An Implementation of a Parasitic Routing Algorithm

Eoin Bailey

A dissertation submitted to the University of Dublin in partial fulfillment of the requirements for the degree of Master of Science in Computer Science

September 13th 2004

Declaration

I declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed: _____

Eoin Bailey

Date: September 13th 2004

Permission to lend and/or copy

I agree that Trinity College Library may lend or copy this dissertation upon request.

Signed: _____

Eoin Bailey

Date: September 13th 2004

Acknowledgments

I would like to thank Mads Haahr, my supervisor for his help and support, along with the whole class of NDS 2004, the year was an enjoyable and memorable one. I would like to take this opportunity to thank both Larry Page and Sergey Brin also, without their work and dedication this dissertation would have been significantly more laborious. Finally, my family deserves some mention, they were all great, and helped when it was required (basically through supplying me with coffee!).

Abstract

Portable personal computers with low-power requirements are fast becoming a necessity as access to up to date information is required by users no matter what their location. Unfortunately the ability to deliver this information between disjoined users is not as advanced. Mobile ad-hoc networks offer a partial solution to this problem, allowing users to transmit information through intermediate nodes; however the protocols that are currently favoured in these networks rely on constant end-to-end connectivity.

If a mobile ad-hoc network consists of few nodes relative to the area it covers and consists of low-power devices i.e. devices with short transmission ranges, the network will likely not fulfill the current model and will consist of a number of partitioned networks or isolated nodes. This dissertation addresses this scenario and proposes a solution that makes use of the changing network topology, using the nodes movements as a help and not a hindrance.

Simulations results included in this dissertation evaluate the implementation and compare a number of different scenarios including a simulation making use of a 'traditional' ad-hoc routing protocol: Dynamic Source Routing.

Contents

List of Figures

Chapter 1

Introduction

1.1 Motivation

The advent of small, inexpensive, computing devices has brought with it a greater need for computer devices to exchange information. These devices range from sensing devices that need to collate the data collected by individual sensors to PDA's that computer users frequently rely upon as a means of staying connected, to full-size laptops allowing increasingly complex tasks to be achieved at faster speeds. The devices mentioned here all have the potential to be mobile; it would be expected that sensors and PDA's are mobile; whereas users were previously restricted to large stationary workstations. These same users now demand a connectivity from these devices that was previously unheard of, wanting a continuous update of information to ensure that important

information is not missed. The current communication methods are unable to fulfill all of these requirements, as such a new method is essential.

Sensor networks are also in need of a better communication model, in these environments there are two extremes: the first is a sparsely populated sensor network, consisting of few sensors e.g. tags attached to animals to track their movements. The second is a more densely populated sensor network, consisting of a large number of extremely small devices with very short range transmission capabilities; these networks may be used to monitor tidal flows in an ocean or various metrics about the weather for example. Often sensors must be recovered to retrieve the information stored within them, in this scenario a lost sensor means that all the information it had obtained is also lost. These sensors are often not equipped with wireless transmission capabilities as the power requirements are prohibitive for long-range transmissions, however short-range transmissions are viable.

Mobile devices are now being equipped with wireless connectivity as standard [1] [2] allowing for, potentially, quick, easy, temporary connections to be made between them. This allows for the creation of a network known as an ad hoc network.

Regardless of the device or the specific information being exchanged a means to transfer the bits is required, conventional wisdom asserts that a complete end-to-end connection is essential to this process. In the scenarios outlined previously it is unlikely that these connections can be found, or indeed

will exist. As such a new model for transmitting this information is required. This dissertation details such a model, it is known as parasitic routing.

1.2 Ad Hoc Networks

The Internet Engineering Task Force (IETF) have a working group which specifically deals with Mobile Ad hoc Networks (MANETs), this working group defines a MANET as:

> "A 'mobile ad hoc network' (MANET) is an autonomous system of mobile routers (and associated hosts) connected by wireless-links – the union of which form an arbitrary graph. The routers are free to move randomly and organize themselves arbitrarily; thus, the network topology may change rapidly and unpredictably. Such a network may operate in a standalone fashion, or may be connected to the larger Internet" [3]

The above definition describes the technical aspect of ad hoc networks, however it does not describe the scenarios in which they can be used, these include though are not limited to, military situations, disaster areas, sensor networks, essentially any environment where there is no fixed communication infrastructure or to which there is no access to said infrastructure.

Ah hoc networks have their roots in a 1972 Department of Defense sponsored project known as Packet Radio Network (PRNet), which then became the Survivable Radio Network (SURAN) in the 1980s [4]. It was not until the early 1990s with notebook computers and viable communications based on RF and infrared that the idea of infrastructureless mobile hosts was proposed [5], and the IEEE 802.11 working group adopted the phrase “ad hoc networks”.

An example of a MANET can be seen in figure 1.1. In the figure, initially it is possible for node A and node C to communicate via node B. It is not possible for either node A to communicate directly with node C or at all with node D, similarly node B cannot communicate with node D, nor can node C. However, when node B moves to position ‘B1’ node C and node D can now communicate via node B. Obviously in the scenario below if only node B is capable of movement there will never be a completely connected network, i.e. all four nodes will never be able to communicate simultaneously. In the example below it is also unlikely that node D will be able to communicate with node A at any point if a pro-active routing protocol is used, though it should be noted that for this specific simple example that a reactive routing protocol may deliver some messages depending on the timeouts for the routes. The differences between reactive and pro-active routing protocols will be explained further down in this document. In figure 1.1 the transmission of data from node A to node B requires an intermediate transmission to node B, this is known as a multi-hop environment.

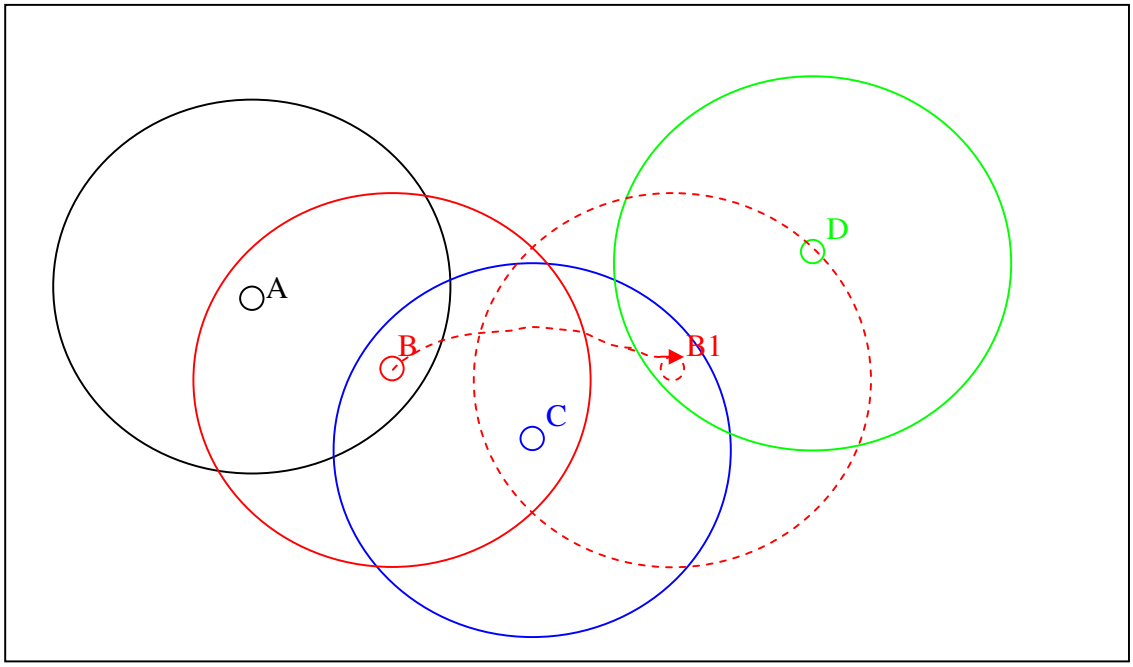


Figure 1.1 A Simple Mobile Ad Hoc Network

1.2.1 Attributes of Ad Hoc Networks

As previously discussed an ad hoc network consists of an arbitrary number of nodes that group together to form a network. Ad hoc networks are temporary and transient as their name suggests, as such nodes are free to come and go from these networks as they please. All nodes within a MANET are equal in the sense that each node is a router for all other nodes as such there is no central authority or deliberate hub in a MANET. However a node may inadvertently become a hub through which a large majority of messages pass should it be the only connection between two node clusters.

The capacity a MANET is capable of is constrained by the mutual interference of concurrent transmissions between nodes. As such if all the nodes

in a MANET are capable of long range transmission, the interference between concurrent transmissions will increase. Therefore to improve the overall capacity of a MANET constraining the range that nodes are capable of transmitting should increase the number of nodes capable of transmitting concurrently. In doing this however the number of connections between nodes is also likely to fall. In [6] the authors describe a test involving the lowering of a nodes transmission range, and compare the throughput to the original range. What they found was that provided that the nodes are mobile relative to each other that the overall throughput increased, i.e. that mobility can increase a nodes throughput. A paper by the same authors as [6] and S. N. Diggavi in 2003 [7] provides an in-depth mathematical analysis proving that even one-dimensional mobility within the network increases the capacity.

MANETs rely on wireless communication, which despite ever increasing speeds is not, nor is it likely to be, on a par with wired communications. Thus the sending of extraneous information needs to be kept to a minimum; this information may be repeated messages or control information, such as the route discovery and route maintenance messages for various ad hoc routing protocols. Though the newest network speeds for wireless know exceed 100Mb/s, this has yet to be standardized, despite a standard for 54Mb/s [8] the current speed in most widespread use is 11Mb/s [9], well below the 100Mb/s [10] LAN speed in common use.

Also worth noting is that the devices that make up an ad hoc network are often constrained by energy concerns. The devices tend to have an exhaustible

energy supply, this is another reason to limit the transmission range, as this is an excellent way to conserve energy. Devices that users carry with them can be charged, though most users of mobile devices will want them to last as long as possible on a single charge, however certain devices may not easily have their energy supplies replenished, taking as an example: sensors in a sensor network designed to track an animals movements. It is infeasible to continually recharge the power supply on such a device therefore the number of transmissions and the range over which data is required to be sent needs to be controlled.

1.2.2 Routing Protocols

Every member of a MANET is a device that can and will forward messages on behalf of any other members of the network. The decision making process that each node uses to decide where to forward a message is also known as the routing protocol.

Routing protocols are classified into two distinct categories, pro-active and reactive. Pro-active protocols attempt to track every change in a network's topology, this approach allows for instant transmission of a packet as the route is already known at each node. OSPF is an example of a proactive routing protocol for wired networks, which was tested in MANETs and failed due to the excessive overhead it required to maintain its records. MANET specific examples include DSDV [11] and ZHLS [12].

Reactive protocols differ from pro-active as they only attempt to determine a route when one is needed or “on-demand” only. When a route is required a search mechanism is initiated and propagated throughout the network in an attempt to acquire a route. MANET specific routing protocols of this type include AODV [13] and DSR [14].

Pro-active protocols work best in a network where the topology is either constant or slowly changing, they perform poorly if the topology is changing rapidly as well is in EMCON sensor networks (as only one way communication is permitted). This restriction makes protocols of this type a poor choice for military networks [15]. Reactive protocols while performing considerably better than pro-active protocols in a changing network environment introduce additional latency and are inappropriate for static networks or for cluster-based topologies due to the overhead of route discovery.

Both categories of routing protocols, reactive and pro-active, share a commonality: they are both “store-and-forward” protocols. This is the traditional model for routing protocols, working in a similar manner to the Internet at large in a general sense. Hybrid protocols also exist, such as ZRP [16], which uses proactive routing locally and reactive routing globally in an attempt to achieve higher levels of efficiency and scalability.

Neither reactive nor pro-active routing protocols deal adequately with the situation where the MANET consists of a number of nodes spread out over a large area, i.e. a sparse network environment. In this scenario the nodes are

mobile and are not often within transmission range of more than a single other node at any time, and may not be in transmission range of any at all.

1.2.3 Uses of Ad Hoc Networks

Ad hoc networks are used in environments where infrastructure is either non-existent or not accessible; these may include disaster areas, military situations, or sensor networks. However the use that may prove to be the killer application for ad hoc networks is conferencing. A group of people come together at a conference, planned or otherwise, and wish to exchange information, this could be achieved quickly and easily through an ad hoc network. Even if a fixed infrastructure is available, it may be quicker and indeed easier to form an ad hoc network to share files, exchange e-mail, and communicate, as ad hoc networks generally do not require authentication or verification from a central authority to allow access, unlike conventional wired and wireless networks.

Free access to a MANET is however a double-edged sword, allowing users to come and go as they please creates a serious security concern. If users are not required to register to access the network, any Tom, Dick, or Harry can gain access to every other user's mobile device. These issues [17] are on top of all of the security problems that already plague conventional wireless networks [18].

Sensor networks are a potential killer app, not for general users of MANETs, but for the scientific community. Sensor networks can be designed to

be self-organising, self-repairing, autonomous devices. These features would allow for incremental deployment of the networks and self-assembly without a central administration, while also providing the capability to dynamically adapt to failures or network degradation. Potentially the uses for such networks are extremely broad, including agriculture, manufacturing and inventory tracking, health care, environmental tracking, surveillance and security, and animal research. Part of the reason for so broad a range of uses is the varying uses a sensor can have (acoustic, thermal, visual, radar, magnetic, etc…) coupled with the wide variety of conditions that can potentially be monitored (humidity, location, noise levels, temperature etc…).

1.3 Objectives

Upon completion of this dissertation the following objectives hope to have been achieved:

1. A survey of the general area surrounding ad hoc networks, with specific attention paid to routing protocols that follow a “store-carry-and-forward” paradigm.

2. Design of an algorithm for a routing protocol that follows the “store-carry-and-forward” paradigm for an ad hoc network, this routing protocol will heretofore be called a parasitic routing algorithm.

3. Implement the algorithm and create a number of simulation environments to verify the correctness of the algorithm.

4. Run a number of simulations using the parasitic routing algorithm as the routing protocol within the ad hoc network. These simulations will consist of a number of different mobility patterns and traffic patterns in an attempt to cover as broad a range of scenarios as possible to verify that the algorithm works in each and to compare determine which scenario the algorithm is most suited to.

5. Recommend improvements that could be made to future versions of the algorithm.

1.4 Document Structure

In an effort to make this document as easy to follow and understand as possible this section will detail the overall structure of this dissertation and give a brief description of each section.

Chapter 1: Introduction discusses the motivation behind the Parasitic routing algorithm, introduces the area on which it is based and will be applied, and details the objectives.

Chapter 2: State of the Art addresses the area of store-carry-and-forward style routing algorithms and their implementations.

Chapter 3: Analysis and Design describes the different options that could be used in the algorithm, and details the final choice.

Chapter 4: Implementation and Environment looks at the specifics of the implementation and the environment test-bed that is used.

Chapter 5: Simulation Results and Evaluation discusses the final outcome of the various simulations and the impact on the results.

Chapter 6: Conclusion outlines potential improvements that could be made and winds up the project.

Chapter 2

State of the Art

In recent years there have been a large number of different routing algorithms proposed for ad hoc networks, a significant portion of which are not even implemented. As ad hoc networks developed out of conventional networks, the routing algorithms proposed have had a natural tendency to be based on similar assumptions to these networks. In traditional wired networks, such as that shown in figure 2.1, if device A attempts to send a packet to device B, the router will forward on that packet, device B will receive it and presumably acknowledge that receipt. If device A was then to attempt to send a packet to another device, call it device C (not shown in the figure), located on a different domain about which the router has no knowledge, the router will attempt to discover a route, however if no route is found, the message will not be sent.

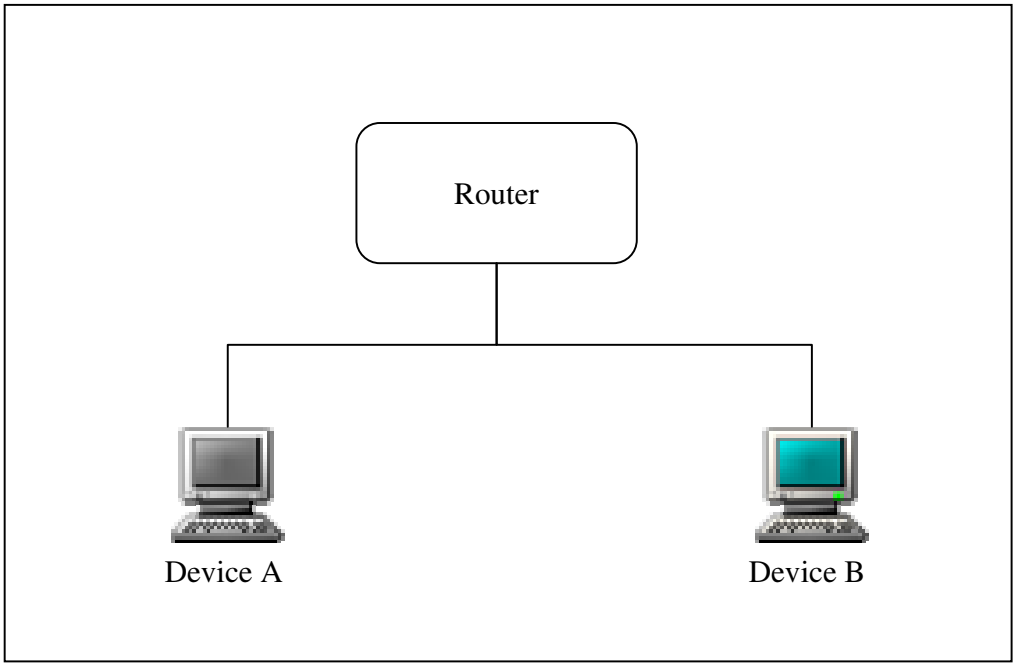


Figure 2.1: Example of Wired Connectivity

The same is true for wireless networks which make use of a base station, if no route can be found to the destination device the message will not be sent. Most ad hoc routing protocols make similar assumptions, attempting to discover a route if one is not known and if one cannot be found the message is not sent.

