Mobility Pattern Aware Routing in Mobile Ad Hoc Networks

by
Savyasachi Samal

Thesis submitted to the Faculty of the
Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science
in
Computer Science

Dr. Amitabh Mishra, Chairman
Dr. Srinidhi Varadarajan, Co-Chairman
Dr. Luiz DaSilva, Committee Member

May 2003
Blacksburg, Virginia

Mobility Pattern Aware Routing in Mobile Ad Hoc Networks

Savyasachi Samal

(ABSTRACT)

A Mobile ad hoc network is a collection of wireless nodes, all of which may be mobile, that dynamically create a wireless network amongst them without using any infrastructure. Ad hoc wireless networks come into being solely by peer-to-peer interactions among their constituent mobile nodes, and it is only such interactions that are used to provide the necessary control and administrative functions supporting such networks. Mobile hosts are no longer just end systems; each node must be able to function as a router as well to relay packets generated by other nodes. As the nodes move in and out of range with respect to other nodes, including those that are operating as routers, the resulting topology changes must somehow be communicated to all other nodes as appropriate. In accommodating the communication needs of the user applications, the limited bandwidth of wireless channels and their generally hostile transmission characteristics impose additional constraints on how much administrative and control information may be exchanged, and how often. Ensuring effective routing is one of the greatest challenges for ad hoc networking.

As a practice, ad hoc routing protocols make routing decisions based on individual node mobility even for applications such as disaster recovery, battlefield combat, conference room interactions, and collaborative computing etc. that are shown to follow a pattern.

In this thesis we propose an algorithm that performs routing based on underlying mobility patterns. A mobility pattern aware routing algorithm is shown to have several distinct advantages such as (a) A more precise view of the entire network topology as the nodes move (b) A more precise view of the location of the individual nodes (c) Ability to predict with reasonably accuracy the future locations of nodes (d) Ability to switch over to an alternate route before a link is disrupted due to node movements.

Acknowledgements

I would like to thank my advisor, Dr. Amitabh Mishra, for his able guidance during each stage of this research. His encouragement was a great source of motivation for me. I would especially like to thank him for his extensive comments on the thesis write up, as they have helped improve my writing skills and have also helped me to present my research in a much better way.

A special word of thanks goes to Dr. Luiz DaSilva for his critique on the write up and for the valuable suggestions, which I received from him when I was in the midst of defining my problem statement. I am also grateful to Dr. Srinidhi Varadarajan for his constructive comments on the thesis.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

With the ever-increasing acceptance of mobile ad-hoc networks in areas like disaster recovery, battlefield scenarios, conference room scenarios, collaborative computing, and many others, the demands placed on these types of networks have massively expanded. With the increase in demand for various types of applications, the need for efficient routing algorithms is also becoming a major requirement. The fulfillment of this requirement has been a complex problem mainly due to the lack of fixed infrastructure. The absence of fixed infrastructure for ad hoc networks means that the nodes communicate directly with one another in a peer-to-peer fashion. The mobility of these nodes imposes limitations on their power capacity, as well as their transmission range. Mobile hosts are no longer just end systems; each node must be able to function as a router, and also must relay packets generated by other nodes. As the nodes move in and out of range with respect to one another, including those that operate as routers, the resulting topology changes must somehow be communicated to all other nodes so the up-to-date topology information for routing purposes is maintained. In addition, the communication needs of the user applications, the limited bandwidth of wireless channels, and the generally hostile transmission characteristics all impose additional constraints on the type, size, and frequency of information to be exchanged. Thus ensuring effective routing is one of the greatest challenges for ad hoc networking.

1.1 Motivation

Routing in ad hoc networks has been an active area of research for some time now. In an effort to maximize the throughput of ad hoc networks, researchers have proposed various routing algorithms, which take into account some of the above-mentioned sources of service impairment in an ad hoc network. Significant work has been done on routing in ad hoc networks, some of the important works so far are the destination-sequence distance vector (DSDV) protocol [1], wireless routing protocol [2], the temporally ordered routing protocol [3], the spine based routing algorithm [4] and the zone routing protocol [5], dynamic source routing protocol [6] and ad hoc on demand routing protocol [7]. The emphasis in these routing algorithms has been on providing the shortest path between the source and the destination, with an attempt to provide a high degree of route availability. But with the increasing acceptance of mobile ad hoc networks, the demands now being placed on the performance of these types of networks have increased with the main difficulty in achieving these goals being the non-deterministic nature of the network topology in ad hoc networks. This non-determinism is an inherent property of ad hoc networks, but there are cases where the mobile node movements are predictable to some extent and this thesis deals mostly with such type of ad hoc networks.

Thus going by intuition, a possible solution to the problem of routing in ad hoc networks arising due to in-deterministic network topology could be, if by some means of classification we could separate out those networks which show some amount of determinism in their movement patterns to those that are totally random. Thus now we

have a set of networks where the node motions would be deterministic to some extent and in these types of networks because of the deterministic and predictable nature of node movements, we can optimize on the routing overheads. This intuition forms the basis of our research.

To accommodate the above restrictions and requirements, the research has now been shifting to the mobility characterization of the mobile nodes (i.e., an attempt is being made to quantify the randomness). Hence, mobility management in wireless ad hoc networks has received significant attention from recent literature as in [8-18]. Most of the research so far, however, has been to characterize the individual mobility models being followed by the nodes [19, 20]. But considering only the mobility of single nodes in the network is of little use, as in most of the scenarios the nodes in an ad hoc network instead show cohesive properties. This observation stems from the basic fact of why ad hoc networks came into existence (i.e., to assist people working in groups). Later [20-24] appeared which bring forth the effect of other nodes in a network on the movement of a particular node and further show the importance of treating the mobility as a pattern being followed by a group of nodes or the network as a whole than by an individual node.

With the realization of the importance of mobility, in coming up with an efficient routing algorithm for ad hoc networks, some routing algorithms as described in [25-28] have been put forward which consider the effect that mobility of nodes has on the efficiency of routing. However none of the algorithms proposed so far have addressed the impact of the various **mobility patterns**[1] (e.g. in battle field or rescue scenarios where nodes show particular movement patterns) followed by the nodes. Thus in researching the impact of different types of mobility patterns on transience of the routes, various types of non-determinism, reduction of routing overhead, ease in guaranteeing the quality of service (QoS), or whether the extra burden of mobility management has any effect on the efficiency of the underlying routing algorithm at all in the first place becomes an interesting field of study.

Therefore, in this thesis, we propose a solution for routing in ad hoc networks by tackling the core issue of random movements by first identifying the **mobility patterns** that the underlying nodes are following. Identifying the mobility pattern of these nodes assumes a certain importance, as it not only allows us to approximate the topology of the entire network, but also provides a way to handle the mobility of individual nodes as well. Finally, identifying such a pattern provides a means to control the random nature of the mobile nodes. This information is crucial for guaranteeing QoS as well. For example, now we can (1) predict with reasonable accuracy the future location of a node based on its mobility pattern; (2) switch over to an alternate route before a link is disrupted because of a node moving away from the path to a link (revealed by the mobility pattern), and (3) approximate the degree of imprecision in the state information (bandwidth, delay etc) of individual nodes. For example, consider the battlefield scenario as shown in *figure 1.1*, where even though various platoons of soldiers move randomly among themselves, they all ultimately are going towards the goal. The fact that nodes are moving in small groups, and all the groups are following a general direction, are both being used in our algorithm to form a wired-line type infrastructure.

1. For more detailed information on mobility patterns please refer to chapter 3.

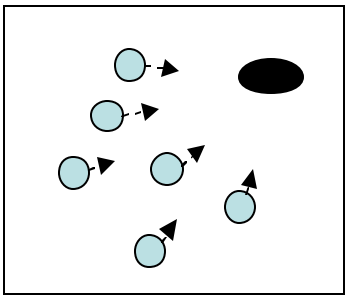


Figure 1.1: "*Soldiers moving as a group toward the target in a battlefield scenario*"

However, characterizing every possible mobility pattern in ad hoc networks where the motion is basically random in nature is difficult, if not impossible. So in our approach, we have tried to classify mobility as either highly predictable random motion or not-so predictable random motion or totally random motion, which we have referred to as the deterministic model, semi-deterministic model, and random model respectively. This classification broadly covers most of the mobility patterns being followed by the nodes (e.g., the urban traffic pattern, the battlefield pattern, or a rescue operation pattern).

By restricting the mobility patterns to only three types as mentioned above we made our problem domain small i.e. now the routing algorithm only has to distinguish between three types of mobility patterns. Thus in the attempt to identify the mobility patterns followed by the nodes in an ad hoc network, the routing algorithm here uses a new data structure (i.e., **history**), which is an ordered pair of the past visited locations of a node and the corresponding time stamps at which the node was at that particular location. The idea of using history to identify the mobility pattern, and henceforth predict future locations, is very much similar to the mathematical problem of extrapolation, from a given set of data points. But mere extrapolation is not easy, with a given set of data points, as the mobility pattern doesn't fall into any exact mathematical equation. So based on the differences among the collected past locations of an individual node, we intend to categorize its movement to a particular mobility pattern. This is an important aspect of our algorithm, as along with the ability to characterize the mobility pattern of the nodes; this information helps us to predict possible future locations of the node, which are further used for routing performance enhancements. We assume that this method is not error free, so we also attempt to minimize the errors.

Keeping in view the above mentioned motivation, we propose a mobility pattern aware DSDV routing algorithm in this thesis and perform its performance comparison with the standard DSDV.

1.2 Thesis Organization

The organization of the rest of the thesis is as follows: Chapter 2 - "Background," presents a brief review of the existing routing algorithms in ad doc networks. Chapter 3-4, the proposed research; Chapter 5, our simulation methodology and results; and Chapter

6, the educational component part of this proposal along with some words on future extensions.

1.3 Thesis Contributions

This thesis proposes a mobility pattern aware routing algorithm for wireless ad hoc networks based on DSDV.

All these items can be summed up in the following list:

1. Classify the various types of mobility models being followed by nodes in mobile ad hoc networks.
2. Simulate various types of mobility models, so as to validate the proposed classification of mobility.
3. Propose a modified routing algorithm based on DSDV incorporating the location information only.
4. Extend the modified DSDV to be mobility pattern adaptive, where we decouple the routing algorithm from any prior knowledge of the underlying mobility pattern.
5. Evaluate the performance of the new routing algorithm with respect to standard DSDV.

1.4 Chapter Summary

In this chapter, we looked into the mobility related issues that have to be considered while designing routing protocols for ad hoc networks. Next we provided the motivation for our thesis and presented an organizational overview of how the thesis is organized.

Chapter 2: Background

2.1 Introduction to Ad Hoc Networks

A wireless mobile ad hoc network consists of mobile nodes that are interconnected by wireless-multi-hop communication paths. These ad hoc wireless networks are self-creating, self-organizing, and self-administering. *Figure 2.1* depicts a sample mobile ad hoc network.

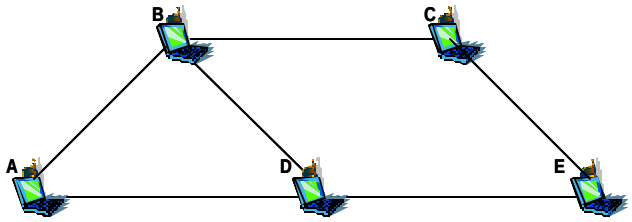


Figure 2.1: *"A sample ad hoc network consisting of mobile nodes"*

These mobile ad hoc networks offer unique benefits and versatility for certain environments and applications. With no prerequisites of fixed infrastructure or base stations, they can be created and used anytime, anywhere. Such networks could be intrinsically fault-resilient, for they do not operate under the limitations of a fixed topology. Since all nodes are allowed to be mobile, the topology of such networks is necessarily time varying. The addition and deletion of nodes occur only by interactions with other nodes. Thus these types of networks offer many advantages where setting up wired-line networks is not feasible. Such advantages attracted immediate interest in its early use among military, police, and rescue agencies, and especially under disorganized or hostile environments, including isolated scenes of natural disaster and armed conflict. Recently, home or small-office networking and collaborative computing with laptop computers in a small area (e.g., a conference or classroom, single building, convention center, etc.) have emerged as other major areas of application. In addition, people have recognized from the beginning that ad hoc networking has an obvious, potential use in all the traditional areas of interest for mobile computing.

This concept of mobile ad hoc networking emerged as an attempt to extend the services provided by the traditional Internet to the wireless mobile environment. All current works, as well as our research here, consider the ad hoc networks as a wireless extension

to the Internet, based on the ubiquitous IP networking mechanisms and protocols. Today's Internet possesses an essentially static infrastructure where network elements are interconnected over traditional wire-line technology, and these elements, especially the elements that provide the routing or switching functions, do not move. In a mobile ad hoc network, by definition, all the network elements move. As a result, numerous more stringent challenges must be overcome to realize the practical benefits of ad hoc networking.

In addition, the mobility of nodes imposes limitations on their power capacity, and hence, on their transmission range. These nodes must often also satisfy stringent weight limitations for portability. Further these mobile hosts are no longer just end systems; they are also required to relay packets generated by other nodes, hence each node must also be able to function as a router. As the nodes move in and out of range with respect to each other, including those that are operating as routers, the resulting topology needs to be communicated to all other nodes as appropriate to maintain the connectivity information. In accommodating the communication needs of the user applications, the limited bandwidth of wireless channels and their generally-hostile transmission characteristics, impose additional constraints on how much administrative and control information may be exchanged, and how often. Ensuring effective routing is one of the major challenges for ad hoc networking.

As these mobile ad hoc networks are being increasingly considered for more and more complex applications, the various Quality of Service (QoS) attributes for these applications must also be satisfied as a set of pre-determined service requirements. In addition, because of the increasing use of the ad hoc networks for military/police use, and also due to the increasing commercial applications being envisioned to be supported on these type of networks, various security issues also need to be addressed. These issues, however, are out of the scope of this thesis.

The organization of the rest of the chapter is as follows. Section 2.2 presents a brief review of the operating principles of an ad hoc network and introduces some networking concepts pertinent to routing and QoS. The general issue of routing in mobile ad hoc networks is reviewed in Section 2.3. This section also cites some of the existing routing protocols in mobile ad hoc networks, with special emphasis on the DSDV (Destination Sequence Distance Vector) algorithm, as this algorithm is the basis of our research.

2.2 Ad Hoc Wireless Network: Operating Principles

To illustrate the general operating principles of a mobile ad hoc network, consider *figure 2.2*, which depicts the peer-level, multi-hop representation of a sample ad hoc network. Here, mobile node A communicates directly (single-hop) with another such node B whenever a radio channel with adequate propagation characteristics is available between them. Otherwise, multi-hop communication is necessary where one or more intermediate nodes must act as a relay (router) between the communicating nodes. For example, there is no direct radio channel (shown by the lines) between A and C or between A and E as shown in *figure 2.2*. Nodes B and D must serve as intermediate routers for communication between A and C, and between A and E, respectively. Thus, a distinguishing feature of ad hoc networks is that all nodes must be able to function as routers on demand along with acting as source and destination for packets. To prevent packets from traversing infinitely long paths, an obvious essential requirement for choosing a path is that it must be loop-free. And this loop-free path between a pair of nodes is called a route.

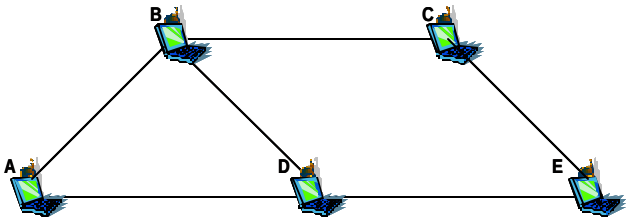


Figure 2.2: *"Example of an ad hoc network"*

An ad hoc network begins with at least two nodes, broadcasting their presence (*beaconing*) with their respective address information. If node A is able to establish direct communication with node B as in *figure 2.2*, verified by exchanging suitable control messages between them, they both update their routing tables. When a third node C joins the network with its beacon signal, two scenarios are possible. The first is where both A and B determine that single-hop communication with C is feasible. The second is where only one of the nodes, say B, recognizes the beacon signal from C and establishes direct communication with C. The distinct topology updates, consisting of both address and route updates, are made available in all three nodes immediately afterwards. In the first case, all routes are direct. For the other, the route update first happens between B and C, then between B and A, and then again between B and C, confirming the mutual reachability between A and C via B.

As the node moves, it may cause the reachability relations to change in time, requiring route updates. Assume that, for some reason, the link between B and C is no longer available as shown in *figure 2.3*. Nodes A and C are still reachable from each other, although this time only via nodes D and E. Equivalently, the original loop-free route $\langle A \leftrightarrow B \leftrightarrow C \rangle$ is now replaced by the new loop-free route $\langle A \leftrightarrow D \leftrightarrow E \leftrightarrow C \rangle$. All five nodes in the network are required to update their routing tables appropriately to

reflect this topology change, which will be first detected by nodes B and C, then communicated to A, E, and D.

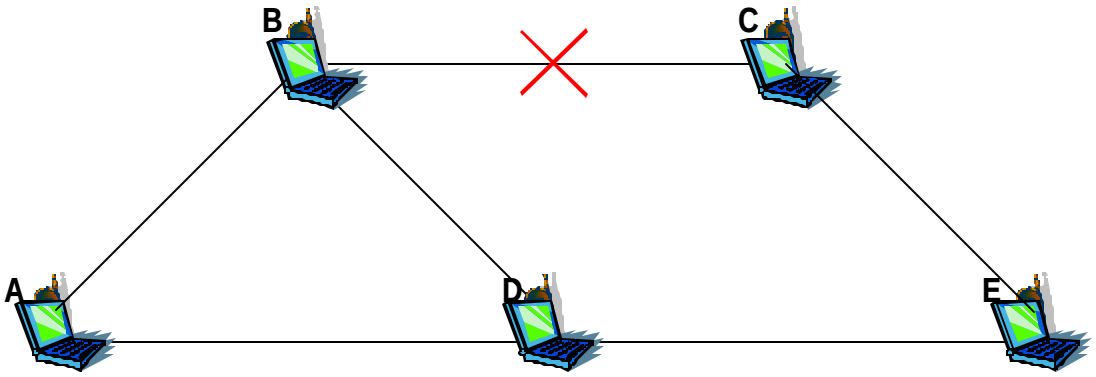


Figure 2.3: *"Topology update owing to a link failure"*

This reachability relation among the nodes may also change for various reasons. For example, a node may wander too far out of range, its battery may be depleted, or it may just suffer from software or hardware failure. As more nodes join the network, or some of the existing nodes leave, the topology updates become more numerous, complex, and usually, more frequent, thus diminishing the network resources available for exchanging user information (i.e., data).

Finding a loop-free path between a source-destination pair may therefore become impossible if the changes in network topology occur *too frequently*. *Too frequently* here means that there may not be enough time to propagate to all the pertinent nodes the changes arising from the last change in network topology. Thus the ability to communicate degrades with increasing mobility and as a result the knowledge of the network topology becomes increasingly inconsistent. A network is combinatorially stable if, and only if, the topology changes occur slow enough to allow successful propagation of all topology updates as necessary or if the routing algorithm is efficient enough to propagate the changes in the network before the next change occurs. Clearly, combinatorial stability is determined not only by the connectivity properties of the networks, but also by the efficiency of the routing protocol in use and the instantaneous computational capacity of the nodes, among others. Combinatorial stability thus forms an essential consideration for attaining efficient routing objectives in an ad hoc network.

In the next section, we list some of the characteristics that an efficient routing algorithm for an ad hoc network must posses in order to increase the throughput of the network with the least amount of overhead.

2.2.1 Desirable properties of an efficient routing algorithm

Many routing protocols for ad hoc networks have been proposed so far, each one offering some advantage over the previous approach. But in general, there are some common desirable properties that any routing protocol for an ad hoc network should possess as mentioned in [30]. These are:

- Loop free: Presence of loops in the path from the source to the destination result in inefficient routing. In the worst-case situation, the packets may keep traversing the loop indefinitely and never reach their destination.

- Distributed control: In a centralized routing scheme, one node stores all the topological information and makes all routing decisions; therefore, it is neither robust, nor scalable. The central router can be a single point of failure; also, the network in the vicinity of the central router may get congested with routing queries and responses.

- Fast routing: The quicker the routing decisions are made, the sooner the packets can be routed towards the destination, as the probability that the packets take the chosen route before it gets disrupted because of node mobility is quite high.

- Localized reaction to topological changes: Topological changes in one part of the network should lead to minimal changes in routing strategy in other distant parts of the network. This will keep the routing update overheads in check and make the algorithm scalable.

- Multiplicity of routes: Even if node mobility results in disruption of some routes, other routes should be available for packet delivery.

- Power efficient: A routing protocol should be power efficient. That is the protocol should distribute the load; otherwise shut-off nodes may cause partitioned topologies that may result in inaccessible routes.

- Secure: A routing protocol should be secure. We need authentication for communicating nodes, non-repudiation and encryption for private networking to avoid routing deceptions.

- QoS aware: A routing protocol should also be aware of Quality of Service. It should know about the delay and throughput for a source destination pair, and must be able to verify its longevity so that a real-time application may rely on it.

2.3 Routing in Mobile Ad Hoc Networks

Now that we have a general overview of an ad hoc network and its operating principles, we move on to take a deeper look into the various approaches for routing in ad hoc networks proposed so far.

Several approaches toward routing in ad hoc networks have been proposed with the goal of achieving efficient routing. And with the ever-changing topology, an intuitive approach of routing messages could be that the sender of the message specifies the exact path that the message should take from the sender to the receiver. But this assumes that the sender knows the entire topology of the network, which is not quite a realistic assumption. Another alternative could be to forward the message to a neighbor in the general direction of the destination, and the neighbor then makes a similar decision regarding how to route the message.

