Encounter Gossip: A High Coverage Broadcast protocol for MANET

Thesis by

David Edward Cooper

In Partial Fulfillment of the Requirements

for the Degree of

Doctor of Philosophy

Newcastle University

Newcastle upon Tyne, UK

2008

(Defended 31st July, 2008)

To all the people who ask “Are you finished yet?”.

Yes!

Acknowledgements

Like all parts of life, this PhD has been an incredible journey. There have been monumental highs and devastating lows but as reach its completion I must thank those who have made it possible. Firstly I would like to thank my supervisor Paul Ezhilchelvan, for his support and guidance over the duration of this thesis. I would like to thank Isi Mitrani, not only for his analytical skills that shaped our analytical model but also for the support he gave in as an undergraduate supervisor, inspiring me to go onto further academic study. I thank Einar Vollset for joining me in the eternal struggle with GloMoSim. I would like to thank Alex Bystrov for the loan of the PDAs without whom I would not have done the real world experiment.

My parents I thank for all the love and support they have given me through my life and PhD. To the many friends who have been there with me as I despaired or bounced elatedly I thank you all.

This work was carried out as part of the research project PACE (Protocols for Ad-hoc Collaborative Environments), I would like to thank the UK Engineering and Physical Sciences Research Council for their funding.

Finally I would like to thank my partner Liz for taking this journey with me. “We made it! Yay!”

Abstract

Mobile Ad-hoc Networks (MANETs) allow deployment of mobile wireless devices or nodes in a range of environments without any fixed infrastructure and hence at a minimal setup cost. Broadcast support that assures a high coverage (i.e., a large fraction of nodes receiving a broadcast) is essential for hosting user applications, and is also non-trivial to achieve due to the nature of devices and mobility. We propose Encounter Gossip, a novel broadcast protocol, which holds minimal state and is unaware of network topology. Coverage obtained can be made arbitrarily close to 1 at a moderate cost of extra message traffic, even in partition-prone networks. Under certain simplifying assumptions, it is shown that a high coverage is achieved by making a total of $O(n \ln n)$ broadcasts, where n is the number of nodes, and the time to propagate a message is $O(\ln n)$. The effect of various network parameters on the protocol performance is examined. We then propose modifications to minimise the number of redundant transmissions without compromising the achieved coverage. Two approaches are pursued: *timer based* and *history based*. The effectiveness of each of these approaches is assessed through an extensive set of simulation experiments in the context of two mobility models. Specifically, we introduce a new heuristic alpha policy which achieves significant reduction in redundancy with negligible reduction in coverage. A generalisation to multiple broadcasts proceeding in parallel is proposed and the protocol is refined to reduce problems that can occur due to the effects of high mobility when transmitting a large number of messages. Finally, we implement and validate Encounter Gossip in the context of a real-life mobile ad-hoc network. All these investigations suggest that the protocol, together with the proposed modifications and refinements, is suited to MANETs of varying degrees of node densities and speeds.

Declaration

All work contained within this thesis represents the original contribution of the author. Most of the material in this thesis has been, or will shortly be, published in conference proceedings and a journal, as listed below. The material in chapter 3 has been published in 1 and 2. The material in chapter 4 has been published in 3 and 4. In addition an extract from each of the chapters in this thesis has been submitted for publication as 5.

1. D.E. Cooper, P. Ezhilchelvan, and I. Mitrani. High coverage broadcasting for mobile ad-hoc networks. *The Third IFIPTC6 Networking Conference*, 2004.

2. D.E. Cooper, P. Ezhilchelvan, and I. Mitrani. A Family of Encounter-Based Broadcast Protocols for Mobile Ad-Hoc Networks. *Wireless Systems and Mobility in Next Generation Internet: First International Workshop of the EURO-NGI Network of Excellence, Dagstuhl Castle, Germany, June 7-9, 2004: Revised Selected Papers*, 2004.

3. D.E. Cooper, P. Ezhilchelvan, I. Mitrani, and E. Vollset. Optimization of Encounter Gossip Broadcasting in Mobile Ad-Hoc Networks. University of Newcastle upon Tyne *Technical Report Series, CS-TR-921*, 2005.

