INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.

UMI

A Bell & Howell Information Company
300 North Zeeb Road, Ann Arbor MI 48106-1346 USA
313/761-4700 800/521-0600

UNIVERSITY OF OKLAHOMA

GRADUATE COLLEGE

ALGORITHMS FOR ROUTING AND REROUTING IN MOBILE WIRELESS AND AD-HOC NETWORKS

A Dissertation

SUBMITTED TO THE GRADUATE FACULTY

in partial fulfillment of the requirement for the

degree of

Doctor of Philosophy

By

Venkatagopal Racherla

Norman, Oklahoma

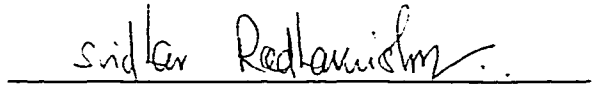
1999

ALGORITHMS FOR ROUTING AND REROUTING IN MOBILE WIRELESS AND AD-HOC NETWORKS

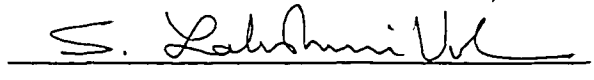
A DISSERTATION APPROVED FOR THE

SCHOOL OF COMPUTER SCIENCE

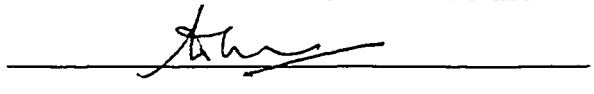
BY



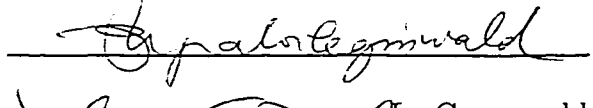
Sridhar Radhakrishnan, Chair



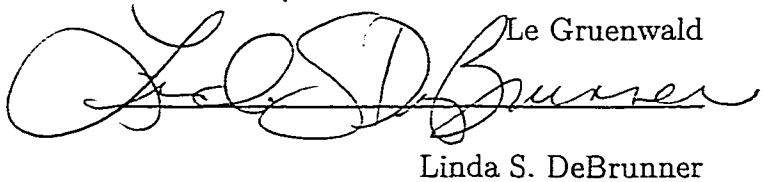
S. Lakshmivarahan



Sudarshan K. Dhall



Le Gruenwald



Linda S. DeBrunner

Glossary of Abbreviations

ARQ	Automatic Repeat Request
ATM	Asynchronous Transfer Mode
BISDN	Broadband Integrated Service Digital Network
BS	Base Station
CBT	Core Based Tree
COS	Cross Over Switch
ELN	Explicit Loss Notification
FDL	Form description language
FEC	Forward Error Correction
FHRP	Footprint Handover Re-route Protocol
HTML	Hyper Text Markup Language
IP	Internet Protocol
I-TCP	Indirect TCP
LAN	Local Area Network
LEO	Low Earth Orbiting
MAC	Medium Access Control
MCDS	Minimum Connected Dominating Set
MH	Mobile Host
MR	Mobile Representative
MS	Mobile Station
MSC	Mobile Switching Center
MSS	Mobile Support Station
MTU	Maximum Transfer Unit

NCNR	Nearest Common Neighbor Rerouting
NNI	Network-Network Interface
OSI	Open Systems Interconnection
QoS	Quality of Service
RTP	Reliable Transport Protocol
SACK	Selective Acknowledgments
SH	Stationary Host
TCP	Transfer Control Protocol
URL	Uniform Resource Locator
UIL	User Interface Logic
WAN	Wide Area Network

Contents

List of Figures

Abstract

Mobile computing - sometimes called Nomadic Computing - as the name suggests is computing with mobile users. The ideal case for mobile computing would be a complete transparent networking and computing for all users irrespective of their physical location, motion, computing environment and communication network. In a connection-oriented cellular environment, the network consists of a stationary fixed backbone made up of base-stations and a set of mobile hosts that can move around in the network. When the mobile hosts moves to the new cell, a new route has to be found between the base stations corresponding to the new cell and the cell at the other end in a communication session. This process of routing is called rerouting. A specialized case of mobile wireless network is an *ad-hoc* network where the network contains only mobile hosts and no base stations or any centralized routing infrastructure. The ad-hoc networks are characterized by highly dynamic and fast changing topology. Routing of packets in ad-hoc network is done by a store-and-forward mechanism from the source mobile host to the destination mobile host using intermediate mobile hosts. A pair of mobile hosts in an ad-hoc network can communicate with each other if they are in radio contact. This dissertation focuses on the following issues involved in routing and re-routing in connection-oriented mobile networks and ad-hoc networks:

1. *Rerouting in Connection Oriented Mobile Networks*: We have devised a framework for comparing, analyzing, and evaluating various rerouting schemes for connection-oriented mobile networks using several performance metrics. We also provided a comprehensive classification of rerouting schemes. In order to study various ways to improve the performance of network protocols, we

studied parallelism in messaging in the rerouting protocol. We suggest mechanisms to improve the performance of rerouting including radio-hints, parallel messaging.

2. *Rerouting in Mobile-Mobile Connections in Connection-Oriented Networks*: We have presented a distributed rerouting algorithm for mobile networks where both the ends of a connection are mobile hosts (Mobile-Mobile connections). Rerouting techniques proposed for connections with only one end that is mobile fail to perform rerouting effectively. Other rerouting schemes proposed for mobile-mobile connections perform non-optimal rerouting. We have compared our proposed algorithm with all the other Mobile-Mobile rerouting schemes using several performance metrics, and in all cases our algorithm outperforms the other schemes.

3. *Evaluation of the effect of the various rerouting schemes on the performance of Wireless TCP schemes*: In a mobile environment, TCP can not discriminate packets dropped due to lossy links, intermittent connectivity, handoffs and the consequent rerouting with those dropped due to network congestion. We have studied the effect of how rerouting schemes effect the performance of Wireless TCP using simulation. Our study is the first to analyze the effect of various rerouting schemes on the throughput of various wireless TCP's schemes.

4. *Routing in Ad-Hoc Networks*: We propose an efficient distributed spanning tree based routing algorithm for ad hoc networks that adjusts dynamically to fast changing dynamic topologies of ad-hoc networks. We achieve this by maintaining a forest of spanning trees. Our approach is practical and has

less time, space and message complexities when compared with the algorithms described in literature.

Chapter 1

Introduction

1.1 Introduction to Mobile Computing

Computer and Communication technologies have undergone a tremendous amount of change in the past five years. The World Wide Web is a household name. Companies, organizations and individuals give the URL (Uniform Resource Locator) of their homepages in commercials on television, radio and in magazines. Portable and notebook computers, smart credit card devices, high speed modems, hand-held spread spectrum radios, global positioning system (GPS) in rental automobiles herald the coming of age of computer communications. Telephone, cable and computer companies are joining forces to build the information super highway of the future which will carry data, voice, image, video, graphics using digital and analog technologies, wireless and wired technologies.

Most of us are *nomadic/mobile* - constantly on the move between offices, homes, planes, trains, automobiles, conference rooms, class rooms in schools etc. However, we constantly need to tap into computer resources either at our "home" base or at

some other geographically distant location. Due to the lack of set standards for all computer and communication technologies, there are many different computer and communication technologies that one could have to use when away from the home base. This gives rise to a lot of compatibility and resource problems. For example, a graduate student is working on a computationally intensive graphical computer simulation on his advanced workstation at an engineering laboratory, in America. He travels to a conference in Germany and is unable to access the simulation results from the small PC provided by the conference organizers because of the slow network connectivity and lack of proper software to handle the communication with the student's home base namely his workstation. In essence, there are presently no system supports which provide a true *transparency* in computing and communications especially with regards to *mobile* users.

Mobile Computing - sometimes called Nomadic Computing - as the name suggests is computing with mobile users. The ideal case for mobile computing would be a complete transparent networking and computing for all users irrespective of their physical location, motion, computing environment and communication network. Though wireless communications could be a part of mobile computing, it is not an essential component. This is because, the users need not be always in motion and there could be a connectivity between the users' "present" computer and their home base through hardwired access points.

Mobile Computing has invoked a lot of interest because of the following reasons:

- It embodies the changing nature of our everyday "digital" lives.
- It aims to provide a lot of convenience to the user - saving time and resources.

- Mobile computing also aims to provide transparent functioning of computer systems alleviating compatibility and portability problems.

- It is multidisciplinary area. Computer scientists, computer engineers, communication engineers, semiconductor physicists, space and satellite technologists are immensely interested in it owing to its wide applicability in the daily lives of people.

1.1.1 Architecture of a Typical Mobile Computer System

A typical mobile computer system (See 1.1) consists of the following components:

- A static communication backbone using wired means.

- A set of static hosts.

- Sets of mobile hosts (MHs) that can communicate with a predetermined static host.

- Mobile subnets/Wireless cells.

The static communication backbone connects all the static hosts sometimes called mobile support stations (MSSs) or base stations(BSs). As the name suggests, the static communication backbone and the static hosts are not mobile. Each MSS can communicate with a set of MHs. The MHs can move within a limited geographical "region of influence" referred to as a wireless *cell* or a mobile subnet. However, the MHs maintain connection possibly through wireless means to the assigned BSs from different locations in the cell at different times. In effect, the network topology

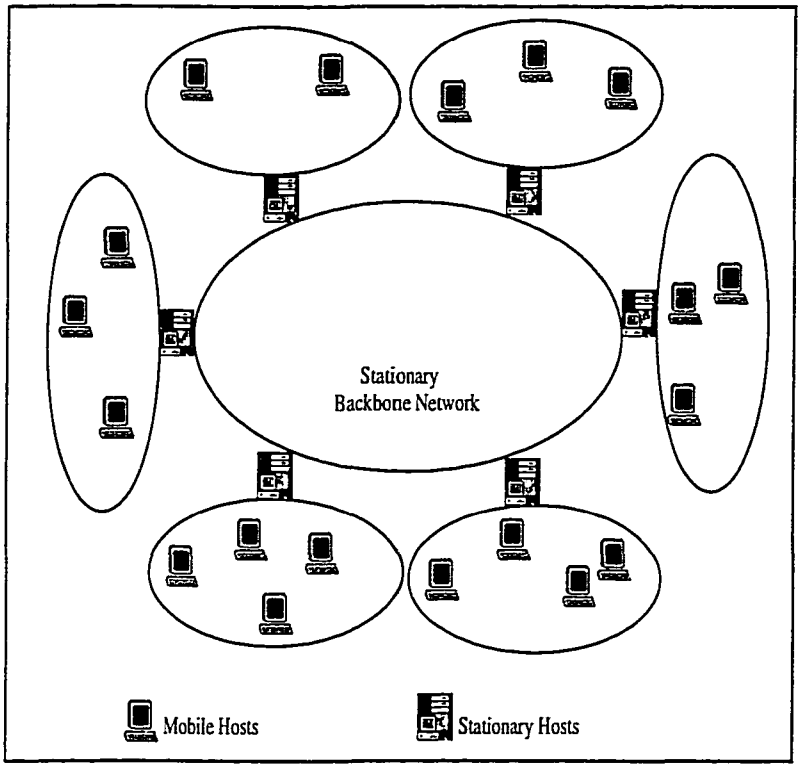


Figure 1.1: Architecture of a Mobile Network System

changes dynamically with time. The processing power of MHs such as PDAs, palm-top/notebook/personal computers is usually much smaller than that of the BSs. Thus, MHs rely on the computing power of the BSs to perform their own computations. Communication between the MHs and the BS can be either point-to-point or a broadcast message.

It is desirable for mobile hosts in some situations to form a temporary network without the aid of centralized administration or an established network backbone. For example, when a group of workers are to be deployed quickly during a natural disaster or business delegates assembled in a lecture hall. This temporary network is called an *ad-hoc* network. Ad-hoc networks can be considered a special case of

mobile networks. We shall discuss about the issues involved in ad hoc networks later in the dissertation. Figure 1.2 shows an example of an ad hoc network. The mobile hosts can talk to each other if they are in radio-proximity.

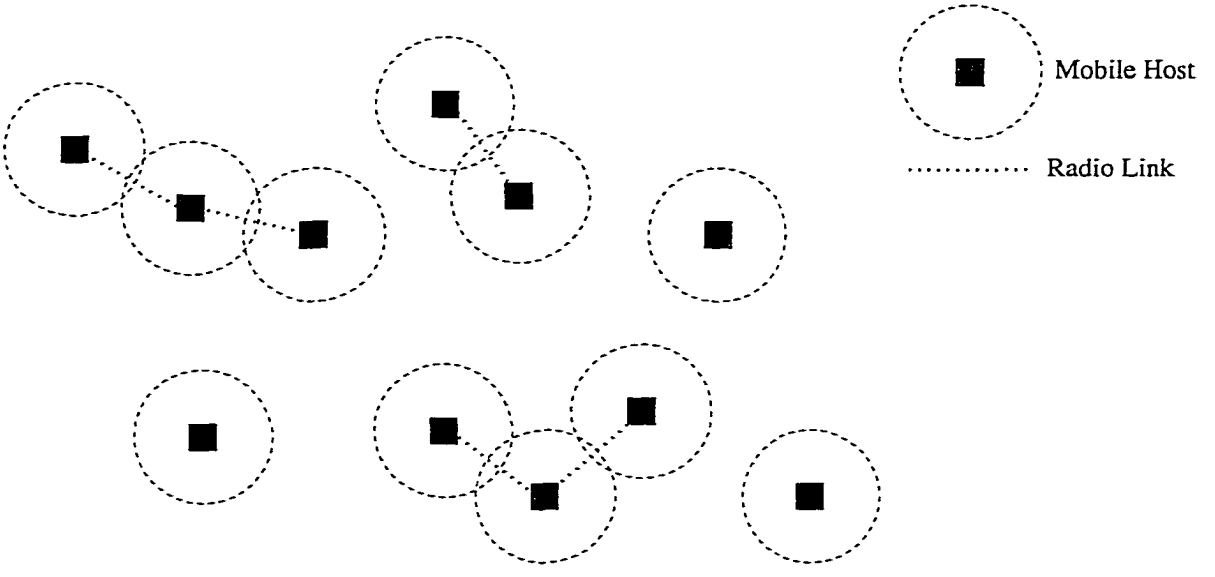


Figure 1.2: An ad hoc mobile network

1.1.2 Important Characteristics of Mobile and Wireless Networks

Mobile and wireless networks have the following characteristics:

- *Mobility*: In mobile and wireless networks, the user and the computer hosts can move.
- *Nomadicity*: Users can move and connect from various points to the network.
- *Limited Resources*: Mobile hosts (typically hand held, wearable computers) have limited computing capability, limited memory and work on batteries

which has a small life (typically few hours)

- *Lossy Wireless Medium*: The wireless medium has a limited bandwidth is error-prone and the consequently the network suffers from frequent disconnections and link down time

1.2 Research Issues in Mobile Computing

Mobile computing is a paradigm shift in computing and telecommunications caused by the mobile and nomadic lifestyles of people. This area is still in its infancy as there are many unresolved research issues. Researchers have been working on: Mobile Distributed Applications [4, 9, 15, 23, 24, 25, 69, 132], Routing in Mobile Systems [43, 92, 106, 107, 114, 116, 118, 127, 135, 160], Privacy and Security issues [22, 49, 61, 143], Mobile Multicast algorithms [3, 5, 6, 42, 58], Mobile Location and Management Strategies [26, 94, 119, 120, 123, 124, 125, 140], Distributed File Systems and Caching [17, 18, 16, 19, 29, 44, 57, 72, 73, 76, 78, 77, 90, 96, 97, 104, 150],Web and Internet Applications [7, 27, 28, 34, 37, 38, 86, 145], Database Applications [35, 59, 84, 89], New communication architectures [8, 11, 50, 80, 54, 55, 62, 65, 70, 133, 153], Mobile Environments [36, 41, 98, 109, 110, 111, 162], Mobile Agents and Objects [39, 46, 56, 82, 163], Mobile IP [21, 45, 48, 71, 101, 135], Wireless TCP [30, 52, 53, 93, 165], and Specialized Protocols [2, 31, 74, 95, 100, 121, 129, 151].

As mentioned earlier, the area of mobile computing is multidisciplinary in nature. We describe below briefly at some specific research issues in mobile computing.

1.2.1 Design of Mobile System Specification

- Design the architecture and protocols of the mobile computer.
- Provide interfaces for interoperabilty and compatibility of computer and communication devices.
- Deal with mobility, unpredictability of users and computer and communication components in the system.
- Maintain a high QoS and fault-tolerance in the system.
- Allow for system expansion, coping with failures, dynamic reconfiguration of system components.
- Design system with the cooperation among system components as a bedrock principle.
- Design a reference model like the OSI/ISO model which will allow a hierarchical delegation of work and resources.

1.2.2 Transparent Virtual Networking

- Design techniques to locate the MHs dynamically.
- Find ways to enhance the bandwidth between the MHs and the MSSs.
- Devise protocols for point-to-point and broadcasting mode of communication.
- Design compression algorithms for managing the huge volumes of data involved in communication.

- Design and implement security measures for transactions, user authentication.
- Devise efficient routing strategies for communication in the static network and the cells.

1.2.3 Adaptive Database Management and Information Storage

- Implement Location services for locating users and hosts.
- Implement queries on location dependent data.
- Perform consistent replication of data.
- Deal with disconnection of links and nodes.
- Perform efficient database recovery upon failures.
- Design intelligent databases to deal with servers which have different granularity, consistency, schema of data.
- Maintain cache consistency.
- Devise good caching and cache replacement algorithms.
- Design specialized data structures for storing multimedia data. Design data labeling, storage and retrieval schemes to improve system efficiency.
- Devise file naming conventions to facilitate network navigation.

1.2.4 Discovery of Resources and Agents

- Design dictionary and directory databases that acquire knowledge from the system and the users.
- Deal with mismatches between data as stored on different servers viz. instance mismatch, unit mismatch and context and structure mismatch.
- Automate databases to study the profiles of clients, middleware and servers and store the information using collection agents.
- Devise strategies for adaptive agents, relaxation agents, explanation agents and association agents

1.2.5 Miscellaneous

- Reduce power consumption of MHs since these usually operate on batteries.
- Reduce the size and weight of portable terminals.
- Design mobile computer system simulations to evaluate designs. Design simulation languages tailored for such systems.
- Design a set of performance measures for evaluation and build tools to evaluate the same for such a system.
- Design ergodynamic user interfaces.
- Monitor and manage dynamically changing network resources and topology.

1.3 Mobile Computing Projects and Implementations

There are dozens of groups working in academia, research laboratories and industries researching on mobile computing. We briefly describe below some examples of real life mobile computing systems.

DIANA

Researchers from Sun Microsystems and Stanford University [11] have devised a new application architecture named DIANA – Display Independence and Asynchronous Network Access. This architecture de-couples the user interface logic and communication logic from the processing logic of each application to deal with the diversity of user interface and varied communication paradigms. DIANA defines UIL - User Interface Logic - that is independent of the client applications. A form description language FDL has been developed to express the user interface types to the UILs. The network manager called the courier supports network management using multiple network protocols. The couriers and agents together manage the network using a simple protocol. Various network management issues like naming, delegation, meta-information, user and application network managers, application surrogates have been studied in depth in this approach. Future work includes dealing with disconnectivity, workflow support, customizing interfaces, and replaying of disconnectivity interaction.

UBICOMP

The UBICOMP [144] – short for UBIquitous COMPuting – project being developed at XEROX PARC is a pioneering work in developing the framework for ubiquitous transparent mobile computing. Many hardware computing devices such as the ParcTab and ScratchPad have been built to prototype their research ideas. Various issues like applications, privacy, computational methods, pen devices and low power computing have been implemented as part of the project. UBICOMP lays emphasis on real-time capabilities for multimedia data using packet routing over standard networks with wired and wireless media. Locator services, automatic phone and mail forwarding, frequent user polling have also been incorporated in the UBICOMP project.

Telemedicine Projects

There has been a great surge in the use of mobile telecomputing for medical applications. Patient medical information can be sent from a moving vehicle like an ambulance to a health care professional located at a stationary site (see Fig. 1.3). The advantages of mobile telemedicine include the fact that it can potentially provide very fast diagnosis of the patient's condition thus saving time and resources. Among the many telemedicine projects underway, the NASA/ACTS [85] project is prominent.

Other notable projects include the Olivetti Active Badge project [162], the DATAMAN project [7] at Rutgers University, the Monarch project [88] at Carnegie Mellon University. A good collection of several of the mobile computing projects can be

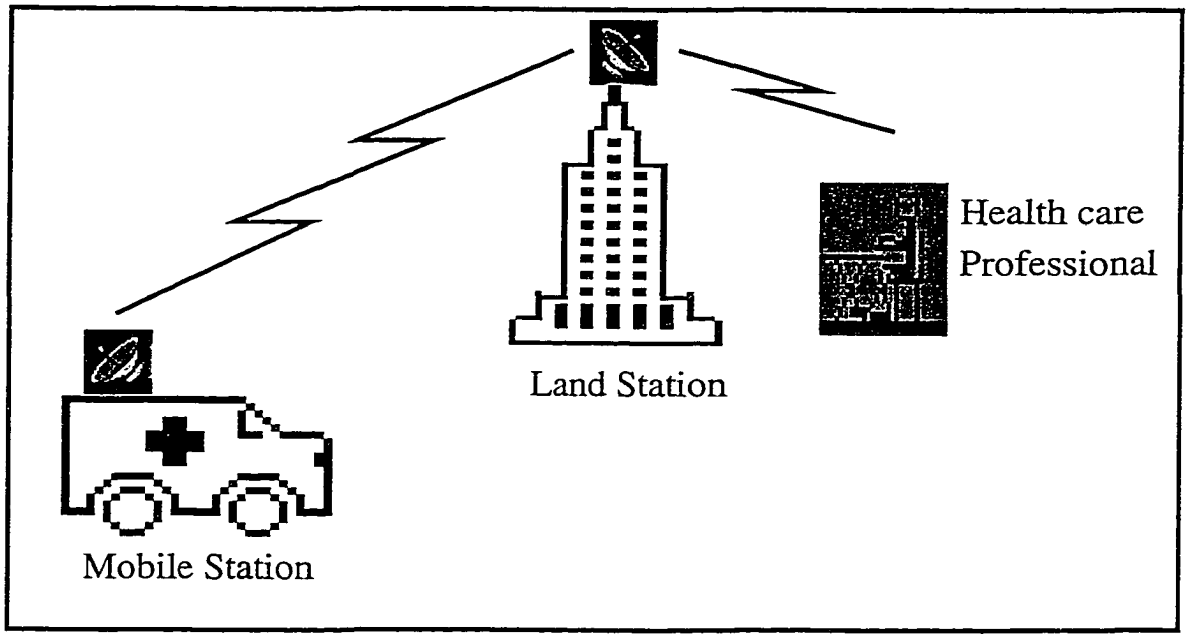


Figure 1.3: A Mobile Telemedicine System

found on the internet are referenced in [117].

1.4 Handoff, Routing and Rerouting in Mobile Networks

In this dissertation, we focus on routing and rerouting issues for mobile wireless and ad-hoc networks. We now take a detailed look at issues concerning routing and rerouting and look at ways to providing solutions to solve the problems arising from routing and rerouting.

Mobile networks consist of mobile hosts – typically a palmtop or a personal digital assistant, and a underlying wired network consisting of base stations and fixed intermediate routers. In addition, the mobile network may contain static hosts

(either independent machines or base stations) which are fixed and do not move. Communication can be either between: two static hosts (Static-Static), a static host and a mobile Host (Static-Mobile), or two mobile hosts (Mobile-Mobile). Each base station has an area of influence called a *cell.* Hosts communicate among each other using the base station and the underlying wired network. Figure 1.1 depicts the architecture of a cellular mobile network system. The figure shows the stationary backbone consisting of mobile hosts and static hosts (base stations).

Mobile hosts move from one cell to another. When a mobile host moves from one cell to another, it registers with the base station of the new cell. Consider that there is a currently a communication session between a source host and a destination host. If one of the host moves from its present cell, the session is interrupted because of the movement. In order of the session to be restarted, a *handoff* needs to take place. Handoff is the process of transferring the control and responsibility for maintaining communication connectivity. Handoff is used by the mobile network to provide the mobile host freedom of motion beyond the cell coverage of a base station. Handoff [14, 155] in mobile and wireless networks has been a topic of research and development for the last few years. Rerouting is the process of routing after the handoff has occurred.

The rerouting problem has been studied extensively in cellular, mobile and wireless networks with respect to connection-oriented networks – including wireless ATM [1, 13, 14, 142, 134, 51, 131], picocellular networks [87], cellular networks [146], wireless LANs [155, 156] and connection-less networks – based on Mobile IP proposals including the Sony VIP routing [154], IETF Mobile IP routing [40], Panasonic Mobile IP routing [161] to name a few. The rerouting problem in LEO (Low Earth

Orbiting) satellites has been studied extensively by Hussein, Yen and Akyildiz [159]. Since a satellite is continuously moving, its coverage area keeps changing too. In order to keep continuous connectivity between a source and a destination ground station, satellites may need to perform "handovers" among one another so that continuity of service is maintained. Handovers can either be *inter-orbit* or *intra-orbit*. The routing and the re-routing algorithms for satellite networks draw inspiration from the routing and re-routing algorithm for terrestrial wireless systems. Among the proposed re-routing algorithms for satellite networks is the Footprint Handover Re-route Protocol (FHRP) [159].

Communication in connection-oriented mobile networks can be either [47]: between two Static Hosts (Static-Static) or between a Static Host and a Mobile Host (Static-Mobile) or between two Mobile Hosts (Mobile-Mobile). Static-Static communication has been studied extensively and the routing algorithms for this type of communication are well known. Static-mobile communication have been studied in the context of rerouting by [13, 14, 47, 87, 156]. It is the last type (mobile-mobile) that has not been explored much in the literature. The problem with the rerouting schemes for connection-oriented mobile networks described in literature is that assume that only one of the parties communicating in a session can be a mobile host (typically the destination) and the other is stationary. Only [47, 102, 87] suggest schemes for mobile host to mobile host communication. These schemes do not provide rerouting that use optimal routing. In addition, these schemes do not look at different types of rerouting strategies [138, 139, 146]. We propose a scheme for performing optimal rerouting in mobile-mobile networks. We also compare and contrast all of the above given mobile-mobile rerouting schemes. We believe, that

this is the first such effort.

Several rerouting strategies which evaluate and optimize various performance metrics have been proposed. For example, Acampora's Virtual Connection Tree routing scheme [1] tries to optimize probability of cell overload (base station handling more connections than its capacity) and the mean time spent in overload while Akyol and Cox's NCNR [14] scheme evaluates metrics including signaling bandwidth, number of signaling messages during handoff and rerouting and user bandwidth allocated for handoff. Toh [155, 156]'s scheme evaluates service disruption time for upstream and downstream traffic, buffer size, cell redirection time, path setup and tear down time among others. Seshan's [146] rerouting scheme evaluate many metrics including, service disruption time, total rerouting time, buffering requirements at base stations and mobile hosts.

1.5 Overview of the dissertation

In this dissertation, we have addressed several problems in the area of routing and rerouting in mobile and wireless networks. We now briefly describe the problems addressed in this dissertation. The first three problems addressed in this dissertation are related to rerouting in connection oriented networks and the final problem is to design an efficient routing algorithm for ad-hoc networks.

1.5.1 Rerouting in Static-Mobile Connections

We provide a framework for comparing rerouting strategies for Static-Mobile connections in connection-oriented networks using appropriate characteristics. The rerout-

ing strategies and earlier attempts at such classification are described in [13, 146, 155, 156]. We differ in our approach from [146, 155, 156] in that we allow for schemes based purely on cell forwarding [81] and virtual trees [1]. We also subsume the classification proposed by Akyol and Cox [13] by including different tree based rerouting schemes and separate schemes for full and partial rerouting. In this sense, our work can be viewed as a generalization of previous works. We analyze and evaluate the performance of rerouting strategies using metrics common to [13, 146, 155, 156] and metrics specialized to specific schemes. We have also abstracted the common handshaking signals from all the rerouting schemes (with and without radio hints - described later) to avoid repetition and also provide a framework to compare these common handshaking signals. We have provided methods to decrease the total rerouting completion time by exploiting parallelism in message exchanges during rerouting. We evaluate the performance of the rerouting schemes using analytical cost modeling.

Our classification of the rerouting schemes are based on the several characteristics including: protocol used for rerouting, whether used for local or wide area networks, type of traffic (time-dependent like audio and video data or throughput-dependent like text files and ascii data or both) and the granularity of cell (macrocells, microcells or picocells) In order to compare and contrast the rerouting schemes, we use a set of common performance metrics including: Rerouting Completion time, Buffering required at the mobile host, Number of nodes involved in handoff and rerouting, Number of user connections established for rerouting, and Degree of parallelism (Number of messages that can be done in parallel and the total number of messages that have to be exchanged sequentially).

We classify the rerouting strategies broadly as: Full, Partial, Tree, Cell Forwarding. Because there are overlaps in the cell coverage of adjacent cells, MH_D gets a radio "hint" before it enters the its cell. Using the radio hint MH_S can signal BS_{old} to inform BS_{new} to set up the required connections in advance. Each of the following schemes can be either with or without a radio hint [14, 146, 155, 156].

Full rerouting involves establishing a new routing path from the BS_{new} base station to the BS_S. Full rerouting schemes perform poorly as they are slow and inefficient. Partial rerouting involves finding the "cross-over" point of the route between BS_S and BS_{old} and the route between BS_S and BS_{new} with a view to increase route reuse. Tree rerouting involves routing using a tree based structure. The base stations in the system form the nodes of the tree. The tree has a special designated base station which forms the root of the tree. Some implementations have multiple trees which form the communication structure. Multicasting is typically performed using this scheme. Tree rerouting can either be to a group (Tree - Group rerouting) or from a single source to a single destination using a virtual tree (Tree - Virtual rerouting) where only one branch of the tree is active at a time. Tree - Group rerouting can either have a static or a dynamic group with which to communicate. The static version of Tree - Group rerouting involves a group consisting of members that do not change over time while in case of a dynamic rerouting the membership of a group may change. Cell Forwarding rerouting involves designating a specialized base station to forward data packets to the MH_D when it moves from a "home" area.

We use message signaling diagrams (the actual set of messages exchanged between different entities involved in handoff and rerouting) to perform analytical cost modeling using the network and the system parameters in order to calculate the

various performance metrics.

1.5.2 Rerouting in Mobile-Mobile Connections

Static-Static communication has been studied extensively and the routing algorithms for this type of communication are well known. Static-Mobile rerouting has been studied in the context of rerouting by [14, 13, 47, 87, 146, 155, 156]. The problem with the rerouting schemes for connection-oriented mobile networks described in literature is that assume that only one of the parties communicating in a session can be mobile host (typically the destination) and the other is stationary. Only Biswas [47], Ghai and Singh [87] and Cellular Telecommunication standard IS-41c [102] suggest schemes for mobile host to mobile host communication. Biswas's [47] strategy uses an already pre-established route between two stationary hosts that houses the mobile agents in-charge of the communication. The only rerouting involves both the source and destination mobile hosts establishing paths to these stationary hosts. The disadvantage of this scheme is that if the mobile hosts keep moving, the path used for communication may be inefficient as the scheme does not dynamically update the paths based on the location of the mobile hosts. The scheme proposed by Ghai and Singh [87] suffers from the same problem. In addition, Ghai and Singh's scheme uses dynamic multicast rerouting only. The architecture of the setup in more complex using a three tier hierarchy (mobile host, base station and supervisory host). In case of cellular telecommunication using the EIA/TIA IS-41 (c) standard, cell forwarding takes place continuously as the mobile host move away. The obvious disadvantage is that the routes used are not optimal. We shall see this in more detail later as we compare all these schemes. In addition, these schemes do not look at

different rerouting strategies known for static-mobile connections. In [138], we have proposed a generic framework for mobile-mobile rerouting allowing unlimited movement by both the source and destination mobile hosts while alleviating the problem of non-optimal paths.

We know of only four techniques [47, 87, 102, 138] to perform rerouting in mobile-mobile connections. In this section, we discuss these techniques in more detail and look at their drawbacks and look closely at the technique that we have proposed [138] in order to alleviate these drawbacks.

We present a distributed algorithm for rerouting in mobile-mobile connections which eliminates the problems described above. In the discussion that follows, we designate one of the mobile hosts as the source mobile host (MH_S) and other as the destination mobile host (MH_D). The algorithm allows communication irrespective of the rerouting technique used is full, partial, cell forwarding or tree based. However, in the case of tree rerouting, we assume that when there is a communication tree rooted at source base station with the destination base stations forming leaves of the tree. Tree routing involves, dynamically forming trees rooted at the current base station of MH_S as MH_S moves. In addition, if MH_D moved out of the tree, either the tree has to be extended to cover the base station that MH_D has moved to, or use cell forwarding to maintain communication between MH_S and MH_D.

We use a simple framework for comparing the performance of all the mobile-mobile rerouting schemes. The reason, we choose this simple scheme (which we explain in detail later) is that the mobile-mobile rerouting schemes are not tied to the type of rerouting (full, partial, cell, tree-based). These rerouting schemes basically concentrate on deciding the end points on the connections and can, in theory,

use any type of rerouting. Our comparison is based on calculating the total rerouting distance, the cumulative connection path length and the amount of resources used as the mobiles move. As seen earlier, the metrics that determine efficiency of rerouting schemes in static-mobile connections are dependent on connection length. For example, the total rerouting path length gives a good estimate of metrics such as throughput, the total rerouting time, the service disruption time and buffering requirements at the base stations.

1.5.3 Effect of rerouting on the performance of wireless TCP

TCP is a one of the most widely used transport layer protocol. In a mobile environment, TCP cannot discriminate packets dropped due to lossy links, intermittent connectivity, handoffs and rerouting with those dropped due to congestion in the network. TCP reacts to packet losses as it would in a wired environment – it drops its transmission window size before retransmitting packets, initiates congestion control or avoidance mechanisms like the slow start algorithm and resetting its retransmission timer [32, 33]. This results in an unnecessary degradation in system performance. In the past, several proposals have been made to alleviate this problem [32, 33]. However, these proposals do not take into account the type of rerouting viz. full rerouting, partial rerouting, tree based rerouting or cell forwarding. Also, these proposals assume that only one end in a session is mobile while the other is fixed.

We study the network performance with TCP under different rerouting schemes with either one end or both the ends in a session being mobile using analytical cost modeling. Various performance metrics like the system throughput, total rerouting

time, service disruption time, amount of buffering required at the mobile hosts and the base stations are calculated for comparative analysis. We propose measures and mechanisms to improve the network performance using TCP under these rerouting schemes.

1.5.4 A Novel Routing Algorithm for Ad-hoc networks

An *ad-hoc* network is characterized by a collection of mobile hosts with frequently changing network topology. In the presence of dynamic network topology and the absence of centralized routing management system, *flooding* from a given source is often the only way to send message units to destinations. We present a distributed algorithm that will maintain a spanning tree of the collection of mobile hosts. Routing from a given source to a destination is performed on this spanning tree. Thus the routing scheme is loop-free and uses a minimal number of control messages to determine the route. We present several optimizations to improve our basic spanning tree construction algorithm. We will also examine properties of the spanning trees constructed and evaluate it for routing and reconstruction efficiencies taking into account demand patterns. Finally, we present analytical cost models for evaluation of worst case bounds and show average case performance of our proposed algorithms.

Due to the increasing number of mobile computers and networking hardware, extensive work has been done in communication protocols in the mobile environment using a fixed, backbone network infrastructure. There are situations when such a backbone network may not be present. Examples of this include scenarios such as - when a group of workers are to be deployed quickly during a natural disaster or a group of business delegates assembled in a lecture hall. It is desirable for mobile

hosts in such situations to form a temporary network without the aid of centralized administration or an established network backbone. This temporary network is called an *ad-hoc* network. we propose a novel routing protocol that is reactive but uses considerably less number of messages. We achieve this by maintaining a forest of spanning trees. We believe our approach is practical and has less time, space and message complexities when compared with the above algorithms.

The basic idea of the algorithm is to construct a set of dynamic trees (also known as a dynamic forest) $T_1, T_2, ..., T_k$ where each T_r, at a given time, has nodes (mobile hosts) $h_1, h_2, ..., h_p$ such that any node h_i can communicate to a host h_j. We assume that each node h knows the IDs of the neighboring nodes within its direct transmission range and it is denoted by $N(h)$. In this manner, a node can communicate to any one of its neighbors directly. On the other hand, if a node wants to send a message to another node which is beyond its direct range then it would try to use other nodes as routers. The key ingredient of the algorithm is to dynamically readjust the direct connections between the nodes as the hosts move around. In simple graph-theoretic terminology, this means that a host h_i can try to communicate to a host h_j by sending its message via the tree edges and in some special situations as described later, using non-tree edges.

1.6 Organization of the dissertation

The rest of the dissertation is organized as follows. Chapter 2 describes our proposed framework for comparative analysis of rerouting schemes for connection-oriented mobile networks. Chapter 3 elaborates on our proposed distributed rerouting algorithm

for connections with mobile ends. Chapter 4 presents our approach to evaluate the performance of Wireless TCP scheme in conjunction with different rerouting schemes. In Chapter 5, we describe an efficient routing algorithm for ad-hoc networks. Finally, in Chapter 6, we present our conclusions and future work.

We have tried to make each chapter complete and self-contained as much as possible. For each chapter, we introduce the scope and the motivation of the chapter at the beginning of the chapter and we also summarize the results and our contributions at the end of each chapter.

Chapter 2

Rerouting in Static-Mobile Connections

2.1 Introduction

In this chapter, we survey, classify, analyze and evaluate all known rerouting techniques for connection oriented networks for Static-Mobile connections. Connection-oriented networks provide guarantees on performance (even under congestion) needed for delivery of multimedia data on mobile hosts. Hence the motivation for concentrating on connection-oriented networks in this work.

