of EXPLORE messages used is equal to number of vertices in the graph, denoted by V(G).

(ii) Total number of PUP messages

In the STGCFA, PUP messages are used to link each node with its tree neighbors (parent and child nodes). A node once changes its state to ACTIVE state sends only one PUP message to the node which activated it, and does not respond to any further EXPLORE messages. Therefore, the total number of PUP messages is equal to the number of edges in the tree formed by the STGCFA (denoted by E(T), where T is the tree formed by the STGCFA).

Total number of PUP messages = E(T)

But, we know that the number of edges in a spanning tree of a graph is equal to the number of vertices in the tree minus one. Therefore,

$E(T) = V(T) - 1$

Also, the number of vertices in any spanning tree generated from a connected graph is equal to the number of vertices in the graph itself.

i.e, $V(T) = V(G)$

Therefore, total number of PUP messages = E(T)

$= V(T) - 1$

$= V(G) - 1$

(iii) Total number of RETURN messages used

In the STGCFA, each node sends only one RETURN message and it is sent to only one of the neighbors from which a RETURN message was not received. Therefore, the total number of RETURN messages is equal to the number of edges in the tree formed by the STGCFA (denoted by E(T), where T is the tree formed by the STGCFA). Therefore, total number of RETURN messages = E(T) = V(G) – 1

(iv) Total number of CENTER messages

In the STGCFA, the first node to reach the TERMINAL state is at most radius of the tree (Denoted by R(T)) units away from the center node. It is only on the path between the first node to saturate and the actual center any CENTER messages are sent. This path can be at most R(T) hops long.

Therefore, total number of CENTER messages = R(T)

$$= D(T)/2$$

From (ii) in the above theorem we have D(T) is at most 2 D(G)

$$R(T) = 2\,(D(G)/2)$$

$$= D(G)$$

Adding total number of messages used in each of the above sections (i), (ii), (iii) and (iv) we get: V(G) + V(G) – 1 + V(G) – 1 + D(G) = 3 V(G) + D(G) – 2.

Therefore, STGCFA forms the tree and finds the center of the tree, using at the most 3 V(G) + D(G) - 2 messages.

5.4 Maximum Time Required by STGCFA Based on Simulation Area

In this section we present a brief study of the properties of STGCFA in a simulation scenario. Let us consider the scenario where the network graph is a long chain with *n* number of nodes along the chain. This scenario is illustrated in Figure 5.7

In this particular scenario, the graph G is itself a tree. Therefore the values of D(G) and D(T) are equal.

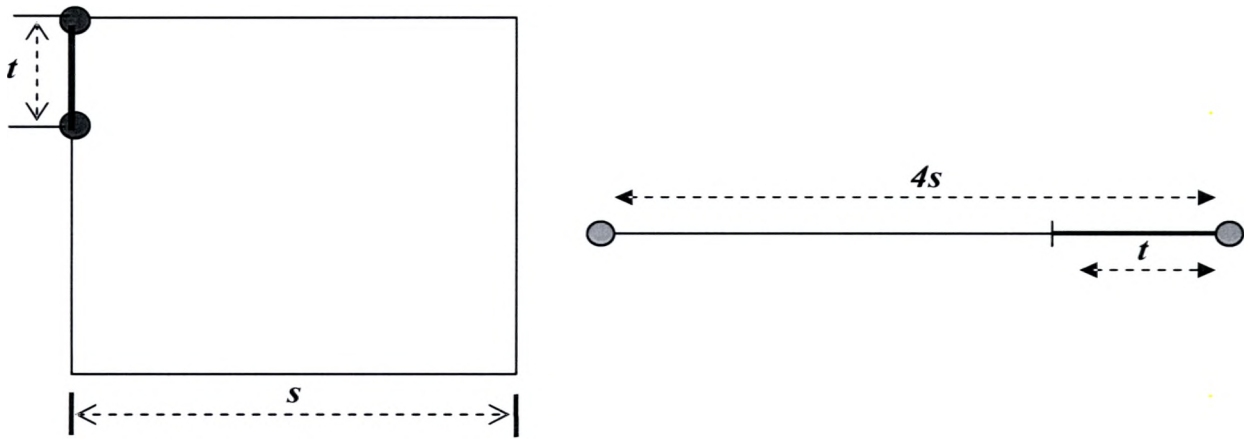


Figure 5.7: Maximum Time Required by STGCFA Based on Simulation Area.

$$\text{i.e., } D(G) = D(T) = \frac{(4s - t)}{t}$$

Also, the minimum number of nodes required to form such a network is equal to D(G). We also know that the time required by the STGCFA is equal to D(G) + D(T)/2. Therefore, the time required by STGCFA = D(G) + D(G)/2

$$= 3\ (D(G)/2)$$

$$= 3\frac{(4s - t)}{(2t)} \text{ ----------} \rightarrow \text{(i)}$$

Let us now consider the scenario where the R(T) = D(G). A good example of such a topology would be a grid of nodes, where each node is $(t - \Delta)$ far from it's neighbors such that Δ is very small or negligible. Shown in the Figure 5.8 is the grid graph, the bold lines show a possible spanning tree where R(T) = D(G).

