Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

PROTOCOLS FOR PACKET SWITCHED COMMUNICATION AND RELIABLE MULTICASTING IN FULLY-DYNAMIC MULTI-HOP WIRELESS NETWORKS

by
James Pelech

Designing protocols for a fully dynamic wireless packet switched networks pose unique challenges due to the constantly changing topology of the network. A set of protocols is presented that are capable of handling a fully dynamic wireless network in which switching centers and base stations are mobile as well as the end users. The protocols provide basic message delivery, network routing information updates, and support for reliable multicasting.

There are four contributions of this work: (i) a hierarchical architecture for a fully dynamic wireless network, (ii) improved routing and update protocols with reduced control traffic, (iii) a method to provide reliable multicasting in a wireless environment that is near optimal in terms of the number of messages sent, and (iv) a set of load balancing algorithms that allow the network to autonomously and dynamically reconfigure the network topology to even out the load on the base stations.

A detailed simulation of the protocols is developed and exercised to evaluate the performance of the protocols. For point to point delivery, the protocols successfully deliver all packets even when the rate of motion of the terminals causes more than 1/2 of them to be in a transitional state at any time. The results are similar for base station

PROTOCOLS FOR PACKET SWITCHED COMMUNICATION AND RELIABLE MULTICASTING IN FULLY-DYNAMIC MULTI-HOP WIRELESS NETWORKS

by
James Pelech

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of
Master of Science in Computer Science

Department of Computer and Information Science

May 1998

APPROVAL PAGE

PROTOCOLS FOR PACKET SWITCHED COMMUNICATION AND RELIABLE MULTICASTING IN FULLY-DYNAMIC MULTI-HOP WIRELESS NETWORKS

James Pelech

Dr. Bülent Yener, Thesis Advisor — Date
Assistant Professor of Computer and Information Science
New Jersey Institute of Technology, Newark, NJ

Dr. Dennis Karvelas, — Date
Assistant Professor of Computer and Information Science
New Jersey Institute of Technology, Newark, NJ

Dr. Murat Tanik — Date
Director, Electronic Enterprise Engineering
New Jersey Institute of Technology, Newark, NJ

BIOGRAPHICAL SKETCH

Author: James Pelech

Degree: Master of Science

Place of Birth: Boston, Massachusetts

Undergraduate and Graduate Education:

- Master of Science in Computer Science,
  New Jersey Institute of Technology, Newark, NJ, 1998

- Master of Science in Mechanical Engineering,
  Stevens Institute of Technology, Hoboken, NJ, 1987

- Bachelor of Science in Mechanical Engineering,
  Rensselaer Polytechnic Institute, Troy, NY, 1983

Major: Computer Science

To Teresa , Sabrina, and Jared

ACKNOWLEDGMENT

I would like to express my sincere appreciation to Dr. Bülent Yener for providing guidance and insight into the many issues that were confronted by this work. I also wish to thank Dr. Murat Tanik and Dr. Dennis Karvelas for all of their efforts in reviewing this work. Their comments and suggestions were very valuable.

Thanks also go out to Wesley Shanks and Tracey Dwyer who provided valuable comments and improvements during the review of the work.

Most of all, I wish to express my special thanks to my wife, Teresa, and children, Sabrina and Jared, who supported me throughout this effort and provided me the time and space to complete the work.

TABLE OF CONTENTS

TABLE OF CONTENTS
(Continued)

Chapter **Page**

TABLE OF CONTENTS
(Continued)

Chapter **Page**

LIST OF FIGURES

Figure **Page**

LIST OF FIGURES
(Continued)

Figure **Page**

LIST OF SYMBOLS

Symbol	Description
BS	Mobile Base station
B/TS	Bytes per Time Step
DR	data rate of the line (Bytes/time step)
DR_1	data rate of the link between the terminals and the BS
DR_2	data rate of link between the BS and its parent BS
DR_3	data rate between the parent BS (root) and the SC
DR_4	data rate between the switching centers
DU	Distance unit (unit of distance)
n	number of terminals attached to a single BS
m	number of BS that are part of a single tree in the VLA
p	number of trees in a VLA
PS	packet size of the data packet
q	number of VLAs.
SC	Switching center (assumed to be satellite based)
TM	Terminal- the users of the network
TS	Time step (unit of time)
VLA	Virtual Location Area, coverage area of a switching center

LIST OF SYMBOLS
(Continued)

μ	mean service time (packets/time step)
λ	mean arrival time (packets/time step)
ρ	λ/μ (dimensionless)

CHAPTER 1

INTRODUCTION

1.1 Objective

The objective of this thesis is to present an architecture and set of protocols for a fully dynamic wireless network in which every network entity is mobile, including switching centers (SCs) and base stations (BSs). The terminals (TMs) and the base stations are capable of dynamically changing location within the network topology while remaining in communication with other components of the network. This architecture and protocols support both point to point and multicast transmission.

The protocols to support routing for such a system are developed and shown to be robust for even extreme cases of mobility of the end users and base stations. It will be shown that possible race conditions associated with updating the routing information are properly handled, thus eliminating the possibility of inconsistent or inaccurate routing information. These protocols support both unicast and multicast transmission.

To prove the effectiveness of the protocols, a detailed simulation is presented. The simulation results show that the protocol is robust and accurate for wide mobility ranges of both the end users and the base stations

For autonomous networks, a set of load balancing algorithms are presented and demonstrated via the simulation. The load balancing algorithms provide a means for an autonomous network to share the load between the base stations

by moving lightly loaded base stations to assist the heavily loaded base stations. Both a centralized and distributed algorithm are presented.

1.2 Background Information

Recently, there has been significant interest in the area of wireless networks. Much of the work has centered on extending the current IP based networks to wireless users and in implementing Asynchronous Transfer Mode (ATM) on wireless Local Area Networks (LANs) [1,2,3,5,6,8,9].

Current research on wireless LANs provides mobility to a user, however, there are generally restrictions on the mobility of the network switching nodes. For example, the nodes may be moveable in a semi-static sense, they are easily moved but must be taken off line to do so.

There has also been extensive work in the area of extending the Internet Protocol (IP) to mobile users[1,7-10]. Here, the emphasis has been on how to extend access to/from a standard network to users of mobile equipment. The challenge is that the user may change locations in the network while it is in the process of communicating. This makes the job of routing more difficult. However, the entities that make up the network itself are static in location, both physical and logical.

There has also been some work in the area of providing multicasting capability to wireless networks [8,9]. The proposals to date have been inefficient. They either require excessive overhead in order to keep the routing data current, or they require a form of flooding to insure message delivery.

The work described herein differs from previous efforts in that it provides support for fully dynamic networks with multicasting capability even when the network is large and the mobility very high. Updating of the routing information and routing of the messages will be shown to remain stable and consistent for virtually any rate of mobility in the system. This will be shown to be true for point to point as well as multicast message services. Previous work in this area [2,3,8,9] exhibit problems when either the system is highly mobile or too large.

One of the most important aspects of any network is maintaining the routing information [4]. The three most common methods for dealing with this are (i) to have nodes share information with each other, (ii) to process all messages through a common, master node, or (iii) to flood the routing information through the network. Each of these methods have significant drawbacks in the area of dynamic wireless networks. The first method propagates the changes too slowly, the second requires a heavy concentration of resources in the central node (which may not remain central over time) and the third requires too much bandwidth.

In contrast with the above approaches, the model suggested in [4] is adapted. A distributed database that organizes the network nodes into trees is used to facilitate routing in the network. Partitioning of the network nodes and the global state information induces a *virtual network* on which the amount of state information is minimized at the expense of optimal routing.

This work extends the previous work in [4] in several ways: First, a new model is presented in which the switching centers, base stations, and terminals are all mobile. Second, a strict hierarchical architecture is employed to reduce routing update messages

and simplify the updating process. Third, the limitations on the size and shape of the network are eliminated. Fourth, support for reliable multicasting is added and lastly, load balancing algorithms are introduced into the protocol to improve overall network performance.

These protocols are important for applications where the load on the network fluctuates significantly in both location and time. Under these conditions, it is necessary to either provide sufficient hardware in all locations to handle the peak load (very costly) or to provide a mechanism for moving the capacity of the network around as the load on the network moves. These protocols support the later of these possibilities, thus potentially providing large savings in hardware costs. Two possible applications for these protocols are a LAN in which the terminals are highly mobile causing the distribution of the load on the system changes with time and a military wireless network supporting units deployed in the field.

The remainder of the work is organized as follows. In Chapter 2 the network model is presented and discussed. In Chapter 3, the algorithms for routing packets in the network and updating the network routing information is presented. In Chapter 4, compatibility with two of the existing dominant data network protocols is discussed. In Chapter 5, algorithms for providing autonomous load balancing is presented. In Chapter 6, the use of these protocols to support mulitcasting is handled. In Chapter 7, a method for provisioning limited network resources is suggested. In Chapter 8, results of simulations of the protocols are presented. In Chapter 9, this work is contrasted with previous work in this area. And finally, in Chapter 10 the conclusions are presented.

CHAPTER 2

NETWORK MODEL

In this chapter, the network model is presented. This model is used as the basis for the development of the routing protocols.

The network architecture is hierarchical and consists of a collection of switching centers, base stations and terminals as suggested in [4]. However, this model is enhanced in several significant ways: First, the SCs are permitted to be mobile. For example SCs can be deployed as a set of low orbit satellites to increase system flexibility and coverage. Second, the heterogeneous network is represented by a weighted graph. Each of the links and nodes contain a weight associated with the cost of sending and processing a message, respectively. In the previous work [4] there was little consideration given to the effects that limited network resources will have on the design and performance of the protocol. Third, the restrictions on the size and shape of the network have been eliminated. To support fully autonomous networks (motion of the base stations is automatically controlled), algorithms for reducing asymmetry have been introduced. Fourth, the methods used for updating the network routing information has been revised to simplify the messaging traffic. Fifth, reliability of the information has been increased by eliminating all race conditions. Lastly, the capability to perform reliable multicasting has been added.

2.1 Connectivity Model

In the architecture, the SCs serve as the backbone for the network and are connected by a logical ring. The BSs are organized in tree structures connected to the SCs and the TMs are connected to the BSs. The network architecture is shown in figure 2.1.

Each SC serves a discrete physical area. This coverage area for a SC is called a virtual location area (VLA). The union of the areas for all of the SCs comprise the entire service area for the network.

The tree structure for the BSs is the minimum depth required to connect all of the BSs. Forcing the tree to be of minimum depth provides a minimal path length between any two nodes (for a tree structure). The leaves of the tree are the TMs.

There is no maximum depth for the tree structure. There are no set limits on the number of BSs or TMs that can be supported by the network. However, as the number of TMs in a any portion of the system increases, the level of service that can be supplied to each TM decreases (due to the limited system resources, especially bandwidth).

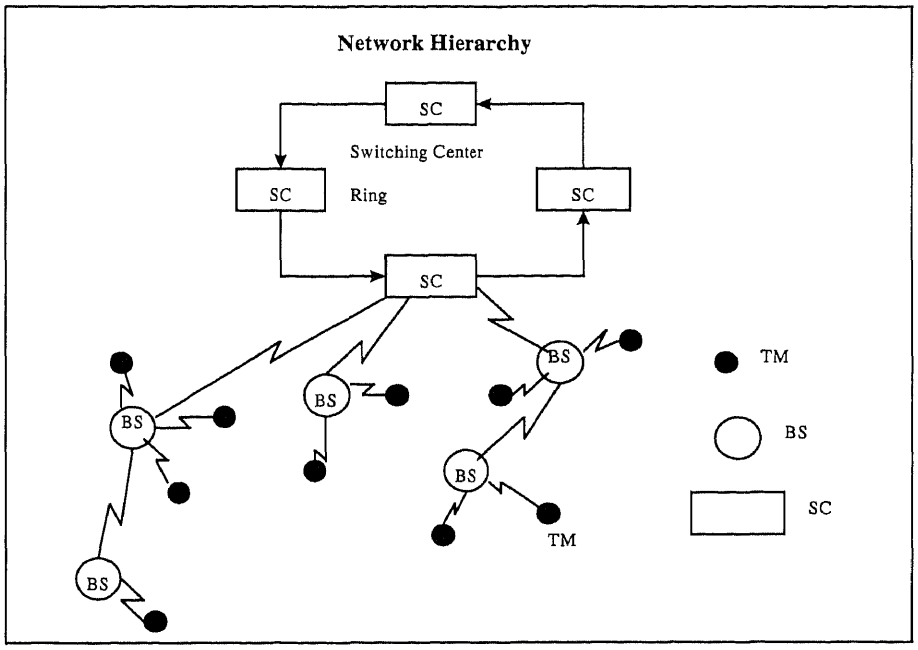

Figure 2.1: Network Architecture.

2.2 Communication Model

The network is based on a purely hierarchical model with objects in the network capable of communicating only with their parents and their direct children. As will be seen in the updating section, this will lead to a highly reliable method for ensuring that the routing information in the network is always accurate. The one exception to this hierarchical rule are the SCs which act as peers, passing information amongst them in a ring fashion.

Specifically then, SCs can communicate with the root BSs that they serve. In addition, the SC communicated with the previous SC in the ring to receive data and the next SC in the ring to sends data. BSs are capable of communicating only with their immediate children and their parent.

2.3 Control and Data Message Specification

The control message consists of the following fields:

1. Message destination ID. This is a network ID. Assumed 32 bits
2. Message source ID, 32 bits.
3. The type of message. Currently there are four types, TM move, TM add, BS move, BS add. therefore this field requires a minimum of 2 bits.
4. A time stamp. This should be no longer than 32 bits (probably 16 or less).
5. Acknowledgment flag, 1 bit.
6. If the message is a BS move, the network ID for all TMs communicating with the BS. This requires up to 32* N bits, where N is the number of TMs that the BS is serving.

The network IDs have the general form of an IP address in that they have the form netid.hostid. However, because of the mobility of the TMs within the network, the network portion of the ID must encompass the entire network that the host can roam within. When routing within the wireless network, only the host portion of the ID is used.

The size of the control message is therefore approximately 12+4*N bytes of data plus the bytes required for error correction, flow control, etc. As should be obvious, the BS move message is significantly larger than a move message for a TM. As will be demonstrated in the simulation section, even when the mobility rates of the TMs and the BSs are the same, the number of TM moves dwarfs the number of BS moves. For data messages, the only overhead information that is required for the routing is the ID of the destination (32 bits) and a time stamp (for deleting old messages).

There are, of course, other overhead data that has to be sent with the messages. Most notably there is error control, flow control, and sequence numbers. While these data are important to the successful implementation any set of protocols, they deal with issues that are outside the scope of this work and therefore are not considered here. Because much of the overhead associated with the actual implementation of the protocol is not addressed in this work, a complete packet format is not provided.

CHAPTER 3

LOCATION INFORMATION AND ROUTING ALGORITHM

In this chapter the organization of the routing information and the algorithm used for routing packets through the network are explained. For successful delivery of messages, it is necessary that (i) the location information is accurate, and (ii) a physical path (wireless link) to the destination exits. In this section, a method for organizing and maintaining location information as well as a method for routing messages based on this information will be demonstrated.

3.1 Distributed Network Database

In order to facilitate an efficient method for routing the messages through the network, the routing information is distributed to the BSs. Due to the hierarchical nature of the architecture, the only information that is needed at any single node is:

1. The ID of the parent node for this node,
2. The IDs of the nodes in the subtree below this node (both BSs and TMs), and
3. The "next node" for each node in the subtree (next hop in path to node).

The resource requirements of a BS is dependent on its location in the network with the maximum requirements occurring at the root nodes. This implies either that there must be a specialization of the BSs (where, for example, only certain BSs can become a root) or that the capabilities of all BSs have to be sized such that any of them are able to handle the maximum load expected for a root BS. The latter option is chosen to maximize

the flexibility of the network. For BSs that are not roots, actual resource allocation of shared resources (notably RF bandwidth) will be reduced commensurate with where the BS is in the network.

As with any hierarchical network design, there are issues with the ability to scale up the design. These protocols are intended to handle the traffic within a single wireless network and are not considered to be applicable to the broader internetwork case.

3.2 Routing Algorithm

There are two types of messages used by the routing algorithm: (i) data messages bound for TMs, and (ii) control messages bound for BSs. For a data message the following procedure is used:

1. The TM generates a data message, complete with destination ID and forwards it to its parent BS.
2. If the destination TM is in the subtree of the processing BS, the BS will forward the message to the appropriate node in the subtree (or directly to the TM). If the destination TM is not in the subtree, the BS forwards the message to its parent.
3. This process is repeated until the message reaches a SC or the message is delivered. If the message reaches the SC, it will either forward it to another tree in its VLA (if the destination TM is reported to be in that tree), or the SC will forward the message to the next SC in the ring.

Note that the method for delivering a message is straight forward and requires only that each of the nodes possess knowledge of the components that reside in the subtree that

is below them. The method that is used for the delivery of a control message is analogous.

Since the routing algorithm is hierarchical in nature, it is simple (and inexpensive) to maintain the routing database, as will be shown. With one small exception, the only nodes that handle a control message are the ones required to deliver it. This procedure will be shown to remain robust for even the highest rates of motion that could be generated in the simulation.

Within the framework of the network layout, this protocol will provide the shortest path. However, due to the hierarchical nature of the network, there are shorter possible paths that the protocol does not support. For example, if there are two BSs that are within communication distance of each other but reside in different subtrees, the message must travel up to the first common node of the two subtrees and then down the other subtree to the destination. Although this routing requires more hops than the minimum, by making the trees of minimum depth, the degree to which this is a problem, is diminished.