4. D.E. Cooper, P. Ezhilchelvan, I. Mitrani, and E. Vollset. Optimization of Encounter Gossip Propagation in Mobile Ad-Hoc Networks. *Proceedings of the 13th IEEE International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems-Volume 00*, pages 529–532, 2005.

5. D.E. Cooper, P. Ezhilchelvan, I. Mitrani. Encounter-Based Message Propagation in Mobile Ad-Hoc Networks *Submitted to Elsevier, Ad Hoc Networks*

Contents

Chapter 1

Introduction

Mobile Ad-hoc Networks (MANETs) are networks of nodes that communicate over multi hop wireless links without the support of network infrastructure or base stations. These nodes are also mobile, either self locomotively or through being carried by users or external events. A good summary of the scale of problems posed by communication within MANET can be read in [19], this attempts to cover the vastness of the problem including physical layer, routing protocols, mobility, simulation cross layer research, security and more.

In this thesis we define a protocol whose principal objective is to deliver a message, originating at any node, to all other nodes with high probability. A secondary objective is to minimise, as far as possible, the memory requirements at each node. In fact, what will be defined is not a single protocol, but a family of protocols depending on an integer parameter, τ, the number of encounters on which a node transmits each message. Specifically this protocol is a high coverage broadcast protocol for MANET. Most existing protocols are generally designed to work in low mobility, high density contexts, resulting in connected networks. Our focus falls on the area where these opposite assumptions are also possible: Low density networks with many frequent partitions and high mobility. Those protocols that are developed are commonly tested exhaustively under simulation, however coverage and network usage in the real world may be different due to the assumptions made in simulation. We take our protocol from creation, simulation, optimisation and on to real world experiments to examine its performance.

Mobile devices with wireless network capabilities are all around us. Their numbers are growing, spurred on by rapid progress in wireless communication. The pervasiveness of WiFi (802.11X variants) in PDAs (Personal Digital Assistants) and laptops, and Bluetooth in mobile phones means that many people use these wireless capable devices daily. The mobility of such devices is naturally only limited by the imagination; people, animals, vehicles etc may all carry mobile network capable devices. These mobile devices are capable of being connected to infrastructure based network but also in an ad-hoc mode. These MANET may consist of any number of mobile devices we call *Nodes*. Generally these

devices transmit messages on a single channel and all *Nodes* within transmission range may receive these messages. Naturally a *Node* wishing to communicate with another *Nodes* which is outside its wireless range will have to use an intermediate node to route the message. Thus all devices in a MANET may be routers for the messages of others.

When compared to desktop computer, devices used in MANET are assumed to have limited resources: low memory, low processing power, limited wireless range and limited battery life; these all contribute additional challenges when designing for MANET. The limited processing power of mobile devices means that both applications and networking protocols cannot use overly complex calculation. The constrained memory of mobile devices means that stored information is clearly finite. Therefore the more information a protocol generates and stores, the faster this resource will be consumed, and the sooner data must be discarded.

The specific property of mobility and the ability to form networks, ad-hoc, as and when mobile devices come together opens up a host of new applications. Currently the application of these networks include: Disaster Recovery, Military Operations, Vehicle to Vehicle and intra Vehicle communications, collaborative computing and monitoring of environments which are difficult to access by other means. The success of such collaborative undertakings require reliable communication, group management or even consensus and these depend to a large extent on the provision of efficient multicast [32].

The ad-hoc property of a MANET implies that there is no inherent infrastructure. This may be due to the lack of base stations and central management or simply that these are too expensive to utilise. Thus routing in MANET requires new protocols to be able to cope with arbitrary topologies and configurations of *nodes*. To send a message immediately from one *node* to another (unicast) requires a route to be known in advance (pro-actively) or discovered as needed (reactively) so that the message can be passed *node-to-node* until the destination is reached. If delivery time is not an issue, in MANET, there is also the possibility that the mobile *nodes* wishing to communicate may enter communication range, and thus *encounter* each another, and transmit messages directly avoiding all intermediate transmissions.