Based on the above-mentioned approaches, many routing algorithms have been put forward. And as we have mentioned earlier, most of the routing algorithms proposed so far for ad hoc networks have been adapted from the routing techniques employed in wire-line networks that have fixed network topology. But the routing algorithms devised for wire-line networks are based on the determination of the shortest path between the source and the destination. Hence these routing algorithms cannot be applied to ad hoc networks without modification as the network topology in ad hoc networks is always in a state of flux with changes in graph topology in one region of the network altering the shortest path between several pairs of nodes. Also, in ad hoc networks, this information takes time to propagate to other nodes in the system. If the topology information is not updated promptly, the nodes may continue to route messages based on the outdated information, which may lead to packet losses.

Having seen the general approach to routing in ad hoc networks, let us take a closer look.

2.3.1 Broad Classification of Routing algorithms for Ad Hoc networks

Having seen the composition of mobile ad hoc networks, a centralized routing scheme is completely ruled out, as this might lead to a single point of failure. This leads to the requirement of a distributed routing protocol where every node takes part in making routing decision and maintaining the topology by sharing information among them. These distributed routing protocols can be classified broadly, first by how they intend to determine the topology of the network, and second by when they make a decision to find a route to a destination. The first category of topology intended routing can further be classified into hierarchical and flat routing, and the second category can further be divided into proactive and reactive routing.

2.3.1.1 Hierarchical Routing

In hierarchical routing algorithms as described in [31], a set of nodes is divided into clusters. Each cluster has a node, which is designated as the cluster-head. So, every node is either a cluster head or one wireless hop away from the cluster head as shown in *figure 2.4*. A node that is not a cluster head, but adjacent to more than one cluster head, is referred to as a gateway. Packets between cluster-heads are routed through gateways. Finally, nodes that are neither cluster heads nor gateways are referred to as ordinary nodes. The subnet comprising the cluster heads and gateways is referred to as the backbone network. Here, each cluster head maintains information about other nodes in its cluster, and from time to time, this information is exchanged between cluster heads over the network. Thus, the cluster heads gather network topology information. A node that has a packet to send to another node can obtain routing information from its cluster head. It is not necessary for a packet to be routed through the backbone network, as data packets may be routed along other more efficient routes in the network.

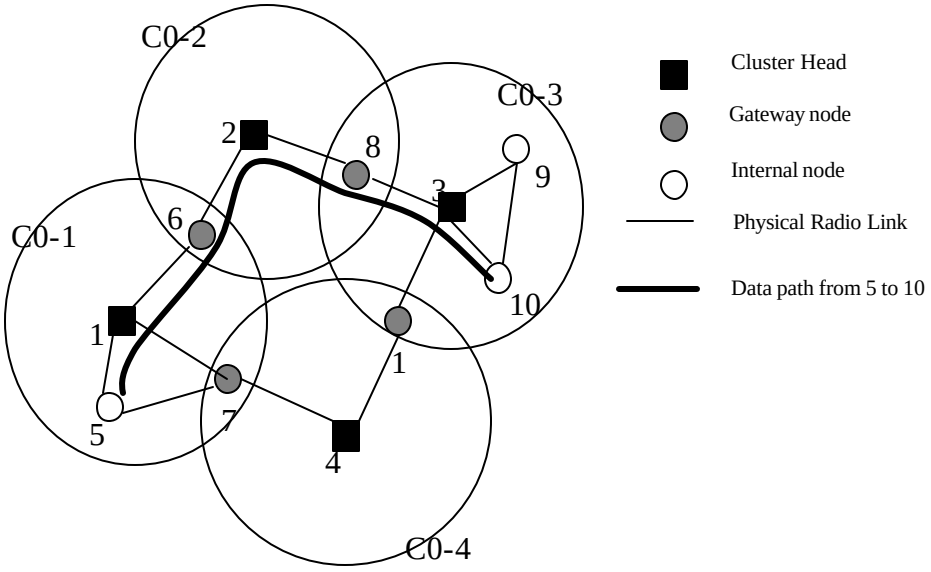


Figure 2.4: *"An example of clustering"*

This approach is also termed as topology aware routing as the nodes use the knowledge of the network topology to route messages. There are several ways of implementing this approach. The first possibility is that each node determines the optimal path to every node in the system and stores this information. Each time a stream of packets has to be sent from a source to a destination, a connection is established between two end-points and all the packets follow this path. However, with a changing topology, nodes will have to update their routing information and reestablish paths that were broken during communication. If the network topology does not change very often, it is likely that the path establishment costs are incurred once in the beginning and every subsequent packet

is routed without additional overhead. The second possibility is connectionless routing, where a route is determined on the fly for every packet as it moves from one node to another. This method will require nodes to store less information about network topology. However, every packet incurs the routing overhead.

2.3.1.2 Flat Routing

In case of flat routing algorithms all nodes act as routers and share the responsibility of forwarding packets destined to other nodes. Thus, there is no need to elect cluster heads and periodically reorganize the network. Most flat routing algorithms try to implement a distributed version of the shortest path algorithm or flooding where the general approach is that the sender sends a copy of the message to every neighbor. The neighbors then propagate copies of the message to all their neighbors except the one from which they received the message. This process is repeated until the network is entirely flooded with the message. If the destination node is in the same partition as the source, the message is sure to reach the destination.

2.3.1.3 Proactive Routing

In proactive routing algorithms, each node maintains a routing table containing the next hop information for every other node in the network, and hence a route between the source node and the destination node is always available making the approach proactive. Examples of proactive protocols include DSDV and the Fisheye State Routing (FSR) [32].

2.3.1.4 Reactive Routing

In reactive routing algorithms, a path discovery process determines the path to the destination only when the node has a packet to forward that is it reacts to a request to send data to a host. These types of routing algorithms are also referred to as on-demand routing protocols. Two prominent examples are Dynamic Source Routing (DSR) and Ad-hoc On demand Distance Vector (AODV) routing algorithm.

We presented a broad classification of the various approaches proposed so far, but different authors use different classification schemes for these basic routing functions. Regardless of the approach, routing protocols for ad hoc networks need to perform a set of basic functions in the form of route identification and route reconfiguration. For communication to be possible, at least one route (i.e., a loop-free path) must exist between any node pair. Route identification functions, as the name suggests, identify a route between a pair of nodes. Route reconfiguration functions are invoked to recover from the effects of undesirable events, such as host or link failures of various kinds, and traffic congestion appearing within a sub-network. Evidently, the recognition of changes in the network topology and the topology update functions constitute an important part of the route reconfiguration functions. A separate category of resource management

functions are also considered to ensure that all the network resources are available, to the extent possible, in support of some special objectives, such as those associated with QoS or security.

2.3.2 Well known routing algorithms

Based on the various approaches toward routing in mobile ad hoc networks described in the last section, now we take a look at the specific algorithms proposed in the literature.

2.3.2.1 Destination Sequenced Distance Vector (DSDV)

The Destination Sequence Distance Vector is the best-known protocol for a pro-active routing scheme. It is based on the classical Distributed Bellman Ford (DBF) routing algorithm for wired networks. DSDV is a table driven routing algorithm, like every other table driven routing algorithm, the nodes here maintain routing tables, which provide information about every possible destination within the network. A sample routing table in DSDV is shown in *Table 2.1*.

Table 2.1: "*Example routing Table entry in DSDV*"

Destination Addr.	Metric (no. of hops)	Addr. of next hop	Sequence no.
Host 1	2	host 2	S1
Host 2	1	host 2	S2

Here each table must contain the destination node address, the minimum number of hops to that destination and the next hop in the direction of that destination. The tables in DSDV also have an entry for sequence numbers for every destination. These sequence numbers form an important part of DSDV as they guarantee that the nodes can distinguish between stale and new routes. Here each node is associated with a sequence number and the value of the sequence number is incremented only by the node the sequence number is associated with. Thus, these increasing sequence numbers here emulate a logical clock. Suppose a node receives two updates from the same source, then the receiving node here makes a decision as to which update to incorporate in its routing table based on the sequence number. A higher sequence number denotes a more recent update sent out by the source node. Therefore it can update its routing table with more actual information and hence avoid route loops or false routes.

Having seen the table entries in DSDV, let us see how DSDV works. DSDV determines the topology information and the route information by exchanging these routing tables, which each node maintains. The nodes here exchange routing updates whenever a node detects a change in topology. When a node receives an update packet, it checks the sequence number in the packet. If the information in the packet is older than the receiving node has in its routing tables, then the packet is discarded. Otherwise, information is

updated appropriately in the receiving node's routing table. The update packet is then forwarded to all other neighboring nodes (except the one from which the packet came). In addition, the node also sends any new information that resulted from the merging of the information provided by the update packet. The updates sent out in this case, by nodes resulting from a change, can be of two types that is either a full update or a partial update. In case of full updates, the complete routing table is sent out and in case of a partial updates only the changes since last full update are sent out.

2.3.2.2 Dynamic Source Routing (DSR)

DSR uses a modified version of source routing. Operation of DSR can be divided in two functions – route discovery and route maintenance. Route discovery operation is used when routes to unknown hosts are required. Route maintenance operation is used to monitor correctness of established routes and to initiate route discovery if a route fails.

In DSR, when a node needs to send a packet to a destination it does not know about, the source node will initiate *route discovery*. The node sends route discovery request to its neighbors. The neighbors can either send a reply to the initiator or forward the route request message to their neighbors after having added their address to the request message (i.e. source routing). Every node that receives the route request message does the following:

- If the node has already seen this request, then the request is discarded.
- If the node has not seen it, but the route request message already has address of this host, then also it is discarded.
- Otherwise, if this host is the target of the route discovery message, then it appends the address of this host and returns it to the initiator of the route request message. The route request packet contains the route from the initiator to this host, which is the destination.
- If this host is not the destination, then just append the host's address in the packet and forward it to all of hosts' neighbors.

The route reply message can be returned to the initiator in two ways. If the host that sends reply already has the route to the initiator, it can use that route to send the reply. If not, it can use the route in the route request message to send the reply.

Route maintenance is performed when there is an error with an active route. When a node that is part of some route detects that it cannot send packets to next hop, it will create a Route Error message and send it to the initiator of data packets. The Route Error message contains the addresses of the node that sent the packet and of the next hop that is unreachable. When the Route Error message reaches the initiator, the initiator removes all routes from its route cache that have address of the node in error. It then initiates route discovery for a new route if needed.

2.3.2.3 Ad Hoc On-Demand Distance Vector (AODV)

Ad hoc On-Demand Distance Vector Routing is also an on demand routing algorithm, but in contrast to DSR it is a not source based routing scheme rather every hop of a route maintains the next hop information by its own.

Operation of the protocol here is also divided in two functions – *route discovery* and *route maintenance*. At first all the nodes send *Hello* message on its interface and receive *Hello* messages from its neighbors. This process repeats periodically to determine neighbor connectivity. When a route is needed to some destination, the protocol starts route discovery. The source sends *Route Request Message* to its neighbors. If a neighbor has no information on the destination, it will send message to all of its neighbors and so on. Once request reaches a node that has information about the destination (either the destination itself or some node that has a valid route to the destination), that node sends *Route Reply Message* to the *Route Request Message* initiator. In the intermediate nodes (the nodes that forward *Route Request Message*), information about source and destination from *Route Request Message* is saved. Address of the neighbor that the *Route Request Message* came from is also saved. In this way, by the time *Route Request Message* reaches a node that has information to answer *Route Request Message*; a path has been recorded in the intermediate nodes. This path identifies the route that *Route Request Message* took and is called reverse path. Since each node forwards *Route Request Message* to all of its neighbors, more than one copy of the original *Route Request Message* can arrive at a node. When a *Route Request Message* is created at the initiator, it is assigned a unique id. When a node receives *Route Request Message*, it will check this id and the address of the initiator and discard the message if it had already processed that request.

A node that has information about route to the destination sends *Route Reply Message* to the neighbor from which it received *Route Request Message*. This neighbor then does the same. This is possible because of the *reverse path* created by the *Route Request Message*. While the *Route Reply Message* travels back using *reverse path*, that path is being transformed into *forward path*, by recording the node that *Route Reply Message* came from (i.e. same procedure as mentioned above just in opposite direction). When *Route Reply Message* reaches the initiator, the route is ready, and the initiator can start sending data packets.

If one of the links on the forward path breaks, the intermediate node just above the link that failed sends new *Route Reply Message* to all the sources that are using the forward path to inform them of the link failure. It does this by sending the message to all neighbors using the *forward path.* In turn, they will send to their neighbors until all upstream nodes that use *forward path* are informed. The source nodes can then initiate new route request procedures if they still need to route packets to the destination.

2.3.2.4 Temporally Ordered Routing Algorithm

Temporally Ordered Routing Algorithm (TORA) is a distributed routing protocol based on a "link reversal" algorithm. It also discovers its routes on demand, provide multiple routes to a destination, establish routes quickly, and minimize communication overhead by localizing the reaction to topological changes when possible. Route optimality (shortest-path routing) is considered of secondary importance, and longer routes are often used to avoid the overhead of discovering newer routes. It is also not necessary (nor desirable) to maintain routes between every source/destination pair at all times.

2.3.2.5 Zone Routing Protocol (ZRP)

The Zone Routing Protocol (ZRP) is also called a hybrid ad hoc routing protocol as it combines proactive and reactive schemes. The ZRP maintains routing information for a local zone, and establishes routes on demand for destinations beyond its logical neighborhood. It limits the scope of the local zone by defining a maximum hop number for the local zone (e.g. 3 hops). Using ZRP with a maximum hop count of zero for the local neighborhood creates a reactive routing algorithm, and using it with maximum hop count as infinity creates a pure proactive routing algorithm. A route to a destination within the local zone can be established from the proactively cached tables of the source node. The routing algorithm used in the local zone can be based on any table driven routing algorithm, but it has to be extended in a way, that packets contain the “time to live” (TTL) information, which describes the maximum hop count of the local zone. For routes beyond the local zone, route discovery happens reactively. The source node sends a route request to its border nodes, containing its own address and a unique sequence number. Border nodes are nodes, which are exactly the maximum number of hops to the defined local zone away from the source. The border nodes check their local zone for the destination. If the requested node is not a member of this local zone, the node adds its own address to the route request packet and forwards the packet to its border nodes. If the destination is a member of the local zone of the node, it sends a route reply on the reverse path back to the source. The source node uses the path saved in the route reply packet to send data packets to the destination. The main advantage of the ZRP is a reduced number of required RREQ messages and further on the possibility to establish new routes without the necessity to completely flood the network. The main disadvantage is the increased complexity of the routing algorithm.

2.3.2.6 Clusterhead Gateway Switch Routing Protocol (CGSR)

In CGSR [34], a set of nodes is divided into clusters as shown in *figure 2.4*. Each cluster has a node, which is designated as the cluster-head. So, every node is either a cluster head or one wireless hop away from the cluster head. A node that is not a cluster head, but adjacent to more than one cluster head, is referred to as a gateway. Packets between cluster-heads are routed through gateways. Finally, nodes that are neither cluster heads nor gateways are referred to as ordinary nodes. The subnet comprising the cluster heads and gateways is referred to as the backbone network. Here, each cluster head maintains

information about other nodes in its cluster, and from time to time, this information is exchanged between cluster heads over the network. Thus, the cluster heads gather network topology information. A node that has a packet to send to another node can obtain routing information from its cluster head. It is not necessary for a packet to be routed through the backbone network, as data packets may be routed along other more efficient routes in the network. In a dynamic network, cluster heads can cause performance degradation problems because of frequent cluster-head elections, so CGSR uses a Least Cluster Change (LCC) algorithm. In LCC, cluster head change occurs only if a change in network causes two cluster heads to come into one cluster, or if one of the nodes moves out of the range of all the cluster heads. When a source transmits the packet to its cluster head, it is forwarded to gateway nodes. And this continues until the destination is reached.

2.3.3 Recent Advances in Routing Algorithms

In the last section we explained some of the well known routing algorithms proposed for mobile ad hoc networks. In addition to these basic routing schemes, many variants of these routing algorithms have also been proposed. But in all these algorithms, the nodes were assumed to be blind; that is, there was no means by which nodes could get their spatial information with respect to other nodes in the network. In fact, the only way they could obtain information about the network topology was by sharing information among them. Recently, a different approach called the location-based (or position-based) routing was put forward. In addition to the topology-based information, these protocols also use information about the physical location of the mobile nodes, and to determine their respective positions, nodes often use the Global Positioning System (GPS). A recent survey, including comparative information on the time and communication complexities of various protocols of this category, is presented in literature [35-45].

The next stage of refinements in the routing algorithms came in when the existing traditional routing algorithms became inefficient in handling large ad hoc networks. Consider the average size of the routing table if the size of the network grew to 1000 nodes. Then each node would be having a table containing 1000 records, which would measure up to considerable overhead when updating neighbors. And thus innovative routing schemes were needed to overcome such issues.

2.3.3.1 Location Aided Routing (LAR)

Location Aided Routing (LAR) [46] is a variant of the DSR routing scheme and works on demand. The aim of the protocol is to limit route discovery flooding, so as to reduce control packet overhead. It accomplishes this by using GPS data. Using this information, the flooding is done within a defined rectangular region. Mobile nodes here can also store the speed of the target node, so that it can predict where the target node will possibly be at a later point of time.

A distinguishing characteristic of these location-based protocols is that, to forward packets, a node only requires three pieces of information, that is its own position, that of the destination, and those of its adjacent (one-hop) neighbors. A transmitting node uses a location service to determine the location of the destination, and includes this information as part of the destination address in its messages. Routes do not need to be established or maintained explicitly; thus, there is no need to store routing tables at the nodes, and no need for routing table updates. Typically adjacent nodes are identified by broadcasting limited range beaconing messages and by employing various time-stamping mechanisms. The beaconing message includes distance limits; a receiving node discards the message if its location lies beyond the distance limit.

The availability of accurate location information at each node is essential for location-based routing to work, which, in turn, requires timely and reliable location updates as nodes change their locations. One or more nodes, designated to act as location servers, coordinate these location service functions, which are necessarily decentralized because of mobility. A large part of the ongoing research, as the references cited above show, is focused on designing efficient location services.

2.3.3.2 Global State Routing (GSR)

Global State Routing (GSR) [47] is similar to DSDV, with changes to reduce the overhead, which normal DSDV would incur with increasing network sizes. In GSR, each node keeps only its neighbor's link state in its tables. The next hop table and distance table is also kept. The topology table contains the link state information as reported by the destination and the timestamp of the information. The next hop table contains the next hop to which the packets for a destination node should be forwarded. The distance table keeps the shortest distance to each destination node.

As in all link state protocols, route update messages are generated upon link change. Upon receiving a routing message, a node updates its topology table if the sequence number of the message is higher than the value stored in the table. Fisheye methods, where the approach is to restrict the propagation of topology changes to a subset of the network if the effect is not so profound due to the change, works fine since it will work with both small and large networks. Thus with Fish eye method, nodes keep the most accurate topology information about the surrounding region, and know a little about outer regions. These methods are known to scale well.

2.3.3.3 Hierarchical State Routing (HSR)

Hierarchical State Routing (HSR) [48] employs a multilevel clustering and logical partitioning scheme. The network is partitioned into clusters and a cluster-head is elected as in a cluster-based algorithm. Cluster heads again organize themselves into clusters up to any desired clustering level as shown in *figure 2.5*. Within a cluster, nodes broadcast

their link information to one another. A cluster head summarizes its cluster information and sends it to neighboring clusters through a gateway node. A gateway node is one, which is adjacent to one or more cluster heads. Here cluster heads are members of a higher-level cluster. At each level, summarization and link information exchanges are performed. The way the information is exchanged in this hierarchy is, first information is collected among the nodes forming the base level cluster, it is then passed on to the cluster head which in turn passes to its next hierarchical cluster head and from there on the information is disseminated into other cluster heads and thus the information traverses down the hierarchy. Here each node has a hierarchical address, which may be obtained by assigning numbers from the top root to the bottom node. But as a gateway can be reached from the root from more than one path, so a gateway can have more than one hierarchical address.

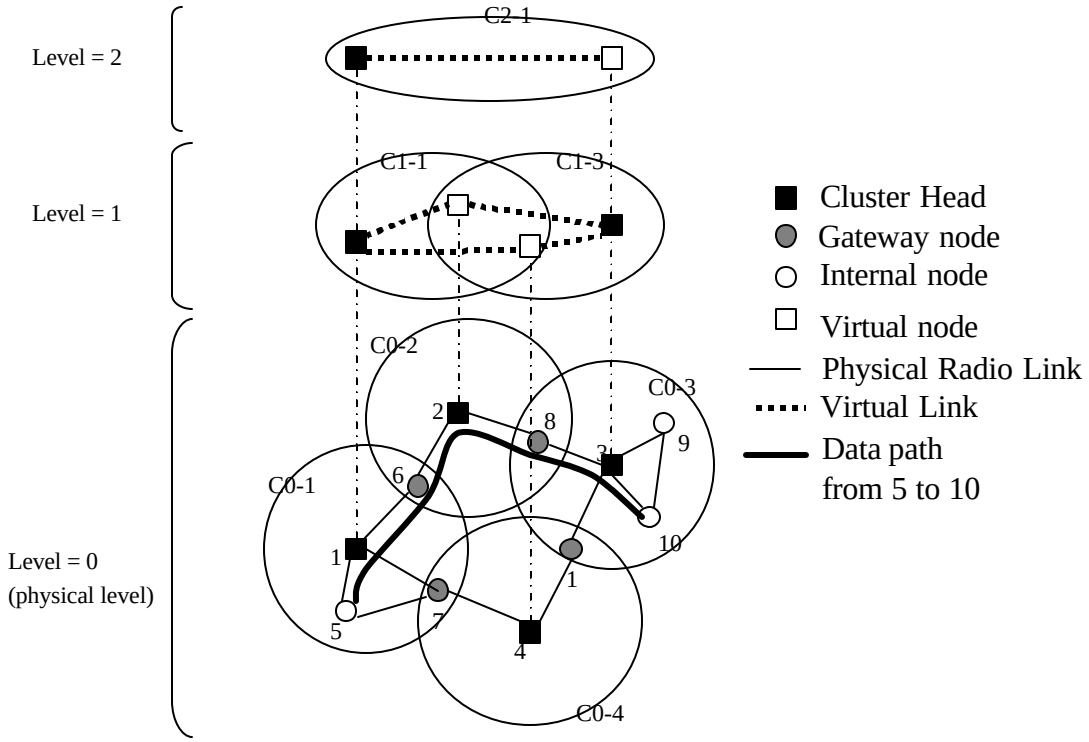


Figure 2.5: "*An example of clustering, forming hierarchies*"

Also, each subnet contains a location management server (LMS). All nodes in the subnet are registered with the local LMS. LMS has to inform upper levels, and upper level information comes to local LMS server. When two nodes wish to communicate, they send their initial data to the LMS, and the LMS then forwards it to the destination. But if the source and destination know each other's hierarchical addresses, they communicate directly. The protocol is highly adaptive to network changes. A location management scheme, but with ticket advertisement, was proposed by Stojmenovic [49]. There are other clustering based studies that are adaptive to mobility, one such study is an adaptive method [50], which relies on the probability of path availability to provide a basis for dynamically changing clusters. There is also a study by Liang *et.al.* [51] on creating a

virtual backbone on a multi-hop network. Their study mainly focused on providing reliable location information to establish and maintain connectivity among virtual backbone nodes.