Timers and timeouts are used to cull lost messages from the network and to ensure that lost control messages are resent. While this adds to the complexity of the network, it provides robustness to the protocols. Even though the protocol itself does not cause messages to be lost, there may exist circumstances under which messages can become lost. To avoid unnecessary congestion problems, the protocol must have a mechanism to deal with lost messages.

3.3 Location Update Procedure

The location update procedure is the method used by the network to ensure that the routing information held by the network nodes is kept accurate. As in any routing protocol, the accuracy of the routing information is crucial to the success of the protocol. The update procedure executes when a network node moves from one logical network location to another. There are several cases to consider:

Case I: the TM moves to another BS

Case II: a BS moves to another tree in the network (a BS will not change location within a single tree)

Case III: a BS is moving to another location in the network while a TM is attempting to either attach to or leave that same BS.

Case IV: a BS is moving to another a location in the network at the same time that one of its child BSs is moving.

Case I

This is the simplest case because it involves the movement of only a leaf in the network. Figure 3.1 illustrates the change in the network. The decision to move to another location in the network is made by the TM. The TM may choose to move to another BS due either to a poor signal or overloading of the BS. For a TM to move to another BS, the new BS must be measurably better than the current BS. The steps involved in the move therefore are:

1. The TM determines that it needs to move and the BS that it needs to move to (based on signal strength, loading conditions, or other criteria)
2. The moving TM sends a message directly to the new BS, BS_{new} Included in the message will be the ID of the old BS (BS_{old}). BS_{new} will create a control message, setting the destination to BS_{old}. This message will be forwarded through the network to the old BS. The TM will inform the old BS that it is in the process of moving. The old BS will then mark the mobile as moving by setting a flag in its database and will set a timer and wait for the message to come through the network from the new BS.
3. As the message from BS_{new} propagates back to BS_{old}, each node will update its database entry for the TM. In the case where BS_{old} and BS_{new} exist in the same tree, the message will be forwarded up to the SC for the VLA to ensure that all ancestor nodes have correct entries in their databases. This is the only case where the control message is sent to a node not directly involved in the message routing.
4. When the ACK gets to BS_{old}, BS_{old} removes the TM from its database and orders the TM to communicate with BS_{new}. The move is now complete.

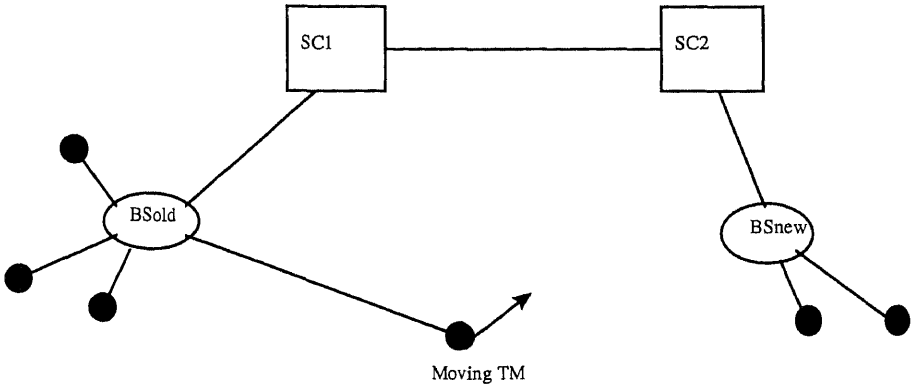

Figure 3.1: One TM moving from one BS to BS in different VLA.

Note that the update is handled by a single message/reply and does not require propagation to nodes other than those in the delivery path. Also, the time that the TM is out of reach is only the time required by the mobile to "retune" to the new BS, which would typically be a small fraction of a second.

Case II

Moving a BS, as shown in figure 3.2, is more involved than moving a terminal since the BS serves terminals and possibly has other BS that exist in its subtree. When a BS moves, something has to be done with the subtree for which it is the root.

Because of the added cost of moving a BS, it is essential that BSs change their location in the network hierarchy only when there is significant advantage in doing so. When a BS move is required, the operations for the move are similar to that of the move of a terminal with some exceptions. Those exceptions are:

1. The moving BS attempts to find a new location in the network that is at minimum depth in the network. The tree depth (i.e. layer) of the candidate base stations is determined by the moving BS based on information contained in the overhead message that each BS sends out. Contained in the overhead message is information about the BS including its ID and layer. The new parent BS must also be in the same VLA that the moving BS is currently in (note that this VLA may not be the same one as the old parent of the moving BS). Minimum depth is achieved by looking for an acceptable (sufficient signal strength and available capacity) BS that is the closest to a SC in the network, based on a layer number broadcast by the BS.
2. Upon finding a new location in the network to move to, a control message is sent to initiate a BS move. This control message is sent to the current parent of the moving BS. This control message is then routed to the new parent of the moving BS via normal routing through the network.
3. When the message reaches the new parent for the moving BS, the new parent will enter the moving BS into its routing table. It will then generate an Acknowledgment message send the acknowledgment back to the old parent of the BS. At each BS along the path of the acknowledgment, the BS will determine the correct action that is needed to properly update the routing tables. In general this will consist of either adding the moving BS to the routing information (if the BS is an ancestor of the new parent BS) or deleting the moving BS (if the BS is an ancestor of the old parent BS.
4. When the message to move reaches the old parent, the old parent updates the entry for the moving BS and starts to rebuild the subtree for which the moving BS was the root, if necessary. This is done by choosing the child with the strongest signal. The

parent sends this child its directory information for all the BSs in the subtree. To complete the build, the new root for the subtree must query each of its new children BSs (formerly siblings) to determine what is in their subtree and to inform them that it is now the root.

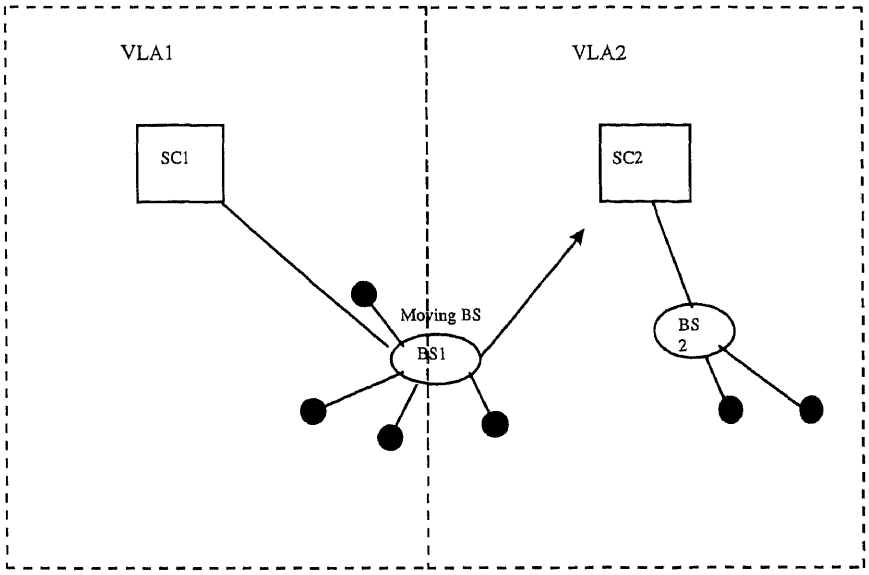

Figure 3.2: BS moving from one VLA to another.

Left out intentionally was any discussion of what happens to the terminals that were being served by the moving BS. The network protocols require that the TMs decide which BS to connect to based on criteria evaluated by the TM (signal strength is one such criteria). Since the decision to move from one BS to another is left entirely to the TM, the TMs are free to remain connected to the logically moving BS when it moves from one location to another in the network. Note that a logical move of the BS does not necessarily correspond to a significant physical move. If the movement of the BS results

in an unacceptable signal at any or all of the TMs it serves, those TMs will initiate a move on their own. Therefore it neither necessary, nor desirable, to force a TM to move to a new BS when its parent BS moves.

Case III

TM X is attempting to attach or leave BS Y that is also in the process of moving, as shown in figure 3.3. This case requires special handling because when the BS sends a message to its parent to initiate that move, it includes all of the TMs that it serves. This list of TMs is then used to update the intermediate nodes in the network, as well as the new parent. If the TMs served by the BS change during the BS move, the distributed database for the network may become inconsistent. An inconsistent routing database can lead to lost messages and severe congestion of the network.

There are four sub cases to this problem, (i) the TM tries to move after its serving BS starts a move, (ii) the serving BS starts to move while the TM is moving, (iii) the TM attempts to move to a BS that is also moving, (iv) the receiving BS starts to move after the TM starts to move. For case (i) we simply do not allow the TM to move while its serving BS is moving. If a TM attempts to move when its parent BS is in the process of moving, the request will be denied. The TM will then wait a short period of time and try again. This effectively briefly delays the TM move but does not prevent it entirely. Cases iii and iv are the reason that the mobile requests a move directly from the new BS. With this method, if the new BS is moving, it rejects the move request directly and the TM (temporarily) stays where it is. If the new BS is not moving at the time of the request, then it accepts the TM. If the new BS then starts its own move before the TM actually

moves, the new BS will include the moving TM in its own move message. In either case, the distributed database will remain correct and consistent. Note that if the TM is blocked from moving, it will wait a short period of time and try again. The period of time it will wait will be roughly equal to the expect time required to (logically) move a BS in the network.

Case(ii) is similar to case (iv). When the serving BS builds the control message for its own move, it will check all of the TMs that it serves and marks in the control message the TMs that are moving. Each intermediate node then handles the marked TMs specially to ensure proper updating of the routing data.

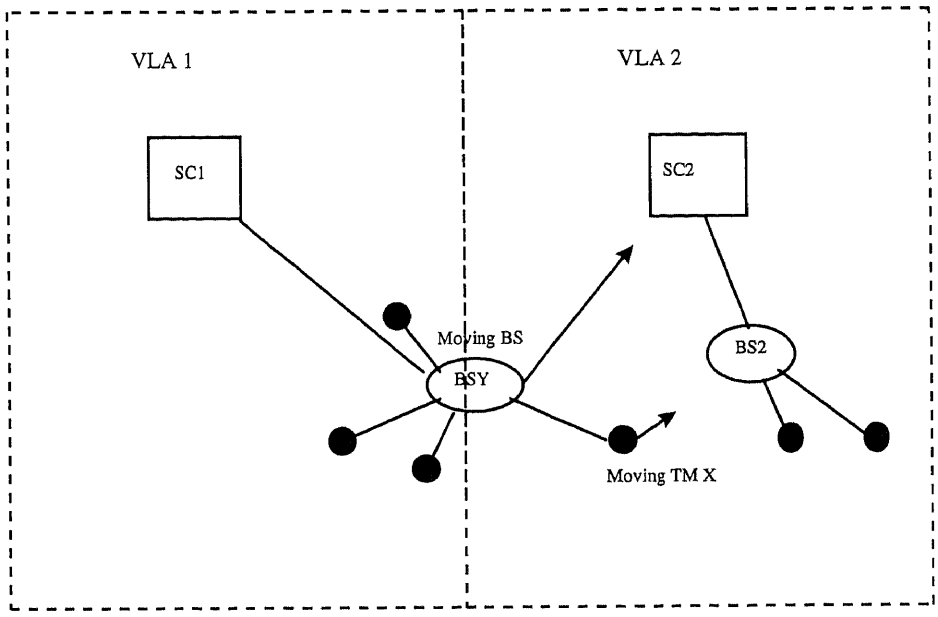

Figure 3.3 TM moving while its serving BS is moving.

Case IV

BS A is moving and one of its child BSs (BS B) is also moving (figure 3.4). While this case sounds like an extension of the previous case, it is actually quite different, and it turns out, trivial. The problems caused by the previous case were the result of the fact that the BS takes its TMs with it when it moves. In the present case, the two BS moves can be treated completely independently, with no loss of accuracy in the routing information. BS A and BS B send their respective move messages when they are ready to move. The move message for BS A will contain information only about BS A (and the TMs that it directly supports) and the same is true for BS B. Since BS A will drop all information it has about BS B after BS A moves, it is not important that BS A be aware of BS B's move. The same is true for BS B. Therefore the moves are independent and can occur with any time relation. This is true even if BS A receives the move request from BS B but does not forward it until after BS A has moved to another part of the network. In this case, the message from BS B that is being forwarded by BS A will still be routed to the new parent for BS B. Since the message is not acted on by the network until it reaches the new parent, the path the message takes to get to the new parent is unimportant. As long as the replies to the move messages come back to the parent of A, the routing information will remain correct.

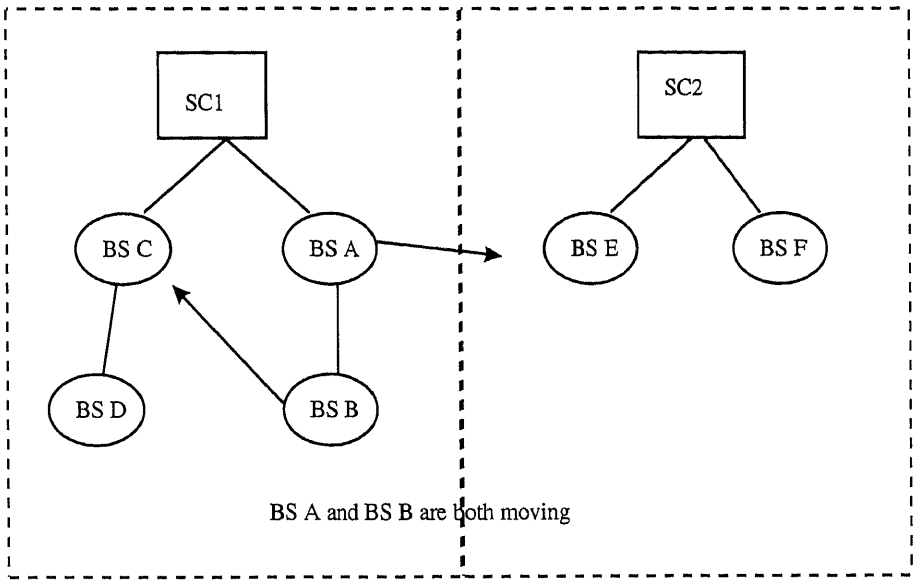

Figure 3.4: A BS is moving as well as one of its children BSs.

This last case actually points out something interesting in the protocol. Since the updating of the routing information is dependent on the control messages themselves being properly routed (going all of the ancestor nodes for both the old and new parent BS), it is important that care be taken to ensure that this happen. In specific there is a case where the protocol can break down if the control messages are not treated properly. When a BS moves to another location in the network, it may have control messages in its queue that are waiting to be sent out. Assume that one of these control messages is an acknowledgment for a move (of either a BS or a TM) and that the BS that this control message is at moves to another tree in the network. The route that this acknowledgment will take to get to the old BS of the moving entity will no longer go through all of the ancestors of new BS and their routing information will not be properly updated. The routing information for that moving entity would therefore be broken resulting in lost packets and possibly severe congestion in the network. The cure for this problem is to

force the moving BS to empty its control message queue prior to actually moving to the new location in the network. This can be accomplished by breaking the incoming links to the moving BS (to ensure that it receives no more messages) prior to breaking the out going links.

CHAPTER 4

INTEGRATION OF THE MULTIHOP NETWORK INTO A GLOBAL DATA NETWORK

The proposed set of protocols is designed for the special case where the network is highly mobile. While it performs well under those conditions, it clearly is not a protocol that would be adopted for a more conventional network. While a network may be deployed using these proposed protocols, it is highly likely that the rest of the world will continue to run the other more widely used protocols, such as TCP/IP and ATM. Any network deploying these dynamic wireless protocols will therefore have to be compatible with these other protocols, or risk being isolated. In this chapter it will be shown that it is possible, especially in the case of TCP/IP to provide compatibility, and thus enable connections to other networks.

4.1 TCP/IP

TCP/IP is widely used as the protocol to pass data from one computer to another in an internet environment. TCP/IP relies on the concept of a unique network address for each of the nodes in the network. The address (IP) is broken down into 4 one byte numbers. These numbers contain the network the node is on as well as the node address within the network. For large networks, the node number may be further broken down into a sub network number and a node number. To deliver a packet, a router determines from the network number which network the destination resides on and then determines from its

routing tables the next hop in the path to that network. Once inside a network, routers make use of the subnet number and the node number to find the destination node. For a standard network, the IP address represents a (semi)static location within the network.

The proposed protocol also makes use of network ID numbers to determine routing, although in a different manner. To merge the two protocols requires that gateways be provided between the dynamic wireless network and the remainder of the Internet. The wireless network will make use of the IP address for routing purposes (since it will be a unique address it will meet the address requirements of the wireless network). When the gateway sees a packet that has an address that belongs to the wireless network, it treats the packet in the manner described by the protocol. When the IP address does not belong to the network, it is routed out to the Internet using standard IP routing methods. The gateway is therefore required to run both a standard IP protocol as well as the routing protocol described by this paper.

Assignment of the IP addresses for the wireless network would be as follows. When a terminal first attempts to access the network, it would be required to obtain an IP address. The BS that interfaces with the terminal during this registration process would request an IP address be assigned. This request would flow up the tree to the serving switching center. The SC would assign an IP address from its own available pool. This IP address would then remain with the terminal for the entire time that it is connected to the network. When the terminal logs off the network, it would then release the IP address.

If the SC runs out of addresses it can request an address from one of the other SCs in the wireless network. Although this will lead to inefficient routing, it is considered superior to denying service. Each of the VLAs is considered to be a separate network

(according to the IP numbers) and the SC acts as a gateway to the remainder of the Internet. When a packet from outside this network is sent to the network, it would go through the gateway (SC) for the particular network address. In general this should be the VLA that the terminal is in since most terminals will stay within the same *VLA* in which they obtained their IP address However, there is no requirement that a terminal stay within the same VLA.