The focus of this thesis in on network-wide broadcasting. Network-wide broadcast protocols attempt to propagate messages from a source *node* to all *nodes* within a network. *Network-wide Broadcast* is a major communication primitive required in MANET by both applications and other network protocols. Broadcast protocols are an important foundation for realisation of several middleware functionalities such as replication[4], group management[63] and consensus[100]. Moreover broadcast often forms the basis of a service discovery mechanism. In fact routing protocols such as DSR[42] and AODV[78] along with many others require broadcast to discover routes. Broadcast is also a key construct for many applications such as content distribution: fire fighting [48], USENET on the fly [3], P2P file sharing [61]. In MANET that are especially dynamic, broadcast may be the

best way forward for achieving other network primitives such as multicast.

Recent research into routing both unicast (AODV [78, 80], DSR [42], DSDV[79]) and multicast (MAODV[84], CAMP[26], ODMRP[57], FGMP [17]) has focussed on applying traditional tree based wired network solutions to the MANET environment. Where mobility and network partitions cause this to break down they each have recovery mechanisms but naturally if routes break down too often then the protocols perform poorly or even fail completely.

Under traditional routing protocols, mobility causes a substantial reduction in performance. Coverage is usually heavily affected as routes break more frequently with higher node mobility. This is often because recalculating a route requires flooding the network with a route request. So if every route breaks immediately, the protocol performs worse than the simplest of all broadcast protocols, flooding. In flooding every node transmits every message once, either immediately upon receipt or after a random interval. The protocol keeps minimal state (for detecting redundant receptions) and requires no information on network topology. Flooding has been shown in these cases to be a good starting point for building more robust broadcast and multicast protocols for MANET [73].

So far all protocols we have mentioned assume that the network is connected. But in a MANET it is clear that this will not always be the case as mobility and obstacles to wireless transmission may cause partitions in the network. When the network is partitioned the existing protocols perform poorly in message delivery. In fact in the most extreme case, if the network is partitioned into as many partitions as there are nodes, the message delivery coverage (fraction of nodes that receive a message) will be 0. A different approach is needed to achieve coverage in these partitioned scenarios. One based on a simple store and forward mechanism, where messages are rebroadcast on *encounter* of new nodes, is an approach that we and others have examined.

1.1 Applications of MANET

MANET are suitable for deployment when traditional infrastructure networks are too costly, impractical or even impossible to deploy, or when the network must be deployed rapidly. These scenarios cover a wide range of operational parameters. A common civilian form of MANET is a Vehicular Ad-Hoc Network (VANET), that is formed by mobile nodes attached to vehicles moving on roads. A common application for VANET is road safety and traffic flow management [82, 94]. A VANET may experience a large variation in the number of nodes and density of the network, as they fluctuate with traffic and road network. Speeds in VANET will therefore also range between stationary or slow moving traffic, say at $4mph$ (approx $2ms^{-1}$) to $70mph$ (approx $31ms^{-1}$). There is no reason though that VANETS could not be deployed on aircraft capable of much higher speeds.

Wireless Sensor and Actuator Networks (WSAN) are deployed to simplify the monitoring of hard to observe systems. Often this may require fast response times to deploy countermeasures to the

observed situation, eg reduce temperature if it gets too hot. In [1] we find a good survey of WSAN. A good example of the use of WSAN is in ZebraNET [43], which deploys sensor nodes on Zebras to provide novel studies on Zebra migration whilst simultaneously studying energy and position aware MANET.

Military scenarios are particulary suited for MANET deployment. It may be difficult or impossible to set up an infrastructure network, as enemy operations cause disruption. Examples of military MANET projects include: Near-Team Digital Radio (NTDR)[85] and Joint Tactical Radio System(JTRS)[23]. NTDR provided a prototype MANET radio packet network for the US Army consisting of up to 400 radios and a network management terminal to serve a 20 x 30 km area. JTRS is a software defined radio for voice and data that will be backwards compatible with a large number of other military and civilian radio systems. It also includes wideband networking software to implement full-featured MANETs.

Related heavily to military scenarios are disaster recovery and emergency operations. Disasters, natural or otherwise, may knock out traditional infrastructure networks. At these times a fast deployed MANET can be the ideal solution for providing group communication and collaboration. In [44] a system is proposed which combines MANET, satellite IP network and terrestrial internet to provide a collaborative emergency response system.