Recently there have been a number of attempts to overcome this problem, the methods and results of these will be discussed in this section of this document.

2.1 Position-Based Routing

Position-based routing uses, as the name implies, the location of nodes to determine a route to the destination. The distinction drawn in [19] is that proactive, reactive and hybrid protocols are “topology-based”, meaning that they

make use of information about links stored at nodes to perform packet forwarding, whereas position-based protocols do not have the necessity of storing routing tables and make forwarding decisions based on the destination's position and the position of a node's neighbours.

In order for a node who wishes to send a packet to do so, it must know the location of the destination node. This is achieved through a *location service*. A location service can be hosted by some of the nodes in the network or all the nodes in the network, furthermore each of these services may contain position on some of the nodes or all of the nodes, making four different possible services available: some-for-some, some-for-all, all-for-some and all-for-all. For ad hoc networks it is advisable to have an all-for-all approach as there are no guarantees that the nodes running the location service will be available at a given time while simultaneously a node would not necessarily know which node was a location service.

A packet is forwarded to a neighbour if that neighbour is in the direction of the destination relative to the sender. A packet may be forwarded to a single neighbour, e.g. GPSR [20], or may be forwarded to multiple neighbours, e.g. DREAM [21]. The addition of hierarchies into the forwarding decision can help in the scalability of a system such as this. If packets are forwarded to a neighbour only if that neighbour is closer to the destination then an obvious failure can occur, as highlighted in figure 2.2 below. The sender will not forward the packet to its only neighbour, despite the presence of a complete route.

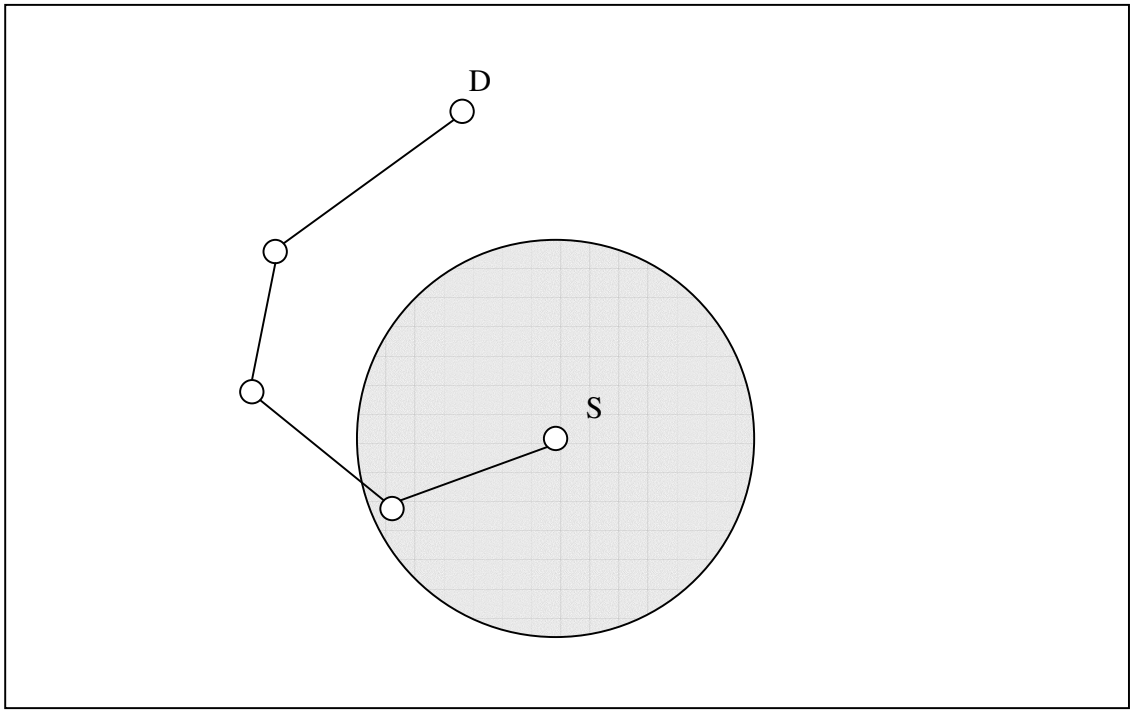


Figure 2.2: Failure of Position-Based Routing

The overhead required to maintain a table with information on the position of each node in the network is not trivial, and as such whether this approach is more efficient than DSR, which is widely accepted as the 'best' protocol currently available [22], is questionable. Results, obtained by the authors of GPSR, have shown that GPSR delivers more messages than DSR, however in their scenarios it was assumed that the nodes knew the positions of every other node, removing the requirement to determine the location of the destination.

An out-of-band means to determine the location of all the other nodes is not viable as it would not be possible in the scenarios where ad hoc networks are most useful, therefore, whether this method is more efficient is doubtful. However if a location service was viable a position-based routing protocol may be modified to function as a store-carry-and-forward protocol with relative ease.

2.2 Epidemic Routing

Epidemic routing is the name given to flooding the network with each packet. This name first appears in a paper by Vahdat and Becker [23]. The view put forward in their paper is that given unlimited resources and unbound time constraints, every message will eventually be delivered and the delivery can be guaranteed to be in the shortest possible amount of time.

The scenario put forward is one in which there is a partially-connected ad hoc network, as shown in figure 2.3 below.

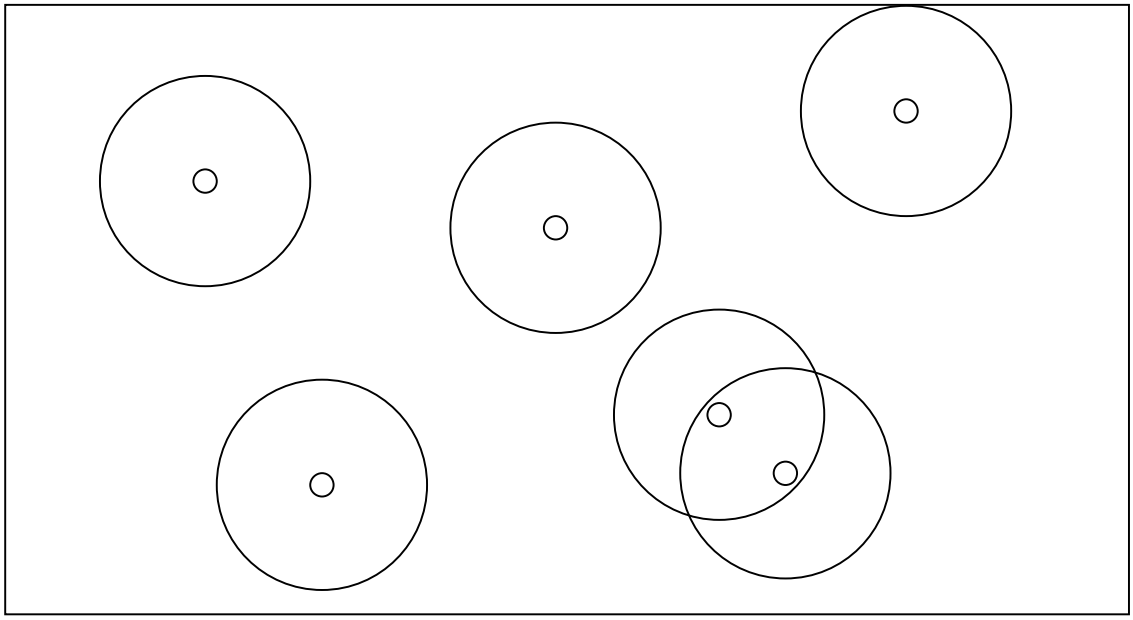


Figure 2.3: A Sparse Ad Hoc Network Environment

The nodes in the network move in an arbitrary fashion, this is necessary for the algorithm to work, otherwise messages are not exchanged.

This is the first example of a store-carry-and-forward algorithm in which tests have been carried out in a scenario similar to that in which parasitic routing will eventually be tested. Epidemic routing is in essence the flooding of every message to every node in the network. This is achieved in the following manner: when two nodes move within communication range of each other they synchronize the messages stored on each other, i.e. a message found on node one but not on node two is copied from node one to node two. This occurs for every message at every encounter. In this way a message is guaranteed to reach its destination in the shortest time possible as it is guaranteed to take the shortest route. However, the message will also be guaranteed to have taken every other potential route, and in fact will have been transferred to nodes that have no chance of ever delivering the message.

Obviously in a real world implementation nodes cannot have unlimited resources, it is infeasible to give a node a buffer large enough to store every message transmitted during the lifetime of the network. For this reason an improvement to the system was put forward [24]. In this paper various queue methods were employed in an attempt to determine which of these methods ensures the highest number of messages are delivered to their destination. Four different queue styles are discussed: Drop-Random, Drop-Least-Recently-Received, Drop-Oldest, and Drop-Least-Encountered. Using drop-random a packet is chosen at random to be replaced, this method can be most equitable, punishing nodes that create a large percentage of the traffic. Drop-least-recently-received drops the packet that has been stored in the buffer for longest,

the basis for this is that these packets are most likely to have been forwarded on to other nodes. Drop-oldest drops the packet that has been in the network globally the longest, the assumption being that these packets are the ones most likely to have been delivered. Finally drop-least-encountered, this is an adaptive strategy that drops packets based on the estimated likelihood of delivery.

Using a number of metrics to compare the drop strategies it was found that drop-oldest and drop-least-encountered performed best and similarly.

They also attempt to limit the number of times a packet is forwarded by making forwarding decisions based on whether a node is likely to deliver the message.

2.3 DAKNET

Daknet [25], (dak meaning “post”, i.e. post-network) is a system whereby messages are ferried between nodes via a mobile node which travels along a known route. The system was designed to be used in rural areas and is in use in India connecting isolated villages. Daknet was conceived as a cheap means to allow communication between rural villages, figure 2.4 below shows the layout of the system.

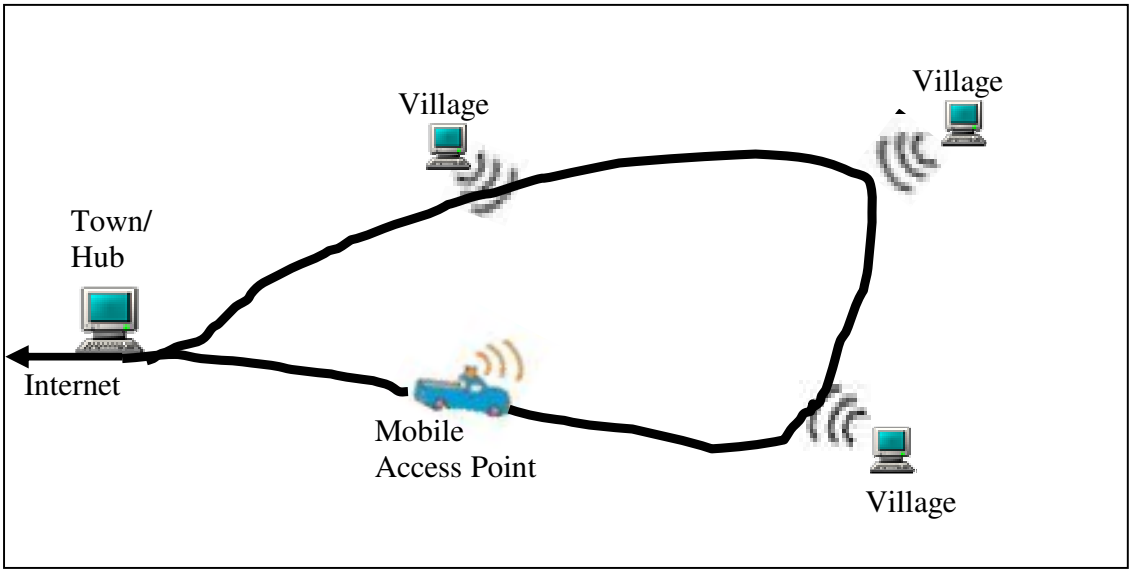


Figure 2.4: DakNet network architecture

As the mobile access point moves around the pre-ordained path, it comes within transmission range of the workstations at each village. Messages are exchanged in both directions, allowing for the delivery of messages to the workstation and the sending of new messages out. This allows for the people to communicate relatively quickly and easily between villages. Users can also request information from the internet, which is transferred in the same way. This system has been implemented and is in use although in certain cases the mobile access point is a mule!

A system that works in a similar manner is described in [26], the protocol is called message ferrying. The workings are similar, except that nodes can proactively move to deliver messages also. Nodes may be similar to the mobile access point above and move along pre-destined paths that do not change, except in this case they may not be designed to pass by all the other nodes. Nodes may also move in a pattern specifically designed to promote message

delivery. The example given in the paper is one where robots are deployed in a disaster area, and one or more robots may move in a pattern chosen to enhance delivery rates of messages.

2.4 Summary

Little work has been done on the area surrounding store-carry-and-forward algorithms for ad hoc networks, particularly when compared with the work done on protocols such as AODV, DSDV, or DSR particularly. As such the area is still highly experimental and ideas that at first seem likely to solve all the problems throw up more than originally encountered (the case with position-based systems and the location service, is this a more complex problem than the original routing problem?)

Chapter 3

Analysis and Design

This section will discuss the various different options open to a store-carry-and-forward algorithm and also the positives and negatives of these various options. The general overview of a store-carry-and-forward algorithm will first be discussed, followed by an in-depth analysis of each individual component.

3.1 Store-Carry-and-Forward Paradigm

A store-carry-and-forward style algorithm is one that makes use of the movements of the nodes within an ad hoc network. If an ad hoc network is visualized as a ring of people playing the game of 'chinese whispers', these people are all sitting close enough together that they do not need to move to communicate with the neighbour on either side. It is possible for a message to pass around the network immediately. However, if there is a break in the ring,

i.e. if two people are sitting at a distance apart whereby they cannot communicate with each other, in order for the message to be passed along, one of the two people must move location, towards the other, then once within communication range, the message is passed on. The message is initially stored by the person upon hearing it, they then carry the message while they move to the next person, the message is then retold to the neighbour, i.e. forwarding it on, this is a simple analogy of a store-carry-and-forward system. Though it is unlikely that a human being would be the node for transmitting the information, a person could easily have a small computer device on them that would enable this system to work for data and not just spoken words.

The major difference between the store-carry-and-forward paradigm and the store-and-forward paradigm employed in the majority of routing protocols is that the former takes advantage of one more resource of a node, its mobility. Store-and-forward protocols take advantage of an agent's buffer, transmission capabilities, some processing time, and some energy, while store-carry-and-forward protocols use all the previous, and use a nodes mobility to enhance the likelihood of delivery, in doing so a message will most likely be held in an agent's buffer, or queue, for an extended period of time also. The mobility may be random, designed in advance to deliver messages, or may be dictated on-the-fly in an attempt to ensure speedy delivery of all messages.

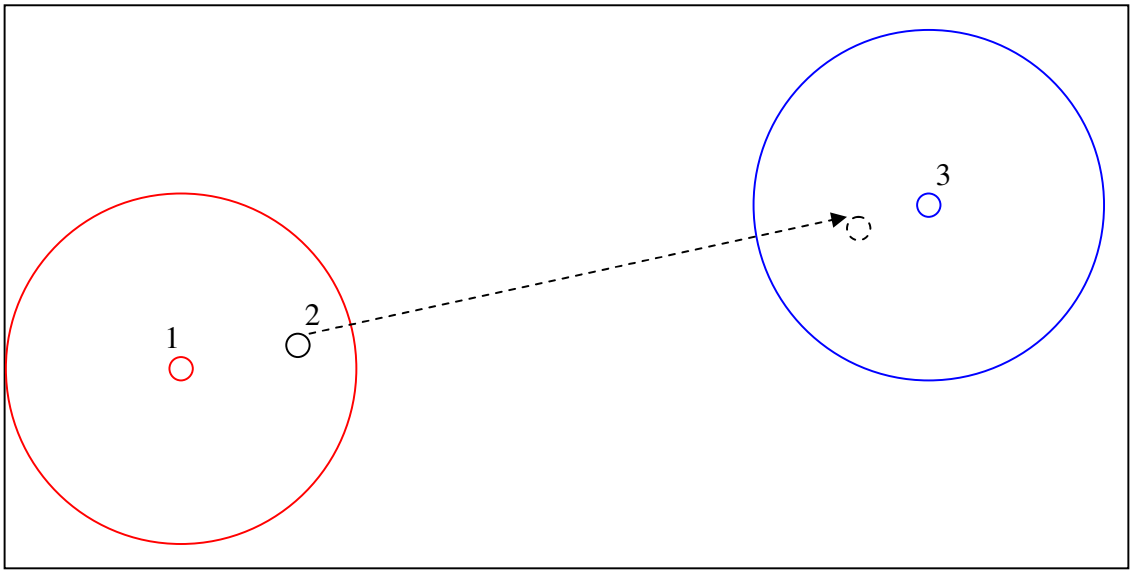


Figure 3.1: Store-carry-and-forward paradigm

Figure 3.1 above is an example of the simplest case of store-carry-and-forward: node 1 communicates a message to node 2 that is eventually destined for node 3. The decision to forward the packet to node 2 is one of the key areas of parasitic routing. Node 2 then moves to the destination shown in the figure, carrying with it the message from node 1 for node 3. Upon entering transmission range of node 3, the message is delivered.