If the terminal moves from one SC to another, the original SC will forward the packet to the next SC in the ring until the proper SC is found (this is the normal behavior of the network). Note that the level of service that is received within the dynamic wireless network is approximately that of the external IP network. Since of the protocols are connectionless, they both provide service on a "best effort" basis.

It is also possible to have intra network communication that is more secure in nature than this model would allow. The problem with the above approach is the network addresses maybe dynamically allocated. That is, when a TM comes on to the system, it is assigned an ID. When it leaves the system, the ID is released back into the unused pool of IDs and may be reassigned to another TM. For example, TM X is assigned ID ZZ. At some point later TM X logs off the system and ID ZZ is released. TM Y then logs on to the network and ID ZZ is assigned to it. Therefore it is possible that a packet that was intended for user X could end up at user Y. To prevent this, there are a couple of possibilities. The simplest (from the network's point of view) is to have the TMs perform a handshake prior to the exchange of any sensitive material. The TMs would be required to initiate the handshake as well so that the network would not be involved. The second possible solution is to have the network ID consist of two parts. The first part is the IP

address while the second part is a terminal specific ID that is unique to each terminal (a hardware specific address). When a BS gets a packet, it first checks to determine whether it serves the destination terminal based on the IP address. If it does, it checks the terminal specific ID. If it matches the ID that is recorded for the IP address, the message is delivered. If it does not, the message is rejected with an error message sent to the sender (in an attempt to stop further transmissions to this IP address from the source). If the terminal specific ID is blank (as would be the case with a packet that came from outside the network), the packet would be delivered regardless of the terminal ID.

The advantage of the first proposal is that the header for the packets need not be altered from the standard TCP/IP configuration at the cost of the extra handshaking that is required prior to start of communication. The advantage to the second proposal is that each of the packets is verified prior to sending to the terminal.

4.2 ATM

ATM presents a more difficult problem. The main advantages of ATM are:

- Provides a connection oriented service by setting up virtual paths and virtual circuits,
- Provides guarantees on the quality of service, and
- Packets are guaranteed to arrive in order.

For some classes of ATM, there is no provision for re-ordering of the packets. ATM is intended for a network that is both static and has very low data error rates. Neither of these properties is true for a dynamic wireless network. There are a number of proposed protocols to offer ATM service in standard wireless network [18-20](only the terminals are allowed to move). All of the protocols offer the same general solution, namely to set

up a pipe between the original BSs for the terminals and then to forward the cells as necessary in order to reach the current destination of the terminals. While this works, it is rather cumbersome . Attempting to extend this approach to a network that may have the intermediate points moving in addition to the end points is impractical because it would require the fundamental routing information for the network to be reconfigured each time a BS moved (namely the VCs and VPs). Therefore another approach is necessary.

Since the addresses schema for ATM is not practical in a dynamic wireless network, it will be necessary to translate between the ATM addressing (VPI and VCI) and what is required for the dynamic network (network ID). This would be accomplished at the gateways to the network. These gateways (the SCs) would have to maintain tables that provide a conversion between the ATM addresses and the dynamic network addresses. Essentially when a connection is attempted to be made from the outside world to a node in the dynamic network, an ATM connection would be made to the SC. The SC would then translate the ATM address to a network address and forward the packet accordingly. The connection within the network would remain connectionless. The SC would be free to assemble multiple ATM cells into a large packet for more efficient routing within the network.

A packet headed from the dynamic network to the ATM network would be disassembled at the SC and converted to ATM cells. The SC would be responsible for setting up the connection through the ATM network.

The dynamic wireless protocol assumes that the re-ordering of the packets is accomplished by higher level layers of the protocol stack and has therefore been ignored. Since the ATM cells carry no ordering information, the SC will have to provide ordering

information. This is possible because the cells will arrive at the SC from the ATM network in order, it is possible for the SC to provide a sequence number to the packets that it generates from the ATM cells.

The other important property of ATM is the quality of service. The dynamic wireless protocol makes no provisions for providing quality of service guarantees. All packets are routed with a "best effort" quality of service. To be able to connect to and communicate with ATM networks does not require that this be changed, however, it is desirable to provide service that at least approximates that of ATM in the area of quality of service guarantees. One of the possible means for achieving this goal is to introduce packet priorities into the network protocol. Packets coming from (or going to) an ATM connection that require a guarantee on delivery time and data throughput rate could be assigned higher priority. The higher priority packets would be served first. To prevent starvation of the lower priority packets, only a certain percentage of the bandwidth would be allowed to be dedicated to packets with priority. Once that quota has been consumed all remaining packets that arrive with priority would be treated as having no priority.

To provide guarantees of the quality of service would require that reservations for bandwidth be made at each of the nodes in the network involved in the "connection". While this is possible, it may not be very practical due to the dynamic nature of the network. If reservations are made, every time a change in the path occurs (due to a change of the network topology) a new reservation would be required. This may turn out to be impractical, depending on the rate of change of the network topology. The notion of a reservation, therefore should probably be limited to only those connections that require real time data flows, where delays in the network cannot be tolerated.

CHAPTER 5

LOAD BALANCING

In this chapter it will be shown that it is possible to provide a degree a of load balancing between the base stations. This is accomplished by providing a set of algorithms that direct the motion of the base stations based on the loading of each of those base stations. Both a distributed and centralized set of algorithms will be presented.

5.1 Background and Rationale

Standard wireless networks (connection oriented) enforce restrictions on the number of TMs that may attach to any of the BSs. These restrictions are necessary because each of the TMs requires dedicated resources, channel(s), from the serving BS. When all of the channels available to the BSs are exhausted, no more TMs may attach to the BS. In the proposed packet (connectionless) network, this restriction need not necessarily apply since dedicated channels are normally not required[1]. By removing the restrictions on the number of TMs that a BS can support, a newly arriving TM will always be allowed to connect to the BS. However, removing the restrictions on the number of TMs that a BS can "support" has its own difficulties. Although the BS is allowed to "serve" an unlimited number of TMs, the data rate that the BS can handle does not increase. Therefore, as the

[1] In general, a connectionless network provides a set of shared resources instead of dedicated resources. A mechanism must then be provided to allow for sharing of the resources, such as CSMA/CD or token passing[13]. The TMs therefore compete for access to the network and the amount of bandwidth that each receives depends on the total demand.

number of TMs attempting to access the network through one BS increases, the average data rate that each TM is supplied is reduced. If a contention based access method is used, the data rate available to each of the TMs will be further reduced due to packet collisions and retransmissions[12]. This may lead to unacceptable performance for certain applications, such as voice traffic. For networks that intend to handle such applications, restrictions in the number of TMs that can communicate with a BS would have to be enforced.

Intuitively, maximum throughput for the network will occur when the load of the network is uniform since all of the resources of the network are being used to their fullest extent. This requires that the individual BSs all handle the same amount of traffic from the TMs and that the tree structure of the BSs be balanced. The underlying protocols for the delivery of the data packets and routing information provide no mechanisms for distributing the load throughout the network. In fact, the distributed nature of the protocols makes true load balancing a difficult task. It is possible within the protocols to limit the number of TMs that attach to a BS (one version of the protocols in fact did this), either by limiting the number directly or by limiting the data rate demand on the BS. However, this produces the problem of denial of service to a TM that arrives late to a fully loaded BS, as noted above.

Although there is no known previous attempt to provide load balancing in an application such as the current one, there has been extensive work in the area of attempting to balance the load on multiple processors in a distributed computing environment [13-16]. Typical approaches consist of sender initiated and receiver initiated algorithms that generally use a threshold(s) to determine when a node becomes a sender

or a receiver. Typical approaches also will use either a "first fit" or a "best fit" approach to finding a home for the process that is to be migrated and may be either distributed or centralized in nature.

To reduce the asymmetry that might arise in the network topology due to TM motion some algorithms are proposed for moving (physically) the BSs so as to distribute the load more evenly among them. These algorithms are essentially variations of the ones that show up in the literature, altered to be suitable for the current problem.

To start, it should be noted here that although the BSs can move completely independently of the TMs, in general the reason for their motion is to accommodate the shifting load on the network. Therefore, the motion of the BSs should be linked to the spatial distribution of the TMs. By doing this in an automated fashion, a degree of load sharing can be accomplished. The algorithms used to control the motions of the BSs are layered on top of the routing and network updating protocols and are independent of them. Note that these algorithms are only necessary in systems where the network functions autonomously (without human intervention).

Two general methods for load balancing of the BSs are proposed, a distributed and a centralized method. Both of these methods direct underutilized BSs to move in the direction of the overloaded BSs in order to distribute the load between them. Since the distributed method conforms to the distributed nature of the rest of the protocols, it will be discussed first.

5.2 Distributed Load Balancing Algorithm

The distributed method for load sharing amongst the BSs relies on the individual BS making the decision of when and where to move. This decision is based on its own current state (load level) and information that is gotten from the SC on the overloaded BSs. For the distributed method of load sharing the following is proposed:

1. At regular intervals, all *overloaded* BSs transmit a message to the SC indicating the (physical) location of the BS and the estimated load. The length of time between transmissions of this information is dependent on the rate of change of the information and is therefore application dependent.
2. The SC maintains a list of *overloaded* BSs (location and load level).
3. Periodically the SC multicasts (see section 6) the list of overloaded BSs to those BSs in the VLA that are potentially *underutilized* (based on number of TMs BS is serving, which the SC has knowledge of).
4. The *underutilized* BSs autonomously decide whether and which overloaded BS to attempt to help. This is done with the following algorithm:
    - BS chooses a random place in list to start search. The BS starts searching in a random place in order to keep all of the underutilized BSs from helping the same overloaded BS.
    - BS chooses a random minimum number of TMs that overloaded BS must have before this BS will attempt to help. Again, this is done in order to help distribute the helping BSs amongst the overloaded BSs.
    - BS searches the list of overloaded BSs for a BS that is within X distance that meets the minimum overload condition. The distance that a BS will travel to

help another BS is limited based on the premise that if a BS travels a great distance to help, by the time that it arrives, it will be too late to help. The limit on the distance that a BS is allowed to travel to help is a crucial parameter for the algorithm.

- If a BS is found that meets the above criteria, the helping BS moves towards it. If not, the minimum overload condition is reduced and the list is searched again. By setting a minimum overload condition, we can give preference to the BSs that are severely overloaded, since they are the ones that are most in need of assistance.
- If no overloaded BS is found within X distance, the BS does not attempt to help any other BS

Once the underutilized BS has found an overloaded BS to help, it will move in the direction of that overloaded BS. As it moves, it will acquire new TMs to serve and lose others. If the load on the BSs rises to the point where it is becomes fully loaded, the BS will stop moving to help the overloaded BS. The moving BS will also stop when it gets within a predefined distance (application dependent) of the BS that it decided to help.

When a new update of the overloaded BS list arrives, the moving BS will check to determine if the BS it is moving to help is still on the overloaded list. If it is, it will adjust its direction based on the new reported location of the BS and continue to move in the direction of that BS. If it is no longer on the list (it is no longer overloaded), the moving BS will choose a new overloaded BS to help based on the new list.

By having the underutilized BSs start in a random part of the list and to have a random initial overload threshold, the probability that all of the under utilized BSs will choose just one or two BSs to help is greatly diminished. This algorithm therefore provides an open loop distributed method for balancing BS load with minimal messaging cost. To help ensure that all BSs receive assistance, an aging function is applied so that the longer a BS has been overloaded, the higher its priority (their reported load is artificially raised). By properly setting the maximum search area, the definitions of overloaded and under utilized, and the aging function, it is possible to significantly improve the performance of the network, as will be shown.

5.3 Centralized Load Balancing Algorithm

The centralized load balancing algorithm requires a much more active role by the SC in the determination of the motion of the BSs. In this algorithm, it is the SC that makes the decision as to which underutilized BS will help which overloaded BS. This proposed algorithm is:

1. Periodically *every* BS transmits its load and (physical) location to the SC. The length of the period between transmissions is application dependent (based on the expected mobility of the TMs).
2. The SC maintains a list of all BSs, their load and (physical) location
3. Periodically the SC will search the list for overloaded BSs. The SC starts at a random point in the BS list (to increase fairness). For each overloaded BS that the SC finds, the SC will attempt to assign an underutilized BS to assist it. This is done on a "best fit" basis. The SC applies a weighting function to each of the underutilized BSs that

accounts for the distance between the BSs (the one in need of assistance and the underutilized BS that is a candidate for assisting) and the relative load on the two BSs. The weighting function was arrived at empirically in the simulation and had the form of:

$$\text{assistance metric} = \text{distance} - \alpha(\text{TMs attached to overloaded BS})^{\beta} + \kappa(\text{TMs attached to the underutilized BS})^{\gamma}$$

For example, in the simulations, the distance was treated as a dimensionless unit with the total size of the simulation area 3000 units by 6000 units. The number of TMs that were attached to a BS at any time was normally in the range of 5-40 and the target number was 15 or less. For this set up α and κ were both 1.0, β was 2 and γ was 3. The goal of the function is to attempt to find a local BS to assist the overloaded BS. However, as the overloading becomes more severe, the allowable distance that a BS can travel to provide assistance is increased. The exact form that any particular design would use is highly dependent on the design parameters

4. The BS that produces the lowest metric is then chosen to assist the overloaded BS. That BS is then marked as being committed to assisting a particular BS and is sent a message from the SC to move towards the overloaded BS it has been assigned to help.

There is no attempt in this algorithm to optimize the overall solution regarding which BSs are sent to help the overloaded BS. Instead, the selection policy determines whether to move a BS based on a pair wise comparison (characteristics of the potential sender and receiver are considered independent of all of the other nodes in the system).

In this respect, the proposed algorithm is similar in many ways to algorithms used for balancing loads in a distributed computing environment[13]. In the distributed computing environment, both distributed [14,15] and centralized [16] algorithms are used. In all of the algorithms that were found (centralized, distributed, sender initiated, receiver initiated, hybrids), the selection policy is always straight forward and determines whether to transfer a task based on a pair wise comparison rather than attempting to provide an optimal global solution.

To provide an "optimal global" solution to the problem would require an algorithm of significant complexity. A truly optimized solution would have to anticipate the effects of moving the BSs including the reduction in system capacity in the area where the underutilized BS was. Such an algorithm is considered to be beyond the scope of this work. Instead, for each of the BSs that are overloaded, the SC finds the best BS to help from among the list of under utilized BSs that have not already been committed. This method will therefore find the optimal solution for the first overloaded BS that the SC processes, and increasingly sub optimal solutions for the remaining BSs.

In order to increase the fairness of the algorithm, when the routine that assigns underutilized BSs is run, the SC starts at a random point in the list of BSs that it serves to look for overloaded BSs. Therefore, each overloaded BS has an equally likely possibility of being the first BS to be assigned a helper (and therefore receiving the optimal solution).

5.4 Coverage Versus Capacity

The load balancing algorithms proposed above address the issue of how to automatically (without human intervention) move the capacity of the network around within the service area to balance the load on the network. These algorithms address the issue of providing capacity in the areas where it is needed. Providing capacity in the proper areas, although necessary, is insufficient for a satisfactory network design. It is also necessary to provide coverage to the entire service area. The algorithms above provide a means to move excess capacity around to meet the changing needs of the customers (the TMs), but they do not address the issue of whether there is coverage to all areas in the network. If these algorithms were applied to all of the BSs in the service area, it is likely that areas of the network where the load is light would lose coverage. This is because the BSs in those areas would be underutilized and therefore candidates to be moved to another area to relieve congestion. However, if the only BS serving an area is moved, that area will no longer have service. Clearly, this is not an acceptable solution.

There are only two solutions to the above problem. One is to split the BSs into two groups, a group that remains stationary and whose purpose is to provide coverage to the service area (but not much capacity) and another group whose role is to move around in the service area and provide the capacity of the network in the areas where it is most needed. The second solution is to provide a global positioning solution in which the algorithm assures not only that the capacity of the network is provided in the areas where it is needed, but also that all areas of the service area receive at least some minimal coverage. This solution would require knowledge of the terrain as well as the location of all of the BSs in order to attempt to predict the coverage area of the network. While it is

possible to provide such a solution, the computational requirements are significant and the coverage area of the network would never be more than just an educated guess due to the uncertainty of the prediction tools. So, while it is at least theoretically possible to provide an algorithm that provides for both coverage and capacity needs, the solution does not appear to be practical.

Separating the BSs into two groups, coverage and capacity, is therefore chosen as the more practical solution to the problem. As noted above, the capacity BSs would be stationary (at least as long as the boundaries of the service area were stationary) and their purpose is to ensure that all areas of the VLA have at least minimal support. These "fixed" BSs may have very little capacity, relying on the capacity BSs to help when more than just a few TMs are in their coverage area. The purpose of the capacity BSs is to provide system capacity in the areas where it is needed. These are the BSs that will be moved to support the shifting load on the network. By providing a set of "coverage" BSs and "capacity" BSs, the complexity of the BS algorithms is greatly reduced. Note that the "restriction" of having some of the BSs stationary is not a requirement for the routing and updating protocol, but rather is a consequence of the algorithms that automatically move the capacity of the network to meet the demand on the network. In fact, if the capacity of the network were moved by human intervention, the restriction could be removed.

CHAPTER 6

MULTICASTING SUPPORT

In this chapter, support for multicasting is presented. Multicasting is a one to many type of communication. One of the goals for a multicasting design is to minimize the traffic that is generated in the network to support multicasting while ensuring that all of the addressees receive all of the data that is being transmitted. Support for multicasting is easily provided by the routing protocols described above. The hierarchical nature of the network and the built in message passing algorithms can be exploited to provide a near optimal message flow for the multicasting case. The use of trees or groups as a means for facilitating the dissemination of the multicast packets has been proposed previously [21,22]. However in those cases, the groups or trees had to be artificially generated. With this set of protocols, the tree structure needed for the efficient routing of the multicast messages is inherent in the design of the network.