2.3.3.4 Power and QoS Aware Protocols

As the acceptance of ad hoc networks grew, these networks were envisioned to support varied types of applications such as multimedia applications and etc. Thus with growing applications the needs put forth in the routing algorithm became more and more constrained. Some applications needed QoS support, where as, some had security as their main concern and there were some which required the network to be up and running for a longer period of time and hence in such types of networks power efficiency became a major issue to be tackled. So in literature there are protocols [52] that optimize parameters for power-efficient. One of those studies is Singh's [53], and their protocol considers the following metrics:

- Minimize energy consumed per packet: This implies the shortest-hop path.
- Maximize time to network partition: Here, the routing protocol tries to divide work amongst nodes to maximize life of the network.
- Maximize variance in node power levels: This implies the load-sharing of distributed systems, and is equivalent to bin-packing problem.
- Minimize cost per packet: Here, the goal is to maximize the life of individual nodes in the network, including costs other than energy.
- Minimize the maximum node cost: Here the minimization is done for per-node energy consumption of contributing nodes for forwarding packets.

There are also QoS related studies [54] on ad-hoc networks. To provide QoS, there is a bandwidth reservation proposal [55] that takes care of QoS parameters, such as to provide required bandwidth for real-time traffic on a route. Another study about QoS is Core Extraction Distributed Ad hoc Routing algorithm (CEDAR) [56]. In this protocol, QoS information is calculated at specific core nodes and they are propagated to other nodes forming the core of the network for overall QoS awareness in the network. A similar QoS related study can be found in the literature [57], which introduce the QoS requirements of wireless mobile networks and make single modified routing protocol (e.g. DSDV) simulations, in general.

2.4 Chapter Summary

In this chapter we presented an introduction to ad hoc networks and discussed the general operating principles of the routing algorithms as well as the properties that are desirable in a routing protocol in ad hoc networks. We also touched on the classification of the routing algorithms presented so far, followed by detailed discussion on well known routing protocols. Further in *table 2.2* we summarize the similarities and differences among some of the well-known routing protocols presented here.

Table 2.2: *"Comparison of some of the ad-hoc routing protocols"*

Protocol Property	DSDV	AODV	DSR	ZRP	TORA
Loop Free	Yes	Yes	Yes	Yes	Yes
Multiple Routes	No	No	Yes	No	Yes
Distributed	Yes	Yes	Yes	Yes	Yes
Reactive	No	Yes	Yes	Variable	Yes
Unidirectional Link Support	No	No	Yes	No	Yes
QoS Support	No	No	No	No	No
Multicast	No	Yes	No	No	No
Security	No	No	No	No	Possible
Power Efficiency	No	No	No	No	No
Periodic Broadcast	Yes	Yes	No	Yes	Yes

Chapter 3. Mobility Patterns and History

In the previous chapter, we looked at how routing is performed in ad hoc networks. In doing so, we highlighted the challenges arising due to the mobility of nodes, which the routing algorithms have to deal with. In this chapter, we take a closer look at how mobility affects routing in these types of networks and what can be done to tackle them.

3.1 Need for Characterization of Mobility

All the research in ad hoc networks till now has been in pursuit of one goal that is, trying to increase the throughput of the network with minimum possible overhead. And we argue that for efficient operation of the routing protocols it is necessary that the correct network topology information is available to the nodes in the network so that packets could be forwarded correctly between a sender and a receiver. In order for the nodes to have a correct topological view of the network, it is required that the nodes have reasonably correct location information of the nodes in their neighborhood. If the nodes are stationary, then their location information remains unchanged which leads to a correct topological view of the network and routing should work well as it does in the wire-line networks. But in ad hoc networks, the nodes move to different locations overtime, thus creating a different network topology. Almost all well-known routing protocols are shown to perform poorly for a network where the topology is changing at random.

With the realization of importance of node mobility in the routing process of ad hoc networks, quite a lot amount of work has been done in mobility characterization of the mobile nodes. This mobility characterization research is attempting to quantify the randomness in the mobility of the nodes [17,18]. Most of the research in this area of mobility characterization has however been towards mobility characterization of individual nodes [19].

Considering individual node mobility is of minor significance in ad hoc networks as the nodes in the ad hoc network show cohesive properties. This observation is directly related to the very existence of ad hoc networks that is, to support group collaboration and/or group activities. The effect of movement of neighboring nodes on a movement of an individual node was recently reported in [22]. In this paper authors argued about the importance of treating the mobility of mobile nodes as a group or the network as a whole, over the mobility model followed by an individual. But the paper does not throw any light on how the group mobility information can be used in improving the performance of routing algorithms.

Thus as can be seen, the major thrust of most of the routing algorithms proposed so far, merely have been an attempt to mitigate the effects arising due to mobility that is, either by letting the nodes in the network constantly update each other of the latest neighborhood information or by flooding. Either ways, it leads to increased overhead. But what most of the algorithms so far have over looked is the main cause, which leads to this instability in the topology i.e. the mobility of the nodes.

Thus it is the mobility of the nodes that renders the network topology in a state of flux, resulting in the degradation in performance of the ad hoc networks. And a possible solution to the problem of routing in ad hoc networks arising due to non-deterministic network topology could be, if by some means of classification we could separate out those networks which show some amount of determinism in their movement patterns to those that are totally random. Thus now we have a set of networks where the motion would be deterministic to some extent and in these types of networks because of the deterministic and predictable nature of node movements, we can optimize on the routing overheads.

Hence, in our research, we have attempted to resolve the issue of mobility by trying to define mobility into various patterns. There by attempting to characterize the mobility of the nodes into some pre-identified mobility patterns where the movements can be approximated to simple movement pattern and we can then use this information to bringing in predictability to motion.

After having looked at the need for mobility characterization in the routing algorithms, let us now take a look at how mobility can be characterized and how the various mobility patterns used in this thesis evolve out of this characterization.

3.2 Mobility Patterns

Considering the importance of node mobility in designing an effective routing algorithm for mobile ad hoc networks, the focus of recent research, as in [70,73], has now been shifting towards mobility characterization. Most of the prior research on ad hoc networks however assumed that the nodes follow random mobility model, as in [1-7]. And this random model was simulated as "pause and move," with the next move's direction and speed being independent of previous move made by the nodes. However, it is reasonable to expect a high degree of correlation between future and past movement speeds and direction. Thus, while each node acts separately from all other entities, its decisions for the next move are somehow affected by the context it is in and by the movements surrounding it. And if we could extract this dependence factor, we could bring in some amount of predictability into the next movement which the node will make

To illustrate the predictability of mobility patterns, consider the fleet of buses operated by the BT (Blacksburg Transit) in the Virginia Tech area in Blacksburg, Virginia. The movement of each bus is not completely random; else the whole purpose of the bus service would be defeated. Thus given the parameters such as the speed limits, the route to be followed and the time-checks, it is quite possible, at any given time, to predict the location of the buses with reasonable amount of accuracy. On the other hand, situations may arise where nodes do not move in a predetermined way. For example, consider a soldier in action in an earthquake rescue mission. Here, the soldier might follow a random path trying to cover as much area as possible, and perhaps communicating with

other members of the rescue squad. In this case, the mobility pattern tends to be more unpredictable. Hence to cover all the scenarios of mobility patterns from totally random to highly predictable random motion we classify mobility into three patterns as **deterministic (**highly predictable random motion**)**, **semi-deterministic (**not so predictable random motion**)** and **random.** The classification is shown in *figure 3.1*.

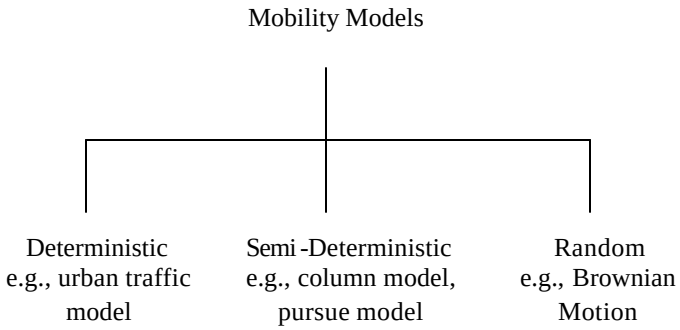


Figure 3.1: *"General Classification of Mobility"*

Before we look into each of the mobility patterns, we explain the basis of our classification. Consider *figure 3.2,* it shows a mobile node at *position 1* at instant *t*, and at *position 2* at instant *t + ?t*. In each position, the mobile node is associated with a direction vector. This direction vector may be different for each position. Here we argue that that, it is the deviation of the newly acquired direction vector from the old one, which dictates the type of mobility pattern being followed by the nodes. This deviation is shown in *figure 3.2* as ϕ.

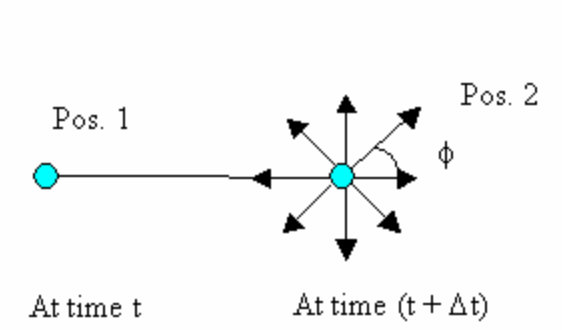


Figure 3.2: *"Node movement with respect to time"*

3.2.1 Deterministic Mobility Model

Deterministic mobility model is the most predictable type of motion and the most simplistic of all mobility models. For example, if the mobile nodes were moving in a

straight-line (i.e., following a deterministic model), then the deviation of the direction vectors associated with any two positions would be zero, as the mobile nodes continue to move in the same direction. A sample scenario resembling a deterministic mobility model is followed would be, cars moving in an urban traffic area, where the speed of the cars is restricted and the direction in which the cars can move are also predefined i.e., either in straight-line or turning only at cross lights. This scenario is depicted in *figure 3.3*, which is called the urban traffic model.

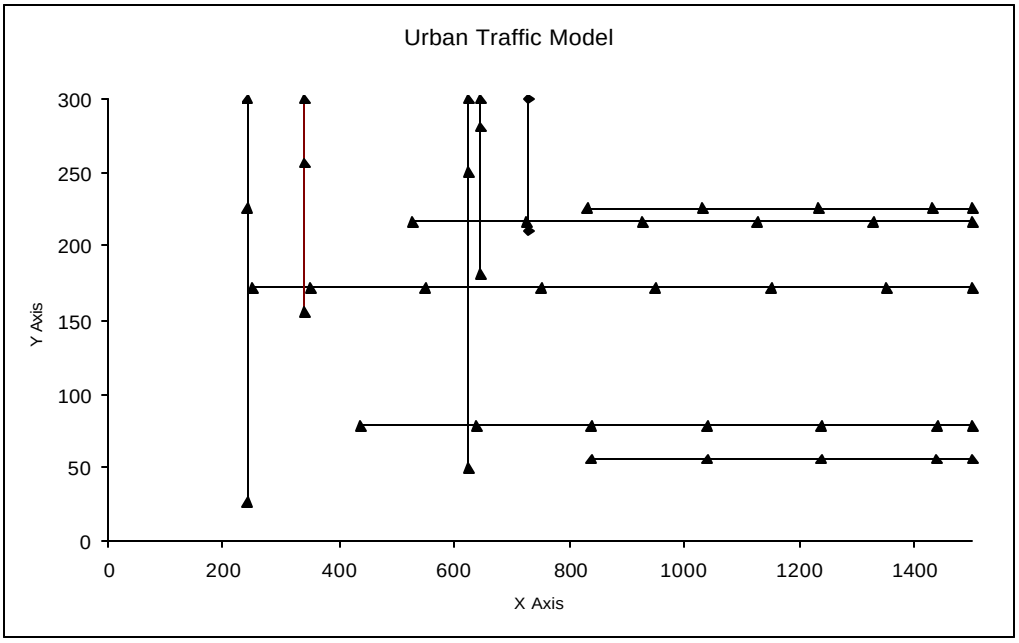


Figure 3.3: *"The urban traffic model"*

3.2.2 Semi-Deterministic Mobility Pattern

Having seen the deterministic mobility pattern, now let us look at the scenario where the mobility pattern followed by the nodes is not so deterministic. Consider, for example, a battalion of battle tanks marching ahead. Here, the path followed by each tank is not specified, but they do move in a general direction (i.e., towards the war front). Even though the individual tanks do not have a specified direction, we can see a general pattern of a column evolving out of it. Such a mobility pattern is termed as a column model. *Figure 3.4* shows a pictorial representation of the column model, where the ants can be treated as tanks moving in the general direction of the goal in a column.

Figure 3.4: *"The column model"*

In this case, the deviation between the direction vectors of two positions can range from **(-90^o to + 90^o)** depending on the width of the column as shown by the deviation ϕ in *figure 3.5*. This deviation also signifies the amount of non-determinism (i.e., more the average deviation, the more non-deterministic or un-predictable is the mobility pattern). That is, with the increase in column width, the maximum possible deviation also increases. The assumption made here is that, if nodes are following a column model, the nodes will pursue a general direction and not just wander 180^o degrees in the opposite direction of the destination. In our view, by varying the deviation (i.e., ϕ in the semi-deterministic model) we can generate various types of mobility patterns for example the pattern followed by the tanks in the battlefield.

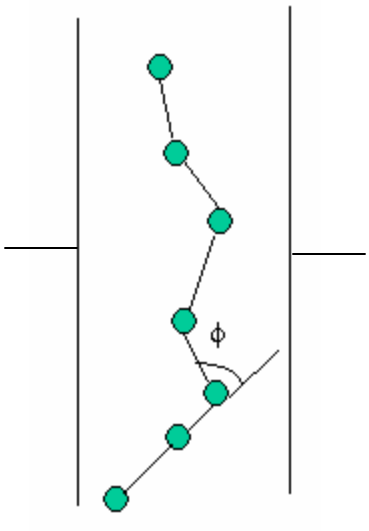


Figure 3.5: "A *mobile node following column model"*

3.2.3 Random Mobility Pattern

Now consider the random motion as in *figure 3.6*. This motion is totally stateless, that is the future movement here is completely independent of the past movement and hence there are no bounds imposed on the max deviation which the nodes can take up for their next movement. And this randomness in choosing the next direction vector renders this type of motion completely unpredictable.

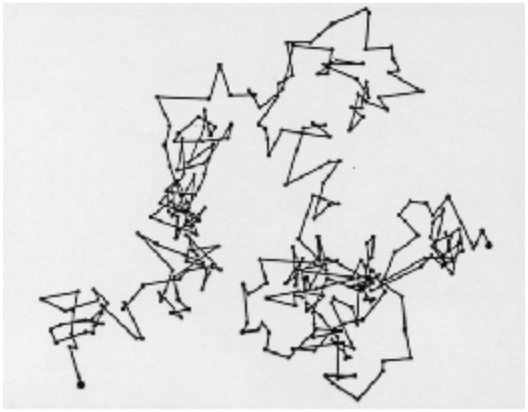


Figure 3.6: *"A node in random motion"*

After having seen how the classification of various mobility patterns is done, in the next section we take a look at how this mobility characterization can be used for node location prediction to enhance the efficiency of routing algorithms.

3.3 Location Prediction

In addition to the simple routing algorithms for ad hoc networks mentioned in the literature such as DSDV, DSR, AODV, TORA etc, which have tried to find a way to perform routing in the mobile environment, there have also been some routing algorithms in literature as in [9, 13, 16, 17, 18, 20, 24, 25, 26, 27], which pay special consideration to the problems arising due to mobility either by using the location information or the neighborhood information. As can be seen in their comparative study, the performance of these mobility centric routing algorithms have definitely been better than the simple routing algorithms because they take into account issues such as the time taken for propagating the topology changes, intelligent handling of route failures, scalability, etc. But none of these algorithms so far have looked into what the extra piece of information about mobility patterns followed by the nodes will have on routing algorithms. Thus, researching the impact of various types of mobility patterns (i.e.. random, semi-deterministic, or deterministic patterns) on transience of the routes, on the reduction of

routing overhead, ease in guaranteeing QoS or whether the extra burden of mobility management has any effect on the efficiency of the underlying routing algorithm at all on the routing algorithm.

For example, consider *figure 3.7*, which shows a sample ad hoc network. Where nodes A and C are relative stable but node B is moving towards node A at a very slow pace where as node D is moving away from nodes A and C at a very fast pace. In this scenario suppose node A gets a route request to node C. Thus a route to node C, which is 2 hops away from node A, could have been either through node B or node D, the task is just to choose any one of the possible routes, either is fine. But if the routing algorithm chooses node D as the intermediate hop on the route from node A to node C, the life of this route is going to be very short, as very soon node D will be out of reach of node A as well as node C and node A very soon will again have to initiate the process of route discovery. But if the routing algorithm chooses node B as its intermediate node on the route to node C, this route is going to have a longer lifetime. Thus in this case where we want the routes to remain stable for a longer time, it would be better if we intelligently choose our intermediate nodes for our routes. And here comes the location prediction, which is possible now because of the availability of the information about the mobility pattern being followed by nodes.

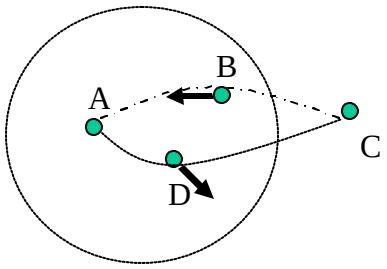


Figure 3.7: *"A sample ad-hoc network"*

Now suppose that the routing algorithm is made aware of the underlying mobility pattern being followed by nodes B and D. The routing algorithm will now be in a position to predict the possible future locations of nodes B and C with considerable amount of accuracy. And when the routing algorithm is faced with the option of choosing between node B and node D as the intermediate route towards node C, it can now figure out of its own that as node D is moving away from the transmission range of both node A and node C, it would be a good idea to choose node B as its intermediate node towards node C.

On the same lines, we foresee many advantages of approximating the motion of mobile nodes to a mobility pattern, namely, we can pre-compute routes to a destination before the current link is snapped, we can attempt to guarantee QoS, as now we can also predict

the state of a link. Thus with the knowledge of mobility pattern, we will be in a position to save a lot of overhead that the routing algorithm could have incurred otherwise.

3.4 History

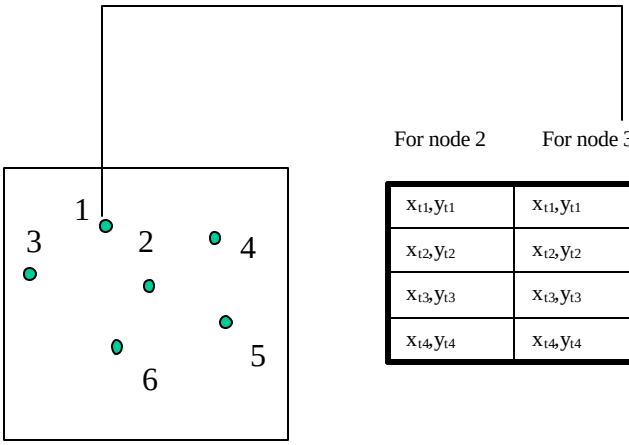
Until now, we discussed how mobility affects the efficiency of the routing algorithms in ad hoc networks and characterized mobility into different patterns. We also discussed how this information could further be used in designing efficient routing algorithms with location prediction of mobile nodes. And in doing so, we assumed that the routing algorithm knows about the underlying mobility pattern being followed by the nodes, which is not the case as the routing algorithm doesn't have any idea about the underlying mobility pattern followed by the nodes. Thus we have to figure a way by which the routing algorithm can learn about the underlying mobility pattern and then use this information accordingly. In this section, we explain as to how the routing algorithm learns about mobility and how it characterizes the mobility into various patterns.

This extra information about mobility, that is, the mobility pattern being followed by the nodes that the routing algorithm we assumed to be having, does not come free. The cost associated with it is coined as HISTORY. To explain history, consider *figure 3.8*, where the nodes in the ad hoc network are placed as shown in the first snapshot at time = t_1, and after certain time and a number of moves, assume the nodes are placed as shown in the second snapshot at time = $t_1 + k?t$, where k is 4 (some integer value). Each node here is associated with a tuple, that is, (x_t, y_t), which is the location co-ordinate of the node at time t.