3.2 Analyzing Store-Carry-and-Forward

Figure 3.2 below shows an overview of store-carry-and-forward routing.

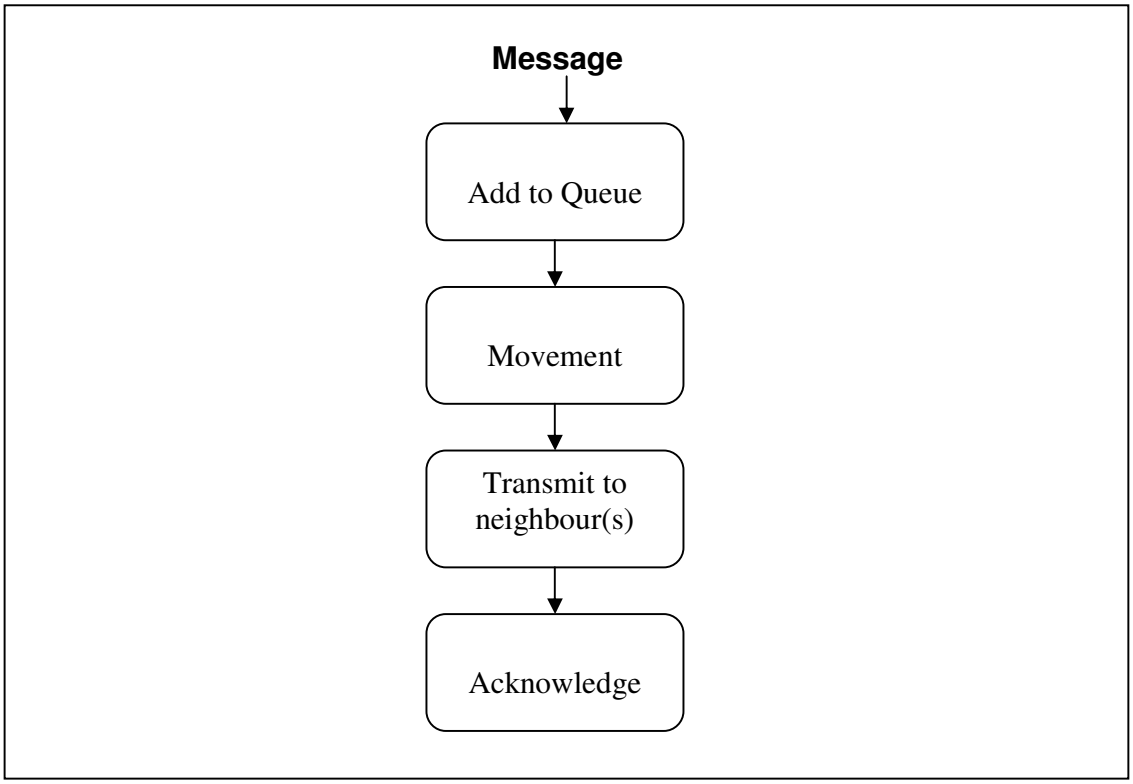


Figure 3.2: Store-Carry-and-Forward Algorithm Overview

For each section shown in figure 3.2 there are a number of possible approaches that can be taken, a number of each will be discussed in the sections that follow.

3.2.1 Queue Management

The size of the queue that each node has will most likely be constrained by the resources available to it, though this may seem obvious, it is an important observation nonetheless. Despite this, the size of the queue is still an issue, the larger a queue is, the more messages that it is capable of storing, this will lead to

fewer messages being dropped, which in turn should lead to higher rates of delivery.

In a system where messages must be dropped to cater for new incoming messages a drop strategy must be employed. A number of different drop strategies will be outlined in this section of the document, including: drop precedence, drop-arrivals, and various drop-from-queue or pushout mechanisms.

Drop precedence works on the basis that each message is assigned a level of importance, or precedence, when a message arrives at a congested queue messages of lowest precedence are dropped first, whether they are the incoming messages, or messages already stored in the queue. Precedence levels can be assigned at the node that originated the message, this method can be used to ensure forwarding of the message takes place and that a message is not dropped at any location, however a malicious node would be able to abuse the system and send all of it's packets at a high precedence level. Precedence levels may be set at each receiving node also. This would allow for per-hop behaviour to be controlled, a possible approach for this would be to decrease the precedence level at every hop, in a similar fashion to a TTL value, except in this case once the lowest level is reached the message may still be forwarded provided there is either no need to drop it, or there are messages also at the same precedence level, in this case a message would most likely be chosen at random from the messages at the same precedence level, to be dropped.

Drop-arrivals is a simple system whereby once a queue becomes full, incoming messages are always dropped. In an ad hoc network environment that

is making use of a store-carry-and-forward routing protocol nodes are not in constant communication, and it cannot be said that a node will necessarily have a neighbour at a given moment in time, as such, if a node is creating messages to deliver, the queue will be filling with messages that originated at that same node. In the case where drop-arrivals is used, incoming messages will not be accepted, thus messages do not propagate throughout the network and will therefore not be delivered. This is a worst case scenario, though in a sparse network environment, nodes can easily become isolated for an arbitrary length of time, allowing this situation to develop.

Drop-from-queue, sometimes called pushout, methods are most likely to succeed in a PR environment, as they allow for fair dropping of messages within an ad hoc network. There are a number of generic pushout methods including drop-random and drop-oldest. Drop-random is a highly simplistic approach, in which a message is chosen at random to be dropped to make room for an incoming message. This approach is viewed as highly fair, punishing nodes that tend to create a significant portion of the messages in the network as they are more likely to have a message dropped. Drop-oldest can work in two ways, the term ‘oldest’ can be applied globally dropping the oldest message within the network (the globally-oldest message stored on the node in question) or it can apply in a local sense, dropping the oldest message received, i.e. if messages are ordered temporally based on the time received at a node, the last message in the ordering would be dropped. A queuing method that is specific to a PR environment is drop-least-encountered. This method is also a pushout queuing

style, and works as follows: if a node has encountered a node at some time in the past an assumption is made that those two nodes are likely to meet again at some time in the future. Thus if a node has not encountered a node for which it has a message in it's queue it is assumed that message is less likely to be delivered than messages destined for nodes that the host node has encountered in the past, making that message the one to be dropped.

A pushout queuing method is likely to achieve the best results for a PR protocol than either drop-arrivals or drop precedence. The queuing method that will be implemented will be a drop-oldest approach. Although drop-least-encountered seems like the logical choice, provided the decision to forward a packet to a neighbouring node is made correctly, a node is unlikely to receive and store messages that are not likely to be delivered, thus making the extra check redundant.

3.2.2 Mobility Patterns

Mobility patterns are used to describe the general movement of the nodes within the topology of the network and may be separated into three separate categories, random, pre-conceived, and repeating patterns. Each will be dealt with briefly in this section of the document.

Though it seems oxymoronic to describe random movement as a pattern, in this particular case 'pattern' is describing the style and not the particular movements. As the name implies the nodes movements are random, each node

moving in a random direction at a random speed, and pausing for a random amount of time before choosing a new heading and speed. As the definition of an ad hoc network states nodes move in an arbitrary fashion, random motion stays truest to that meaning. Whether nodes actually move in a random fashion within an ad hoc network depends on the scenario, in the situation where nodes are PDAs used by human beings, a discernible pattern would likely emerge if the movements were observed for a length of time.

Pre-conceived or pre-chosen patterns are movement patterns that are chosen and dictated in advance of the network forming, for example the mobile access points used in DakNet. Patterns of this nature are less likely within an ad hoc network, and whether this is in fact ad hoc networking is in question, as it requires co-ordination of all the nodes' locations in advance.

Repeating, or emerging, patterns are movements that occur on numerous occasions within the network. If nodes have the ability to learn that these routes exist the system can work in a manner similar to systems using pre-conceived patterns, as the nodes will be able to make predictions on the future locations of nodes. Animals roaming within their territory is a good example of a pattern in this category, as animals tend to follow particular patterns, dictated by the resources in their territory at different seasons. A second example is the movement of people particularly within an urban environment as movements are constrained by structures, rivers, etc… and people will tend to travel along certain routes on a day-to-day basis.

Random patterns should be used as a base to compare the effectiveness, or ineffectiveness of other movement patterns. This should allow an even and fair analysis of the pattern in question. Emerging mobility patterns should be most effective in a PR environment; however, random mobility patterns should also be experimented with for the stated reason of comparison.

3.2.3 Forwarding Strategies

When two nodes come within transmission range of each other and one or both contain undelivered messages in their buffers a decision has to be made regarding which messages should be exchanged. All messages can be exchanged, a limited number may be exchanged, or messages may be exchanged based on a number of other metrics, including position or likelihood of contact with destination node. The acceptance of messages should go hand-in-hand with the queue type being used, in other words if the queue type in question will not allow any messages to be overwritten, then no messages should be transmitted, as it would be a waste of the nodes limited resources.

The forwarding of all messages to every node in the network is an excellent idea to ensure the swiftest delivery possible, however, each node must be capable of storing every message that is sent within the network, which is unlikely, and unrealistic in an environment where the network is long-lived.

A limited number of messages may be exchanged, this limit can be an arbitrary choice set at each node, or a global constant, in this way a node wishing

to transfer a large number of messages would have to send them to a number of different nodes, which should ensure that a node does not have its message buffer overwritten on these encounters. The limit may be proportional to the size of the queue or to the number of messages a node wishes to send.

A decision to forward a message may be made based on the position of the neighbour node. If the node lies in the direction of the destination node, the message will be transmitted, if in transmitting the message it will now be further away from the destination node the message should not be transmitted. There are a number of problems with this method, including the case in which the only route available starts by transmitting away from the destination, which was discussed previously. There is an added problem when a sparse mobile network is the environment; initially a node may not be a choice to transmit to as it is further away from the destination than the sending node, but its movements may take it closer. For this to be taken into consideration, the direction the node is moving needs to be examined and it may still fail (even if eventually the path the node takes will pass right by the destination) if initially it is not moving in the right direction.

The final forwarding strategy examined is one in which the decision to transmit is based on a node's previous encounters. A node keeps track of nodes with which it has come within transmission range; this information is in effect a routing table. Presence of a node in the table does not guarantee a future contact but does indicate a likelihood of a future contact. This strategy should

work most efficiently in networks where an emerging or pre-conceived mobility pattern is used.

3.2.4 Acknowledgments

When a message is delivered to the destination, to minimize the wasted bandwidth and buffer space by the continued propagation of the message through the network, a system that acknowledges the receipt of a message should be in place. Acknowledgements may be spread through the network in a similar manner to the spreading of messages, using the same queuing algorithm. If a node receives an acknowledgment for a message that it has stored in its queue the message should be removed from the queue to free space and the acknowledgement stored also to inform other nodes of the delivery.

3.3 Final Solution

This implementation will make use of a drop-oldest queuing algorithm in conjunction with a message forwarding scheme using previous encounter information. Also implemented will be a version of the epidemic routing algorithm previously discussed in an effort to introduce a comparison between the two. Both algorithms will also be implemented with and without acknowledgments to determine the impact, if any, they have on the effectiveness of the algorithms.

3.3.1 Algorithm

Figure 3.3 outlines the decision process for message forwarding for the PR algorithm.

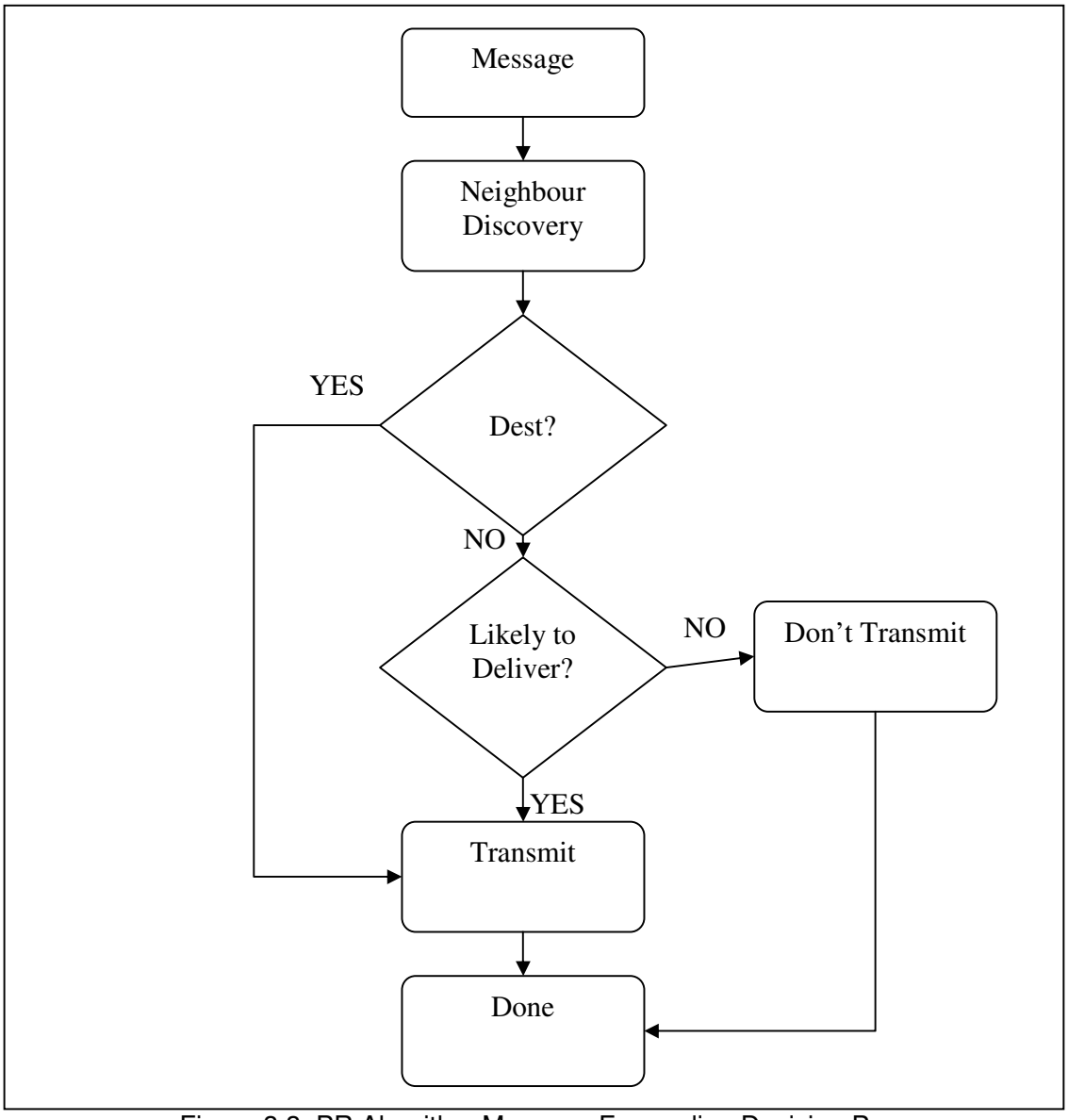


Figure 3.3: PR Algorithm Message Forwarding Decision Process

The extra step of deciding whether a node is 'likely to deliver' a message is the distinction between PR and epidemic routing, in epidemic routing that step does not exist, the message is transmitted regardless.

Figure 3.4 outlines the process used to spread acknowledgments throughout the system.