6.1 Reliable Multicasting

Reliable multicasting requires that the messages sent to the consumers in the network be explicitly acknowledged. Because of the one to many relationship, the acknowledgment of all of the packets from all of the consumers can lead to an "acknowledgment implosion" problem which will lead to network congestion. Care must be taken to ensure that this problem does not arise. The hierarchical nature of the network proposed allows

for the distribution of multicast messages in a very efficient manner. Reliable multicasting is achieved in the hierarchical dynamic wireless network by the following means:

Group Setup:

- A new multicasting group is started by the master of the group sending a broadcast message to the network (or by other means of advertising)
- To join the group, a consumer sends a join message to the master of the group
- The master adds the consumer to the list of addressees for the group
- When one of the addressees requests permission to multicast data (request sent to the master), the master passes the "token" to the requester (on FCFS basis). The token message contains list of all addressees. This is a point to point message.
- The requester then becomes the producer. It begins sending messages to the multicast group. Each message header contains the address of each member of the group[2]. The producer is specially marked. Only one copy of each message is sent.

Message Delivery:

At each BS, the following tasks are performed:

- Send a copy of the message to any addressee served by the BS

[2] For large multicast groups this approach may be impractical. An alternative approach is to broadcast the IDs of the multicast group to each BS when changes to the group occur and assign the group a special ID. When the BS receives a packet with a multicast ID, it checks to determine whether any of the groups members are in its subtree. The remainder of the routing is the same as the main case.

- Send a copy of the message to each subtree that contains at least one addressee (at most one message per subtree)
- Maintains a record of which TMs and subtrees that were sent a copy (required for the ACK portion of the protocol).
- Remove from the address list, those addresses that it has sent a message to.
- If the message has not been to the SC ring yet, sends a copy of the message to its parent with the (possibly) shortened list of addressees

  At each SC the following is performed:
- Sends a copy of the message to each subtree that contains at least one addressee (at most one message per subtree)
- Maintains a record of which TMs and subtrees were sent a copy (required for the ACK portion of the protocol).
- Removes from the address list, those addresses that it has sent a message to.
- Sends the message with the (possibly) reduced list of addressees to the next SC in the ring When the message either reaches a leaf BS or has no addresses left in it, it is deleted.

ACK Messages:

At each BS the following tasks are performed on the reverse (ACK/NACK) direction:

- Buffers the ACK/NACK until a response is received for each copy of the message it sent (either to a TM or to a subtree)

- Builds a single message containing all of the ACKs and NACKs received. The message contains the original message sequence number and a list of TMs that replied with ACK and TMs that replied with NACK.
- Upon receipt of all of the ACKs/NACKs or the expiration of a timer, the BS sends the ACK/NACK message to its parent

At each SC the following is performed

- Waits for a multicast ACK from each subtree to which the message was sent
- When all multicast ACK messages have arrived or the timer expires, it concatenates the messages and sends a single multicast ACK to the producer.

The timers at each SC and BS limit how long the node will wait for ACKs/NACKs. If a timer expires, the node will include the missing message as a NACK and forward the completed message. If the BS subsequently receives the missing ACK/NACK, it discards it.

The number of messages that will be sent (under static conditions) to the producer will be equal to the number of SCs in the system (a relatively small number). At the producer, as the ACK messages arrive, it removes from the retransmit list (which starts with all consumers on it) all consumers that replied with an ACK. There is a timer for each message, at the expiration of the timer (or when all ACKs/NACKs are received), the producer will retransmit the message to all consumers that remain on the retransmission list.

Special treatment must be given to the case where a BS or TM moves during this process. When a BS moves, it will remove from the waiting list any subtree message that

it was waiting for (since it will not receive them). It will not send an ACK/NACK for any subtree message that was removed. The ACK/NACK will be sent directly to the producer by the root of that subtree. In the case of a TM moving between the time it receives a multicast message and when it can respond, it will send an ACK/NACK to its new parent. If possible, the new parent will incorporate this unexpected ACK/NACK in its own message. If not, it will simply forward the message on towards the producer. In general, if a node receives an ACK/NACK that was not expected, it will incorporate it in the message it is building or, if that is not possible, forward the message to the producer.

6.2 "Semi-Reliable Multicasting"

As can be seen from the above discussion of the reliable multicasting feature, the cost of acknowledging each packet from the consumers back to the producer is significant. As the mobility of the TMs increases, the number of acknowledgments that will not be able to be incorporated into other acknowledgment messages (and will have to be sent back to the producer by themselves) will increase, thus increasing the load on the network. In many instances, the overhead of a fully reliable multicasting protocol may not be necessary. For example, it is not necessary (or even desirable) to acknowledge all packets for an application that is sending "real" time video or audio. It is better to drop packets than to have then arrive very late (as would be the case for a packet that was retransmitted).

For the family of applications that do not require an absolute guarantee of packet arrival at all receivers, it is possible to reduce the overhead of the multicasting protocol

by eliminating the ACKs/NACKs. This version of the multicasting protocol has been dubbed the "semi-reliable version". It uses the same mechanisms for the delivery of the multicast messages as the fully reliable version. The protocols differ only in the manner in which they ensure delivery. Whereas the fully reliable version required that the consumers acknowledge the message directly to the producer, the semi-reliable version only acknowledges point to point delivery. As soon as a node receives the message properly, it acknowledges the message to the node that it received the message from. It does not wait to ensure that the nodes it will send the message to have received the message. Each node is responsible for buffering the message until an acknowledgment is received from each node that it directly sent the message to. When an acknowledgment is received from each of those nodes, the packet is removed from the buffer.

The semi-reliable protocol therefore relies on the robustness of the underlying routing protocol to ensure that the message is delivered to the proper locations. To guarantee delivery of the multicasting messages then, the underlying routing protocols have to guarantee the accuracy of the routing information, even when the BSs and TMs are changing locations in the network. The simulation results will prove this inherent reliability in the underlying protocols.

6.3 Evaluation of the Reliable Multicasting Protocol

A set of criteria have been proposed for the evaluation of multicasting protocols by the MIST project[3]. The criteria suggested provide a means for evaluating proposed multicasting protocols. Below is an assessment of the proposed multicasting protocol for each of the criteria suggested by MIST.

6.3.1 Reliability

As will be shown in the simulation results section, even under conditions of very high mobility, the basic protocols reliably deliver all messages to all destinations. Previous work in the area of multicasting in a wireless network [8,9] have had problems either with ensuring accurate routing information, or with the cost of keeping the information updated. Using these protocols as the basis of the multicasting protocol, it is possible to show that the routing information will always be accurate with low cost (there is no additional cost for the multicasting routing information).

Proper distribution of the multicast message in the case where the network is stable is trivial. When one or more of the destination TMs is in transit, the protocol still provides delivery to all destinations, even without the ACK/NACK portion of the protocol. As proof, consider the case illustrated in figure 6.1 where mobile M1 wishes to go from BS3 to BS6. M1 will send a control message directly to BS6 requesting service.

---

[3] The MIST (Multicast Implementation Study) project is funded by the Defense Advanced Research Projects Agency (DARPA). The purpose of MIST is to investigate issues relating to multicasting with particular emphasis on reliable multicasting. More information is available at http://www.tascnets.com/mist/doc/mist.html.

If M6 can provide service, it will send a control message back through the network to update the location of M1. The new path to M1 is created prior to the teardown of the old path (a natural consequence of the message flowing from the new BS to the old BS)

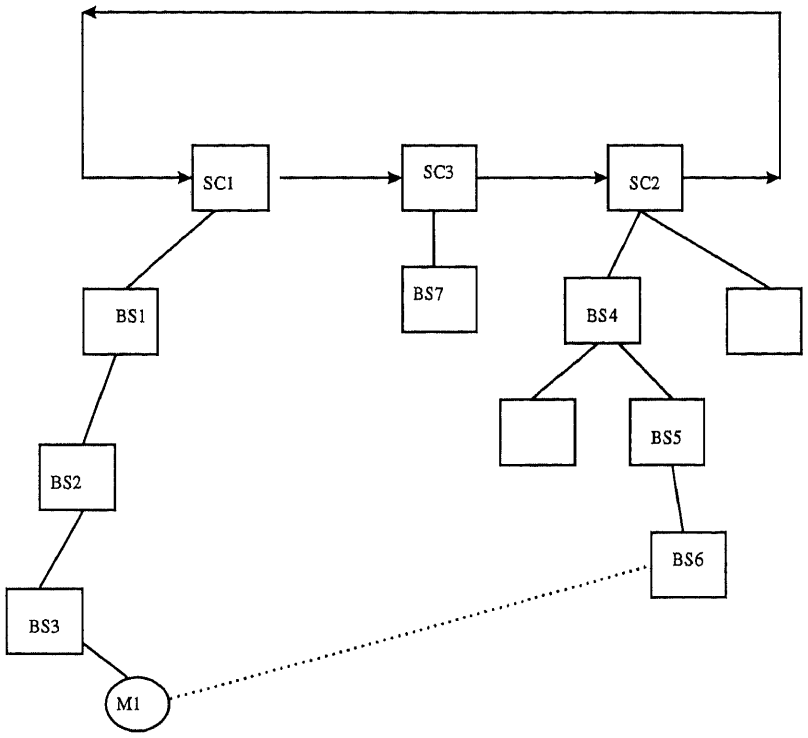

Figure 6.1 Multicasting TM in the process of relocating.

If a multicast message at a BS involved in the update while the update is in progress, one of two things can happen, either it arrives at the node before or after the control message. If the multicast message arrives before the control message at one of the nodes in the old path, it will be routed to the old location of M1. Provided that the

multicast messages are given the same priority as the control messages, once the multicast message is in front of the control message it will remain in front. Therefore the multicast message will follow the old path to M1 down to BS3 in front of the control message that tears down this path. Once at BS3, the multicast message is transmitted to M1 since M1 will communicate with the old BS until the control message arrives at the old BS.

If the control message arrives at the node in the old path prior to the multicast message, the old path will have been torn down (at least at this node) and the multicast message will not go through the old path. However, since the new path already exists, when the message gets to the head of the new path, it will be routed there.

For example, at node SC1, if the multicast message arrives first, it will be routed through BS1, BS2,and BS3 to get to M1. If the control message arrives first, SC1 will not pass the multicast message down into the subtree but will merely forward to SC3 which will forward it to SC2. At SC2, the multicast message will be routed down through BS4, BS5 and BS6. This path exists at this time because it was created prior to the old path being torn down. At BS6, it is possible that the message may have to be held because M1 has not arrived yet.

Now suppose that the multicast message originated at BS7. It will be forwarded to SC3. From SC3 it will go to SC2. If the control message gets to SC2 prior to the multicast message, there will be a path to M1 through BS4, BS5, and BS6 that is constructed and waiting for traffic. So the multicast message can be forwarded through this path. If the multicast message gets there first, there will be no path through SC2 and

the message will be forwarded on the ring to SC1. At SC1, the old path still exists, so the message is forwarded along this path instead.

This logic extends to the case where the source and destination are in the same VLA and the destination is moving within the VLA. In this case, the root of the tree that contains both the source and destination acts in a manner similar to SC1 in the prior example.

6.3.2 Heterogeneous Networks

As applied to this discussion, heterogeneous networks will contain receivers (as well as the intermediate base stations) that have different capabilities (buffer sizes, processing speeds, etc.) Also, fully reliable delivery may not be required for all addressees.

The first of these heterogeneous properties problem is solved by incorporating flow control in the protocol (see sect 6.2.4). The second property is best handled by instituting two levels of reliability: full reliability (defined above) and a best effort reliability, where ACKs are not used. The protocol could be expanded to allow the marking of TMs as requiring full reliability, or best effort. Providing the two tier service can potentially reduce the number of ACKs required (at the expense of complexity in the message delivery and some loss of reliability).

6.3.3 Scalability

This protocol is highly scaleable due to the hierarchical nature of the routing. Message replication occurs only when required and messages go only to nodes required for direct

delivery (no flooding is ever done). In the reverse direction, the ACKs and NACKs are all collected prior to forwarding, thus minimizing the traffic in the reverse direction (no ACK implosion problem).

6.3.4 Flow Control

The message transmission rate is controlled in two manners. First, the sender may only send a limited number of unacknowledged messages. Once it reaches this limit, it must wait until one of the earlier messages is fully acknowledged prior to sending another message. Second, a mechanism is proposed by which the rate of transmission is tied inversely to the rate of NACKs received. As the rate of NACKs increases, the allowable transmission rate is reduced (a higher rate of NACKS is likely a result of congestion in the network).

6.3.5 Late Arrival/Early Departure Problem

TMs requiring inclusion in a multicasting group send a join message to the master of the group. The master adds the address of the joining TM to the list of group members. If the transmitting token is currently lent out (a mobile is in the process of transmitting), the master will send an update message to the transmitting mobile, indicating that a new group member has arrived.

Early departure is handled in the same manner as a late arriving member. In the special case where the departing member is doing so because either it, or the link to it, has gone down, its parent will detect the loss of the TM. When a multicast message is

received for that TM, the parent will respond to the producer that the TM is not available. The producer will remove the TM from the address list and send an update to the master so the that the master list is also updated. If the TM comes back on line, it sends a join request to the master, if it desires to rejoin the group.

6.3.6 Fragmentation and Reassembly

There is no provision for fragmentation and reassembly. This must be handled by a higher layer.

6.3.7 Ordering

Each of the messages has a sequence number, which is necessary to allow for proper acknowledgment of the messages. These sequence numbers will also be used to reorder the messages, should they arrive at the mobile out of order. No special treatment is given to the messages while traversing the network to ensure that they remain ordered. However, as will be shown in the simulation results section, even in highly mobile networks only a few percent of the messages are received out of order.

CHAPTER 7

HETEROGENEOUS NETWORKS WITH LIMITED RESOURCES

As in all networks, the resources are limited. In the case of a wireless network, the bandwidth of the radio link is typically the limiting resource. While it is always possible to increase the computing power and the maximum length of the buffers in network nodes, typically the bandwidth of the over the air link is constrained and cannot be increased. For this reason it is important to allocate the bandwidth in such a way as to maximize its usefulness (i.e. maximize the throughput through the network). In this section, the problems associated with limited resources for a network will be considered and a method will be offered for allocating those limited resources to maximize the throughput of the network. Much of the current research into packet radio networks ignores the constraints on the system due to limited system resources (and in particular, limited bandwidth on the air interface).

In the next sections a method will be presented for apportioning the available bandwidth to maximize the throughput of the network. Following the apportionment of the bandwidth, we will present a method for determining the number of buffers required to support the wireless links.

7.1 Determining the Data Rates on the Network Links

As mentioned above, the bandwidth on the over the air link is generally the bottleneck in the design of a wireless network. Proper allocation of the available bandwidth is therefore crucial to the design of the network. The following is a method for allocating the bandwidth of the network. Since the details of the physical link (including the encoding scheme) are outside the scope of this work, the discussion will center on the available data rate of the wireless link rather than the bandwidth. It is noted that there is a unique relationship between the bandwidth and the data rate, once the encoding scheme is chosen.

Let λ denote the mean traffic load per user. Then the total load on a base station with n terminals is $n*\lambda$. If there are m BSs that exist within a tree in the VLA then the load at the root of the tree will be $(m*n-n*m/(n*m*p*q)*\lambda$. The $n*m/(n*m*p*q)$ term represents that fraction of the traffic that stays within the tree. Similarly, if there are p trees in a VLA, then the load on the switching center will be

$$[p*m*n-(p*m*n)/(n*m*p*q)]* \lambda + \Sigma_{i=1 \text{ to } q} [(q-i)/q]*p*m*n = \lambda_{SC} \quad (1)$$

The second term represents the traffic contribution coming from the other VLAs. In order to balance the load through the system, it is desirable to have each of the links have the same delay. The delay for any of the links, assuming that the load and processing time are both Poisson processes is

$$\text{Delay} = N/\lambda = \rho/(\lambda-\rho)/\lambda = 1/(\mu - \lambda) \quad (2)$$

The mean service time is the packet size divided by the data rate of the outgoing line (processing time is assumed negligible). The mean service time is therefore

$$\mu = DR/PS$$

It is the term DR (data rate) that we wish to vary in the network in order to make the delay at each of the nodes the same. To find a relationship for the data rates, we plug the relationship for the service rate back into the delay equation (2) to get:

$$\text{Delay} = 1/(DR/PS - \lambda)$$

Since the delay should be the same for all of the links, we make the following relation:

$$1/(DR1/PS - \lambda) = 1/(DR2/PS - n*\lambda) = 1/(DR3/PS - (m*n-n/(n*m*p*q))*\lambda) = 1/(DR4/PS - \lambda_{SC}) \quad (3)$$

The above relationship provides a manner for determining the relative data rates that should be used in the network. However, this is not enough to determine absolute values. There are some constraints that need to be applied to the problem, (i) the amount of bandwidth (and hence the maximum data rate) is fixed and (ii)there is most likely a

maximum amount of delay that is allowed in the network (stated as an end-to-end figure or for each link).

Applying constraint (i) to the problem yields the relation:

$$\text{Max data rate} = p*m*n*DR1 + p*m*DR2 + p*DR3 + DR4 \quad (4)$$

Applying constraint (ii) requires that equation (3) be modified to:

$$1/(DR1/PS - \lambda) = 1/(DR2/PS - n*\lambda) = 1/(DR3/PS - m*n*\lambda) = 1/(DR4/PS - p*m*n*\lambda) = \text{Max Delay (per hop)}$$

This can be simplified to:

$$\text{Max Delay} = 1/(DR1/PS - \lambda) \quad (5)$$

Rearranging:

$$DR1 = (1/ (\text{Max Delay}* PS)) - \lambda$$

This problem, as described, is over constrained. We cannot simultaneously satisfy all of the constraints (number of TMs, max data rate, and max delay). Typically the allowable delay, the max data rate, and the load per user would be inputs while the

maximum number of users that the network can support is the output of the design problem.