First Snapshots at t_1 Fifth Snapshot at t_5

Figure 3.8: *"A sample ad-hoc network depicting the different snapshots of the location of the nodes taken time instant* t_1 *and* t_5*"*

Now consider node 1, assuming that it was at position (x_{t1}, y_{t1}) at instant t_1 and then it moved through (x_{t2},$y_{t,2}$) at t_2, (x_{t3},y_{t3}) at t_3, (x_{t4},y_{t4}) at t_4 before arriving at (x_{t5},y_{t5}) at t_5. And if this information about the past locations of node 1 is propagated to every other node in the network either being sent as special updates or by piggybacking this information onto regular network layer updates, then each node would have a history about the past locations of node 1 that would look something similar to as shown in *figure 3.9*.



For node 2	For node 3	For node 4	For node 5	For node 6
x_{t1},y_{t1}	x_{t1},y_{t1}	x_{t1},y_{t1}	x_{t1},y_{t1}	x_{t1},y_{t1}
x_{t2},y_{t2}	x_{t2},y_{t2}	x_{t2},y_{t2}	x_{t2},y_{t2}	x_{t2},y_{t2}
x_{t3},y_{t3}	x_{t3},y_{t3}	x_{t3},y_{t3}	x_{t3},y_{t3}	x_{t3},y_{t3}
x_{t4},y_{t4}	x_{t4},y_{t4}	x_{t4},y_{t4}	x_{t4},y_{t4}	x_{t4},y_{t4}

Fifth Snapshot at t_5

Figure 3.9: *"History Vector maintained at node 1 at the end of 5th time instant"*

Thus history can now be defined as a collection of past visited locations of the mobile nodes consisting of two coordinates (x_t, y_t). And a collection of these history entities make up a history vector that consists of a certain number of tuples, which in *figure 3.9* is 4.

Having explained the concept of history and history vector, now let us look at how this additional information can be used. Consider node 1 in *figure 3.8* again, if it has to discover a route to node 5, it can now use the location information available locally to it from the history vector of the intermediate nodes, first to arrive at the general mobility pattern being followed by the nodes and then use this information to predict possible future locations of these nodes and then find out the best possible options for the route. To visualize this, consider a simple scenario of a node being recorded at (1_1,0_1), (2_2,0_2), (3_3,0_3) and (4_4,0_4). By a simple extrapolation of the history, we can conclude that the node is moving in a linear fashion and that the node most probably will be at position (5_5,0_5) at time t = 5.

3.4.1 Optimal Size of History Vector

One important aspect that is to be considered about the history being kept at each node is, what should be the optimal amount of history to be kept at each node so that any given node can predict the position of the destination node with considerable amount of accuracy. This is so because the accuracy of prediction is directly proportional to the amount of history, but we cannot just add on more history to achieve a greater accuracy. This point is explained more clearly in the next chapter in section 4.2.3.2 where we take a look at our proposed algorithm.

3.5 Chapter Summary

In this chapter, first we explained the need for mobility characterization, which led us to the classification of mobility to various mobility patterns. Next we explained how the information about the mobility patterns could be helpful in designing an efficient routing in ad hoc networks. One important outcome of this chapter was history, which is a collection of past visited locations of a node being collected at other nodes, which help the routing algorithm to identify the underlying mobility pattern.

Chapter 4. Mobility Pattern Adaptive Routing Protocol

In the previous chapter, we discussed the importance and impact of mobility on the efficiency of a routing algorithm. In this chapter, we propose the new routing protocol called Mobility Pattern Aware Routing Protocol that is based DSDV. In doing so we first explain the shortcomings that the DSDV faces and how our algorithm overcomes the shortcomings in the original DSDV algorithm. Toward the end of the chapter, we provide a detailed discussion about the changes that can be done to the existing algorithm so as to make it QoS aware.

4.1 Review of DSDV

4.1.1 Protocol Description

DSDV [1] is a proactive routing protocol that applies distance vector routing in an ad-hoc network. In DSDV, packets are transmitted between the nodes of the network using routing tables stored at each node. Here the routing table lists all available destinations along with the number of hops to reach them. Each routing table entry in DSDV is tagged with a sequence number that is originated by the destination node. To maintain consistency of routing tables and to gain the updated view of the topology, each node periodically transmits updates, when significant new information is available. These updates indicate which nodes are accessible from the nodes sending the update along with the number of hops necessary to reach them.

4.1.1.1 Routing Table Entry Structure

Each row in the routing table contains its new sequence number and the following information for each new route:

- The destination address
- The number of hops required to reach the destination
- Address of the next hop node to reach a particular destination
- The sequence number of the information received regarding that destination, as originally stamped by the destination

In addition to the above mentioned fields the routing table also contains some other fields as “install time” and “stable data”, but these fields are not of much relevance to our discussion here.

For a pictorial explanation, consider $node_4$ in *figure 4.1* which shows an ad hoc network. *Table 4.1* shows a possible routing table maintained at $node_4$. Suppose that address of each node, i is represented as $node_i$ and the sequence number associated with each node be S_i.

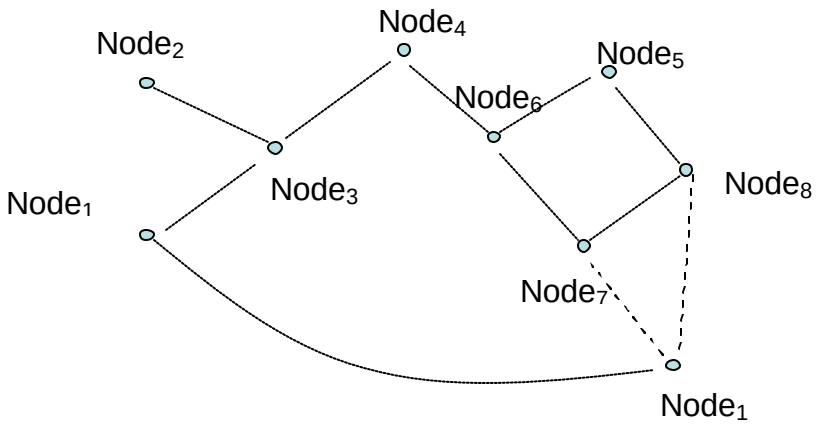


Figure 4.1: *"Movement in Ad Hoc Network"*

And now suppose $node_1$ moves away from $node_2$ to the vicinity of $node_7$ and $node_8$. This movement then affects the routes from $node_4$ and there is a corresponding change in its routing table entries. This change in routing table is shown in *table 4.2*. And this information is to be updated to its neighbors as well.

Table 4.1: *"Sample routing table entry at node4 before node1 moves from its position"*

Destination	Next hop	Metric	Sequence #
$Node_1$	$Node_3$	2	S_1
$Node_2$	$Node_3$	1	S_2
$Node_3$	$Node_3$	2	S_3
$Node_4$	$Node_4$	0	S_4
$Node_5$	$Node_6$	2	S_5
$Node_6$	$Node_6$	1	S_6
$Node_7$	$Node_6$	2	S_7
$Node_8$	$Node_6$	3	S_8

Table 4.2: *"Sample routing table entry at node4 after node1 moves from its position"*

Destination	Next hop	Metric	Sequence #
$Node_1$	$Node_6$	3	S_1
$Node_2$	$Node_3$	1	S_2
$Node_3$	$Node_3$	2	S_3
$Node_4$	$Node_4$	0	S_4
$Node_5$	$Node_6$	2	S_5
$Node_6$	$Node_6$	1	S_6

$Node_7$	$Node_6$	2	S_7
$Node_8$	$Node_6$	3	S_8

The routing updates sent in DSDV carry either the full routing table or the partial routing table, depending on whether the update is a full or partial. Thus, the header of the packet contains the hardware address and (if appropriate) the network address of the mobile computer transmitting them. The routing tables also include a sequence number created by the transmitter. Routes with more recent sequence numbers are always preferred as the basis for forwarding decisions, but they are not necessarily advertised. As route tables are propagated, the sequence number is sent to all mobile nodes, which may each decide to maintain a routing entry for that originating mobile node.

4.1.1.2 Route Advertisements

The DSDV protocol requires each mobile node to advertise, to each of its current neighbors, its own routing table (for example, by broadcasting its entries). The entries in the routing table may change fairly dynamically over time, so the advertisements must be made often enough to ensure the every mobile node can almost always locate every other node in the collection.

4.1.1.3 Response to Topology Changes

Topology changes which are brought in by the movement of nodes need to be updated as soon as possible. These changes are either sent out as part of the periodic updates or if the changes are significant they are sent out as triggered updates. For example if a neighboring node is found to be dead or moved out, then all the routes for which the dead node was the next hop are designated as broken links and the metric is changed to *infinity* that is the highest metric. And since this is a substantial route change, such modified routes are immediately broadcasted.

4.1.1.4 Route Selection Criteria

When a node receives a new routing (either in a periodic or triggered) update, that information is compared to the information already available in its routing table. Any route with a more recent sequence number is used; routes with older sequence numbers are discarded. A route with sequence number equal to an existing route is chosen if it has a better metric, and the existing route is discarded. The metrics for routes chosen from the newly received broadcast information are each incremented by one hop. Newly recorded routes are scheduled for immediate advertisement to the current mobile node's neighbors.

4.1.2 Critique of DSDV

In this section, we discuss some of the shortcomings of the DSDV algorithm. A few of the problems have been taken care of in the recent versions of the DSDV algorithm, but there are some issues that are yet to be addressed.

- DSDV, like other table-driven protocols, suffers from the drawback of high overhead. This high overhead is mainly due to frequent routing updates, which DSDV makes in order to keep updated topology information. With scarce resource such as bandwidth and battery power, these transmission overheads lead to considerable performance degradation.

- In cases of high mobility, it is possible that DSDV might not to converge.

- Nodes have no knowledge of the spatial location of other nodes in the network as the topology information is maintained in a routing table in the form of number of hops required for any node to reach any other node. There are two disadvantages of not having the location information available. First, the routing algorithm is not location aware, which can greatly reduce the extra overhead in case of route selection or route repair.

- Second, the non-availability of location information lends the routing algorithm clueless, as to what mobility pattern is being followed by the underlying nodes. As we argued in chapter 3, that information about mobility pattern will further help in adapting the routing algorithm to different kinds of mobility patterns and hence resulting in an efficient routing algorithm rather than treating every motion as motion as random.

Having discussed how DSDV works, and its shortcomings, we now see how our routing algorithm addresses these issues.

4.2 Mobility Pattern Aware Routing Protocol

In the process of describing the new routing algorithm, we first make DSDV *location aware* and in the next step, we extend it to be mobility pattern adaptive. At the end, we also discuss how this can further be extended to guarantee QoS requirements.

4.2.1 Location Aware DSDV

As we have mentioned earlier, the knowledge of location is important for successful routing of packets in an ad hoc network. And in order to make DSDV location aware, we need some means by which the nodes can determine their present physical location. Thus in making DSDV location aware we assume that each node is equipped with a GPS,

which constantly updates the mobile host with its latest location information, with a fair degree of accuracy.

4.2.1.1 New Data Structures

In the modified DSDV algorithm, we have an extra piece of information that is the location information, which needs to be handled. Thus, in addition to the regular routing table which each node maintains locally, it also maintains a new data structure called the *history table* as shown in *figure 4.2*, where ($x_{t,i}$, $y_{t,i}$) denotes the location co-ordinates of $node_i$ at time t. This table is maintained at each node for every other destination node in the network. And this table is populated progressively, as the nodes receive routing updates from neighboring nodes and, accordingly, the newer information is stored while the older information gets flushed out.

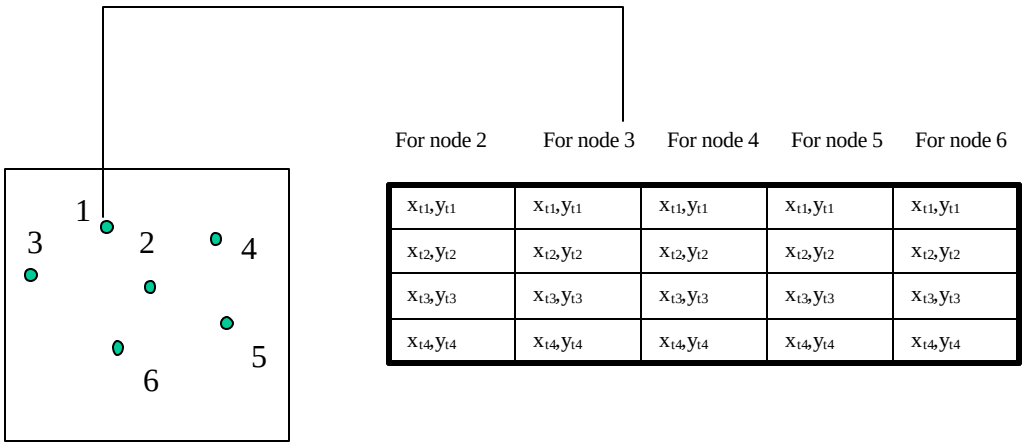


Figure 4.2: *"History table maintained at $node_1$ for every other node in the network"*

The GPS attached to the mobile nodes, only provides the location information of the node to which it is attached. But in order to obtain location information of other nodes in the network, the location information needs to be shared among the nodes in the network. One way to achieve this is by piggy backing the location information to the regular routing updates. The new data structure of the modified routing updates being sent in location aware DSDV is show in *table 4.3*, where *Last X* and *Last Y* are the location co-ordinates of $node_1$.

Table 4.3: *"Sample routing update"*

Destination Addr.	Metric (no. of hops)	Addr. of next hop	Sequence no.	Last X	Last Y
$Node_1$	2	$Node_2$	S_1	x1	y1

4.2.1.2 Route Selection Criteria

To explain the working of *location aware* DSDV, consider *figure 4.3*, which depicts a snapshot of a sample ad hoc network, where a fair amount of time has passed after the network came into existence. At this stage, the routing tables have settled down after the initial surge in broadcast of routing tables, as every node tries to acquire the topology information. By this time the nodes would have also gathered location information in their history tables. Now suppose *node 1* has some data to be sent to *node 8*, which in this case is the destination node. Under the normal DSDV, *node 1* would look in its routing table for the record that lists *node 8* as its destination node, pick up the intermediate hop node, if any, and transmit to it. However, if it does not have an entry for *node 8*, it will respond by saying the destination is unreachable.

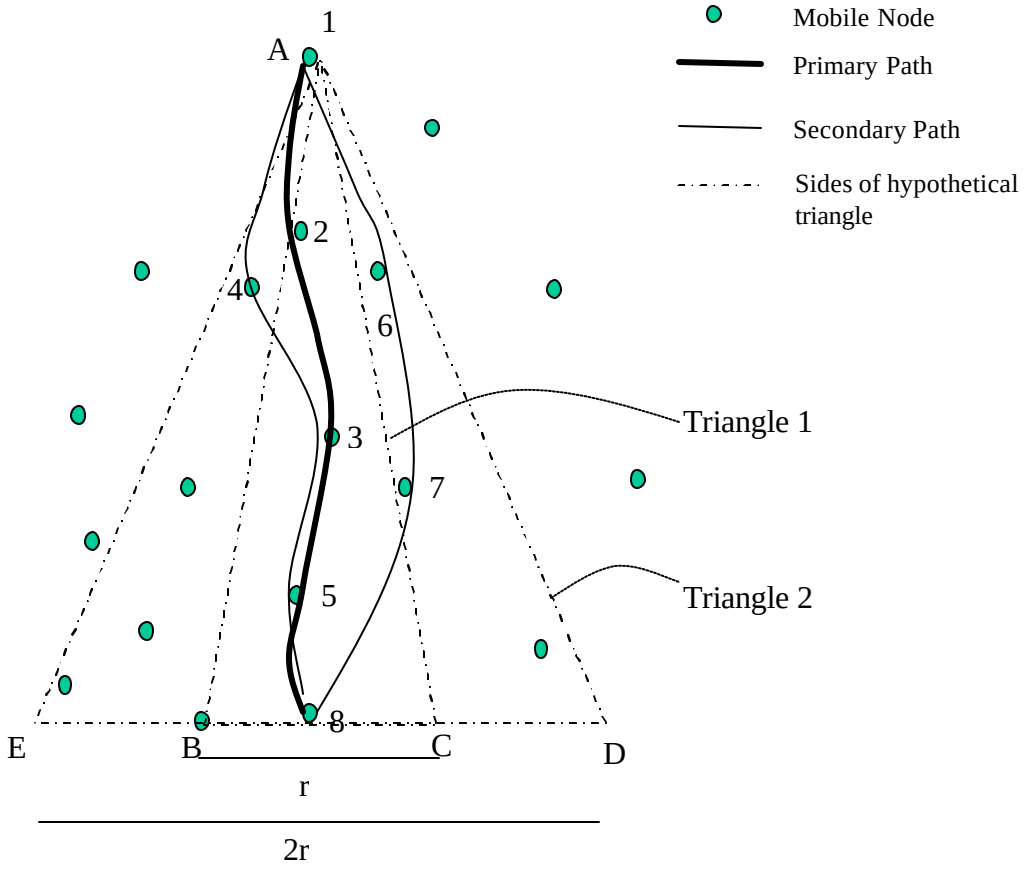


Figure 4.3: *"A sample ad hoc network"*

Now we see how the same node will respond, when it has some data to be sent to *node 8* in modified DSDV algorithm.

Step 1: First, the source node (i.e., *node 1*) looks into its routing table to see if it has an entry for the destination node, which in this case is *node 8*. If it has an entry, it then follows the steps of normal DSDV algorithm.

Step 2: Else if source node does not have an entry for the destination node, the source node tries to find intermediate nodes in the general direction of the destination through which it can route the data packets. To find intermediate nodes in the general direction of the destination, the source node (i.e., *node 1*) hypothetically constructs an isosceles triangle, which is shown as ?*ABC* in *figure 4.3*, with the destination node being the centre of the base *BC*. The length of the base is equal to the transmission range of the mobile nodes. Then, the source node forwards the route request to a neighboring node that is only one hop away and located inside the hypothetical triangle. In case it finds multiple nodes in its neighborhood to which it can forward the route request, it forwards it to the nearest one since such a link would be more stable compared to one farther away. In this case it would be *node 2*. Now, *node 2* will forward it to *node 3*, and then to *node 5*, which eventually will find the destination node, *node 8* in its neighborhood list.

Step 3: In case the source, or any other intermediate node participating in the route discovery process, does not find a node in its neighborhood that is in the general direction of the destination, the source node will expand its search area by increasing the size of the base of the isosceles triangle. Suppose, *node 2* was not at its present location in *figure 4.3* then *node 1* in its effort to find a route to the destination would not find any node in its neighborhood that is in the general direction of the destination node. It then expands its search area by looking into intermediate nodes (i.e., ? *AED*), which has a base twice the transmission range of the nodes. Now, it finds two nodes, that is, *node 4* and *node 6*, to which it can forward the route request. These nodes again employ step 2 in finding a route to the destination and they can eventually figure out the route through either *node 3* or *node 7*, respectively.

4.2.1.3 Frequency of Periodic Updates

DSDV, in its approach towards being proactive, heavily relies on periodic updates, to gain information about the topology changes occurring in its neighborhood. This leads to higher frequency of periodic updates, which in turn increases the routing overhead of the network, as periodic updates are mostly a full dump of the routing table. In the location aware DSDV algorithm, however, we suggest suppressing the periodic updates for a longer period of time and encourage triggered updates, which would be much less compared to the number of periodic updates sent by all the nodes in the network. In the modified DSDV a triggered update can result in two scenarios, first when a node discovers that a link to one of it neighbors is broken, in which case it has the responsibility to inform its neighbors of the broken link. Second, if it feels the distance it traveled since the last update it sent out, is long enough to break any link, then also it sends out a triggered update. This is feasible because with GPS attached to the nodes, it can find out its location at any instant and hence compute its relative distance to the location from where it last sent out its location update to its neighbors.

4.2.2 Implications of the new Technique

1) The new routing algorithm, given a choice between feasible, multiple routes, tends to find routes that are more stable. In other words, when a source node has a route request to a destination node, and the route selection process is faced with a situation where it has multiple nodes in its neighborhood to which it can forward the request, it chooses to forward the route request to its nearest node. Because the rate at which a nearer node will move away from the source node will always be less than the rate at which the farther node will move away. As a result by choosing a nearer node as its next hop node for the destination node, we are making the routes more stable.

2) The route discovered by following the general direction of the destination will also be the shortest as the algorithm always tends to find a path that nearly converges to a straight line. Forming a straight line aims to provide the shortest distance between the source and the destination.

3) In this, we looked at the scenario where we had to initiate a new route to the destination, but this algorithm also works well for the scenario where there is a route failure of an established route. With no packet drop, and on the fly discovery of an alternative route to the destination, this saves a lot of overhead that DSDV normally incurs.

4) Reduced transmission over head, as now the periodic updates have been nearly ruled out.

Thus, in this section, we saw how the location information about the destination node being available locally at the source node and the intermediate nodes helps in overcoming some of the shortcomings of DSDV. In the modified DSDV algorithm, however, we made an assumption that when a source node draws the hypothetical triangle towards the destination, the location information of the destination available at the source node is very recent. But this is quite an unrealistic assumption, considering that the nodes are mobile and the information available at the source node might be old. In such situations, our algorithm might draw the hypothetical triangle that may not be in the general direction of the destination, and hence nullifying all the proposed enhancements. In the next section where we propose a mobility pattern aware routing scheme, we see how the information about the mobility pattern helps us in dealing with such a scenario by means of location prediction. We also look into achieving a few other optimizations, as in the reduction of the number of updates.

4.2.3 Incorporating information provided by Mobility Pattern

In the previous section, we modified DSDV to be location aware. In order to make DSDV location aware, location information of the nodes were exchanged in addition to the regular routing updates. As a result of the exchange of location information, each node built its own history table for every other node in the network. But the use of these past locations in location aware DSDV has not been much other than predicting the possible future location of mobile nodes.