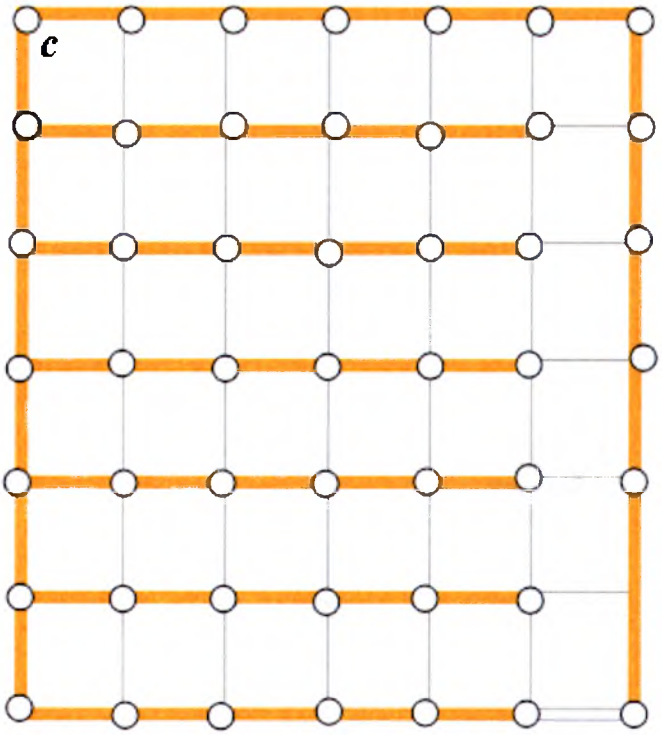
$$D(G) = 2\ \frac{s}{t}$$

R(T) = D(G) implies D(T) = 2 (D(G))

Most amount of time required by STGCFA = D(G) + D(T)/2

= 2 (D(G))

$= 4 \frac{s}{t}$ ----------→(ii)



Node *c* is a center of the tree *T* of the *n* x *n* square grid graph *G*. The radius *r*(*T*) = 2(*n*-1), and *r*(*G*) = *n*-1

Figure 5.8: Orange lines indicate a spanning tree of a n x n Grid network.

The minimum number of nodes required to form such a grid is $\frac{s}{t} * \frac{s}{t}$

In any realistic simulation the value of *s* is significantly larger than *t,* for instance *s = 1000* and *t = 50. C*omparing the expressions (i) and (ii), we can clearly observe that (i) yields grater value than (ii). Example: For *s = 1000* and *t = 50,* (i) yields 118.5 and (ii) yields 80 which is almost two thirds of 118.5. *This indicates that the most amount of time required by the STGCFA is better represented as D(G) + D(T)/2 rather than 2D(G).*

5.5 Impact Of Mobility on MANETs

In this section we present a brief study of the impact of mobility on MANETs rather than any individual MANET routing protocol. We believe this study is important due to the fact that, it is the MANET that is affected by the mobility, not the MANET routing protocol. For a given MANET scenario, we can establish weather a routing protocol is appropriate or not by analyzing the characteristics of that particular scenario. In section 5.6, we use the study presented in this section to establish few general MANET scenarios that are appropriate and inappropriate for our EPRT technique.

In the recent past there seems to be a heavily grown interest in MANET research. Several protocols were proposed and several outrageous claims have been made about the performance of such protocols. Many authors have tried to justify their claims based on some simulation results. Some authors have paid least interest to the mobility aspect of the simulations. For example [51] and [52] seem to have used very unrealistic radio transmission range and there seems to be no mobility model or defined mobility characteristics. Many more authors have used only a single mobility model through out their simulation study while most of the interest was paid towards measuring the performance. As long as desired simulation results are achieved, no attention was paid to the parameters that had impact on the performance.

We believe a better approach is to first study the influence of parameters such as mobility, transmission range, node speed etc, on the MANET and then analyze how individual protocols perform under a varying influence of these control parameters. In the rest of this section we will introduce four popular mobility models and study the impact of these mobility models on MANETs, under various control parameters.

5.5.1 Mobility Models

The authors of [54] presented a survey of various mobility models that can be used for MANET research. Although none of the mobility models can accurately represent a real world scenario, these models induce some basic real world challenges into the simulation environment. Four most popular mobility models are:

- Random Waypoint:

  In Random Waypoint model, at every instance a node randomly chooses a destination and moves towards it with a velocity chosen uniformly and randomly between an upper limit and a lower limit.

- RPGM (Reference Point Group Mobility model) with four groups:

  In this model each group has a group leader that decides the motion of the group. Initially each member of the group is uniformly distributed in the neighborhood of the group leader. At every instance each node in the network has a speed and direction that is derived by randomly deviating from that of the group leader.

- RPGM (Reference Point Group Mobility model) with one group:

  This model is same as the RPGM with four groups except the total number of groups in this model is equal to one. Hence only one leader and the entire network is a group.

- Manhattan Grid model:

  Manhattan Mobility Model emulates an urban area where some roads/paths run vertically and some run horizontally essentially forming a grid. Mobile stations traveling in this mobility model can take turn in any of the possible 4 directions (left, right, front and back) at any intersection.

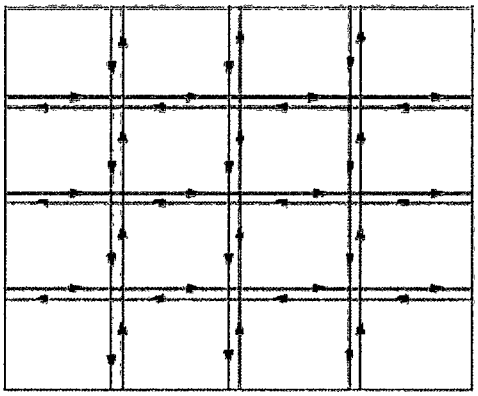


Figure 5.9: Manhattan Grid.