7.2 Provisioning the Buffers in the Network Nodes

Once the data rates have been provisioned for the network nodes, it is possible to determine the buffers that are required to support the expected traffic. The inputs to this part of the provisioning problem are the expected load, the service rate (i.e. the data rate as determined above) and the allowable dropped packet rate.

The purpose of the buffers is to provide overflow space for the packets. The buffers should be large enough to ensure that the node does not drop messages due to buffer overflow (except in the extreme case). Since the average delay through each of the nodes is known (a product of the design), it is possible to determine the size of the buffers that are needed to accomplish this goal.

Basic queuing theory[17] provides the equations to relate the buffer size to the dropped packet rate. Assuming that the packets are segregated based on the link that they are designed for (uplink or downlink), it is possible to analyze the system as an M/M/1 system where a set of buffers is dedicated to a particular wireless link. For an M/M/1 system, the probability that the buffer will be full for a buffer of N spaces with a arrival rate of λ and a service rate of μ is

$$P[n] = \rho^n * (1-\rho)/[1-\rho^{(n+1)}] \quad (6)$$

where

P[n] = probability that the buffer is full

n = number of packets that the buffer can contain

ρ = arrival rate / service time = λ / μ

The drop rate is essentially the probability that the buffer will be full (since if the buffer is full the next packet to arrive will be dropped). If the acceptable drop rate is known (an assumed input to the problem), then there is sufficient information to solve for N, the number of buffers that are required.

CHAPTER 8

SIMULATION RESULTS

In this chapter, the results of a simulation of the network routing protocols described above will be presented. It will be shown that the protocols can successfully deliver messages even when the mobility of the TMs and BSs within the network is very high. Successful delivery of both point to point and multicast messages will be demonstrated. In addition, the performance of the load balancing algorithms will be evaluated against each other as well as the case where there is no attempt to balance the load on the network nodes.

8.1 Description of the Simulation

The simulator was developed as a test bed for the protocols described above. It provided a means to test the correctness of the protocols to properly deliver data packets and maintain the routing information. In addition, the simulator provided the means for evaluating the performance of the protocols as a function of the network parameters. It was developed using the C++ language and runs on a PC. Appendix 1 provides more information on the simulator.

In order for the simulator to perform its required tasks, it was neither necessary nor feasible to simulate all aspects of the protocols. However, all of the major aspects of a network running these protocols were simulated. In the simulator, each node had its own

routing information, queues for data and control messages, and could move independently of the other nodes. The messages were processed by the nodes and sent to the next node according to the local routing information. Each of the nodes was assigned a fixed amount of resources (bandwidth and buffers). When the demand exceeded the resources, messages were dropped.

Since a distributed computing environment was being simulated on a single processor, it was necessary to break up continuous time into discrete steps. A time step is of arbitrary length but small enough to provide a reasonable simulation of continuous time. During each time step, each node was visited and processed sequentially. All actions that occurred during one time step were considered to have occurred simultaneously. This was enforced by not letting the actions of a node processed early in the sequence have an effect on a node processed later in the time step.

The SCs are implemented in the simulation as if they were a set of low orbit satellites. They have a fixed orientation with respect to each other, but they serve a VLA for only a limited amount of time [11]. When the SCs change to a new VLA, they transfer their database to the SC behind them.

Traffic in the network was generated by having each of the TMs create messages with of a random length (up to the maximum allowed length of a message). Every TM had the same probability of generating a message during each time step. The messages were then broken down into packets for transmission. The destination for each message was a randomly selected TM in the network. The size of the network used in the simulations was variable. During the routing verification portion of the simulations, a

network with 4 VLAs, each initially containing 7 BSs and 28 TMs was used. This network topology was used because it was small enough to allow time for hundreds runs of a sufficiently long time scale to ensure that the network performance had reached equilibrium while still providing for all of the major components of a real network based on these protocols. In particular, 4 VLAs were used to ensure that the protocols were capable of handling the movement of TMs and BSs between VLAs.

After proving the correctness of the basic routing and updating protocols larger networks were used to prove in the load balancing algorithms (2 VLAs, 32 BSs and 320 TMs) and a still larger network to evaluate the performance of the network versus the network topology (2 VLAs, 64 BSs and 640 TMs). While the number of TMs that are connected to a BS at the start of a simulation run was on the low side, the total number of TMs was sufficient to cause severe local overloading of the network and thus provide a reasonable platform for studying the load balancing algorithms (peaks of 80-90 TMs on one BS were observed in the uncontrolled cases).

BSs and TMs were free to move anywhere within the service area, including crossing from one VLA into another. When a BS or TM hit a service area boundary, the direction of the mobile, or BS was reversed. Details of the simulation are contained in Appendix 1.

8.2 Results and Interpretations

In this section the results of simulations performed will be presented and the performance of the protocols will be assessed. The percentage of the offered load that was dropped is used as the primary measure of system performance. To provide a base to reference against, the resources required to deliver all of the messages when there is no motion in the network were determined to start. This set of resources is referred to the base set. This base set is used merely as a means for measuring the effect of changing parameters in the simulation.

8.2.1 Verification of Routing Protocols

The routing of data messages and the proper updating of the distributed database is at the heart of the proposed protocols. To prove that these are properly performed, the simulation was first run with an added restriction that each BS had a maximum number of TMs that it could service (this restriction will be removed later). Restricting the network mobility of the TMs and BSs in this manner limits the amount of asymmetry that can exist in the network. While this is an artificial constraint on the system, it permits the evaluation of the routing protocols to be isolated from the evaluation of load balancing in the network. This restriction will be removed once the routing and updating protocols have been verified.

For the verification of the routing and updating protocols, a network containing four rectangular VLAs with 7 BSs and 28 TMs originally in each VLA was used. This rather small network contained all of the components that were required to ensure that the

protocols were fully exercised. To help ensure that all of the possible conditions in the network were tested, a set of long simulations were run (5000 time steps). As noted previously, in order to simulate the distributed computing environment of a network, it was necessary to break continuous time up into discrete "time steps". Within each time step, all of the network nodes were visited and processed, sequentially. To ensure temporal correctness, no action by one node was allowed to impact another node within the same time step. For example, if node X sent a message to node Y during time step 100, node Y would not be allowed to do anything with the message until time step 101. By making the time steps small, the continuous case can be approximated.

The initial step in the simulation work was to establish a set of "base" resources that are required if there is no mobility in the network. These resources are those necessary for the (nearly) balanced hierarchical tree that exists when there is no motion in the network. The resources required for this case can be estimated based on the equations presented in section 7 assuming that the average traffic generated by each of the TMs in the system is known. A typical plot of the lost packet rate as a function of one of the resources, in this case the maximum data rate between the root BS and its subtree, is shown in figure 8.1.

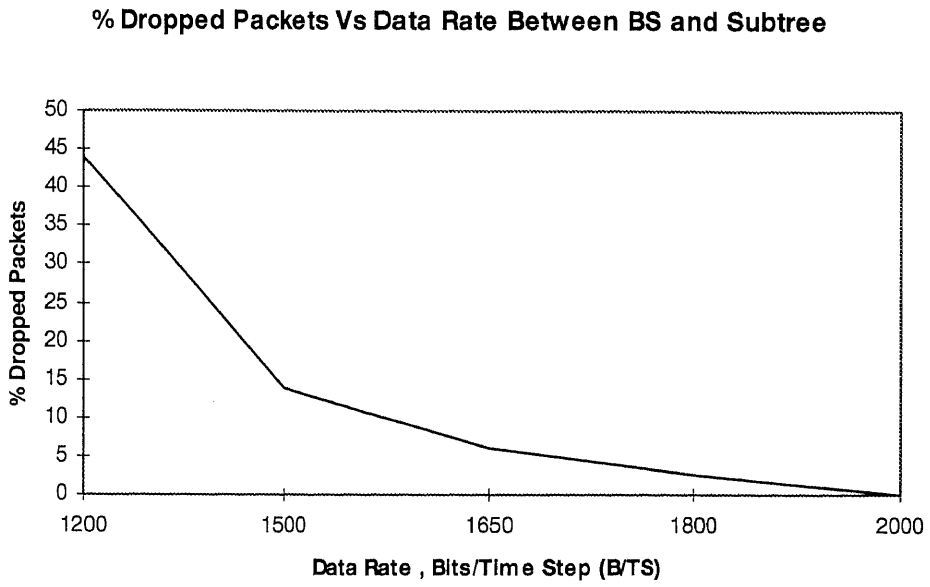

Figure 8.1: Lost Packet Rate Versus the maximum data rate between a root BS and its Subtree.

The general method for determining the basic resource requirements for a network would entail an analytical calculation based on the equations in Chapter 7 followed by a verification of each of the components. As an example, consider the results shown in figure 8.1. This is for a root BS which is required to handle the traffic from an average of 10 TMs. Assuming the same arrival and service times as were used in the simulation, the equations in Chapter 7 suggest that the root BS be allocated a data rate of 2500 Bytes/TS for communication with its subtree. A similar analysis for the number of buffers results in the allocation of approximately 100 buffers to handle the traffic. The simulation results suggest that the data rate be approximately 2000 bits/time step and the number of buffers should be approximately 125.

By varying the components one by one it is possible to find the optimal resource allocation for each of the components. Note that there is a significant amount of coupling

between the resources and therefore there are some design tradeoff decisions that generally have to be made. A good example of this is the tradeoff between the number of buffers and the maximum data rate for a node. As the number of available buffers decreases, the maximum data rate that a node needs to support without dropping packets increases since there is nowhere to store packets when the arrival rate exceeds the average arrival rate. In general then, the verification of the resources requires an iterative solution. Note that this is part of the reason for the difference between the analytical results and the simulation results. The simulation settled on a higher number of buffers and therefore required less bandwidth.

Having performed the above study to determine the base resources to ensure that only a tiny fraction (less than 0.1%)of the packets are dropped when there is no motion in the network, it is then possible to study the effects that motion in the network has on the required resources as well as to determine the correctness of the protocol for updating the distributed routing information. Figure 8.2 shows the percentage of the packets that were lost as a function of the resources allotted when the mobility of the TMs was added into the simulation. The resources are normalized to the those required for the case where the TMs and the BSs were all stationary. The units of distance and time for the simulation are arbitrary and therefore are represented as generic distance units and time steps, respectively.

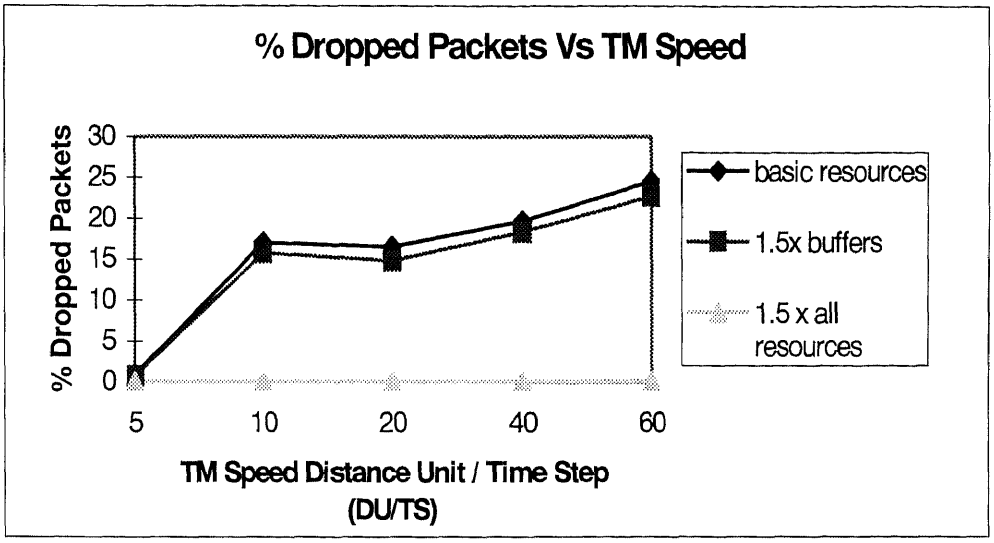

Figure 8.2: Dropped Packet Rate Versus TM Mobility, network asymmetry limited.

The plot shows that the important resource here is the data rate between the nodes. This is to be expected since if the arrival rate is greater than the service rate, the buffers will overflow no matter how large they are. However, once more resources are added in the form of increased data rates between the nodes, the lost packet rate drops to (exactly) zero.

The curve of the lost packet rate when the resources were all 1.5 times the base set also provides strong evidence that the protocol is properly updating the routing information. Incorrect routing information for even one of the TMs at any of the BSs in the network (i.e., there is no entry for the TM when there should be or the entry is

incorrect so that a packet cannot reach the TM) will result in lost packets in the network due to congestion. This fact was demonstrated on many occasions during the design of the protocol and was one of the principal methods used for finding race conditions that had existed in the early versions of the protocol. When there is one or more "lost" TMs, there was no (practical) amount of resources that would provide loss-less service. To confirm this conclusion, the network routing tables were examined at periodic points in the simulation to ensure that the tables were all consistent and that every BS and TM were properly accounted for.

It should be pointed out that the mobility rates that were used in the simulation were quite high. The general assumption was that a TM would transmit no more that 5 packets within a time step, suggesting that a time step is a fraction of second in length[4]. Note that at a TM mobility of 60 DU/TS, there were approximately 9,000-10,000 TM moves between BSs during the 5000 step simulations which indicates that each of the TMs moved 80 times during the simulation.

An attempt was made to find the mobility rate for the TMs that would lead to a break down in the protocol even if infinite resources could be applied. This might be thought of as the pole point for the system. This study was performed by taking the network described and increasing the mobility rate of the TMs until no amount of resources would cause the network to reliably deliver all of the packets. As it turns out, it was not possible to make the protocol breakdown. The rate of mobility of the TMs was

[4] Although it is assumed that the length of a time step in the simulation is a fraction of a second, the actual length of the time step is not important. When the simulation was set up it was assumed that the length of a data packet was 100 bytes. For a 28.8Kbps modem, this would translate into between 25 and 35 packets/second (depending on the overhead).

increased until the number of TM moves (from one BS to another) saturated and no breakdown in the protocol occurred. Even at the highest rates tested, the protocol successfully delivered all of the packets. Figure 8.3 shows the number of TM moves as a function of the mobility rate.

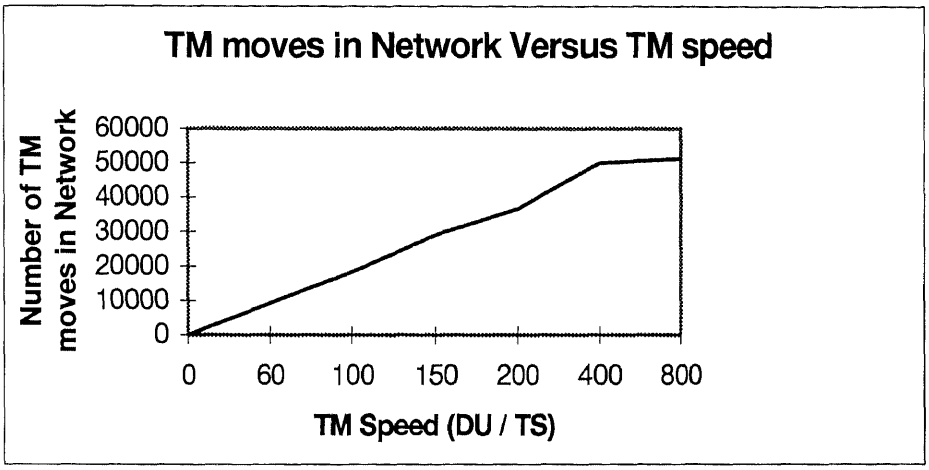

Figure 8.3: TM moves (from one BS to another) as a function of TM Mobility.

At the highest mobility rates tested, approximately ½ of the TMs were in the process of moving from one BS to another at any given time during the simulation. This is clearly a mobility rate that is much greater than any that would be experienced in a real system.

When the mobility of the BSs is added into the simulation, the results are essentially unchanged. For this set of simulation runs, the BSs moved in a semi random manner, of the same nature as the TMs. Again, the number of TMs that each BS was allowed to service was limited in order to limit the asymmetry in the network. Figure 8.4 shows the

dropped packet rate as a function of the mobility of the BSs. In these runs the mobility of the TMs was fixed at 20 DU/TS.

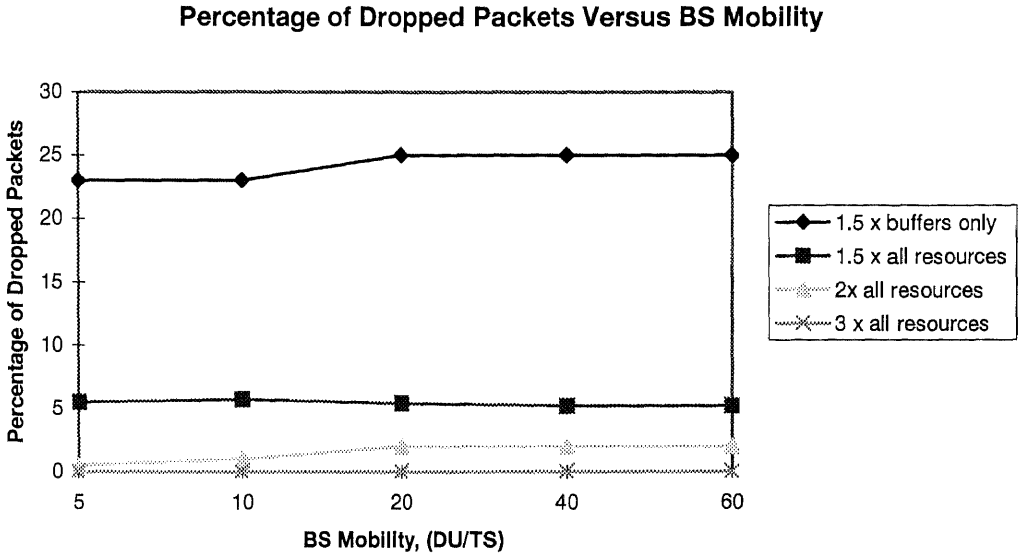

Figure 8.4: Dropped Packet Rate Versus. Mobility of the BSs, limited asymmetry in the network.