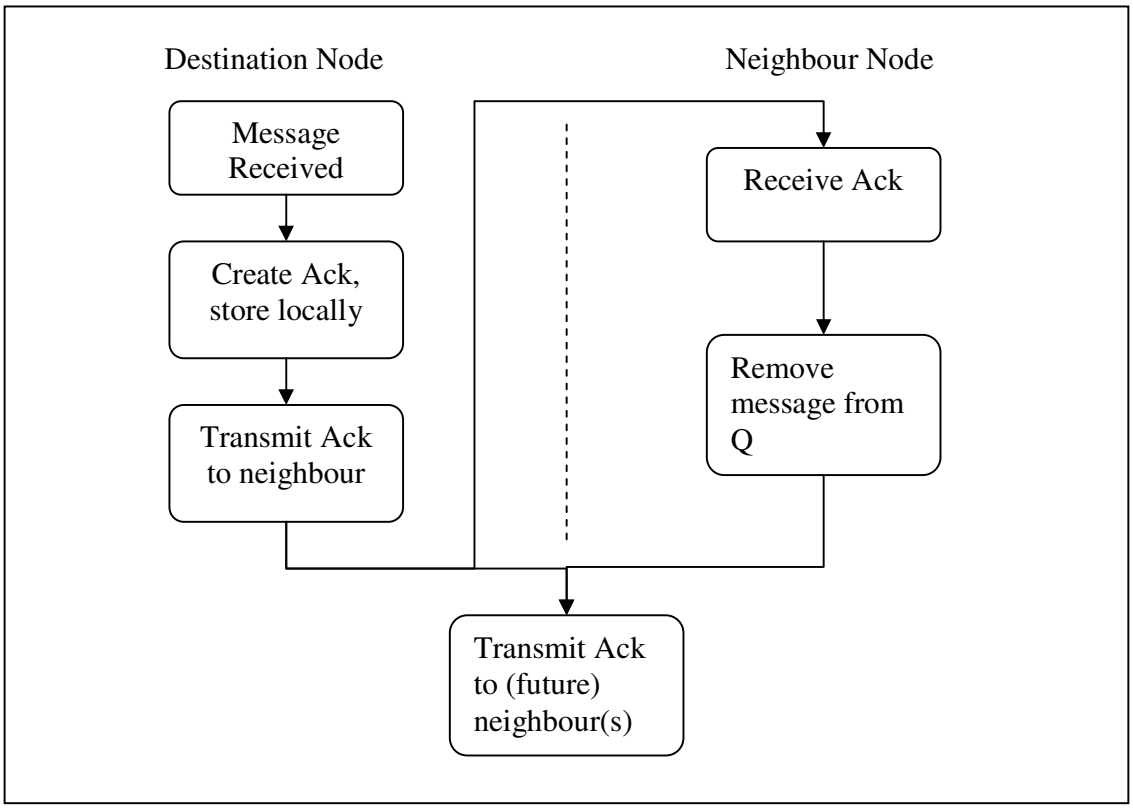


Figure 3.4: Creation and Propagation of Acknowledgments

The method by which nodes will make the decision to forward a message is based on whether there is information stored regarding that node in the table of 'encountered nodes'. An example of this table being filled is shown in figure 3.5. As can be seen in the table that accompanies figure 3.5 initially (part 1) only nodes A and B are within range of each other and as such are the only two

nodes to have any information stored on potential routes. In part 2 node B has migrated to beside node C, and they now exchange information. Node C also makes a note of node A in its tables, despite only encountering node B. This is because node C assumes that node B can be used as an intermediary to transmit a message to node A in the future. In part 3 a similar exchange takes place, with node D noting that it is aware of a potential route to A, B, and C.

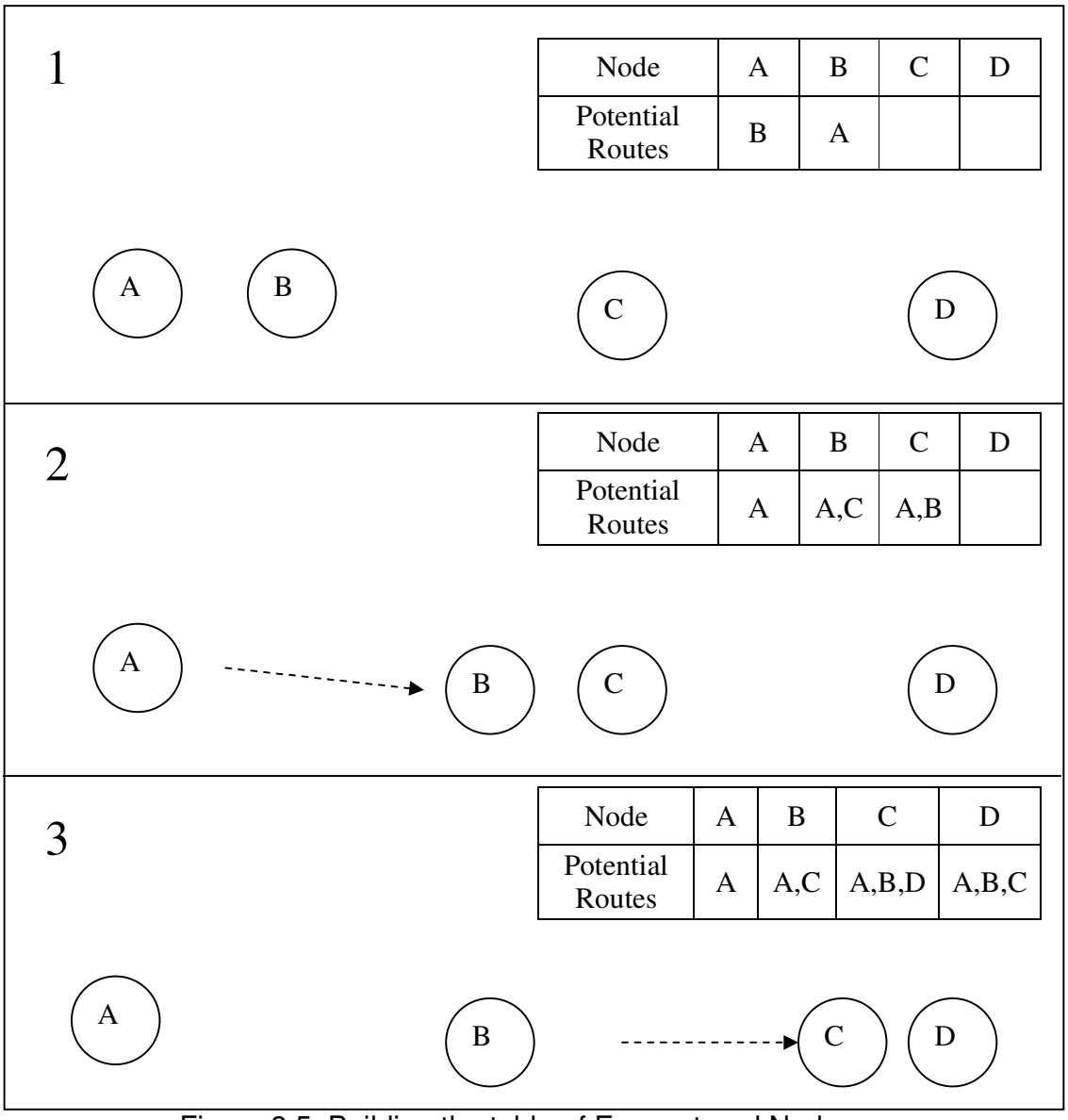


Figure 3.5: Building the table of Encountered Nodes

Chapter 4

Implementation and Environment

This section will deal with the implementation of the PR algorithm, discussed and outlined previously in Chapter 3, and the environment that will be used as a test-bed. Initially an implementation for use on a PDA or laptop equipped with a wireless device, most likely Bluetooth, was discussed; however the challenge of this approach was not the implementation, but the verification upon completion of the coding. Small-scale testing would not be an issue, but to ensure a full round-up of test scenarios users of PDAs and laptops would need to be convinced to install the protocol also and make use of it, time constraints unfortunately ruled this out.

The choice of simulators is quite varied, however, a majority of the simulators are geared towards a specific style of network or testing, e.g. Gossip, a QoS IP simulator, NS-2 is the accepted standard network simulator, and has the support of SUN, DARPA, Xerox PARC along with numerous contributions

from various researchers the world over. NS-2 will be used to simulate an ad hoc network within which the nodes will make use of PR. Problems encountered during the implementation and testing of the PR algorithm will also be discussed in the following sections.

4.1 Implementation

In an effort to simplify the implementation process the algorithm was broken down into separate sections, each section having a specific task. Figure 3.3 in the previous chapter outlines the algorithm however, within each of the elements depicted there exists subtasks. Figure 4.1 details the subtasks and the variables associated with them. Messages need to be created at nodes and added to the message queue, associated with each message is a value representing the sender and the destination as well as a timestamp and a unique identifier. When messages reach their destination they should be removed from the queue. It is at this point that an acknowledgement should be created. Acknowledgments will be stored in a separate queue unique to them. Acknowledgements are assigned a unique identifier that matches the UID associated with the message they acknowledge, this enables other nodes in the system to know which message an acknowledgment is acknowledging and assuming a message's UID is unique ensures that an acknowledgment does not acknowledge a message incorrectly.

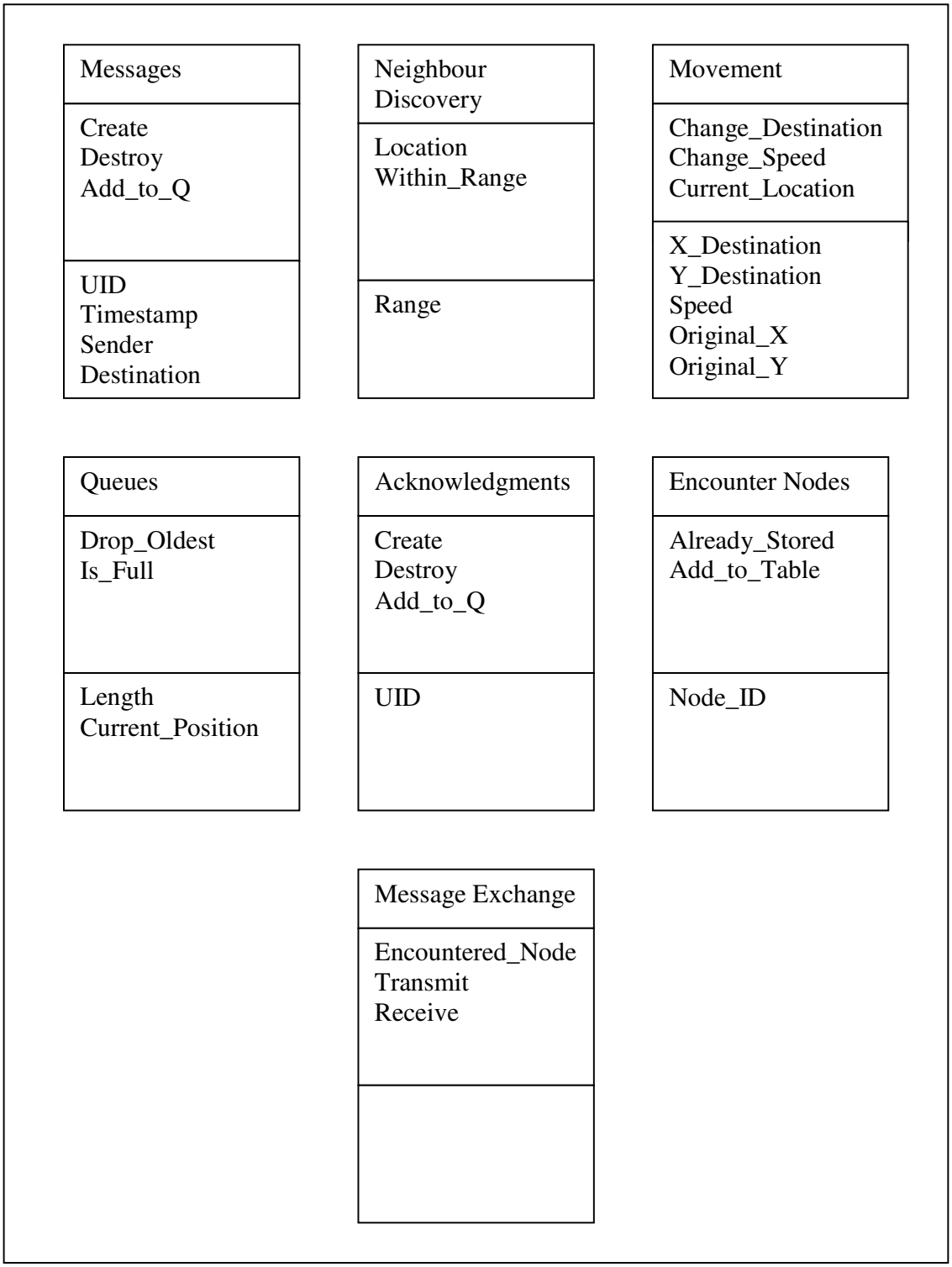


Figure 4.1 Task Breakdown of PR Protocol

As the implementation will be making use of NS-2, the protocol will be implemented in Tcl, the language used by NS-2.

4.1.1 NS-2

NS-2 is a discrete event, packet level network simulator. It is primarily used by the academic community to confirm research in the area of networking. The project is open-source and users are free to add or modify any sections as they deem necessary provided the key aim is one of research, even if the research is carried out in a commercial environment. Ns is capable of supporting both wired and wireless (local and satellite) networks, as well as substantial support for simulation of TCP, routing, and multicast protocols. The simulator runs in a non-real-time fashion, all commands are preprocessed and ordered in advance.

Ns makes use of Tcl and C++ to implement protocols and setup and direct simulations. C++ is often used when control is needed at the packet level however the boundaries are blurred, and in successive changes and builds this boundary has been eroded significantly. In this same time, Tcl has been growing as a language, whereas originally it was strictly a scripting language that had the sole purpose of ordering events, Tcl is now a mature language full of control structures and procedures.

Nam is a companion tool for Ns, it is an animator, designed specifically to visualize the protocols implemented in ns. Figure 4.2 is a simple example of a visualization produced by nam. The circles represent nodes, in this case the nodes are wired, as can be seen by the thick black lines connecting them. Nam

is also capable of visualizing traffic flow, queues, dropped packets and a significant number of other network related issues.

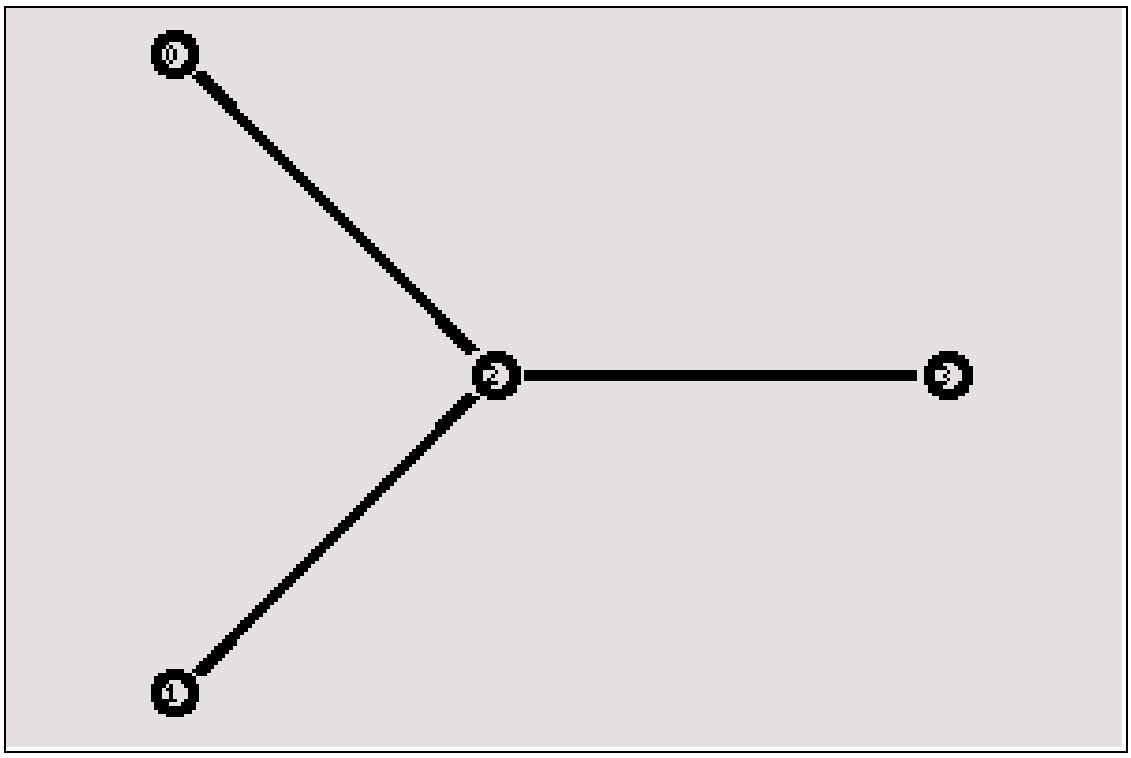


Figure 4.2: Example Nam Output

Ns was not capable of wireless simulations until the addition of CMU Monarch extensions, sometimes referred to as Rice Monarch extensions for ns-2. Though this project was not only limited to the addition of wireless capabilities to ns they are the most significant and important addition. The extensions allowed for the simulation of mobile nodes with programmable trajectories, four ad hoc routing protocols, DSR, AODV, TORA, and DSDV, a wireless network interface modeling the Lucent WaveLAN DSSS radio (at the time of the project, this product set new standards for coverage, range, reliability, and throughput at

low power consumption), and visualizations of the scenarios and simulation traces (there were a number of other additions but they are less relevant to this project).