The rest of the chapter is organized as follows: In this section, we continue to explore the rerouting process in more detail. We study the characteristics for comparison and classification of rerouting. We classify various rerouting schemes in four major categories and finally we survey the related work in detail. In Section 2.2, we analyze and evaluate the various classes. Each class is explained in detail

including the protocol for rerouting, advantages and disadvantages of the class, various examples and variations of strategies in this class. Section 2.3 evaluates the above mentioned performance metrics including specialized metrics for each scheme using analytical cost modeling. Section 2.4 briefly describes the comparison of all the individual rerouting schemes based on the performance metrics. In Section 2.5, we suggest schemes to improve rerouting schemes by using parallelism in protocol messages. Section 2.6 presents conclusions and future work.

2.1.1 Characteristics for Comparison and Classification of Rerouting

We use a set of characteristics in order to compare and classify various static-mobile rerouting schemes. Our classification of the static-mobile rerouting schemes are based on the following characteristics:

1. The characteristics of the system:

    - Protocol used for rerouting: The protocol could be either cell forwarding, full rerouting, partial rerouting, or tree rerouting.
    - Whether used for local or wide area networks.
    - Whether the scheme is tailored for ATM based networks.
    - Type of traffic: Traffic can either be time-dependent (audio and video data) or throughput-dependent (text files and ascii data) or both.
    - Granularity of cells: Cells can be either macrocells, microcells or picocells [1, 155, 156].

2. A set of common performance metrics used to *evaluate* the rerouting strategy - like:

   - Service Disruption Time ($T_{disrupt}$).
   - Rerouting Completion Time ($T_{complete}$).
   - Buffering required at the Mobile Host (B_{mh}) .
   - Buffering required in Base Station for uplink (B_{bsup}).
   - Buffering required in Base Station for downlink (B_{bsdn}).
   - Number of messages exchanged during handoff (N_h).
   - Number of messages exchanged during rerouting (N_r).
   - Number of nodes involved in handoff and rerouting (N_n).
   - Number of user connections established for rerouting (N_c).
   - User bandwidth allocated for handoff and rerouting (B).
   - Whether the scheme is proposed for WANs or LANs (W/L).

3. Specialized metrics for each class of rerouting scheme. For example, in the case of partial rerouting, the new route shares part of the old route. Thus, one of the important metrics for this scheme is the partial re-use efficiency (η_{part}) which is defined as the ratio of the length of the route that is common to both the old and the new route and the length of the new route.

4. Implementations: Schemes and protocols from literature that exemplify the rerouting scheme (See Figure 2.2).

5. Advantages and disadvantages of the rerouting strategy.

2.1.2 Classification of rerouting schemes

Rerouting in mobile environments occurs as result of a handoff. When a mobile host (MH) moves from the region of influence (cell) of a base station (BS) to another, a handoff is said to have taken place. In order to maintain communication connectivity, packets have be re-routed to and from the MH. This process of re-establishment of a route is called as rerouting. Figure 2.1 depicts the rerouting process. The source MH namely MH_S is in a session with the destination MH namely MH_D when it migrates from the cell of BS_{old} to BS_{new} after performing a handoff. The route between the BS_D and BS_{old} and BS_{new} may be the same, partially same or completely different.

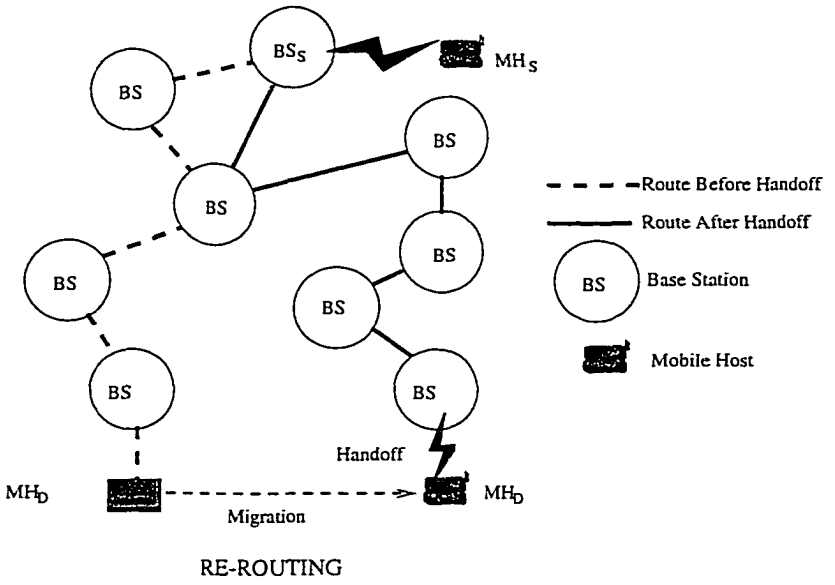


Figure 2.1: The rerouting process

We classify the rerouting strategies broadly as:

1. Full

2. Partial

3. Tree

4. Cell Forwarding

Because there are overlaps in the cell coverage of adjacent cells, MH_D gets a radio "hint" before it enters the its cell. Using the radio hint MH_S can signal BS_{old} to inform BS_{new} to set up the required connections in advance. Each of the following schemes can be either with or without a radio hint [14, 146, 155, 156].

Full rerouting involves establishing a new routing path from the BS_{new} base station to the BS_S. Full rerouting schemes perform poorly as they are slow and inefficient. Examples of such schemes are Full Re-Establishment (without hint) and Full Re-Establishment (with hint) [146].

Partial rerouting involves finding the "cross-over" point of the route between BS_S and BS_{old} and the route between BS_S and BS_{new} with a view to increase route reuse. Examples of these schemes include Incremental Re-Establishment (without hint) and Incremental Re-Establishment (with hint) [146], Incremental Re-Establishment (without hint) and Incremental Re-Establishment (with hint) [155], and NCNR [14].

Tree rerouting involves routing using a tree based structure. The base stations in the system form the nodes of the tree. The tree has a special designated base station which forms the root of the tree. Some implementations have multiple trees which form the communication structure. Multicasting is typically done using this scheme. Tree rerouting can either be:

- To a group (Tree - Group rerouting) as described in Multicast Re-Establishment (with and without hint) [146] and the Picocellular Network Architecture [87]

- From a single source to a single destination using a virtual tree (Tree - Virtual rerouting) where only one branch of the tree is active at a time. Such a scheme is explained in the Virtual Tree scheme [1] and the SRMC scheme [164].

Tree - Group rerouting can either have a static [146, 164] or a dynamic group [87] to communicate with. Static rerouting involves a group consisting of members that do not change over time while in case of a dynamic rerouting the membership of a group may change.

Cell Forwarding rerouting involves designating a specialized base station to forward data packets to the MH_D when it moves from a "home" area. Such schemes include the ones described in the Adaptive Routing scheme[81] and the BAHAMA scheme [79].

Figure 2.2 shows the classification scheme with examples. Rerouting schemes can be either full, partial, tree and cell forwarding. Tree rerouting can either use virtual (static) tree based rerouting or dynamic multicast based tree rerouting. Each of these schemes may or may not radio hints. Figure 2.2 gives examples for each of the rerouting schemes.

2.1.3 Related Work

In this section, we briefly describe other work in the area of comparative analysis of handovers and rerouting. Specifically, we discuss the work done by Toh [157], Ramanathan and Steenstrup [141], Bush [51], Cohen and Segall [60], Ramjee et al. [142], Song et al. [148], Mishra and Srivastava [134] and Hopper et al. [131].

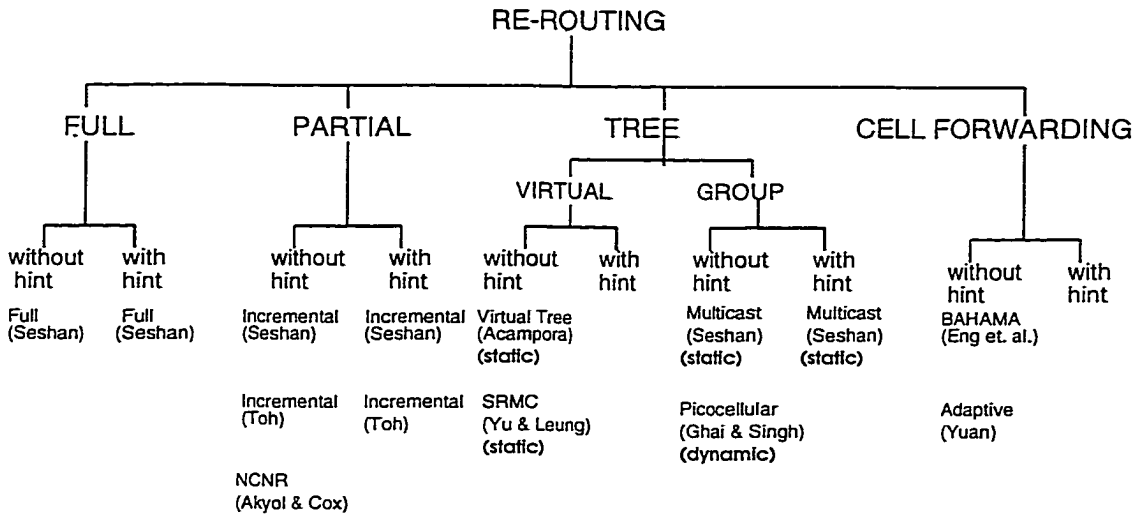


Figure 2.2: Classification of rerouting strategies

Toh [158]

Toh [158] explains how handovers in multicast connections can be achieved irrespective of the kind of multicast tree (source-based, server-based or core-based) used in the wireless ATM environment. Toh proposed solutions to handle handover and rerouting for both multicast and unicast connections without categorizing rerouting strategies. His scheme also does not consider pure cell-forwarding schemes. The rerouting algorithm used in Toh's approach is partial rerouting using a crossover discovery algorithm. Toh demonstrates how this handoff and rerouting scheme can be used for both unicast and multicast connections using either distributed or centralized connection management. Toh's work considers partial rerouting with static-mobile connections.

Toh's work does not consider different kinds of rerouting (full, cell forwarding

and tree-based) schemes for comparison and also does not deal with mobile-mobile connections. Toh's analysis is restricted to partial rerouting only. The analysis does not calculate the number of messages exchanged during handoff (N_h), number of messages exchanged during rerouting (N_r), number of nodes involved in handoff and rerouting (N_n), number of user connections established for rerouting (N_c) and u user bandwidth allocated for handoff and rerouting (B).

Ramanathan and Steenstrup [141]

Ramanathan and Steenstrup [141] make an extensive survey of routing techniques for cellular, satellite and packet radio networks. They have surveyed various methods for location tracking and location update strategies in PCS, internet and packet radio networks. They categorize different types of handoffs in cellular telecommunication networks. These include mobile-controlled handoff (the mobile chooses its new base station based on relative signal strength), network controlled handoff (the mobile host's current base station decides the occurrence of a handoff based the signal strength from the mobile host), mobile-assisted handoff (the mobile host's current base station requests the mobile host to inform it the signal strengths from several nearby base stations after measurement and then decides when a handoff has occurred in consultation with the mobile switching center) and soft handoff (the mobile host may be affiliated to multiple base stations with approximately equal signal quality).

However, the survey does not consider rerouting specifically. Also, the authors do not compare and contrast the techniques using any performance analysis.

Bush [51]

Bush et al. [51] classifies handoff schemes for Mobile ATM networks. The classification is specific to handovers (and not rerouting) for connection oriented networks. These include pivotal connection handoff (a specific base station is chosen a pivot to perform handoff), IP mobility based handoff (handoffs require IP packet forwarding using mechanisms such as loose source routing), handoff tree (a pre-established virtual circuit tree [1] is used to automatically detect handoffs in a wireless ATM environment).

The classification is specific to handovers (and not rerouting) for connection oriented networks. The analysis does not calculate several performance metrics calculated by us including the number of messages exchanged during handoff (N_h), number of messages exchanged during rerouting (N_r), number of nodes involved in handoff and rerouting (N_n), number of user connections established for rerouting (N_c) and u user bandwidth allocated for handoff and rerouting (B).

Cohen and Segall [60]

Cohen and Segall [60] describe a scheme for connection management and rerouting in standard (not wireless/mobile) ATM networks. Their rerouting is a NNI protocol which can be invoked when an intermediate link or node in the virtual path fails. The protocol reroutes all the effected to other alternate virtual path.

The authors do not compare and evaluate different types of rerouting schemes. In addition, the evaluation is not for wireless/mobile networks.

Ramjee et al. [142]

Ramjee et al [142] have performed experimental performance evaluation of five types of rerouting protocols for wireless ATM networks. Their rerouting protocols are primarily based on cross over switch based rerouting [158] using mobile-directed handoff. In order to perform rerouting in an ATM environment, the ATM switch at the cross over point must change the appropriate entry in the translation table. This involves dismantling the old entry (break) and installing the new entry (make). Their evaluation includes rerouting schemes: make-break (make followed by break), break-make (break followed by make), chaining (cell forwarding from the old base station to the new base station), make-break chaining (make-break with chaining) and break-make chaining (break-make with chaining). The authors classify the handoff schemes as:

- Optimistic: These schemes rely on a simple and fast handoff and assume an optimistic view towards disruption of traffic. The virtual circuit tree [1] is an example of this scheme.

- Ordered: These schemes provide in-order delivery of data with no loss during handoffs. Examples of this type of handoff scheme are the full re-establishment, incremental re-establishment, and multicast-based re-establishment [146].

- Predictive: These schemes try to predict the movement of the mobile host and multicast the data to the group of cells where the mobile host is likely to move into. The picocellular architecture [87] uses this kind of handoff mechanism.

- Chaining: These schemes simply extend the connection from the old base station to the new base station. BAHAMA [79] and [134] use this scheme.

These schemes essential perform cell forwarding from the old base station to the new base station.

Comparative evaluation by the authors does not consider mobile-mobile connections and uses only disruption time and handoff latency as metrics for comparison.

Song et al. [148]

Song et al. [148] define five kinds of rerouting schemes for connection-oriented networks. These are:

- Connection-Extension Rerouting: This rerouting is the same as cell-forwarding rerouting.
- Destination-Based Rerouting: In this scheme, the rerouting point is predetermined at the connection time. This is similar to connection-extension rerouting except the that the rerouting base station is the a predetermined base station and not necessarily the old base station.
- Branch-Point-Traversal-Based Rerouting: This rerouting is the same as partial rerouting:
- Multicast-Join-Based Rerouting: This rerouting is the same as Tree-group Rerouting.
- Virtual-Tree-Based Rerouting: This rerouting is the same as Tree-Virtual Rerouting.

Song et al. propose a general framework for perform rerouting based on connection-information. They do so by maintaining information about the connection identifier

and router address and the rerouting base station is chosen as the closest base station to the new base station on the existing old path. The authors compare the rerouting schemes based on connection distance, moving distance, reroute distance, reroute optimality, size of connection information and size of rerouting information and rerouting time.

However, the authors do not compare buffering requirements, disruption time. Also, their comparative evaluation is does not involve analytical modeling using actual real-world values for network and system parameters. Also, the authors do not consider mobile-mobile connections for evaluation.

Mishra and Srivastava [134]

In [134], Mishra and Srivastava classify rerouting in an ATM environment as:

- Extension: Same as cell forwarding or chaining.
- Extension with loop removal: Specialized case of cell forwarding with removal of any possible path loops caused by chaining.
- Total Rebuild: Same as full rerouting.
- Partial Rebuild: Same as partial rerouting.There are two variants in this scheme (fixed or dynamically chosen cross over point).
- Multicast to Neighbors: Same as Tree-group rerouting.

The authors compare and contrast these schemes for CBR and TCP sources. The authors compare using throughput, packet delay and loss rate. The work does

not consider mobile-mobile connections and use only a small subset of the metrics that we have used for comparative evaluation purposes.

Hopper et al. [131]

In [131], Hopper et al. classify rerouting in the wireless ATM LANs as:

- Virtual Connection Tree: Same as Tree-virtual rerouting.
- Path Rerouting: Same as partial rerouting.
- Path Extension Scheme: Same as cell forwarding.

In this work the authors suggest a COS (cross over switch) rerouting (partial rerouting) to meet a set of requirements including minimal disruption and cell loss, no reordering and other QoS metrics. The COS is fixed and predetermined. In case of excessive and inefficient handoffs, the COS can move dynamically. The authors compare their proposed scheme with schemes proposed in [1, 13, 79, 10, 81].

The authors do not compare all the rerouting schemes proposed by them. They analyze only the COS rerouting algorithm. Also, the use only minimal disruption and cell loss as performance metrics. The analysis does not calculate several performance metrics calculated by us including the number of messages exchanged during handoff (N_h), number of messages exchanged during rerouting (N_r), number of nodes involved in handoff and rerouting (N_n), number of user connections established for rerouting (N_c) and u user bandwidth allocated for handoff and rerouting (B).

From the previous related work described above, we see that our classification encompasses all the proposed rerouting schemes. In this sense, our work can be

viewed as a generalization of previous works. We also use a large set of performance metrics for comparison between all the rerouting schemes. Our contribution in this work is six-fold:

1. We have provided a comprehensive framework for comparing rerouting strategies for connection-oriented networks using appropriate performance metrics. We also subsume the classification proposed by various authors in [13, 51, 148, 131, 134, 142, 146, 155, 158], that is every rerouting scheme that can be classified under any of the other classification schemes can also be classified under our proposed classification scheme.

2. We analyze and evaluate the performance of rerouting strategies using a large array of metrics *including all the ones proposed by the above mentioned authors.*We also propose and evaluate metrics specialized to specific schemes. *The calculation of special metrics is unique to our comparison and classification.*

3. We have also abstracted the common handshaking signals from all the rerouting schemes (with and without hints) to avoid repetition and also provide a framework to compare these common handshaking signals.

4. Also, we have compared and contrasted *actual implementations* of the the rerouting protocols (See Figure 2.12). *None of the related work (described above) has done such a comparison.*

5. We compare and contrast the performance of rerouting in mobile-mobile connections. This is the first such attempt according to best of our knowledge.

6. We have provided methods to decrease the total rerouting completion time by exploiting parallelism in message exchanges during rerouting.

2.2 Analysis and evaluation of classes of rerouting schemes

In this section, we describe for each class of rerouting – the basic protocol, and its different implementations, advantages and disadvantages. Self descriptive figures are provided to aid in the explanation. In the descriptions of the rerouting schemes, we assume the following:

1. There is coverage overlap between cells.
2. An MH communicates with only BS at a time.
3. Each MH can measure the radio signal strength of the base stations in order to know its entrance into a new cell.
4. Handoff and rerouting is initiated by the destination mobile host – MH_D.
5. The source mobile host – MH_S is assumed to be stationary during the rerouting process.

2.2.1 Common handshaking signals for rerouting schemes

There are certain handshaking signals during rerouting that are common to all the rerouting schemes. We have abstracted out these signals in order to avoid repetition in all the rerouting schemes. Using the framework that we describe below, we

can selectively compare the rerouting schemes - with or without these handshaking signals. The handshaking protocol is assumed to have been completed before the actual rerouting protocol. We describe these signals for rerouting schemes - with and without hints, separately. In case of schemes without hints, the handshaking protocol is the same for all the rerouting schemes. However, this protocol varies for schemes that use hints. We now describe briefly these handshaking schemes.

Without hints

Handshaking protocol for all rerouting schemes without hints consists of the following protocol as shown in Figure 2.3.

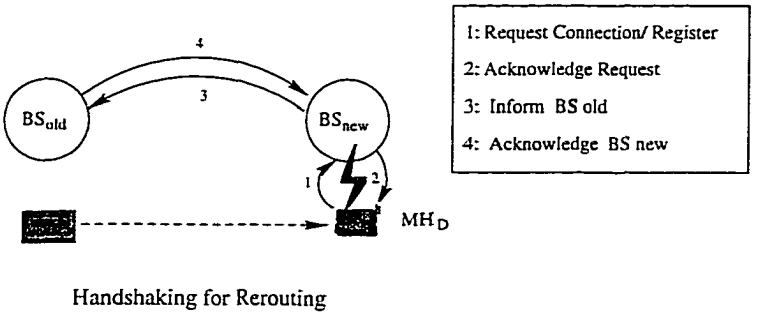


Figure 2.3: Handshaking for rerouting without hints

1. MH_D enters the new cell and request a connection with BS_{new} after identifying itself. MH_D informs BS_{new} the identity of BS_{old} and its present connections.

2. BS_{new} acknowledges MH_D request. MH_D can continue any transmissions.

3. BS_{new} requests BS_{old} to forward MH_D's data to itself. The message also requests that it be allowed to be the "anchor" for MH_D's transmission.

4. BS_{old} acknowledges and grants permission to BS_{new}. After this, BS_{new} begins forwarding MH_D's transmitted data to MH_S through BS_{old}.

With hints

Handshaking protocol for rerouting with hints is dependent on the rerouting scheme. Only the partial rerouting scheme has a different handshaking protocol as shown in Figure 2.4. In the schemes other than partial rerouting, the handshaking protocol is as follows:

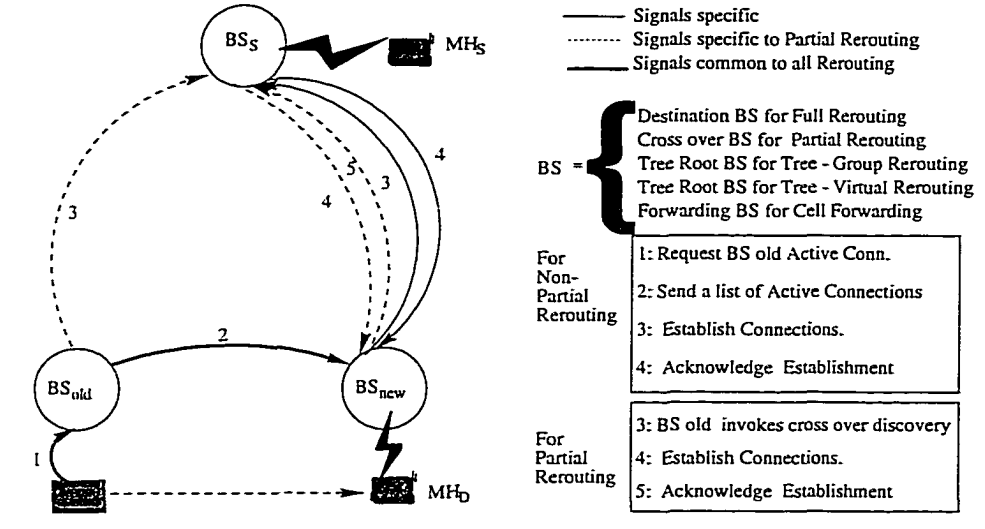


Figure 2.4: Handshaking for rerouting with hints

1. MH_D requests BS_{old} to send a list of active connections to BS_{new}.

2. BS_{old} sends the list to BS_{new}.

3. BS_{new} establishes the connections with BS.

4. BS acknowledges the establishing of the connections.

2.2.2 Full rerouting

Full rerouting can occur with or without hints. We describe in this and the subsequent subsections the generic rerouting schemes without hint. Detailed protocol descriptions of all rerouting schemes (with and without hints) are explained in [139]. Figure 2.5 and Figure 2.6 describe respectively the protocol of a generic Full rerouting (without hints) and Full rerouting (with hints).

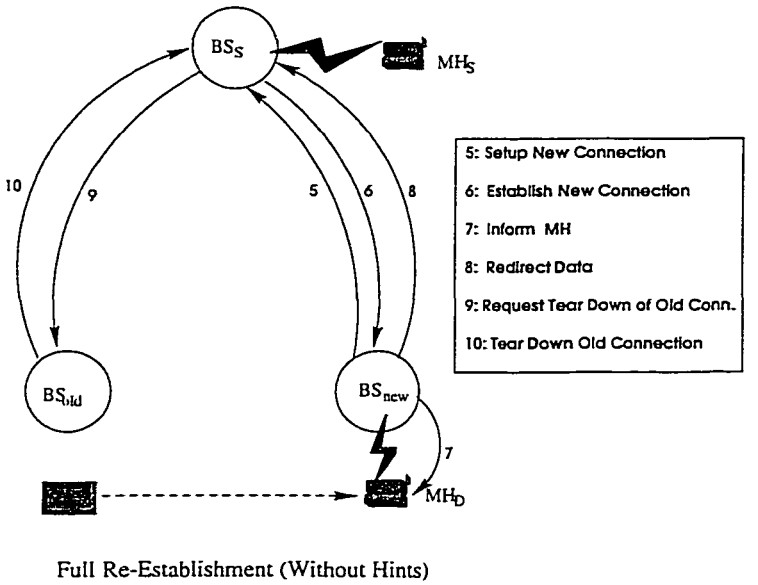


Figure 2.5: Full rerouting without hints

Implementations

The implementations of Full rerouting are described by Seshan [146]. Seshan describes full re-establishment schemes with and without hints. The generic full rerouting explained above is the same as Seshan's implementation. The source (a video server in the implementation) is assumed to be fixed while the destination is the mobile host MH. The mobile host moves from the old base station BS_{old} to the new

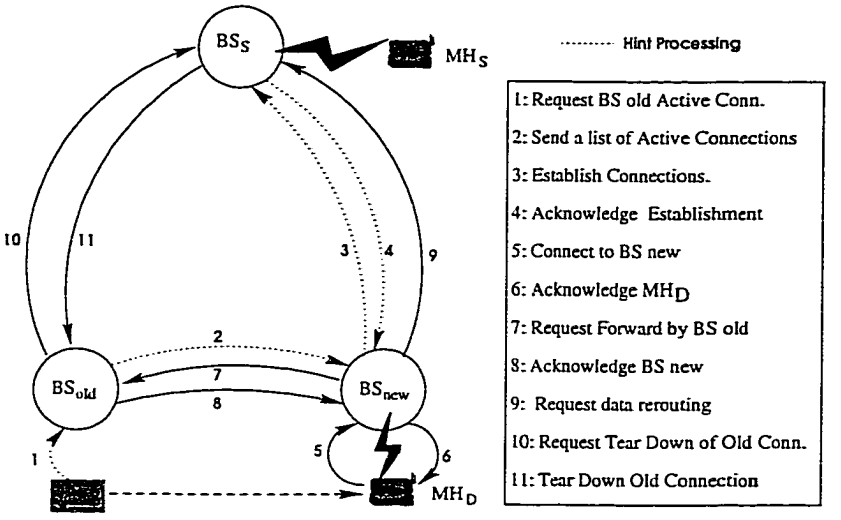


Figure 2.6: Full rerouting with hints

base station BS_{new}. The rerouting involves finding a new route from BS_{new} to the source.

Advantages and disadvantages

Full rerouting is the simplest and naive among all the schemes. However it is inefficient and slow. Full rerouting with hints performs much better than Full rerouting without hints.

Special Metrics

Special Metrics are those that are applicable specifically to this routing scheme. We can use special metrics to compare similar rerouting schemes. We will study special metrics for each rerouting schemes separately. With respect to full rerouting, we look at two special metrics namely, old connection tear down time and new connection

set up time.

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{dest} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

2. New connection setup time (T_{new}): The total time required from the time MH_D informs BS_{old} to send a list of active connections to the time when the BS_{dest} confirms the establishment of the connections.

2.2.3 Partial rerouting

Partial rerouting involves maximizing the route reuse. Once the "cross-over" point is found using a cross-over point discovery algorithm like the ones described [155, 156]. We assume that BS_{cross} is the base station at the cross-over point. Figure 2.7 describes the protocol of a generic Partial rerouting (without hints).

Implementations

There is many variations in implementation of Partial rerouting. Seshan [146] gives implementation - called Incremental Re-Establishment - with and without hints. The protocol of Seshan's implementation is the same as the generic protocol described above. The protocol does not specify the cross-over discovery mechanism.

Toh [155] also gives an implementation called as Incremental Re-Establishment (with and without hints). Toh's implementation is for wireless local area networks. Toh's strategy for rerouting (without hints), unlike the generic Partial rerouting (without hints) assumes that the wireless link between the MH and BS_{old} has failed

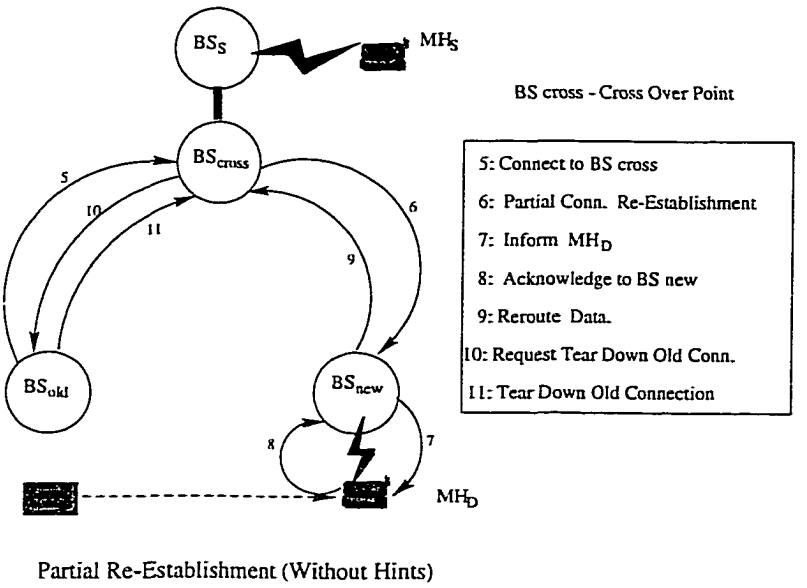


Figure 2.7: Partial rerouting without hints

resulting in the unavailability of hints. In the Toh's Incremental Re-Establishment (without hints) protocol, BS_{old} does not acknowledge the MH (Message 2 in the generic protocol). Also, Messages 3 and 4 are absent and there is cell loss as BS_{new} does not request BS_{new} to forward cells. In addition, there are no explicit messages to tear down the old connection (Messages 10 and 11). In case of Toh's Incremental Re-Establishment (with hints), BS_{new} invokes the cross-over discovery algorithm (Message 3) unlike BS_{old} as described in the generic Partial rerouting (with hints) scheme. Also, Messages 8 and 9 intended for cell forwarding from BS_{old} to BS_{new} are absent resulting is cell loss. Toh describes many algorithms for discovery of the cross-over switch.

Akyol and Cox's [14, 13] strategy - called as NCNR rerouting - performs partial rerouting by choosing the cross-over discovery as the nearest common neighbor of the BS_{old} and BS_{new}. Their scheme considers whether there is a direct link between

BS_{new} and BS_{old} and whether the traffic is time-dependent (e.g. voice, video) or through-put dependent (e.g. data).

It should be noted that the performance of the Partial rerouting is strongly dependent on the performance of the cross-over discovery algorithm.

Advantages and disadvantages

Partial rerouting tries to maximize the re-use of the resources – connections, paths and bandwidth. Since the performance of rerouting is directly dependent on the length of the new path to be established, Partial rerouting is expected to perform better than the Full rerouting. In the worst case, the performance of Partial rerouting is the same as that of Full rerouting. Also, this scheme has the onus of executing the cross-over discovery algorithm.

Special Metrics

With respect to partial rerouting, we study several performance metrics including old connection tear down time, new connection setup time, the time required to invoke cross-over discovery algorithm, the time required to actually discover the cross-over switch and the efficiency of path reuse.

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{cross} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

2. New connection setup time (T_{new}): The total time since the MH_D requests a connection with BS_{new} till the confirmation of the establishment of the new connection.

3. Time required to invoke cross-over discovery algorithm (T_{cross}): The total time in the rerouting process till the cross-over switch discovery algorithm has been invoked.

4. Time to discover the cross-over switch ($T_{discover}$): The total time for finding the cross-over switch after invoking the cross-over discovery algorithm. This term depends on the algorithm used.

5. Partial Re-Use Efficiency (η_{part}): The fraction of the new path that has been re-used.

2.2.4 Tree rerouting

Tree routing involves setting up and using a tree with base stations as nodes to communicate to a either multicast to a group or a single base station using a virtual tree. Let BS_{root} be the root of the communication tree.

Tree - Group

Figure 2.8 describes the protocol of a generic Tree - Group rerouting (without hints). In this case, there is a dynamic multicast tree built that is used to multicast data to a group base stations. The base station BS_{root} is the root of the multicast tree. The messages used to perform rerouting are described in this figure.

Tree - Virtual

Figure 2.9 describes the protocol of a Tree - Virtual rerouting (without hints). Let BS_{root} be the root of the virtual tree which connects a group of base stations. There

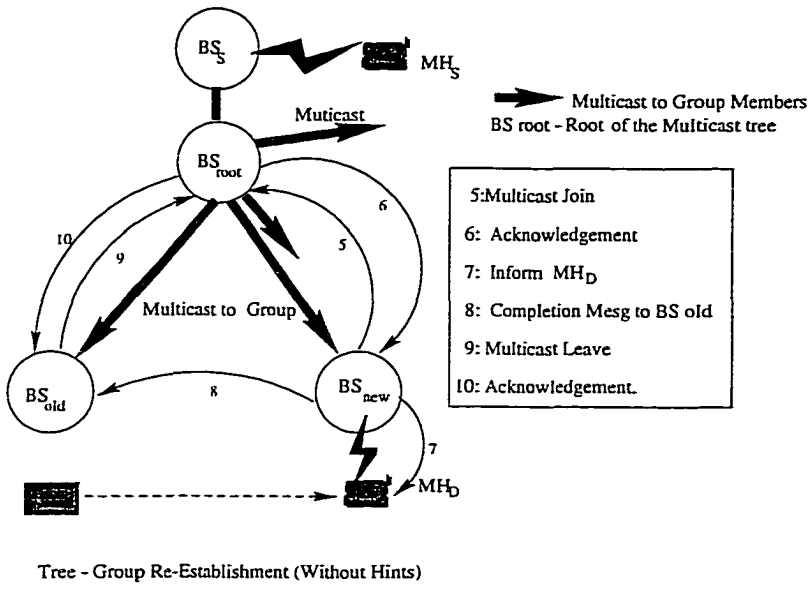


Figure 2.8: Tree - Group rerouting without hints

are no Tree - Virtual rerouting algorithms (with hints) described in literature. However, it is easy to conceive such a class. We describe the messages shown in the figure. The most important aspect of this class of rerouting is that one path (from the root of the tree to the leaves of the tree) is active at a time unlike the Tree - Group rerouting. We assume that the virtual tree is established statically and in place before the commencement of the rerouting process.

Implementations

Acampora [1] describes a rerouting scheme using virtual connection trees. A Virtual connection tree is a collection of base stations and wired switching nodes and connecting links. Each virtual connection tree which is statically built prior to rerouting, has a fixed root node and the leaves of the tree are base stations. For each mobile connection, the tree provides a set of virtual connection numbers in each direction

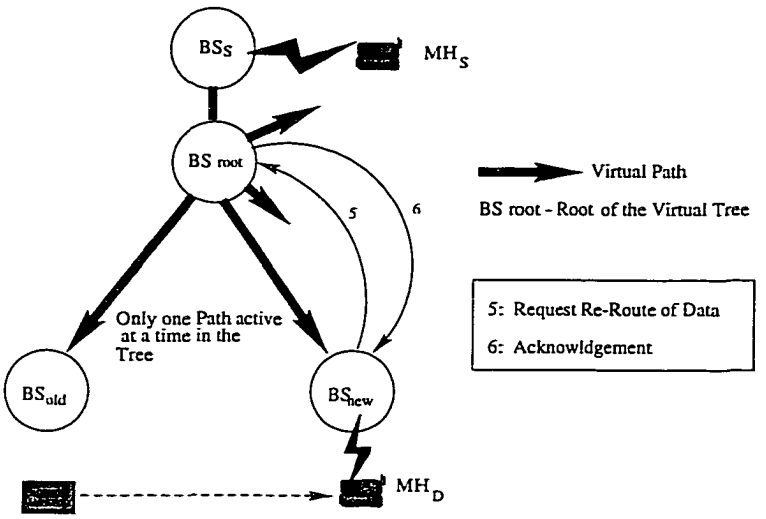


Figure 2.9: Tree - Virtual rerouting without hints

– each associated with a path from a leaf and root. When a mobile host that is already in the tree wishes to handoff to another base station (in the same tree), it starts to transmit ATM cells with the connection number assigned for use between itself and the new base station. The cells flow along the fixed path between the new base station and the root. Appropriate translation tables are maintained the nodes of the tree to understand and implement handoff and the consequent rerouting.

Seshan [146] describes a scheme for Tree Group rerouting called Multicasting Re-Establishment. The protocol of the generic Tree - Group rerouting described earlier is based on Seshan's proposed scheme. Seshan's scheme includes strategies for multicasting rerouting with hints and without hints. Seshan's scheme does not address the issue of how the members of the groups are decided.

Ghai and Singh [87] describe a Tree-Group rerouting scheme based on dynamic grouping of base stations based on the mobility characteristics of the MH. The

scheme is for a picocellular network with a three tier hierarchy of cells. The scheme does not take advantage of hints to aid in rerouting.

Advantages and disadvantages

Tree - Virtual rerouting is the fastest and works well if enough resources are present. However, there is an onus of setting multiple virtual connections. This in turn, requires the prediction of handoff events and pre-established routing. Data loss is still possible in this scheme. Deciding the members of a virtual tree is a difficult problem to solve satisfactorily. This scheme too is prone to data loss. In addition, this scheme is not easily scalable.

Tree - Group rerouting's performance is similar to that of Tree - Virtual rerouting. In this scheme, the data is multicast to a group of base stations.The membership of the group can either be static or dynamic. The advantage of the scheme is that, if there is no crunch in the resources, this scheme is extremely fast and efficient. The chances of data loss is the least among all the rerouting schemes owing to the very nature of multicasting. The requirement of data forwarding from BS_{old} and BS_{new} is not present unless the MH moves out of the group. However, the obvious disadvantage of this scheme is requirements of large resources. Also part of the bandwidth is wasted because though the data is sent to a group, some of them may have to discard it as the MH may not arrive in their respective cells.