In this section we see how the information about the past locations of nodes, stored in the history table can be further used to gain knowledge about the underlying mobility pattern. And once the location aware DSDV figures out the underlying mobility pattern, how can this information further be used to effectively route the messages in an ad hoc network.

But before we explain the new routing algorithm, let us see how various types of mobility patterns that we discussed in *Chapter 3* can be generated.

4.2.3.1 Generating Different Mobility Scenarios

As our study is based on simulations, one of the first tasks is, to be able to generate the three types of mobility patterns. These mobility patterns are fed to the mobile nodes in our simulation. But before we proceed any further, we make two assumptions, first, the mobility pattern followed by the nodes in any particular scenario is homogenous, which means, the entire set of nodes follow any one particular type of mobility pattern. Second, the speed of the nodes remains constant through out the simulation. It is only the direction of the nodes which change, and hence resulting in various types of mobility patterns.

4.2.3.1.1 Generating Semi-Deterministic Mobility Model

For generating the semi-deterministic mobility model (i.e., the column model) as discussed in *section 3.2.1.2*, the effect of non-determinism (i.e., the deviation from the original direction) is considerable. For our simulations, we designed two types of column movement, depending on the maximum allowable deviation from the original direction. Thus, if the maximum deviation was less than 15^o, then the mobility pattern was assumed to follow a strict column model. If, however, the maximum deviation was greater than 15^o and less the 30^o, it was assumed to follow a less strict column model. The pseudo code for generating such a mobile pattern is given in *figure 4.4*, where procedure *move_column* is called by the scenario generator module of NS2 [75] at regular time intervals (i.e. ***t*)** .

```
loop
   pos proc move_column (node)

    /* move_column is a procedure that is called in a loop
```

```
every time the next node position is to be determined
and returns the next position                                   */
    pos     old_position = get_current_position (node);
   speed   old_speed = get_current_speed (node);
   direction old_direction = get_current_direction (node);
   direction new_direction = get_new_direction ();
     /*new direction is a randomly selected value, but it should not
        exceed the range of +/- 15° as the semi-deterministic model
        puts a limit on the maximum deviation.    */
   if the deviation is positive from the reference point of final destination
        {
             pos new_position_x = old_x + v * t * cos(F) ;
             pos new_position_y = old_y + v * t * sin(F) ;

        }
  if the deviation is negative from the reference point of final destination
        {
             pos new_position_x = old_x - v * t * cos(F) ;
             pos new_position_y = old_y - v * t * sin(F) ;
        }

     return (new_position) ;
end main loop
```

Figure 4.4: *"Pseudo code for generating column model"*

To explain the mathematical part of the pseudo code consider *figure 4.5*, which shows a node at location (***old_x,old_y***), moving horizontally and now undergoes a deviation of ***F*** (i.e. $+15^{o}$). Assuming the node continues to move at the same speed (i.e. ***v***) with which it was moving, the new x and y co-ordinates (i.e. ***new_ x, new _y***) are given by ***new_x = old_x + v * t * cos(F)*** and ***new_y = old_y + v * t * sin(F).*** A possible column model is shown in *figure 4.6*.

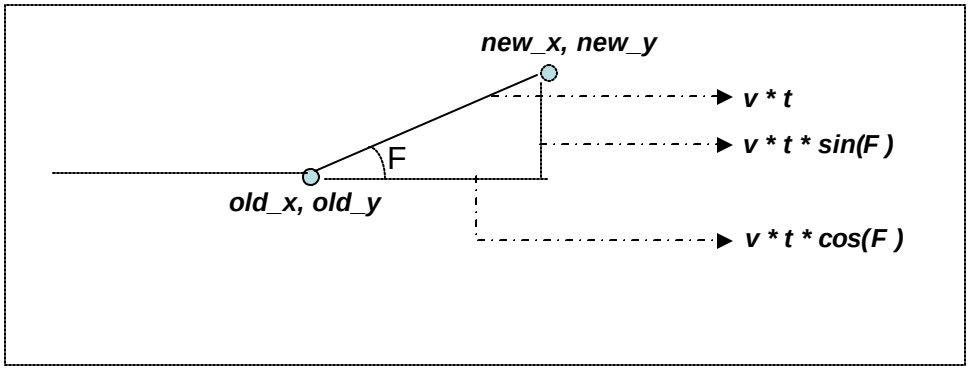


Figure 4.5: *"Co-ordinates Computation Model"*

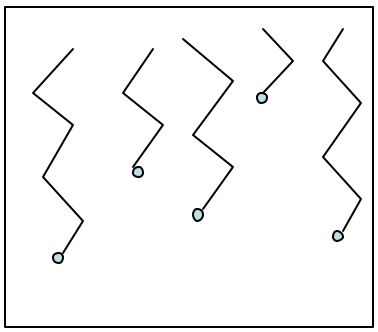


Figure 4.6: *"Possible Semi-deterministic Model"*

4.2.3.1.2 Generating Random Mobility Model

For generating random mobility pattern, there is no fixed direction of the nodes. Here the nodes after every pause pick up a random direction, in a truly random scenario the speed of the nodes would be a random variable, but for our simulations the speed is constant through the duration of simulation. The pseudo code for generating this type of mobility pattern is similar to the pseudo code for semi-deterministic model shown in *figure 4.4*, except that the, ***get_new_direction()*** function here returns a random value ranging from 0^o – 360^o. A possible random model is shown in *4igure 4.7*.

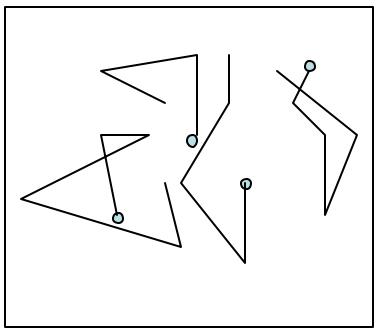


Figure 4.7: *"Possible Random Model"*

4.2.3.1.3 Generating Deterministic Mobility Model

For generating deterministic mobility patterns, the nodes are imparted a particular speed with which they travel but they move only in vertical or horizontal directions. When a node hits the boundary of the layout, it retraces its path with the same speed. To implement this mobility model, we fed the scenario generator module of NS with location

co-ordinates as well as the turns which the node would take, resembling an urban traffic model, where the paths and speeds are predefined. A possible deterministic model is shown in *figure 4.8*.

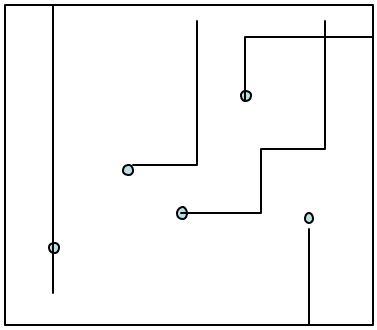


Figure 4.8: *"Possible Deterministic Model"*

4.2.3.2 Making the Routing Algorithm Mobility Pattern Aware

Now that we know how to generate the mobility patterns, which the nodes in the mobile ad hoc network will follow, we look at how the location aware routing algorithm can be modified to make it mobility pattern aware.

Now suppose, the nodes in an ad hoc network are imparted a particular type of mobility pattern. The routing algorithm would have no knowledge of the underlying mobility pattern being followed by the nodes. Thus, in order to make the routing algorithm mobility pattern aware, it should first be made self learning that is, given any of the three (urban traffic, column or random) mobility models, the routing algorithm should be in a position to identify them. Second, upon identifying the underlying mobility pattern, the routing algorithm should adapt itself accordingly to perform more effective routing.

4.2.3.2.1 Identifying Underlying Mobility Pattern

In section 3.2, where we classified mobility into various patterns. In order to classify mobility, we used $\boldsymbol{F}$, which is the deviation of the new direction of motion from the old direction, as the main parameter to categorize mobility. Now when the routing algorithm has to identify the mobility pattern being followed by the nodes, it also uses the deviation $\boldsymbol{F}$, as its basis for classification. To determine this deviation $\boldsymbol{F}$, the history of past location co-ordinates of nodes stored in history tables is used. The pseudo code for determining mobility pattern is shown in *figure 4.9*, where $\boldsymbol{i}$, is a counter variable whose value ranges from zero to the number of rows in history table.

```
pattern_type proc learn_pattern (history_table, node)
/* learn_pattern is a procedure that is called up
   with a node and its corresponding history as its
   argument. It decides on the mobility pattern and
   returns it to the calling procedure                 */
```

```
var       old_direction, new_direction, deviation, average_deviation;
average_deviation = 0; // set to zero initially
for ( each entry in the history table for the node)
{
        old_direction = diff (node.history_table[i], node.history_table[i+1]);
        new_direction = diff (node.history_table[i+1], node.history_table[i+2]);
        deviation = diff (old_direction, new_direction);
        average_deviation = average_deviation + deviation;
} // end for loop
/* at the end of the for loop, we will have the
   sum of all deviations, and then will classify
   our mobility pattern based on the average value of deviation */

average_deviation = average_deviation / total_number_of_iterations_in_for_loop;
pattern = get_type_of_pattern_based_on_max_deviation (max_deviation);
return (pattern);
end
```

Figure 4.9: "*Pseudo code for determining the mobility pattern*"

The logic used for computing the deviation angle is the difference in slopes of two consecutive entries in the history table for any node. For example consider the sample history table entries for node2 stored at node1, $(x_{2,1}, y_{2,1})$, $(x_{2,2}, y_{2,2})$, $(x_{2,3}, y_{2,3})$ and $(x_{2,4}, y_{2,4})$. Then the slope of the line from $(x_{2,1}, y_{2,1})$ and $(x_{2,2}, y_{2,2})$ is given by $(x_{2,1} - x_{2,2}) / (y_{2,1} - y_{2,2})$ and similarly for the line between points $(x_{2,2}, y_{2,2})$ and $(x_{2,3}, y_{2,3})$ is given by $(x_{2,2} - x_{2,3}) / (y_{2,2} - y_{2,3})$. And the difference in their slopes will give us the deviation between the two lines.

Thus, as can be seen from the algorithm, it is the average deviation, which helps in deciding the mobility pattern and in recognizing the mobility patterns proposed in our study.

4.2.3.2.1.1 Number of Rows in History Table

Till now we had assumed that the history table had at least four entries for each node. But the question that has to be addressed here is, as to what should be the optimal size of the history table i.e. how many entries for each node should be kept so that we can satisfactorily conclude about the underlying mobility pattern. Obviously, keeping more history will be beneficial and will lead to a more accurate characterization of the underlying mobility pattern. But for our study, however, as the maximum number of possible mobility patterns under consideration is only three, it suffices to have four history table entries. Because by keeping four location co-ordinates, we can compute two deviation angles and from those two deviations we can find the average deviation allowed, which is sufficient to categorize the mobility being followed into deterministic, semi-deterministic or random.

4.2.3.2.2 Adapting to Mobility Patterns

In the *route selection criteria*, section 4.2.1.2 for location aware DSDV algorithm, we explained how the source node in its attempt to find a route to the destination node tries to find intermediate nodes in the general direction of the destination node. And the general direction of the destination node there was determined by drawing an isosceles triangle, ? ABC, as shown in *figure 4.4*, with the destination node being at the centre of the base of the isosceles triangle. The accuracy of this general direction towards the destination node is dependent on the location of the destination node. In location aware DSDV, the location co-ordinates from the last location update from the destination node was used to determine the general direction of the destination. But as the nodes in an ad hoc network are constantly in motion, the last location update received about the destination node, might not be destination node's current location. Hence to draw an isosceles triangle, which will more accurately represent the general direction of the destination node, we use the mobility pattern information to predict the possible location of the destination node.

On similar lines, location prediction is also used to effectively repair route failures.

But location prediction comes to use only when the underlying mobility pattern is either deterministic or semi-deterministic. In case of random motion, we cannot predict future location of nodes, as a result in that case mobility pattern aware DSDV reverts back to its standard DSDV way of routing messages.

Figure 4.10 shows a simple scenario where location prediction is employed. We accept that this method of location prediction is not error free, but if we are able to predict the future location of the nodes with an error which is less than half of the nodes' transmission range, it suffices our needs. To visualize this, consider *figure 4.10 again*; here the node is moving along the line joining *pos1* to *pos5*. In this case as the path from *pos1* to *pos 4* is a straight line and hence when we predict the next location for the node after *pos4* it is predicted to be at pos6, where as it is at pos5. But still this error is tolerable as the deviation from the predicted location is within the transmission range of our predicted location. The pseudo code for the location prediction is shown in *figure 4.11*.

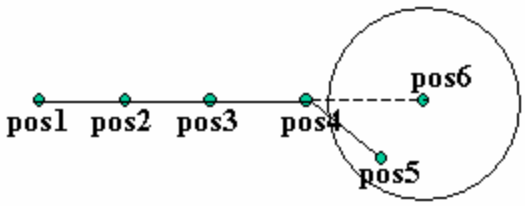


Figure 4.10: "*Location Prediction*"

```
pos proc predict_location ( node, history_table, pattern_type)
        /* the predict_location procedure is called with node, history-table,
           and pattern_type as argument. This procedure predicts a possible
           future location for a node if only the pattern type is not random, in
           which case there is no rational reason to predict future location      */
                if (patter_type == random)
                        return (NULL);
                if (pattern_type == column)
                        predict (old_pos, new_pos, speed, old_deviation);
                        // this procedure takes in the last position, last speed, and the
                        // last deviation, and if the deviation between consecutive
                         // history is random, less the 15° or 30°, or not
                         // diff at all, then this results in a random, column, pattern or
                         // deterministic mobility model, accordingly.
                if (patter_type == pursue)
                …
                …
                return (predicted_pos);
end proc
```

Figure 4.11: *"Pseudo code for predicting next position"*

4.3 Provision for Imposing QoS Routing (future work)

Despite the above-mentioned enhancements that the proposed routing algorithm offers, it is not sufficient for offering QoS guarantee. To impose QoS guarantee on routing over a wireless multi-hop path, it mandates that QoS state information should also be propagated along with the routing updates within the network. Thus, the key to supporting QoS is in making the QoS state information of all the nodes available locally at every other node in the network. Thus now when a node is faced with a route selection decision, it can make its decision based on the locally available QoS state information of the possible intermediate nodes. The QoS requirement for a new path may be a delay constraint, a bandwidth constraint, or both. So, the nodes participating in the route discovery process should now satisfy the additional constraints imposed by the QoS requirements in addition to discovering a route to the destination.

To perform QoS based routing, the routing update packets are now required to carry the location information as well as QoS state information along with network ID and sequence number. This helps the receiving node to have an idea of the latest QoS state of the neighboring node that sent the broadcast and it also records its last position along with time.

With the extra information, the routing table entry for each node now would look like as shown in *table 4.4* where *Last X* is last known x-coordinate of the node, *Last Y* is last

known y-coordinate of the node, *Max Delay* is the maximum delay a packet will experience at the node, *Min Bandwidth* is the minimum available bandwidth at the node.

Table 4.4: "*New routing table entry*"

Node ID	Next Hop	Seq #	Hop count	Last X	Last Y	Max Delay	Min Bandwidth

Now we take a closer look into the algorithm. Given a source node and a destination node there can be a requirement to find a feasible path (P) satisfying either a delay requirement (D) or a bandwidth constraint (B) depending on whether it is a delay-constrained routing or a bandwidth constrained routing. Thus the computed route should satisfy either delay (P) <= D or bandwidth (P) >= B. In *figure 4.12* we present the algorithm for QoS routing. The algorithm starts with a request for QoS guaranteed route request to a source node. The source node first looks in its updated routing table to find out which nodes satisfy the QoS requirements and then routs the packet to that. Else if it doesn't find one then it expands it search area to two hop away nodes and so on as explained in section 4.2.1.2.

```
proc QoS_routing (src; dest; < qos requirements >)
   /* QoS_routing is the procedure called to find out
    a feasible route between the source and the
   destination which satisfies the QoS requirements */

      valid routes := {}; // initially the set of valid routes is null
      cand list := compute cl (src; < qos requirements >); // find out which nodes in the
                                                              neighborhood satisfy the qos
                                                              requirement
      if cand list ! ={} // i.e. if the list has some entries
            then if dest ? cand list // and if the dest is one among them
                  then Directly forward packet to destination;
                  else foreach entry in cand list do
                        find route (c; dest; < qos requirements >)
                  end of do
            end of if
      end of if

      if valid routes = {}// i.e. if the list turns out to be empty
            predict_list = predict_location() // of all of the two hop away nodes
                                                 and see if they are expected to be in
                                                 the neighborhood shortly and
                                                 satisfy the qos requirements
      if predict_list = {}
            Noroute reject connection;
      else Output shortest distance routes from valid route
```

end of if

Figure 4.12: *"Pseudo code for QoS routing"*

4.4 Chapter Summary

In this chapter, we first explained how DSDV routs messages in an ad hoc network. There we highlighted some of the shortcomings of DSDV which we argued could be dealt with if we had location information of the nodes in the network. And hence we proposed the location aware DSDV. Finally we proposed the mobility pattern aware DSDV, which is location aware as well. Last, we also mentioned some insight oh how QoS support could be incorporated into the mobility pattern aware routing algorithm. In the next chapter we present the simulation environment, implementation details and also discuss the obtained results.

Chapter 5. Simulation Methodology and Performance Evaluation

In this chapter we evaluate the performance of the new routing algorithm that is mobility pattern aware DSDV with standard DSDV. Before presenting the results, we explain the simulation environment as well as the methodology used to carry out the tests.

5.1 Simulation Environment

To simulate the new routing algorithm and compare its performance with DSDV, we have used NS2 [75] version 2.1b6a. The primary reason for choosing NS2 was its support of a multi-hop wireless environment. Secondly, as most of the studies cited in the literature here have used NS2 as their simulation environment, their performance results of the standard DSDV, as in [76], forms a basis for verification for our results for standard DSDV as well.

5.1.1 NS2 Simulator

NS2 is a discrete event simulator developed by University of Berkley. It provides support for both wired as well as wireless networks in addition to various other types of networks. It is an object oriented simulator, written in C++, with OTCL/Tk as its scripting language. To conduct experiments on existing protocols under different scenarios such as different routing algorithms or mobility, one can write an OTCL script where in various components such as nodes, links, protocols, traffic etc can be defined and run. However to modify the working of an existing routing algorithm or to implement a new routing algorithm, one has to implement it in C++. NS2 reports each event during the simulation in a trace format, which can later on be analyzed to obtain the desired statistics.

For our simulations, we mostly had to deal with the DSDV module and the mobility scenario generation module of NS2.

5.2 Simulation Methodology

Having previewed the simulation environment, in this section we will put forward our simulation methodology. In an effort to stick to our goals while at the same time, conduct comparisons with the published results as in [76] for the standard DSDV, we chose our network to be a 50 node ad hoc network through out the simulations. We also keep other variable parameters such as the amount of data traffic, size of the data packets and the link capacity constant through out the simulations. Thus the different test scenarios arise

by varying the underlying mobility pattern along with the employed routing algorithm to see how the standard DSDV and the mobility pattern aware DSDV perform in various scenarios.

In the rest of the section we present the topography, mobility model, traffic model, protocol implementation and the metrics used for performance evaluation.

5.2.1 Simulation Topography

The network simulations carried out for our study are based in 1500 x 300 meter flat grid topography. The reason for selecting a rectangular topography is that, it results in a larger number of hop count between source and destination than in a square topography as argued in [76]. Since an evaluation of a routing protocol for ad hoc network must test its forwarding ability along with its ability to effectively deal with link breakage and route repair scenarios. Thus rectangular topography seemed to a right choice for our simulations which provides a more rigorous environment for performance comparison.

5.2.2 Traffic Model

For all our simulations, we have used the same traffic model that is of 20 CBR sources, transmitting at 4 packets per second with the size of each data packet being fixed at 64 bytes in accordance with [76], which argues that larger sized data packets lead to higher packet drops due to queuing delay. And as the ad hoc network under consideration is a 50 node network, we made sure that no node acts as both source and sink for data packets. So each 20 CBR sources would have 20 different sinks for the CBR packets. The advantage of having a uniform traffic model for all the test case scenarios is that, with the total number of data packets sent out being constant, the total number of data packets successfully received at the destination will reflect a measure of the over all packet delivery ratio which in turn is a measure of how well the routing algorithm had performed under various scenarios with different mobility patterns and different routing algorithms.

5.2.3 Mobility Pattern and Movement Scenarios

We have considered three types of mobility patterns for our simulations that is random, semi-deterministic and the deterministic mobility models as explained in *section 3.2*. Here the mobility pattern refers to the movement pattern of the nodes in the ad hoc network as a whole, where as the movement scenario is generated for individual nodes and refers to the individualistic node movements, which is in accordance to the mobility pattern it is following.

The random mobility pattern's movement scenario for our simulations were generated using the *setdes*t utility of NS2 with the number of mobile nodes, the grid area, the max

speed and pause time as its parameters. *Setdest*, uses the random waypoint model to generate random mobility. At the start of simulation, all the 50 nodes are laid out randomly on the rectangular topography and then start moving towards a random destination at some random speed, which is a value between 0 and the maximum specified speed. After reaching the destination, the node pauses for a certain time as specified by the pause time parameter before moving towards the next randomly chosen destination. The speed however doesn't change when the nodes are mid way to destination.

In random waypoint model even though the max speed is specified, it has been observed that the average speed of the nodes in the network is less than half of the maximum specified speed and this goes on decaying with the duration of simulation. In our study where we are trying to measure the performance of the new routing algorithm with respect to different mobility pattern, the average speed of the nodes in the network forms an important aspect and performing simulations with ever decaying average speed is not a good measure of performance for the routing algorithm, as we cannot claim the results to be valid at some particular speed. So we modified the existing *setdest* utility so as to restrict the speeds chosen after every pause between some maximum and minimum value which in this is same as the maximum speed. And we call this type of mobility as random mobility with no trailing effect. Thus for our simulations with random mobility we use two types of movement scenarios that is, one with trailing and one without trailing.