This model can be used to study the behavior of the MANET in which the mobile nodes are in motion similar to people or traffic in motion in a typical urban downtown. Figure 5.9 illustrates a Manhattan grid.

5.5.2 Simulation Platform

After investigating two popular MANET simulations platforms ns2 [58] and Glomosim [59], we realized that both of them focus on the study of the impact of various control parameters on the MANET routing protocols rather than the network graph. Therefore, we decided to use neither of them. A further pursuit of a suitable candidate for our study pointed us towards BonnMotion.

BonnMotion is Java software which creates and analyses mobility scenarios. It is developed within the Communication Systems group at the Institute of Computer Science IV of the University of Bonn, Germany, where it serves as a tool for the investigation of mobile ad hoc network characteristics.

5.5.3 Simulations and Analysis

Using BonnMotion we simulated all four popular mobility models we discussed above and analyzed their impact, under the influence of various control parameters, on the MANETs.

Eight scenarios which differ in speed as: (1, 5, 10, 20, 30, 40, 50, 60) m/s, per mobility model were generated with two different transmission ranges (50m and 100m). While the speed changes, other parameters like pause time and transmission range are kept constant. The simulations were carried out with 100 nodes in a 1000 x 1000 simulation area.

To better understand the impact of these simulation parameters on the MANET, we measured two network graph characteristics, the average link duration and the link break percentage. Plots of our simulation results are presented in Figures 5.10, 5.11, 5.12 & 5.13 and an analysis is presented below.

(a) Node Speed vs. Average Link Duration

Shown in Figures 5.10 and 5.11 are two plots with the average link duration against node speed with a transmission range of 50m and 100m respectively. Here we can notice for both transmission ranges of 50m as well as 100m, the average link duration is in the order Manhattan model < Random Waypoint < RPGM 4 groups < RPGM 1 Group. While the nodes in Random waypoint model choose a destination and move towards it with a velocity chosen uniformly randomly between an upper limit and a lower limit, the nodes in group mobility models derive their speed and direction of motion from that of the group leader. Therefore, the relative velocity of nodes is higher in Random waypoint than in the group mobility models. Because of the turns at every intersection and mobility in opposite directions, the nodes in Manhattan model experience highest relative velocity. From the plot in Figure 5.10 we can clearly see that the average link duration of these mobility models for a given speed is in the order of their relative velocities. Therefore, the relative velocity of the mobile nodes is directly proportional to the average link duration of the network.

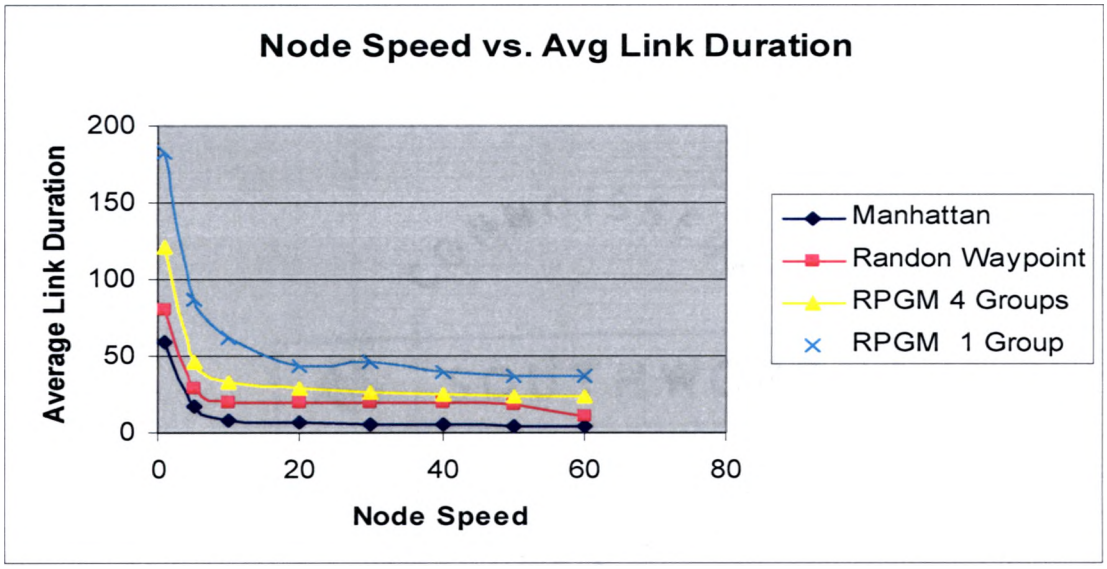


Figure 5.10: Average Link Duration vs. Node Speed, while transmission range = 50m.