Special Metrics

For tree - virtual rerouting, we study special metrics including virtual tree setup time, virtual tree tear down time. For tree - group rerouting, the special metrics

include multicast tree setup and tear down time, the number of nodes involved in the multicast tree and the time for leaving and joining the multicast tree.

For Tree - Virtual rerouting:

1. Virtual Tree setup time ($T_{vtree-setup}$): The time required for setting up the virtual tree which includes the time to choose the root, broadcast the information to all the nodes of the tree and for all the nodes to join the tree.

2. Virtual Tree tear down time ($T_{vtree-tear}$): The time required to tear down the virtual tree.

3. Number of Nodes in Virtual Tree (N_{vtree}): N_{vtree} is determined statically and remains fixed.

For Tree - Group rerouting:

1. Multicast Tree setup time ($T_{mcast-setup}$): The time required to set up the multicast tree.

2. Multicast Tree tear down time ($T_{mcast-tear}$): The time required to tear down the multicast tree.

3. Number of Nodes in the Multicast Tree (N_{mcast}): N_{mcast} depends on the algorithm in question. In static algorithms, this value is fixed and remains constant while in algorithms that dynamically decide the number on-the-fly, the number changes dynamically.

4. Multicast Join/Leave Time ($T_{mcast-jn}$, $T_{mcast-lv}$):

   The time required for node of the multicast tree to join or leave the tree.

2.2.5 Cell Forwarding

Cell Forwarding involves using a specialized base station as an "anchor". Let BS_{fwd} be the anchor. Cell Forwarding is used in many rerouting schemes implicitly. Full rerouting and Partial rerouting use specialized Cell Forwarding schemes. The "anchor" in all these cases is BS_{old}. Figure 2.10 describes the protocol of a generic Cell Forwarding rerouting (without hint). There are no hints based cell forwarding schemes described in literature though such a scheme can be easily conceived.

In the remaining part of the dissertation, when we refer to cell forwarding, we assume that the anchor is the old base station. Thus, in this case data is simply forwarded from the previous destination base station. In that case, we simply do not need to any signaling other than the regular handshake (without hints). In other words signaling steps 5 - 10 do not exist. This way, we avoid the overhead of setting up the new path and tearing down the old path.

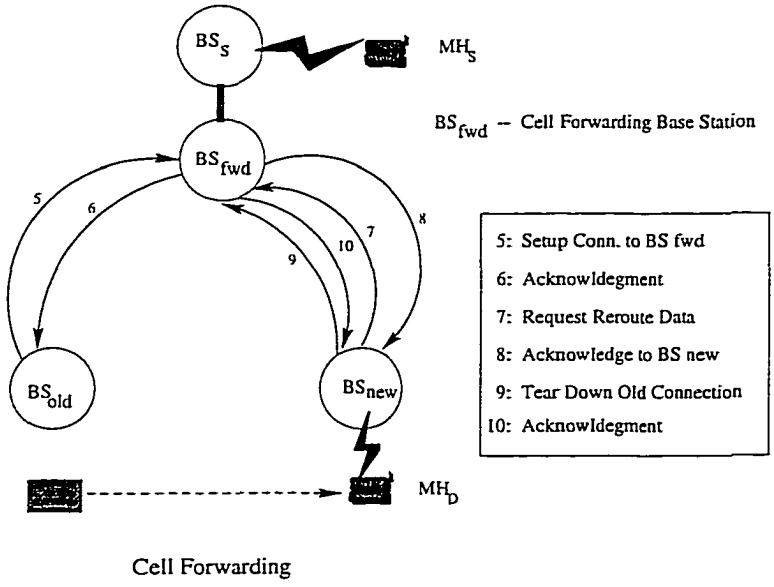


Figure 2.10: Cell Forwarding rerouting without hints

Implementations

Cell Forwarding is used in variations is almost all rerouting schemes. In most cases the old base station BS_{old} acts as the anchor. Yuan [81] describes a scheme using a specialized "anchor" to perform the cell forwarding.

Advantages and disadvantages

Cell Forwarding has the advantage of being a very simple scheme which also preserves the cell sequence. The disadvantages are that the scheme is slow, inefficient and is prone to data loss. In addition, for a fast moving user with a small coverage area, cell forwarding exacerbates the inherent inefficiency of the scheme.

Special metrics

In case of cell forwarding, we study the old connection tear down and new connection setup time and the time to establish the cell forwarding path.

1. Old connection tear down time (T_{tear}): This is true only if there is a specialized anchor used for forwarding data.

2. New connection setup time (T_{new}):This is true only if there is a specialized anchor used for forwarding data.

3. Time to establish path between BS_{old} and BS_{fwd} ($T_{cellfwd-path}$):

2.3 Calculation of performance metrics

2.3.1 Network parameters

In this section, we use analytical cost modeling to evaluate the performance metrics based on the Seshan's [146] and Toh's work [155, 156]. Our cost modeling first calculates the time required for each step in the protocol of the rerouting scheme. These individual times are used to calculate the proposed metrics. We describe the calculation of the common metrics for one of the schemes namely the Full rerouting (without hints). The calculations for the other schemes are done similarly.

We consider the following network parameters for our cost models [146].

1. BW_{wl} - Bandwidth of the wireless link - 2 Mbps.
2. BW_{w} - Bandwidth of the wired backbone network - 155 Mbps.
3. L_{wl} - Latency of the wireless link including data link and network layer processing - 2 ms.
4. L_{w} - Latency of the wired backbone including data link and network layer processing - 500 μs.
5. PPT_{fixed} - Protocol Processing Time (PPT) for control messages - 0.5 ms.
6. PPT_{adm} - Protocol Processing Time for admission control - 2 ms.
7. S_{ctrl} - Maximum size of a control packet - 50 B.
8. S_{data} - Maximum size of a data packet - 1 KB.
9. T_{acq} - Time for a MH to acquire a wireless channel from a BS. - 5 ms.

These parameters are realistic and typical considering the present day technology [14, 146, 155, 156]. We measure the performance of the metrics by changing the number of hops in the appropriate paths depending on the rerouting scheme. These hops are denoted as follows:

1. $H_{server-new}$ - Number of hops in the new connection.
2. $H_{server-old}$ - Number of hops in the old connection.
3. H_{ctrl} - Number of hops in the control bath between BS_{old} and BS_{new}.

The assumption of the cost models [146] are as follows:

1. Perfect delivery of messages.
2. Maximum throughput for a connection is the throughput of the bandwidth of the wireless link.
3. Calculations are per-channel basis.

2.3.2 Cost model for Full rerouting (Without hints)

We calculate the total time required for each message (See Figures 2.3 and 2.5) which includes the time for transmission, forwarding and processing using the model proposed by Seshan [146] and Toh [155, 156]. We denote the Message k in the protocol schematic in Figure 2.5 as M_k. Also the time for M_k is denoted as T_k. Messages 1–4 are used in the handshake process 2.3 and Messages 5–10 are used to perform the actual rerouting 2.5. Refer to the description of the protocol for explanation of the messages.

1. Time for M_1 is the sum of time for acquiring the wireless channel, propagation on the wireless link, fixed PPT on the new BS.

$$T_1 = T_{acq} + \frac{S_{ctrl}}{BW_{wl}} + L_{wl} + PPT_{fixed} \tag{2.1}$$

2. Time of M_2 is the sum of transmission and propagation delay of the acknowledgment and the processing time at MH.

$$T_2 = \frac{S_{ctrl}}{BW_{wl}} + L_{wl} + PPT_{fixed} \tag{2.2}$$

3. Time for M_3 is the end-to-end delay on the control channel between BS_{old} and BS_{new} and fixed PPT in BS_{old}.

$$T_3 = (\frac{S_{ctrl}}{BW_w})H_{server} + L_{wl} + PPT_{fixed} \tag{2.3}$$

4. Time for M_4 is the same as that for M_3.

$$T_4 = T_3 \tag{2.4}$$

5. Time for M_5 is the total time required for transmission and propagation time and fixed PPT along each hop on the new connection.

$$T_5 = (\frac{S_{ctrl}}{BW_w} + L_w + PPT_{fixed})H_{server-new} \tag{2.5}$$

6. Time for M_6 is the same as the time for M_5.

$$T_6 = T_5 \tag{2.6}$$

7. Time for M_7 is the sum of time for sending the control packets, latency over the wireless link, fixed PPT and PPT for admission control

$$T_7 = \frac{S_{ctrl}}{BW_{wl}} + L_{wl} + PPT_{fixed} + PPT_{adm} \tag{2.7}$$

8. Time for M_8 is the same that of M_6

$$T_8 = T_6 \tag{2.8}$$

9. Time for M_9 is the total time required for transmission and propagation time and fixed PPT along each hop on the old connection.

$$T_9 = (\frac{S_{ctrl}}{BW_w} + L_w + PPT_{fixed})H_{server-old} \tag{2.9}$$

10. Time for M_{10} is the same that of M_9

$$T_{10} = T_9 \tag{2.10}$$

2.3.3 Calculation of metrics common to all schemes

We know define and describe in detail various performance metrics (service disruption time, rerouting completion time and buffering requirements at the base station and the mobile host) that are common to all the schemes. We show how to calculate these metrics using the analytical cost modeling done for the signaling messages in the previous subsection.

1. Disruption Time ($T_{disrupt}$): It is the amount of time during which the network service to the downlink to the MH is disrupted. The total time during this process is the time to acquire the channel plus the time to BS_{new} to arrange

forwarding of the data meant for MH to BS_{old} plus the time for the MH to receive the first packet of forwarded data.

$$T_{disrupt} = T_1 + \frac{S_{ctrl}}{BW_{wl}} + T_3 + T_4 + \frac{S_{data}}{BW_{wl}} + L_{wl} \tag{2.11}$$

2. Completion Time ($T_{complete}$) is the total time for all the routing to be completed. Since all the messages in the protocol occur sequentially except for acknowledgments, $T_{complete}$ is the sum of times for all messages except the acknowledgments.

$$T_{complete} = T_1 + \frac{S_{ctrl}}{BW_{wl}} + T_3 + \frac{S_{ctrl}}{BW_w} + T_5 + T_6 + T_7 + T_8 + T_9 + T_{10} \tag{2.12}$$

3. Buffer Size required by the MH (B_{mh}): It is the product of the time during which the MH can not transmit and the bandwidth of the wireless link.

$$B_{mh} = (T_1 + T_2)BW_{wl} \tag{2.13}$$

4. Buffer size at BS_{old} for downlink data (B_{bsdn}): It is the amount of data that has to be forwarded to the MH using BS_{new}. The time period in this context is the one during which BS_{new} arranges the forwarding of data and awaits the corresponding acknowledgment. In addition, the buffers should be able to store data that is being transmitted from MH_{dest} intended for MH_{src}. It is assumed that the first packet of data immediately follows M_4 — acknowledgment of the forwarding request.

$$B_{bsdn} = (T_1 + \frac{S_{ctrl}}{BW_{wl}} + T_3 + \frac{S_{ctrl}}{BW_w} + L_{wl})BW_{wl} \tag{2.14}$$

5. Buffer size at BS_{new} for uplink data (B_{bsup}): This is dependent of two factors. The first is the time to perform data forwarding (without the acknowledgments) minus the time needed for the first data packet to have arrived at BS_{new}. The second is the difference in the delay on the old and the new connections.

$$B_{bsup} = max(T_3 + T_4 - (2L_{wl}), [L_w + \frac{S_{data}}{BW_w}]max(H_{ctrl} + H_{old} - H_{new}, 0))BW_{wl} \tag{2.15}$$

The calculations of these metrics for the other schemes are shown in [139].

2.3.4 Calculation of special metrics for various rerouting schemes

We now describe the calculation of metrics specialized for each rerouting scheme. For each scheme we formulate the cost in terms of the times of messages in the rerouting scheme. The detailed cost modeling of each of these times can be done similar to the cost modeling shown above in Section 2.3. It should be noted the time T_k is the time required for message M_k for that *specific* rerouting scheme.

Full rerouting (without hints)

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{dest} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

$$T_{tear} = T_9 + T_{10} \tag{2.16}$$

2. New connection setup time (T_{new}): The total time required from the time MH_D informs BS_{old} to send a list of active connections to the time when the BS_{dest} confirms the establishment of the connections.

$$T_{new} = T_1 + T_2 + T_3 + T_4 + T_5 + T_6 \tag{2.17}$$

Full rerouting (with hints)

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{dest} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

$$T_{tear} = T_{10} + T_{11} \tag{2.18}$$

2. New connection setup time (T_{new}): The total time required from the time MH_D informs BS_{old} to send a list of active connections to the time when the BS_{dest} confirms the establishment of the connections.

$$T_{new} = T_1 + T_2 + T_3 + T_4 \tag{2.19}$$

Partial rerouting (without hints)

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{cross} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

$$T_{tear} = T_{10} + T_{11} \tag{2.20}$$

2. New connection setup time (T_{new}): The total time since the MH_D requests a connection with BS_{new} till the confirmation of the establishment of the new connection.

$$T_{new} = T_1 + T_2 + T_3 + T_4 + T_5 \tag{2.21}$$

3. Time required to invoke cross-over discovery algorithm (T_{cross}): The total time in the rerouting process till the cross-over switch discovery algorithm has been invoked.

$$T_{cross} = T_1 + T_2 + T_3 \tag{2.22}$$

4. Time to discover the cross-over switch ($T_{discover}$): The total time for finding the cross-over switch after invoking the cross-over discovery algorithm. This term depends on the algorithm used.

$$T_{discover} = T_4 \tag{2.23}$$

5. Partial Re-Use Efficiency (η_{part}): The fraction of the new path that has been re-used.

$$\eta_{part} = \frac{H_{crossnew}}{H_{new}} \tag{2.24}$$

Partial rerouting (with hints)

1. Old connection tear down time (T_{tear}): The total time required is the time required for the BS_{cross} to inform BS_{old} to tear down the old connection and for the BS_{old} to comply.

$$T_{tear} = T_{11} + T_{12} \tag{2.25}$$

2. New connection setup time (T_{new}): The total time since the MH_D requests BS_{old} to send the list of its active connections to BS_{new} till the confirmation of the establishment of the new connection.

$$T_{new} = T_1 + T_2 + T_3 + T_4 \tag{2.26}$$

3. Time required to invoke cross-over discovery algorithm (T_{cross}): The total time in the rerouting process till the cross-over switch discovery algorithm has been invoked.

$$T_{cross} = T_1 + T_2 + T_3 \tag{2.27}$$

4. Time to discover the cross-over switch ($T_{discover}$): The total time for finding the cross-over switch after invoking the cross-over discovery algorithm. This term depends on the algorithm used.

$$T_{discover} = T_4 + T_5 \tag{2.28}$$

5. Partial Re-Use Efficiency (η_{part}): The fraction of the new path that has been re-used.

$$\eta_{part} = \frac{H_{crossnew}}{H_{new}} \tag{2.29}$$

Tree - Group rerouting (without hints)

1. Multicast Tree setup time ($T_{mcast-setup}$):

$$T_{mcast-setup} = T_1 + T_2 + T_3 + T_4 + T_5 \tag{2.30}$$

2. Multicast Tree tear down time ($T_{mcast-tear}$):

$$T_{mcast-tear} = T_9 + T_{10} \tag{2.31}$$

3. Number of Nodes in the Multicast Tree (N_{mcast}):

   N_{mcast} depends on the algorithm in question. In static algorithms, this value is fixed and remains constant while in algorithms that dynamically decide the number on-the-fly, the number changes dynamically.

4. Multicast Join/Leave Time ($T_{mcast-jn}$, $T_{mcast-lv}$):

$$T_{mcast-jn} = T_5 \tag{2.32}$$

$$T_{mcast-lv} = T_9 \tag{2.33}$$

Tree - Group rerouting (with hints)

1. Multicast Tree setup time ($T_{mcast-setup}$):

$$T_{mcast-setup} = T_1 + T_2 + T_3 \tag{2.34}$$

2. Multicast Tree tear down time ($T_{mcast-tear}$):

$$T_{mcast-tear} = T_8 + T_9 \tag{2.35}$$

3. Number of Nodes in the Multicast Tree (N_{mcast}): N_{mcast} depends on the algorithm in question. In static algorithms, this value is fixed and remains constant while in algorithms that dynamically decide the number on-the-fly, the number changes dynamically.

4. Multicast Join/Leave Time ($T_{mcast-jn}$, $T_{mcast-lv}$):

$$T_{mcast-jn} = T_3 \tag{2.36}$$

$$T_{mcast-lv} = T_8 \tag{2.37}$$

Tree - Virtual rerouting

1. Virtual Tree setup time ($T_{vtree-setup}$): Let the cost of choosing the root be $T_{vtree-root}$ and N_{vtree} be the number of nodes in the virtual tree. The total cost of the statically constructing the tree involves involve BS_root broadcasting its presence to all the base stations in the neighborhood ($T_{broadcast}$) and the nodes acknowledge and perform multicast join ($T_{mcast-jn}$). Let $T_{mcast-lv}$ be the time required for the nodes to perform a multicast leave.

$$T_{vtree-setup} = T_{vtree-root} + T_{broadcast} + T_{mcast-jn} \tag{2.38}$$

2. Virtual Tree tear down time ($T_{vtree-tear}$):

$$T_{vtree-tear} = T_{mcast-lv} \tag{2.39}$$

3. Number of Nodes in Virtual Tree (N_{vtree}):

   N_{vtree} is determined statically and remains fixed.

Cell Forwarding

1. Old connection tear down time (T_{tear}):

$$T_{tear} = T_9 + T_{10} \tag{2.40}$$

2. New connection setup time (T_{new}):

$$T_{new} = T_1 + T_2 + T_3 + T_4 + T_5 \tag{2.41}$$

3. Time to establish path between BS_{old} and BS_{fwd} ($T_{cellfwd-path}$):

$$T_{cellfwd-path} = T_4 + T_5 \tag{2.42}$$

2.4 Comparison of rerouting schemes

In order to perform comparison the rerouting schemes, we performed analytical cost modeling of metrics for each rerouting scheme. We first look at the advantages and the disadvantages of each of the rerouting scheme.Then, we compare the schemes using the metrics. To this end, we first consider metrics that are not dependent on the path length (in terms of number of hops) of the old/new connection. Next, we consider the metrics that are dependent on the length of the old/new connection. In this case, we vary the path length to determine its effect on these metrics.

2.4.1 Advantages and Disadvantages of the Rerouting Schemes

The advantages and disadvantages of the various rerouting schemes are tabulated in Figure 2.11. This is a summary of the advantages and disadvantages of the rerouting schemes are described in earlier sections when discussed each of the rerouting scheme separately.

Rerouting Scheme	Advantages	Disadvantages
Full	- Simple - Easy to implement	- Naive - Inefficient - Slow - Prone to data loss
Partial	- Maximizes resource utilization - Reuses scarce resources	- Prone to data loss - Onus of cross-over discovery
Tree-Virtual	- Fastest - Works well if enough resources present - Efficient - Low data loss	- Requires setting up of multiple connxns. - Virtual Tree is static : does not allow incorporating dynamic groups - Difficult to decide tree membership
Tree-Group	- Fast - Membership can be dynamic. - Efficient - Low data loss	- Requires multiple connxns and large resources. - Wasted bandwidth because of multicasting
Cell Forwarding	- Simple - Easy to implement	- Inefficient especially for fast moving mobile host in small coverage area. - Slow - Prone to data loss - Requires special anchor

ADVANTAGES AND DISADVANTAGES OF REROUTING SCHEMES

Figure 2.11: Advantages and Disadvantages of Various Rerouting Schemes.

2.4.2 Metrics not dependent on the connection length

These metrics are fixed and give a complexity of the rerouting protocol in question. We base this comparison on the work done by Akyol and Cox [13]. These include number of messages exchanged during handoff and rerouting, the number of nodes and user connections involved for rerouting, the user bandwidth allocated, whether the scheme is tailored for wide-area networks or local-area networks or ATM based networks. Refer to the Table in Figure 2.12.

SCHEME \ METRIC	Handoff Messages	Rerouting Messages	Nodes Involved	User Connections	Bandwidth	WAN/ LAN	Rerouting Type	Cell Type	Traffic Type	Ref/Author
Full (no hint)	2	8	2+D	1	1	W	Full(Without hint)	micro/pico	Both	Seshan [58]
Full (hint)	6	5	2+D	1	1	W	Full(With hint)	macro/micro	Both	Seshan [58]
Incremental (no hint)	2	9	3+D	1	1	W	Partial(Without hint)	macro/micro	Both	Seshan [58]
Incremental (hint)	7	5	3+D	1	1	W	Partial (With hint)	macro/micro	Both	Seshan [58]
Multicast (no hint)	2	8	3+D	N	N	W	Tree - Group(Without hint)	macro/micro	Both	Seshan [58]
Multicast (hint)	6	3	N+D	N	N	W	Tree - Group(With hint)	macro/micro	Both	Seshan [58]
SRMC	3	9	N+D	N	1	W	Tree - Virtual (Without hint)	macro/micro	Both	Wu & Leung [67]
BAHAMA	2	5	4+D	1	1	W	Cell Forward	macro/micro	Both	Eng et. al. [27]
Virtual Tree	1	1	N+D	1	1	W	Tree - Virtual (Without hint)	macro/micro	Both (?)	Acampora et.al. [1]
Adaptive	2	4	3+D	1	1	W	Cell Forward	macro/micro	Both (?)	Yuan [29]
Incremental (no hint)	4	7	3+D	1	1	L	Partial(Without hint)	macro/micro	Both	Toh [63]
Incremental (hint)	3	4	3+D	1	1	L	Partial (With hint)	macro/micro	Both	Toh [63]
NCNR (Direct Link)	7	2	2	1	1	W	Partial(Without hint)	macro/micro	Both	Akyol & Cox [8]
NCNR (No Direct Link)	9	4	4+D	1	1	W	Partial(Without hint)	macro/micro	Both	Akyol & Cox [8]
Picocellular (Subnet)	2	0	N+D	N	N	W	Tree - Group(Without hint)	micro/pico	Both	Ghai & Singh [30]
Picocellular (Diff Subnet)	4	2	N+D	N	N	W	Tree - Group(Without hint)	micro/pico	Both	Ghai & Singh [30]

COMPARISION OF RE-ROUTING SCHEMES

? - Not Described in the Reference
D - Dependent on the Topology of the Network.
N - Number of leaves in the multicast/virtual tree
W - WAN
macro/micro - macrocells and microcells
L - LAN
micro/pico -macrocells and microcells and picocells
:Both - Both Time-Dependent and Throughput Dependent

Figure 2.12: Comparison of Example rerouting schemes

Number of messages exchanged during handoff

The Partial rerouting Scheme (with hints) requires the maximum number of messages for handoff while Full rerouting (without hints), Partial rerouting (without hints), Tree - Group rerouting (without hints), Tree - Virtual rerouting and Cell

Forwarding require the least.

Number of messages exchanged during rerouting

Cell Forwarding requires the maximum number of messages exchanged during rerouting while Tree - Group (with hints) requires the minimum number. The number of messages for the other schemes lies in the between the maximum and the minimum.

Number of nodes involved in handoff and rerouting

The number of the nodes required depend on the network. However, in terms of the minimum requirements, the Full rerouting schemes are the best as they require at least 2 nodes (BS_{old} and BS_{new}) in addition to BS_{dest} while the other schemes require at least three nodes in addition to BS_{dest}.

Number of user connections established for rerouting

The Tree - Group rerouting schemes can handle N_{mcast} connections while the others handle only one connection.

User bandwidth allocated for handoff and rerouting

If we consider the bandwidth allocated for handoff and rerouting for the Full rerouting (without hints) as unity then all the schemes have the same bandwidth requirements with the exception of the Tree - Group rerouting schemes. The Tree - Group rerouting require a bandwidth that is N times the unit bandwidth requirements.

Whether the scheme is proposed for WANs or LANs

All the schemes can work for WANs and LANs. However, from Figure 2.12, we see that the schemes proposed by Toh are specifically for LANs.

Whether the scheme is tailored for ATM based networks

The BAHAMA scheme [79], the Virtual Tree Scheme [1], the Adaptive Scheme [81], and the Incremental Rerouting Schemes [155, 156] are tailored for ATM based schemes.

Figure 2.12 tabulates these metrics for all the example rerouting schemes explained in this work.

2.4.3 Metrics dependent on the connection length

We now study the metrics that dependent on the network (specifically, the connection path length). These include the service disruption time, the rerouting completion time, the buffering required at the mobile host and buffering required at base station for the uplink and the downlink. Figures 2.13, 2.14, 2.15, 2.16, 2.17 show the effect of old connection path length on the service disruption time, total rerouting time, buffering requirements at the mobile host, buffering requirements at the base station on the uplink and buffering requirements at the base station on the downlink respectively. In these figures, we vary the old connection path length from 1 hop to 10 hops. We assume that the new connection path length is of the same length as the old connection and the control path length is twice the size of the connection path length.

We now discuss the performance of the various rerouting schemes discussed above in detail. Figures showing the performance the rerouting algorithms are provided to aid the discussion.

Service Disruption Time

Figure 2.13 depicts the effect of old connection path length on the service disruption time. From the figure, we see that:

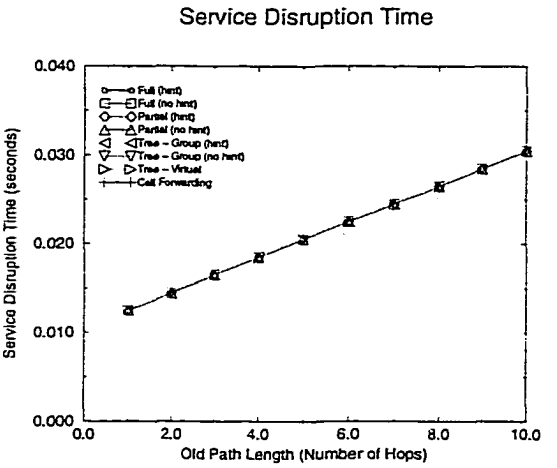


Figure 2.13: Service Disruption Time v/s Number of Hops in the Old Path

1. The minimum service disruption time is for the Tree - Group (with hints) rerouting Scheme and is fixed. It does not vary with the number of hops in the path.

2. For all schemes except the Tree - Group (with hints), the Service Disruption Time is dependent on the path length between BS_{old} and BS_{new} namely H_{ctrl}. This is because that in the case of Tree - Group (with hints) scheme, there is no need to forward the data from BS_{old} to BS_{new} as the data is being multicast to both BS_{old} and BS_{new}.

3. The service disruption time for the Full (without hints),Full (with hints), Partial (without hints), Partial (with hints), Tree - Group (without hints), Cell Forwarding rerouting schemes is the same because all of these schemes depend on forwarding of downlink data from BS_{old}. The service disruption time for Tree (Virtual) is not dependent the length of H_{ctrl} as it does not rely on downlink data forwarding. It is however dependent on the length of the new path.

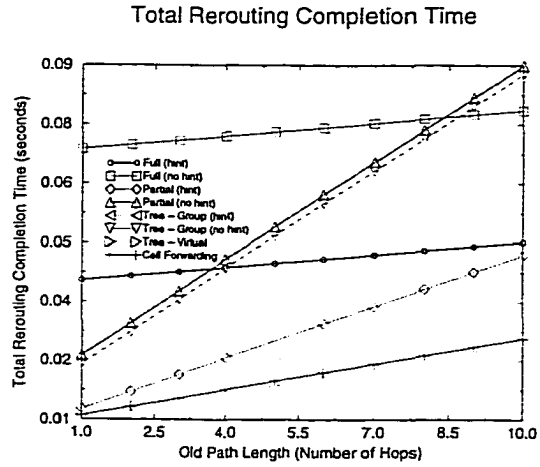


Figure 2.14: Rerouting Completion Time v/s Number of Hops in the Old Path

In case of the Tree (Virtual) rerouting, when the mobile host moves to a new base station, the root automatically recognizes that a handoff has taken place. There is no need to establish a new path or remove an old path. However, a message to reroute data on the new path needs to be sent from the new base station and a acknowledgement confirming the completion of rerouting has to be sent (hence the dependence of the length of the new path). Since, we assume that the new and the old path lengths are equal and half the size of the control path, we see that the service disruption of the Tree (Virtual) is indirectly dependent on the old path length.

Rerouting completion time

Figure 2.14 depicts the effect of old connection path length on the rerouting completion time. From the figure, we see that:

1. The minimum rerouting time is for the Tree - Virtual rerouting Scheme. However, the metric varies with the number of hops H_{new} (and hence with the

number of hops in the old path) as it. However, there is no need to either forward downlink data or form new connections or delete connections as the Virtual Tree is fixed.

2. The rerouting completion time for cell forwarding is almost as good. In case of cell forwarding, there is no need to form new connections or delete old connections as the connection path is simply extended. The problem with cell forwarding is that as the mobile host moves farther from its original base station (like in the case of multiple movements), the length of the downlink forwarding path can become very large giving rise to inefficiency. However, in case of one movement, cell forwarding performs very well.

3. For Tree - Group (with hints) rerouting the completion time is dependent on H_{old} performs fairly well because it does not require any downlink data forwarding.

4. Full rerouting and Partial rerouting perform worse than the other rerouting scheme. When the old path is small, full rerouting performs worse than partial rerouting. However, as the old (and the new path) path length increases, partial rerouting performs worse than full rerouting because of the additional onus of computing the "cross-over" point.

5. The completion time for the schemes with hints is lesser than their counterparts which do not take advantage of hints. The completion time for rerouting schemes without hints is dependent on the length of the new path.

6. In case of rerouting schemes with hints, the completion time is dependent and

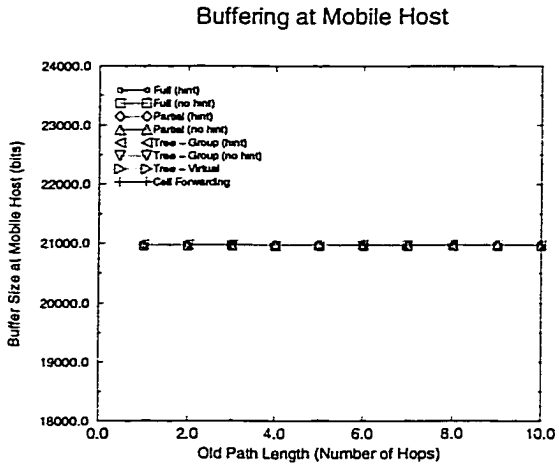


Figure 2.15: Buffering at the Mobile Host v/s Number of Hops in the Old Path

closely related to time needed for forwarding down link data and tear down old connections.

Buffering required at the mobile host

Figure 2.15 depicts the effect of old connection path length on the buffering requirements at the mobile host. From the figure, we see that in each rerouting scheme, the buffering at the MH is only used to buffer data for the two specific messages in their respective protocol. The first is the message that MH sends to the BS_{new} to register and the second is for the acknowledgment that it receives in response to the first message, The cost of the messages is constant for the given parameters for all the schemes. Thus, all the schemes require the same amount of buffering at the MH.

Buffering required in base station for the uplink

Figure 2.15 depicts the effect of old connection path length on the buffering requirements at the base station for uplink. From the figure, we see that:

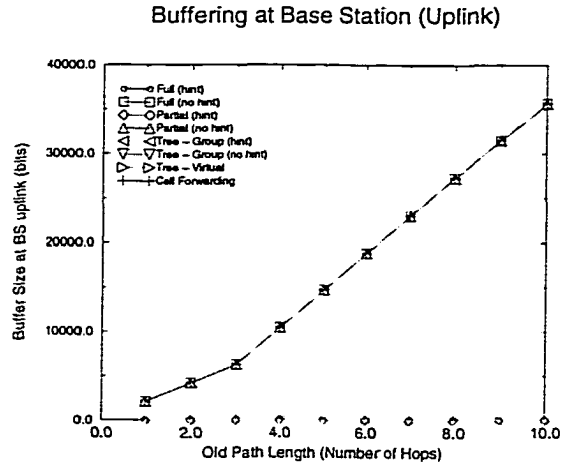


Figure 2.16: Buffering at the Base Station for the Uplink v/s Number of Hops in the Old Path

1. The buffering requirements for uplink data for all the schemes without hints including Tree (Virtual) and Cell Forwarding are the same.

2. There are no buffering requirements for uplink data for all the schemes with hints if the length of new and old forwarding path is the same (as assumed from the parameters). It should be noted that we need a different analysis if this assumption is removed. In general, buffering is dependent on the difference of the old and new path lengths.

3. Except for the case of Tree (Virtual) rerouting, the buffering requirement for uplink data is proportional to the forwarding path length H_{ctrl} - the path length between BS_{old} and BS_{new} . In case of Tree (Virtual) rerouting, the buffering requirement for uplink data is proportional to the new path length between BS_{source} and BS_{new} (proportional to H_{ctrl} from our assumption).

Buffering required in base station for the downlink

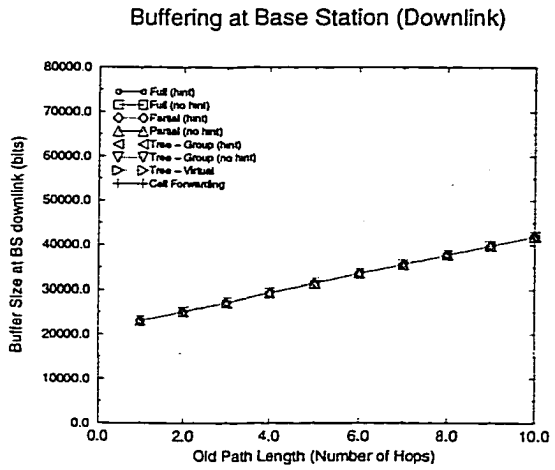


Figure 2.17: Buffering at the Base Station for the downlink v/s Number of Hops in the Old Path

Figure 2.17 depicts the effect of old connection path length on the buffering requirements at the base station for the downlink. From the figure, we see that:

1. This metric is very closely dependent on the time required for forwarding downlink data from BS_{old} to BS_{new}.

2. Since all schemes with the exception of the Tree - Group (with hints) and Tree (Virtual) scheme require data forwarding of downlink data, the buffering required in the base station for downlink for them is larger than the requirements for the Tree - Group (with hints) scheme.

3. The buffering requirements are maximum for the Tree (Virtual) rerouting as it does not rely on downlink data forwarding on the control path between BS_{old} to BS_{new}. It buffering requirements are based on the new path length. That is to say, the new base station BS_{new} has to buffer all the data on the downlink until it gets an acknowledgment from the server (source) that it has accepted

the request to reroute data to BS_{new}.

4. There is a fixed amount of downlink buffering for the Tree - Group (with hints) scheme is to buffer the data during the handoff period alone while the others require downlink buffering for the handoff and the time period till BS_{new} requests forwarding of data.

Refer to the Figure 2.18 that shows the values of some of these metrics for typical values of the system parameters (gathered from literature including [146, 155, 156]) which are also shown in the figure and described in [139].

Rerouting Scheme / Special Metric	Without Hints	With Hints
Full		
Old connxn tear down time	12.36 ms	12.36 ms
New connxn set up time	23.82 ms	22/76 ms
Partial		
Old connxn tear down time	12.36 ms	12.36 ms
New connxn set up time	16.18 ms	15.55 ms
Time for cross-over discovery	1.06 ms	1.06 ms
Partial Reuse ratio	0.5	0.5
Tree-Group		
Tree tear down time	12.36 ms	12.36 ms
Tree set up time	16.43 ms	16.43 ms
Tree-Virtual		
Tree tear down time	12.36 ms	12.36 ms
Tree set up time	16.43 ms	16.43 ms
Cell Forwarding		
Old connxn tear down time	12.36 ms	12.36 ms
New connxn set up time	16.18 ms	16.18 ms
Time to connect to anchor	2.12 ms	2.12 ms

Evalution of Special Metrics and Sytem Parameters

System Metrics

- Bandwith of wireless link - 2 Mbps
- Bandwidth of wired link - 155 Mbps
- Latency of wireless link - 2 ms
- Latency of wired link - 500 µs
- Protocol processing time for control msgs - 0.5 ms
- Protocol processing time for admission control - 2 ms
- Maximum size of control msgs - 50 bytes
- Maximum size of data packets - 1024 bytes
- Time to acquire a wireless channel - 5 ms
- Number of hops in old path - 6
- Number of hops in new path - 6
- Number of hops in path between old BS and the special BS. - 3
- Number of hops in control path between the old and new BS - 6
- N = Number of Nodes in the Tree.
- Time for choose root of the tree 0.25 ms !

Figure 2.18: Values of Special Metrics for Typical Values of System Parameters.

We see from the above discussions that:

1. The Tree - Group (with hints) and Tree - Virtual schemes perform the best.

However, each of these schemes have the overhead of using enormous resources or pre-establishing the tree. However, this overhead may be worthwhile for time-critical applications with a goal of minimizing rerouting completion time and service disruption.

2. In order to minimize service disruption, Tree - Group (with hints) is the best choice. If the rerouting completion time is the criterion to be considered, Tree (Virtual) and Cell Forwarding perform the best as they do not require forming new paths and tearing down old paths.

3. It is certainly advantageous to use rerouting with hints. This readily improves service disruption time (for Tree (Group) rerouting), total rerouting completion time and uplink buffering requirements at the base station.