For random mobility we chose to simulate at two speeds i.e. 20 m/s and 10 m/s. With trailing the average speed was be less than 10 m/s and 5 m/s respectively, but with no trailing the average speeds remained constant at 20 m/s and 10 m/s respectively. We also simulated with seven different pause times at the above given speeds that is, 0s, 30s, 60s, 120s, 300s, 600s and 900s. Each movement simulation lasted for a period of 900s. Since the protocol is sensitive to movement pattern and relative placement of nodes on the topography, we generated 10 different movement scenarios for each pause time and the average of ten simulations was used for collecting data points for presenting the results.

For simulations with deterministic mobility pattern, the movement scenarios here also were generated using modified *setdest* utility. Here the nodes after each pause were allowed to pick a destination only in a specific direction that is, either horizontal or vertical with respect to the grid and this direction was specified as a parameter to *setdest*. The speed with which the nodes moved towards the destination was also constant through out the simulation and was specified as a parameter to the *setdest* while generating the movement scenarios for the nodes. Thus the nodes seemed to be moving together in a frame like structure. For deterministic mobility pattern also we chose to simulate at two speeds i.e. 20 m/s and 10 m/s and with seven different pause times as mentioned above with the duration of each simulation being 900 seconds. And for each simulation, the initial layout of the nodes on the topography was in a random fashion, thus giving us spatial diversity for each simulation.

In our study involving semi-deterministic mobility the movement scenarios were generated with a fixed deviation of 15^{o} as explained in *section 3.2.2*. And the rest of the details remain same as that of generating deterministic movement scenarios.

5.2.4 Protocol Implementation Decisions

5.2.4.1 Standard DSDV

The algorithm used for running simulations with standard DSDV is the one provided in NS2. Its workings are same as described in section 4.1, with periodic updates being sent out by every node at an interval of 3 seconds and the number of periodic updates missed before a link is declared dead is 3. This algorithm employs both periodic as well as triggered updates. Triggered updates are sent out, either when a node discovers a broken link or when there are sufficient unreported changes to the routing table which in this case happens when a routing update with a new sequence number is received. As the paper suggests, this implementation of DSDV makes no special effort to repair a broken link, it only reports the link breakage to its neighbors by a triggered update.

5.2.4.2 Mobility Pattern Aware DSDV

In order to implement the new mobility pattern aware DSDV, we modified the existing implementation of standard DSDV in NS2 to incorporate the changes as explained in *section 4.2*.

The new routing algorithm employs both triggered as well as periodic updates. But the periodic updates have been suppressed by increasing their interval to every 15 seconds and 12 seconds in case of deterministic and semi-deterministic mobility pattern respectively. The reason being once the location information of all nodes have been exchanged and once all the nodes get a snapshot of the location of other nodes in the network, there is no need to keep polling the neighborhood to find out who its current neighbors are. But there can be some abrupt changes, which might affect the connectivity graph of the network, in such cases a triggered update is sent out. The way it is implemented here is, each node maintains a timer which is reset every time an update is sent out and if the node finds out that it has moved more than half of the transmission range, which in this case is roughly 100 meters, from the last time it sent out an update, a triggered update. As the frequency of the periodic has been decreased considerably, the number of periodic updates missed before a link is declared dead has been brought down to 1 that is at least one update in every 15 seconds or 12 seconds accordingly. Also in the normal DSDV, in case of a link breakage, there is no provision for a link repair. But in the modified DSDV, we perform *link repair* by location prediction as explained in section 4.2. An implication of the link repair is, that upon hearing a link failure there is no immediate triggered update to inform the neighbors about the link failure, but first *link repair* is employed and incase this also fails then a triggered update is sent out.

One of the important features of the new routing algorithm is to be able to figure out the underlying mobility pattern and accordingly adapt itself to the mobility pattern. As the routing algorithm here is not provided with the information about the underlying mobility pattern, the routing algorithm first has to learn about the mobility pattern being followed. It learns about that with the help of location history which each node here keeps about every other node. Thus when the mobility pattern is random, it functions as standard DSDV and when the mobility pattern is deterministic or semi-deterministic, it switches to modified DSDV.

A detailed explanation of the algorithm can be found in *section 4.2*.

5.2.5 Performance Metrics

For evaluating the new routing algorithm against the standard DSDV, we chose three parameters which are commonly used in the literature to evaluate the performance of various MANET routing protocols. They are

- *Packet Delivery Ratio*: The ratio between the number of packets originated by the application layer CBR sources and the number of packets received by the CBR sink at the final destination.

- Routing Overhead: The total number of routing packets transmitted during the simulation. For packets sent over multiple hops, each transmission of the packet (each hop) counts as one.

- Average Delay: The average time delay between the time when a data packet is given to IP layer at the source node and the time when the packet arrives at the IP layer of the destination.

Packet delivery ratio is an important metric as it describes the loss rate that will be seen by the transport protocols which run on top of the network layer. Thus packet delivery ratio in turn reflects the maximum throughput that the network can support. For all our simulations we have kept the number of data packets sent out as constant, so the number of packets successfully received at their destinations will give us a comparison as to how efficient the underlying routing algorithm is under similar traffic load.

Routing overhead can be computed either as a count of the total bytes sent out as routing packets or a count of total number of routing packets sent out. Here we have used the number of routing packets as our metric because, as we claim that the new routing algorithm suppresses the unnecessary periodic updates sent out by nodes, so we needed a measure of how many routing packets were saved as a result of the new routing algorithm.

With more routing overhead, the amount of delay experienced by data packets in reaching their destination would be more as there would be congestion, collision and

queuing delay. But now as we are reducing the routing overhead, it naturally would lead to better packet delivery times.

5.3 Verification and Validation

5.3.1 Verification

The first step towards verifying our study was to verify the movement scenarios generated for various mobility patterns. In case of deterministic mobility pattern we collected the location of nodes every 10 seconds after the nodes were laid out randomly on the rectangular grid and plotted them. As can be seen in *figure 5.1*, the plot conforms to our expected mobility pattern as that of a frame. Similarly for semi-deterministic mobility pattern we collected the location of nodes at every 3 seconds and it is shown in *figure 5.2*, which show the paths followed by individual nodes.

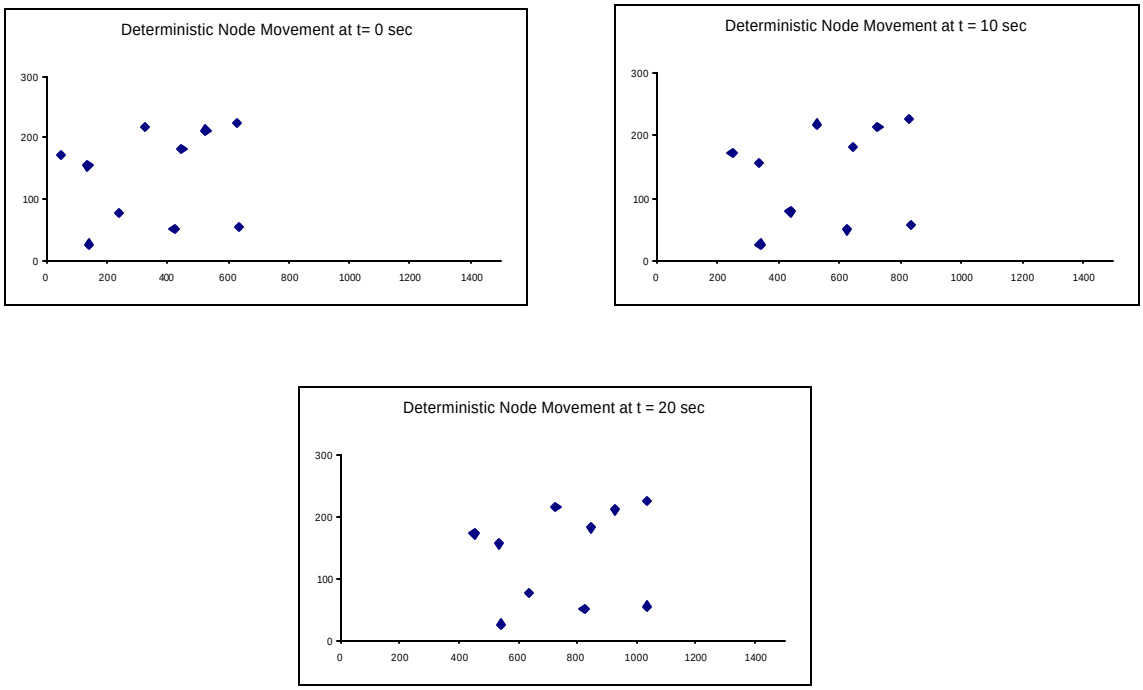


Figure 5.1: *"Nodes following a deterministic mobility pattern"*

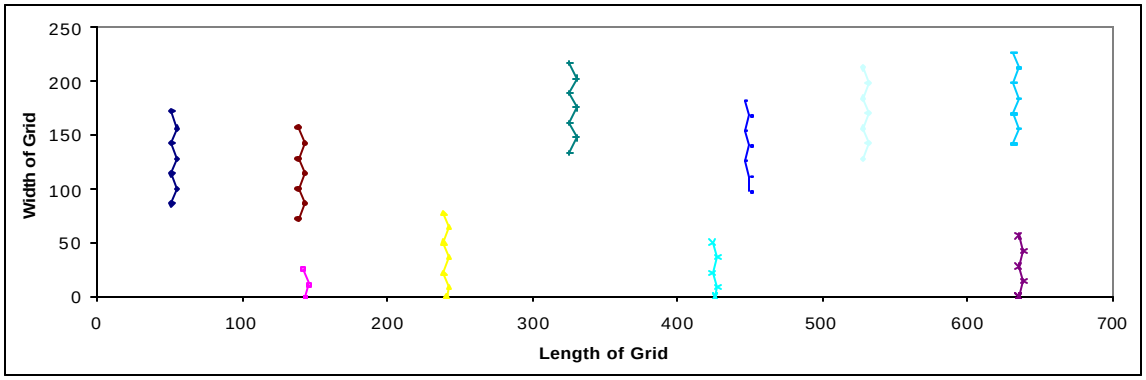


Figure 5.2: *"Nodes following a semi-deterministic mobility pattern"*

To verify the routing algorithms, we started with verification of the existing standard DSDV for specific features. In order to very the standard DSDV, we first considered a network of two nodes. The simulation starts with the nodes placed at a distance more than the transmission range of the nodes and then the nodes move towards each other as the simulation proceeds. On examining the trace files, we could see that initially when the distance between the nodes was more than the transmission range, the packets were dropped but as the nodes moved within transmission range of each other, packets were exchanged. We also verified that standard DSDV does send periodic updates every 3 seconds.

Next to verify our routing protocol, we kept the same topology as well as the movement scenario. Here also initially when the nodes are out of transmission range of each other, the packets were dropped, but as they moved within transmission range of each other, there were exchange of packets. As the routing packets in the new algorithm are bigger than the standard DSDV so as to accommodate the location and time information of the sending node, we verified from the trace file that indeed the size of the routing packet was more by 32 bytes. We also verified that the periodic updates sent out here were suppressed till 15 seconds. Then towards the end of the simulation as the nodes again moved out of each other's transmission range the link snapped as a result of which the *link_repair* module was invoked but as there was no way to repair the link, the nodes just sent out triggered updated which were never received by the destination and eventually the packets were dropped at the link layer.

5.3.2 Validation

As there was no access to physical systems we had no means to validate whether NS2 correctly modeled an ad hoc network. But we tried to validate the base case of our simulation comprising of a 50 node ad hoc network with the nodes following random mobility pattern in a rectangular grid of 1500 x 300 meters, with 20 CBR sources and standard DSDV, with the results published in [76]. As can be seen from our plot of packet delivery ratio for our simulation shown in *figure 5.3*, if not an exact match, the data points and the shape are consistent in the range and the shape of the published results.

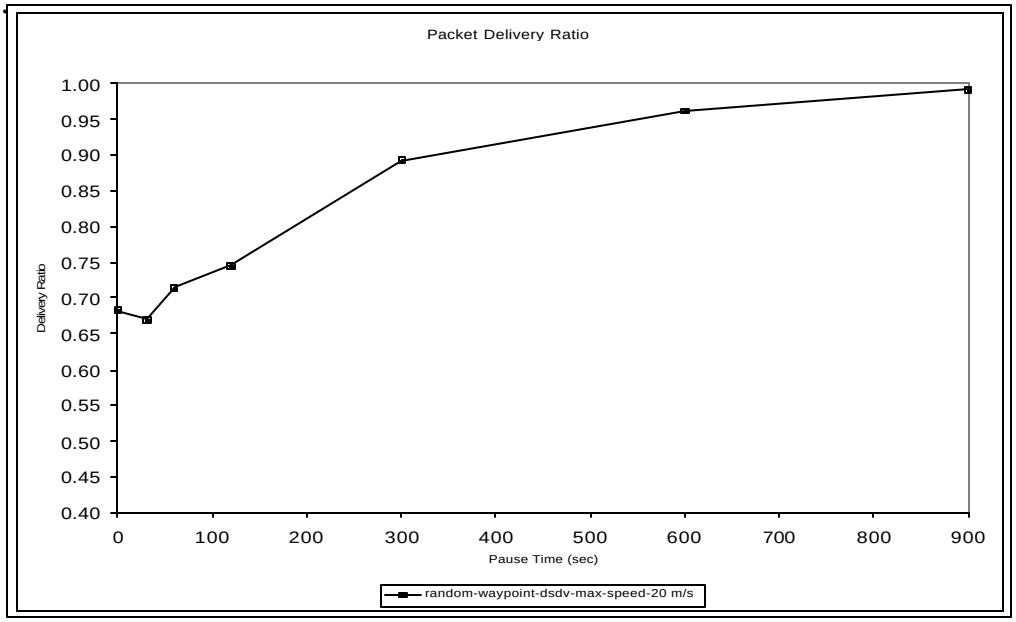


Figure 5.3: *"Standard DSDV packet delivery Ratio as obtained in validation run"*

5.4 Results and Analysis

In this section we present our simulation results for the performance comparison between standard DSDV and mobility pattern aware DSDV. As described earlier, for all the simulations the traffic pattern, the number of mobile nodes, the duration of simulation, the data packet size, the transmission range of the nodes and the link capacity are kept uniform. The variable parameters for every simulation are node speed, pause times, the routing algorithm, the underlying mobility pattern and the initial layout of nodes on the rectangular grid.

5.4.1 Packet Delivery Ratio

As we discussed when defining the simulation metrics, packet delivery ratio is calculated by dividing the total number of data packets received at all the nodes, by the total number of data packets sent out by the CBR sources. Packet delivery ratio forms an important metric for performance evaluation of an ad hoc routing protocol because, given similar scenarios, the number of data packets successfully delivered at the destination depends mainly on path availability, which in turn depends on how effective the underlying routing algorithm is in a mobile scenario.

In the figure 5.4, we plot the packet delivery ratios of the standard DSDV at different speeds to see how the packet delivery ratio varies at speeds of 20m/s and 10 m/s. Here for each speed different movement scenarios were generated using the random waypoint model first with a max speed of 20 m/s and then without the trailing effect of speed that is at a fixed node speed of 20 m/s. Similarly, the experiments were also conducted for speed 10 m/s, with and without the trailing effect. Each data point used to plot the graphs is an average of 10 runs where each run starts with a different layout of the mobile nodes on the topography.

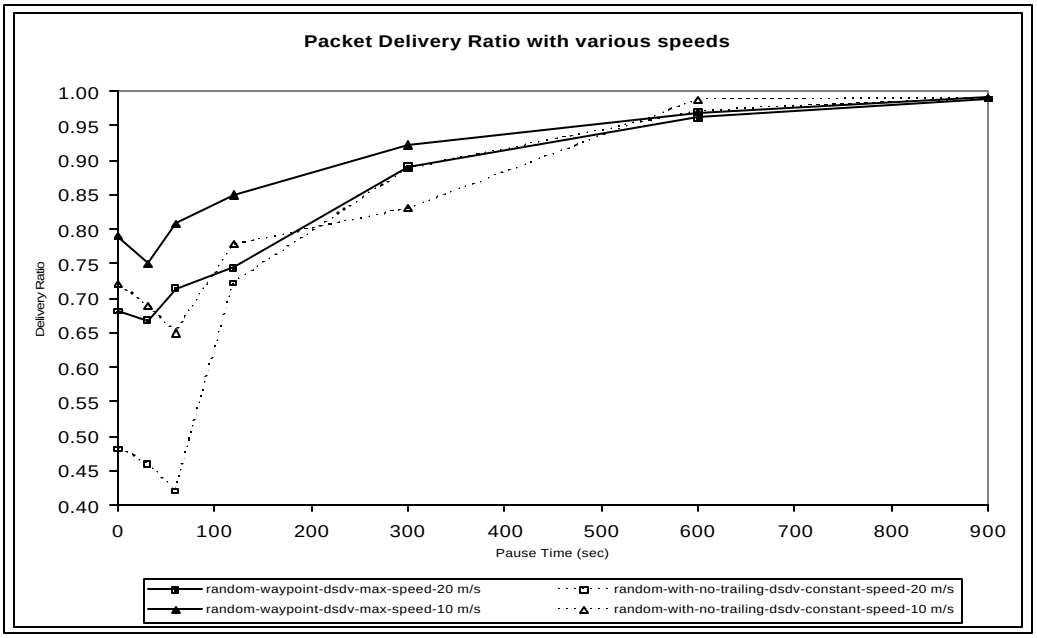


Figure 5.4: *"Packet delivery Ratio with Standard DSDV"*

As can be seen from the graph, standard DSDV's packet delivery ratio is poor at high mobility i.e. at smaller pause times i.e. less than 300 seconds because of stale routing tables and hence the packets were forwarded over a broken link which were eventually dropped. The other reason for packet drops was due to the delay at the MAC layer because of congestion caused by the routing overhead. But its performance improves with increasing pause times indicating that at low mobility the packet delivery ratio of standard DSDV is better.

Next we plot the packet delivery ratio of the mobility pattern aware DSDV in *figure 5.5* and *figure 5.6* at speed 20 m/s and 10 m/s respectively. Here in each plot there are three curves, with each curve corresponding to the packet delivery ratio of the modified DSDV with a different underlying mobility pattern that is random, semi-deterministic or deterministic.

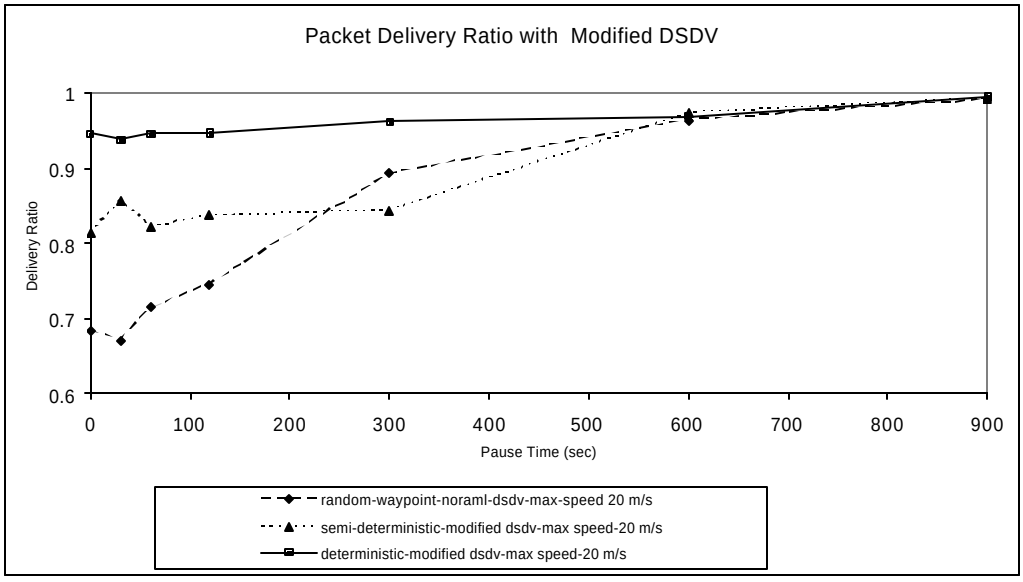


Figure 5.5: *"Packet delivery Ratio with Mobility Pattern Aware DSDV at speed of 20 m/s"*

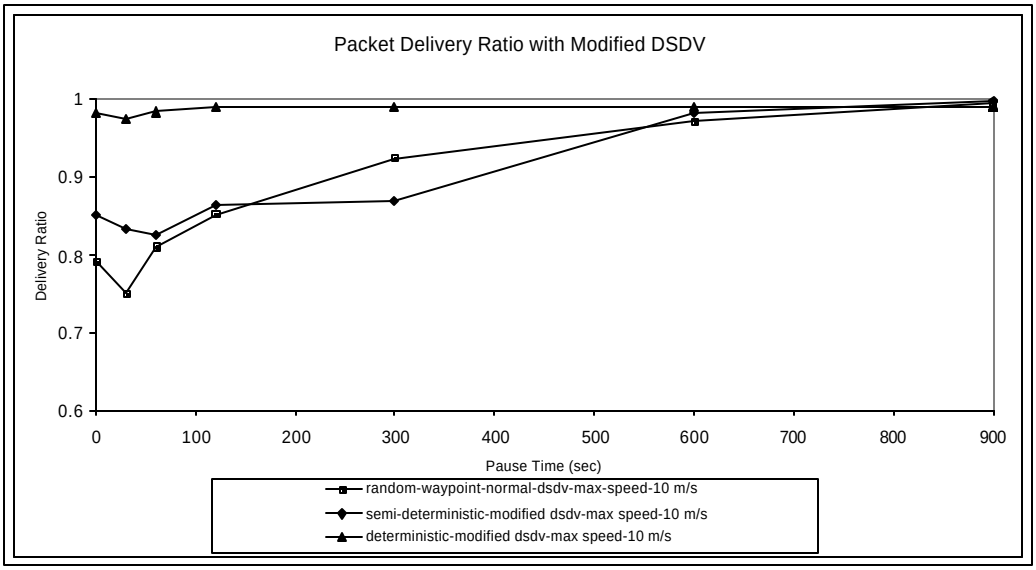


Figure 5.6: *"Packet delivery Ratio with Mobility Pattern Aware DSDV at speed of 10 m/s"*

Consider the curves corresponding to random mobility pattern in the graphs, here as soon as the mobility pattern aware DSDV finds out the underlying mobility pattern is random, it switches its mode of operation to standard DSDV and hence these curves are very similar in shape and range to the standard DSDV's packet delivery ratio curves shown in *figure 5.4*. But in case of semi-deterministic and deterministic mobility pattern, as can be seen from the plots there is considerable improvement in packet delivery ratio. The reason being, as the nodes here moved in a frame like structure with their relative location being same throughout the duration of simulation, there were no or very few broken links.