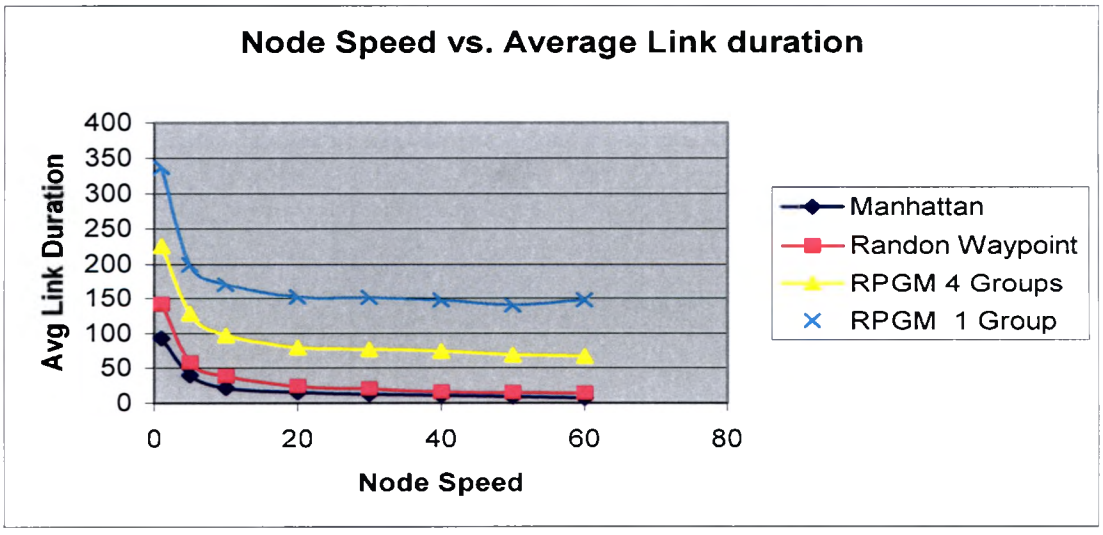


Figure 5.11: Average Link Duration vs. Node Speed, while transmission range = 100m.

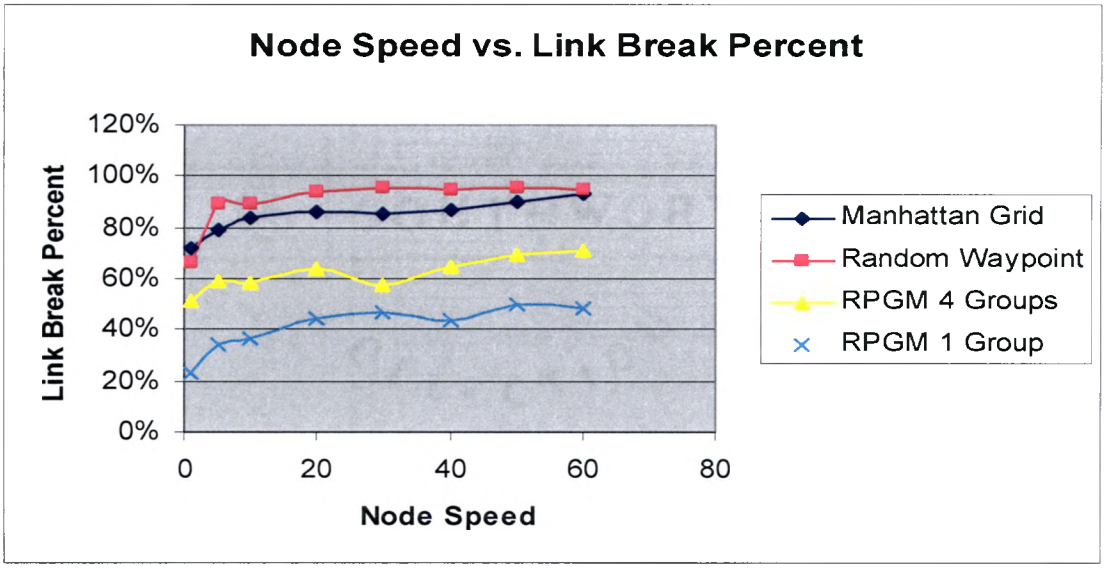


Fig 5.12: Percent of total number of links that broke in each mobility model as the speed changes, Transmission range = 100m.

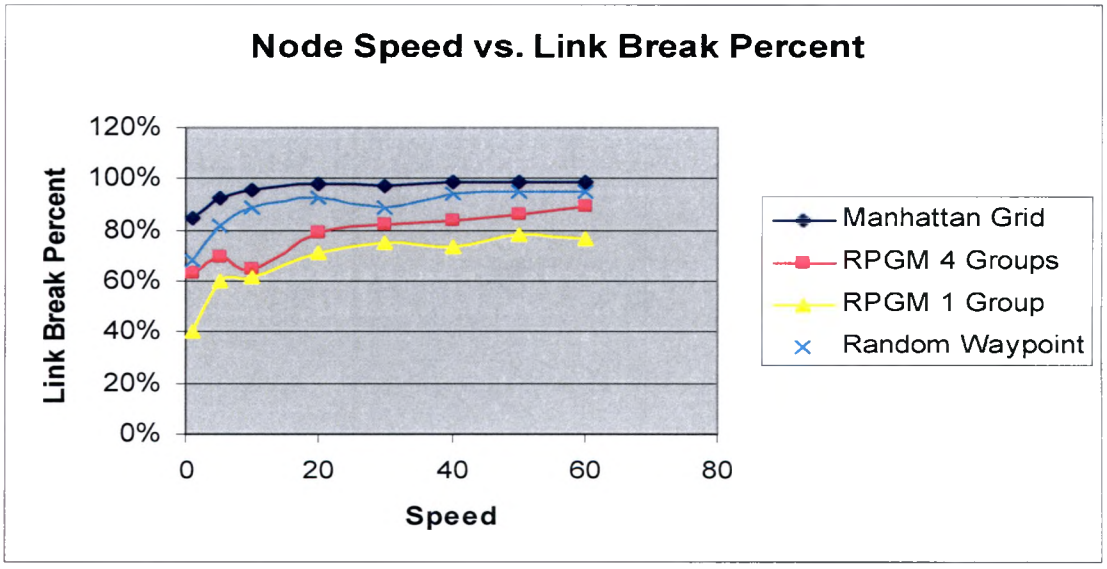


Fig 5.13: Percent of total number of links that broke in each mobility model as the speed changes, Transmission range = 50m.