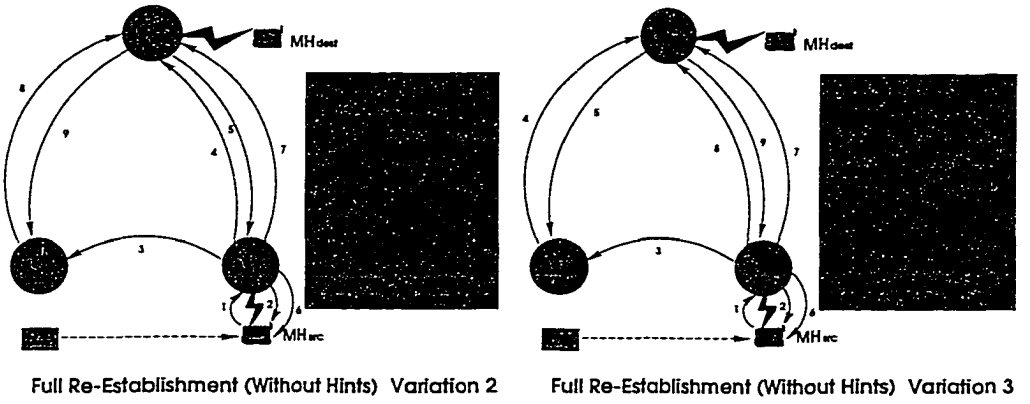
4. The topology of the network is important. In essence, the lengths of the new and old path is of vital importance.

5. Most of the schemes behave somewhat similarly because the underlying mechanisms use downlink data forwarding.

6. The Full rerouting and the Partial rerouting schemes perform the worst as they involve forming new paths, tearing down old paths and downlink data forwarding.

2.5 Parallelism in rerouting protocols

Parallelism can greatly improve the performance of computational tasks. With a view to improve the performance of the rerouting protocols, we look at extracting



(a) (a) Variation 2

(b) (b) Variation 3

Figure 2.19: Full Rerouting without hints: Variations 2 and 3

any implicit parallelism at the signaling level. To this end, we look at the message signaling for the rerouting process.We note that in all the schemes, the acknowledgments can be processed while sending other messages [146, 155, 156]. We explain this with respect with Full rerouting as an example.Full rerouting can be implemented with the following changes to the original Full rerouting (which we call - Variation 1). We briefly describe these schemes:

1. **Variation 2**: BS_{new} informs BS_{old} to stop forwarding its buffered data (either during handoff or explicitly during handshaking before rerouting) received from BS_{dest} and simultaneously establishes new connections and after the establishment, tears down the old connection.

2. **Variation 3**: Same as above except that the old connections are removed before establishing the new connections.

Figure 2.19 shows the protocol for these variations. In this figure, the handshaking protocol has been combined with the rerouting protocol. These variations have a larger degree of parallelism than the original version. Similar variations can be implemented to the other rerouting schemes.

In the Full rerouting variations (for example, Variations 2) described above, the following message exchanges can be done in parallel:

- Acknowledgments can be done processed in parallel with their corresponding succeeding signal.
- Messages 3 and 4 can be done in parallel. This is because the new connection can be established without bothering about receiving buffered data from BS_{old}.
- Messages 6 and 7 can be done in parallel. This is because BS_{new} can inform MH about the new connection and simultaneously can ask BS_{dest} to redirect the data.

Similar analysis for the other schemes can be done to improve their performance. Figure 2.20 shows the total completion time for full rerouting (without hints) for the all the above mentioned three variations as an example. The system parameters are the shown in Figure 2.18. The figure shows that variations 2 and 3 provide up to 46% improvement in the rerouting time over that of variation 1 in this case.We see that parallelism improves the total rerouting completion time and the performance for rerouting in general. Some of the signals and messages in the protocols described in previous works can be done in parallel thus improving the system's performance.

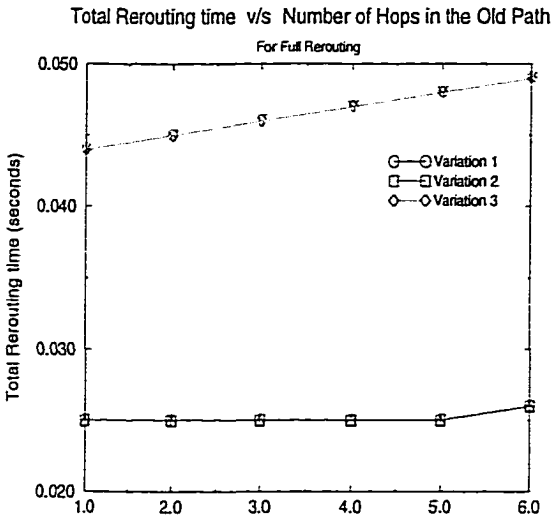


Figure 2.20: Total completion time for Full Re-routing in view of parallelism

2.6 Summary

In this chapter, we have examined existing various Static-Mobile rerouting schemes for connection-oriented mobile computing environments and have proposed a scheme to classify them. We also have conducted an exhaustive survey of various rerouting techniques and rerouting classification proposed by other researchers. We have proposed generic protocols for each of the rerouting scheme classes. In addition, we studied each class by looking at its different variations that have been proposed and/or implemented. We computed a variety of performance metrics to evaluate these rerouting schemes. These metrics include the ones common to all the rerouting schemes and the ones that are specific to each class. Finally, we compared these rerouting schemes based on the proposed metrics. Finally, we studied ways to improve the performance of the rerouting schemes by parallel execution of protocol messages.

Chapter 3

Rerouting in Mobile-Mobile Connections

3.1 Introduction

Most of the rerouting schemes for connection-oriented mobile networks proposed in literature are Static-Mobile rerouting schemes, that is, they assume that only one of the parties communicating in a session can be mobile host (typically the destination) and the other is stationary. Only Biswas [47], Ghai and Singh [87] and Cellular Telecommunication standard IS-41c [102] suggest schemes for mobile host to mobile host communication. Biswas's [47] strategy uses an already pre-established route between two stationary hosts that houses the mobile agents in-charge of the communication. The only rerouting involves both the source and destination mobile hosts establishing paths to these stationary hosts. The scheme proposed by Ghai and Singh [87] has a similar solution. Ghai and Singh's scheme uses dynamic multicasting

to perform rerouting. In case of cellular telecommunication using the EIA/TIA IS-41 (c) standard, cell forwarding takes place continuously as the mobile host move away. We extend Biswas's scheme by replacing the fixed path connecting the mobile representatives with a fixed core based tree. We will discuss this scheme in detail later in the chapter.

In this chapter, we propose a generic framework for Mobile-Mobile rerouting in connection-oriented networks while allowing unlimited movement by both the source and destination mobile hosts and alleviating the problem of non-optimal paths. Connection-oriented networks provide guarantees on performance (even under congestion) needed for delivery of multimedia data on mobile hosts and hence the motivation. We also compare and contrast all proposed Mobile-Mobile rerouting schemes including the one proposed in this chapter. We believe, that this is the first such effort.

The rest of the chapter is organized as follows: In Section 3.2, we present all the preliminaries required for understand the issues involved in rerouting in Mobile-Mobile connections. We allude to the problem that exists when the known Static-Mobile rerouting schemes are adopted for Mobile-Mobile rerouting. We also analyze the drawbacks of other Mobile-Mobile rerouting schemes. In Section 3.3, we propose a distributed algorithmic framework for performing optimal rerouting in Mobile-Mobile connections. We give a detailed example to show the working of the our proposed algorithm in Section 3.4. A detailed study of all the known solutions for rerouting in Mobile-Mobile connections is presented in Section 3.6. We compare these solutions using several performance metrics including the total rerouting distance, the cumulative connection path length and the number of connections as the

mobile hosts move. Finally in Section 3.7, we present our conclusions and future work.

3.2 Preliminaries

Communication in connection-oriented mobile networks can be either [47] between: two static hosts (Static-Static), or a static host and a mobile host (Static-Mobile) or two Mobile Hosts (Mobile-Mobile). Static-Mobile communication have been studied in the context of rerouting by [13, 47, 87, 156]. It is the last type (Mobile-Mobile) that has not been explored much in the literature. Rerouting in mobile environments occurs as result of a handoff. When a mobile host (MH) moves from the region of influence (cell) of a base station (BS) to another, a handoff is said to have taken place. In order to maintain communication connectivity, packets have be re-routed to and from the MH. In a Mobile-Mobile connection, the source mobile host MH_S is in a session with the destination mobile host MH_D. MH_S (resp. MH_D) may move from the its present base station BS_{SO} (resp. BS_{DO}) to BS_{SN} (resp. BS_{DN}).

3.2.1 Problems using the Static-Mobile rerouting algorithms for rerouting in Mobile-Mobile connections

Rerouting techniques that work for Static-Mobile connections [13, 47, 87, 146, 156] do not work for Mobile-Mobile connections as they create certain problems. We discuss the problems below:

Inefficiency

The original protocol assumes that only one end of a session is mobile. So, if MH_S moves (and assumes that MH_D is fixed) it establishes the new path between BS_{SN} and BS_{DO}, whereas if MH_D also moves (and assumes that MH_S is fixed), it establishes the new path between BS_{DN} and BS_{SO} whereas the correct path should be established between BS_{SN} and BS_{DN}. Hence we see that the rerouting and path establishment is inefficient.

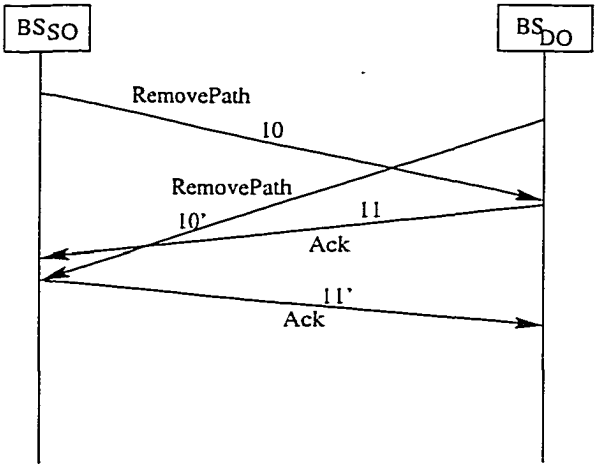


Figure 3.1: Problem with removal of the old path using the existing rerouting strategy.

Lack of Co-ordination

There is no mechanism to co-ordinate between the source and destination base stations (See Figure 3.1). Say at some point in time, both BS_{SO} and BS_{DO} are trying to remove their old paths. It should be noted that to remove a path a base station requests the base station at the other end of the connection to remove the path

(message RemovePath). When it receives an acknowledgment (message Ack), resources on the path are de-allocated and the path is removed and the connection terminated. As shown in the Figure 3.1, BS_{SO} may request to BS_{DO} for the path connecting them to be removed after it has been removed already by BS_{DO}. So, the request for path removal by BS_{DO} is unnecessary. Also because of the removal of the path, the request may not reach BS_{SO}. So, the bottom line is that the adaptation of the Static-Mobile rerouting for rerouting in Mobile-Mobile connections fails because of the lack of co-ordination between the source and the destination base stations.

3.2.2 Problems with other Mobile-Mobile rerouting algorithms for rerouting in Mobile-Mobile connections

As mentioned earlier, Biswas [47], Ghai and Singh [87] and Cellular Telecommunication standard IS-41c [102] have suggested schemes for mobile host to mobile host communication. Biswas's [47] strategy uses an already pre-established route between two stationary hosts that houses the mobile agents in-charge of the communication. The rerouting process involves establishing a path from both the source and destination mobile hosts to the respective stationary hosts. The disadvantage of this scheme is that, if the mobile hosts keep moving, the path used for communication may be inefficient as the scheme does not dynamically update the paths based on the location of the mobile hosts. The scheme proposed by Ghai and Singh [87] suffers from the same problem. In addition, Ghai and Singh's scheme uses dynamic multicast rerouting only. The architecture of the setup in more stringent and complex and uses a three tier hierarchy (mobile host, base station and supervisory host). In case

of cellular telecommunication using the EIA/TIA IS-41 (c) standard, cell forwarding takes place continuously as the mobile host move away. The obvious disadvantage is that the routes used are not optimal. We shall analyze the performance of all these schemes and the drawbacks in section 3.6.

3.3 Proposed Algorithm for Mobile-Mobile Rerouting

We present a distributed algorithm for rerouting in Mobile-Mobile connections which eliminates the problems described in section 3.2.1. In the discussion that follows, we designate one of the mobile hosts as the source mobile host (MH_S) and other as the destination mobile host (MH_D). The algorithm allows communication irrespective of the rerouting technique that is used is full, partial, cell forwarding or tree based. However, in the case of tree rerouting, we assume that there is a communication tree rooted at source base station with the destination base stations forming leaves of the tree. Tree routing involves, dynamically forming trees rooted at the current base station of MH_S as MH_S moves. In addition, if MH_D moves out of the tree, either the tree has to be extended to cover the base station that MH_D has moved to, or use cell forwarding to maintain communication between MH_S and MH_D.

3.3.1 Assumptions

We assume that there are no lost messages and that the messages are delivered in a FIFO manner on all of the channels. A mobile host can move any number of times, however, sometime in the future, it eventually stays in a cell long enough

for the rerouting to be completed. Each MH receives a "radio hint" informing it of its movement into another cell which can be used to setup connections apriori. We assume that there is initially a connection between the base stations in-charge of MH_S and MH_D. These base stations are designated as BS_{SO} and BS_{DO}, respectively. The identities of BS_{SO}, BS_{SN}, BS_{DO}, BS_{DN} are updated during the execution of the algorithm appropriately. Each BS maintains data structures in local memory to keep track of connections for mobile hosts in its cell, movements and identities of mobile hosts (MH_S and MH_D).

3.3.2 Data Structures Maintained at the Base stations

- Boolean Flag MH_SMoved (resp. Mh_DMoved) (Initialized to False) keeps track of whether the source (resp. destination) mobile host has moved.

- List of current path (which may be in the process of being removed) and the newly established path for the MH_S – MH_D session with the base station has one of the ends. This includes the old and the new paths. Initialized for BS_{SN} and BS_{DN} to NULL and for BS_{SO} and $BS_{DO}$to the old path. We assume that a path identity for each path between the source and the destination mobile host.

- Queue of the requests for connection establishment (resp. connection removal) through the base station and the associated action by the base station (Accept, Reject).

3.3.3 Overview of the Algorithm

The algorithm is event-driven and distributed in nature and runs on each of the entities (base stations and mobile hosts) in the mobile network. Depending on the type of the entity (a base station or mobile host), the appropriate portion of the algorithm is executed by the entity. Events can either be the arrival of message at the entity from another or in the case of a mobile host the initiation of movement (Moving) or reaching a new cell (ReachedNewCell). Messages are of the form MessageName(Src, Dest) where MessageName is the name of the message sent by Src to Dest (See Section 3.3.4 for details of events and messages).

The mobile hosts are responsible for informing the old base station about their movement using radio hints. BS_{DO} is responsible for informing BS_{SO} about MH_D's movement and for rejecting or accepting the request for the establishment of a new path to it. It also accepts the establishment of the destination forwarding path (between BS_{DO} and BS_{DN}). BS_{SO}'s responsibilities include requesting BS_{SN} to pre-establish a new path if MH_S has moved, informing BS_{SN} that MH_D has moved, if necessary, establishing new path if MH_S has not moved and removal of the old paths. BS_{SN}'s primary function is to establish the new path with the appropriate destination base station and removal of old paths. BS_{DN}'s responsibilities include trying to establish the destination forwarding path, accepting or rejecting the requests for establishment of new path and requesting that data be rerouted on the new path.

We describe below the processing at various entities involved in the rerouting process.

Processing at the Mobile Hosts

The mobile host MH_S (MH_D) inform its old base station when it begins to move

using the message MH_SMoved (MH_DMoved) along with the identity of the new base station taking the advantage of a radio hint. On reaching a new cell, the mobile host registers with the new base station (message Register). It updates its local data structures to note that has moved to a new cell when it receives an acknowledgment from the new base station (message RegisterAccept).

Processing at BS_{DO}

When BS_{DO} realizes that MH_D has moved it informs BS_{SO} about the movement of MH_D. It also informs the identity of the new base station BS_{DN} where MH_D has moved to. Whenever BS_{DO} is requested for a path establishment by BS_{SN} (if MH_S has moved), it rejects it if MH_D has moved, otherwise it accepts the connection. Also, BS_{DO} accepts the request for the establishment of a forwarding path between itself and BS_{DN}. This path is used to forward packets that arrived at the BS_{DO} which meant for MH_D. BS_{DO} also accepts the request for the removal of an old path through it when it receives a request from either BS_{SO} or BS_{SN}.

Processing at BS_{SO}

BS_{SO} sends a list of its active connections to BS_{SN} when it learns of MH_S's movement. When it realizes that MH_D has moved (on getting a message from BS_{DO}), it forwards this information to BS_{SN} if MH_S has moved, otherwise it tries to establish a path to BS_{DN}. When BS_{SO} receives a request to reroute data to BS_{DN}, it tries to remove the old path after the new path has been established. It also tries to remove the old path when it is requested by BS_{SN} to do so.

Processing at BS_{SN}

BS_{SN} acknowledges the registration of MH_S (message RegisterAccept). If it realizes that MH_D has moved it tries to establish a new path to BS_{DN}. If BS_{DO} rejects

BS_{SN}'s request for the establishment of a path then BS_{SN} realizes that the MH_S has moved and so tries to establish a path to BS_{DN}. It also tries to establish this new path when BS_{SO} informs BS_{SN} that MH_D has moved. If a destination base station requests BS_{SN} to reroute data on the new path and if the new path has been established the old paths are requested for removal. When BS_{SO} requests BS_{SN} to pre-establish the new path after sending the list of its active connections, BS_{SN} tries to establish a path to BS_{DN} if MH_D has moved otherwise it tries to establish a path to BS_{DO}.

Processing at BS_{DN}

BS_{DN} is responsible for acknowledging the registration of MH_D. It then requests the establishment of a forwarding path between itself and BS_{DO}. BS_{DN} rejects a request to establish a path to it from a source base station if MH_D has moved otherwise it accepts the connection. In case the new path has been established, it sends a request to reroute data on the new path.

3.3.4 Our Algorithm – Events, Messages and Processing

We now describe the distributed algorithm in detail including the various events and messages and the processing at the source and destination mobile hosts and base stations (old and new). First, we describe the all the events and the messages that as the basis of the algorithm. We, then present the algorithm in a distributed fashion. In order to do this, we specify all the events and the associated action for each of the entities in the system (mobile hosts and base stations).

Events and Messages

EVENTS

- Moving : Mobile Host moving.
- ReachedNewCell : Mobile Host enters the new cell and is ready to register.

MESSAGES

- MHMoved(MH, BS_O) : MH sends the message to BS_O informing that it is moving to BS_N.
- MH_DMovedToBS_{DN}(BS_{DO}, BS_{SO}) : BS_{DO} sends the message to the BS_{SO} informing that MH_D has moved to the new base station BS_{DN}.
- Fwd MesgMH_DMoved(BS_{SO}, BS_{SN}) : BS_{SO} forwards information about MH_D movement to BS_{SN} if MH_S has moved.
- EstablishPath(BS_S, BS_D) : BS_S sends a request to BS_D establish a new path BS_S to BS_D.
- RemovePath(BS_S, BS_D) : BS_S requests BS_D to remove the old existing path between them.
- IndirectRemovePath(BS_{SN}, BS_{SO}): BS_{SN} requests BS_{SO} that the old connection between BS_{SO} and BS_{DO} be removed.
- EstablishFwdPath(BS_O, BS_N) : The old base station BS_O requests the establishment of a forwarding path between BS_N and itself.

- SendDataOnNewRoute(BS_{DN}, BS_S) : BS_{DN} requests BS_S to transmit the data on the new established path.

- Accept(BS_D, BS_S) : Destination BS_D send to a source base station BS_S to acknowledge the establishment of a new route.

- Reject(BS_D, BS_S) : Destination BS_D send to a source base station BS_S to deny the establishment of a new route.

- Register(MH, BS_N) :Mobile host MH registers with the new base station BS_N after moving into its cell.

- RegisterAccept(BS_N, MH) : New base station BS_N accepts MH's registration.

- SendListofActiveConnxns(BS_O, BS_N) : BS_O informs BS_N of the active connections of the mobile host and requests the of the establishment of the new connections.

Algorithm

// AT THE MOBILE HOST

1. **Case (Event** at MH)
2. Moving:
3. MHMoved(MH, BS_O);
4. ReachedNewCell:
5. Register(MH, BS_N);

6. **Mesg** RegisterAccept(BS_N, MH):

7. $BS_O = BS_N$;

8. **End**

// AT THE OLD DESTINATION BASE STATION

9. **Case** (**Event** at BS_{DO})

10. **Mesg** MH_DMoved(MH_D, BS_{DO}):

11. MH_DMovedToBS_{DN}(BS_{DO}, BS_{SO});

12. **Mesg** EstablishPath(BS_{SN}, BS_{DO}):

13. **If** MH_D has moved to BS_{DN} **Then**

14. Send **Mesg** Reject(BS_{DO}, BS_{SN});

15. **Else**

16. Send **Mesg** Accept(BS_{DO}, BS_{SN});

17. **Mesg** EstablishFwdPath(BS_{DN}, BS_{DO}):

18. Accept(BS_{DO}, BS_{DN});

19. **Mesg** RemovePath(BS_S, BS_{DO}):

20. Accept(BS_{DO}, BS_S);

21. **End**

// AT THE OLD SOURCE BASE STATION

22. **Case** (**Event** at BS_{SO})

23. **Mesg** MH_SMoved(MH_S, BS_{SO}):

24. SendListofActiveConnxns(BS_{SO}, BS_{SN});

25. **Mesg** $MH_D MovedToBS_{DN}(BS_{DO}, BS_{SO})$:

26. **If** MH_S has moved to BS_{SN} **Then**

27. $FwdMesgMH_D Moved(BS_{SO}, BS_{SN})$;

28. **Else**

29. $EstablishPath(BS_{SO}, BS_{DN})$;

30. **End If**

31. **Mesg** $Accept(BS_{DN}, BS_{SO})$:

32. New path has been established;

33. **Mesg** $SendDataOnNewRoute(BS_{DN}, BS_{SO})$:

34. **If** new path has been established **Then**

35. $RemovePath(BS_{SO}, BS_{DO})$;

36. **End If**

37. **Mesg** $IndirectRemovePath(BS_{SN}, BS_{SO})$:

38. $RemovePath(BS_{SO}, BS_{DO})$;

39. **End**

// AT THE NEW SOURCE BASE STATION

40. **Case** (**Event** at BS_{SN})

41. **Mesg** $Reject(BS_{DO}, BS_{SN})$:

42. $EstablishPath(BS_{SN}, BS_{DN})$;

43. **Mesg** $Accept(BS_D, BS_{SN})$:

44. New path has been established;

45. **Mesg** $SendDataOnNewRoute(BS_D, BS_{SN})$:

46. **If** new path has been established **Then**

47. **If** Old Path exists (BS_{SN} to BS_{DO}) **Then**

48. RemovePath(BS_{SN}, BS_{DO});

49. **Else**

50. IndirectRemovePath(BS_{SN}, BS_{SO});

51. **End If**

52. **End If**

53. **Mesg** MH_DMovedToBS_{DN}(BS_{DO}, BS_{SN}):

54. EstablishPath(BS_{SN}, BS_{DN});

55. **Mesg** Register(MH_S, BS_{SN}):

56. RegisterAccept(BS_{SN}, MH_S);

57. **Mesg** SendListofActiveConnxns(BS_{SO}, BS_{SN}):

58. **If** MH_D has moved **Then**

59. EstablishPath(BS_{SN}, BS_{DN});

60. **Else**

61. EstablishPath(BS_{SN}, BS_{DO});

62. **End If**

63. **Mesg** FwdMesgMH_DMoved(BS_SO, BS_SN):

64. EstablishPath(BS_{SN}, BS_{DN});

65. **End**

// AT THE NEW DESTINATION BASE STATION

66. **Case** (**Event** at BS_{DN})

67. Mesg Register(MH_D, BS_{DN}):

68. RegisterAccept(BS_{DN}, MH_D);

69. EstablishFwdPath(BS_{DN}, BS_{DO});

70. Mesg EstablishPath(BS_S, BS_{DN}):

71. If MH_D has moved Then

72. Reject(BS_{DN}, BS_S);

73. Else

74. Accept(BS_{DN}, BS_S);

75. Mesg Accept(BS_{DO}, BS_{DN}):

76. If path established between BS_{DN} and BS_{SO}

77. SendDataOnNewRoute(BS_{DN}, BS_{SO});

78. Else

79. SendDataOnNewRoute(BS_{DN}, BS_{SN});

80. End

3.4 An Example of Rerouting using the Proposed Framework

We describe briefly the functioning of our proposed framework using the example depicted in Figure 3.2. Figure 3.2 depicts an example case involving one movement by the source mobile host and a couple by the destination mobile host. The example depicts one movement by the source mobile host and a couple by the destination mobile host. In this example, as with the rest of the chapter, MH_S and MH_D refer to the source and destination mobile host while BS_{SO} (BS_{SN}) and BS_{DO} (BS_{DN}) denote

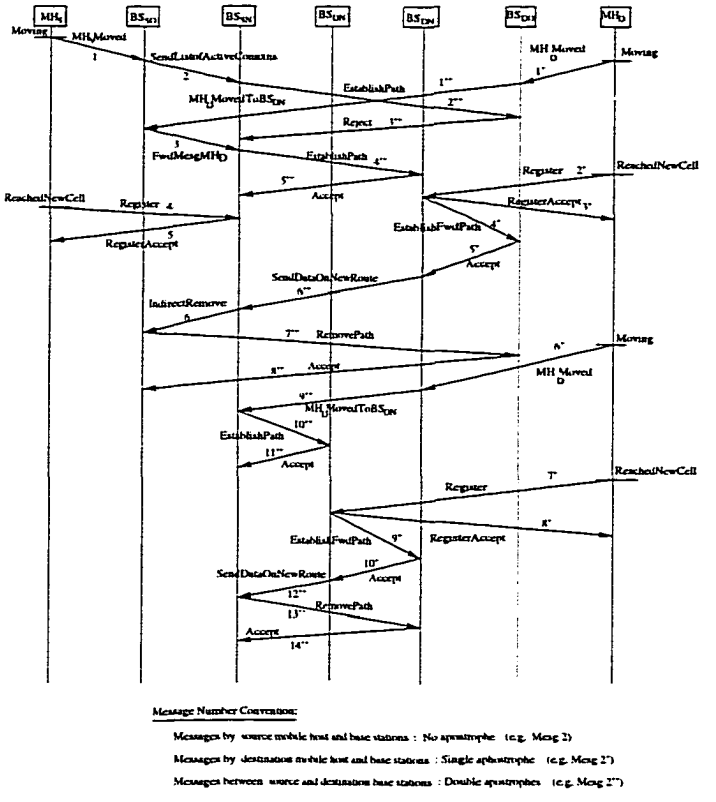


Figure 3.2: Timing diagram of our protocol for a case.

the old (new) source and destination base stations. In addition, $BS_{DN'}$ refers to the second new destination base station (for the second movement of the destination mobile host). The numbering scheme for message is as follows:

Messages amongst source mobile hosts and base stations have no apostrophes next to the message numbers (e.g. Message 2). Messages amongst destination mobile hosts and base stations have one apostrophe along with the numbers (e.g. Message 2'). Messages between source and destination base stations have two apostrophes next to the message numbers (e.g. Message 2").

First, MH_S moves and it sends a MH_SMoved message (1) to BS_{SO} which in turn sends the list of MH_S's active connections to BS_{SN} (SendListofActiveConnxns (2))

to BS_{SN} requesting it try to pre-establish a new path before the MH_S enters the new cell. BS_{SN} requests for a new path to BS_{DO} (2") which rejects the request as MH_D has moved (3"). Meanwhile, MH_D moves and it informs its old base station BS_{DO} (1'). BS_{DO} informs BS_{SO} that MH_D has moved to its new cell (1") because it assumes that MH_S has not moved. BS_{SO} in turn informs BS_{SN} that MH_D has moved (3). On getting this forwarded message, BS_{SN} successfully establishes the new path to BS_{DN} (4" and 5"). On reaching the new cell, MH_S registers with BS_{SN} (4) and is acknowledged (5). Similarly, on reaching the new cell, MH_D registers with BS_{DN} (2') and is acknowledged (3'). Simultaneously, BS_{DN} requests a establishing of a forwarding path to BS_{DO} (4') and is successful (5'). On the successful establishment of the forwarding path, BS_{DN} requests BS_{SN} to send the data on the established new path between BS_{SN} and BS_{DN} (6"). BS_{SN} requests BS_{SO} to remove the existing old path (6). BS_{SO} successfully removes the old path (7" and 8"). During the removal process, MH_D moves and informs the present old destination base station BS_{DN} (6') which informs the present old source bases station BS_{SN} (9"). BS_{SN} successful establishes the new path with $BS_{DN'}$ (10" and 11"). On reaching $BS_{DN'}$, MH_D registers and is acknowledged (7' and 8'). Simultaneously, $BS_{DN'}$ establishes successfully a forwarding path to the present old destination base station BS_{DN} (9' and 10'). The new destination base station requests the forwarding of data on the new path (12") to BS_{SN}. BS_{SN}, then removes the old path between itself and BS_{DN}.

3.5 Proof of correctness of our proposed algorithm

The following theorem established the proof of correctness of our proposed algorithm.

Theorem 1 *When the source and destination mobile hosts stay in cells long enough for the rerouting to be completed, a path is established between their corresponding base stations.*

Proof

In the proof, we reference messages and events using the line numbers of the appropriate steps of the algorithm described in 3.3.4.

When MH_S moves, it informs its present base station BS_{SO} about its movement (Lines 10–11). This base station then requests BS_{SN} to set up MH_S connections apriori (Lines 23–24). Similarly, when MH_D moves, it informs BS_{DO} about its movement (Lines 2–3). BS_{DO} then informs the last known source base station BS_{SO} about the movement (Lines 10–11). Thus, BS_{SO} keeps track of the mobility of both MH_S and MH_D. In this context, we consider the general case where both MH_S and MH_D have moved from the BS_{SO} to BS_{SN} and BS_{DO} to BS_{DN} respectively.

It should be noted that the current destination base station informs the last known source base station about MH_D's movement (Lines 10–11). On receiving this message, the source base station either establishes the new path (if MH_S has not moved) or informs the new source base station to which MH_S has moved.

When MH_S moves from $BS_{SO'}$ to BS_{SN}, it informs $BS_{SO'}$ about its movement (Lines 2–3)on receiving this information about MH_S's movement, $BS_{SO'}$ sends BS_{SN}

a list of the MH_S's active connections requesting it to set up the connections apriori (Lines 23–24). On receiving this request, BS_{SN} tries to establish a path to the last known destination base station (Lines 57–61). If MH_D has not moved from its last known base station, the path is established otherwise the path establishment is aborted by the destination base station (Lines 12–16, 70–74). If MH_D has moved, the destination base station sends a "Reject" message and informs the identity of the new base station where MH_D has moved to (Lines 12-14, 70–72). At time T_N, when MH_D moves into BS_{DN}'s cell and stays there long enough, a path is established between BS_{SN} and BS_{DN}. ■

3.6 Techniques for rerouting for Mobile-Mobile connections in connection-oriented networks

We know of only four techniques [47, 87, 102, 138] to perform rerouting in Mobile-Mobile connections. In this section, we discuss these techniques in more detail and look at their drawbacks and look closely at the technique that we propose in order to alleviate these drawbacks.

In this section, we will discuss the known techniques for rerouting in Mobile-Mobile connections. Specifically, we will discuss strategies proposed by Biswas [47], Ghai and Singh [87], the EIA/TIA IS 41(Revision c) cellular telecommunication standard. We also propose an extension of Biswas's work using a fixed multicast core based tree rather than a simple fixed path. We also present our proposed solution to overcome the drawbacks of the other proposed rerouting schemes.

3.6.1 Biswas's strategy: Mobile Representative and Segment Based Rerouting

Biswas [47] proposes the use of software mobile agent called mobile representatives that handle the connection management operations of mobile host. The representatives reside on one of the intermediate routers. Each mobile host has a corresponding mobile representative. Assume that the source mobile host MH_S (resp. the destination mobile host MH_D) is in the cell corresponding to the base station BS_S (resp. BS_D) and its mobile representative is MR_S (resp. MR_D). Thus, the initial route in the source mobile host - destination mobile host connection is: $MH_S - BS_S - MR_S - MR_D - BS_D - MH_D$. The crux of Biswas's rerouting strategy is that the path connecting the corresponding the mobile representatives is the same during the lifetime of the connection. Only the part of the connection between the mobile host and the mobile representative changes with handoff. The disadvantage of this scheme is that if the mobile hosts keep moving, the path used for communication may be inefficient as the scheme does not dynamically update the paths based on the location of the mobile hosts.

3.6.2 CBT (Core Based Tree) strategy: Extending Biswas's work

We propose an extension of Biswas' approach by using a core based tree connecting the source mobile representative to a group of destination mobile representative instead of a simple path connecting the mobile representatives. The core based tree allows multicasting of messages from the source mobile representative to the

group of destination mobile representatives. Since the core based tree is fixed and decided apriori it does not have the additional onus that is associated with dynamic multicasting as is the case with Ghai and Singh's scheme (discussed below).

The advantage with this scheme is that the total rerouting distance can be reduced significantly compared to Biswas's strategy. We will see in detail how this can be accomplished when we analyze the performance of this strategy. The only problem with the CBT is the excess use of resources as the core based tree has to be built apriori for the purpose of rerouting.

3.6.3 Ghai and Singh's strategy: Two level Picocellular Rerouting

Ghai and Singh [87] describe a picocellular architecture wherein as cell size decreases, the number of handoffs and therefore handoff overhead within the networks increases. The authors describe a method for reducing handoff overhead and buffer space required using multicast groups and mobile trajectory prediction is described, as well as a network architecture that supports this technique. Base Stations (referred to as mobility support stations or MSSs) do not have any intelligence, but rely on a centralized authority called a supervisory host (SH). The supervisory host calculates the mobile's likely trajectory, and forms a multicast group for MSSs that the mobile is likely to handoff to in the near future. All packets are multicast to this group; MSSs that do not currently host the mobile buffer packets in anticipation of a handoff; such buffered packets are tossed when the MSSs receive an update of the mobile's acknowledged sequence numbers. A connection-oriented network

architecture is also described in this paper. Virtual circuits are set up between end-points. The communication is optimized for the case of two mobiles is done using the base stations and supervisory host(s). Multiple supervisory hosts are needed if the communicating mobile hosts are under the supervision of different SHs. The scheme proposed by Ghai and Singh [87] suffers from the following drawbacks: The architecture is fixed, rigid and requires tree or tree-like topology. In addition, it requires the an extra level of hierarchy (supervisory hosts) as compared to the other schemes that we have discussed above and this results in longer paths. In addition, this scheme performs rerouting using dynamic multicasting and does not allow for the other rerouting strategies [138, 146].

3.6.4 EIA/TIA IS-41 (Revision C): Current Cellular Communication Handoff and Rerouting Management Protocols

The EIA/TIA IS-41 (Revision C) protocol for cellular communications is designed for dealing with handoffs and the subsequent forwarding of connections as the mobile hosts move. This is done by cell forwarding rerouting (chaining). Chaining in this case is done using switches as opposed to base stations used in all the other schemes described earlier. Since the connections are through switches, the links to old base stations are automatically removed during switching. These protocols are meant for circuit switched networks and not packet switched networks. Data loss and data ordering is not a important concern in this case. We cannot compare this scheme with the other schemes described earlier as this scheme is for circuit switched networks

and as switches are used as base stations. However, since the rerouting involves cell forwarding, this scheme is non-optimal and inefficient.

3.6.5 Comparison of rerouting schemes for Mobile-Mobile connections

We use a simple framework for comparing the performance of all the Mobile-Mobile rerouting schemes. The reason, we choose this simple scheme (which we explain in detail later) is that the Mobile-Mobile rerouting schemes are not tied to the type of rerouting (full, partial, cell, tree-based). These rerouting schemes basically concentrate on deciding the end points on the connections and can use any type of rerouting. Our comparison is based on calculating the total rerouting distance, the cumulative connection path length and the amount of resources used as the mobiles move. As seen earlier, the metrics that determine efficiency of rerouting schemes in Static-Mobile connections are dependent on connection length. For example, the total rerouting path length gives a good estimate of metrics such as throughput, the total rerouting time, the service disruption time and buffering requirements at the base stations. We use the extended cross-bar network and analysis as suggested by Song et al. in [148] for calculating the rerouting distance. The extended cross-bar architecture allows for simple calculations. This analysis can be extended to other types of networks.

In the architecture, the nodes are the base stations. The horizontal and vertical links are of length 1 while the slanted links are $\sqrt{2}$ in length. We use $RD_{s,d}$ to represent the reroute distance between nodes s and d. We use TRD(a, d) to represent

the total rerouting distance between BS_{SN} and BS_{DN} where the initial path length between the source and destination base station is a and the movement distance of the mobile host from its initial position is d.

We assume that:

- The MH_S and MH_D are moving in a straight line in the same direction and the top and the bottom rows of the grid respectively.
- Without any loss of generality, the mobile hosts both move d hops.
- MH_S moves from cell of its old base station BS_{SO} to a new base station BS_{SN}
- MH_D moves from cell of its old base station BS_{DO} to a new base station BS_{DN}
- There is a connection path between the old base stations for the mobile hosts to start with. Let the minimum length of this route be the straight line path connecting BS_{SO} and BS_{DO}. Let the length of this path be b.
- The mobile hosts stay at the new base station long enough for the rerouting to be completed.
- For simplicity of analysis, we assume that the source and the mobile hosts start moving at the same time and continue moving at the same speed.

We calculate the following metrics for comparing the performance for all of the Mobile-Mobile rerouting schemes. We consider each case separately for calculating the metrics.