5.4.2 Routing Overhead

As we discussed in *section 5.2.5*, the routing overhead graphs shown in this section are based on total number of routing packets sent by the network layer during the course of the simulation.

The simulation parameters for plotting the graphs shown in this section are same as mentioned in the previous section. In *figure 5.7*, we plot the routing overhead incurred by standard DSDV at speeds 20 m/s and 10 m/s.

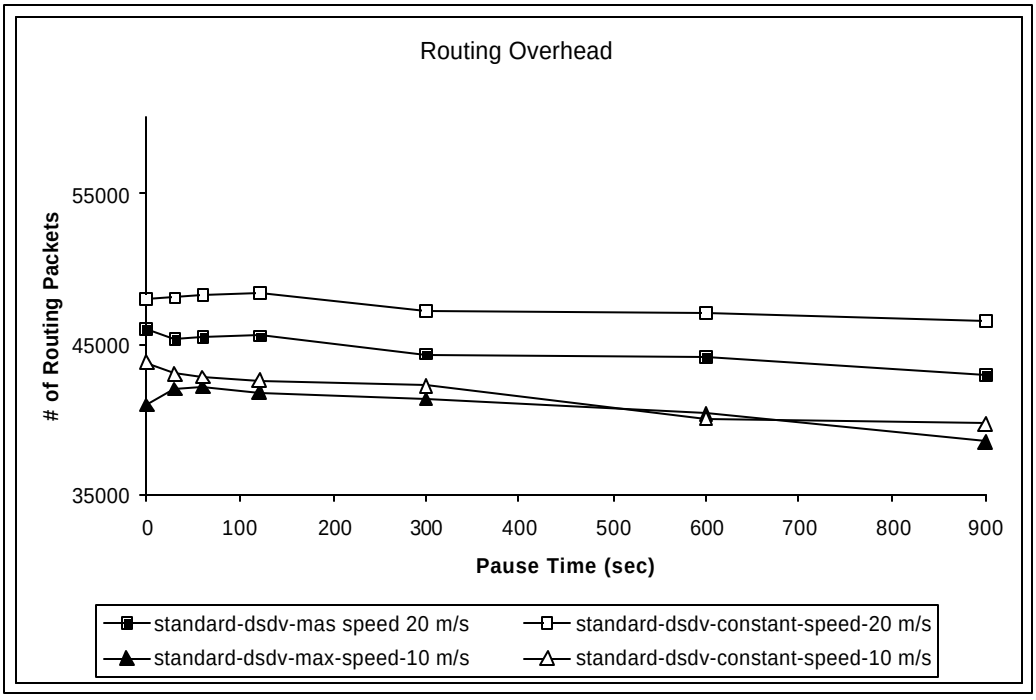


Figure 5.7: "*Routing overhead incurred with Standard DSDV*"

As can be seen from the plot, the routing overhead is high with standard DSDV and mobility doesn't seem to have much effect on the routing overhead, this is because of the periodic updates sent out as well as the triggered updates. However with increasing pause times the curves droop a little, because of fewer triggered updates being sent out at low mobility.

Next we plot the routing overhead with mobility pattern aware DSDV in *figure 5.8* and *figure 5.9* at speed 20 m/s and 10 m/s respectively. Here also in each plot there are three curves and curve corresponds to routing overhead incurred by the modified DSDV with a different underlying mobility pattern that is random, semi-deterministic or deterministic.

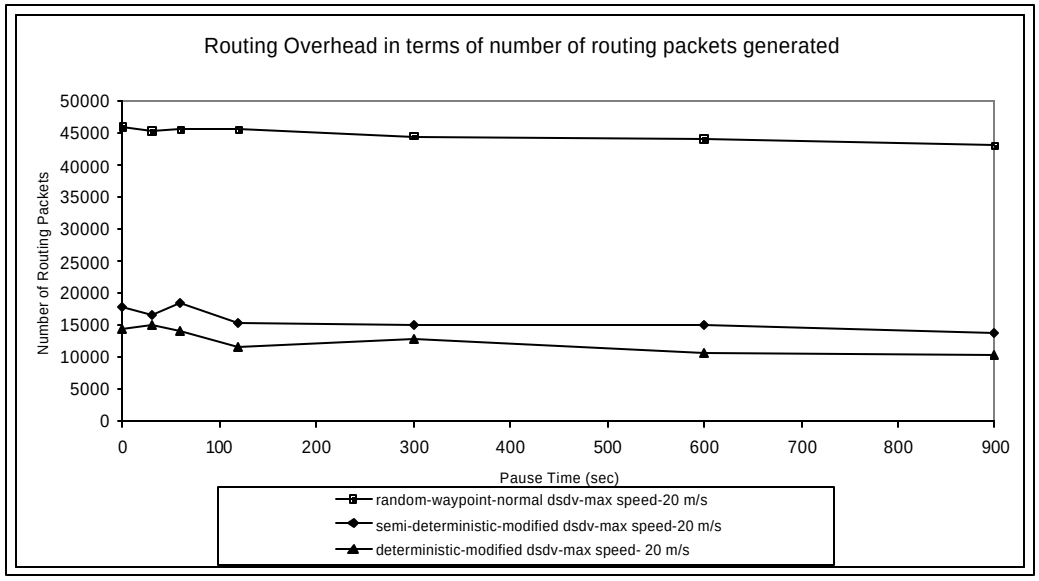


Figure 5.8: *"Routing overhead incurred with Mobility Pattern Aware DSDV with speed 20 m/s"*

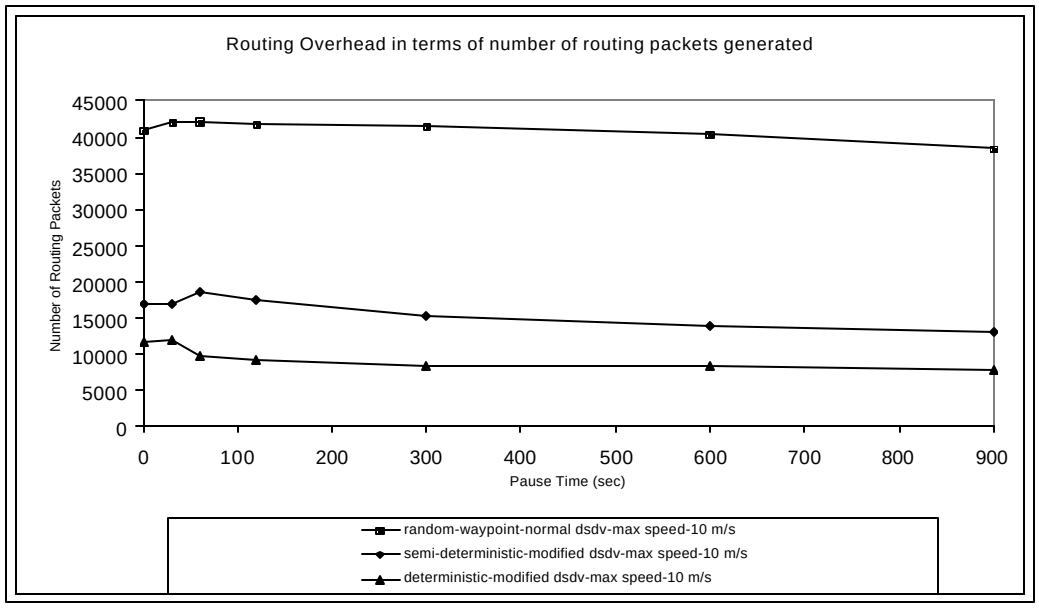


Figure 5.9: *"Routing overhead incurred with Mobility Pattern Aware DSDV with speed 10 m/s"*

As can be seen from the graphs, the routing overhead in mobility pattern aware DSDV is much less when the nodes follow either deterministic or semi-deterministic mobility pattern when compared to that of random. This is so because, with random mobility pattern, the mobility pattern aware DSDV functions as standard DSDV, which suffers from higher routing overhead resulting from frequent periodic as well as triggered updates. These triggered updates have an exponential effect on the number of routing packets generated, because each triggered update generates a new sequence number which in turn causes the receiving nodes also to send out triggered updates. But when the nodes follow either deterministic or semi-deterministic mobility pattern, the new routing algorithm restraints itself from sending frequent periodic updates. In addition to that, the number of triggered updates generated due to link failure is also considerably less because the strategy here is to first try to repair the broken link and report link failure if it cannot be repaired.

5.4.3 Average Delay

Average delay is the time a data packet takes in traversing from the time it is sent by the source node till the point it is received at the destination node. This metric is a measure of how efficient the underlying routing algorithm is, because primarily the delay depends upon optimality of path chosen, the delay experienced at the interface queues and delay caused by the retransmissions at the physical layer due to collisions. Routing overhead is a major factor affecting the interface queuing delay as well as the retransmissions. Because the higher the routing overhead the delay experienced at the queues will be longer as well as the number of collision would be high.

Here also the simulation parameters for plotting the graphs shown in this section are same as mentioned in the previous sections. In *figure 5.10*, we plot the delay incurred by standard DSDV at speeds 20 m/s and 10 m/s. To collect the data points for this graph, we added up the delay times of 100 successfully delivered packets and took an average to rule out errors.

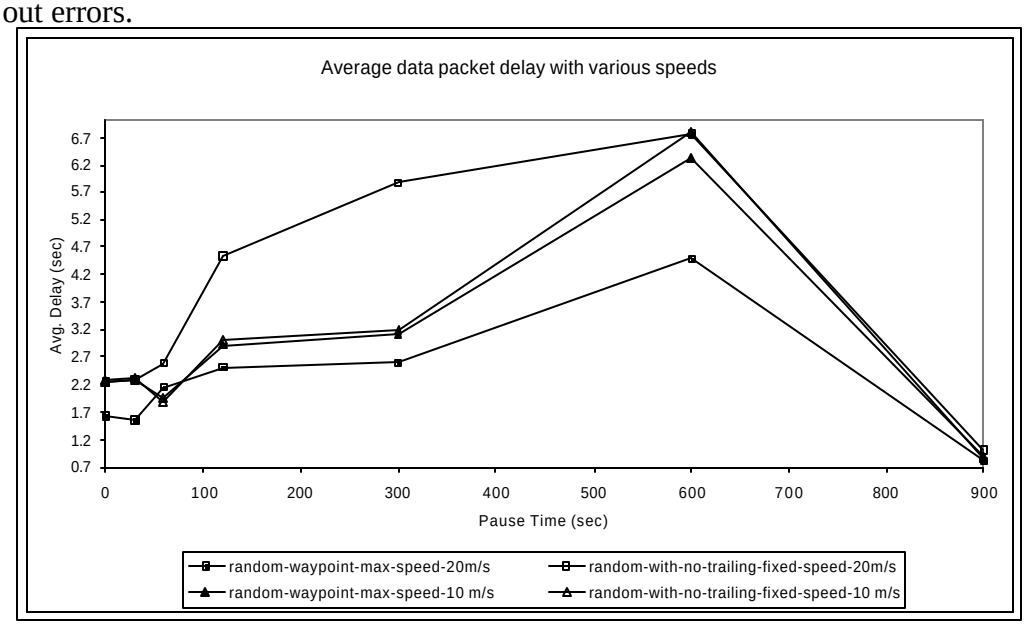


Figure 5.10: *"Average Delay incurred with Standard DSDV"*

As can be seen from the plot, overall delay experienced by packets in standard DSDV is quite high which is at its peak at pause time of 600 seconds, which mean that in addition to the queuing delay and delay due to retransmissions, the paths are not available at early stages of simulation.

Next we plot the average delay with mobility pattern aware DSDV in *figure 5.11* and *figure 5.12* at speed 20 m/s and 10 m/s respectively. Here also in each plot there are three curves and curve corresponds to average delay incurred by the modified DSDV with a different underlying mobility pattern that is random, semi-deterministic or deterministic.

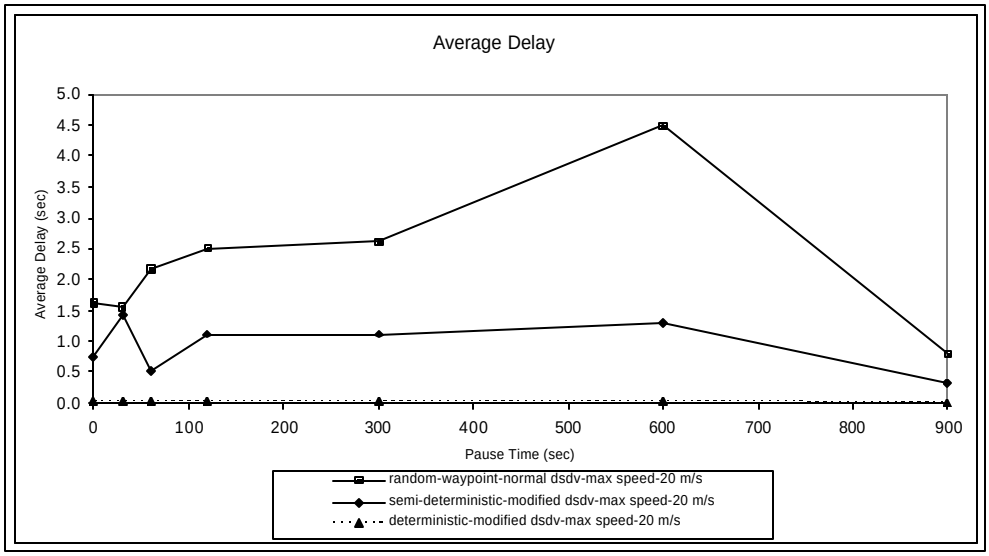


Figure 5.11: *"Average Delay incurred with Mobility Pattern Aware DSDV with speed 20 m/s"*

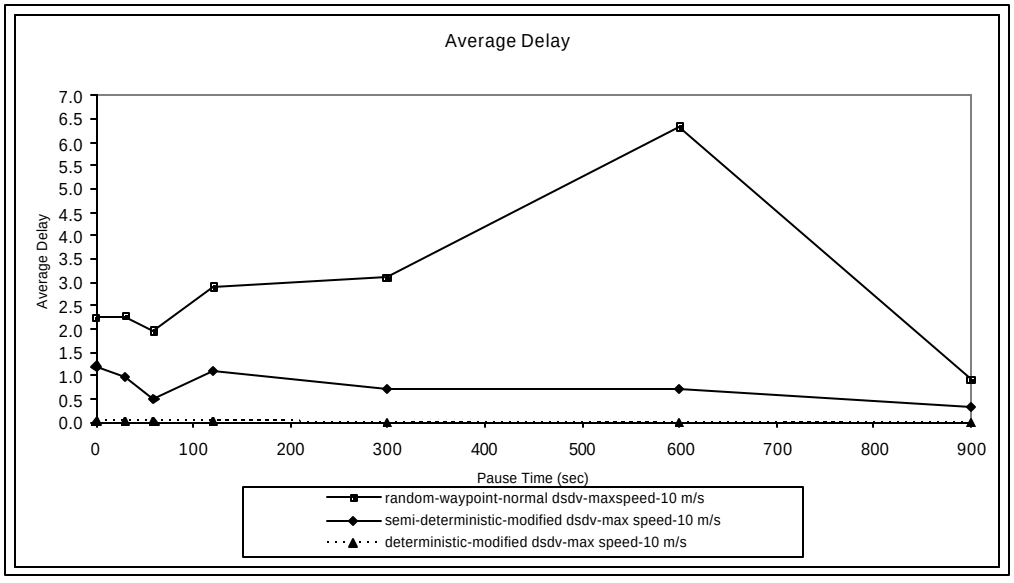


Figure 5.12: *"Average Delay incurred with Mobility Pattern Aware DSDV with speed 10 m/s"*

As can be seen from the graphs, the average delay in mobility pattern aware DSDV is much less when the nodes follow either deterministic or semi-deterministic mobility pattern when compared to that of random. One of the reasons is, with random mobility pattern, the mobility pattern aware DSDV functions as standard DSDV, which suffers from higher routing overhead which results in increased queuing delay as well as higher retransmission dues to congestion resulting in higher end to end delay in delivering packets which is not the case in deterministic and semi-deterministic model.

5.5 Additional Results – Accuracy of Location Prediction

One of the important features of our new routing protocol, the mobility pattern aware routing algorithm is to be able to predict the future location of mobile nodes when the underlying mobility pattern is either deterministic or semi-deterministic. But the experiments we have carried out so far did not quite bring forth this feature, as in both deterministic and semi-deterministic mobility pattern, the nodes were moving as a frame. As a result, there were never any link failures. So to bring forth the location prediction feature of the new routing algorithm, we designed a slightly different movement scenario. Here even though the nodes follow a deterministic mobility pattern, the network as whole does not move as a frame that is the node movement is independent. This model closely resembles to the urban traffic model, where the nodes are restricted to move either in vertical or horizontal direction as shown in *figure 5.13* where we plot the sample path followed by 10 nodes traveling at 20 m/s.

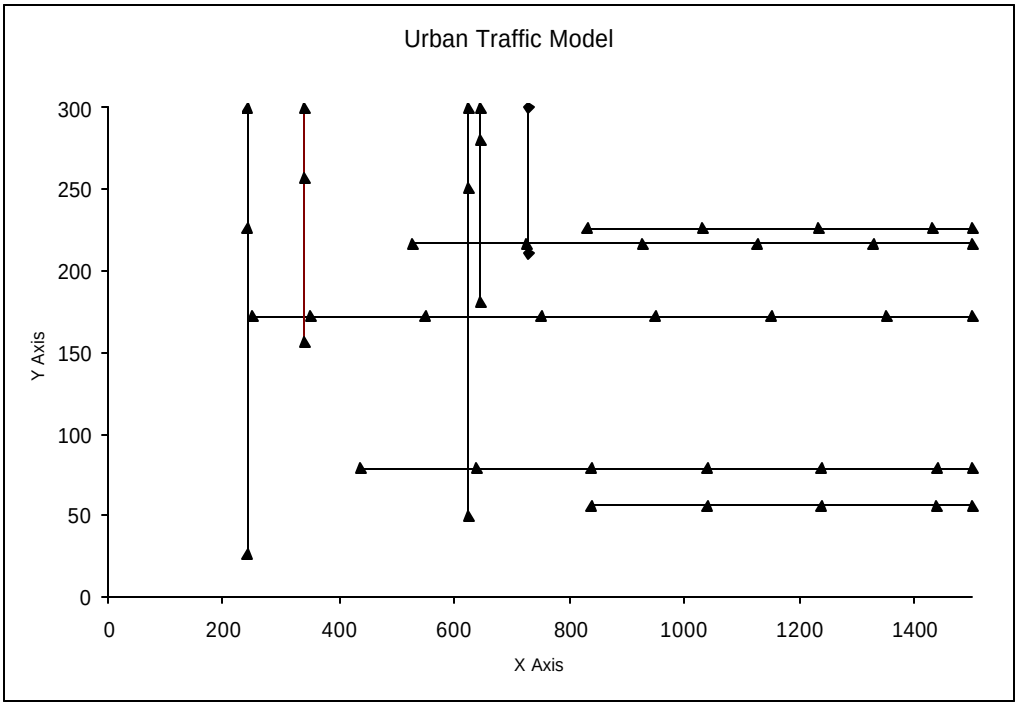


Figure 5.13: *"Movement Scenario of an Urban Traffic"*

Here we used prediction error metric to measure the accuracy of our prediction algorithm. This prediction error is defined as the percentage of predictions that were off the actual location by a particular distance. The location_prediction module is invoked when an attempt to repair a broken link is made. At that instant we collected the distance error between the actual and the predicted location of the node and then we grouped them into ranges to plot. *Figure 5.14* shows the predicted error of our prediction algorithm.