As can be seen from figure 8.4, the rate of dropped packets was due entirely to the amount of resources that were allocated. Again, this shows that the protocol is properly updating the database. Note that the curves for the case where the BSs are allowed to move are much flatter than those in which only the TMs were allowed to move. When the BSs are allowed to move, the asymmetry in the network develops much faster (i.e. at a lower mobility rate) than it does when only the TMs are allowed to move. The curves therefore represent the performance of the network as a function of the resources that are allocated to it.

The combination of the data in figures 8 and 10 form the basis for the conclusion that the underlying protocols are properly delivering messages and updating the network. Again, the rates of mobility of the network components was very high, much higher than might be expected in a real network.

8.2.2 Performance versus Network Configuration and Traffic Patterns

The previous section dealt with the issues of whether the protocols are capable of maintaining the routing information and properly routing messages as a function of the mobility of the network components. In the section, we will consider the performance as a function of the topology of the network and the traffic patterns.

To study the network topology effects on performance a network with 2 VLA, 64 BSs and 640 terminals was used. Within a VLA, the network was setup with three different topologies to start, 2 root BSs each with trees 4 levels deep, 4 root BSs each with trees 3 levels deep, and 8 root BSs each with trees 2 levels deep. The trees all started in a balanced state. Initially each BS had 10 TMs communicating with it. Two traffic distributions were used in the simulation. The first had a random distribution of traffic (the TM randomly chose a destination for a message from among the entire population of TMs). The second traffic pattern assumes that the traffic is actually likely to be much more local in nature. When a TM generates a message, with Probability P1 it will send it to a TM that is communicating with its own BS, with probability P2 that is in its own tree, and with probability P3 with a random TM. Since the TMs do not know where other TMs are, this action had to be simulated.

Figure 8.5 shows the number of hops required for performance of the three network topologies when the BS were restricted from moving. For this case, the topology of the network remained stable for the life of the simulation and was therefore useful in examining the difference in performance due to the different network topologies.

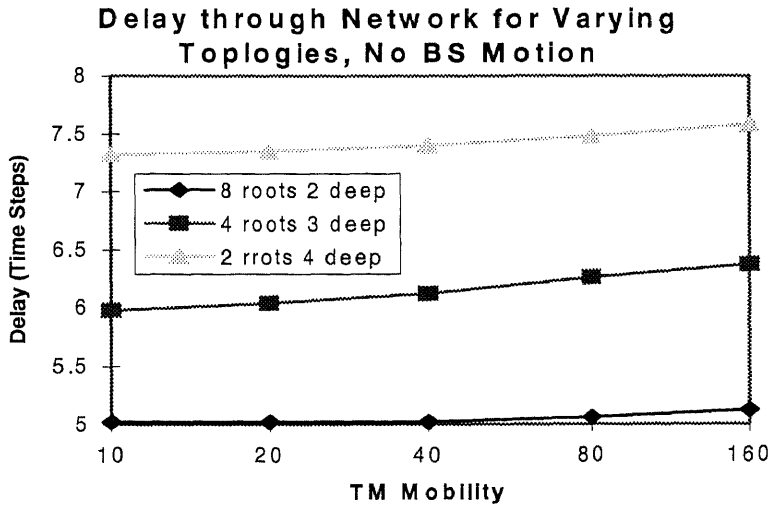

Figure 8.5: Delay through the Network as a function of network topology when the BSs are fixed.

The data rate allocated to each of the BSs was based on the layer in the network with more resources allocated to those BSs that are higher in the network. Since the 8 root case had more BSs higher in the network than the 2 root case, it had a larger total data rate available to it than the 2 root case.

When the BSs are allowed to move, the same performance metric changes somewhat. Figure 8.6 shows the delay through the network as a function of the TM mobility when the BS move with probability 0.5 and at a rate of 10 distance units per time step when they do move.

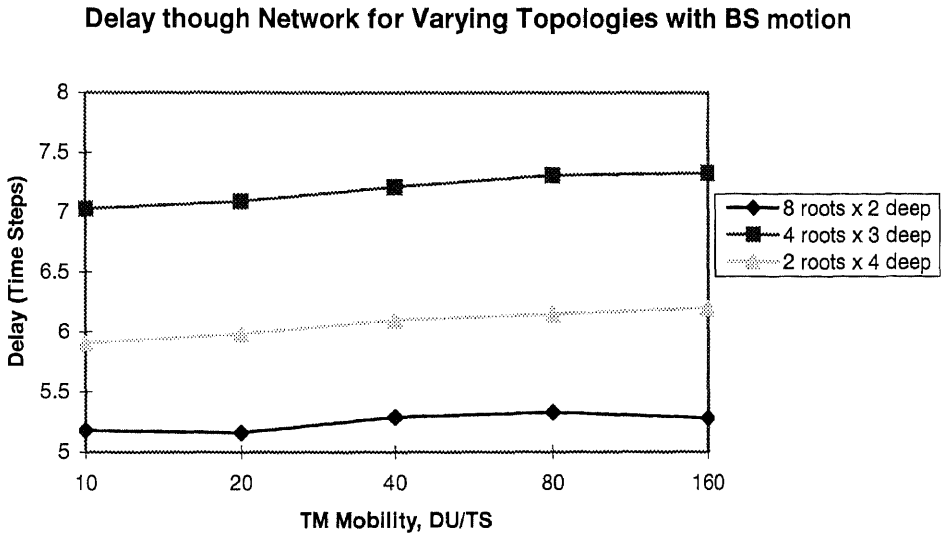

Figure 8.6: Delay through the network when the BSs are allowed to move. The motion of the BSs was set to have a probability of occurring during a time step of 0.5. When the BS does move, it moves 10 distance units.

Note that the performance of the network actually improved when the BSs were allowed to move for the cases of 4 roots and 2 roots. An examination of the data reveals that for these cases, as the BSs move, they tend to reconnect to the network at higher levels in the, thereby reducing the mean distance a message has to travel to get to the BS. The mobility of the BSs therefore allows the network to transform itself into a more efficient structure (i.e. a minimum depth tree). The fact that the absolute magnitude of the

delay increases with increasing TM motion is a reflection of the increase in control traffic.

The number of control messages that were required to keep the network routing information up to date for the three network topologies is shown in figure 8.7. As can be seen from the figure, the 2 root topology required the least number of control messages to remain up to date. In fact, the number of control messages increased with decreasing depth of the trees. However, although the number of control messages decreases, the number of total hops required for the control messages increases. As an example, when the mobility of the TMs was 160 DU/TS, the total number of hops required for the 2 root case (64,800 control messages) was 514,400 while for the 8 root case (85,800 control messages) it was only 477,000. So although the number of control messages was less, the total cost (total number of hops) was more.

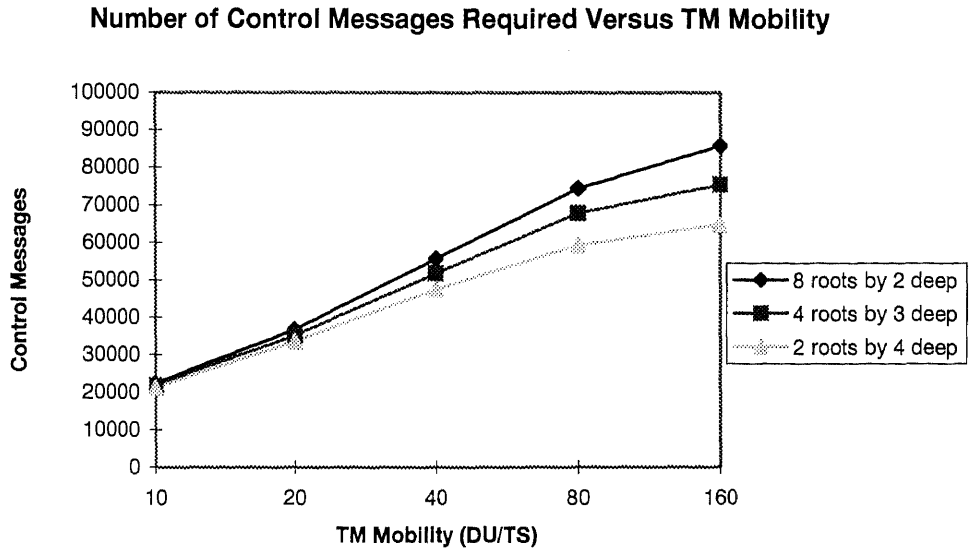

Figure 8.7: Number of Control Message required as a function of the TM Mobility. BS mobility was fixed at a Probability of 0.5 of moving 10 DU each time step.

Figure 8.8 shows the effect on the average number of hops required to deliver a message as a function of the mobility of the BSs for the 4 root by 3 deep topology. As can be seen from the figure, as the BS motion increases, the average number of hops required to deliver a message decreases. This is due to the transformation of the network towards a minimum depth tree. As the BS motion increases, the speed at which the topology approaches a minimum depth configuration increases. The decreasing number of hops required does not continue for ever, In fact, the figure indicates that are 30 DU/TS, the rate of decrease is leveling off.

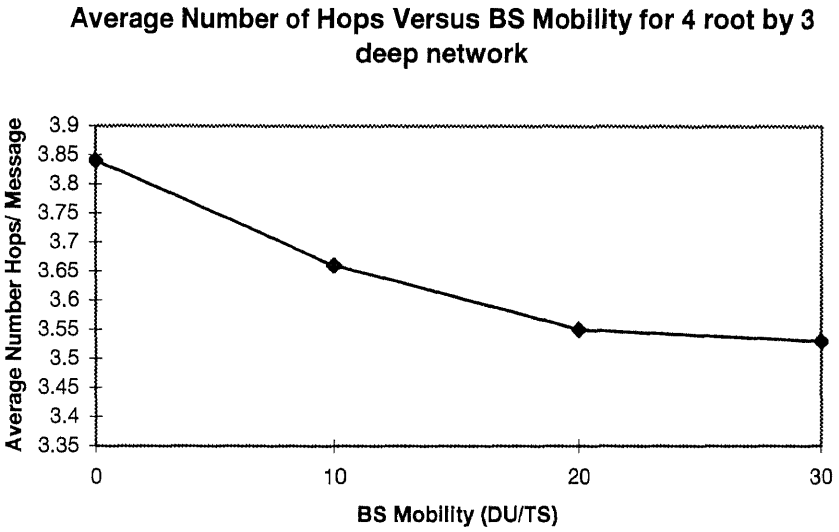

Figure 8.8: Average Number of Hops required to deliver a message as a function of the mobility of the BSs for a network with an original configuration of 4 roots and levels deep.

As noted earlier, a second traffic model was also analyzed. For the local traffic model case, it was assumed that of the traffic that the mobile generated, 60% would remain in its own tree with the other 40% randomly distribute amongst all of the TMs (including those in its own tree). Figure 8.9 shows the average delay through the network as a function of the mobility of the TMs (BS mobility was fixed at P=0.5 of 10 DU/TS motion). The delay through the network for these cases were lower than the entirely random traffic case (for obvious reasons).

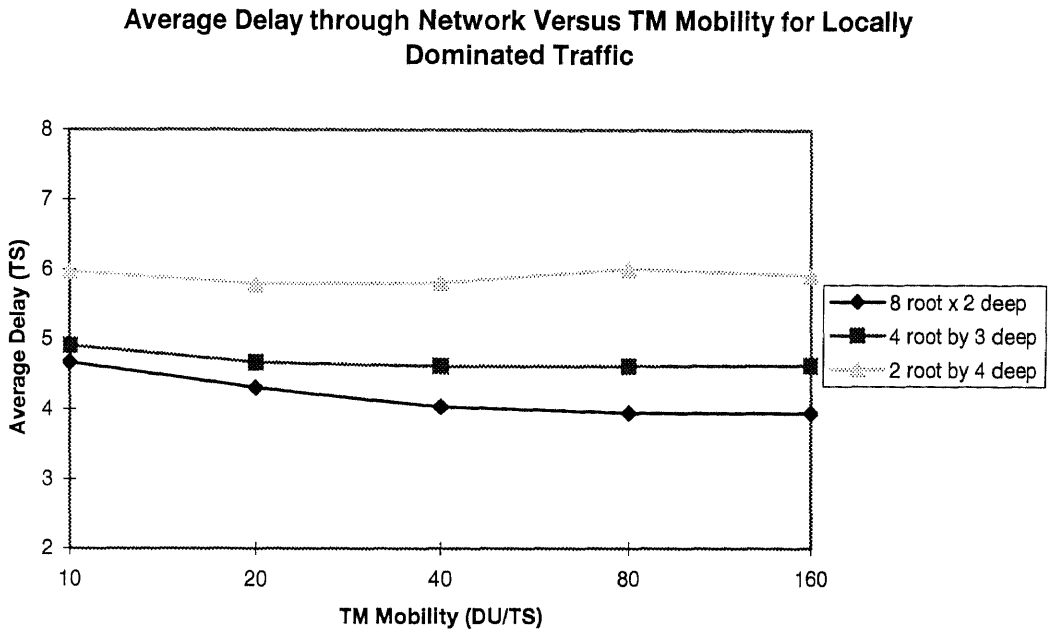

Figure 8.9: Average Delay through the network for the case where the messaging is dominated by local traffic. For this plot, it was assumed that 60% of the traffic would remain within the tree of the sending TM.

8.2.3 Performance with Unrestricted Mobility in the Network

When the restriction on the amount of asymmetry allowed in the network is dropped, and no procedures are implemented to attempt to equalize the loading on the network, the performance of the network drops, as would be expected. For this portion of the simulation, the network architecture was changed to a network that had 2 VLAs each initially containing 16 BSs and 160 TMs. Because the restriction on the loading of a BS was dropped, it was necessary to put more TMs and BSs into the simulation in order to improve the statistical relevance of the simulation. The new network was first tuned (i.e.

the resources were set) so that the performance of the new network matched the original (4 VLA, 28 BSs, 112 TMs) network. Following the tuning of the resources, the new network was used to study the effects of eliminating the asymmetry restrictions. Figure 8.10 shows the performance of the new network (with 2 times the base resources) with and without restriction on the network asymmetry.

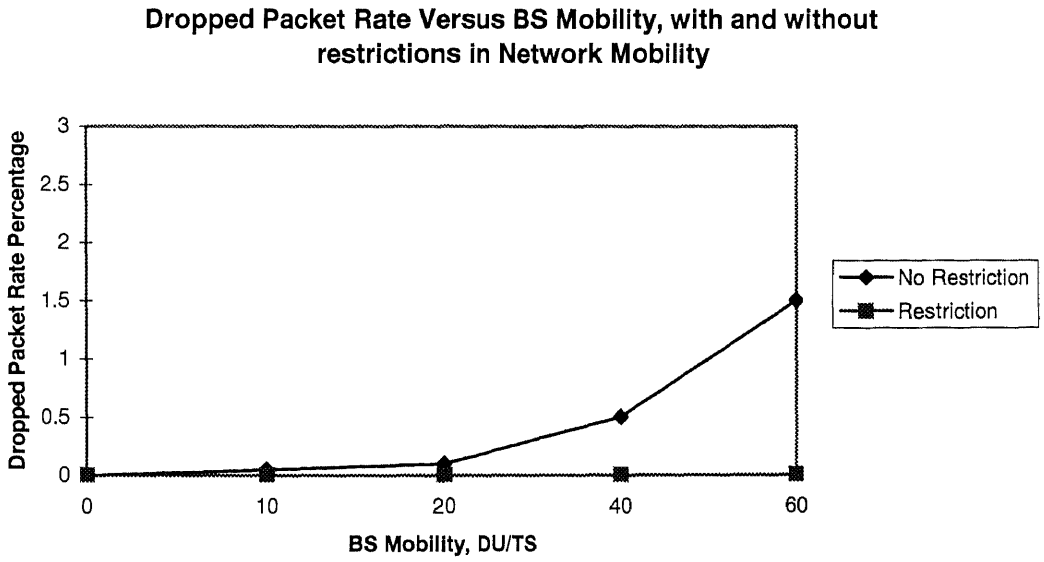

Figure 8.10: Dropped Packet Rate Versus Rate of Motion of the BSs with and without a limit on network asymmetry, motion of the TMs and BSs were both random.

As can be seen from figure 8.10, the dropped packet rate has increased as a result of the dropping of the BS loading restriction. When the motion of the TMs is random (as was the case for this simulation), the probability is fairly good that the TMs will remain relatively spread out in the service area. Because of this, the loading on the BSs, while not

even, is unlikely to become extremely unbalanced. If the conditions of the simulation are changed so that the TMs now have a direction to their motion, the results, as shown in figure 8.11, are dramatically different. Here, 50% of the TMs moved in a directed fashion. That is, they were directed, at different times in the simulation to move to specific areas of the service area. This type of motion might be expected of the TMs if their actions are coordinated (as in a military application).