1.2 Problem Statement and Goal

For every single broadcast in a MANET there is an Ideal Minimum Propagation Time (IMPT). This is the time from which the source node transmits the broadcast until the last partitioned node comes into contact with a node which can forward the message. For a connected static network $IMPT \approx 0$. For a disconnected network $IMPT > 0$ and grows with the increase in partitions. High speed of mobility may reduce IMPT. IMPT is a property of the network at the instant the message is broadcast.

Khelil [50] defines two terms which we adopt here to describe the connected and the partitioned scenario's that broadcast algorithms fall into. In a connected network a broadcast can be almost instantaneously made to all nodes within the connected space: *Broadcast-in-space*. When the network is partitioned, the only way for a network wide broadcast to occur is for a node from the source partition to travel to and connect with the nodes in the remaining partition, which may take minutes, hours, days or longer: *Broadcast-in-time*.

We present and discuss the properties of a collection of broadcast protocols covering both *Broadcast-in-space* and *Broadcast-in-time* scenarios in Chapter 2.

In an ideal world, under the *Broadcast-in-space* scenario flooding should suffice to achieve perfect coverage. Unfortunately the broadcast storm problem means that perfect coverage may not be

achieved using such a simple protocol[72]. A large number of broadcast protocols have been proposed for avoiding this problem. The broadcast storm problem applies to MANET in the following way. Nodes using WiFi or similar under ad-hoc mode use carrier sense multiple access (CSMA), but no collision detection (CD). This leads to three issues which are collectively called the broadcast storm problem.

1. Because the radio propagation is omnidirectional and a physical location may be covered by the transmission ranges of several hosts, many rebroadcasts are considered to be redundant.

2. Heavy contention could exist because rebroadcasting hosts are probably close to each other.

3. Collisions are more likely to occur because the RTS/CTS (request to send/clear to send) dialogue is inapplicable and the timing of rebroadcasts is highly correlated.

More recent protocols have addressed the *Broadcast-in-time* scenario. There are two key questions which need to be addressed when trying to broadcast across a partitioned network:

1. When should the protocol retransmit broadcast messages?

2. When should the protocol terminate?

Protocols that address this problem space can be split broadly into those which use repeated flooding and those which use a negotiation method to determine when to retransmit. Both types, however rely on the lifetime of a broadcast message as a terminating condition. Effectively they must predict the IMPT and use a lifetime is greater than this to achieve high coverage. Without the ability to know a priori, how long it may take for a message to propagate, within an arbitrarily mobile network,(the IMPT) it is not possible to provide an appropriate broadcast lifetime for all mobility scenarios. If the lifetime is too small and mobility is low, coverage will also be low as the broadcast times out before it covers all nodes. If the lifetime is too high this causes the protocols to continue propagating actions even after maximum coverage is reached: flooding protocols retransmit too many times causing more redundant transmissions; negotiations enter the three way handshake negotiation more than necessary; see Section 2.9.2.

Instead of a broadcast lifetime our approach uses an *encounter* threshold which determines the number of retransmissions made of any broadcast message. In Chapter 3 we model the algorithm and network behaviour to determine an appropriate value based on network size. Naturally as mobility changes the frequency of encounters also changes accordingly and so our retransmissions cease after an appropriate number of retransmissions.

We measure the density of a MANET as the average number of nodes found in an area equivalent to a node's range of wireless transmission. If this value is 1 or below then for a majority of the time each node is alone in a partition of size 1. Without infrastructure or knowledge of when or how

nodes will move, the fastest way of propagating a broadcast is for every node to pass all broadcast messages to every node it meets. Message propagation protocols typically send two types of message, data messages and control messages. In the most extreme case of mobility, nodes move in such a manner that only *a single message* may be successfully received upon an encounter (the case where no message may be received is identical to no contact being made at all); upon detection of an encounter, the message that is transmitted must be a data message for it to be of any real use.

Existing broadcast protocols for MANET have been developed that work within a range different partition and mobility scenarios, but none to our knowledge that address this combination of extremes. It is with this worst case scenario in mind that we propose *Encounter Gossip* our encounter based propagation protocol which we introduce in Chapter 3.