Figure 4.3 depicts a logical overview of how nodes are connected together using the CMU Monarch extensions. The channel depicted is in effect a set frequency and if desired nodes can be connected to a number of different channels. When a packet is sent every node in fact receives the packet, however, the packet is only passed up the stack, if the radio propagation model determines that they are in range of the sender. An assumption is made that nodes do not move often or fast [27], however in the scenarios in which PR is expected to be used this may be an unsafe assumption. As such, the implementation of PR undertaken will not rely on the extensions provided by the CMU Monarch project for this section.

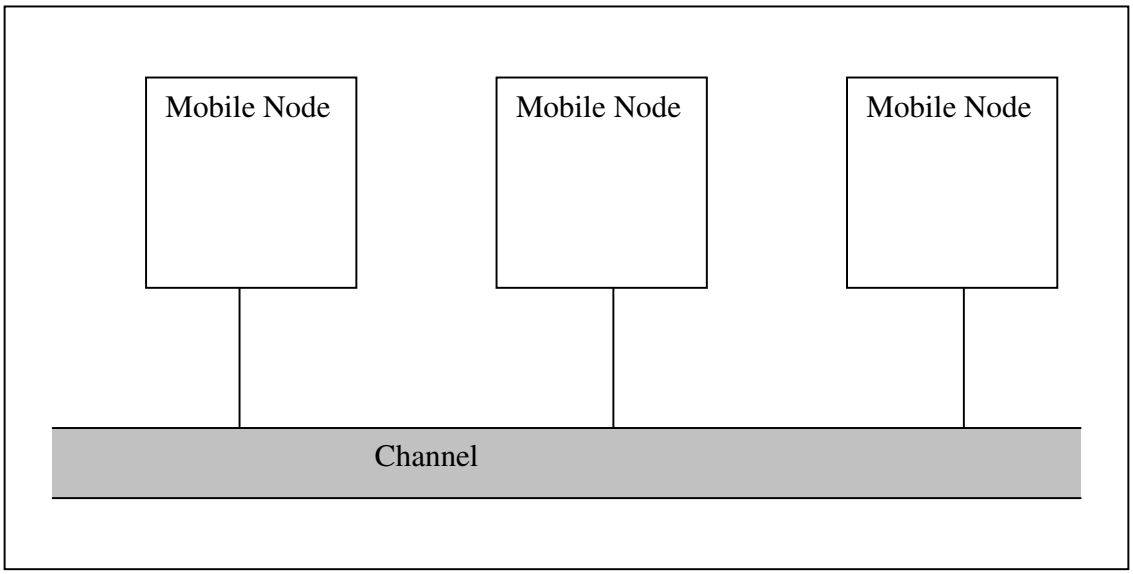


Figure 4.3: Logical overview of how nodes are connected by CMU Monarch

Ordinarily new protocols are implemented in C++, and Tcl is used as the control language, in effect tcl can be used to stack together pieces from the various implementations of protocols, network interfaces, applications etc…, however, as previously started the assumption on node mobility and speed made by the authors of the CMU Monarch extensions would be wholly unsound. The authors also state that this assumption is the most “hard-to-fix” problem with the physical layer, as such re-writing the extensions is not viable, or part, of this project. Moving the propagation model into Tcl allows for a simple method of determining packet delivery, as a node’s location can be updated using recursive calls in tcl, though this is computationally expensive for the simulation, as ns is not real-time it should not affect the results.

Trace files can be automatically generated by ns with large amounts of data on the various layers within the communication stack, and on the packets within the system, including the creation, forwarding and receiving of packets.

4.1.2 Tcl

At approximately the same time that ns began, so did Tcl (1988). Tcl is a scripting language, that allows for the connecting of various other languages (originally C, now also C++, Java, Eiffel, Prolog etc…) if the user so desires. Scripts can be written to allow the bringing together of a number of different sections of code in these languages. However, Tcl is also programmable and allows for complex commands and structures to be created. Often the programs are recursive, and it is in this way that procedures, looping command, and

conditional commands work. Tcl was originally designed with the philosophy that a user would use two or more languages, and tcl would be used as the intermediary to allow communications between the languages. The languages would make use of the standard tcl syntax, plus this would allow users to issue commands, in tcl, to a program using a language they may not necessarily understand.

In an initial attempt to overcome the potential errors that may be introduced by the CMU Monarch extensions, some of the implementation had to be moved to tcl. Once this was done a new problem arose, ns would only allow DSR, AODV, TORA, or DSDV to be chosen as an ad hoc routing protocol. To solve this problem, NOAH [28] was implemented in ns. NOAH is a routing agent which does no routing, packets are received by it and it does not attempt to forward them on. As a result of attempting to circumvent the problems with the radio propagation model in ns, the project was forced to use tcl to implement the PR protocol.

Custom trace files were also output by the program, with information on each message sent within the system, details such as the sender id, destination id, sending time and the time at which the message was eventually received were noted. As was information on the number of messages sent throughout the entire system and the eventual number delivered at the end of the simulation.

4.2 Environment

The environment in which the protocol will be tested will start with a simple system with just three nodes, set up as shown in figure 4.4. Node 1 will initiate a transmission with the destination set as node 3. Node 2 will receive the packet and then move in the direction of node 3. Once within range the packet should be delivered. This scenario would not deliver a message in a system making use of the full PR protocol using information in the 'encountered nodes' table. In order to test the complete protocol, the scenario will start as shown in figure 4.4; node 1 will initiate delivery of a message destined for node 3. Initially node 2 should not accept the message, however, the node will then move within range of node 3, add that node to its table of encountered nodes, then move back within range of node 1, this time the node should accept the message for node 3. Once node 2 moves within range of node 3 again, the message will be delivered. A scenario where node 2 moves backwards and forwards between node 1 and node 3 continuously is a simple way to test the various features of the protocol.

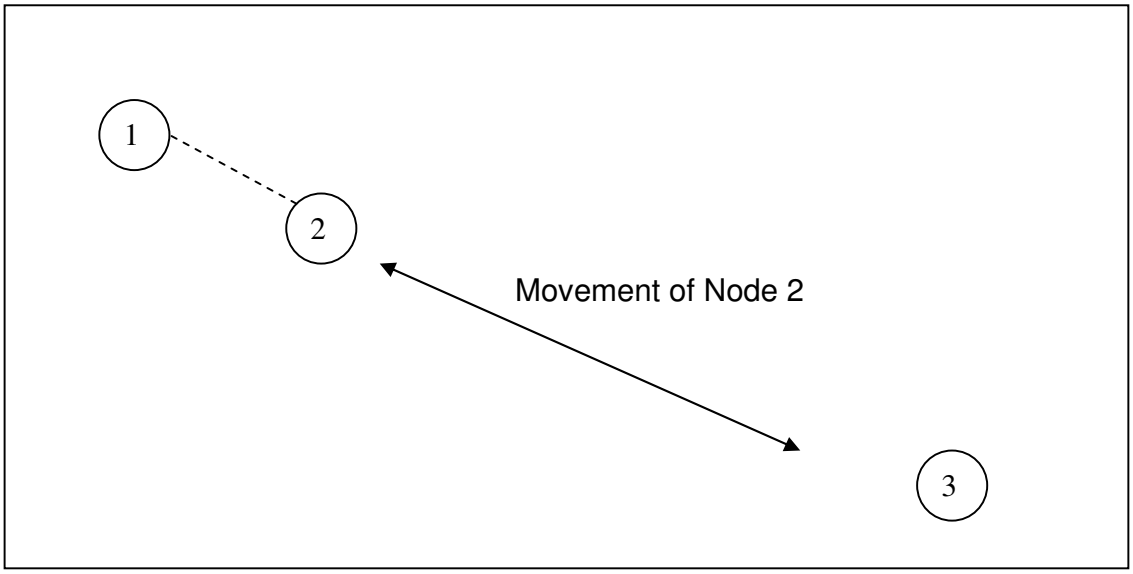


Figure 4.4: Initial Test Scenario for PR protocol

The test area is a square with a side of 300m. Each node has a queue size of 100 packets for messages and the same for acknowledgments. Scenarios are run for 500 seconds, and a check is performed every quarter of a second for neighbour discovery at each node.

4.2.1 Movement Pattern

Figure 4.5 shows an initial layout of the nodes for the scenarios. The PR algorithm is designed with an emerging movement pattern in mind. As such a movement pattern emerges on a per node basis. However, random motion was also used in some simulation runs in an effort to verify that the predictive element of the routing does have an advantage. Nodes are confined to two dimensional movements.

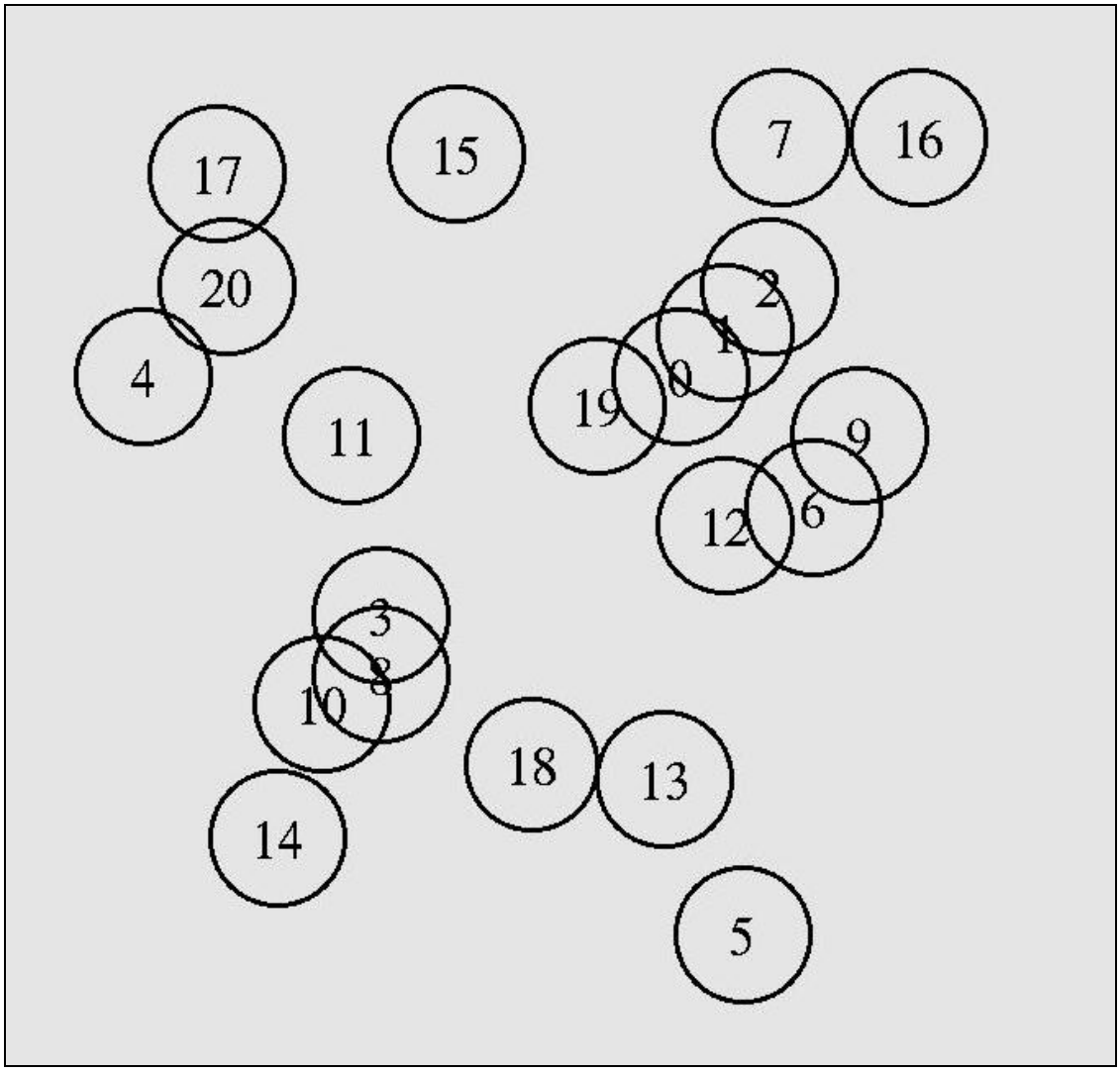


Figure 4.5: Initial Layout of Nodes (taken from nam)

The random movement pattern assigns nodes a random heading and speed, the nodes pause-time is also random. Even when using emerging patterns there is no guarantee that nodes that met in the past shall do so again in the future. A nodes movement pattern is unique to it, more importantly the timing of the patterns are unique, this implies that two nodes may pass within range of each other at some point and may not do so for a significant time again in the future, or possibly never.

4.2.2 Message Pattern

Messages will be created at random within the simulation, the destination will also be chosen randomly. Each message is assigned a unique identifier, within the simulation it is a monotonically increasing number, as it is possible to make this global, obviously in a true environment this would not be possible, however, nodes are assigned unique addresses when they join an ad hoc network, this coupled with a message identifier that monotonically increases at the sender node would ensure a unique message id system.

The number of messages is limited, though the number created within the time of the scenario is guaranteed. In half of the simulations, the maximum number of messages in the system equals the size of the message buffers at each node. Messages not delivered in this scenario can only have failed to have been delivered due to a lack of route to the destination from the sender, as no messages are dropped from the queue. These results can be used to normalize the results where messages significantly outnumber the queue size at each node. An example of this is shown in table 4.1.

Scenario	% Delivered	Normalized delivery %
Messages in system = Q size	75	100
Messages in system >> Q size	45	60 (45*100/75)

Table 4.1: Normalizing delivery rates

For the other half of the simulations the number of messages created and sent will be much greater than the maximum queue size. The number of messages will be twenty times the queue size for these simulations.

Chapter 5

Simulation Results and Evaluation

This chapter details the results of the various simulations using the PR protocol implemented. The scenarios being implemented are not conducive to delivery of messages by the routing protocols already implemented in ns (DSR, DSDV, TORA, AODV), therefore a comparison is being made between PR and an epidemic routing protocol, which is in effect the PR protocol with intelligent forwarding removed, i.e. when checking if a node is in the encountered-table, a true value is always returned.

5.1 Results Using Emerging Movement Patterns

Figures 5.1 and 5.2 show end-to-end delay times, both of these graphs are for scenarios making use of emerging movement patterns with the number of

messages greatly exceeding the queue size. Figure 5.1 is the end-to-end delay using PR, when this is compared to the end-to-end delay of messages delivered using epidemic routing, there are subtle differences. The delay experienced by messages at the early phase of the scenario making use of PR are on longer on average than the delay experienced by messages delivered using epidemic routing. This is due to the time it takes to populate the table of encountered nodes in the PR protocol. Initially this table is either empty, or poorly populated, this leads to messages not being forwarded to nodes that have the potential to deliver messages in the future. This is not the case in epidemic routing; messages are forwarded to all neighbours immediately, ensuring a minimum end-to-end delay. As will be shown later, there is a trade-off for epidemic routing, with regards to the percentage of messages actually delivered. In figure 5.1 there are also a couple more tall peaks than appear in figure 5.2, these appear for the same reason as the increased end-to-end delay for the early messages, but are due to nodes whose movement pattern is particularly long and slow. The messages with this increased end-to-end delay appear almost exclusively in the time range of 100-300 seconds as by this time, the nodes have all populated their encountered nodes tables.