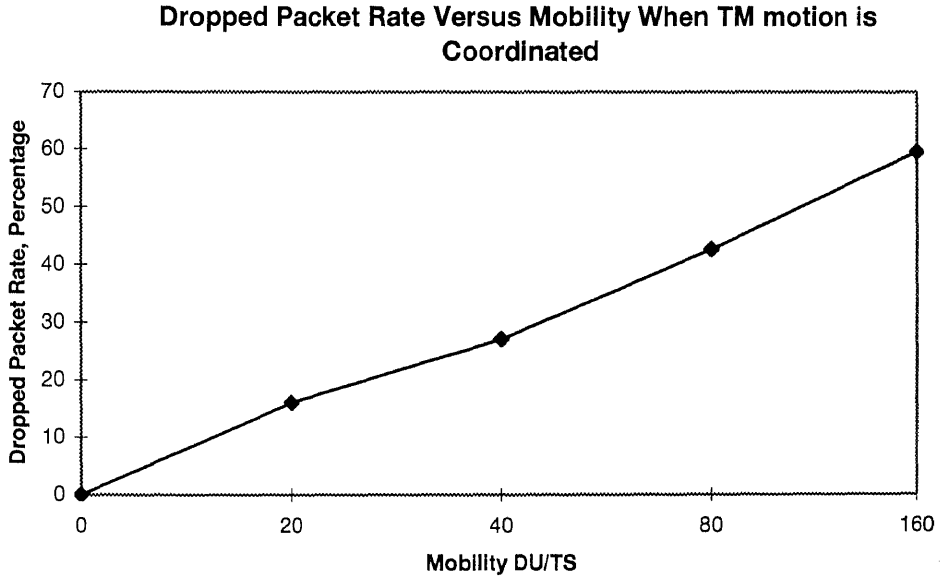

Figure 8.11: Dropped Packet Rate versus TM and BS Mobility. No Restrictions in Network Asymmetry. 50% of the TMs were directed to move to specific areas of the service area during the simulation.

Clearly, once the motion of the TMs is no longer random, severe problems exist in the network. To handle those problems, the algorithms presented in section 5 were used

to direct the motion of the BSs to help load balance the network. Figure 8.12 shows the results of adding in the distributed load balancing algorithm.

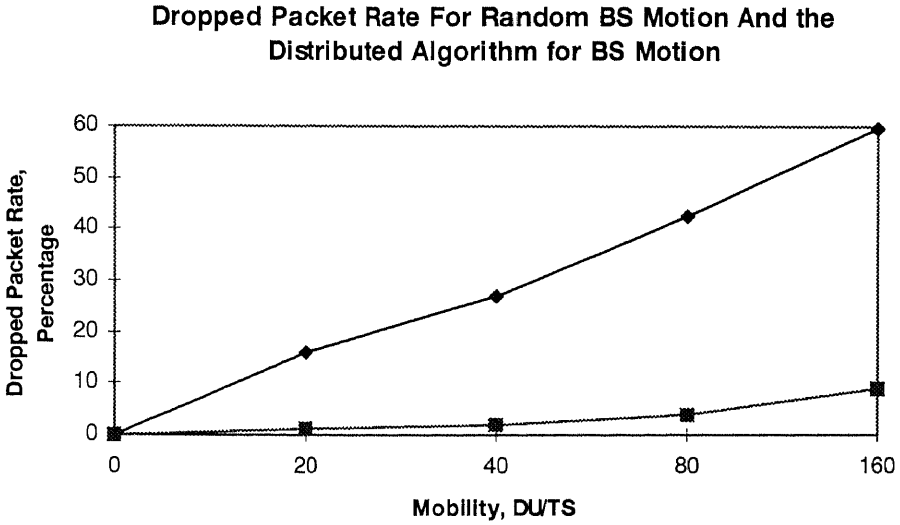

Figure 8.12: Comparison of the performance of the Network for Directed TM motion with no restrictions on Network Asymmetry with random BS motion and a distributed algorithm to direct the BS motion.

The performance of the network with the addition of the distributed algorithm greatly outperforms the case where no attempt is made to load balance the network. Upon examining the output of the simulation, it becomes apparent that the dropped packets occur in clusters and is the result of the fact that that algorithm is reactive in nature. That is, the algorithm uses feedback to generate the commands for the BSs rather a predictive algorithm. The two chief advantages to this approach is that, provided that the feedback gain is low enough, the solution will remain stable and the algorithm is straight forward and easy to understand making it easier to implement and debug. The drawback to a

feedback type of solution, however, is that a problem has to start to arise before the algorithm will attempt to handle it. For the particular network being studied here, there is little margin between when a BS is considered to be overloaded and when it starts to drop packets. This was done on purpose in order to make the solution to the problem more demanding (and hence stress the possible solutions harder). Providing more margin between the "overloaded" threshold and the point at which the BS starts to drop messages will increase the apparent performance of the algorithms at the expense of wasted network resources (more of the resources are essentially held in reserve).

Figure 8.13 shows the dropped packets as a function of time for the cases where the BSs were allowed to move randomly and when they moved to balance the load. The mobility rate for the plot was 160 DU/TS for both the TMs and the BSs.

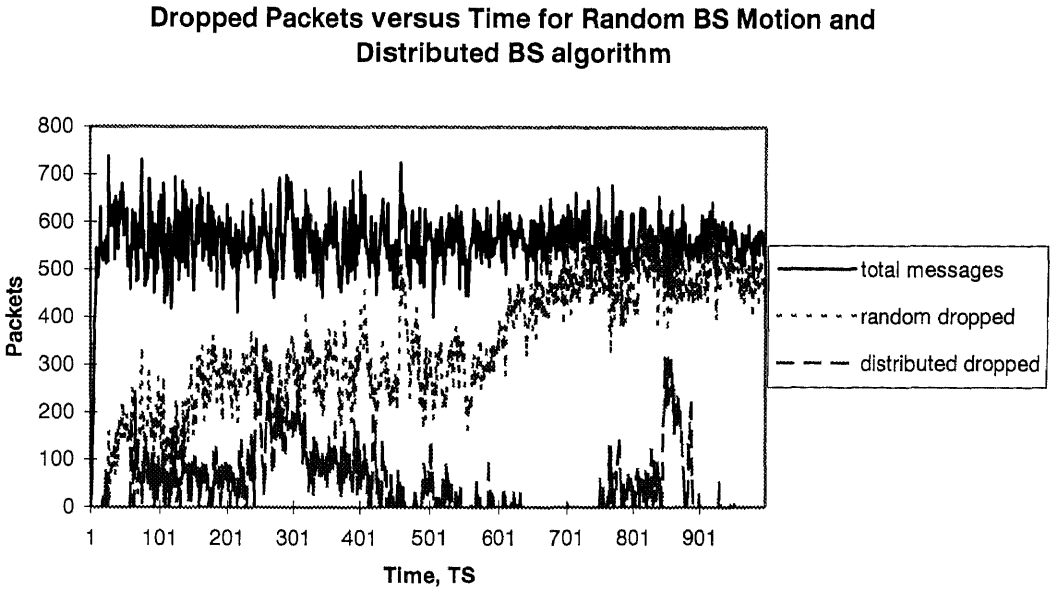

Figure 8.13: Dropped packets as a function of time for random BS motion and directed BS motion. Mobility rate was 80 DU/TS for BSs and TMs.

When the centralized algorithm for directing the motion of the BSs is used, the performance of the network improves somewhat, as might be expected. Figure 8.13 shows the dropped packet rate when the centralized algorithm is used. The network conditions were the same as for the case that is shown in figure 8.12. The main difference between the distributed and the centralized algorithm is that in the centralized case there is no chance that there is more than one BS attempting to help an overloaded BS whereas with the distributed algorithm it is at least possible that all of the "receiver" BSs are attempting to help the same overloaded BS (although this extreme case is very unlikely). A direct comparison of the distributed and centralized approaches to directing the BS motion is shown in figure 8.14.

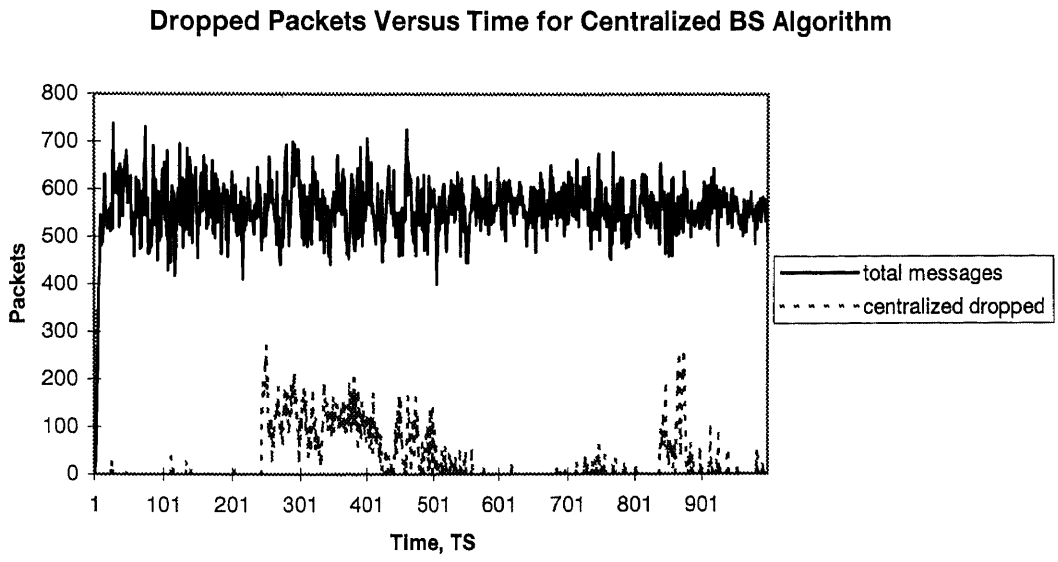

Figure 8.14: Dropped Packets versus time for the case where the BS where directed to move via a centralized algorithm. Both the BSs and the TMs had mobility rates of 80 DU/TS.

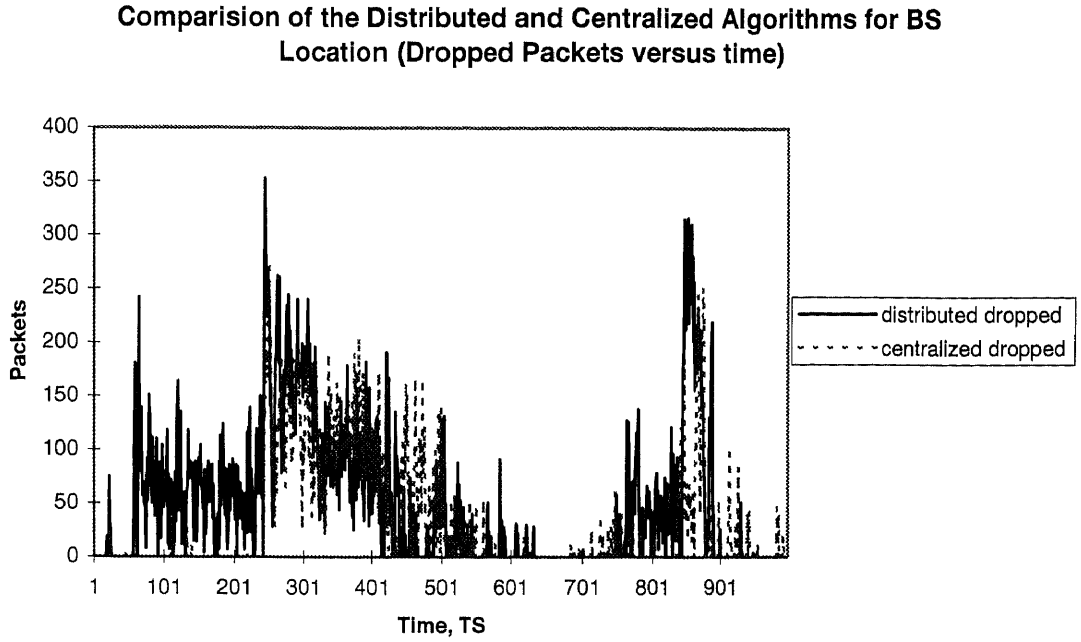

Figure 8.15: Comparison of the distributed and the centralized algorithms for control of the BS location.

8.2.4 Multicasting

Simulation of a multicasting group in the hierarchical network was performed using a single multicasting group which had 1 producer and a total of 20 members (all chosen at random). All TMs other than the multicast producer generated and sent messages in the same manner as in the previous simulations. One message was sent to the group each time step. A record was kept of each message that was delivered to each group member. In order to be able to study the performance of the multicasting portion of the protocols, the allowed asymmetry in the network was limited to ensure that dropped messages were not the result of overloading due to asymmetry. To facilitate data reduction of that record,

the simulation runs with multicasting were limited to 1000 time steps. Runs were made with no mobility in the network, TM mobility only, and both TM and BS mobility (both low and high).

The record of delivered messages was analyzed to determine the number of messages that were not delivered to one or more members of the group, the number that were delivered more than once, and how many were delivered out of sequence. For all of the above cases, it was found that the protocols successfully delivered 100% of the messages to all of the members of the group. This was true even for the case where the BS and TM rate of mobility was at 60 and 80 DU/TS, respectively. At times, the protocol (as implemented in the simulation) will deliver a message to one of the members of the group more than once (but never more than twice). This occurs due to the mobility of the nodes and can be eliminated with a minor change to the multicast protocol. Figure 8.16 shows the percentage of time that messages are delivered twice to a TM as a function of the TM and BS mobility. At the highest mobility rates tested, double delivery of messages occurred approximately 1% of the time.

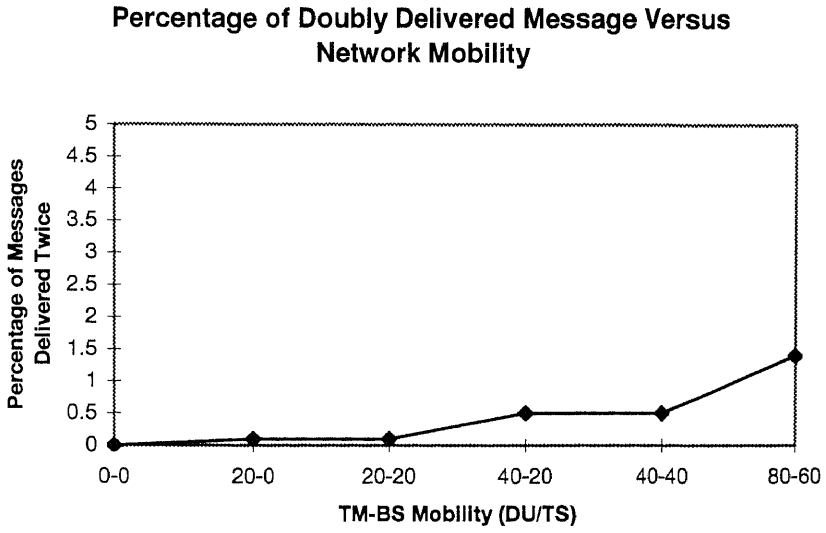

Figure 8.16: Multicasting message delivered twice versus network mobility.

Finally, the percentage of messages that were delivered out of sequence as a function of the network mobility is shown in figure 8.17. Again, this is caused by the mobility of the network (note the rate when mobility is zero).

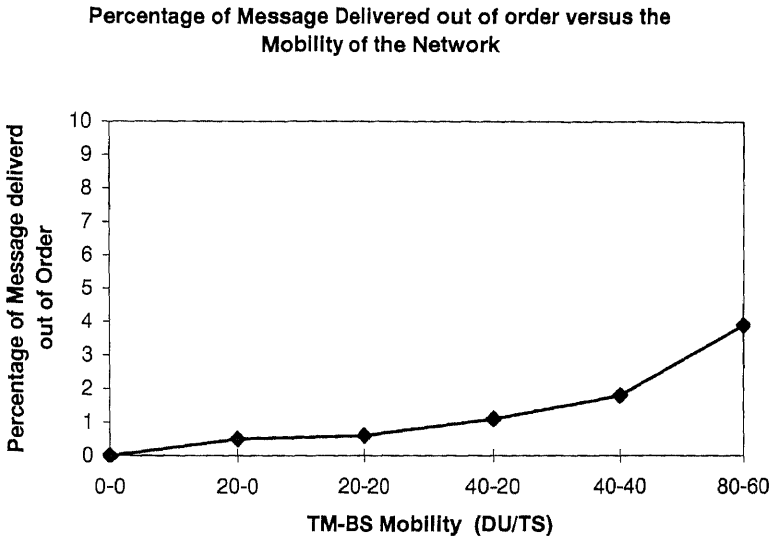

Figure 8.17: Out of order message delivery for Multicast messages as a function of the mobility of the network.

Note that there is no provision in the protocols to deliver messages to the destinations in order. The fact that the messages are largely delivered to the destinations in order is an indication of the speed with which the network updates the routing information.

CHAPTER 9

COMPARISON TO PREVIOUS WORK

The current proposal for the protocols build on the work in [4]. However, the current work differs from [4] in several important ways. The differences between the two works are enumerated below.

1. The requirement that each BS know about all other BSs and TMs in the tree has been dropped. To achieve this requirement meant that each BS must be informed of all moves that affect the tree. While this provision made it possible for BSs in different subtrees to communicate directly, it increased the complexity of the database and the number of control messages needed.
2. The database in the SC has been improved to include information on all nodes in the VLA. The previous work limited the SC database to information on the root BSs. This limitation forced the SC to query each root BS to determine the location of the destination node. That required a minimum of 2 control messages (query and reply) and on average will require (2*number of roots in the VLA)/2 control messages (since the message was unicast). The present proposal eliminates this query and respond entirely.
3. The requirement that update information be partially flooded has been eliminated. Each update is handled by a single message/reply and does not require propagation to nodes other than those in the delivery path.