(b) Percent of Total Links that Broke vs. Node Speed

Another interesting effect of mobility model on MANET is the percent of total number of links that broke at various speeds. Shown in Figures 5.12 and 5.13 are two plots of node speed against the percent of total number of links that broke in each

mobility model at transmission a range of 50m and 100m respectively.

The percent of total links that broke also differentiates between various mobility models clearly. Similar to the average link duration the percent of total links that broke also is almost in the same order: Manhattan model < Random Waypoint < RPGM 4 groups < RPGM 1 Group. However the Random Way point model performs worse than the Manhattan model at a higher transmission range.

5.6 Feasibility Study of EPRT

In this section we present a brief feasibility study of the EPRT technique. What makes MANETs different from the infrastructure based wireless networks is the dynamic topology. Due to mobility some links in a MANET may break while newer links may be added to the topology, making the topology dynamic. In such challenging conditions the choice of the routing protocol must be made very carefully. Not all protocols are necessarily suitable for a given MANET scenario. Therefore, in the rest of this section we will provide a basic analysis of MANET scenarios in which EPRT can be and can not be used.

The two characteristics of a network graph that we studied in the above section, Average Link Duration and Link Break Percentage are very clearly affected by not only the mobility model but also the transmission range and node speed. Therefore we can not make the choice of EPRT based on a single parameter. For example, we can not say that the EPRT can be used for all mobility models and all transmission ranges as long as the speed is less than 10m/s.

What we can observe from 5.13 and 5.12 is: in Manhattan model and Random Waypoint model, at high speed like 50m/s and 60m/s the percentage of links that broke is over 90% for transmission range of 100m and close to 100% for transmission range of

50m. At lower transmission range (50m), the percentage of links broke even in the group mobility models like RPGM 1 group as well as RPGM 4 groups is beyond 70% for speeds greater than 20m/s. Similarly in Figures 5.10 and 5.11 we can observe that the average link duration is very short for Random Waypoint and Manhattan Models, while that of the group mobility models is significantly higher.

Although proactive protocols such as package routing technique presented in [48], LCMRMG, EPRT etc, are traditionally considered to be suitable candidates for MANETs with lower node speeds, looking at the above results we can predict that these protocols can prove to be very expensive not only at high speeds but also at low speeds with mobility similar to Manhattan grid model or Random Waypoint model. These proactive protocols can be safely used in scenarios with high group mobility at low speeds.

Instead of finding combination of parameters suitable for an individual protocol or a class of protocols, we can use derived parameters such as relative velocity. These derived parameters are typically a cumulative effect of more than one simple parameter. For instance, relative velocity combines the effects of node speeds as well as node motion. MANETs of low node speeds are not necessarily suitable candidates for proactive protocols, but the networks with low relative velocity are.

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this thesis we developed an enhanced package routing technique (EPRT) on top of the Package Routing Technique proposed in [48]. The original technique is a spanning tree based routing technique, in which a generation table is associated with each node to keep track of relations between mobile hosts. Simplicity and no route acquisition latency are the important features of this algorithm. However, this algorithm has several drawbacks. It builds the routing tree starting at the root node, choice of which has a tremendous impact on the convergence time and routing overhead of this algorithm. Such a critical root node is chosen randomly. A root node once selected remains the root for the life of the network, which increases the chances of network falling apart due to the failure of single root node. This technique does not support node failure; it is assumed that the nodes never fail. Despite several alternative paths in the actual network graph, in [48] there is only one route between a given source and destination pair. This limited number of routes reduces the reliability and throughput of the network.

Although the protocols LCMRMG [49] and LCMRMGCS [50] claim to over come the above mentioned problems of [48], an analysis proves them to be worse than [48]. LCMRMG for instance leads to tremendous increase in control overhead; it maintains several generation tables at each node. In the worst case, LCMRMG might even end up maintaining all the links in the network graph. LCMRMG promises to establish alternative routes, on demand, upon detecting spikes in traffic in small local

areas of the network. In reality LCMRMG builds a network wide tree rather than few newer paths in a small region of the network.

LCMRMGCS uses a forest of trees rather than a spanning tree. To achieve locality caching, LCMRMGCS uses the same physical location based technique used by LCMRMG. By creating new trees, LCMRMGCS does not add any more new links to create alternative paths; it rather breaks the spanning tree into smaller sub trees with limited communication between them. Both [49] and [50] fail to address the issues of node failure or incomplete support for mobility.

The Enhanced Package Routing Technique (EPRT) addresses the issue of bad root choice by using the center of the tree as the root. A better choice of root in EPRT lead to a significant reduction of the time taken by the protocol, number of messages used by the protocol to converge and the also route lengths. Root node in EPRT is not static; it changes as the topology changes. EPRT supports node failure, regardless of weather the failed node is a root, leaf or any other node. EPRT will have significantly less over all routing overhead in comparison to [48], [49] & [50]. Table 6.1 shows a comparison of EPRT against the above mentioned three protocols.

It is extremely tedious, if not impossible, to test new MANET protocols on a real MANET. Testing on such platforms involves developing protocol implementations for various proprietary or non-proprietary operating systems, and deploying such implementations on a set of devices in the network. Due to the enormous complexity involved in creating such test beds, majority of MANET research today is verified using simulations.