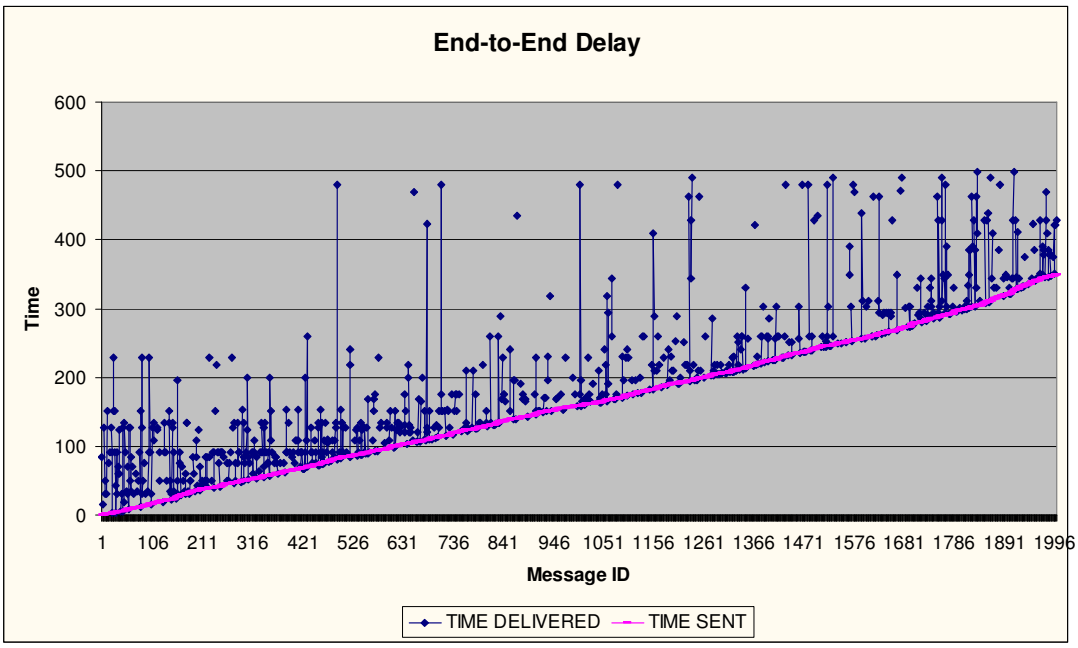


Figure 5.1: End-to-End delay, for messages delivered using PR

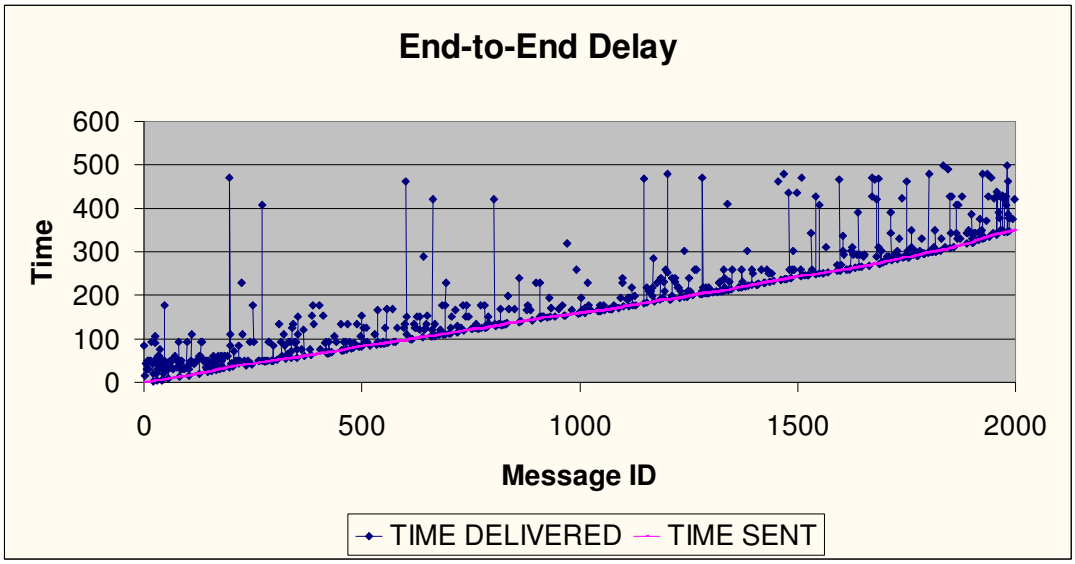


Figure 5.2: End-to-End delay, for messages delivered using Epidemic Routing

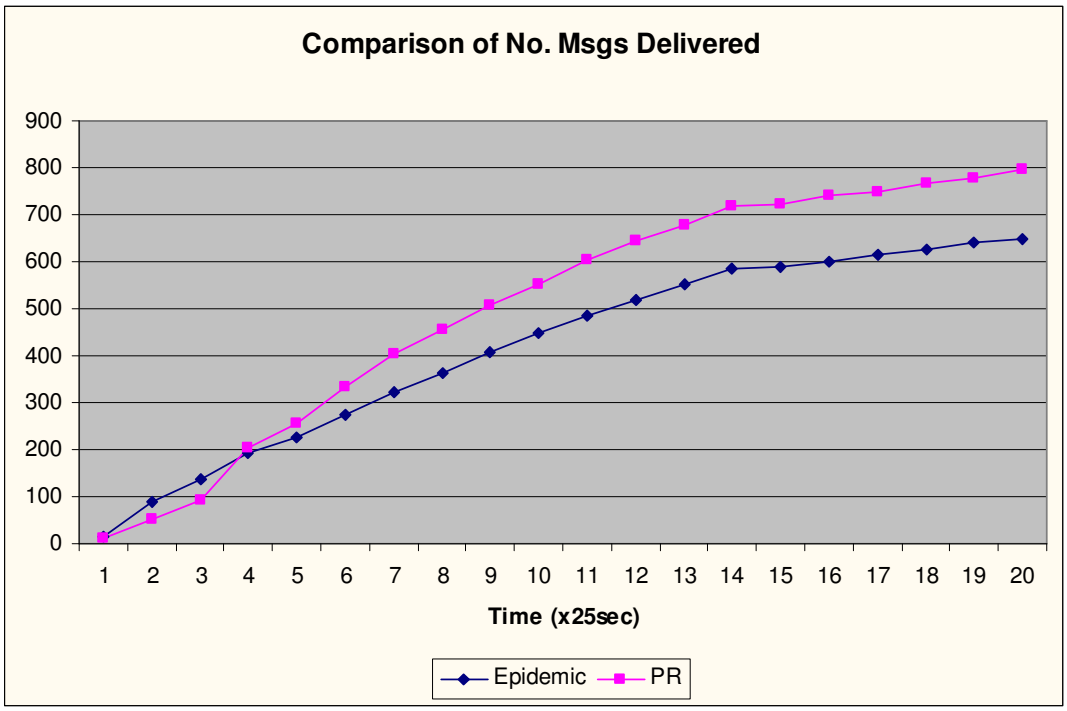


Figure 5.3: Comparison of the number of messages delivered by Epidemic routing and PR, with a large number of messages

In figure 5.3, there are two interesting features to note. Initially epidemic routing exceeds the delivery rate of PR; this is an effect of the need to populate the encounter tables also. However, PR soon regains the lost ground and overtakes epidemic routing. The leveling off of the graph (and of future similar graphs) is an anomaly of the procedure that creates and sends messages, as this procedure reaches its message limit before the simulation ends. If this did not happen, it can be expected that the cures would continue to diverge, and PR would show a more dramatic improvement over epidemic routing.

When the number of messages in the system equals the size of the message queue at each node, the end-to-end delay results are similar to that of the previous scenario, in which PR initially has larger delays, but this disadvantage disappears in a relatively short space of time.

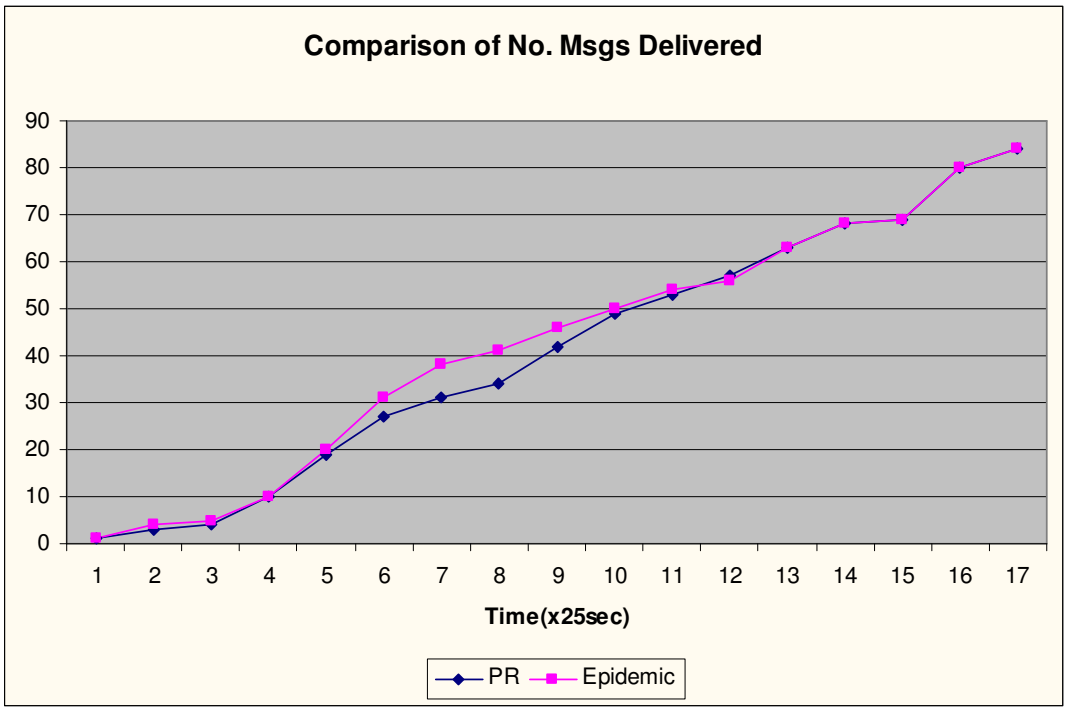


Figure 5.4: Comparison of the number of messages delivered by Epidemic routing and PR, with a small number of messages

Figure 5.4 is interesting as in this simulation all messages can potentially be stored at every node, and due to epidemic routings very nature, this ensures it will deliver all messages in the shortest time possible. And yet PR, despite the delays caused by populating the encounters table, compares very favourably, only in the mid section losing ground to epidemic routing, but soon re-catching epidemic routing's delivery rate and matching it message for message.

5.2 Results Using Random Movement Patterns

Figure 5.5 depicts the number of messages delivered every twenty-five seconds for both epidemic routing and PR in the scenario where the number of

messages equals the queue size per node. Similar to the previous case Epidemic routing initially outperforms PR by a small percentage, and in the mid-section again performs ahead of PR, and again PR delivers the same number of messages by the end of the simulation. This is not surprising, what is intriguing however is that when random motion is used, one hundred percent of messages are delivered. For both PR and epidemic routing the maximum delivered was only eighty-four percent. The explanation can be inferred from the figures themselves; epidemic routing only managed to deliver eighty-four percent of messages, in a scenario where messages are never dropped, this implies that there were nodes that while not necessarily isolated from the rest of the nodes, came into contact with nodes before messages destined for them had propagated to these same nodes. Therefore it was most likely an anomaly brought about by the movement patterns coupled with the message patterns.

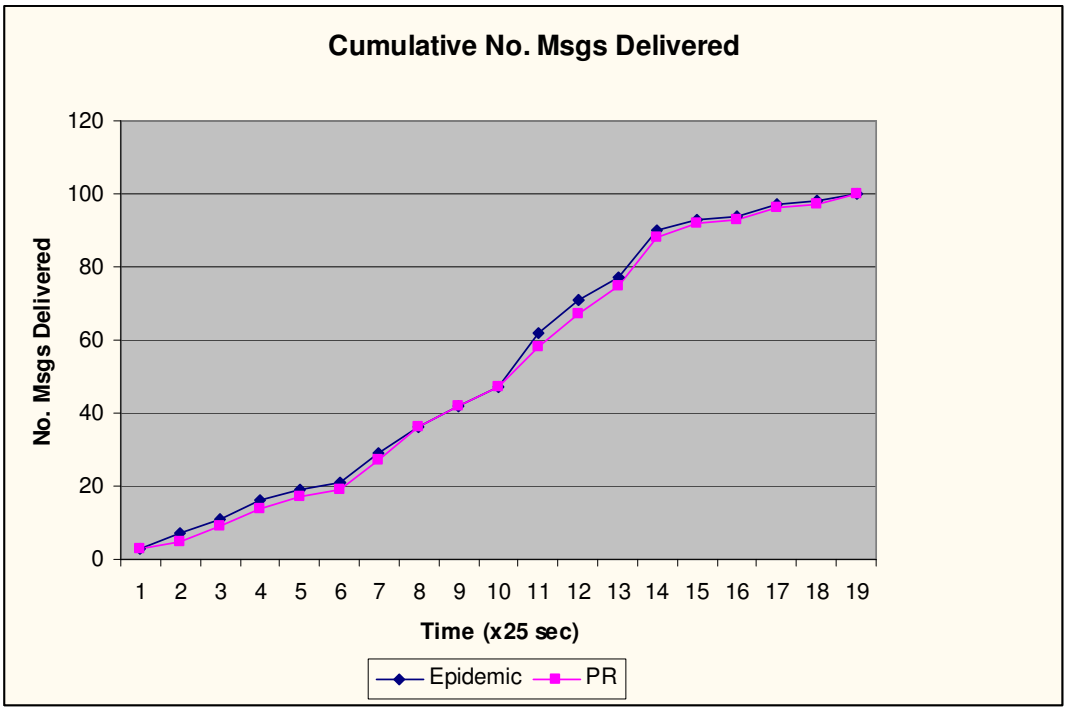


Figure 5.5: Comparison of the number of messages delivered by Epidemic routing and PR, with a small number of messages

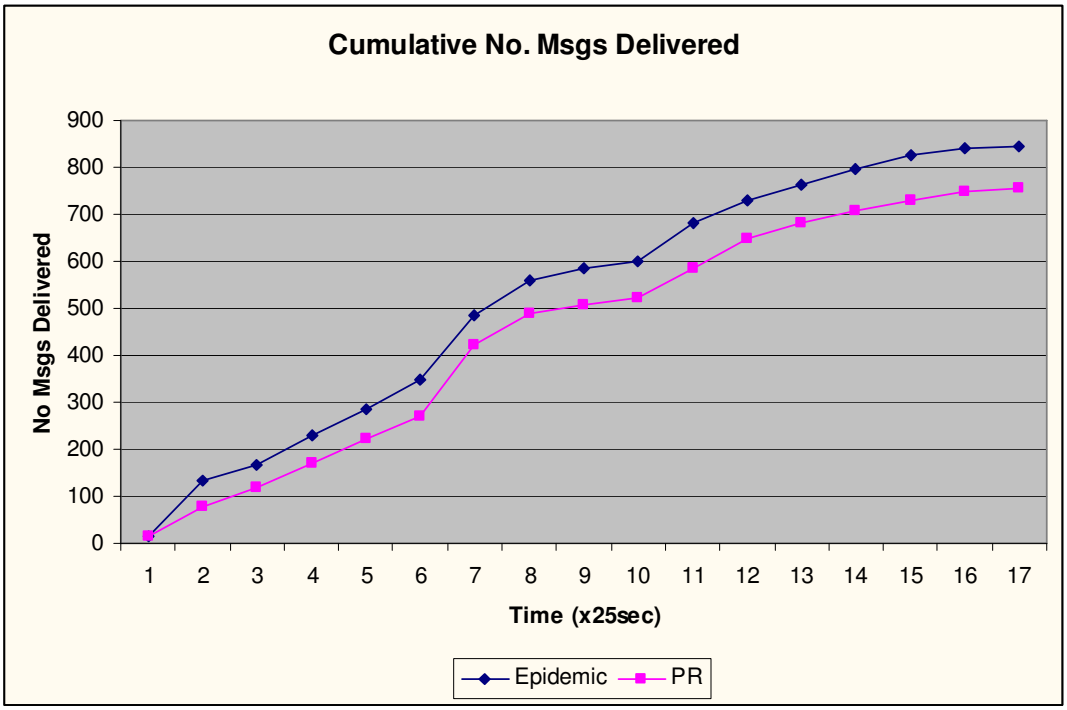


Figure 5.6: Comparison of the number of messages delivered by Epidemic routing and PR, with a large number of messages

Figure 5.6 depicts the total number of messages delivered by epidemic routing and PR, in the scenario where the number of messages eclipses the size of the message buffer at each node. A similar pattern emerges at the early stages, whereby epidemic routing delivers a higher number of messages. In this case, there are no emerging patterns, as such epidemic routing is unable to deliver a large number of messages in a short period as nodes do not start to accept messages for a particular set of nodes thus messages are more likely to be dropped. PR does however manage to prevent epidemic routing from seriously outperforming it, this is most likely due to the fact that PR will tend towards epidemic routing itself, as the likelihood of encountering a particular node will tend to a value of one divided by the number of nodes in the network, i.e. each node's encounters table will tend to have a value for each node in the network.

5.3 Evaluation

The scenarios used in for this analysis would likely yield very low results for the ad hoc routing protocols already available in ns (DSR, DSDV, AODV, TORA). A simulation was engaged in which made use of DSR and an emerging movement pattern, the results were not good, only a handful of messages were delivered. Tables 5.1 and 5.2 give a breakdown of the delivery rates for epidemic routing and PR. The delivery rates are normalized as discussed earlier in this chapter. This gives the results an even keel for discussion.

Scenario	% Delivered	Normalized Delivery Rate (%)
Epidemic, ack, msgs = Q size (E)	84	100
Epidemic, ack, msgs >> Q size (E)	32.40	38.57
Epidemic, ack, msgs = Q size (R)	100	100
Epidemic, ack, msgs >> Q size (R)	42.15	42.15

Table 5.1: Percentage Delivery Rates for Epidemic Routing

Scenario	% Delivered	Normalized Delivery Rate (%)
PR, ack, msgs = Q size (E)	84	100
PR, ack, msgs >> Q size (E)	39.75	47.32
PR, ack, msgs = Q size (R)	100	100
PR, ack, msgs >> Q size (R)	37.70	37.70

Table 5.2: Percentage Delivery Rates for PR

NOTE: E – Emerging Movement Pattern

R – Random Movement Pattern

As expected PR has a significantly higher delivery rate than epidemic routing when emerging movement patterns are used. That is to be expected, and reinforces the belief that making use of past information to dictate whether a node accepts a message or not is effective. The delivery rate is improved by approximately twenty-three percent. When emerging movement patterns are used, nodes tend to encounter the same nodes more than once, thus they build up knowledge of an area of the network. The intelligent forwarding mechanism improves the delivery rate because of this. Whereas with epidemic routing a message is blindly passed to a node that does not already possess a copy of the

message, possibly (and likely) forcing another message to be dropped, a message is only forwarded in PR if there is a good probability of the receiving node being capable of delivering it. This has a double effect, first of all, messages become less likely to be dropped, and therefore remain at each node for a longer period of time, this in and of itself increases the likelihood of delivery. There is a secondary effect due to the choice of queuing algorithm: when a message is dropped, it is dropped based on the time it was received at the node, the older messages are most likely to have been delivered.