- Total rerouting distance ($TRD(a, d)$): This is the path length connecting the new source and destination base station through the rerouting path when the

mobile hosts have moved d hops from their initial positions and the initial path length between their corresponding base stations is a. *This metric gives a good estimate of the system throughput for the connection between the source and the destination mobile hosts.*

- Cumulative connection path length (CCPL): This is the cumulative total of the lengths of new reroute paths formed and torn down as the mobile hosts move. In order to calculate the CCPL for the rerouting scheme, we calculate the cumulative length of new path paths formed (CLNP) and the cumulative length of old paths torn down (CLOP). So, CCPL is the sum of CLNP and CLOP. *This metric gives an estimate of system disruption, cumulative rerouting time and buffering requirements for the connection between the source and the destination mobile hosts.*

- Number of Connections (NC): We calculate the amount of resources reserved of each connection pair in terms of the maximum number of connections that can exist at any time during the entire rerouting process. This metric gives a reflection of the resources used by the system. In case of the CBT strategy and Ghai and Singh's scheme, since multicasting is used, excess resources have to be used.

We denote the performance metrics for each of the rerouting schemes by using the name of the scheme as a subscript. For example we represent the total rerouting distance for Biswas's scheme and the CBT scheme as $\mathrm{TRD}_{\mathrm{biswas}}(a, d)$ and $\mathrm{TRD}_{\mathrm{cbt}}(a, d)$, respectively.

Biswas's Strategy

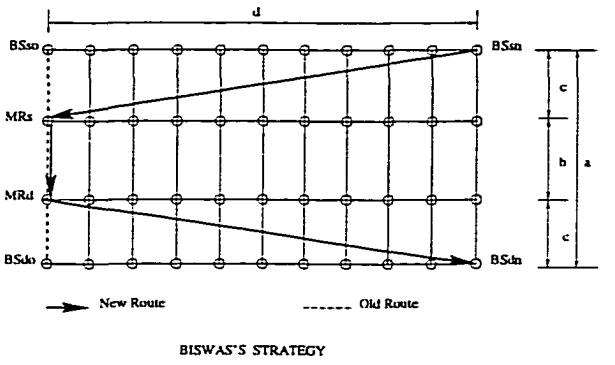


Figure 3.3: Rerouting Distance Calculation: Biswas's Strategy

Figure 3.3 depicts the rerouting using Biswas's strategy. MR_S and MR_D are the source and destination mobile representatives and the path connecting them is a fixed path. Let $RD_{MR_S,MR_D R} = b$ and $RD_{MR_S,BS_{SO}} = RD_{MR_D,BS_{DO}} = c$. Thus, $a = b + 2c$. Now, $RD_{MR_S,BS_{SN}} = RD_{MR_D,BS_{DN}}$ is the minimum routing distance between the mobile representative and the corresponding new base station.

Total Rerouting Distance

The total rerouting distance is the sum of path lengths between the new source and destination base station passing through the fixed path connecting the source and the destination mobile representatives.

if $c = d$,

$$RD_{MR_S,BS_{SN}} = RD_{MR_D,BS_{DN}} = \sqrt{2}d \tag{3.1}$$

if $c < d$,

$$RD_{MR_S,BS_{SN}} = RD_{MR_D,BS_{DN}} = d - c + \sqrt{2} \tag{3.2}$$

else

$$RD_{MR_S,BS_{SN}} = RD_{MR_D,BS_{DN}} = c - d + \sqrt{2} \tag{3.3}$$

The total rerouting distance is:

$$\mathrm{TRD_{biswas}(a,d)} = \mathrm{RD_{MR_S,BS_{SN}}} + \mathrm{RD_{MR_D,BS_{DN}}} + \mathrm{RD_{MR_S,MR_{DR}}} \tag{3.4}$$

Cumulative connection path length

The cumulative connection path length is the sum of the lengths of the new paths being established (CLNP) and the sum of the old paths being torn down (CLOP) as the mobile hosts move.

$$\mathrm{CLNP_{biswas}} = \mathrm{CLOP_{biswas}} = \sum_{i=1}^{d} \mathrm{TRD}_{biswas}(a,i) \tag{3.5}$$

and

$$\mathrm{CCPL_{biswas}} = 2\sum_{i=1}^{d}(\mathrm{TRD}_{biswas}(a,i)) \tag{3.6}$$

Number of Connections

For a given connection, at any point in time, there can be at most 5 connections namely - the fixed path between the mobile representatives (1 connection), old paths (to be removed) from the source and the destination base stations to the appropriate mobile representatives (2 connections) and the new paths (established) from the new source and the destination base stations to the appropriate mobile representatives (2 connections).

CBT Strategy

Figure 3.4 depicts the rerouting using the CBT strategy. $\mathrm{MR_S}$ is the source mobile representative and a fixed core based multicast tree is built apriori with $\mathrm{MR_S}$ as the root. The leaves of the tree are the destination mobile representatives named $\mathrm{MR_{D_1}}$ .. $\mathrm{MR_D D_d}$. Let the path connecting the original source and the destination base stations through $\mathrm{MR_S}$ and the base station $\mathrm{MR_{D_1}}$ at the leaf of the core

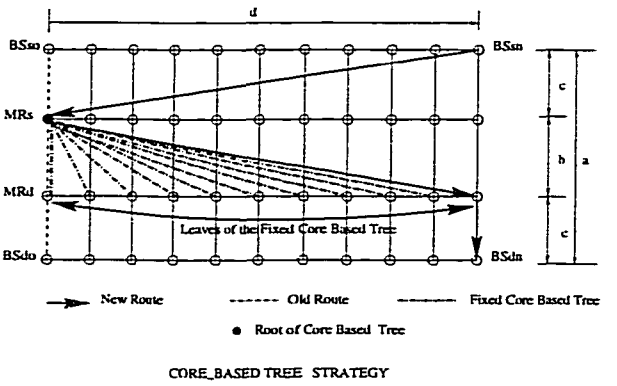


Figure 3.4: Rerouting Distance Calculation: CBT Strategy

based tree. Let the original path between connecting the source mobile representative and the original destination mobile representative be b i.e. $\mathrm{RD}_{\mathrm{MR_S,MR_DR}} = \mathrm{b}$ and $\mathrm{RD}_{\mathrm{MR_S,BS_{SO}}} = \mathrm{RD}_{\mathrm{MR_{D_1},BS_{DO}}} = \mathrm{c}$. Thus, $a = b + 2c$. Now, $\mathrm{RD}_{\mathrm{MR_S,BS_{SN}}} = \mathrm{RD}_{\mathrm{MR_{D_1},BS_{DN}}}$ is the minimum routing distance between the mobile representative and the corresponding new base station.

Total Rerouting Distance

The total rerouting distance is the sum of path lengths between the new source and destination base station passing through the shortest fixed path connecting the source and the destination mobile representatives.

First, we calculate the minimum path length ($\mathrm{RD}_{\mathrm{MR_S,BS_{SN}}}$) between the final source base station $\mathrm{BS_{SN}}$ and the mobile source representative $\mathrm{MR_S}$.

if $c = d$,

$$\mathrm{RD}_{\mathrm{MR_S,BS_{SN}}} = \sqrt{2}\mathrm{d} \tag{3.7}$$

if $c < d$,

$$\mathrm{RD}_{\mathrm{MR_S,BS_{SN}}} = \mathrm{d} - \mathrm{c} + \sqrt{2} \tag{3.8}$$

else

$$\mathrm{RD}_{\mathrm{MR_S,BS_{SN}}} = \mathrm{c} - \mathrm{d} + \sqrt{2} \tag{3.9}$$

The total rerouting distance is the sum of path lengths between the new source and destination base station passing through the shortest fixed path connecting the source and the destination mobile representatives. Now, the total rerouting distance is:

$$\mathrm{TRD_{cbt}(a,d)} = \mathrm{RD}_{\mathrm{MR_S,BS_{SN}}} + \min_{\mathrm{i=0}}^{\mathrm{d}}(\mathrm{RD}_{\mathrm{MR_S,MR_{D_i}}} + \mathrm{RD}_{\mathrm{BS_{DN},MR_{D_i}}}) \tag{3.10}$$

Cumulative connection path length

In this case, since the core based tree is pre-established and it is not torn down, CLOP $= 0$, and

hence,

$$\mathrm{CCPL_{cbt}} = \mathrm{CLNP_{cbt}} = \sum_{i=1}^{d} \mathrm{TRD_{cbt}}(a,i) \tag{3.11}$$

Number of Connections

For a given connection, at any point in time, there can be at most $d+4$ reserved paths namely - the fixed paths between the source mobile representative and the d destination mobile representatives (d connections), old paths (to be removed) (1 connection) and from the source and the destination base stations to the appropriate mobile representatives (2 connections) and the new paths (established) from the new source and the destination base stations to the appropriate mobile representatives (2 connections).

EIA/TIA IS-41's Strategy

In Figure 3.5, the rerouting is done using chaining (cell forwarding).

Total Rerouting Distance

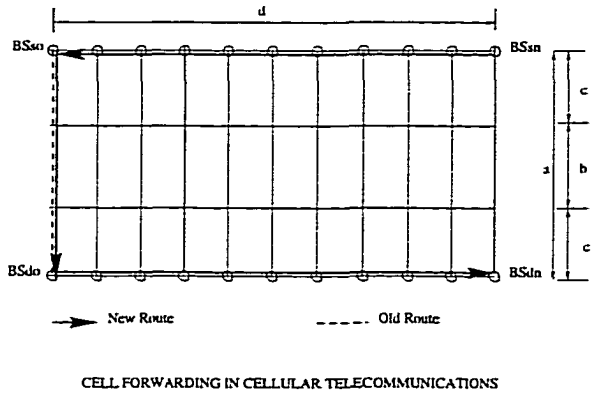


Figure 3.5: Rerouting Distance Calculation: Cellular Telecommunication Strategy

It is easy to see that in this case, we see that the total rerouting distance is:

$$\mathrm{TRD_{cell}(a, d)} = 2\mathrm{d} + \mathrm{a} \tag{3.12}$$

Cumulative connection path length

In case of cell forwarding, we note that there is no need to remove the old paths. As the mobile host moves, a new path extension is added to the original path to extend it. So, CLOP = 0 and CCPL = CLNP.

hence,

$$\mathrm{CCPL_{cell}} = \mathrm{CCNP_{cell}} = a + 2(\sum_{i=1}^{d} 1) = a + 2d \tag{3.13}$$

Number of Connections

When both the mobile hosts move d hops from their original base station, there are $2d + 1$ connections reserved - one for the fixed path connection between the original source and destination base stations, d single hop paths for cell forwarding each for the source and the destination mobile host.

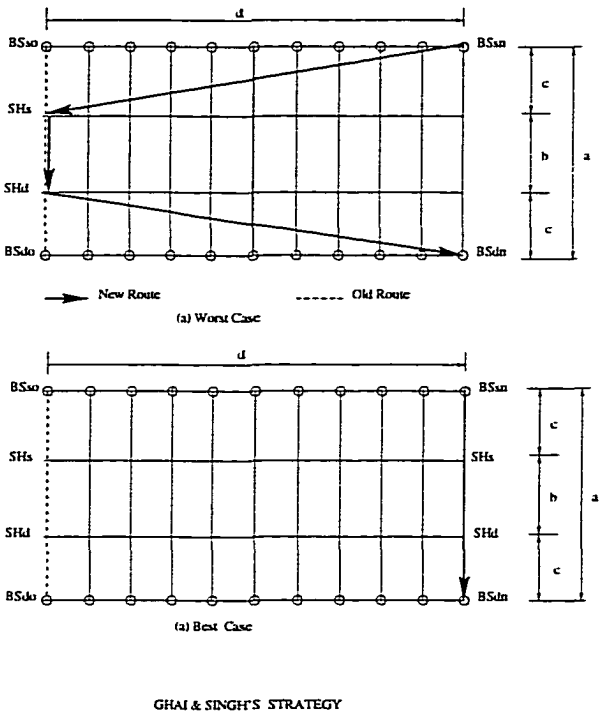


Figure 3.6: Rerouting Distance Calculation: Ghai and Singh's Strategy

Ghai and Singh's Strategy

Consider Figure 3.6 for the calculation of the rerouting distance using Ghai and Singh's Strategy. We assume that the two sets of supervisory hosts are aligned on the two horizontal lines and that each supervisory host dynamically multicasts to the d base stations in the path of the mobile hosts movement. We assume that the distance between the corresponding sets of supervisory hosts is b. The route distance between the old base stations and supervisory host is a minimum of c where $a = b + 2.c$. We consider two cases representing Ghai and Singh's strategy (See Figure 3.6). In both cases, the same dynamic multicast tree is formed for the purpose of rerouting. In the best case, the final rerouting is done through the supervisory hosts that are on the same vertical straight line path as the new (and final) base stations. In the worst

case, the rerouting is done through the path containing the supervisory hosts that are present in the original path connecting the original source and destination base stations.

Best Case: In this case, the route distance between the old (and the new) base stations and their corresponding supervisory hosts is c where $a = b + 2c$. We design the best case with a view to use very few (between 1 to 4) number of supervisory hosts while keeping the total rerouting distance to a minimum.

Total Rerouting Distance

In this case we can see that,

$$\mathrm{TRD}_{\mathrm{ghaibest}}(a, d) = b + 2c = a. \tag{3.14}$$

Cumulative connection path length

In this case, we assume that the multicast tree is in place and old paths are not torn down i.e. CLOP $= 0$. Now, we calculate the CLNP and CCPL for this case. The calculations are the same as Biswas's scheme, and

hence,

$$\mathrm{CCPL}_{\mathrm{ghaibest}} = \mathrm{CLNP}_{\mathrm{ghaibest}} = \sum_{i=1}^{d} \mathrm{TRD}_{\mathrm{biswas}}(a, i) \tag{3.15}$$

Number of Connections

The maximum number of connections is $2d+1$ because there are d connection from the source (resp. destination) supervisory host to the source (resp.) mobile host.

Worst Case: In this, there are only two supervisory hosts (one for the source mobile host and the other for the destination mobile host). This is the same as Biswas's scheme.

Total Rerouting Distance

if $c = d$,

$$\mathrm{RD}_{\mathrm{SH_S,BS_{SN}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DN}}} = \mathrm{RD}_{\mathrm{SH_S,BS_{SO}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DO}}} = \sqrt{2}\mathrm{c} \qquad (3.16)$$

if $c < d$,

$$\mathrm{RD}_{\mathrm{SH_S,BS_{SN}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DN}}} = \mathrm{RD}_{\mathrm{SH_S,BS_{SO}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DO}}} = \mathrm{c} - \mathrm{d} + \sqrt{2} \quad (3.17)$$

else

$$\mathrm{RD}_{\mathrm{SH_S,BS_{SN}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DN}}} = \mathrm{RD}_{\mathrm{SH_S,BS_{SO}}} = \mathrm{RD}_{\mathrm{SH_D,BS_{DO}}} = \mathrm{d} - \mathrm{c} + \sqrt{2} \quad (3.18)$$

For the worst case of Ghai and Singh's scheme, the total rerouting distance is calculated as follows:

$$\mathrm{TRD}_{\mathrm{ghaiworst}}(a, d) = b + \mathrm{RD}_{\mathrm{SH_S,BS_{SN}}} + \mathrm{RD}_{\mathrm{SH_D,BS_{DN}}} \qquad (3.19)$$

Cumulative connection path length

Now, we calculate the CLOP and CLNP and CCPL for this case. We assume that the old paths are not torn down. The calculations are the same as the best case scheme described above as the same connections are formed as the best case.

So,

$$\mathrm{CLNP}_{\mathrm{ghaiworst}} = \mathrm{CCPL}_{\mathrm{ghaiworst}} = (\sum_{i=1}^{d} \mathrm{TRD}_{\mathrm{ghaiworst}}(a, i)) \qquad (3.20)$$

Number of Connections

The maximum number of connections is the same as the best case. The maximum number of connections is $2d + 1$.

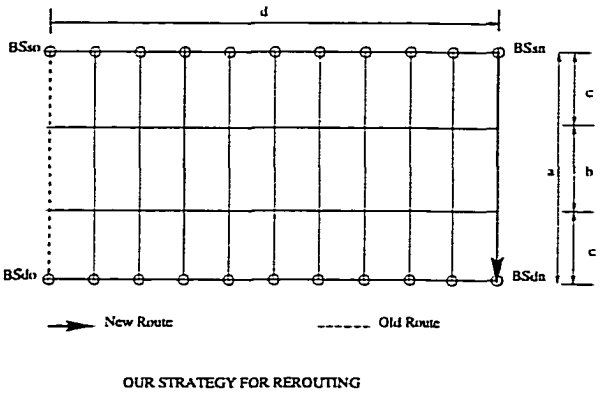


Figure 3.7: Rerouting Distance Calculation: Our proposed Strategy

Our proposed Strategy

Figure 3.7 depicts rerouting distance calculation using our proposed algorithm. In this case, the rerouting distance is the length of optimal route between the source and the destination base stations.

Total Rerouting Distance

In this case, we can see that $\text{TRD}_{\text{racherla}}(a, d) = a$ (See Figure 3.7) . This is because, our algorithm allows for optimal routing connecting the source and destination base stations by the shortest path.

Cumulative connection path length

In order to calculate the CCPL for our proposed scheme, we calculate the CLOP and CLNP. In our case,

$$\text{CLOP}_{\text{ours}} = \text{CLNP}_{\text{ours}} = \sum_{i=0}^{d} a = ad \tag{3.21}$$

hence,

$$\text{CCPL}_{\text{ours}} = 2ad \tag{3.22}$$

Number of Connections

At any during the entire d hop movement of the mobile hosts, there can be at

most 2 connections - the old connection that will be removed and the new connection that has been formed.

3.6.6 Analysis of rerouting distance in the Mobile-Mobile rerouting schemes

We now analyze the performance of the Mobile-Mobile rerouting algorithms described above with reference to the total rerouting distance, the cumulative connection path length and the number of reserved fixed paths. We discuss the performance of each of the metrics separately and draw conclusions based on the results obtained.

It is to be noted for all the discussions below that: in case of Biswas's scheme, the initial path length between the source and destination base station is equal to $a = b + 2c$ where b is the length of the fixed path between the source and destination mobile representatives and c is the length to the path between the initial source (resp. destination) base station and the source (resp. destination) mobile representative. In case of the CBT strategy, the initial path length from the source (resp. destination) base station to the source (resp. initial destination) mobile representative is equal to c and the path length between the initial pair of source and destination mobile representative is b. Similarly, in case of Ghai and Singh's scheme, b is the distance between the old (resp. new) source and destination supervisory hosts and c is the length of the path between the initial source (resp. destination) base station and the source (resp. destination) supervisory host. In each of the sub-figures, we specify the length of the initial path a and the path lengths b and c (specific to Biswas and Ghai and Singh's scheme). For all the metrics that we discuss below, we plot

graphs showing the effect of moving distance (d) on the total rerouting distance. The moving distance is varied from 1 to 25. We denote Biswas's scheme, the CBT scheme, Ghai and Singh's worst and best schemes, the IS-41 cellular scheme and our proposed scheme using the legends Biswas, CBT, Ghai (worst), Ghai (best), Cell and Ours respectively.

Total Rerouting Distance

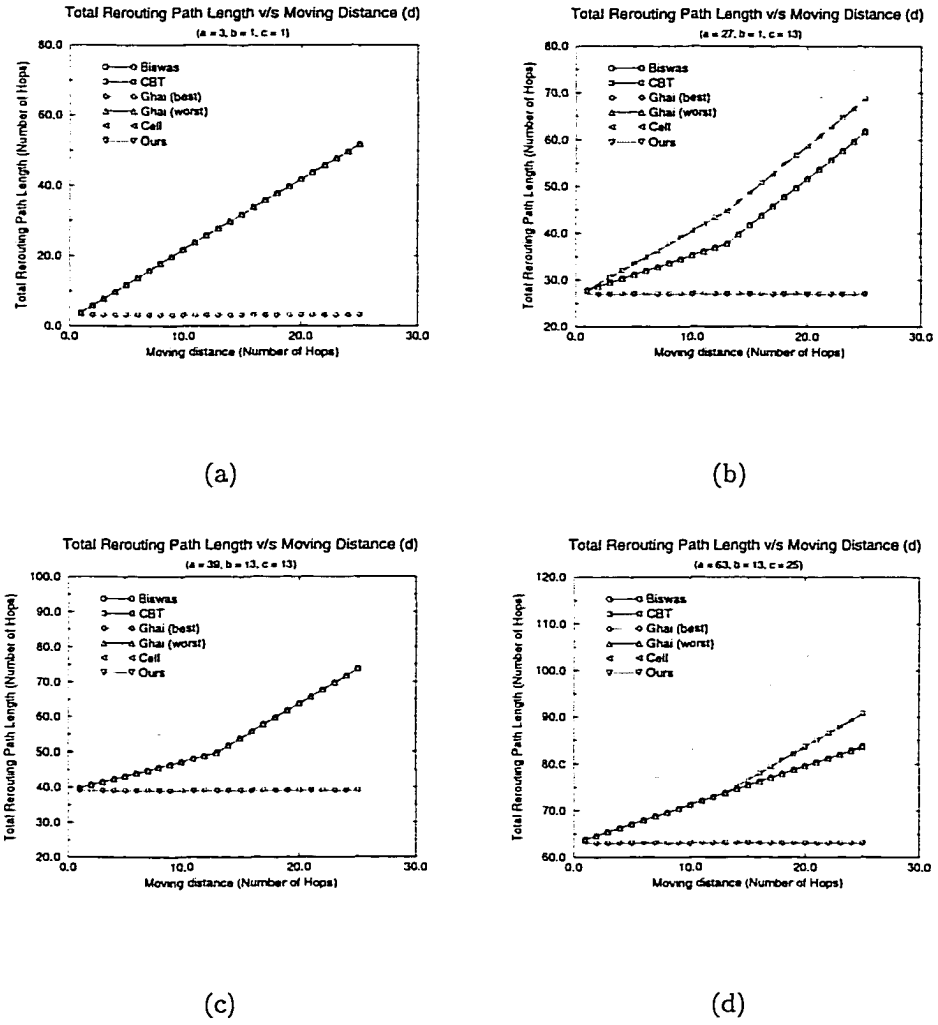


Figure 3.8: Sub-figures (a) (b), (c) and (d) depicts the effect of the moving distance on the Total Rerouting Length for various rerouting schemes.

Figure 3.8 show the total rerouting distance as a function of mobile host movement distance (d) for the five rerouting schemes described above. The figure contains four sub-figures the plots for various values of the initial path length a ($a = 3$, 27, 39, 63).

From these figures, we make the following observations:

- Our proposed scheme and Ghai's (best) scheme perform the best because these schemes setup the optimal route. Cell forwarding performs the worst with increasing values of d because of its naive approach of cell forwarding continuously. The performance of Ghai's (worst) scheme, the CBT scheme and Biswas's schemes fall in between.

- The rerouting distance for our proposed scheme and Ghai (best) is optimal and for the given topology (crossbar grid) is a constant and does not vary with increasing values of d. For all the other schemes, the rerouting distance increases with increasing values of d.

- For Biswas's scheme, it can been seen that total rerouting length depends on the length of the initial fixed path. If this initial fixed path length $b = a$ ($c = 0$), then, Biswas's scheme is the exact same scheme as cell forwarding.

- Biswas's strategy is a specialized case of the CBT strategy wherein the core based tree is replaced by a simple path. It is better to use Biswas's strategy than the CBT strategy in order to reduce the total rerouting distance, if $c \geq b$. In case, $b > c$, it is better to use the CBT strategy.

- For Ghai's (worst) scheme, the CBT scheme and Biswas's scheme, the rerouting distance increases linearly with the increasing values of d. However, the

rerouting distance improves when $c = d$. This is intuitive because when $c = d$, the new (and shortest) route is along the slanted edges.

Cumulative connection path length

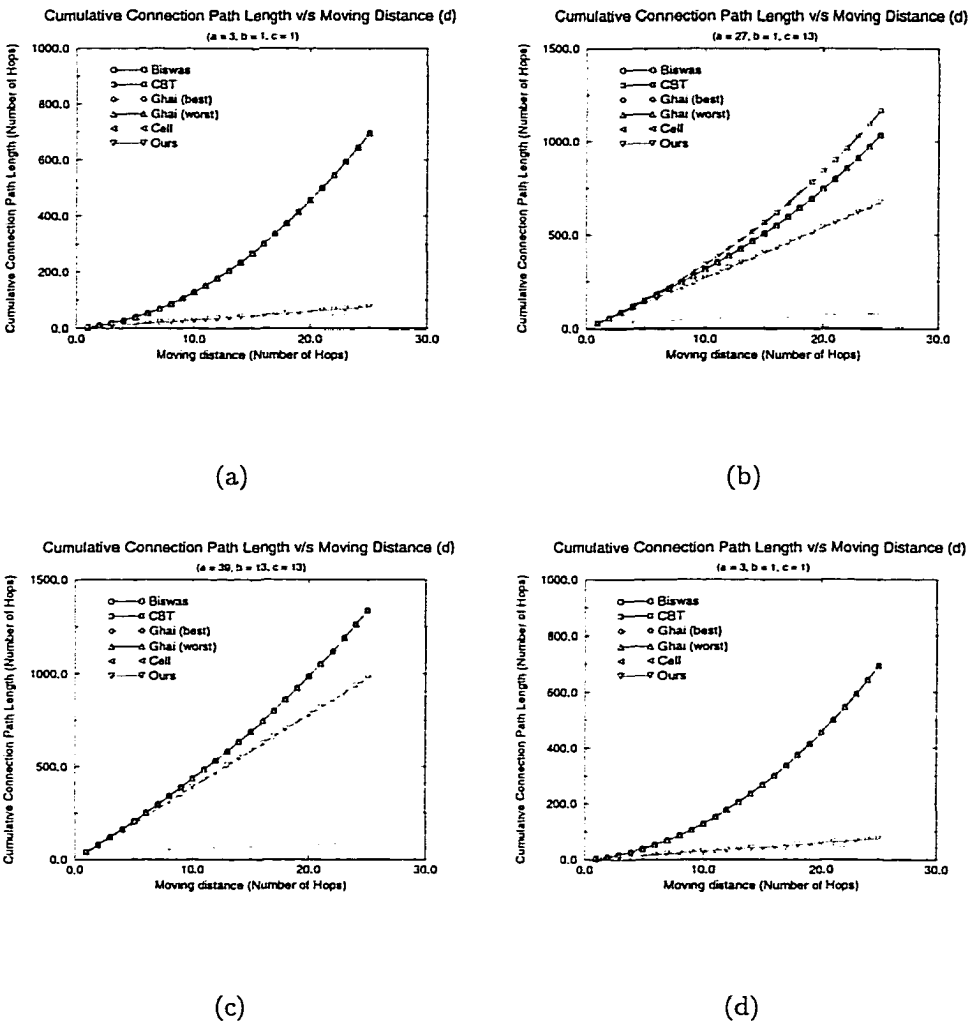


Figure 3.9: Sub-figures (a) (b), (c) and (d) depicts the effect of the moving distance on the Cumulative Connection Path Length for various rerouting schemes.

In Figure 3.9, we show the cumulative connection path length as a function of mobile host movement distance (d) for the rerouting schemes described above. The figure contains four sub-figures which are plots of the total rerouting distance for various values of the initial path length a (a = 3, 27, 39, 63).

From this figure, we make the following observations:

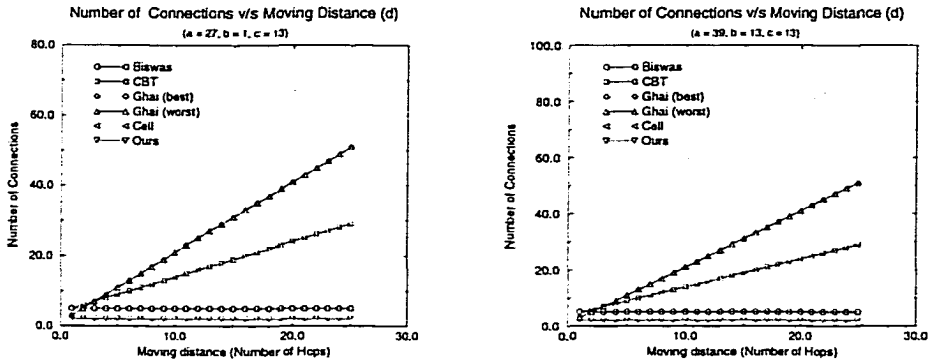
- The cell forwarding scheme does the best because there is no need to remove any of the old paths and every move of a mobile host results in a unit length increase in the cumulative path.

- Our strategy performs the next best as the cumulative path length is increased by a (the original path length between the source and the destination base stations) every time the mobile moves. This is because for every move a new optimal path of length a has to be established. This is true because from our initial assumption that the source and the destination mobile hosts start moving at the same time and move at the same speed.

- In case of all the other schemes, the cumulative path length is much higher and grows exponentially. This can be seen from our derivation for the formulae for the CCPL of these schemes. The CCPL for d moves in each of these cases is a sum of all the total rerouting distance values for all moves 1 .. d.

- We see again that, it is better to use Biswas's strategy than the CBT strategy in order to reduce the cumulative path length, if $c \geq b$. In case, $b > c$, it is better to use the CBT strategy.

Number of Connections



(a) (b)

Figure 3.10: Sub-figures (a), (b) depict the effect of the moving distance on the Number of Connections established or torn down for various rerouting schemes.

In Figure 3.10, we show the number of connections as a function of mobile host movement distance (d) for the rerouting schemes described above. The figure contains two plots for of the initial path length (a) values of 3 and 39. From this figure, we observe that:

- The number of connections is minimum and constant (equal to 2) in case of our strategy. Biswas's strategy also requires a constant number of connections (5).

- The number of connections for the other schemes is dependent on the moving distance. Among these remaining schemes, the CBT strategy requires the least number of connections. Ghai's (best) and Ghai's (worst) schemes and the cell forwarding schemes require the same number of connections and perform the worst.

From the observations of the performance of all these rerouting schemes with

respect to the metrics, we conclude that our proposed scheme performs the best as far as the total rerouting distance and the number of connections are concerned. Ghai's (best) scheme performs as well as our proposed scheme in terms of the total rerouting distance. It however requires a lot more number of connections to be established and incurs more cumulative cost in establishing connections. Also in the worst case, the routes for Ghai and Singh's schemes are sub-optimal. In addition, Ghai and Singh's schemes require specialized supervisory hosts and a tree based network for optimal performance. Cell forwarding performs the best with respect to the cumulative cost incurred in establishing connections. However, cell forwarding requires a larger number of connections and its total rerouting distance is larger than our proposed scheme. The CBT scheme is a generalized case of Biswas's scheme and it requires more resources and cost. Its performance advantage in terms of total rerouting distance over Biswas's scheme is highly dependent on the underlying topology and the structure of the core based tree.

3.7 Summary

In this chapter, we presented an algorithmic framework for rerouting to accommodate unlimited movement of mobile hosts that form the ends of a Mobile-Mobile connection. Our proposed framework overcomes the drawbacks of the other Mobile-Mobile rerouting schemes by making the rerouting source-initiated and by rejecting the establishment of incorrect paths. We also proved the correctness of our proposed framework with respect to rerouting. In addition, we also compared our proposed algorithm with the other known Mobile-Mobile rerouting schemes using several per-

formance metrics. The performance analysis indicates that our proposed algorithm is superior to other schemes.

Future work includes increasing the array of performance metrics, Finally, it would be interesting to show how our framework can be applied to a wireless ATM environment.

Chapter 4

Effect of rerouting on the performance of wireless TCP

4.1 Introduction to TCP

TCP (Transmission Control Protocol) [136] is a protocol used along with the Internet Protocol (IP) to send data in the form of message units between computers over the Internet. While IP takes care of handling the actual delivery of the data, TCP takes care of keeping track of the individual uuits of data (called packets) that a message is divided into for efficient routing through the Internet. For example, when an HTML file is sent to you from a Web server, the Transmission Control Protocol (TCP) program layer in that server divides the file into one or more packets, numbers the packets, and then forwards them individually to the IP program layer. Although each packet has the same destination IP address, it may get routed differently through the network. At the other end (the client program in your computer), TCP re-assembles

the individual packets and waits until they have arrived to forward them to you as a single file. TCP is known as a connection-oriented protocol, which means that a connection is established and maintained until such time as the message or messages to be exchanged by the application programs at each end have been exchanged. TCP is responsible for ensuring that a message is divided into the packets that IP manages and for re-assembling the packets back into the complete message at the other end. In the Open Systems Interconnection (OSI) communication model, TCP is in layer 4, the Transport Layer.

TCP is a means for building a end-to-end reliable byte stream on top of the unreliable packet Internet Protocol (IP). TCP is the protocol that supports nearly all Internet applications. The combination of TCP and IP is referred to as TCP/IP and many people imagine, incorrectly, that TCP/IP is a single protocol. The basic method of operation involves:

- wrapping higher level application data in segments
- wrapping the segments into IP datagrams
- associating port numbers with particular applications
- associating a sequence number with every byte in the data stream
- exchanging special segments to start up and close down a data flow between two hosts
- using acknowledgments timeouts to ensure the integrity of the data flow

Every byte that is transmitted from the source TCP entity has a 32 bit sequence number. The sequence number is used for acknowledgements and for managing

the windowing mechanism. The TCP sender and the receiver communicate using segment. The segment size can vary but is limited to a size of 64KB. The segment size is also limited by the underlying network's maximum transfer unit (MTU) which is the maximum size of any packet that traverse over the network.

TCP entities use the *sliding window protocol* [152] to communicate. The sender starts a timer when it transmits a segment. When the segment is received by the receiver, it sends an acknowledgment with a acknowledgment number that equals the next segment sequence number that it expects to receive. If the sender gets the acknowledgement before the timer expires then it continues the sending process with the next segment otherwise it re-transmits the segment again. To complicate the situation, segments and acknowledgments may be lost or may arrive out of order. Also, the segments may arrive late because of network congestion. TCP deals with all these problems using an adaptive transmission policy and intelligent flow and congestion control.

The TCP receiver, when sending an acknowledgment, also sends the amount of buffer size (called the *receiver window*) that it has available to receive new data. The TCP sender uses this information to control the amount that it sends to the receiver. TCP adapts its transmission to deal with congestion that may occur in the network while increasing trying to increase the system throughput. In order to deal with congestion, the sender uses and maintains two windows - the receiver window and the *congestion window*. The size of sender's data is the minimum of the these two window sizes. The congestion window is indicative of the size of the that the sender thinks it can use to send without causing congestion in the network and thus not getting the acknowledgement in time. When a TCP connection is established,

the congestion window is initialized to the maximum segment size in use on the connection. If the segment is acknowledged before the expiration of the timer, the congestion window is doubled. This exponential increase of the congestion window is called as the *slow-start algorithm.* The TCP protocol uses a third parameter called the *threshold* (initially set to 64K) to ebb the exponential increase of the congestion window. When the congestion window equals the threshold, the congestion window is increased linearly. In case, there is a re-transmission timer timeout, the congestion window size is re-set to one segment and the threshold is halved. In addition,the TCP re-transmission time-out value is doubled. This doubling is known as Karn's re-transmission timer backoff [113].

4.2 TCP in Mobile Networks

TCP is a one of the most widely used transport layer protocol. In a mobile environment, TCP cannot discriminate packets dropped due to lossy links, intermittent connectivity, handoffs and rerouting with those dropped due to congestion in the network. TCP reacts to packet losses as it would in a wired environment – it drops its transmission window size before retransmitting packets, initiates congestion control or avoidance mechanisms like the slow start algorithm and re-setting its retransmission timer [32, 33]. This results in an unnecessary degradation in system performance. In the past, several proposals have been made to alleviate this problem [32, 33]. However, these proposals do not take into account the type of rerouting viz. full rerouting, partial rerouting, tree based rerouting or cell forwarding [139, 137]. Also, these proposals assume that only one end in a session is mobile while the other

is fixed.

Proposed TCP schemes for mobile networks can be classified under the following categories [32]:

- **Link-Level protocols:** These protocols hide the packet loss on the wireless link by performing local retransmission at the link-level and forward error correction over the wireless link. The two main approaches adopted by these protocols include forward error correction (FEC) and retransmission in response to Automatic Repeat Request (ARQ). Examples of link-level protocols tailored for wireless links include the AIRMAIL [20]. If the link level protocol uses the TCP acknowledgments effectively without generating its own acknowledgment some overhead is minimized. This phenomenon is called as link-level TCP awareness. the snoop protocol [33] is an example of such a scheme. In case of the snoop protocol, a snoop module at the base station caches packets and retransmits if thinks that the data has been lost or corrupted while being transmitted on the wireless link.

- **Split connection:** The TCP connection is broken at the base station: a regular TCP connection between the stationary host and the base station and while a modified TCP connection is established between the mobile host and the base station. In doing so, data flow on the wireless link is isolated from the flow on the wired connection. Indirect-TCP [30] is an example of the split-connection approach.

- **Fast Retransmit:** The receiver sends a certain number of duplicate acknowledgments informing the sender the packet loss is not due to congestion. The

sender then does a fast retransmission of the packet without resorting to congestion control measures.

- **Explicit Loss Notification**: Explicit acknowledgments from the base station are sent to indicate whether or not this loss because of congestion.

- **End to End Protocols** These schemes use TCP connections between the source and the mobile host. However, in order to make the TCP sender handle packet losses, they use either selective acknowledgments (SACKs) to allow the sender to recover from multiple packet losses in a window or an explicit loss notification (ELN) to inform the sender of the packet losses explicitly. TCP Reno, TCP Reno with SACK and TCP Reno with SACK and SMART [115] are examples of end to end protocols. In the SMART scheme, acknowledgments contain the cumulative acknowledgement and the sequence number of the packet that caused the receiver to generate the acknowledgement.