MANET simulations can be regarded as a set of computer programs that try to

	Package Routing	LCMRMG	LCMRMGCS	EPRT
Root/s	One static root	Several dynamic roots	One static root	One dynamic root
Node Failure Support	NO	NO	NO	YES
Back up Routes	One route between all source destination pairs. No backup routes	Backup routes available	One route between all source destination pairs. No backup routes	Backup routes available
Maximum Time Required for the Convergence	D(G) + D(T)	D(G) + D(T)	D(G) + D(T)	D(G) + D(T)/2
Maximum Number of Messages Required for the Convergence	4V(G) - 3	4V(G) – 3	4V(G) – 3	3 V(G) + D(G) -2
Longest Path in the Routing Graph	2D(G)	2D(G)	2D(G)	D(G)
Maintenance	Very few links maintained. Maximum number of links maintained is V(G) – 1	In the worst case, this protocol end up maintaining the whole set of links in the graph: E(G)	Very few links maintained. Maximum number of links maintained is V(G) – 1	The maximum number of links maintained is: V(G) – 1. However, the protocol might know all the links E(G), due to promiscuous operation.

Table 6.1: Comparison of Package Routing, LCMRMG, LCMRMGCS and EPRT.

imitate the behavior of mobile nodes as closely as they are programmed to imitate. These virtual mobile stations react to various physical or topological changes, if such change

data is provided to them. Simulations can be used to study behavior of individual nodes, the impact of each node's behavior on the whole network, impact of physical or topological changes on a node or whole network etc. Simulations can also be used to test the reliability, QOS, performance or any other factor related to MANETs or MANET protocols. In case of new protocols, simulation tools can be used to quantify various properties specific to the simulated protocol, and to compare the performance of the protocol against other existing protocols whose simulation results are verified against a real world data or test bed data.

The credibility of the claims made by the several researchers in much of the MANET related research is at a questionable stage today. The authors of [56] and [57], conducted a survey of MANET research published in the premiere conference for the MANET community, i.e., the Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc) from 2000-2004. They only included the full papers in their survey, not the poster papers. The survey showed the shortfalls in most of the published work when it came to the results claimed. This is because of the authors attempt to obtain results than performing a real study or even due to the lack of realistic idea of potential real world scenarios.

As a future work we propose a comprehensive study, and modeling of the impact of various mobility models on the topology of the network. It is these topological changes and traffic patterns that decide the feasibility of EPRT. Once we are able to model some realistic mobility patterns and study the impact of those mobility patterns on the topology of the network, we can decide the reasonable candidates for further investigation with some realistic traffic patterns in them. At that stage we propose to implement EPRT on a

reasonable simulation platform such as Glomosim [59] or NS-2 [58], which gives the user the control of input parameters at all layers of the communication.

BIBLIOGRAPHY

[01] Philippe Jacquet and Anis Laouiti, "Analysis of mobile ad-hoc network routing protocols in random graph models," Tech. Rep. 3835, INRIA, Domaine de Voluceau, Rocquencourt, BP 105, 78 153 Le Chesany Cedex, France, 1999.

[02] Philippe Jacquet, Paul Muhlethaler, Amir Qayyum, Anis Laouiti, Laurent Viennot, and Thomas Clausen, "Optimized link state routing protocol," Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-manet-olsr-04.txt, March 2001, Work in Progress.

[03] Bhargav Bellur, Richard G. Ogier, and Fred L. Templin, "Topology broadcast based on reverse-path forwarding (tbrpf)," Internet-Draft http://www.ietf.org/internet-drafts/draft-ietf-manet-tbrpf-05.txt,March 2002, Work in progress.

[04] Bhargav Bellur and Richard G. Ogier, "A reliable, efficient topology broadcast protocol for dynamic networks," in Proceedings of INFO-COM 1999.

[05] Shree Murthy and J.J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," Tech. Rep., Computer Engineering, University of California at Santa Cruz, Santa Cruz, CA 95064, 1996.

[06] D. Bertsekas and R. Gallager, Data Networks, pp. 297333, Prentice-Hall, Englewood Cliffs, N.J., 1987.

[07] M. Scott Corson and Vincent Park, Link Reversal Routing, chapter 8,pp. 255298, In Perkins [1], 2001.

[08] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, and Jorjeta Jetcheva, "A performance comparison of multi-hop wireless adhoc network routing protocols," in Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, ACM, Dallas, TX. ACM, October 1998.

[09] C. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," in Proceedings of the ACM Conference on Communications Architectures, Protocols and Applications (SIG-COMM), pages 234244, 1994.

[10] J. Moy, OSPF version 2, RFC-2178, July 1997.

[11] T. Camp, J. Boleng, B. Williams, L. Wilcox, and W. Navidi, "Performance comparison of two location based routing protocols for ad hoc networks," in Proceedings of the 21st Annual Joint Conference of the IEEE Computer and Communications Societies (Infocom 2002), pages 16781687, 2002.

[12] C. Perkins, E. Belding-Royer, and S. Das, "Ad hoc on demand distance vector (AODV) routing. Internet Draft: draft-ietf-manet-aodv-11.txt," June 2002.

[13] D. Bertsekas and R. Gallager, *Data Networks* pp 297-333,Prentice-Hall, Englewood Cliffs, N.J.,1987.

[14] C.-K. Toh, "Long-lived ad hoc routing based on the concept of associativity," Internet Draft: draft-ietf-manet-longlived-adhoc-routing-00.txt, March 1999.