PR has the effect of a message quickly migrating to the area of the network where the destination node resides, without the need to actually know the location of the destination node or any of the intermediary nodes. This is similar to a group of small clusters of ad hoc networks with a gateway node, or nodes, each of which accept messages destined for any nodes in that cluster, and upon their return to that cluster, deliver them.

Random movement patterns remove the advantage PR has over it, by in effect making each encounter one of chance. Past encounters are useless in determining the likelihood of a future encounter. PR routing quickly degrades into epidemic routing, despite this, epidemic routing in fact delivers more messages than PR. This is purely due to the messages epidemic routing is able to deliver at the start of the simulation, while PR still attempts to use encounter information (that has not yet been discovered) to deliver messages, i.e. messages are not propagated through the network. However it can be seen in

figure 5.6 that from approximately seventy-five seconds to the end of the simulation, the same number of messages is delivered by both protocols.

Though the results are not present in the text, simulations were also run in which there were no acknowledgements; obviously this would make no difference to the scenario where a node is capable of storing every message sent in the network however the difference was very apparent when a large number of messages were sent. In the epidemic routing case, the number of messages delivered dropped significantly and the same messages tended to propagate throughout the system in a continuous cycle, a message would be dropped and then it would cycle back around and be in the queue again. In the case where PR was used, the effect was not as severe, though it did still heavily impact on the number of messages delivered. Scenarios that do not make use of acknowledgments would allow for the use of the PR protocol in an EMCON system allowing the propagation of messages in one direction only.

Chapter 6

Conclusion

In this chapter a review of the objectives that were stated at the start of the project is given against what was achieved. The second section in this chapter outlines potential future work, as with all research a number of ideas and issues arose that were outside the scope of the project, but are nonetheless worthy of mention in hope of an upcoming researcher undertaking them.

6.1 Objectives Fulfilled

At the start of this project there were five stated objectives:

1. **Complete a survey of ad hoc routing protocols, specifically protocols that use a store-carry-and-forward paradigm.** This objective was adequately completed at the time of the completion

of this project, as always there will undoubtedly be more research into the area completed in the future, as such any omissions are most likely due to that fact.

2. **Design an algorithm that follows the store-carry-and-forward paradigm.** An algorithm was devised, influenced in certain areas as referenced.
3. **Implement the algorithm and devise simulations to verify its correctness.** The algorithm was implemented in Tcl, and basic tests were performed throughout the development stage to confirm the accurateness of the protocol.
4. **Run simulations and determine the protocols strengths and weaknesses.** A number of different simulations were run and the algorithm was found to be an improvement over an epidemic routing protocol, with a worst case scenario that matched it.
5. **Recommend improvements.** The final section of this chapter will address this objective.

6.2 Future Work

There are a number of improvements that can be made to the project and as well as a number of additions, it is here that they are noted.

Nodes could be connected to different channels to allow for simultaneous transmitting and receiving of messages and acknowledgments. If two channels

were used, a single channel could be for sending and another for receiving, another interesting possibility, which may prove to be excessive, would be to use four channels, one for sending messages, one receiving messages, one for sending acknowledgements, and a final one for receiving acknowledgements.

Integrate the PR protocol as a fully fledged additional ad hoc routing protocol for ns. A single decision at an early stage removed this possibility from this project. Despite the need to implement only a minor amount of code in C++, and use the (flawed) radio propagation model associated with the CMU Monarch extensions, time was not available to do this at the end of this project due to circumstances beyond the author's control.

Test the system in a real-world environment by installing it onto PDAs and laptops. Potentially a large number of users could be issued key-rings which are the mobile nodes, and therefore transport mechanisms. In a University environment this would be ideal as students (and lecturers alike) could be issued the key-rings at the start of the term. The system could be tested fully over an entire semester, and highly detailed results would be available for analysis.

Appendices

Appendix A

Abbreviations

IT – Information Technology

MANET – Mobile Ad hoc Network

DSR – Dynamic Source Routing

NOAH – N Ad Hoc routing

AODV – Ad hoc On-demand Distance Vector routing

DSDV – Destination Sequenced Distance Vector routing

PDA – Personal Digital Assistant

IETF – Internet Engineering Task Force

OSPF – Open Shortest Path First

ZHLS – Zone-based Hierarchical Link State routing

EMCON – Emission Control

NS-2 – Network Simulator version 2

TTL – Time To Live

PR – Parasitic Routing

DARPA – Defense Advanced Research Project Agency

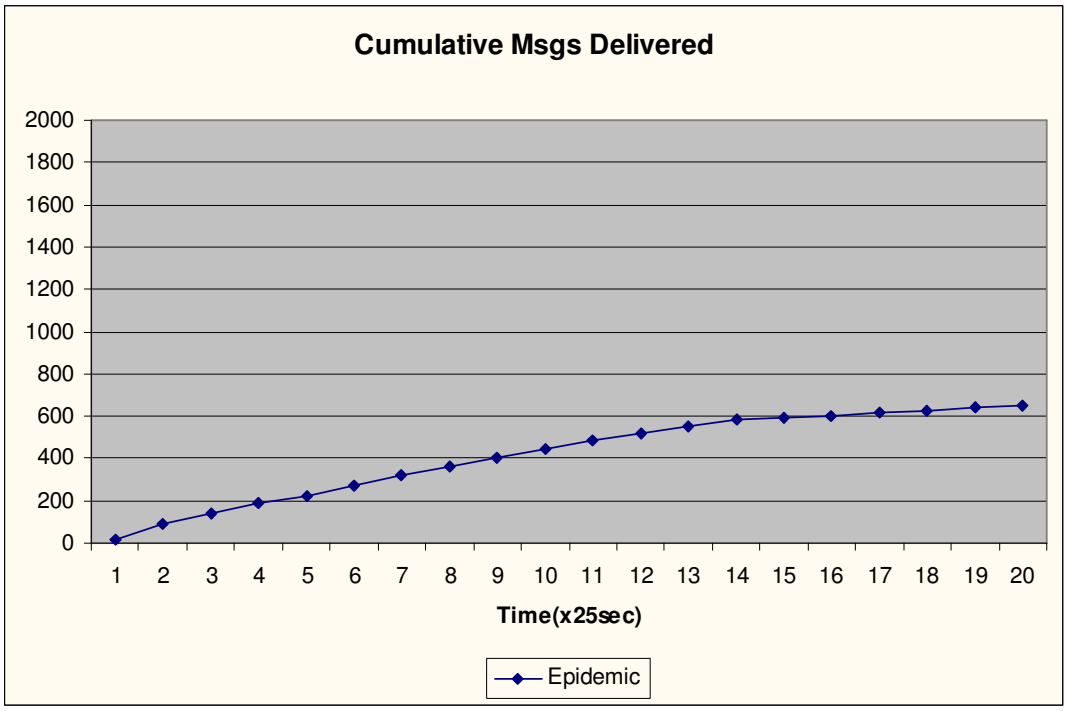
PARC – Palo Alto Research Center

UID – Unique Identifier

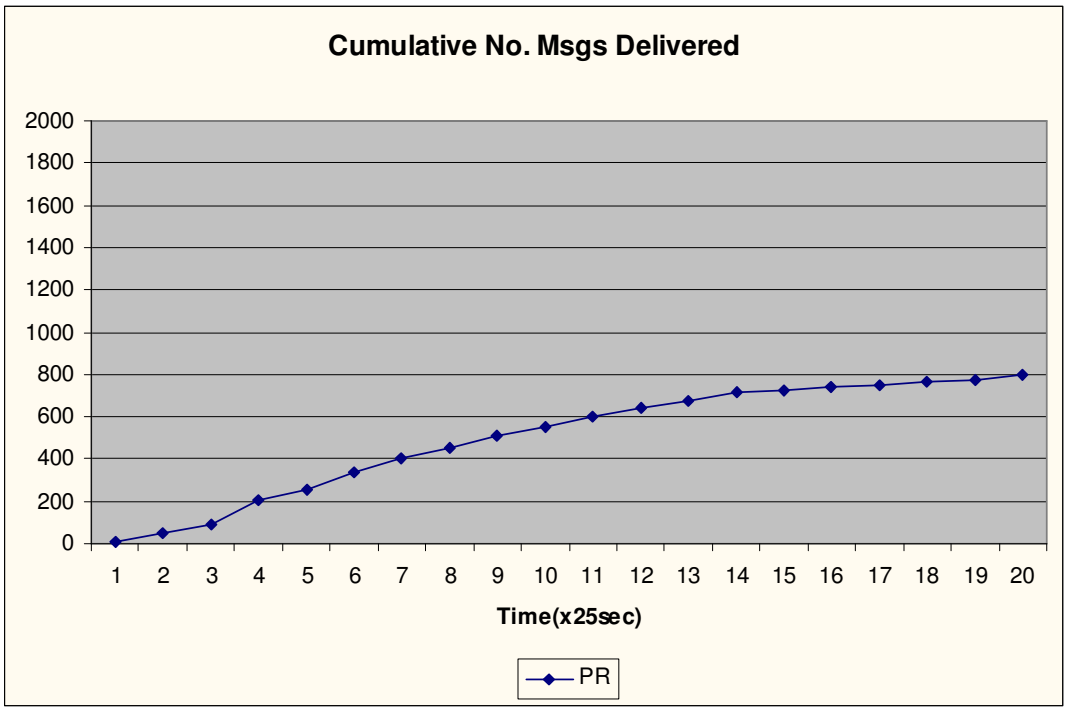
NOAH – No Ad Hoc Routing Protocol

Appendix B

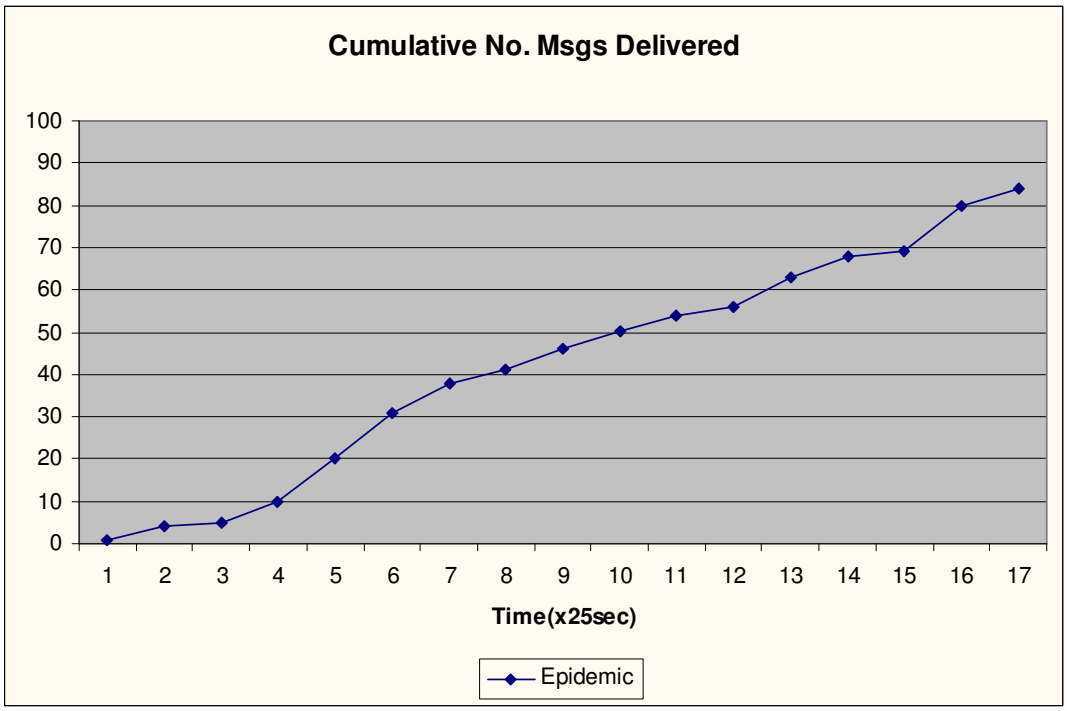
Additional Results



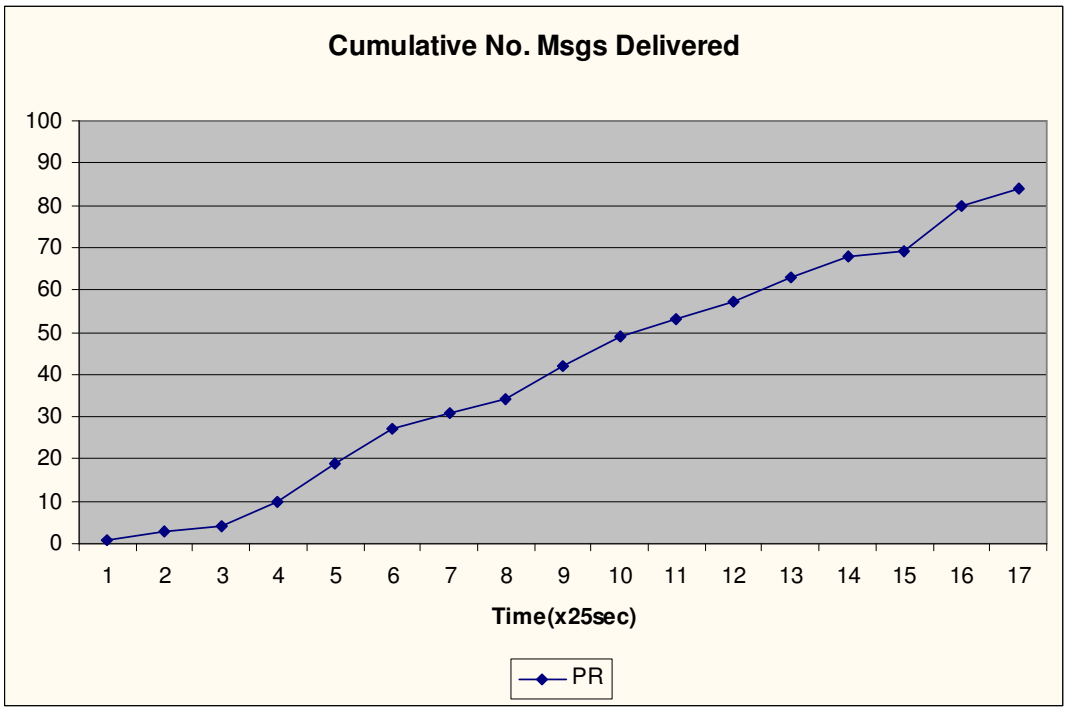
The cumulative number of messages delivered over the length of the simulation, during which two thousand messages were sent, using epidemic routing, with an emerging movement pattern.



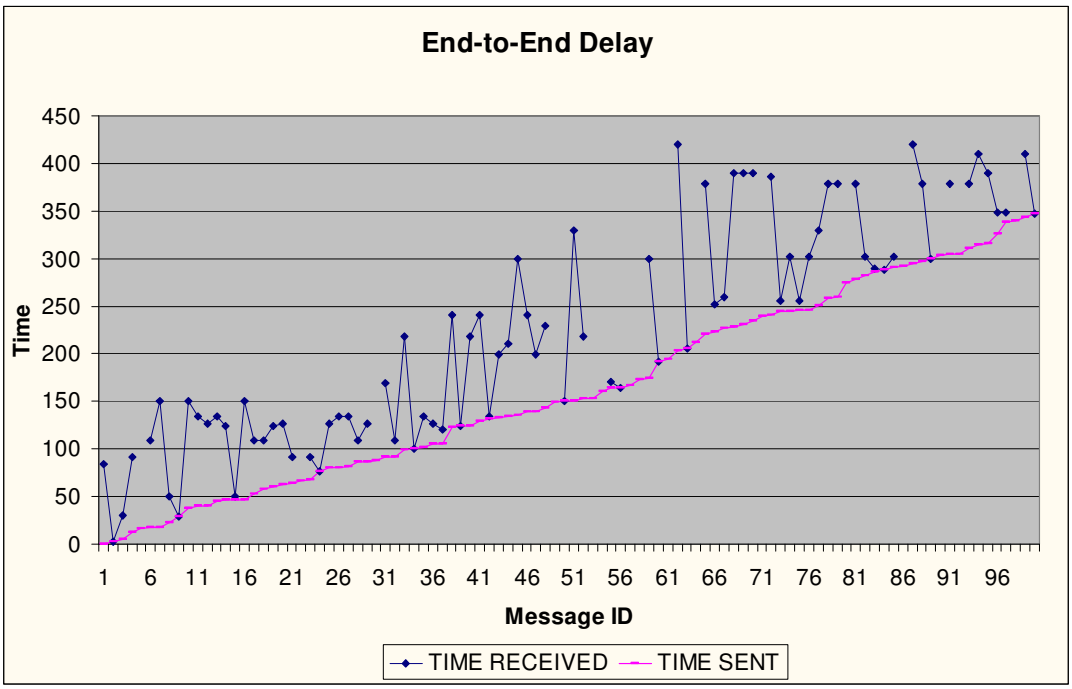
The cumulative number of messages delivered over the length of the simulation, during which two thousand messages were sent, using PR, with an emerging movement pattern.