4.3 Effect of Rerouting Strategy on Wireless TCP scheme

It is to be noted that TCP is a transport layer protocol while the rerouting is a network layer function and hence are independent of each other. The aim of this work is to study the network performance with TCP under different rerouting schemes with either one end or both the ends in a session being mobile using network simulation.

Within the context of communication based on TCP in mobile networks, rerout-

ing is an important issue to consider. It is to be noted that TCP is a transport layer protocol while the rerouting is a network layer function and hence wireless TCP scheme and rerouting process are independent of each other. Rerouting involves setting up new TCP connections in order to restart data transfer between the source and the destination entities. Rerouting involves four stages. In the first stage, new connections are established as required after the handoff is complete and the mobile hosts registers with the new base station. In the second stage, the packets that have been transferred from the source to the old base station are forwarded to the new base station upon request from the old base station. In the third stage, the new base station requests the source to reroute data to the new base station on the newly established routes. In the final stage, the old route between the old base station and the source is torn down. It is to be noted that the second stage in the rerouting process is optional. If the packets stored at the old base station are not forwarded, the destination will request the source to retransmit those specific packets. Thus, rerouting may or may not involve forwarding packets from the old base station to the new base station.

In this chapter, we study the split-connection and the snoop protocol TCP under full, partial, cell forwarding and virtual tree rerouting using simulation with a view to compare and contrast the performance of these two wireless TCP schemes under all of the above mentioned rerouting schemes. Various performance metrics like the system throughput, total rerouting time, service disruption time, amount of buffering required at the mobile hosts and the base stations are evaluated for comparative analysis using simulation. We also propose measures and mechanisms to improve the network performance using TCP under these rerouting schemes after analyzing

the performance of the various Wireless TCP schemes depending on the type of the rerouting strategy used.

The rest of the chapter is organized as follows: we study the split connection protocol and the snoop protocol in more detail in Section 4.4. The simulation setup, our assumptions, the experiments conducted and the results are presented in detail in Section 4.5. We analyze the results obtained from simulation in Section 4.6. It is to be noted that TCP is a transport layer protocol while the rerouting is a network layer function and hence are independent of each other. Various performance metrics like the system throughput, end-to-end delay and service disruption time evaluated for comparative analysis using simulation. Finally in Section 4.7, we summarize our contributions.

4.4 Wireless TCP schemes - Split Connection Protocol and Snoop Protocol

As mentioned earlier, several schemes have been proposed by researchers to adapt TCP to wireless and mobile networks. Among the most popular of them are the split-connection and the snoop protocol. In this section, we will take a detailed look at these schemes.

4.4.1 Split Connection Protocol

I-TCP [30] is an example of the split connection protocol. It was one of the first wireless TCP schemes to be proposed by researchers. The basic idea of the approach is to split the TCP connection at the base station in the last hop into two separate

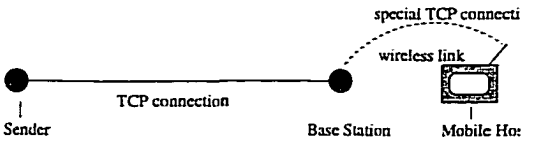


Figure 4.1: The Split Connection Approach

connections. The first connection, from the source (assumed to be a fixed host) to the base station, is a regular TCP connection. There is a "TCP-like" connection on the last hop that is tuned to wireless link. During a handoff, the I-TCP connections have to be moved from the current base station to a new one. This is done by a state transfer of the two connections from the current base station to the new base station. Figure 4.1 depicts the split connection protocol overview. The figure shows how the end-to-end TCP connection between the source and the mobile host is split at the base station.

The I-TCP allows to have separate (and optimized) congestion and flow control and transmission policies for the two connections and thus achieves good throughput. However, the approach has several drawbacks including - the loss of end-to-end semantics; need for re-linking of applications with the I-TCP library; additional software overhead of traversing through the TCP protocol stack four times and large handoff latencies.

4.4.2 Snoop Protocol

The snoop protocol is a TCP-aware link-level protocol. The basis of the protocol is the introduction of a software snoop agent at the base station at the last hop. This snoop agents sniffs the packets passing through the TCP connection and stores the

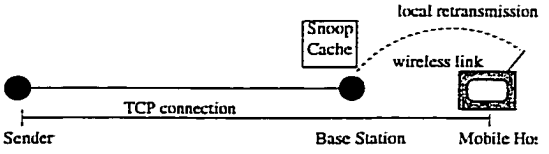


Figure 4.2: The Snoop Protocol

TCP segments that have been sent to the mobile host but have not been acknowledged by the mobile host in a cache. In case the snoop sees a certain number of duplicate acknowledgments from the receiver or in case the snoop's re-transmission timer expires, it realizes that there has been a packet loss. In case of a packet loss, the snoop agent re-transmits the packet (from its cache) to the mobile host. It also suppresses the duplicate acknowledgment. Figure 4.2 shows the general idea behind the snoop protocol. The snoop agent needs to maintain a cache at the base station and perform local TCP-aware link-level retransmissions.

The snoop protocol has the advantage of suppressing the duplicate acknowledgments for lost TCP segment and locally retransmitting the packets. By doing so, invocation of congestion control algorithms and fast retransmissions by the sender is avoided. The disadvantages of the snoop protocol includes the overhead for the deploying the snoop agent, the time required for snooping and the requirement for a snoop cache. In addition, like the other link-level algorithms, it can not complete shield the sender from packet loss on the wireless link.

4.5 Simulation Details

In this section, we present the details of the simulation of the split connection and the snoop protocol under different rerouting conditions. We present our assumptions,

the simulation details including input and system parameters, the experiment setup and the statistics and results gathered after running the simulation experiments. The simulation is done using the PARSEC simulator developed by researchers at UCLA. PARSEC is a discrete-event simulator that allows processes and systems to be modeled as entities that can interact with by passing messages to each other. We now describe the details of the simulation including the assumptions, network topology used to study, parameters used for simulation, the simulation experiments, performance metrics being evaluated and the results.

4.5.1 Assumptions

We assume the following under our simulation model:

- The wired links do not have any transmission and receiving errors while the wireless links errors are produced randomly using a uniform distribution.
- The wired links do not have any transmission and receiving errors while the wireless links errors are produced randomly using a uniform distribution.

The wired links do not have any transmission and receiving errors while the wireless links errors are produced randomly using a uniform distribution.

The mobile hosts uses radio hints to realize that it is moving into the cell of the next base station. Also, we assume that the mobile host's path of movement has complete coverage.

4.5.2 Network Topology

The network topology used for the simulation experiments is shown in Figure 4.3. The topology is a tree with the source node at the root of the tree. We chose this topology so that we could experiment with the same topology for all the rerouting schemes including the virtual tree rerouting which requires the network topology to be a tree. Also, this network topology models the real word case with a source (say a video server) broadcasting to a mobile user (using a video viewing device like a palmtop). It is to be noted that there is a path containing 12 nodes from the source node to each of node 1 and node 2. The tree consists of wired links (155 Mb/s bandwidth, 0.46 ms latency) while the last hop is on a wireless link (2Mb/s bandwidth, 2ms latency). The mobile host moves to and fro between the base stations located the leaf nodes of the network starting with the leftmost leaf node (node 7). The mobile host moves every 3 seconds (See Table 4.1). We assumed that the motion of the mobiles is a linear motion. This type of motion is typical of movement in buildings and has been obtained from [10, 148].

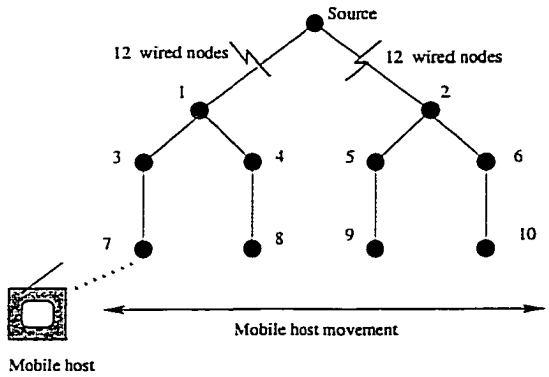


Figure 4.3: Network Topology for Simulation

4.5.3 Simulation Parameters

The parameters used in the simulation are shown below in the Tables 4.1 and 4.2. Table 4.1shows the network parameters including the bandwidth and the latency of the wired and wireless links. Table 4.2 shows the TCP parameters including the window sizes, thresholds and the buffer sizes of the base station and the mobile host. The maximum simulation time is 16.5 seconds.

Bandwidth of Wired Links	155 Mb/s
Bandwidth of Wireless Links	2 Mb/s
Latency of Wired Links	50 μs
Latency of Wireless Links	2 ms
Wireless Bit Error Rate	0.04253
Buffer Size at Base Stations	256 KB
Buffer Size at Mobile Host	64 KB
Frequency of Mobile's Movement	Every 3s

Table 4.1: Network and Mobility Parameters

4.5.4 Simulation Experiments

We simulated the split connection TCP and the snoop protocol TCP for full, partial, cell forwarding and the virtual tree rerouting schemes using the set of parameters specified above on the network topology described above. We will refer to these schemes as full (split), partial (split), cell forward (split), tree (split), full (snoop), partial (snoop), cell forward (snoop), tree (snoop). The simulation uses only a single

TCP Fast Timer for Snoop	200 ms
TCP Slow Timer	500 ms
TCP Max Segment Size	1460 B
TCP Control Packet	50 B
TCP Initial Threshold	32 KB
TCP Maximum Threshold	64 KB
Initial Congestion Window	1 KB

Table 4.2: TCP Parameters

mobile host that acts as the receiver of the data being transmitted from the source. We ran the experiments for varying values of the simulation time (4.5 sec, 7.5 sec, 10.46 sec, 13.5 sec, and 16.5 sec) and calculated the following performance metrics.

- Source Throughput (KB/s): The ratio of the amount of data sent by the source to the duration of the time that the source sends all its packets.
- Average Round Trip Packet Delay ((s/KB): The average of all the round trip packet delays (from source to mobile host and back) per KB of data.
- Source Disruption Time ((s): The duration of the time that the source has been explicitly disallowed from transmitting on the downlink to the mobile host.

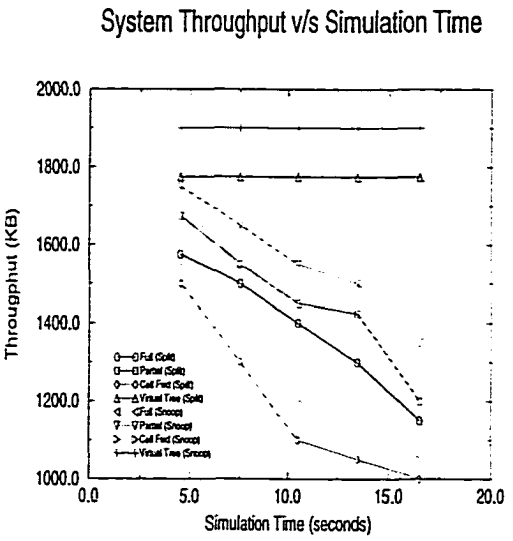


Figure 4.4: Variation of the source throughput for various simulation times

4.5.5 Simulation Results

The results from the simulation are shown in the Figures 4.4, 4.5 and 4.6. Figure 4.4, 4.5 and 4.6 show the variation of throughput average round trip packet delay, and source disruption time respectively for various simulation times for each of the wireless TCP with rerouting cases.

4.6 Analysis Of Results

We now analyze the performance of the various schemes based on the results of our simulation (See Figures 4.4, 4.5 and 4.6). In case of source throughput and average round trip packet delay (Figures 4.4 and 4.5), we see that the snoop protocol TCP performs better the split-connection TCP for all the rerouting schemes. This is because of the faster re-transmissions by the snoop protocol TCP on the wireless link as compared to the split-connection TCP. Also, the split - connection TCP has

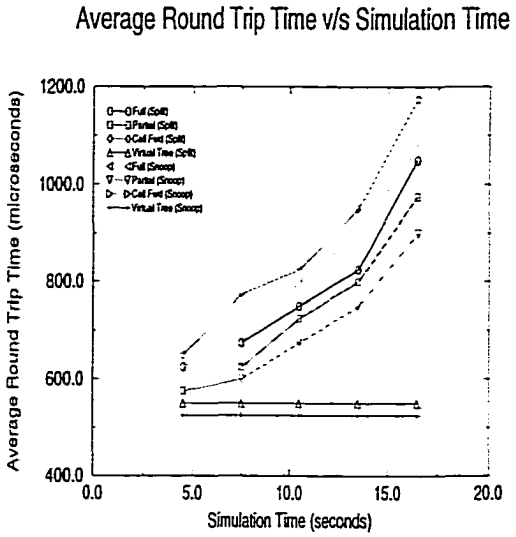


Figure 4.5: Variation of the average round trip packet delay for various simulation times

the additional onus of creating two separate TCP connections as opposed on a single TCP connection in case of the snoop protocol TCP. Among the rerouting schemes, the virtual tree rerouting performs the best for both the wireless TCP schemes. This is so because in case of the virtual rerouting, the source simply multicasts the messages to each of the base stations at the leaf nodes of the network. There is no need for forwarding packets from the old base station to the new base station. The cell forwarding performs the worst, as it needs to make TCP connections for forwarding packets from the old base station to the new base station every time the mobile host moves. This results in a very inefficient routing. For example, in case of cell forwarding, when the mobile host moves from node 7 to node 8 to node 9 to node 10, it needs to establish TCP connections from node 7 to node 8 and from node 8 to node 9 and from node 9 to node 10. Thus, in case of cell forwarding the number of wired TCP connections is proportional to the number of moves while

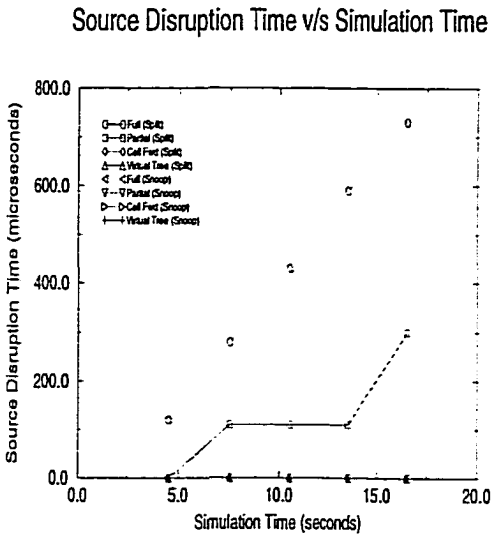


Figure 4.6: Variation of the source disruption time for various simulation times

in case of virtual tree routing, there is effectively only one TCP connection for communication between the source and the mobile host. In case of full and partial rerouting, there are two wired TCP connections - one for forwarding packets from the old base station to the new base station and a connection from the source to the new base station (either directly or through a cross over point). The partial rerouting performs better than the full rerouting as it effectively utilizes the packets sent by the source that stored at the cross over point to forward to the new base station. As far as source disruption time is concerned, only the full rerouting and partial rerouting have explicit source disruption. This happens because the new base station explicitly forbids the source from starting transmission before it gets all the packets from the old base station. Full rerouting 's source disruption time is proportional to the number of moves while in case of partial rerouting, source disruption occurs when the cross over point is the source node.

4.7 Summary

In this chapter, we have studied the performance of two of the important wireless TCP schemes - split connection TCP and the snoop protocol TCP under full, partial, cell forwarding and virtual tree rerouting conditions using detailed packet-level simulation. We find that the snoop protocol with virtual tree rerouting performs the best while split-connection based cell forwarding performs the worst with respect with throughput, disruption time and round-trip delay. We plan to extend this work by extending the comparison to other wireless retransmission schemes like link layer retransmission, explicit congestion notification and fast retransmissions. In addition, we plan to study how wireless TCP performs when both the source and the destination are mobile hosts.

Chapter 5

A Novel Routing Algorithm for Ad-hoc networks

5.1 Introduction

Due to the increasing number of mobile computers and networking hardware, extensive work has been done in communication protocols in the mobile environment using a fixed, backbone network infrastructure. There are situations when such a backbone network may not be present. Examples of this include scenarios such as (i) a group of workers deployed quickly during a natural disaster, (ii) business delegates assembled in a lecture hall, and (iii) exchanging information using wearable computers [83]. It is desirable for mobile hosts in such situations to form a temporary network without the aid of centralized administration or an established network backbone. This temporary network is called an *ad-hoc* network.

Some of the real-life examples of ad-hoc networks are as follows:

1. **ARPANET Packet Radio Network (PRN)**: The ARPANET Packet Radio Network (PRN) [112] is the earliest deployment of an ad-hoc network. This network has been in existence for the past 25 years and has been the testbed for several new ad-hoc and wireless technologies. The network consists of repeaters, terminals and stations. Stations can be used to connect to any static network. The terminals are the mobile hosts used by users. The repeaters are used to forward and route data packets. All components in the PRN can be mobile.

2. **The Survivable Adaptive Networks (SURAN)**: SURAN [122] is the successor of the ARPANET PRN. It aims to have an intelligent network that can survive and adapt in view of node and link failures.

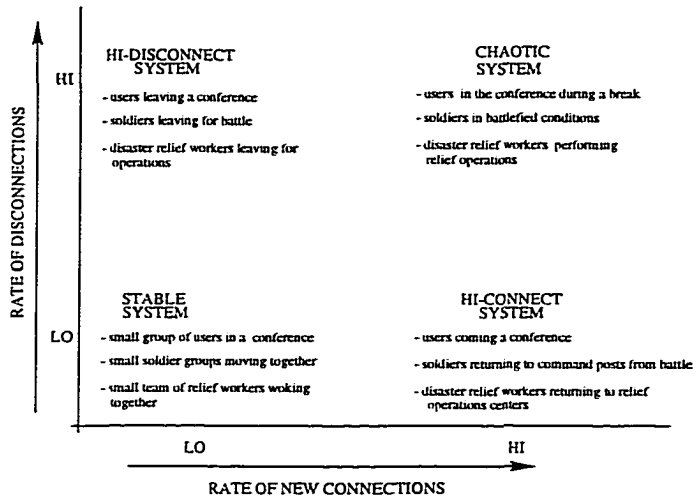
3. **Mobile Conventional Computer Supported Co-Operative Work (CSCW)**: Japanese researchers have developed the Mobile CSCW [158] environment that allows researchers to use cameras, microphone and multimedia workstations to collaborate using ad-hoc wireless networks. The environment supports sharing of data and programs and allows users to refine their work through discussion over the ad-hoc network. The Mobile CSCW environment also allows nomadic collaborative computing.

4. **Motorola Piano Platform**: The Motorola Piano Platform [99] provides short range, wireless connectivity at high bandwidth to a variety of mobile devices. It creates spontaneous, ad-hoc networks between a wide variety of common devices. The Piano platform aims to allows heterogeneous ad-hoc networks by enabling short-range, low-power, unlicensed, wireless connectivity

from the handheld PDAs to a myriad of other devices including PCs, consumer electronics, point-of-sale transaction terminals, cellphones, pagers, Internet proxies, and other electronic devices.

An ad-hoc network consists of mobile hosts that communicate over a wireless channel and is characterized by a dynamic topology. In some cases the topology of the network remains stable after an initial setup period, for example, once the business delegates are seated around a table or in their respective rooms, their laptops may be moved fairly infrequently. On the other hand, there are a number of cases where the network topology might experience rapid changes at least in certain parts. Such examples include: (i) a team of robots employed to build a map of a building where the connectivity change as robots move in and out of various rooms, and (ii) a team of rescuers equipped with wearable computers searching for person in a rugged terrain. In these examples, the information from various robots of rescuers must be exchanged so that the overall mission can be carried out effectively. Our main focus is the latter kind of networks, where the topology might be stable for brief periods of times in certain localities but in the other regions the topology can be highly dynamic. Using a brute force flooding algorithm will result in inefficient use of the bandwidth and buffers.

Figure 5.1 shows how different stages of ad-hoc network applications can be classified based on the rate of connections and disconnections while giving supporting examples. Applications stages (or phases) can be *static* (low rate of disconnections, low rate of connections), *high connection* (low rate of disconnections, high rate of connections), *high disconnection* (high rate of disconnections, low rate of connections) or *highly dynamic* (high rate of disconnections, high rate of connections). The



TAXONOMY OF AD-HOC NETWORK APPLICATIONS
BASED ON RATE OF CONNECTIONS AND DISCONNECTIONS

Figure 5.1: Taxonomy of Stages in ad-hoc applications based on the rate of connections and disconnections

same application can be categorized under any one of these schemes depending the stage in the life time of the application. For example, as depicted as one of the examples in the Figure 5.1, consider a military battlefield environment. Initially, the soldiers (each carrying a personal digital assistant for communication form an ad-hoc network) are briefed at the field headquarters, the connectivity between the users is high as they all are in close physical proximity and the rate of disconnections is low as the soldiers remain there till the briefing is over. So, by our classification, this stage is a high connection phase. When the soldiers move in small groups, the rate of connection is small as there are not radio range with all the other soldiers and the rate of disconnection is low because the soldiers remain in the same group moving together. This stage is a categorized as a static phase of the application. Similarly, the application can be either in a high disconnection phase or a highly

dynamic phase (See Figure 5.1).

Routing protocols in ad-hoc networks can be classified either as *reactive* or *pro-active* [91, 126]. Proactive schemes make use of routing tables at nodes which are kept updated as the network changes. Thus in pro-active schemes whenever a packet is to be routed, the routing path is determined using the routing tables. Algorithms based on distance vector [135] and link state [103, 130] belong to the class of pro-active schemes. Reactive routing protocols determine routes on demand. Flooding is the classical example that belongs to the reactive schemes. The dynamic source routing algorithm by Johnson and Maltz [105, 108] uses flooding effectively to route packets on demand. Another example of a reactive algorithm is called TORA [126, 64] which is based on flooding. TORA reduces the number of messages by intelligently eliminating the sending of duplicate copies of the messages in the network.

5.1.1 Comparison with Previous Work

In a network that is highly dynamic, the amount of information that needs to be passed across the network to maintain the routing table is enormously high. Given the limited bandwidth of mobile networks this will cause serious traffic bottlenecks. Furthermore, maintenance of the routing table at each node can grow as the network grows and may strain the limited storage available at the node. Reactive algorithms overcome the above deficiencies but use excessive amount of messages to determine the route. In this chapter, we propose a routing protocol that is reactive but uses considerably less number of messages. We achieve this by maintaining a forest of spanning trees. The routing algorithms in [105, 108] have high space, time and message complexities and suffer from serious flaws as pointed out by Das and

Bharghavan [66]. The distance-vector routing algorithm in [135] suffers from synchronization problems. The minimum connected dominating set (MCDS) routing algorithm in [66, 67] seems to eliminate some of the above problems but uses some impractical assumptions with regard to partitioning networks. The Signal Stability based Adaptive Routing protocol (SSA)[75] discovers routes on-demand based on signal strength and location stability. While the SSA algorithm is simple, it requires additional work in maintaining information about relative signal strength of messages. Akyildiz, Pelech and Yener [12] have proposed a hierarchical multihop routing algorithm for fully dynamic networks. The routing algorithm operates on a virtual topology obtained by partitioning routing information for the mobile terminals from a hierarchical distributed database.

We propose DST (Distributed Spanning Tree Routing) - an efficient routing protocol for ad-hoc networks that discovers routes on demand but uses considerably less number of messages. We achieve this by maintaining a forest of spanning trees in a distributed fashion. We believe our approach is practical and has less time, space and message complexities when compared with the above algorithms. One of the important design issues we had in mind was to keep the algorithm power-aware [147], that is, try to conserve the battery power of the mobile hosts. This is significant because battery power technology has not been able to pace up with processing and memory technologies. Conserving battery power is even more important because ad-hoc networks are generally involved in critical applications like resource planning and communication in battlefield and disaster-relief conditions. To give an estimate of power requirements, in the Digital Equipment Corporation's Roamabout radio [63], transmission requires 5.76 watts while reception requires 2.88 watts. So, in

order to reduce the power consumption in routing in ad-hoc networks, we try to keep the number of messages transmitted and received as small as possible.

5.1.2 Problem Description

We consider the routing of a datagram in a highly dynamic mobile ad hoc network. We assume that the transmission time of a datagram between two nodes in direct communication is negligible.

The organization of the chapter is as follows: In section 5.2 details the dynamic forest construction algorithm that is fundamental to the implementation of our proposed routing algorithm. In this section, we explain how the algorithm is distributed in nature. We, then, analyze the algorithm for finding its space and time complexities. We also look at various schemes to optimize the performance of the algorithm. In section 5.2.4, we discuss our routing algorithm including its space and time complexities and its comparison with flooding. In order to evaluate the performance of our routing algorithm. We explain our simulation model in section 5.3.1 and examine and analyze the results of our simulation. Finally, we describe our conclusions and future work in section 5.4.

5.2 Dynamic Forest Construction Algorithm

The basic idea of the algorithm is to construct a set of dynamic trees (also known as a dynamic forest) $T_1, T_2, ..., T_k$ where each T_r, at a given time, has nodes (mobile hosts) $h_1, h_2, ..., h_p$ such that any node h_i can communicate to a host h_j. We assume that each node h knows the IDs of the neighboring nodes within its direct transmission

range and it is denoted by $N(h)$. In this manner, a node can communicate to any one of its neighbors directly. On the other hand, if a node wants to send a message to another node which is beyond its direct range then it would try to use other nodes as routers. *The key ingredient of the algorithm is to dynamically readjust the direct connections between the nodes as the hosts move around.* In simple graph-theoretic terminology, this means that a host h_i can try to communicate to a host h_j by sending its message via the tree edges and in some special situations as described later, using non-tree edges.

At any given time, each node is either acting as a router or readjusting information about its direct neighbors and the root ID of the tree it belongs to, as the ambient environment changes. We will first present the forest construction algorithm and then the routing algorithm for clarity, keeping in mind that both of these are concurrently executed at each node.

Each node h_i stores four types of information: its ID (denoted by h_i), its parent's ID (denoted by $P(h_i)$, the ID of the root of the tree it belongs to (denoted by RootID(h_i)) and ID's of its children (denoted by $h_i.child(j)$ for the jth child of h_i). We say two trees T_i and T_j come *in contact* with each other if there exists a node $g \in T_i$ and $h \in T_j$ such that g and h are within direct transmission range of each other. We call g and h the contact nodes of their respective trees. When two trees T_i with root node r_i and T_j with root node r_j come in contact with each other an attempt to merge the trees into a larger tree is made at the contact nodes. The merger is accomplished by creating an edge between the contact nodes. The root node resolves the conflict between possibly several pairs of contact nodes requesting merger. After the merger, one of the root nodes becomes the root of the newly

merged tree. We will make these details precise and elaborate later.

5.2.1 Distributed Algorithm

The algorithm for tree construction is distributed, asynchronous and is characterized by the absence of a global clock. The time taken by a message to travel from source to destination is finite. We further assume that the message sent along a link arrives in the order it was sent and that the nodes are identified (ID's) by positive integers. In general, a node can be either the root of a tree or an internal node of a tree. A node can be in one of three states namely, *router*, *merge*, and *configure*. A root or an internal node in the router state follows the routing algorithm to be presented later. That leaves us with four possible types of status for a node, viz., (root, merge), (root, configure), (internal node, merge) and (internal node, configure). A node in the *configure* state performs updates to its data structures when its parent moves away beyond its communicating range (*Lose Parent*) or when one or more of its children move away beyond its communicating range (*Losing Child*) or when a node comes in contact with it (*Merge Request*). In addition, we define a *bridge* as a connection between two nodes that are in radio contact but belong to two different spanning trees. Bridges are formed to achieve connectivity between two spanning trees when either of the two nodes forming the bridge is likely to move away shortly. By just forming a bridge, the heavy cost involved in re-aligning the tree is avoided. A node can either add a bridge (*Add Bridge*) or remove an existing bridge (*Remove Bridge*) based on the existence or loss of radio contact respectively.

Figure 5.2 depicts the different events that can occur at each node in the network. A node v in the network can either (a) lose its parent or (b) lose its child or (c) can

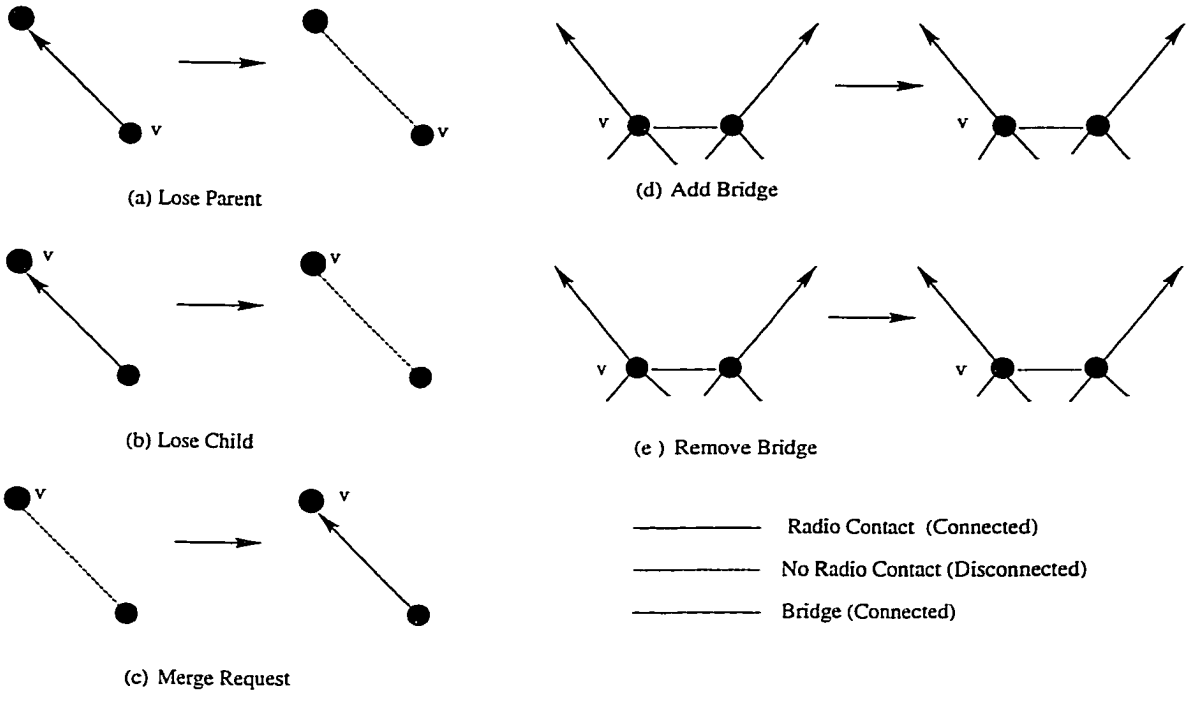


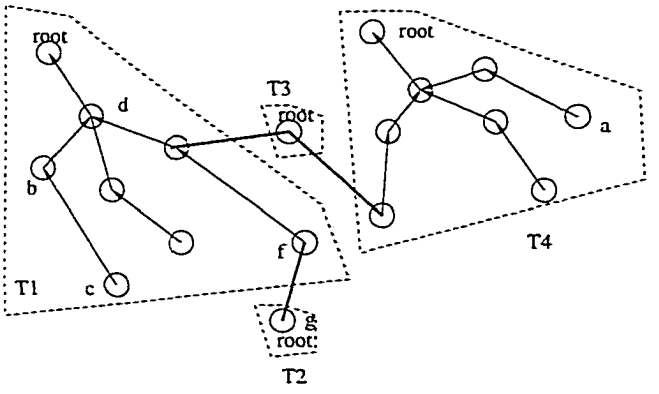
Figure 5.2: Various possible events that can occur at each node v

send a merge request for merging to another node that it comes into radio contact with. The node can also (d) add a bridge or (e) remove an existing bridge.

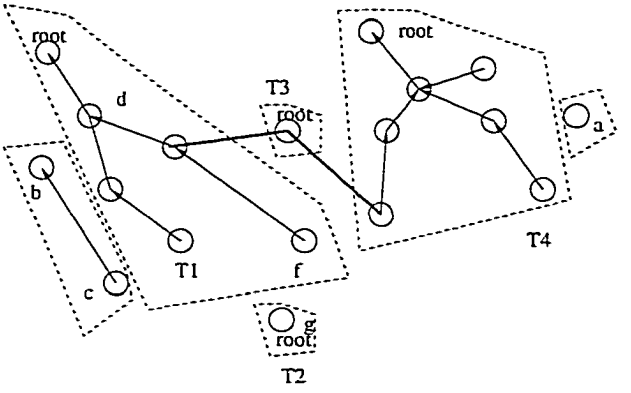
Figure 5.3 shows an example of occurrence of these events in a network. The figure shows the network topology at times $t = \mathrm{T}$ and $t = \mathrm{T} + \delta$. The network topology changes as the nodes move around. For example, node d loses its child b (b loses its parent), the bridge between nodes f and g is removed and node a loses its parent.

A node in the *merge* state performs many actions such as requesting to merge, waiting to merge, performing merger, and aborting merger.

At time $t = 0$, each node h for which $N(h) = \phi$ is in the *configure* state and all of the others are in *(root,merge)* status. We will now describe the actions performed



Network Topology at T = t



Network Topology at T = t+ δ

Figure 5.3: An example scenario showing how the events occur because of the mobility of the hosts.

at each node in great detail.

Merge Request: If a node x comes in contact with a node y where $RootID(x) < RootID(y)$, then x makes a request to node $RootID(x)$ for merger. This node x waits for a estimated time to get permission to merge from the node $RootID(x)$ after which it makes the request again.

Wait to Merge: The node x waits for a *grant* or *deny* message to either proceed with the merge or abort.

Merge Permission: A root node r receiving requests for merger (note that the requests come only from other trees with $RootID$ greater than itself) selects the tree in the first-in-first-out order and sends a message granting permission to merge. The *grant* permission message is sent to the node where the request came from. After having sent the *grant* message, the root node discards all requests for merge from other nodes until it receives an abort message. If the root node x loses a child node v, such that a node in T_v has been granted permission to merge, then it generates a self-abort merge message. Let node I be in the path from root r to node x, where x is the node that has come in contact with node y of a different tree and has made the request to r to merge. Also let node I receive the Merge Permission message from its parent to be sent to one of subtrees of the child y containing the merge requesting node. If the node I loses the link to y, the node I sends an abort message to the root r.

Abort Merge: When the root node receives an abort message, it is ready to grant permission to any other node for merger.

Perform Merge: When node x receives permission to merge, it checks to see if its link to node y still exists. If no such link exists, it sends an *abort merge* message to RootID(x). If the link exists, then it updates its RootID information as $RootID(x) \leftarrow RootID(y)$. Furthermore, x sends a message to other nodes v in its tree to set their $RootID(v) \leftarrow RootID(y)$. $p(x)$ becomes a child of x and $p(x) \leftarrow y$. If x is a root node then it becomes an internal node after the merge.

Lose Parent: If a node x loses its parent $p(x)$, then it takes the responsibility of the root and performs the following:

1. If permission to merge message has traveled through node x to one of the

nodes in its subtree, then it waits for either an abort message or perform merge message. In case of an abort message, it sends a message to all the nodes in its subtree to update their $RootID$ information.

2. Otherwise, it sends a message to all the nodes in its subtree to update their $RootID$ information.

Lose Child: When a node x loses a link to a child node v and if some node in the subtree rooted at T_v has been granted permission to merge, then node x informs the node $RootID(x)$ to abort the merge.

Add Bridge: When a node x comes in contact with another node v that belongs a different spanning tree, it can form a bridge connecting to node v.

Remove Bridge: When a node x loses radio contact with a node v such that there is a bridge connecting nodes x and v, it can remove the bridge.

We show that the distributed algorithm is deadlock free and creates a spanning tree whenever possible. We also show that a given tree at any time contains $O(n)$ nodes with a height of $O(D)$, where D can be at most $n-1$. Hence, the message and time complexity for completing actions corresponding to each event is $O(n)$ and $O(D)$, respectively.

5.2.2 Analysis of the Distributed Algorithm

We have to show that the distributed algorithm is deadlock free and creates a spanning tree whenever possible. In this subsection, we will also derive the message and time complexity of the distributed algorithm.

Theorem: *Prove that the proposed spanning tree based routing algorithm does not produce cycles and is deadlock-free.*

Proof: In order establish correctness of the algorithm, we define the concept of a *immobile node* and a *immobile tree.* A node is *immobile* if its region of influence does not change. A tree is *immobile* if it contains nodes that are immobile. Given two immobile trees, the above algorithm merges the tree with a smaller RootID with the tree that has a larger RootID.

From the algorithm, we observe that:

(i) When a node u receives a *Merge Request* from another node v, node u checks the root identity of the node v. If the root identity of node u and node v are the same (meaning that they belong the same spanning tree), the *Merge Request* is denied.

(ii) When several *Merge Requests* come to nodes in a spanning tree simultaneously (that could potentially create cycles), the root of the spanning tree grants permission to only one of the *Merge Requests* and discard the others.

Thus, the root node of a tree grants permission to at most one node to merge with another tree and thus avoids the creation of a cycle.There are scenarios in which the RootIDs of the trees change after permission to merge has been granted. In such scenarios also a proper tree is created since a tree is granted permission to merge with at most one other tree at a given time.

After granting permission to merge, the root node simply discards all other merge requests. It is entirely possible that the node x to whom permission was granted to have moved. In this case, the algorithm would result in a deadlock without

appropriate abort message reaching the root. We take care of this scenario as part of the actions for *Merge Permission* event. Also note that the actions performed as part of the *Loose Child* event, takes care of issuing the abort message to the root that has granted permission to merge. ■

Message and time complexity of the proposed spanning tree based algorithm

The message complexity of a distributed algorithm is determined by the total number of messages sent during the entire execution of the algorithm. The time complexity establishes the time required to complete the algorithm's execution. Given the ad-hoc nature of the network, its execution never terminates. To this effect, we are only interested in determining the message and time complexity for performing actions corresponding to each event separately. The actions require the traversal of the tree in which the event occurs. A given tree at any time contains $O(n)$ nodes with a height of $O(D)$, where D can be at most $n-1$. Hence, the message and time complexity for completing actions corresponding to each event is $O(n)$ and $O(D)$, respectively.