[15] V. Park and S. Corson, "Temporally-ordered routing algorithm (TORA) version 1, functional specification," Internet Draft: draft-ietf-manet-tora-spec-04.txt, July 2001.

[16] D. Johnson, D. Maltz, Y. Hu, and J. Jetcheva, "The dynamic source routing protocol for mobile ad hoc networks," Internet Draft: draft-ietf-manet-dsr-07.txt, February 2002.

[17] M. Gerla, X. Hong, L. Ma, and G. Pei. Landmark routing protocol (LANMAR).

[18] S. Basagni, I. Chlamtac, V.R. Syrotiuk, and B.A. Woodward, "A distance routing effect algorithm for mobility (DREAM)," In Proceedings of the Fourth Annual ACM International Conference on Mobile Computing and Networking (MobiCom 1998), pages 7684, 1998.

[19] I. Stojmenovic and X. Lin, "Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks," IEEE Transactions on Parallel and Distributed Systems, 12(10):10231032, 2001.

[20] R. Jain, A. Puri, and R. Sengupta, "Geographical routing using partial information for wireless ad hoc networks," IEEE Personal Communications, pages 4857, February 2001.

[21] Z. Haas, M. Pearlman, and P. Samar, "The interzone routing protocol (IERP) for ad hoc networks," Internet Draft: draft-ietf-manet-zone-ierp-01.txt, June 2001.

[22] Z. Haas, M. Pearlman, and P. Samar, "The intrazone routing protocol (IARP) for ad hoc networks," Internet Draft: draft-ietf-manet-zone-iarp-01.txt, June 2001.

[23] B. Karp and H. T. Kung, "GPSR: Greedy perimeter stateless routing for wireless networks," In Proceedings of the Sixth Annual ACM International Conference on Mobile computing and Networking (MobiCom 2000), pages 243254, 2000.

[24] Charles E.Perkins and Pravin Bhagawat, "Highly dynamic destination sequenced distance-vector routing (dsdv) for mobile computers," in ACM SIGCOMM'94

Conference on Communications Architectures, Protocols and Applications,1994, pp 234-244.

[25] Charles E.Perkins and Pravin Bhagawat, "DSDV *Routing over a Multihop Wireless Network of Mobile Computers*," chapter 3,pp 53-74,In Perkins [1], 2001.

[26] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael degermark, "Scenario-based performance analysis of routing protocols for mobile adhoc networks," in *Proceedings of The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99)*. ACM/IEEE, August 1999,pp 195-206.

[27] Rajendra V. Boppana and Satyadeva Konduru, "An adaptive distance vector routing Algorithm for mobile ad hoc networks," in *Proceedings of the Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies*, 2001,vol .3 pp 1753-1762.

[28] Vincent D.Park and M.Scott Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," in *Proceedings of INFOCOM 1997*.

[29] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," Mobile Networks and Applications, 1(2):183197, 1996.

[30] Lars Christensen and Gitte Hansen, "The optimized link state routing protocol," M.S. thesis, Department of Computer Science, Aalborg University, Fredrik Bajers Vej 7, 9220 Aalborg Ost, 2001.

[31] B. Bellur, R. Ogier, F. Templin, and M. Lewis, "Topology broadcast based on reverse-path forwarding (TBRPF)," Internet Draft: draft-ietf-manet-tbrpf-05.txt, March 2002.

[32] I. Stojmenovic and X. Lin, “Loop-free hybrid single-path/flooding routing algorithms with guaranteed delivery for wireless networks,” IEEE Transactions on Parallel and Distributed Systems, 12(10):1023,1032, 2001.

[33] Chai-Keong Toh, "Associativity-based routing for ad-hoc mobile networks," Tech. Rep., University of Cambridge Computer Laboratory, Cambridge CB2 3QG, United Kingdom, 1996.

[34] R.K.Padmanaban, Prof. Dr. V.Ramachandra, S.Manikandan,R.Naveenan, "Optimized associativity-based threshold routing for mobile adhoc networks," Tech. Rep., College of Engineering, Guindy, Anna University, 2001.

[35] Josh Broch, David A. Maltz, David B. Johnson, Yih-Chun Hu, andJorjeta Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in Proceedings of the Fourth Annual ACM/IEEE International Conference on Mobile Computing and Networking, ACM, Dallas, TX. ACM, October 1998.

[36] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark, "Scenario-based performance analysis of routing protocols for mobile ad-hoc networks," in Proceedings of The Fifth Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom'99). ACM/IEEE, August 1999, pp. 195206.

[37] Samir R. Das, Charles E. Perkins, and Elizabeth M. Royer, "Performance comparison of two on-demand routing protocols for ad hoc networks," in Proceedings of INFOCOM 2000 Conference, Tel-Aviv, Israel, March 2000.

[38] David B. Johnson and David A. Maltz, "Dynamic source routing in ad hoc wireless networks," Tech. Rep., Computer Science Department Carnegie Mellon University, 5000 Forbes Avenue Pittsburgh, PA 15213-3891, 1996.

[39] David B. Johnson, David A. Maltz, and Josh Broch, "DSR The Dynamic Source Routing Protocol for Multihop Wireless Ad Hoc Networks," chapter 5, pp. 139-172, In Perkins [1], 2001.

[40] David A. Maltz, Josh Broch, and David B. Johnson, "Experiences designing and building a multi-hop wireless ad hoc network testbed," Tech. Rep., The CMU Monarch Project Computer Science Department Carnegie Mellon University, Pittsburgh, PA 15213, 1999, http://www.monarch.cmu.edu/.