4. Restrictions on the size of the network (and the subtrees that make up the network) have been eliminated. There are no restrictions on where a BS or a TM may attach to the network so that the network is allowed to have any configuration.
5. Load balancing in the network has been added for autonomous network control. These load balancing algorithms provide a means for more evenly distributing the TMs amongst the BSs when the position of the BSs is automatically controlled.
6. The method used to determine the new parent of a moving BS has been improved. The current proposal allows the BS to choose its new parent when moving based on the layer of the candidate parents and their loading. The previous work had relied on a broadcast message from the moving BS and multiple replies to determine the new parent BS. That work also applied the majority function to the BSs' network ID to choose a new BS. Since the network ID does not indicate tree depth, that method would not ensure minimum depth of the tree.
7. The requirement that TMs being served by a moving BS find a new BS to serve them has been eliminated. This results in a decrease in the control message traffic as well as an improvement in the service to the TMs.
8. The protocol has been strengthened in the area of handling race conditions. Previously, there had been little consideration given to race conditions that can occur when both a BS and one (or more) of its TMs attempts to move at the same time. The current proposal includes provisions to ensure that all possible combinations of moves are safe (result in accurate routing data).

9. Support for reliable multicasting has been added.

10. Support for retransmission of lost control messages has been added.

CHAPTER 10

CONCLUSIONS

A set of multi-hop protocols for fully dynamic wireless packet networks have been proposed and examined. These protocols are capable of handling a network in which the base stations as well as the terminals are capable of moving both their physical and logical locations. Delivery of packets is achieved simultaneous with reconfigurations of the network configuration. Simulations of the protocol have shown that it is stable and robust under a wide range of conditions. These protocols are capable of handling multicast as well as unicast transmission.

The protocols that are presented in this work, although not suitable for standard internetworks, are compatible with the current popular protocols used in those networks, especially TCP/IP. The only requirement for the connection of a network running this set of protocols to be connected to the Internet would be to issue it a set of IP addresses and set up the gateways to run IP on one side and this set of protocols on the other.

APPENDIX 1

SIMULATION DETAILS

The simulator used for the verification of the protocols and the performance of the assumed network was developed in C++ on a PC. The main object types for the simulation were the end user terminal (TM) , the mobile base station (BS), and the switching center (SC). All of the above objects were created at the start of a simulation run and existed for the life of the simulation.

In addition to the static objects, there were dynamic objects that were created and destroyed throughout the life of the simulations. These objects were used to represent the messages that were generated and passed between the TMs, BSs, and SCs. In addition there were small structures used to contain the information about an object (TM or BS) that an ancestor of that object would need (see description below). The major components of the objects (both data and methods) are presented below:

Structures

As mentioned above, the structures used in the simulation were designed to hold information to be used by the classes (see below). This information could pertain to the objects in the subtree (base stations and terminals) or the messages that are passed between these objects. In the case of the term and BS structures, the structure were designed to hold the information that a base station (or switching center) would have

about a terminal or other base station. In the case of the messages, the only data held in the message structure is the information that is actually needed by the simulation (i.e. there is no actual message held in the data message). The following were the major structures that were used in the simulation:

1. term

The term structure exists only as a part of the class for the BSs and the class for the SCs. Its purpose is to allow those objects to maintain information about the individual TMs that are in its subtree (including those that are directly attached to the BS). The information held in a term structure is:

1. The ID of the terminal
2. The ID of the BS that directly serves the terminal
3. A flag to indicate if the terminal is in the process of moving
4. A pointer to the next *term* in the list
5. A flag to indicate whether the link between the current BS and the terminal is operational (used as part of the moving process)

2. BS

The BS structure serves a similar purpose for the BS as the term structure does for the TM. The components of the BS structure are:

1. its own ID
2. ID of the next hop to get to the base station
3. a pointer to the next *BS* in the list

3. data_msg

The data message structure contains the information required to route the message. The data message structure therefore contains:

1. The ID of the destination TM (all data messages go from TM to TM)
2. the time that the message was created
3. A sequence number for the message (in the simulation this was a global sequence number that was used for debugging purposes)
4. A pointer to the next *data_msg* structure in the linked list.

4. control_msg

The control message is used to update the routing information that the network has for a particular object (TM or BS). The significant control message structure components are:

1. The ID of the Destination
2. The ID of the source
3. A flag to indicate if this message is an answer flag (acknowledgment)
4. A field for the type of control message (TM move, BS move, etc)
5. A field containing the ID of the object being updated
6. A pointer to a structure containing a copy of the data fields for a *base* object (see below). This pointer is NULL if the object being updated is not a *BS.*
7. A pointer to a list of addresses. The control message structure doubled as a multicast message structure (recall that the multicast messages have the same priority as a control message)
8. A pointer to the next *control_msg* in the linked list.

Classes

The following classes were implemented in order to handle the data and methods needed for each of the three major components in the simulation (TMs, BSs, and SCs). The basics for these classes were:

1. terminal

The TMs maintained the following information:

1. Its location (X Y coordinates)
2. The ID of its "parent" (the BS that it is currently communicating with)
3. The number of packets left in the current transmission
4. The last TM that it communicated with
5. The current direction of travel
6. A flag to indicate whether it is in the process of moving to a new location in the network
7. A flag to indicate if it is the source for a multicast session
8. A counter to determine the amount of bandwidth that is remaining for this time step.
9. A pointer to the next *terminal* in the linked list

The TMs were supplied with the following methods:

1. Generate messages (control and data)
2. Send messages to its parent b*ase*
3. Move to another physical location (change its X Y coordinates)

4. Decide on whether to move to another BS (and initiate move, if appropriate)
5. Generate a multicast message (if it is the source for a multicast session)

2. base

The Base stations are considerably more complex than the terminals. In addition to keeping information about itself, the *base* was also required to hold information about all of the components that existed in the subtree below them. To support all of the functionality of the base station, the *base* class contained the following data as part of the simulation:

1. A linked list of *term* structures for the TMs that it serves (communicated directly with)
2. A linked list of the *BS* structures for the base stations that are part of its subtree.
3. A linked list of *term* structures for the TMs that belong to the base stations in the subtree.
4. A linked list of the *control_msg* structures in the queue to be sent out
5. A linked list of the *data_msg* structures in the queue to be sent out
6. A set of timers for control messages that were originated at this *BS*
7. Its own current physical location
8. The identity of its parent
9. The layer in the network that it currently resides at.
10. The current direction of its motion and the location it is traveling to (if it is helping another BS)

11. A flag to indicate if it is helping another BS (for the autonomous load balancing algorithms)
12. A target set of X Y coordinates for when the base station is moving to help another base station
13. The number of terminals that it is serving (used in the load balancing algorithms)
14. A flag to indicate if it is in the process of moving to another (logical) location in the network
15. Counters to keep track of the number of slots available in the data and control messages queues (when the counters hit zero the queue is full and the *BS* will drop further messages that come in.
16. A pointer to the next *base* in the linked list

The *base* class was provided with the following methods:

1. Receive data and control message and place them on their respective queues.
2. Process control messages (read the contents and update the routing information, if necessary)
3. Determine the next hop for data and control message and send the messages to the next network node.
4. Process multicast messages, sending multiple copies of the message, when appropriate
5. Physically move to a new location (X Y coordinates). This method actually consisted of three different methods, one to move about in a pseudo random way, another to move to help other base stations according to a distributed algorithm

and a third to move to help other base stations according to a centralized algorithm.

6. Decide when to move to a new logical (network) location
7. Execute the distributed load balancing algorithm
8. Initiate a move to another logical location in the network

3. SC

The SCs maintain much of the same information and perform many of the same tasks as the BSs. Two significant differences between the SCs and the BSs are that the SCs do not have the ability to move randomly (since they are a set of satellites) and the SCs do not serve any TMs directly.

The SCs are provided with the following link lists of data:

1. A linked list of *BS* structures for the base stations that are part of its VLA.
2. A linked list of the *term* structures for the TMs in the VLA.
3. A linked list of the *control_msg*s in the queue to be sent out
4. A linked list of the *data_msg*s in the queue to be sent out
5. A set of timers for control messages that were originated at this *SC*
6. Counters to keep track of the available slots left in the control and data message queues (when these counters hit zero, the *SC* will drop messages)
7. A pointer to the next *SC* in the ring.

The methods that were supplied for the *SC* were:

1. Receive data and control message and place them on their respective queues.
2. Process control messages (read the contents and update the routing information, if necessary)
3. Determine the next hop for data and control message and send the messages to the next network node.
4. Process multicast messages, sending multiple copies of the message, when appropriate
5. Execute the centralized load balancing algorithm
6. Generate control messages (answers to other control messages)

Data Organization

The memory space for all of the objects in the simulation is allocated dynamically in order to preserve memory (implementation was on a PC). A master linked list of the *terminal*s is maintained, as well as a separate list for the *base*s and the *SC*s. These master lists were the main data structures for the simulation. The main loop for the simulation consisted of looping through these data structures and visiting each of the nodes, one at a time. All of the *terminal*s were visited followed by the *base*s, and then the *SC*s. One loop through all of the components constituted a "time step". The notion of a time step was necessary because the program is simulating a distributed computing environment on a signal processor system. It is therefore necessary to break continuous time up into small discrete intervals. It is assumed that all actions that occur with a time step are occurring simultaneously. The actual length of the time step is arbitrary but can be considered to be small.

For each time step, the *terminal*s, *BS*s, and *SC*s are all processed, one at a time in that order. For each node, the steps outlined in the protocols descriptions above are executed. For *BS*s and *SC*s, the main effort is centered on processing the control messages and sending out control and data messages. Both of the message lists (control and data) are maintained as a FIFO list. The *BS* (or *SC*) starts at the beginning of the control message list (since the control messages take precedence over the data messages) and processes messages until its allotted data transfer for the time step has been exhausted. For each message, the *BS* (or *SC*) examines the arrival time prior to processing. If the arrival time is the same as the current time, the message arrived during the current time step and no action would be taken. This could (and often does) happen and is a natural consequence of the fact that the distributed computing environment is being simulated on a single processor. The problem is that events that are supposed to occur simultaneously in the real world occur serially in the simulation. If a node earlier in the linked list sends a message to one later in the list (or to one in a list that is processed later), the second node should not be able to actually process the message during that same time step.

Messages are processed by the nodes and sent on to the next node in the route. For each message that is sent, the "bandwidth" available to that node for the time step is reduced based on the type of message (control or data). When the allotted bandwidth is used up, the node stops processing messages and leaves the remainder in the queue for the next time step.

To send a message to a TM, the current processing node (*SC* or *BS*) searches its list of served *term*s, If it is in that list, the message the proper bandwidth allocation is reduced, the message is recorded as being delivered successfully and the message is

deleted. Otherwise the node (*SC* or *BS*) searches the list of *terms* in its subtree. If the TM is there, the node will get the parent *BS* from the *term* structure and search its child *BS* list for that BS. The entry in the child *BS* list will indicate the next hop in the path to that *BS*. If the *term* is not anywhere in the subtree, the message is sent to the parent of the current processing node.

When the proper next node in the path to the destination is found, the master list of that type of node (*SC* or *base*) is searched to find the entry for that next node. When the entry is found, the message is appended to the message list for that node (either *control_msg* or *data_msg*) and is deleted from the message list of the current node (the pointer reference to it is removed). The available bandwidth for the current node is reduced and the move time for the message is updated for the current time.

The length of the queues that each of the nodes are permitted to have is limited (to simulate this fact of a real system). When the queue is full all subsequent attempts to send messages to that node would fail and the incoming messages are dropped (deleted from memory).

Load Balancing

To handle the load balancing algorithms, a central list of the base stations that were in need of help was kept. For the distributed case, this list held the last known location of the base station in need of help (as reported by the base station) as well as an estimate on the load on that base station. During the simulation, base stations that were available to help other base stations would search this list to determine which base station to help based on the algorithms described in the main portion of this work. Although the list of

base stations in need of help was not actually "transmitted" to each base station, the bandwidth available to the base stations was reduced to simulate the sending of the information.

For the case of the centralized algorithm, a similar list was maintained. In this case, the list contained an entry for each of the base stations that existed in the VLA. The centralized algorithm was then used on this list to determine which base stations would be directed to help other base stations. Again, the message traffic required to maintain the list and to issue the directed movement control messages was simulated by reducing the amount of bandwidth available to the base stations when the messages were "sent".

This simulated method for the load balancing efforts was chosen because the amount of traffic generated by this activity was relatively small and did not seem to warrant a separate data structure to handle them. Because of the predictable nature of the traffic, it was a simple matter to properly account for the increased traffic due to the load balancing messaging.

Performance Data Generation

In order to evaluate the performance of the network being simulated, data was collected during the simulation runs. For each time step, the number of messages delivered and the number dropped due to queue overflow (or timeout on the message) were counted. In addition, the number of hops required and the delay through the network were recorded for each of the messages that sent (control messages were counted separately from data messages). At regular intervals during the simulation, the hierarchical relationships in the network were dumped to a file for later examination. This provided the ability to

determine the balance of the load on the network as well as to determine whether the routing information was consistent in the various nodes.

During the early simulation runs (when the protocols were being verified) data on all dropped messages were recorded (source destination, node that dropped it, time, etc.) in order to facilitate debugging of the protocols. In addition, performance data is gathered for each of the nodes in the system. This data included the number messages delivered and dropped for each of the nodes and the number of TMs that were being served by the node (BSs only).

REFERENCES

1. E. Ayanoglu and K.Y. Eng and M.J. Karol, "Wireless ATM: Limits, Challenges, and Proposals," *IEEE Personal Communications Magazine*, Vol. 3, August 1996, pp. 18-34.

1. K.Y. Eng, M.J. Karol, M. Veeraraghavan, E. Ayanoglu, C.B Woodworth, P. Pancha and R.A. Valenzuela, "A Wireless Broadband Ad-Hoc ATM Local Area Network," *ACM/Baltzer Wireless Networks Journal*, Vol. 1, May 1995, pp. 161-174.

2. A. Myles, D. Johnson and C. Perkins, "A Mobile Host Protocol Supporting Route Optimization and Authentication," *IEEE Journal on Selected Areas in Communications, special issue on Mobile and Wireless Computing Networks*, April 1995.

3. F. Akyildiz and W. Yen and B. Yener, "A New Hierarchical Routing Protocol for Dynamic Wireless Networks," to Appear in *WINET Journal* (previous version published in *Proc. INFOCOM'97*), April 1997.

4. R.D. Mouldin and P.A. Young, "Wireless Battlefield Tactical Networking Supporting C20TM and C41 for Army Warrior," *Proceedings of Milcom'94*, 1994.

5. S. Murthy and J. J. Garcia-Luna-Acheves, "A Routing Protocol for Packet Radio Networks," *Proceedings. of ACM Mobicom'95*, 1995.

6. A. Myles and D. Johnson and C. Perkins, "The Internet Mobile Host Protocol (IMHP)," *Proceedings. IEEE INFOCOM'96*, March 1996.

7. A. Acharya, A. Bakre and B.R. Badrinath, "IP Multicast Extensions for Mobile Internetworking," *Proceedings of IEEE INFOCOM'96*, March 1996.

8. A. Acharya, and B.R. Badrinath, "Delivering Multicast Messages in Networks with Mobile Hosts," *Proceedings of IEEE Intl. Conf. on Distributed Computing Systems* , May 1993.

9. Ioannidis D. Duchamp and G.Q. Maguire, "IP-based Protocols for Mobile Internetworking," *Proceedings of ACM SIGCOMM Symposium. on Communication, Architectures and Protocols*, September 1991.

10. Huseyin Uzunalioglu, Wei Yen, and Ian F. Akyildiz," A Connection Handover Protocol for LEO Satellite ATM Networks," to appear in *ACM Mobicom '97*.

11. Andrew S. Tanenbaum, *Computer Networks Third Edition*, Upper Saddle River, NJ, Prentice Hall, 1996, Chap 4.

12. Mukesh Singhal and Niranjan Shivaratri, *Advanced Concepts in Operating Systems*, New York, NY, McGraw-Hill, 1994, pp259-292.

13. M. Stumm, "The Design and Implementation of a Decentralized Scheduling Facility for a Workstation Cluster," *Proceedings of the 2nd Conference on Computer Workstations*, March 1988, pp12-22.

14. F. Douglis. and J. Ousterhout, "Process Migration in the Sprite Operating System," *Proceedings of the 7th International Conference on Distributed Computing Systems*, Sept, 1987, pp18-25.

15. M.J. Litzkow, M. Livny, and M. W. Mutka, "Condor- A Hunter of Idle Workstations," *Proceedings of the 8th International Conference on Distributed Computing Systems*, June 1988, pp104-111.

16. M. Schwartz, *Telecommunications Networks, Protocols, Modeling and Analysis*, Reading MA., Addison Wesley Publishing Company, 1988, pp21-56.

17. J.F. Dosiere, T. Zein, G. Maral, and J.P. Boutes, "A Model for Handover Traffic in Low Earth Orbit (LEO) Satellite Networks for Personal Communications," *International Journal of Satellite Communications, Vol II*, 1993, pp145-149.

18. K.Y. Eng, M.J. Karol, et. al., "A Wireless Broadband Ad-Hoc ATM Local Area Network," *Wireless Networks, Vol I, Issue 2*, 1995, pp161-174.

19. H. Hansen, "Connection Management Functions of a Private Wireless ATM Network," Master's thesis, Department of Computer Science, Helsinki University of Technology, March, 1996.

20. Liming Wei and Deborah Estrin, "The Trade-offs of Multicast Trees and Algorithms," *International Conference on Computer Communications and Networks*, 1994.

21. Sneha K. Kasera, Jim Kurose, and Don Towsley, "Scalable Reliable Multicast Using Multiple Mulitcast Groups," Department of Computer Science, University of Massachusetts, Technical Report TR 96-73, October, 1996.