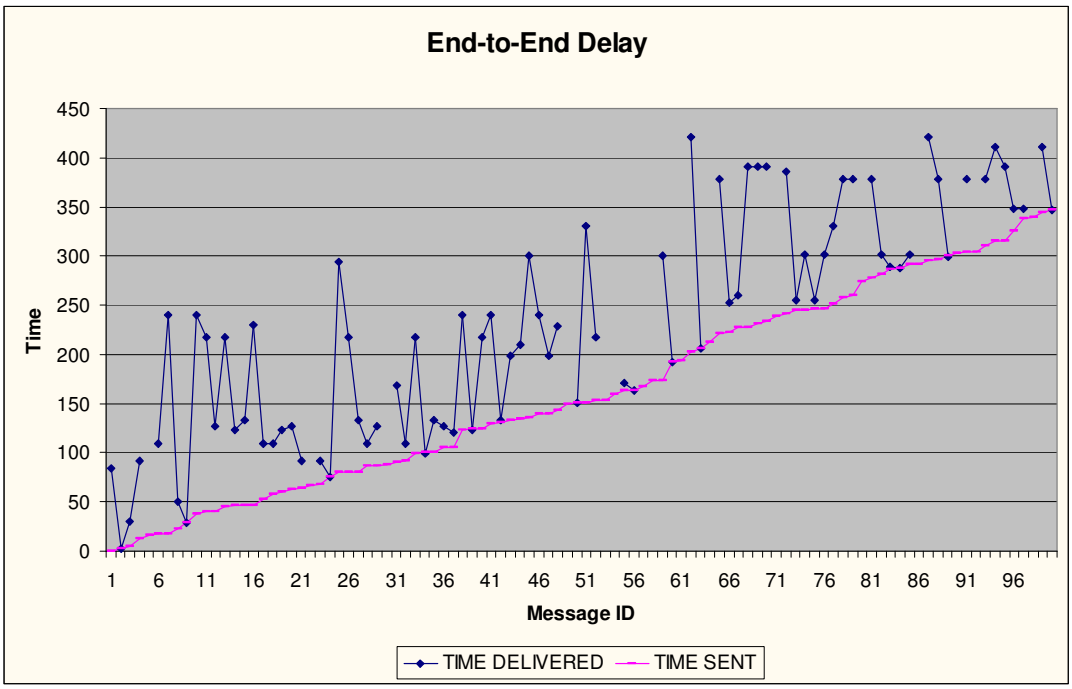
The cumulative number of messages delivered over the length of the simulation, during which one hundred messages were sent, using epidemic routing, with an emerging movement pattern.



The cumulative number of messages delivered over the length of the simulation, during which one hundred messages were sent, using PR, with an emerging movement pattern.



The end-to-end delay on a per message basis, for epidemic, over the length of the simulation, during which time one hundred messages were sent, and an emerging movement pattern was used.



The end-to-end delay on a per message basis, for PR, over the length of the simulation, during which time one hundred messages were sent, and an emerging movement pattern was used.

Appendix C

Sample Trace Output

The following is a sample of the output generated by ns; it is a record of the movements of the nodes:

```
M 21.25000 14 (65.00, 45.00, 0.00), (177.00, 103.00), 7.00
M 24.25000 12 (215.00, 150.00, 0.00), (213.00, 29.00), 8.00
M 27.25000 5 (221.00, 13.00, 0.00), (77.00, 236.00), 9.00
M 30.25000 13 (79.19, 160.33, 0.00), (269.00, 291.00), 15.00
M 33.25000 10 (8.00, 288.00, 0.00), (7.00, 59.00), 1.00
M 36.25000 4 (22.17, 226.91, 0.00), (157.00, 131.00), 13.00
M 39.25000 14 (176.89, 102.94, 0.00), (70.00, 216.00), 0.00
M 42.25000 8 (100.00, 100.00, 0.00), (117.00, 70.00), 11.00
M 45.25000 13 (264.52, 287.92, 0.00), (108.00, 197.00), 14.00
M 48.25000 7 (234.00, 280.00, 0.00), (124.00, 7.00), 16.00
M 51.25000 8 (117.00, 70.00, 0.00), (70.00, 250.00), 6.00
M 54.25000 10 (7.91, 267.00, 0.00), (256.00, 36.00), 4.00
M 57.25000 13 (119.25, 203.53, 0.00), (40.00, 281.00), 4.00
M 60.25000 15 (125.00, 275.00, 0.00), (107.00, 194.00), 15.00
M 63.25000 17 (45.00, 268.00, 0.00), (192.00, 8.00), 5.00
M 66.25000 11 (90.00, 180.00, 0.00), (186.00, 89.00), 4.00
M 69.25000 9 (260.00, 180.00, 0.00), (24.00, 55.00), 6.00
M 72.25000 11 (107.42, 163.49, 0.00), (53.00, 255.00), 8.00
M 75.25000 20 (48.00, 230.00, 0.00), (112.00, 112.00), 2.00
M 78.25000 16 (280.00, 280.00, 0.00), (67.00, 85.00), 15.00
M 81.25000 1 (215.00, 215.00, 0.00), (228.00, 52.00), 5.00
M 84.25000 6 (43.00, 293.00, 0.00), (235.00, 286.00), 0.00
M 87.25000 15 (107.00, 194.00, 0.00), (4.00, 162.00), 17.00
M 90.25000 17 (111.44, 150.48, 0.00), (143.00, 93.00), 13.00
M 93.25000 14 (176.89, 102.94, 0.00), (29.00, 287.00), 6.00
M 96.25000 6 (43.00, 293.00, 0.00), (159.00, 111.00), 17.00
M 99.25000 0 (200.00, 200.00, 0.00), (294.00, 278.00), 10.00
M 102.25000 10 (148.43, 136.16, 0.00), (48.00, 3.00), 11.00
M 105.25000 5 (77.00, 236.00, 0.00), (150.00, 151.00), 11.00
M 108.25000 0 (269.26, 257.47, 0.00), (177.00, 203.00), 6.00
M 111.25000 10 (88.82, 57.12, 0.00), (85.00, 57.00), 14.00
M 114.25000 14 (97.97, 201.16, 0.00), (210.00, 282.00), 15.00
M 117.25000 16 (67.00, 85.00, 0.00), (94.00, 75.00), 6.00
M 120.25000 17 (143.00, 93.00, 0.00), (298.00, 16.00), 1.00
M 123.25000 4 (157.00, 131.00, 0.00), (90.00, 162.00), 0.00
M 126.25000 20 (96.63, 140.34, 0.00), (123.00, 191.00), 13.00
M 129.25000 7 (124.00, 7.00, 0.00), (211.00, 260.00), 4.00
```

The following is a sample of the output from the custom traces used by the PR algorithm, is starts by giving a key to the nodes values, so that the lines can be deciphered:

```
Node 15: _o284
Node 16: _o302
Node 17: _o320
Node 18: _o338
Node 19: _o356
Node 20: _o374
SENDING FROM: _o302 TO: _o374 UID: 0
SENDING FROM: _o140 TO: _o86 UID: 1
SENDING FROM: _o284 TO: _o302 UID: 2
SENDING FROM: _o140 TO: _o284 UID: 3
SENDING FROM: _o212 TO: _o14 UID: 4
SENDING FROM: _o374 TO: _o230 UID: 5
SENDING FROM: _o104 TO: _o68 UID: 6
SENDING FROM: _o176 TO: _o68 UID: 7
SENDING FROM: _o14 TO: _o248 UID: 8
SENDING FROM: _o302 TO: _o248 UID: 9
SENDING FROM: _o194 TO: _o266 UID: 10
SENDING FROM: _o32 TO: _o266 UID: 11
SENDING FROM: _o158 TO: _o122 UID: 12
SENDING FROM: _o176 TO: _o68 UID: 13
SENDING FROM: _o86 TO: _o158 UID: 14
SENDING FROM: _o104 TO: _o14 UID: 15
SENDING FROM: _o158 TO: _o284 UID: 16
SENDING FROM: _o212 TO: _o32 UID: 17
SENDING FROM: _o158 TO: _o32 UID: 18
SENDING FROM: _o266 TO: _o176 UID: 19
SENDING FROM: _o320 TO: _o176 UID: 20
SENDING FROM: _o374 TO: _o32 UID: 21
SENDING FROM: _o194 TO: _o68 UID: 22
SENDING FROM: _o320 TO: _o266 UID: 23
DELIVERED
SENDING FROM: _o248 TO: _o374 UID: 24
SENDING FROM: _o50 TO: _o338 UID: 25
SENDING FROM: _o50 TO: _o248 UID: 26
```

Also a sample from a custom trace, this time, keeping track of the time a message is sent, from what node, and which node is the destination, in this trace, the nodes are identified numerically:

```
TIME  SENDER DEST
347   2     19
```

```
76    10    8
228   16    14
3     10    3
0     16    20
278   10    9
223   12    20
206   17    2
87    8     17
291   0     8
245   16    9
173   5     7
282   20    5
```

References

[1] Market report, “Embedded Wi-Fi Market Undergoing Major Shift”, InStat/MDR, August 2004

[2] Press release Intel, “Intel Advances Intel® Centrino™ Mobile Technology With New Wireless Capabilities, Software Features”, www.intel.com, August 2004

[3] http://www.ietf.org/proceedings/97aug/transit97aug-86.htm, IETF, 39th IETF Meeting Munich, Bavaria, Germany

[4] C. Perkins, “Ad Hoc Networking”, Addison-Wesley, 2001, pp. 29-51

[5] D. Johnson, “Routing in Ad Hoc Networking of Mobile Hosts”, Proc. ACM Mobicom ’94, December 1994

[6] M. Grossgaluser and D. Tse, “Mobility Increases the Capacity of Ad Hoc Wireless Networks”, IEEE/ACM Trans. On Networking, Vol. 10, no. 4 August 2002

[7] S. Diggavi, M. Grossglauser and D. Tse, “Even One-Dimensional Mobility Increases Ad Hoc Wireless Capacity”, ISIT 02 Lausanne, Switzerland, June 2002

[8] IEEE, “IEEE 802.11g-2003”, http://www.standards.ieee.org/wireless/

[9] IEEE, “IEEE 802.11b-1999”, http://www.standards.ieee.org/wireless/

[10] IEEE, ”IEEE 802.3-2002”, http://www.standards.ieee.org/catalog/olis/lanman.html

[11] C.E. Perkins and P Bhagwat, “Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers”, Comp. Comm Rev., Oct 1994, p. 234-244

[12] M. Joa-Ng and I.T. Lu, “A Peer-to-Peer zone-based two-level link state routing for mobile Ad Hoc Networks”, IEEE Journal on Selected Areas in Communications, Special Issue on Ad Hoc Networks, August 1999, pp 1415-1425

[13] C. Perkins et al, “Ad hoc On-Demand Distance Vector (AODV) Routing”, http://www.ietf.org/rfc/rfc3561.txt, July 2003

[14] D. Johnson, D. Maltz and Y.-C. Hu, “The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)” (work in progress), http://www.ietf.org/internet-drafts/draft-ietf-manet-dsr-10.txt, July 2004

[15] P. Sholander, A. Yankopolus and P. Coccoli “Experimental Comparison of Hybrid and Proactive MANET Routing Protocols”, IEEE 2002

[16] Z. Haas and M. Pearlman, “The Performance of Query Control Schemes for the Zone Routing Protocol”, ACM/IEEE Trans. Net., vol. 9, no. 4, August 2001, pp. 427-438

[17] L. Zhou abd Z. Haas, “Securing Ad Hoc Networks”, IEEE Network 1999

[18] J. Walker, “Overview of 802.11 Security”, http://www.groupe.ieee.org/groups/802/15/pub/2001/Mar01/01154rOP802-15_TG3%-Overvies-of-901-11-Security.ppt, IEEE, March 2001

[19] M. Mauve and J. Widmer, “A Survey on Position-Based Routing in Mobile Ad Hoc Networks”, IEEE Network, November 2001

[20] B. Karp and H. Kung, “GPSRL Greedy Perimeter Stateless Routing for Wireless Networks”, Proc. 6th Annual ACM/IEEE Int. Conf. Mobile Computing and Networking, Boston, MA, August 2000, pp. 243-254

[21] S. Basagni, I. Chlamatac, V. Syrotiuk, B. Woodward, "", Proc. 4th Annual ACM/IEEE Int. Conf. Mobile Computing and Networking, MOBICOM '98, Dallas, Texas, 1998, pp. 76-84

[22] J. Broach et al., "A Performance Comparison of Multi-Hop Wireless Ad Hoc Wireless Networks", Proc. 4th Annual ACM/IEEE Int. Conf. Mobile Computing and Networking, MOBICOM '98, Dallas, Texas, 1998, pp. 85-97

[23] A. Vahdat and D. Becker, "Epidemic Routing for Partially-Connected Ad Hoc Networks", Dept. of Computer Science, Duke University, http://citeseer.ist.psu.edu/vahdat00epidemic.html, 2000

[24] J. Davis, A. Fagg and B. Levine, "Wearable Computers as Packet Transport Mechanisms in Highly-Partitioned Ad-Hoc Networks", Proc. Of the 5th IEEE Int. Symposium on Wearable Computers, 2001, pp. 141-148

[25] A. Pentland, R. Fletcher and R. Hasson, "DakNet: Rethinking Connectivity in Developing Nations", IEEE Computer, January 2004, pp 78-83

[26] W. Zhao and M. Ammar, "Message Ferrying: Proactive Routing in Highly Partitioned Ad Hoc Networks", Proc. Of the 9th IEEE Workshop on Future Trends of Distributed Computing Systems, pp 308-314, May 2003

[27] The CMU Monarch Project, "Extensions to ns", http://www.monarch.cs.cmu.edu/, August 1999

[28] J. Widmer, NO Ad-Hoc Routing Agent (NOAH), Swiss Federal Institute of Technology – Lausanne, Senior Researcher, http://www.icapeople.epfl.ch/widmer/uwb/ns-2/noah.html, 2004

Bibliography

"The ns Manual", http://www. Isi.edu/nsnam/ns/ns-documentation.html

M. Adler and C. Scheideler, "Efficient Communication Strategies for Ad Hoc Wireless Networks", ACM Symposium on Parallel algorithms and Architectures, 1998, pp. 259-268

S. Basagni, I. Chlamtac, and V. Syrotiuk, "Dynamic Source Routing for Ad Hoc Networks Using the Global Positioning System", Proc. of the IEEE Wireless Communications and Networking Conference 1999, New Orleans, LA.

U. Bodin and O. Schelen, "Drop Strategies and Loss-Rate Differentiation", Proc. of the 9th Int. Conf. on Network Protocols, 2001, pp. 146-154

A. Boukerche and S. Rogers, "GPS Query Optimization in Mobile and Wireless Networks", Proc. of the 6th IEEE Symposium on Computers and Communications 2001, pp. 198-203

A. Boukerche, V. Sheetal and M. Choe, "A Route Discovery Optimization Scheme using GPS System", Proc. Of the 35th annual Simulation Symposium, 2002

I. Chatzigiannakis, T. Dimitriou, S. Nikoletseas and P. Spirakis, "A Probabilistic Algorithm for Efficient and Robust Data Propagation in Smart Dust Networks",

Proc. Of the 5th European Wireless Conference on Mobile and Wireless Systems beyond 3G, pp. 344-350, 2004

R. Georges, “A NS-2 Version of the ZRP”, Master in Engineering Project, Cornell University, 2002

S. Giordano, I. Stojmenovic, and L. Blazevic, “Position-Based Routing Algorithms for ad hoc Networks: A Taxonomy”, http://www.site.uottawa.ca/~ivan/routing-survey.pdf,

M. Greis, “Marc Greis’ Tutorial for the UCB/LBNL/VINT Network Simulator “ns””, http://www.isi.edu/nsnam/ns/tutorial/, August 2004

K. Hanna, B. Levine and R. Manmatha, “Mobile Distributed Information Retrieval For Highly-Partitioned Networks”, Proc. Of the 11th IEEE International Conference on Network Protocols, 2003, pp. 38-47

B. Hogan, “Bryan’s NS-2 DSR FAQ”, Masters of Engineering Project, University of Limerick, July 2004

M. Lynch, “An Implementation of a Parasitic Routing Algorithm in a Mobile Ad Hoc Environment”, M. Sc. In Computer Science, Trinity College Dublin, 2002

R. Majumdar and K. Ramamritham, “Exploiting User Mobility Patterns for Adaptive Location Management”, IEEE Distributed Systems Online, December 2003

C. du Mouza and P. Rigaux, “Mobility Patterns”, Proc. Of the Second Workshop on Spatio-Temporal Database Management (STDBM ’04), Toronto, Canada, August 2004

A. Nedos, “Direction Based Routing for Mobile Ad Hoc Networks”, M. Sc. In Computer Science, Trinity College Dublin, 2001

B. Welch, “Practical Programming in Tcl and Tk”, Prentice Hall, Third Edition