[41] Ionut D.Aron and Sandeep K.S. Gupta, "On the scalability of on-demand routing protocols for mobile ad hoc networks: An analytical study," Journal of Interconnection Networks, Jan. 2001.

[42] Xiaoyan Hong Mario Gerla and Guangyu Pei, "Landmark routing for large ad hoc wireless networks," in Proceedings of IEEE GLOBECOM 2000, San Francisco, CA, Nov. 2000.

[43] Guangyu Pei, Mario Gerla, and Xiaoyan Hong, "Lanmar: Landmark routing for large scale wireless ad hoc networks with group mobility," in proceedings of the ACM/IEEE Workshop on Mobile Ad Hoc Networking and Computing (MOBIHOC), Boston, MA. ACM/IEEE, August 2000, pp. 1118.

[44] Stefano, Basagni, Imrich Chlamtac, Violet R. Syrotiuk and Barry A.Woodward, "A distance routing effect algorithm for mobility (dream)," in Proceedings of the ACM/IEEE International Conference on Mobile Computing and Networking, Dallas, TX. Oct. 1998, pp. 7684, ACM.

[45] Brad Nelson Karp, "Geographic Routing for Wireless Networks," Dissertation, Harvard University, Cambridge, Massachusetts, Oct. 2000.

[46] Linda Briesemeister, "Group Membership and Communication in Highly Mobile Ad Hoc Networks," Dissertation, Technische Universtiaet Berlin,Nov. 2001.

[47] Rene Meier-Laurent Mazare, Vinny Cahill Marc-Olivier Killijian, Raymond Cunningham, "Towards group communication for mobile participants," Tech. Rep., Trinity College of Dublin, Dublin 2, Eire, 2001.

[48] X. Chen and X. D. Jia, "Package Routing Algorithms in Mobile Ad-Hoc Wireless Networks," the Proceedings of the 2001 ICPP workshops on wireless networks and mobile computing, September 2001, p.485-490.

[49] Xin Zhang, "Design and Simulation of Multi-Root Multi-Generation Package Routing Algorithm for Mobile Ad Hoc Networks," M.S Thesis, Texas State University-San Marcos, December 2004. (Albert B. Alkek Library, Call Number: AS36.T46 Z485)

[50] Li Zhuojing, "Multi-Root Multi-Generation With Color Schema Routing Algorithm for Mobile Ad Hoc Networks," M.S Thesis, Texas State University-San Marcos, May 2005. (Albert B. Alkek Library, Call Number : AS36.T46 L5127)

[51] Wuxu Peng, Xin Zhang and Kia Makki, "Locality Caching Multi-Root Multi-Generation Routing Algorithm in Mobile Ad Hoc Networks," Proc. of the 12th International Conference on Computer Communications and Networks, Dallas, Texas, Oct.20-22, 2003.

[52] Wuxu Peng, Zhuojing Li, and Furman Haddix "Practical Spanning Tree Based MANET Routing Algorithm," Proc. of the 14th International Conference on

Computer Communications and Networks On page(s): 19- 24 ISSN: 1095-2055 ISBN: 0-7803-9428-3.

[53] E Korach , D Rotem , N Santoro, "Distributed algorithms for finding centers and medians in networks," ACM Transactions on Programming Languages and Systems (TOPLAS), v.6 n.3, p.380-401, July 1984.

[54] Mohan Sharma , Jianhua Chen , Sitharama Iyengar, "Distributed algorithms for locating centers and medians in communication networks," Proceedings of the 1992 ACM/SIGAPP symposium on Applied computing: technological challenges of the 1990's, p.808-817, March 1992, Kansas City, Missouri, United States.

[55] T. Camp, J. Boleng, and V. Davies, "A Survey of Mobility Models for Ad Hoc Network Research, Wireless Communication & Mobile Computing (WCMC)," Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications, vol. 2, no. 5, pp. 483-502, 2002.

[56] S. Kurkowski, T. Camp, and M. Colagrosso, "MANET Simulation Studies: The Incredibles," ACM's Mobile Computing and Communications Review, vol. 9, no. 4, pp. 50-61, October 2005.

[57] S. Kurkowski, T. Camp, and M. Colagrosso, "MANET Simulation Studies: The Current State and New Simulation Tools," Technical Report MCS-05-02, The Colorado School of Mines, February 2005.

[58] Ionut D.Aron and Sandeep K.S. Gupta, "On the scalability of on-demand routing protocols for mobile ad hoc networks: An analytical study," Journal of Interconnection Networks, Jan. 2001.

[59] The network simulator, version 2 (ns-2). http://www.isi.edu/nsnam/ns.

[60] Global mobile information system simulation library (Glomosim). http://pcl.cs.ucla.edu/projects/glomosim.

[61] BonnMotion – A mobility scenario generation and analysis tool, Version 1.3. Available at: http://web.informatik.uni-bonn.de/IV/Mitarbeiter/dewaal/BonnMotion

VITA

Anil Maddiboina was born in Guntur, India, on September 01, 1978, the son of Maddiboina Krishna Rao and Savitri Devi. After completing his high school study, he entered Jawaharlal Nehru Technological University in 1997. After four years of study, he received the degree of Bachelor of Technology. In 2001, he entered Texas State University-San Marcos, pursuing a master's degree in Computer Science.

Permanent Address: 1704 Nelms Dr., #2115

Austin, Texas 78744

This thesis was typed by Anil Maddiboina