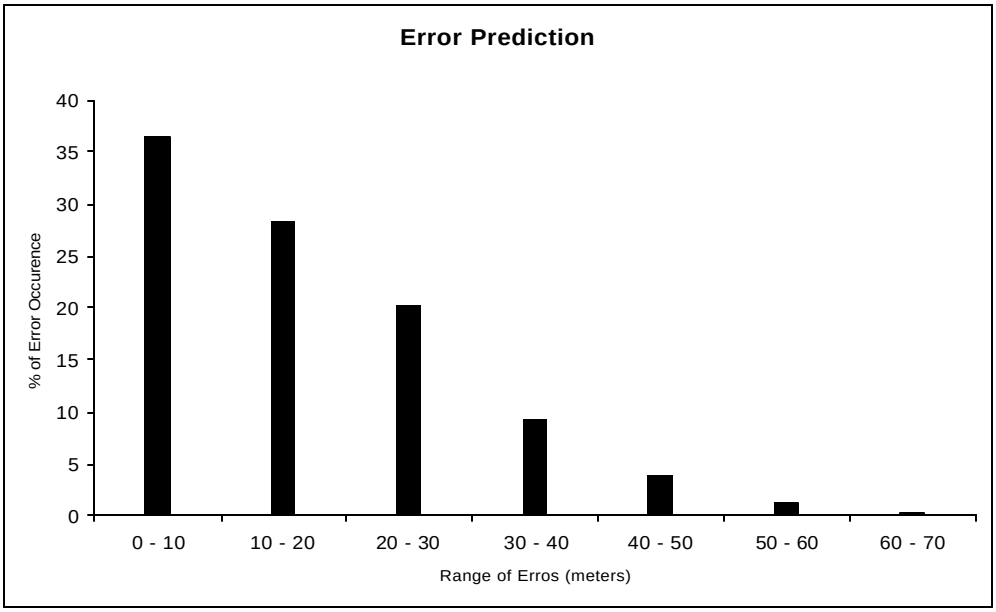


Figure 5.14: *"Prediction Error"*

Thus as can bee seen from the plot, the prediction accuracy is considerably high and even in the worst case of prediction error, the error is not so significant as the error is much less compared to the transmission range of the nodes, so even though the prediction error which goes to a maximum of 69 meters falling in the 60 -70 range, the routing algorithm was still able to find a alternate path for the broken link.

5.6 Chapter Summary

In this chapter we first explained our simulation environment, NS2. Next we looked into the finer details of the simulation methodology and giving each experimental setup. We also talked about how we attempted to validate as well as verify our results. And finally we presented the results from our experiments.

Chapter 6: Conclusion and Future Work

6.1 Conclusion

In this thesis, we proposed a mobility pattern aware DSDV routing algorithm which is self learning about the underlying mobility pattern.

Standard DSV assumes that nodes always move in a random fashion; hence it expends considerable amount of network resources in maintaining topology and connectivity information. But this assumption is not always true, since the nodes can move in a deterministic or semi-deterministic pattern, in which case DSDV unnecessarily expends network resources which can be done away with. We had mentioned in the motivation section of this thesis that the performance of DSDV can be improved if it had knowledge about the underlying mobility pattern of the nodes. Since, DSDV then would not waste its efforts, in trying to maintain the topology and connectivity information.

True to our expectations, mobility pattern aware DSDV does improve the efficiency of the DSDV routing protocol and this is confirmed by our results. Depending upon the underlying mobility pattern, which the algorithm learns of its own using the past location history of nodes, it adapts itself accordingly. And as the frequent periodic updates are suppressed, the routing over head is considerably low, which results in less congestion (collision and retransmissions) in the network resulting in higher packet delivery ratio as well as less delay in packet delivery.

This gain in performance however is not free of cost. And the cost associated with it is the amount of memory used in maintaining the history tables at each node as well as transmitting the location information along with the routing updates which increase the size of the routing updates. With smaller network sizes, this might not be an issue but as the network size increases more and more, the storage overhead could be large.

6.2 Future Work

- One very obvious offshoot of our proposed algorithm is to provide QoS guarantee.

- Scalability could be an issue with the routing algorithm as we discussed earlier due to the increase in size of the history table as well as the routing updates with the increase in number of nodes in the network. This is another area, which can be worked upon. Some algorithms have been proposed in this area so far which can be tried out in this case as well to see how effective they are in reducing the size of the history table and the routing updates (fish eye approach).

- In implementing the new routing algorithm, we had fixed the periodic update intervals at 15 and 12 seconds for deterministic and semi-deterministic mobility patters. It would be interesting to see at what periodic intervals the mobility pattern aware routing algorithm give best performance results.

References

[1] C. E. Perkins, and P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance Vector Routing," ACM SIGCOMM Computer Commun. Rev., Oct. 1994, pp. 234-244.

[2] S. Murthy, and J. J. Garcia-Luna-Aceves, "An Efficient Routing Protocol for Wireless Networks," ACM Mobile Networks and Applications Journal, Special Issue on Routing in Mobile Communication Networks, vol. 1, no. 2, Oct. 1996, pp. 183-197.

[3] V. D. Park, and M. S. Corson, "A Highly Adaptive Distributed Routing Algorithm for Mobile Wireless Networks," Proc. 16th Annual Joint Conf. of IEEE Computer and Communications Societies (INFOCOM 1997), 1997.

[4] B. Das, R. Sivakumar and V. Bharghavan, "Routing in Ad-Hoc Networks Using a Spine," IEEE Intl. Conf. on Communications (ICC 97), Las Vegas, NV, Sep. 1997.

[5] Z. J. Haas and M. R. Pearlman, "Determining the Optimal Configuration for the Zone Routing Protocol," IEEE J. Select. Areas Commun., vol. 17, No. 8, Aug. 1999, pp. 1395-1414.

[6] D. Johnson and D. A. Maltz, "Dynamic Source Routing in Adhoc networks," in Mobile Computing (T. Imielinski and H. F. Korth, eds.), cha. 5 pp. 153-181, Dordrecht, The Netherlands: Kluwer Academic Publishers, February 1996.

[7] C.E. Perkins and E.M. Royer, "Ad-Hoc On-Demand Distance Vector Routing," Proceedings of IEEE WMCSA'99, New Orleans, LA, Feb. 1999, pp. 90-100.

[8] M. Mauve, J. Widner, and H. Hartenstein, "A Survey on Position-Based Routing in Mobile Ad-Hoc Networks," IEEE Network, vol. 16, Nov./Dec. 2001, pp. 30-39.

[9] P. Johansson, T. Larsson, N. Hedman, B. Mielczarek, and M. Degermark, "Scenario-Based Performance Analysis of Routing Protocols for Mobile Ad-hoc Networks," Proc. 5th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MobiCom '99), Seattle, WA, Aug. 15-19, 1999, pp. 195-206.

[10] G. Pei, M. Gerla and T.-W. Chen, "Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks," Proc. 2000 ICDCS Workshop on Wireless Networks and Mobile Computing, Taipei, Taiwan, Apr. 2000, pp. D71-D78.

[11] Z. J. Haas and B. Liang, "Ad-Hoc Mobility Management With Uniform Quorum Systems," IEEE/ACM Trans. on Networking, vol. 7, no. 2, Apr. 1999, pp. 228-240.
[12] B. N. Karp, "Geographic Routing for Wireless Networks," Ph.D. Thesis, Boston, MA: Harvard University, 2001.

[13] Y.-B. Ko and N. H. Vaidya, "Location-Aided Routing (LAR) in Mobile Ad Hoc Networks ," Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and

Networking (MobiCom '98), Dallas, TX, Oct. 1998, pp. 66-75. Also in ACM/Baltzer Wireless Networks Journal, vol. 6, no. 4, 2000, pp. 307-321.

[14] Y.-B. Ko and N. H. Vaidya, "Using Location Information in Wireless Ad Hoc Networks," IEEE Vehicular Technology Conf. (VTC '99), May 1999.

[15] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris, "A Scalable Location Service for Geographic Ad Hoc Routing," Proc. 6th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom '00), Boston, MA, Aug. 2000, pp. 120-130.

[16] A.B. McDonald and T.F. Zanati, “A mobility-based framework for adaptive clustering in wireless ad hoc networks", IEEE Journal on Selected Areas in Communications, Vol 17, No. 8, Aug 1999, pp. 1466-1487.

[17] A.B. McDonald and T. Znati, "Predicting Node Prox-imity in Ad-Hoc Networks: A Least Overhead Adap-tive Model for Selecting Stable Routes", In Proceed-ings of ACM/MobiHoc2000, Boston, MA, August2000, pp. 29-33.

[18] A.B. Mcdonald and T. Znati, "A Path Availability Model for Wireless Ad-Hoc Networks", In Proceed-ings of the IEEE Wireless Communications and Net-working Conference 1999 (WCNC'99), New Orleans,LA, September 21-24, 1999.

[19] T. Liu, P. Bahl, and I. Chlamtac, "Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks,"IEEE J. Sel. Areasin Commun., vol. 16, no. 6, pp. 922-936, Aug. 1998.

[20] M.M Zonoozi and P. Dassanayake, “User Mobility Modeling and Characterization of Mobility Patterns”, IEEE Journal on Selected Areas in Communications, Vol. 15, No. 7, September, 1997.

[21] Sanchez, M. Mobility Models:http://www.disca.upv.es/misan/mobmodel.htm

[22] Tan D.S., Zhou S., Ho, J, Mehta, J.S. Tanabe, H. (2002). “Design and Evaluation of an Individually Simulated Mobility Model in Wireless Ad Hoc Networks”. Communication Networks and Distributed Systems Modeling and Simulation Conference 2002, San Antonio, TX.

[23] Reynolds, C.W. 1987. Flocks, Herds, and Schools: “A Distributed Behavioral Model”. In Siggraph ’87 Conference Proceedings, 21(4), 25-34.

[24] Hong, X., Gerla, M., Pei, G. and Chiang, C. 1999. “A Group Mobility Model for Ad Hoc Wireless Networks”. Proceedings of the 2nd ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems. August 20, 53-60.

[25] S. H. Shah and K. Nahrstedt, "Predictive Location-Based QoS Routing in Mobile Ad Hoc Networks," Tech. Rept. UIUCDCS-R-2001-2242 - UILU-ENG-2001-1749, Dept. of Computer Science, University of Illinois at Urbana-Champaign, September, 2001; to appear in Proc. IEEE Intl. Conf. Commun. (ICC 2002), New York, NY, April 28-May 2, 2002.

[26] S-J. Lee, W. Su and M. Gerla, "Ad hoc Wireless Multi-cast with Mobility Prediction", In Proceedings of 8thInternational Conference on Computer Communications and Networks (ICCN'99), 1999, pp. 4-9.

[27] W. Su, S-J. Lee and M. Gerla, "Mobility Prediction in Wireless Networks", In Proceedings of IEEE MILCOM 2000, Vol. 1, Los Angelas, CA, Oct 2000, pp 491-495.

[28] X. Hong, M. Gerla, G. Pei, and C. Chiang, "A Group Mobility Model forAd Hoc Wireless Networks," In Proceedings of the 2nd ACM International Workshop on Modeling and Simulation of Wireless and Mobile Systems,1999.

[29] Satyabrata Chakrabarti and Amitabh Mishra, “Quality of Service in Mobile Ad Hoc Networks”, Chapter 2.

[30] David B. Johnson, ``Routing in Ad Hoc Networks of Mobile Hosts," *Proceedings of the Workshop on Mobile Computing Systems and Applications, IEEE Computer Society,* Santa Cruz, CA, pages 158-163,December 1994.

[31] R. Ramanathan and M. Steenstrup, “Hierarchically-Organized, Multihop Mobile Wireless Networks for Quality-of-Service Support,” ACM/Baltzer Mobile Networks and Applications, vol. 3, no. 1, Jun. 1998, pp. 101-119.

[32] Pei, G, Gerla, M. Chen, T. Fisheye State Routing: A Routing Scheme for Ad Hoc Wireless Networks Proceedings of the IEEE International Conference on Communications. 2000.

[33] J.J Garcia-Luna-Aceves, ``Efficient Routing in Packet-Radio Networks Using Link-State Information," Proceedings of IEEE Wireless Communications and Networking Conference 1999 WCNC'99, New Orleans, Louisiana, September 21-24, 1999.

[34] C.-C. Chiang, "Routing in Clustered Multihop, Mobile Wireless Networks with Fading Channel" Proceedings of IEEE SICON'97, Apr.1997, pages 197-211.

[35] M. Mauve, J. Widmer, and H. Hartenstein, “A Survey on Position-Based Routing in Mobile Ad-Hoc Networks,” IEEE Network, vol. 16, Nov./Dec. 2001, pp. 30-39.

[36] S. Basagni, I. Chlamtac, V. R. Syrotiuk, and B. A. Woodward, “A Distance Routing Effect Algorithm for Mobility (DREAM),” Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘98), Dallas, TX, Oct. 25-30, 1998, pp. 76-84.

[37] P. Bose, P. Morin, I. Stojmenovic and J. Urrutia, “Routing with Guaranteed Delivery in Ad Hoc Wireless Networks,” Proc. 3rd ACM Intl Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications (DIAL M ‘99), Seattle, Aug. 20, 1999, pp. 48-55.

[38] S. Basagni, I. Chlamtac, and V. R. Syrotiuk, “Geographic Messaging in Wireless Ad Hoc Networks,” Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘98), Dallas, TX, Oct. 25-30, 1998, pp. 76-84.

[39] S. Capkun, M. Hamdi, and J. P. Hubaux, “GPS-free Positioning in Mobile Ad-Hoc Networks,” Proc. 34th Annual Hawaii Intl. Conf. on System Sciences, Jan. 2001.

[40] Y. Xue and Baochun Li, “A Location-aided Power-aware Routing Protocol in Mobile Ad Hoc Networks,” Proc. IEEE Symp. on Ad Hoc Mobile Wireless Networks/IEEE GLOBECOM 2001, San Antonio, TX, Nov. 25-29, 2001.

[41] D. S. J. De Couto and R. Morris, “Location Proxies and Intermediate Node Forwarding for Practical Geographic Forwarding,” Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘98), Dallas, TX, Oct. 25-30, 1998.

[42] Z. J. Haas and B. Liang, “Ad-Hoc Mobility Management With Uniform Quorum Systems,” IEEE/ACM Trans. on Networking, vol. 7, no. 2, Apr. 1999, pp. 228-240.

[43] B. N. Karp, “Geographic Routing for Wireless Networks,” Ph.D. Thesis, Boston, MA: Harvard University, 2001.

[44] B. N. Karp and H. T. Kung, “GPSR: Greedy Perimeter Stateless Routing for Wireless Networks,” Proc. 6th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘00), Boston, MA, Aug. 2000, pp. 243-254.

[45] J. Li, J. Jannotti, D. S. J. De Couto, D. R. Karger, and R. Morris, “A Scalable Location Service for Geographic Ad Hoc Routing,” Proc. 6th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘00), Boston, MA, Aug. 2000, pp. 120-130.

[46] Y.-B. Ko and N. H. Vaidya, “Location-Aided Routing (LAR) in Mobile Ad Hoc Networks ,” Proc. 4th Annual ACM/IEEE Intl. Conf. on Mobile Computing and Networking (MobiCom ‘98), Dallas, TX, Oct. 1998, pp. 66-75. Also in ACM/Baltzer Wireless Networks Journal, vol. 6, no. 4, 2000, pp. 307-321.

[47] T.-W.Chen and M.Gerla, ``Global State Routing: A New Routing Scheme for Ad-hoc Wireless Networks," In Proceedings of IEEE ICC'98, Atlanta, GA, pages 171-175, Jun. 1998.

[48] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla and T.-W. Chen, ``Scalable Routing Strategies for Ad Hoc Wireless Networks,", *JSAC99, IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pages 1369-1379, August 1999.

[49] Ivan Stojmenovic, *Home Agent Based Location Update and Destination Search Schemes in ad hoc wireless networks*, 1999, http://www.site.uottawa.ca/ivan/mobile-home-TR.ps.

[50] A.B. McDonald and T.F Znati, ``A Mobility-Based Framework for Adaptive Clustering in Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pages1466-1487, August 1999.

[51] Ben Liang and Zygmunt J. Haas, ``Virtual Backbone Generation and Maintenance in Ad Hoc Network Mobility Management," *Proceedings of IEEE INFOCOM 2000*, Tel Aviv, Israel, March 2000.

[52] Volkan Rodoplu and Teresa H. Meng, ``Minimum Energy Mobile Wireless Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pages 1333-1344, August 1999.

[53] Suresh Singh, C.S. Raghavendra and James Stepanek, ``Power-Aware Broadcasting in Mobile Ad Hoc Networks," *Proceedings of IEEE PIMRC'99*, Osaka, Japan, September 1999.

[54] S. Chen and K. Nahrstedt, ``Distributed Quality-of-Service Routing in Ad Hoc Networks," *IEEE Journal on Selected Areas in Communications*, Vol. 17, No. 8, pages 1488-1505, August 1999.

[55] C.R. Lin and J.-S, Liu, ``QoS Routing in Ad Hoc Wireless Networks," IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, pages 1426-1438, August 1999.

[56] R. Sivakumar, P. Sinha and V. Bharghavan, ``CEDAR: A Core-Extraction Distributed Ad Hoc Routing Algorithm," IEEE Journal on Selected Areas in Communications, Vol. 17, No. 8, pages 1454-1465, August 1999.

[57] T. W. Chen, J. T. Tsai and M. Gerla, ``QoS Routing Performance in Multihop, Wireless Networks," IEEE 6th ICUPC'97, October 1997.

[58] F. A. Tobagi and L. Kleinrock. “Packet switching in radio channels: Part ii -the hidden terminal problem in carrier sense multiple-access modes and the busy-tone solution.” IEEE Transactions on Communications, COM-23(12):1417{1433, 1975.

[59] T. Ue, S. Sampei, N. Morinaga, and K. Hamaguchi. “Symbol rate and modulation level-controlled adaptive modulation/TDMA/TDD system for high-bit-rate wireless data transmission.” IEEE Transactions on Vehicular Technology, 47(4):1134{1147, November 1998.

[60] Tput J. H. Gass, M. B. Pursley, H. B. Russell, R. J. Saulitis, C. S. Wilkins, and J. S. Wysocarski. “Adaptive transmission protocols for frequency-hop radio networks.” In Proceedings of the 1998 IEEE Military Communications Conference, volume 2, pages 282{286, October 1998.

[61] C. L. Fullmer and J. J. Garcia-Luna-Aceves. “Solutions to hidden terminal problems in wireless networks.” In ACM SIGCOMM '97, pages 14{18, Cannes, France, September 1997.

[62] V. Bharghavan. MACAW: “A media access protocol for wireless LAN's.” In Proceedings of SIGCOMM'94, London, 1994.

[63] S. M. Alamouti and S. Kallel. “Adaptive trellis-coded multiple-phase-shift keying for rayleigh fading channels.” IEEE Transactions on Communications, 42:2305{2314, June 1994.

[64] K. Balachandran, S. R. Kadaba, and S. Nanda. “Channel quality estimation and rate adaption for cellular mobile radio.” IEEE Journal on Selected Areas in Communications, 17(7):1244{1256, July 1999.

[65] R. Caceres and L. Iftode, "Improving the per- formance of reliable transport protocols in mo- bile computing environments," IEEE Journal on Selected Areas in Communications, vol. 13, June 1995.

[66] K. Char&an, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback based scheme for improv- ing TCP performance in ad-hoc wireless networks," in Proceedings of International Conference on Dis- tributed Computing Systems, (Amsterdam), 1998.

[67] R. Caceres and L. Iftode, "Improving the per- formance of reliable transport protocols in mo- bile computing environments," IEEE Journal on Selected Areas in Communications, vol. 13, June 1995.

[68] H. Balakrishnan and R. Katz, "Explicit loss noti- fication and wireless web performance," in IEEE Globecom Internet Mini-Conference, Sydney, Oct. 1998.

[69] M. Gerla, K. Tang, and R. Bagrodia, "TCP per- formance in wireless multi-hop networks," in Pro- ceedings of IEEE WMCSA'99 , (New Orleans, LA), February 1999.

[70] T. Liu, P. Bahl, and I. Chlamtac, "Mobility modeling, location tracking, and trajectory prediction in wireless ATM networks,"IEEE J. Sel. Areasin Commun., vol. 16, no. 6, pp. 922-936, Aug. 1998.

[71] M.M Zonoozi and P. Dassanayake, “User Mobility Modeling and Characterization of Mobility Patterns”, IEEE Journal on Selected Areas in Communications, Vol. 15, No. 7, September, 1997.

[72] Sanchez, M. Mobility Models:http://www.disca.upv.es/misan/mobmodel.htm

[73] Tan D.S., Zhou S., Ho, J, Mehta, J.S. Tanabe, H. (2002). “Design and Evaluation of an Individually Simulated Mobility Model in Wireless Ad Hoc Networks”. Communication Networks and Distributed Systems Modeling and Simulation Conference 2002, San Antonio, TX.

[74] Hong, X., Gerla, M., Pei, G. and Chiang, C. 1999. “A Group Mobility Model for Ad Hoc Wireless Networks”. Proceedings of the 2nd ACM International Workshop on Modeling, Analysis, and Simulation of Wireless and Mobile Systems. August 20, 53-60.

[75] UCB/LBNL/VINT Network Simulator - ns (version2): http://www-mash.CS.Berkeley.EDU/ns/

Appendix A

List of Key Words

ACRONYM	Expansion
AODV	Ad-hoc On-demand Distance Vector (routing)
CSGR	Clusterhead Gateway Switch Routing
DSDV	Destination-Sequenced Distance-Vector (routing)
DSR	Dynamic Source Routing
FSR	Fisheye State Routing
GPS	Global Positioning System
MAC	Medium Access Control
QoS	Quality of Service
SRP	Secure Routing Protocol
STAR	Source-Tree Adaptive Routing
TORA	Temporally Ordered Routing Algorithm
WRP	Wireless Routing Protocol
ZRP	Zone Routing Protocol
CEDAR	Core Extraction Distributed Ad Hoc Routing Algorithm
GSR	Global State Routing
NS	Network Simulator
IERP	Internet-Zone Protocol
RREQ	Rout Request
LMS	Location Management System
TCP	Transmission Control Protocol
RTT	Round Trip Time

Appendix B

VITA

Savyasachi Samal was born in India on November 17, 1977, in in a small town called Udala.. He received a Bachelor of Technology Degree in Computer Science and Engineering from Regional Engineering Warangle India in May, 1999. In August, 2000, he enrolled in the graduate program at Virginia Tech to pursue a Master of Science in Computer Science Department. His focus is on networks with a special interest towards mobile networks. Savyasachi will be working for the Research Division Administration of Virginia Tech, where he will be working as a Database and Application Specialist.