5.2.3 Optimizations to the Algorithm

There are several ways in which the above algorithm can be optimized.

a. *Optimizing the Shape of the Tree*:

It is clear from the earlier discussion that a tree whose depth is smaller improves the time complexity and in some cases the message complexity of the

operations that correspond to events. If the depth of the trees have to be kept smaller then the tree has to be bushier, that is each parent will have many children. In a forest of bushy trees, if a parent node moves away, all the child nodes have to take the responsibility of a root, inform this information to all the nodes in the subtree, there by increasing the message complexity. Hence there are message and time complexities tradeoffs.

The root of the tree can control the choice of the tree to join with by selectively processing the requests for merge that arrive at the root node. For example, if a node $x \in T_i$ is a contact node and requests the root of T_i to merge with T_j at contact node $y \in T_j$, then the following two aspects should be examined before permission is granted.

(a) The size of T_i should be smaller compared to the size of T_j, otherwise we have to send too many messages to set appropriate $RootID$ information. If for example, T_i has 1000 nodes and T_j has 10 nodes, joining with T_j may not be very beneficial and certainly costly in terms of number of messages as all the 1000 nodes in T_i has to change their $RootID$ information.

(b) There may be several pairs of contact nodes that exists in T_i and T_j. The choice of the pair of contact nodes through which the merge should proceed impacts the shape of the tree constructed. For example, if a contact node $y \in T_j$ is closer to its root, then the nodes in T_i can communicate with its new root rather quickly. If both the contact pairs $x \in T_i$ and $y \in T_j$ are closer to their roots, then the depth of the resulting tree can be kept smaller.

(c) If bushier trees have to be avoided, the we have to select a contact node $y \in T_j$ with smaller number of child nodes.

b. *Optimizing Grant Permission Actions*:

If a node x has sent a request for merge message to its parent that originated in its subtree and if x loses its parent, then x assumes the responsibility of the root. This new root x can immediately grant permission to merge. By doing so it can perform actions specified in (1.) of the *Lose Parent* event and avoids the update operation to update the $RootID$ information.

5.2.4 Routing

Each node v receiving the message (initially, $v = source$) performs the following algorithm. Routing packets on the spanning tree can either be (a) **Hybrid Tree-Flooding (HTF)** : packets sent to all possible neighbors in the tree and adjoining bridges in the spanning tree or (b) **Distributed Spanning Tree Shuttling (DST):** when a packet down to reaches a leaf node in a tree then it can be sent up the tree only till reaches a certain height, called *shuttling level* [1] after which it can be sent down the tree or adjoining bridges. We chose shuttling as the mechanism to perform routing of packets as shuttling uses a lesser number of messages as compared to tree-flooding. This is in tune with our design requirements to keep the routing power-aware. Datagrams can be stored at nodes for a duration called the *hold time* during which it tries to route the packet directly to the destination if possible. If it

[1]Shuttling Level is represented as fraction of the total tree height. For example, a shuttling level of 0.5 means the shuttling is done till the packet reaches a node that is at half the tree height. We use rounding of fractions to calculate the actual shuttling height from the shuttling level.

is not able to do so, it is routed using shuttling. *The rationale for holding time is that for systems that are becoming more stable and connected, it might be prudent to buffer packets and route them as the networks connectivity increases over time.* Figure 5.4 shows an example of shuttling on a forest of spanning trees. In the figure, we assume that the shuttling height is 3. The nodes that are shuttling nodes are marked. When a packet comes to a shuttling node from a child, it is sent to its bridge neighbors and all of its children. These packets maintain a hop counter in order to perform shuttling.

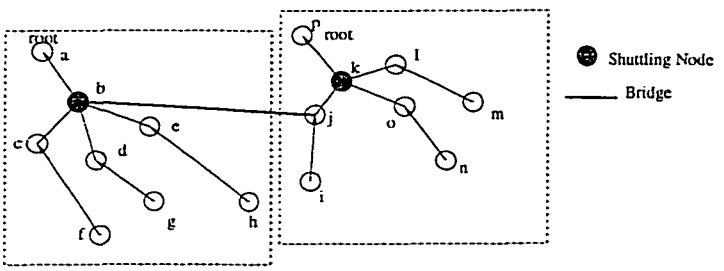


Figure 5.4: An example of Shuttling

Algorithm Routing in DST - Distributed Spanning Tree;

1. **If** *destination* is within the region of influence **Then** send *packet* directly to *destination*

2. **Else** send *packet* along the tree neighbors of v and along the bridges of v using **shuttling.**

5.2.5 Comparison With Flooding

We will calculate the average number of messages (denoted A) needed to route a packet from source to destination using the algorithm DST with the assumptions that the forest of spanning trees is immobile and a message passes through an edge at most once. Let $T(i)$ be the maximum number of messages that are sent along the edges of a tree with i nodes, which is $i-1$. Let $M(i)$ be the maximum number of messages that are sent along the edges of the of a with i nodes, which is $\frac{i(i-1)}{2}$. The DST algorithm sends the message along the tree edges and floods the message across the bridges. The average number of messages is given by:

$$A = \frac{\sum_{i=1}^{n}(T(i)+M(i))}{2} \tag{5.1}$$

$$= \frac{\sum_{i=1}^{n}(i-1)}{n} + \frac{\sum_{i=1}^{n}\frac{(n-i)(n-i+1)}{2}}{n} \tag{5.2}$$

$$= \frac{n^2}{2} + \frac{n}{2} \tag{5.3}$$

Comparing A with the pure flooding (FL) algorithm which requires $\frac{n(n-1)}{2}$ messages, it can be seen that the DST algorithm requires 1/3 the number of messages required by the tree-flooding algorithm.

5.2.6 Analysis of Routing Algorithm

To discuss the performance of routing we need to identify a reasonable criterion. Since the topology is dynamic, it is too weak to expect that a datagram be delivered from s to d only if they are connected at the time of message origination at s. On the other hand, it is unreasonable to expect messages to wait indefinitely long in the network; if d is not reachable from s at all, flooding the messages could lead to inordinate amounts of datagrams being generated, thereby causing denial of service between the nodes that are connected. To address the first issue, we introduce the *concept of connectivity through time*, and for the second issue we employ a time-out mechanism.

Let $CG(t) = (V, E(t))$ denote the *connectivity graph* at time t such that V is the set of nodes and $(u, v) \in E(t)$ if and only if u and v are in direct communication at time t. Let the topology changes occur at unique times, denoted in increasing order by $t_1, t_2, \ldots, t_p$ for $t_i \in [0, T]$. Note that $CG(t)$ remains constant for all $t \in [t_i, t_{i+1}]$. Consider $[T_L, T_H] \subset (t_{i-1}, t_{i+1})$ containing t_i. We define that s and d are $[T_L, T_H]$-*connected* if there exists a node v such that: (i) s and v are connected in $CG(t_i)$, and (ii) v and d are connected in $CG(t_{i+1})$. This definition is naturally extended to for any interval containing more than one t_i's, where we assume that every node v is connected to itself. Intuitively speaking, only if s and d are $[0, T]$-connected, a datagram from s can be delivered to d by possibly buffering it at some intermediate

node. Let $\Delta_T \in [0, T]$ denote the *hold time* which is the *total time* that a datagram is held at nodes. If the hold time of a datagram is greater than $[0, T]$, then s and d are *not* $[0, T]$-connected. We now modify the algorithm DST so that each datagram maintains a variable containing its total holding time so far. When a router detects datagrams whose holding time exceeds T, the datagram is deleted from the router's queue.

Comparing A with the pure flooding algorithm which requires $\frac{n(n-1)}{2}$ messages, it can be seen that the DST algorithm requires 1/3 the number of messages required by the flooding algorithm.

We define a metric called *reachability* (R). The reachability of an ad-hoc routing algorithm is the ability of the routing algorithm to send the packet from the source to the destination. We can say that in a forest of slow merging trees reachability might be improved with DST algorithm. Also in a forest which frequently establishes many trees (when nodes loses their parent quite often) the flooding algorithm is better in terms of reachability.

5.3 Results and Analysis

We simulated our algorithm for studying its performance. Mobile hosts move on a rectangular grid with a constant radio range. Each move of every mobile host is characterized by its speed, distance, direction. We chose several pairs of nodes (N_p) that communicate. Messages arrive at the source at the rate λ_m according to an exponential distribution. The rate of arrival of events causing increased connections (λ_c) (*Merge Request, Add Bridge*) and disconnections (λ_d) (*Lose Parent, Lose Child,*

Remove Bridge) has an exponential distribution. We studied the effect of λ_c, λ_d and λ_m on reachability (the percentage of packets delivered successfully to the destination) and total number of messages for various values of holding times and number of pairs communicating for the DST and Tree Flooding algorithms.

We simulated the performance of our algorithm to steady the following performance metrics.

- *Total Number of Messages (M)*: The total of number of datagram messages is the total number of messages sent between any node to any another node in the entire network for the entire simulation duration.

- *Reachability (R)*: Reachability is the fraction of the messages sent by the source that reach the destination during the course of the entire simulation.

- *Message-Reachability ratio (MR)*: The ratio of the total number of messages (M) to the corresponding reachability (R) is the message reachability ratio for instance of the algorithm.

In order to study the performance and the applicability of the FL, DST and HTF routing algorithms, we define a measure called the *holding time*. Holding time of a message at a node is the duration of the time that the message is held in the node's message queue before being transmitted successfully to a neighboring node. We also, study the effect of holding time on reachability. We also analyze the effect of shuttling height on the performance of the routing algorithms.

5.3.1 Simulation Details

We simulated the three routing algorithms to study their relative performance. In our simulation, we assume that 100 mobile hosts move on a rectangular grid with a constant radio range. We assume that initially the mobile hosts are connected to form a random forest of trees. Each move of every mobile host is characterized by its speed, distance, direction. We chose a certain randomly number of pairs of nodes (N_p) that communicate. Messages arrive at the source at the rate of λ_m according to an exponential distribution. Given that a message originated at v, it is destined to a node u according to a Markov Process. The rate of arrival of events causing increased connectivity (λ_c) (*Merge Request, Add Bridge*) and disconnection (λ_d) (*Lose Parent, Lose Child, Remove Bridge*) also has an exponential distribution. We studied the effect of varying λ_c, λ_d and λ_m on reachability and total number of messages for various values of holding times and number of pairs communicating for each of the routing algorithms.

The confidence coefficient for our simulations was 0.95. We used a sample size of 35 for our experimental runs.

We now present the results of the simulation of the HTF and DST routing algorithms with a view to assessing and evaluating their relative performance. We try to answer the following questions by looking at the performance of the routing algorithms:

1. What is the effect of the increasing the rate of connections (resp. rate of disconnections) on the reachability and number of messages for each the routing algorithms for static, high-connection, high-disconnection and highly dynamic

systems?

2. What is the effect of holding time on reachability and number of messages for each type of rerouting algorithms?

3. How does changing the shuttling height effect the performance of DST algorithms?

4. What is the tradeoff between reachability and the number of messages? In other words, to achieve a certain level of reachability what is the number of messages required by each of the routing algorithms?

5. What are the conditions when one of the algorithm performs better (in terms of the number of messages required to achieve the same level of reachability) than the other?

In order to answer the questions posed above, we consider four cases separately: (i) Low Rate of Disconnections ($\lambda_d = 10$), (ii) High Rate of Disconnections ($\lambda_d = 90$), (iii) Low Rate of Connections ($\lambda_c = 10$), (iv) High Rate of Connections ($\lambda_c = 90$).

For cases (i) and (ii) (With fixed values of λ_d), we plot the following graphs to see:

1. Graph 1: The effect of the rate of connections (λ_c) on reachability in a busy network ($\lambda_m = 90$ and $N_p = 8$) with varying holding times.

2. Graph 2: The effect of the rate of connections (λ_c) on reachability in the DST algorithm for varying shuttling heights.

3. Graph 3: The effect of the rate of connections (λ_c) on the number of messages in a busy network with varying holding times.

4. Graph 4: The effect of the rate of connections (λ_c) on the number of messages in the DST algorithm for varying shuttling heights.

5. Graph 5: The effect of the rate of connections (λ_c) on the Message-Reachability ratio for each of the routing algorithms.

For cases (iii) and (iv) (With fixed values of λ_c), we plot the following graphs to see:

1. Graph 1: The effect of the rate of disconnections (λ_c) on reachability in a busy network with varying holding times.

2. Graph 2: The effect of the rate of disconnections (λ_c) on reachability in the DST algorithm for varying shuttling heights.

3. Graph 3: The effect of the rate of disconnections (λ_c) on the number of messages in a busy network with varying holding times.

4. Graph 4: The effect of the rate of disconnections (λ_c) on the number of messages in the DST algorithm for varying shuttling heights.

5. Graph 5: The effect of the rate of disconnections (λ_c) on the Message-Reachability ratio for for each of the routing algorithms.

For all the Graphs 1–5 (for all the cases i– iv), the number of communicating pairs (chosen randomly) is 8 and the rate of messages λ_m originating at the sources (in the communicating pairs) is equal to 90. These values are consistent with the values used

by other researchers [68]. We varied λ_c (the rate of connectivity) up to a maximum value of 90. At $\lambda_c = 90$, the network was becoming a single component (there was a path from a node to all other nodes). Increasing λ_c beyond 90 did not increase the connectivity any further. Similarly, we varied λ_d (the rate of disconnectivity) up to a maximum value of 90. At $\lambda_d = 90$, the network was becoming a forest of single-node components. Increasing λ_d beyond 90 did not increase the disconnectivity any further. For the cases (i) and (ii) (resp. (iii) and (iv)), in all the Graphs 1–5, the rate of connections λ_c (resp. rate of disconnections λ_d) is varied from 10 to 90 in steps of 20. For Graphs 1 and 3, the rate of messages originating at the source λ_m is equal to 90 for the FL, HTF and DST (with shuttling level = 1.0) routing algorithms. We plot the performance of each of the routing algorithms for different holding times (HT = 0.01, 0.02 and 0.10). For the FL routing algorithm, the holding time is fixed and is equal to 0.01. For Graphs 2 and 4, for the HTF and DST routing algorithms and the holding time is equal to 0.01. The DST routing algorithm is run for shuttling level equal for 0.25, 0.5 and 1.0. For Graph 5, we plot the Message-Reachability ratio for the FL, HTF and DST (with shuttling level = 1.0) routing algorithms.

5.3.2 Simulation Results and Analysis

We no present the simulation results as graphs for all the four cases discussed above. For each of these cases, we plot all the five graphs (Graphs 1 – 5), and analyze these graphs trying to answer the performance related questions that we posed above in Section 5.3.1. The graphs 1–5 are presented as a set of five sub-figures, (a), (b), (c), (d) and (e), respectively, for each of the cases. We will discuss each of the sub-figures separately.

We assume that flooding does not require any messages for maintaining routes in the route tables of the nodes. The DST and HTF require messages of maintaining the routing information (root identity, parent identity, list of identities of children and neighbors). We calculate the number of messages required for route maintenance (building and maintaining the forest of dynamic segment trees) and routing messages (for sending the data packets from source to destination). For graphs 3–5, the number of the messages refer to the total number of messages (sum of messages for routing and the messages for maintenance and update of the forest of spanning trees).

Table 5.1 compares the number of messages required for routing (M_r) and route maintenance (M_m) - the messages for maintaining the forest of spanning trees in case of **DST** and **HTF**.We see from the table from the total number of messages required for **FL** is much larger than total number of messages required for **DST** and **HTF**) for all values of λ_c and λ_d. Also, the M_m forms a small fraction of the total messages for the routing algorithms.

Case (i) and Case (ii) : Fixed Rate of Disconnections ($\lambda_d = 10$ and 90 respectively)

Figures 5.5 and 5.6 show the five graphs discussed above as sub-figures for Case (i) and Case(ii) respectively. We consider now each of the figures and look at the performance of these cases.

Reachability v/s λ_c (Sub-figure (a))

For both the cases, we observe from their corresponding sub-figure (a) that reach-

λ_d	λ_c	**FL**	**HTF**			**DST**		
		M_t	M_r	M_m	M_t	M_r	M_m	M_t
10	10	2410007	1435	320013	321448	1435	14120	15555
10	50	4039892	2808	630189	632997	2808	22047	24855
10	90	7639899	8103	1201379	1202997	8103	29417	37520
50	10	398829	3013	57257	60270	3013	4329	7342
50	50	906160	8704	132345	141049	8704	8967	17671
50	90	2901045	13682	205207	218889	13682	13405	27087
90	10	365231	11124	89099	100223	5124	2294	7418
90	50	652090	14123	145980	160103	14123	6789	20912
90	90	1142586	22951	366295	389246	22951	8314	31265

Table 5.1: Comparison of the Number of Messages for Flooding (**FL**), Hybrid-Tree Flooding (**HTF**) and Shuttling (**DST**) with shuttling height = full height. M_r, M_m and M_t refer to the total number of routing messages, total number of maintenance messages and the total number of messages for routing and route maintenance. λ_c and λ_d are the rate of connectivity and the rate of disconnectivity respectively.

ability is better for FL and HTF than DST for all values of λ_c keeping every other parameter including the holding time the same. However, as λ_c increases, the difference between the reachability of FL, HTF and DST narrows. We also observe that reducing holding time (for both HTF and DST) increases reachability. This is because, when we reduce the holding time, the packets are routed from a node more frequently resulting in better chances of the packet reaching its destination. Reachability for all the routing schemes improves as λ_c increases. This is to be expected as increasing connectivity translates to better reachability.

The reachability in case of Case (i) is higher for Case (ii) as expected. The reachability for Case (i) is in the range of approximate range of 0.18 - 0.90 while the reachability for Case (ii) is in the range of 0.1 - 0.51.

We observe that for highly dynamic situations (high λ_c and high λ_d) and high disconnection systems (low λ_c and high λ_d) reachability values of FL, HTF are better than comparable instances of DST while for stable systems (low λ_c and low λ_d) and high connection systems (high λ_c and low λ_d), the reachability values for FL, HTF and DST are comparable.

Effect of Shuttling Height on Reachability in DST (Sub-figure (b))

Increasing the shuttling level increases the reachability (See sub-figure (b)). This is so, as the shuttling level increases, the messages have a better chance of being transmitted across the bridges and neighbors of the nodes at the shuttling height. For example, consider if shuttling level is zero, the messages that reach the leaf nodes it can be transmitted on any bridges connected to the leaf node. However, if the shuttling level is larger than zero, the messages that reach the nodes at the shuttling level are transmitted not only on the appropriate bridges but also to the

nodes' children. However, the increase is not very significant. This is because, the connectivity of the nodes at the shuttling level is random. The reachability for both the schemes (including the three DST instances with varying height) improves with increasing values of λ_c. The reachability for FL, HTF is again better than the best DST case. Reachability for FL is the best among all the three cases.

The Number of Messages v/s λ_c (Sub-figure (c))

The number of messages in the system as a result of routing for all the routing algorithms increase with increasing connectivity. This is expected as the connectivity increases, nodes can potentially broadcast the message to more nodes thus increasing the number of messages (See sub-figure (c)).

In both the cases, there is a large variation in the number of messages for comparable cases of FL, HTF and DST. The ratio of the number of messages for comparable cases of FL to HTF to DST is $160 - 200 : 20 - 25 : 1$.

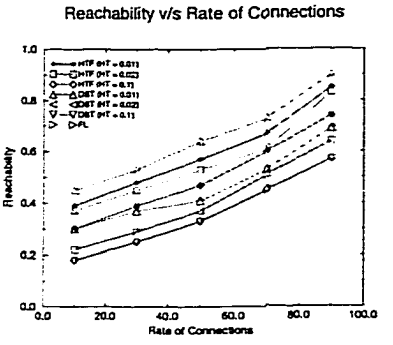
Effect of Shuttling Height on the Number of Messages (Sub-figure (d))

Increasing the shuttling level increases the number of messages in case of DST. This is obvious because as the shuttling level increases, the path (and hence the number of messages) for shuttling increases. Increasing λ_c has a more profound effect on the number of messages in FL, HTF than DST because in case of HTF, the nodes fully use the increasing connectivity as the transmission is done to all possible nodes that are in radio connectivity wit the node. However, in case of DST since the transmission and broadcasting is done only by the nodes at the shuttling level, the effect is not so pronounced.

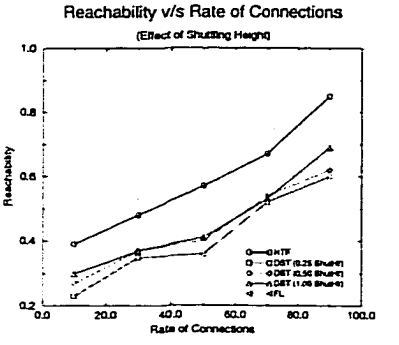
Message-Reachability Ratio (Sub-figure (e))

In this case, we compare the MR of all the three routing algorithms. The message-

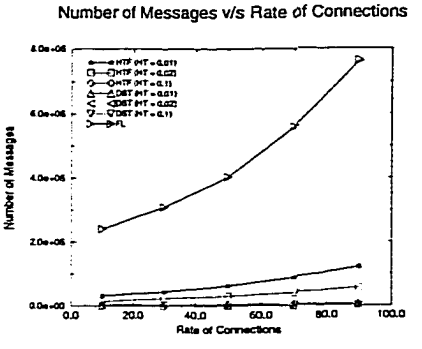
reachability ratio for comparable instances of FL, HTF and DST is 800: 100: 1 which shows that DST can greatly improve the performance of routing. In both cases, with increasing λ_c, the MR reduces for all routing algorithms. The MR for DST is consistently low (and significantly lower than that of FL and HTF) for varying values of λ_c indicating its consistent good performance.



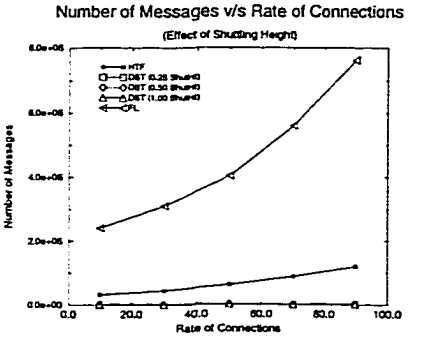
(a)



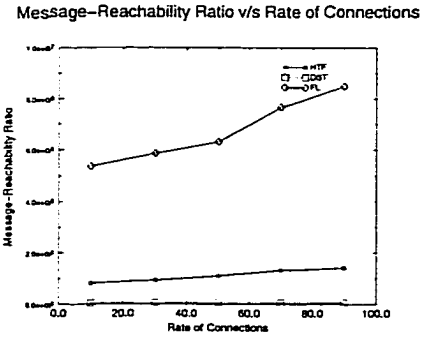
(b)



(c)

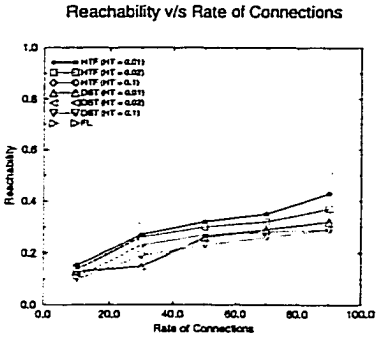


(d)

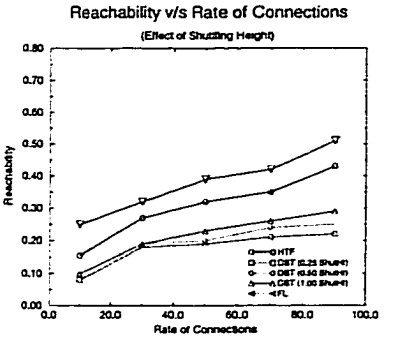


(e)

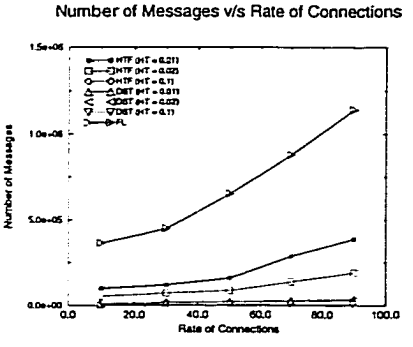
Figure 5.5: Performance Analysis for Case (i): Low Rate of Disconnections (λ_d = 10). Sub-figures (a), (b), (c), (d) and (e) are the Graphs 1 – 5 respectively which are described in Section 5.3.1.



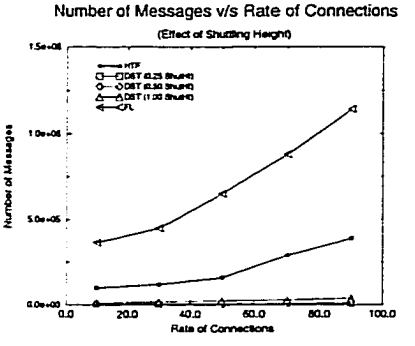
(a)



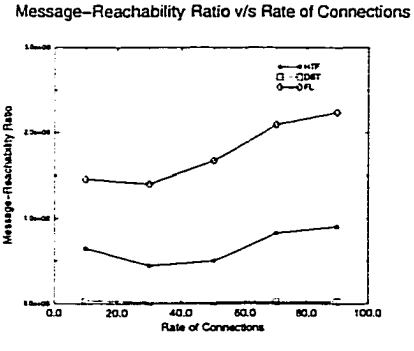
(b)



(c)



(d)



(e)

Figure 5.6: Performance Analysis for Case (ii): High Rate of Disconnections (λ_d = 90). Sub-figures (a), (b), (c), (d) and (e) are the Graphs 1 – 5 respectively which are described in Section 5.3.1.

Case (iii) and Case (iv) : Fixed Rate of Connections (λ_c = 10 and 90 respectively)

Figures 5.7 and 5.8 show the performance of Cases (iii) and (iv). These cases look at the performance of HTF and DST for varying values of λ_d and fixed values of λ_c. We now analyze the performance of these two cases below.

Reachability v/s λ_d (Sub-figure (a))

For Cases (iii) and (iv), reachability decreases with increasing λ_d. Increasing λ_d reduces the connectivity of the network and hence the reachability reduces. However, reachability increases by reducing the holding time for both HTF and DST. This is the same phenomenon that we observed for Cases (i) and (ii). We observe that for highly dynamic situations (high λ_c and high λ_d) and high disconnection systems (low λ_c and high λ_d) reachability values of FL and HTF are better than comparable instances of DST while for stable systems (low λ_c and low λ_d) and high connection systems (high λ_c and low λ_d), the reachability values for FL, HTF and DST are comparable.

Effect of Shuttling Height on Reachability in DST (Sub-figure (b))

For Cases (iii) and (iv), increasing shuttling height increases reachability as we observed for Cases (i) and Cases (ii). We also observe that the increase in reachability is not significant.

The Number of Messages v/s λ_d (Sub-figure (c))

The number of messages in the system as a result of routing for all the routing algorithm decrease with increasing rate of disconnectivity (λ_d). This is expected as the disconnectivity increases, nodes can potentially broadcast the message to fewer

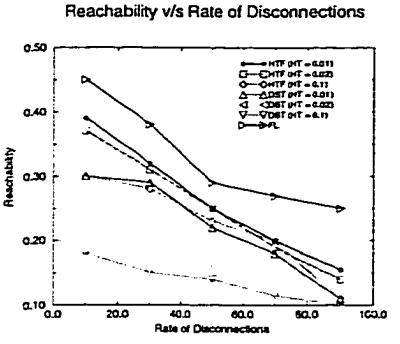
nodes thus decreasing the number of messages (See sub-figure (c)). In both the cases, there is a large variation in the number of messages for comparable cases of FL, HTF and DST. The ratio of the number of messages for comparable cases of FL, HTF and DST is 300 : 10 : 1.

Effect of Shuttling Height on the Number of Messages (Sub-figure (d))

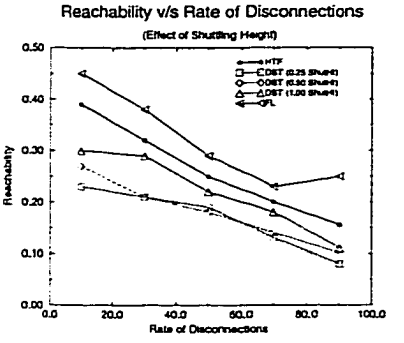
For Cases (iii) and (iv), increasing the shuttling level increases the number of messages in case of DST as was also observed in Cases (i) and (ii). Increasing λ_d has a more profound effect on the number of messages in FL and HTF than DST because in case of FL and HTF, the transmission of messages by nodes is significantly reduced because of increasing disconnectivity since FL and HTF rely on connectivity and flooding. However, in case of DST since the transmission and broadcasting is done only by the nodes at the shuttling level, the effect is not so pronounced.

Message-Reachability Ratio in HTF and DST (Sub-figure (e))

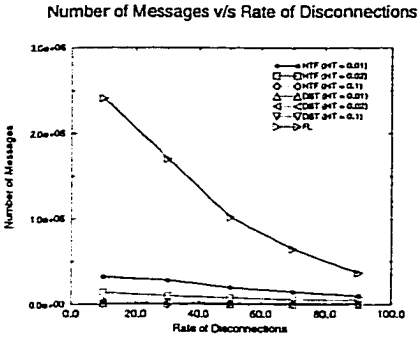
For both Cases (iii) and (iv), the message-reachability ratio worsens for increasing values of λ_d. The message-reachability ratio for FL, HTF and DST for comparable cases of FL, HTF and DST is 75 : 10 : 1. This re-iterates the superior performance of DST as compared to FL and HTF.



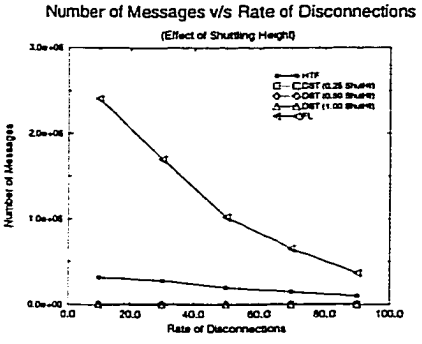
(a)



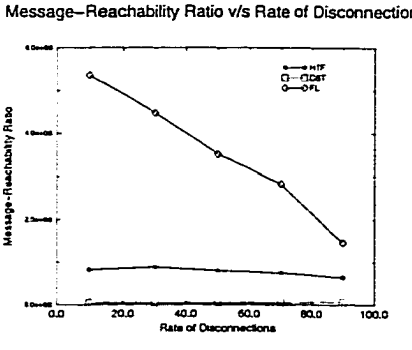
(b)



(c)

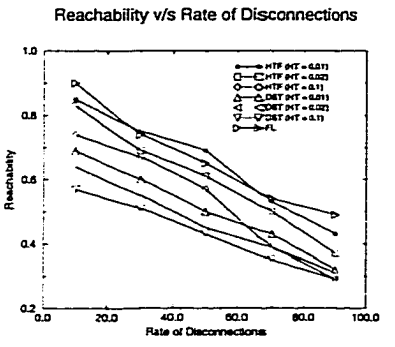


(d)

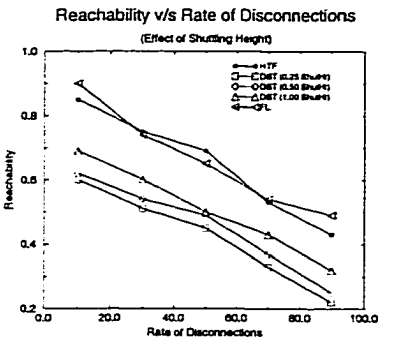


(e)

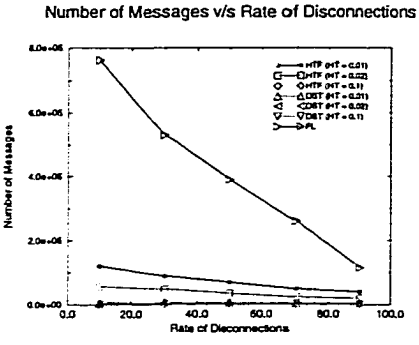
Figure 5.7: Performance Analysis for Case (iii): Low Rate of Connections ($\lambda_c = 10$). Sub-figures (a), (b), (c), (d) and (e) are the Graphs $1 - 5$ respectively which are described in Section 5.3.1.



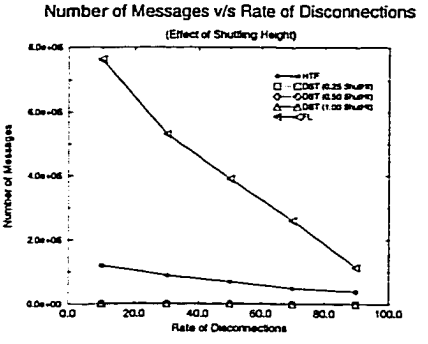
(a)



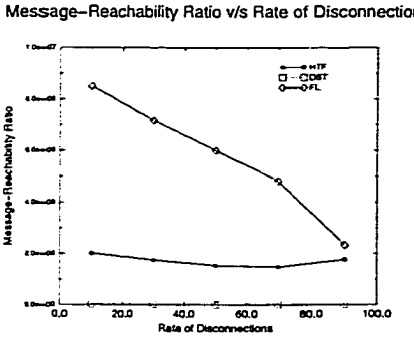
(b)



(c)



(d)



(e)

Figure 5.8: Performance Analysis for Case (iv): High Rate of Connections ($\lambda_c = 90$). Sub-figures (a), (b), (c), (d) and (e) are the Graphs $1 - 5$ respectively which are described in Section 5.3.1.

From the above discussions, we observe that:

- In *stable systems* (low λ_c and low λ_d), and in *high connectivity systems* (high λ_c and low λ_d), reachability using DST, HTF and FL are comparable for varying values of holding times and number of communicating pairs.
- In *high disconnectivity systems* (low λ_c and high λ_d), reachability using DST is lower than that of FL and HTF for varying values of holding times and number of communicating pairs.
- In *highly dynamic* systems (high λ_c and high λ_d), FL and HTF are better options to attain high reachability. However, the number of messages using FL and HTF are significantly more as compared to DST.
- In all comparable instances, DST uses far lesser messages than FL and HTF.
- FL uses about 10 to 20 times the number of messages used to HTF and about 200 to 300 times the number of messages used by DST.
- Reducing holding time in stable and high connectivity systems tends to improve the performance (reachability and the message-reachability ratio). However, for high disconnectivity and highly dynamic systems, increasing holding time does not significantly improve reachability and the message-reachability ratio.

5.4 Summary

In this chapter, we have presented a novel, efficient reactive routing algorithm for ad-hoc networks which uses a forest of spanning trees formed by the nodes in the

network. The routing algorithm uses the forest of spanning trees to perform routing using a process called shuttling. The algorithm is simple and has low space and time complexities. We also present several optimization techniques to improve the performance of the algorithm. We introduced the concept of connectivity through time to rate the performance of our routing algorithm. We introduced the concept of holding time with a view to improve reachability of packets. We studied the performance of the algorithm for systems with varying dynamics of connectivity and disconnectivity. We find through extensive simulation that increasing holding time in stable and high connectivity systems can improve reachability significantly while for high disconnectivity and highly dynamic systems, increasing holding time does not significantly improve reachability.

Chapter 6

Conclusion and Future Work

In this chapter we summarize the work done in this dissertation, list our contributions and suggest future work in this area.

In this dissertation, we addressed four problems in the area of routing and rerouting in mobile wireless and ad-hoc networks. Our contributions in this dissertation are as follows:

- **Rerouting in Static-Mobile Connections**: We presented a framework for comparing, analyzing, and evaluating various Static-Mobile rerouting schemes for connection-oriented mobile networks. We used various performance metrics like total rerouting time, service disruption time and bandwidth requirements to aid in the comparison. We also provided a comprehensive survey and classification of rerouting schemes. In order to study various ways to improve the performance of network protocols, we studied parallelism in messaging in the rerouting protocol. To the best of our knowledge, this is the most comprehensive survey and comparative evaluation of rerouting in connection-oriented networks. We suggest mechanisms to improve the performance of rerouting

including radio-hints and parallel messaging.

- **Rerouting in Mobile-Mobile Connections:** We proposed a distributed rerouting algorithm framework for Mobile-Mobile connections in connection-oriented mobile networks. Adapting Static-Mobile rerouting to Mobile-Mobile connections suffers from many problems. The other rerouting schemes proposed for Mobile-Mobile connections by researchers perform non-optimal rerouting. We have compared our proposed re-routing algorithm with all the other Mobile-Mobile rerouting schemes using several performance metrics, and in all cases our algorithm outperforms the other schemes.

- **Evaluation of the effect of the various rerouting schemes on the performance of Wireless TCP schemes:** In a mobile environment, TCP can not discriminate packets dropped due to lossy links, intermittent connectivity, handoffs and the consequent rerouting with those dropped due to network congestion. Previous work on TCP and wireless environment dealt with links errors and mechanism to alleviate its effects on TCP's throughput. We have studied the effect of how rerouting schemes effect the performance of Wireless TCP using simulation. Our study is the first to analyze the effect of various rerouting schemes on the throughput of various wireless TCP's schemes.

- **Routing in Ad-Hoc Networks:** We propose an efficient distributed spanning tree based routing algorithm for ad hoc networks that adjusts dynamically to fast changing dynamic topologies of ad-hoc networks. We achieve this by maintaining a forest of spanning trees. Our approach is practical and has less time, space and message complexities when compared with the algorithms

described in literature.

6.1 Future work

There are several directions for future work in this area:

1. **Rerouting in Connection-Oriented Networks:**
   - We plan to extend the array of performance metrics for comparison of both Static-Mobile and Mobile-Mobile rerouting schemes. Another interesting research issue would be the study of the effect of traffic patterns and distribution on rerouting schemes on the system's performance. Finally, it would be interesting to show how our framework can be applied to a wireless ATM environment.
2. **Rerouting and Wireless TCP:**
   - We plan to extend our work by extending the comparison to other wireless retransmission schemes like link layer retransmission, explicit congestion notification and fast retransmissions. In addition, we plan to study how wireless TCP performs in Mobile-Mobile connections.
3. **Routing in Ad-hoc Networks:**
   - **Performance**: We intend to compare our algorithm with ZRP, TORA, DSDV, AODV, MCDS, SSA and other routing algorithms. Specifically, we plan to study the following - under what conditions and scenarios do each of these routing algorithm perform well? We look at performance in terms of message and time complexities, reachability and other quality of service metrics.

- **Quality of Service (QoS) issues:** We hope to answer several quality of service (QoS) issues like: What are the important QoS issues and performance metrics for these routing algorithms? Is it important for the packets to delivered in spite of delay? What are the ranges of the QoS metrics? Should the QoS contracts be dynamically changed based on how rapidly the topology is modifying?

- **User Mobility Profile and Traffic Issues:** We wish to study how can the mobility patterns of users be used to effectively improve the performance of the routing? In order to gather information of the user mobility pattern commonly referred to as the user's mobility profile. For example, a person going to office from his/her house starts at a certain time, uses a certain route, travels at a certain speed etc. Also, after reaching the office, the person has a certain mobility profile (meetings, visits to the copy room, cafeteria) which can be modeled.How do these routing algorithms perform in presence of different types of users namely, fast movers, slow movers? What kind of effect does the traffic distribution have on the performance of the routing algorithm? How can the routing algorithms be tuned to these different type of users?

- **Reliability:** How can one provide communication among users in a ad-hoc network with a good degree of reliability? How does a reliable communication protocol like TCP or RTP perform in ad-hoc networks? Is it possible to utilize the prevalent wireless TCP protocols and adapt them for use in ad-hoc networks?

Impact of Our Work

Our study and comparison of rerouting in Static-Mobile and Mobile-Mobile connections will help users and implementors decide in choosing amongst a given choice of rerouting algorithms to use or implement. Our proposed Mobile-Mobile rerouting algorithms has a lot of promise for mobile video and audio conferencing. In addition, we believe that our suggestion about improving the performance of TCP in wireless and mobile networks has a strong impact on a wide suite of applications including web browsing, telnet and ftp in mobile scenarios. We believe our proposed ad-hoc routing algorithms would result in efficient routing for many real-life applications, such as deployment of troops connected by mobile radios in a battle field situation and use of mobile radios to co-ordinate an emergency search and rescue mission.

Bibliography

[1] Anthony Acampora. An architecture and methodology for mobile-executed handoff in cellular ATM networks. *IEEE Journal of Selected Areas in Communications*, 12(8):1365–1375, October 1994.

[2] Arup Acharya and B. R. Badrinath. An efficient protocol for ordering broadcast messages in distributed systems. In *Proceedings of the Third IEEE Symposium on Parallel and Distributed Systems*, 1991.

[3] Arup Acharya and B. R. Badrinath. Delivering multicast messages in networks with mobile hosts. In *Proceedings of the 13th International Conference on Distributed Computing Systems*, pages 292–299, Pittsburgh, PA, USA, May 1993.

[4] Arup Acharya and B. R. Badrinath. Checkpointing distributed applications on mobile computers. In *Proceedings of the Third International Conference on Parallel and Distributed Information Systems*, Austin, TX, USA, October 1994.

[5] Arup Acharya and B. R. Badrinath. A framework for delivering multicast messages in networks with mobile hosts. *ACM/Baltzer Journal of Wireless Networks, Special Issue on Routing in Mobile Communication Networks*, 1995.

[6] Arup Acharya, Ajay Bakre, and B. R. Badrinath. IP multicast extensions for mobile internetworking. In *Proceedings of IEEE INFOCOM '96*, 1996. Technical Report LCSR-TR-243.

[7] Arup Acharya, T. Imielinski, and B. R. Badrinath. Dataman project: Towards a mosaic-like location dependent information service for mobile clients. Technical Report TR-320, Rutgers University, 1994.

[8] Norman I. Adams, Rich Gold, Bill N. Schilit, Michael M. Tso, and Roy Want. An infrared network for mobile computers. In *Proceedings of USENIX Symposium on Mobile and Location Independent Computing*, pages 41–51, Cambridge, Massachusetts, USA, August 1993.

[9] D. Agrawal and A. El Abbadi. An efficient solution to the distributed mutual exclusion problem. In *Proceedings of the 8th ACM Symposium on Principles of Distributed Computing*, 1989.

[10] P. Agrawal, E. Hyden, P. Krzyzanowski, P. Mishra, M. Srivastava, and J. Trotter. SWAN - a mobile multimedia wireless network. *IEEE Personal Communications*, 3(2):18–33, 1996.

[11] Tahir Ahamad, Mike Clary, Owen Densmore, Steve Gadol, Arthur M.Keller, and Robert Pang. The DIANA approach to mobile computing. Technical report, Stanford University, Sun Microsystems, 1993.

[12] I. A. F Akyildiz, J.I. Pelech, and B. Yener. A virtual topology based routing protcol for multihop dynamic wireless networks. Personal Communication, 1999.

[13] B. Akyol and D. C. Cox. Handling mobility in a wireless ATM network. In *Proceedings of IEEE INFOCOM '96*, pages 1405–1413, San Francisco, CA, USA, March 1996.

[14] B. Akyol and D. C. Cox. Rerouting for handoff in a wireless ATM network. In *Proceedings of IEEE ICUPC '96*, Cambridge, MA, USA, September 1996.

[15] Sridhar Alagar and S. Venkatesan. Causally ordered message delivery in mobile systems. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[16] R. Alonso and S. Ganguly. Energy efficient query optimization. Technical Report MITL–TR–33–92, Matasushia Information Technology Laboratory, December 1992.

[17] Rafael Alonso, Daniel Barbara, and Hector Garcia-Molina. Data caching issues in an information retrieval system. *ACM Transaction on Computer Systems*, pages 359–384, September 1990.

[18] Rafael Alonso, E. M. Haber, and H. F. Korth. A database interface for mobile computers. In *Proceedings of the IEEE GLOBECOM 92 Workshop on Networking of Personal Communications Applications*, Orlando, FL, USA, December 1992.

[19] Birger Andersen, Eric Jul, Francisco Moura, and Vitor M. P. Guedes. File system support for semiconnected operation in Amigos. In *Proceedings of the Second USENIX Symposium on Mobile and Location-Independent Computing*, December 1994.

[20] E. Ayanoglu, S. Paul, T. F. LaPorta, K. K. Sabnani, and R. D. Gitlin. Airmail: A link-layer protocol for wireless networks. *ACM/Baltzer Wireless Networks Journal*, 1:47–60, February 1995.

[21] Ashar Aziz. A scalable and efficient intra-domain tunneling Mobile-IP scheme. *Computer Communications Review*, 24(1):12–20, 1994.

[22] Ashar Aziz and Whitfield Diffie. Privacy and authentication for wireless local area networks. *IEEE Personal Communications*, 1(1):25–31, July 1993.

[23] B. R. Badrinath, Arup Acharya, and Tomasz Imielinski. Impact of mobility on distributed computations. *ACM Operating System Review*, 27(2):15–20, April 1993.

[24] B. R. Badrinath, Arup Acharya, and Tomasz Imielinski. Structuring distributed algorithms for mobile hosts. In *Proceedings of the 14th International Conference on Distributed Computing Systems*, Poznan, Poland, June 1994.

[25] B. R. Badrinath, Arup Acharya, and Tomasz Imilelinski. Designing distributed algorithms for mobile computing networks. *Computers and Communications*, 1994.

[26] B. R. Badrinath, Tomasz Imilelinski, and Aashu Virmani. Locating strategies for personal communication networks. In *Proceedings of the IEEE Globecom 92 Workshop on Networking of Personal Communications Applications*, Orlando, USA, December 1992.

[27] Aline Baggio. Disconnectable web proxy. Internal Coda group report, Carnegie Mellon University, September 1996.

[28] Aline Baggio. Cadmium - towards availability and adaptation in mobile environments (poster). In *Proceedings of the Middleware'98 Conference*, The Lake District, England, September 1998.

[29] Aline Baggio. Replication and caching strategies in Cadmium. Technical Report RR-3409, INRIA, April 1998.

[30] Ajay Bakre and B.R. Badrinath. I-TCP: Indirect TCP for mobile hosts. Technical report, Rutgers University, May 1995. Technical report DCS-TR-314 / WINLAB TR-89.

[31] Ajay Bakre and B.R. Badrinath. M-RPC: A remote procedure call service for mobile clients. Technical Report WINLAB-TR-98, Rutgers University, May 1995.

[32] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and R. H. Katz. A comparison of mechanisms for improving TCP performance over wireless networks. In *Proceedings of ACM SIGCOMM*. ACM, August 1996.

[33] H. Balakrishnan, S. Seshan, and R. H. Katz. Improving reliable transport and handoff performance in cellular wireless networks. In *Proceedings of ACM Mobicom*, Berkeley, CA, USA, November 1995. ACM.

[34] Carlos Baquero, Victor Fonte, Francisco Moura, and Rui Oliveira. MobiScape: WWW browsing under discpnnected and semi-connected operation. Technical report, Universidade do Minho, 1994.

[35] Daniel Barbara-Milla and Hector Garcia-Molina. Replicated data management in mobile environments: Anything new under the sun? In *Proceedings of the IFIP Working Conference on Applications in Parallel and Distributed Computing*, pages 237–246, April 1994.

[36] B. Barringer and T. Burd et al. Infopad: A system design for portable multimedia access. In *Proceeding of the Wireless 1994 Conference*, Canada, July 1994.

[37] Joel F. Bartlett. W4 - the Wireless World Wide Web. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[38] Joel F. Bartlett. Experience with a Wireless World-Wide Web client. Technical Report WRL-TN-46, Digital, WRL, March 1995.

[39] U. Baumgarten and J. Maier. Using mobile objects with a PDA. In *Proceedings of the ECOOP'95 Workshop on Mobility and Replication*, Aarhus, Denmark, August 1995.

[40] Carey B. Becker, Basvaraj Patil, and Emad Qaddoura. Ip mobility architecture framework. Internet Draft, 1999. http://www.ietf.org/internet-drafts/draft-ietf-mobileip-ipm-arch-00.txt. Last Ref. 05/08/99.

[41] Frazer Bennett, Tristan Richardson, and Andy Harter. Teleporting - making applications mobile. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[42] Pravin Bhagwat, Partho Mishra, and Satish K. Tripathi. Effect of topology on performance of reliable multicast communication. In *Proceedings of IEEE INFOCOM*, June 1994.

[43] Pravin Bhagwat and Charles Perkins. Highly dynamic destination-sequenced distance vector routing for mobile computers. *Proceedings of ACM SIGCOMM*, 1994.

[44] Pravin Bhagwat, Charles Perkins, and Felix Wu. Caching location data in mobile networking. In *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems*, October 1993.

[45] Pravin Bhagwat and Charles E. Perkins. A mobile networking system based on internet protocol (IP). In *Proceedings of the USENIX Symposium on Mobile and Location Independent Computing*, pages 69–82, Cambridge, Massachusetts, USA, August 1993.

[46] Pravin Bhagwat, Charles E. Perkins, and Satish K. Tripathi. Transparent resources discovery for mobile computers. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[47] S. K. Biswas. *Handling Real time Traffic in Mobile Networks.* PhD thesis, University of Cambridge, 1994.

[48] T. Blackwell, K. Chan, K. Chang, T. Charuhas, J. Gwertzman, B. Karp, H. T. Kung, D. Li, D. Lin, R. Morris, R. Polansky, D. Tang, and C. Young. Secure redirects in mobile IP. In *Proceedings of the USENIX Summer 1994 Conference*, Boston, USA, June 1994.

[49] M. Blaze. Transparent mistrust: Os support for cryptography-in-the-large. In *Proceedings of the 4th Workshop on Workstation Operating Systems*, pages 98–102, Napa, USA, October 1993.

[50] D. Buchholz, P. Odlzko, M. Taylor, and R. White. Wireless in-building network architecture and protocols. *IEE Network Magazine*, 5(6):31–38, November 1991.

[51] S. Bush. Handoff mechanism for mobile ATM systems. Internet Draft, 1995. http://tisl.ukans.edu/. Last Ref. 01/05/98.

[52] Ramon Caceres and Liviu Iftode. The effects of mobility on reliable transport protocols. In *Proceedings of the 14th International Conference on Distributed Computing Systems*, Poznan, Poland, May 1994.

[53] Ramon Caceres and Liviu Iftode. Improving the performance of reliable transport protocols in mobile computing environments. *IEEE JSAC Special Issue on Mobile Computing Network*, 1994.

[54] Anantha Chandraksan, Samuel Sheng, and R. W. Brodersen. Design considerations for a future portable multiledia terminal. In *Proceedings of the Third Generation Wireless Information Networks*, pages 75–97, 1992.

[55] D. Cheeseman. The pan-european cellular mobile radio system. *Personal and Mobile Radio Systems, IEE Telecommunications Series*, 25, 1991.

[56] David Chess, Bejamin Grosof, Colin Harrison, and David Levine. Itinerant agents for mobile computing. Technical Report RC 20010, IBM T.J. Watson Research Center, February 1995.

[57] Tzi-Cker Chiueh. Scheduling for broadcast-based file systems. Technical report, State University of New York at Stony Brook, October 1994.

[58] Kenjiro Cho and Kenneth P. Birman. A group communication approach for mobile computing. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[59] Panos K. Chrysanthis. Transaction processing in mobile computing environment. In *Proceedings of the IEEE Workshop on Advances in Parallel and Distributed Systems*, pages 77–83, Princeton, New Jersey, October 1993.

[60] R. Cohen and A. Segall. Connection management and rerouting in ATM networks. Internet Draft, 1998. http://www.cs.technion.ac.il/ rcohen/journal.html. Last Ref. 05/08/99.

[61] David A. Cooper and Kenneth Birman. Preserving privacy in a network of mobile computers. In *IEEE Symposium on Security and Privacy*, pages 26–38, May 1995.

[62] David Anthony Cooper and Kenneth P. Birman. The design and implementation of a private message service for mobile computers. *Wireless Networks*, 1995.

[63] Digital Equipment Corporation. Digital roamabout radio product guide. http://www.network.digital.com. Last Ref. 05/08/99.

[64] S. Corson, J. Macker, and S. Batsell. Architectural considerations for mobile mesh networking. Internet Draft RFC Version 2, May 1996. http://tonnant.itd.nrl.navy.mil/mmnet/mmnetRFC.txt. Last Ref. 05/08/99.

[65] D. Cox. Wireless network access for personal communications. *IEEE Communications Magazine*, 30(12):96–116, December 1992.

[66] B. Das and V. Bharghavan. Routing in ad-hoc networks using minimum connected dominating sets. Technical report, University of Illinois at Urbana-Champaign, 1996.

[67] B. Das, R. Sivakumar, and V. Bharghavan. Routing in ad-hoc networks using a virtual backbone. Technical report, University of Illinois at Urbana-Champaign, 1996.

[68] S. Das. Comparative performance evaluation of routing protocols for mobile, ad hoc networks. In *Proc. of Int Conf. on Computer Communications and Networks*, pages 153–161, 1998.

[69] Nigel Davies, Stephen Pink, and Gordon S. Blair. Services to support distributed applications in a mobile environment. In *Proceedings of the First International Workshop on Services in Distributed and Networked Environments*, Prague, Rpublique Tchque, June 1994.

[70] Alan Demers, Karin Petersen, Mike Spreitzer, Douglas Terry, Marvin Theimer, and Brent Welch. The Bayou architecture: Support for data sharing among mobile users. In *Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[71] A. Dixit, V. Gupta, and B. Lancki. Mobile-IP for Linux. Technical report, State University of New York, Buffalo, November 1995.

[72] F. Douglis. The compression cache: Using on-line compression to extend physical memory. Technical Report MITL–TR–18–92, Matsushita Information Technology Laboratory, 1992.

[73] Fred Douglis, Frans Kaashoek, Kai Li, Ramn Cceres, Brian Marsh, and Joshua A. Tauber. Storage alternatives for mobile computers. In *Proceedings of the First Symposium on Operating Systems Design and Implementation*, pages 25–37, Monterey, California, USA, November 1994.

[74] R. Drom. Dynamic host configuration protocol. IETF RFC 1541, October 1993. http://www.ietf.org/rfc/rfc1541.txt. Last Ref. 05/08/99.

[75] Rohit Dube, Cynthia D. Rais, Kuang-Yeh Wang, and Satish K. Tripathi. Signal stability based adaptive routing (ssa) for ad-hoc mobile networks. Internet Draft, August 1996. Also CS TR-3646, University of Maryland, College Park.

[76] D. Duchamp. Optimistic lookup of whole NFS paths in a single operation. In *Proceedings of the Summer USENIX Conference*, pages 161–169, Boston, June 1994.

[77] Maria R. Ebling. *Translucent Cache Management for Mobile Computing.* PhD thesis, Carnegie Mellon University, March 1998.

[78] Maria R. Ebling and M. Satyanarayanan. SynRGen: An extensible file reference generator. In *Proceedings of the 1994 ACM SIGMETRICS Conference*, Nashville, TN, USA, May 1994.

[79] K. Y. Eng, M. J. Carol, M. Veeraraghavan, E. Ayanoglu, C. B. Goodworth, P. Pancha, and R. A. Valenzuela. A wireless a broadband ad-hoc ATM

local-area network. *ACM/Baltzer Journal on Wireless Networks (WINET)*, 1(2):161–174, 1995.

[80] J. L. Buxton et al. The travelpilot: A second-generation automative navigation system. *IEEE Transactions on Vehicular Technology*, 40(1):41–44, February 1991.

[81] R. Yuan et al. Mobility support in a wireless ATM network. In *Proceedings of the 5th Workshop on Third Generation Wireless Information Networks*, pages 335–345. Kluwer Publishers, 1995.

[82] Innes A. Fergusson. *Touring Machines: An architecture for dynamic, rational, mobile agents.* PhD thesis, University of Cambridge, 1992.

[83] S. Fickas, G. Kortuem, and Z. Segall. Software organization for dynamic and adaptable wearable systems. In *Proceedings First International Symposium on Wearable Computers*, Cambridge, Massachussets, USA, October 1997.

[84] Marc E. Fiuczynski. Distributed transactions in a mobile computing system. Technical Report CSE-552, University of Washington, 1993.

[85] Telemedicine Gateway. National aeronautics and space administration. Internet Draft, May 1997. http://www.tmgateway.org/. Last Ref. 05/08/99.

[86] Stefan Gessler and Andreas Kotulla. PDAs as mobile WWW browsers. In *Second World Wide Web Conference'94, Mosaic and the Web*, October 1994.

[87] R. Ghai and Suresh Singh. An architecture and communication protocol for picocellular networks. *IEEE Personal Communications*, Third Quarter:36–47, 1994.

[88] Monarch Project Group. The cmu monarch project. Internet Draft, May 1999. http://www.monarch.cs.cmu.edu. Last Ref. 05/08/99.

[89] Richard D. Guy, John Heidemann, Wai Mak, Thomas W. Page, Gerald J. Popek, and Dieter Rothmeiner. Implementation of the Ficus replicated file system. Technical report, University of California, Los Angeles, 1990.

[90] James Gwertzman and Margo Seltzer. World-Wide Web cache consistency. In *Proceedings of the 1996 Usenix Technical Conference*, January 1996.

[91] Z. J. Haas and M.R. Pearlman. The zone routing protocol (ZRP) for ad hoc networks. Internet Draft, November 1997. Mobile Ad Hoc Networking Working Group.

[92] Rolf Hager, Anders Klemets, Gerald Q. Maguire, Mark T. Smith, and Frank Reichert. MINT - a mobile internet router. In *43th IEEE Vehicular Technology Conference*, Secaucus, New Jersey, USA, May 1993.

[93] Jorgen Svaerke Hansen, Torben Reich, and Birger Andersen. Semi-connected TCP/IP in a mobile computing environment. Technical report, University of Copenhagen, 1995.

[94] Andy Harter and Andy Hopper. A distributed location system for the active office. Technical report, Olivetti Research Laboratory, November 1993.

[95] C. Hedrick. Routing information protocol. Internet RFC 1058, June 1988. http://www.ietf.org/rfc/rfc1058.txt. Last Ref. 05/08/99.

[96] John Howard, Michael Kazar, Sherri Menees, David Nichols, Mahadev Satyanarayanan, Robert Sidebotham, and Michael West. Scale and performance in distributed file system. *ACM Transactions of Computer Systems*, 6(1):51–81, February 1988.

[97] Y. Huang, P. Sistla, and O. Wolfson. Data replication for mobile computers. In *Proceedings of the ACM SIGMOD Conference on Management of Data*, pages 13–24, 1994.

[98] Tomasz Imielinski and B. R. Badrinath. Wireless mobile computing: Solutions and challenges in data management. Technical Report DCS-TR-296/WINLAB-TR-49, Rutgers University, 1993.

[99] Motorola Inc. The Motorola Piano home page. Internet Document, May 1999. http://www.mot.com/GSS/SSTG/piano/index.html. Last Ref. 05/10/99.

[100] John Ioannidis. *Protocols for Mobile Internetworking.* PhD thesis, Columbia University, 1993.

[101] John Ioannidis and D. Duchamp. Protocols for supporting mobile IP hosts. Technical report, Columbia University, NY, USA, June 1992.

[102] EIA/TIA IS-41. Cellular radio telecommunications intersystem operations. Internet Draft, 1995. http://www.tiaonline.org/standards/. Last Ref. 05/08/99.

[103] I. Richer J. M. McQuillan and E.C. Rosen. The new routing algorithm for the arpanet. *IEEE Tran. on Communications*, 28(5):711–719, 1980.

[104] R. Jain, Y. B. Lin, C. N. Lo, and S. Mohan. A caching strategy to reduce network impacts of pcs. *IEEE Journal Selected Areas Communication*, 12(8):1434–1445, 1994.

[105] D. B. Johnson. Routing in ad hoc networks of mobile hosts. In *Proceedings of IEEE Workshop on Mobile Computing Systems and Applications*, December 1994.

[106] David B. Johnson. Ubiquitous mobile host internetworking. In *4th Workshop on Workstation Operating Systems*, pages 85–90, Napa, USA, October 1993.

[107] David B. Johnson. Routing in ad-hoc networks of mobile hosts. In *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

[108] D.B. Johnson and D.A Maltz. Protocols for adaptive wireless and mobile networking. *IEEE Personal Communications*, 3(1), 1996.

[109] Anthony D. Joseph, Joshua A. Tauber, and M. Frans Kaashoek. Mobile computing with the rover toolkit. *IEEE Transactions on Computers: Special issue on Mobile Computing*, March 1997.

[110] Anupam Joshi, T. Drashansky, S. Weerawarana, and E.N Houstis. Sciencepad: An intelligent electronic notepad for ubiquitous scientific computing. In *International Conference on Intelligent Information Management Systems*, pages 107–110, Washington, D.C., USA, 1995.

[111] Gerald Q. Maguire Jr., Mark T. Smith, and Tomoki Osawa. Walkstation II project. In *Proceedings of the 2nd International Workshop on Mobile Multi-Media Communications Workshop*, Bristol, England, April 1995.

[112] J. Jubin and J. D. Tornow. The DARPA packet radio network protocols. *Proceedings of the IEEE*, 75(1):21–32, January 1987.

[113] P. Karn and C. Partridge. Improving round-trip time estimates in reliable transport protocols. In *Proceedings of the ACM SIGCOMM*, August 1987.

[114] K. Keeton, B. A. Mah, S. Seshan, R.H. Katz, and D. Ferrari. Providing connection-oriented network services to mobile hosts. In *Proceedings of the USENIX Symposium on Mobile and Location Independant Computing*, pages 83–102, Cambridge, Massachusetts, USA, August 1993.

[115] S. Keshav and S. Morgan. Smart retransmission: Performance with overload and random losses. In *Proceedings of Infocom*, 1997.

[116] Anders Klemets, Gerald Q. Maguire, Frank Reichert, and Mark T. Smith. Mint - a mobile internet router. In *Proceedings of the 1st IEEE International Symposium on Global Data Networking*, Cairo, Egypt, December 1993.

[117] YoungBae Ko. Mobile computing related resources on www. Internet Draft, May 1999. http://www.cs.tamu.edu/people/youngbae/mcrelated.html. Last Ref. 05/08/99.

[118] P. Krishna, M. Chatterjee, N. H. Vaidya, and D. K. Pradhan. A cluster-based approach for routing in ad-hoc networks. In *USENIX Symposium on Location Independent and Mobile Computing*, April 1995.

[119] P. Krishna, N. H. Vaidya, and D. K. Pradhan. Location management in a distributed mobile environment. In *3rd International Conference of Parallel and Distributed Systems*, September 1994.

[120] P. Krishna, N. H. Vaidya, and D. K. Pradhan. Static and dynamic location management in a distributed mobile environment. Technical Report 94-030, Texas A&M University, 1994.

[121] N. Krishnakumar and R. Jain. Protocols for maintaining inventory databases and user service profiles in mobile sales applications. In *Mobidata Workshop*, 1994.

[122] G. S. Lauer. *Packet Radio routing*, chapter Chapter 11, pages 55–76. Prentice-Hall, Englewood Cliffs, NJ, 1995. Routing in Communications Networks. Edited by M. E. Steenstrup.

[123] Y. B. Lin. Determining the user locations for personal communications networks. *IEEE Transaction on Vehicular Technology*, 43(3):466–473, 1994.

[124] George Liu. Exploitation of location-dependent caching and prefetching techniques for supporting mobile computing and communications. In *6th International Conference on Wireless Communications*, Canada, July 1994.

[125] George Liu. Efficient mobility management for wireless mobile computing and communications. Technical Report IT-R 95:07, Royal Institute of Technology, March 1995.

[126] J. Macker. Mobile ad hoc networking. MILCOM 1997 Panel on Ad Hoc Networks, November 1997. Monterey, CA, USA.

[127] John McQuillan, Ira Richer, and Eric C. Rosen. The new routing algorithm for the arpanet. *IEEE Transaction on Communications*, 28(5):711–719, May 1980.

[128] William Mendenhall. *Introduction to Probability and Statistics.* Duxbury Press, Boston, MA, USA, 6th edition, 1987.

[129] D. Mosberger and L. Peterson. Careful protocols or how to use highly reliable networks. In *4th Workshop on Workstation Operating Systems*, Napa, USA, October 1993.

[130] J. Moy. OSPF version 2. Internet Request for Comments RFC 1247, July 1991. http://www.ietf.org. Last Ref. 05/08/99.

[131] J. Naylon, D. Gilmurray, J. Porter, and A.Hopper. Low-latency handover in a wireless ATM LAN. *IEEE Journal on Selected Areas in Communications*, 16(6):909–921, August 1998.

[132] L. J. Ng, R. W. Donaldson, and A. D. Malyan. Distributed architectures and database for intelligent personal communication networks. In *ICWC*, June 1992.

[133] A. R. Noerpel, Y. B. Lin, and H. Sherry. PACS: Personal access communications system - a tutorial. *IEEE Personal Communications Magazine*, 1995.

[134] M. Srivastava P. P. Mishra. Effect of connection rerouting on application performance in mobile networks. In *Proceedings of International Conference on Distributed Computing Systems*, pages 371–390, 1997.

[135] Charles Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *Proceedings of the ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, August 1994.

[136] John Postel. Transmission Control Protocol. Internet RFC, January 1980. http://www.scit.wlv.ac.uk/ c9451595/rfc/rfc7xx/RFC761.html. Last Ref. 05/08/99.

[137] G. Racherla, S. Radhakrishnan, and C. N. Sekharan. Rerouting strategies for connection-oriented mobile networks with both ends being mobile. Technical report, School of Computer Science, University of Oklahoma, 1997.

[138] G. Racherla, S. Radhakrishnan, and C. N. Sekharan. A distributed rerouting algorithm for mobile-mobile connections in connection-oriented mobile networks. In *Proceedings of the Seventh International Conference on Computer Communication and Networks (ICCCN)*, pages 40–44, September 1998.

[139] Gopal Racherla, Sridhar Radhakrishnan, and Chandra N. Sekharan. A framework for evaluation of rerouting in connection-oriented mobile networks. Technical report, School of Computer Science, University of Oklahoma, 1998.

[140] Subhashini Rajgopalan and B. R. Badrinath. Adaptive location management for mobile-IP. In *Proceedings of the first ACM International Conference on Mobile Computing and Networking - Mobicom'95*, November 1995.

[141] S. Ramanathan and M. Steenstrup. A survey of routing techniques for mobile communication networks. *Mobile Networks and Applications*, pages 89–104, 1996.

[142] R. Ramjee, T. F. La Porta, J. Kurose, and D. Towsley. Performance evaluation of connection rerouting schemes for ATM-based wireless networks. *IEEE/ACM Transactions on Networking*, 6(3):249–261, June 1998.

[143] M. Satyanarayanan. Scalable, secure, and highly available distributed file access. *IEEE Computer*, 23(5), 1990.

[144] Bill N. Schilit, Norman I. Adams, Rich Gold, Michael M. Tso, and Roy Want. The ParcTab mobile computing system. In *4th Workshop on Workstation Operating Systems*, Napa, USA, October 1993. http://sandbox.parc.xerox.com/parctab/.

[145] Bill N. Schilit, Fred Douglis, David M. Kristol, Paul Krzyzanowski, James Sienicki, and John A. Trotter. TeleWeb: Loosely connected access to the World-Wide Web. *Computer Networks and ISDN Systems*, 28(7–11):1431, May 1996.

[146] S. Seshan. *Low Latency Handoff for Cellular Data Networks.* PhD thesis, University of California, Berkeley, 1995.

[147] S. Singh and C. S. Raghavendra. PAMAS: Power aware multi-access protocol with signalling for ad hoc networks. *ACM Computer Communications Review*, July 1998.

[148] M. Song, Y. Choi, and C. Kim. Connection-information-based connection rerouting for connection-oriented mobile communication networks. *Distributed Systems Engineering*, 5:47–65, 1998.

[149] R. Sridhar, Gopal Racherla, C. N. Sekharan, N. S. V. Rao, and S. G. Batsell. A routing protocol for ad hoc networks using spanning trees. Technical report, School of Computer Science, University of Oklahoma, 1998.

[150] Carl D. Tait. *A File System for Mobile Computing.* PhD thesis, Columbia University, August 1993.

[151] R. Tanaka and M. Tsukamoto. A CLNP-based protocol for mobile end systems within an area. In *1993 International Conference on Network Protocols*, pages 64–71, October 1993.

[152] A. Tanenbaum. *Computer Networks.* Prentice Hall International, 3rd edition, 1997.

[153] Fumio Teraoka, Yasuhiko Yokote, and Mario Tokoro. A network architecture providing host migration transparency. In *ACM SIGCOMM'91*, September 1991.

[154] F. Teroaka and K. Uehara. The virtual network protocol for host mobility. *Internet Draft*, 1993.

[155] C-K. Toh. The design and implementation of a hybrid handover protocol for multimedia wireless LANs. In *Proceedings of ACM Mobicom*, pages 49–61, Berkeley, CA, USA, November 1995.

[156] C-K. Toh. Performance evaluation of crossover switch discovery algorithms for wireless ATM LANs. In *Proceedings of IEEE INFOCOM*, pages 1380–1387, San Francisco, CA, USA, March 1996.

[157] C-K. Toh. A unifying methodology for handovers of heterogeneous connections in wireless ATM networks. *Computer Communication Review*, pages 12–30, 1997.

[158] C-K. Toh. *Wireless ATM and Ad-Hoc Networks: Protocols and Architectures.* Kluwer Academic Publishers, Londo, U.K., first edition, 1997.

[159] Huseyin Uzunalioglu, Wei Yen, and I. F. Akyildiz. A connection handover protocol for leo satellite ATM networks. In *Proceedings of ACM Mobicom*, Budapest, Hungary, 1997.

[160] Hiromi Wada, T. Ohnishi, and B. Marsh. Packet forwarding for mobile hosts. Technical Report MITL-TR-39-92, Matasushia Information Technology Laboratory, December 1992.

[161] Hiromi Wada, Takashi Yozawa, Tatsuya Ohnishi, and Yasunori Tanaka. Mobile computing environment based on internet packet forwarding. In *Proceedings of USENIX Winter 1994 Technical Conference*, San Diego, CA, USA, Jan 1993.

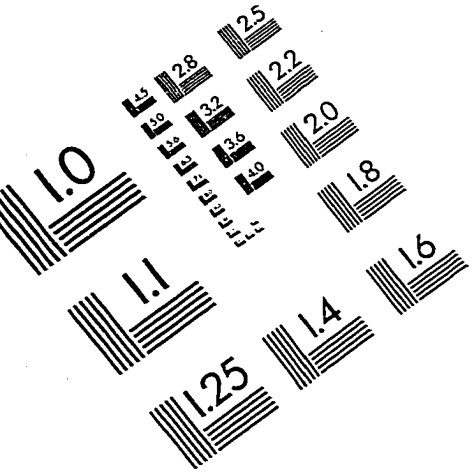
[162] Roy Want, Andy Hopper, Veronica Falcao, and Jonathan Gibbons. The active badge location system. *ACM Transactions on Information Systems*, 10(1):91–102, January 1992.

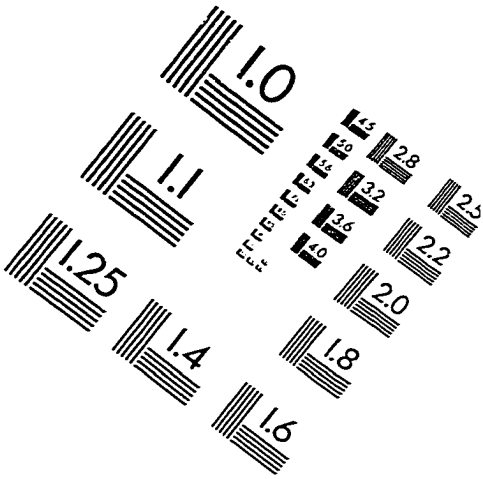
[163] G. Welling and B.R. Badrinath. Mobjects: Programming support for environment directed application policies in mobile computing. In *Proceedings of the ECOOP '95 Workshop on Mobility and Replication*, Aarhus, Denmark, August 1995.

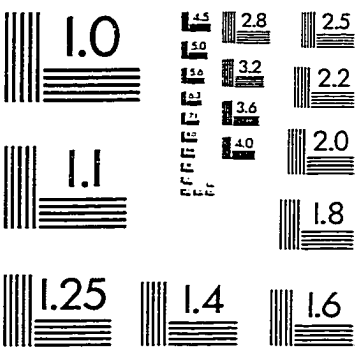
[164] O.T.W. Wu and V.C.M. Leung. B-ISDN architectures and protocols to support wireless personal communications internetworking. In *Proceedings of PIMRC '95*, Toronto, Canada, September 1995.

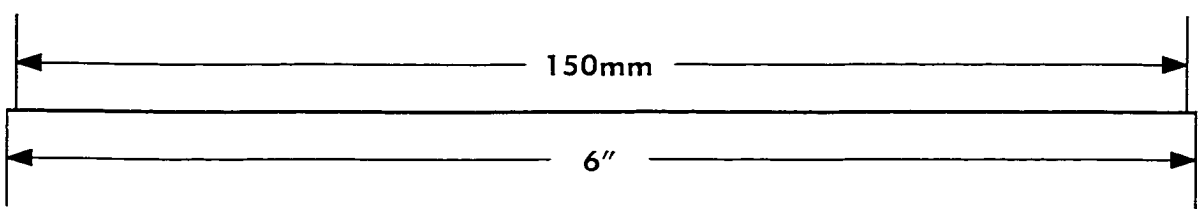
[165] Raj Yavatkar and Namrata Bhagawat. Improving end-to-end performanec of tcp over mobile internetworks. In *Proceedings of the IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, USA, December 1994.

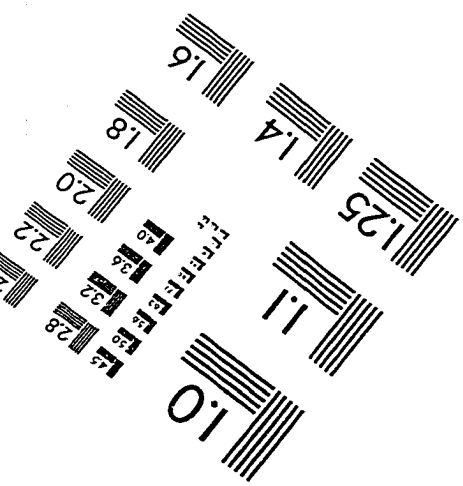
IMAGE EVALUATION TEST TARGET (QA-3)

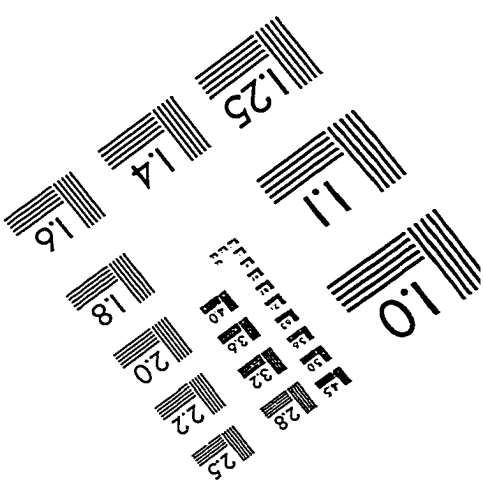












APPLIED IMAGE, Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989