1.3 Primary Contributions

The main contributions of this thesis can be summarised as follows:

1. **Encounter Gossip**

   We introduce Encounter Gossip (EG) which is an encounter based broadcast protocol for MANET. Broadcasts are retransmitted upon the encountering nodes. Nodes stop retransmitting a broadcast message when they have made τ retransmissions.(Chapter 3).

2. **Mobility-independent τ Estimate**

   We model the encounter rate of nodes in a MANET and use this to generate a Mobility-independent estimate for the value of τ that achieves high coverage (Chapter 3, equation 3.6).

3. **α-*reduction* policy**

   Multiple retransmissions increases redundancy so we investigate improvements to the protocol aimed at reducing the number of redundant transmissions (Chapter 4). In particular, we introduce our α-*reduction* policy which is both simple and effective, and hence is worth implementing.

4. **Efficiency of Multiple Messages**

   In generalizing the protocol to multiple parallel broadcasts, we identify two extra causes of redundancy that can occur due to the effects of high mobility when transmitting large numbers of messages. We investigate the causes and introduce measures to minimise these effects (Chapter 5).

5. **Real World Evaluation**

   In order to provide a realistic evaluation of EG we implement for mobile devices and perform a

set of experiments in a challenging real-world environment (Chapter 6).

1.4 Thesis Outline

The outline of the thesis is as follows. In Chapter 2 we show a taxonomy of broadcast protocols and discuss the related work for both *Broadcast-in-space* and *Broadcast-in-time* scenarios. We provide comparisons with our own approach and highlight the merits and limitations of the varied approaches.

In Chapter 3 we introduce our protocol, *Encounter Gossip* (EG). We derive a formula for calculating a mobility independent estimate for τ. We then provide quantitative performance results which show that EG provides high coverage for a wide range of densities and mobility speeds.

In chapter 4 we explore ways of reducing the number of redundant transmissions made by EG. Several policies are evaluated through simulation and their relative merits discussed. The key optimization discovered is our own α reduction policy, which significantly reduces the number of redundant transmissions.

In Chapter 5 we identify two serious performance issues which occur when multiple broadcasts occur in parallel: Encounter Redundancy and Departure Redundancy and introduce modifications to EG to overcome these issues. We demonstrate that a considerable improvement in coverage can be achieved, particularly when comparing high speeds and low densities. We then examine a new performance measure to evaluate the possibility of achieving FIFO Order without recovery mechanism. We show that this is indeed a distinct possibility.

In Chapter 6 we deploy EG into a, real world environment, a university office complex, which provides significant barriers to wireless radio transmission. We use PDAs and a laptop to perform a real world experiment with a large group of users who's mobility is determined only by their normal daily routine. We show that even in this difficult scenario EG is capable of delivering high coverage. We continue by capturing the mobility pattern of the real world and performing simulations to provide quantitative evaluation of EG under ideal network conditions. Finally we compare and contrast our observed real world mobility with the work of Chaintreau et al [15].

In Chapter 7, we summarise our contributions and outline further work which may become the direction of future research.

Chapter 2

Related Work

We commence this chapter with a brief set of definitions that are used throughout this work. We give a brief history of routing as has emerged into MANET whilst holding onto wired network strategies.

We then present a taxonomy for broadcast protocols in MANET. We provide short reviews of representative protocols and outline their strengths and weaknesses.

2.0.1 Definitions

In this thesis we use the following list of some important terms.

Node

A device capable of sending, receiving and routing messages within the network, in our case unless otherwise stated these are wireless.

Transmission

The sending of a message through the air, a MAC layer broadcast. These messages may be overheard by any node within the wireless range.

Multicast

The network primitive, used to send a message from a node or nodes who are members of a group, to all other nodes who are members of that group.

Broadcast

The network primitive, used to send a message from one node to all nodes in the network. Broadcast is in fact a specific instance of multicast, where the multicast group is the entire network.

Neighbourhood Knowledge

Many Ad Hoc routing protocols leverage neighbourhood knowledge in some way. This is achieved by some kind of $HELLO$ message which contains a node id, which announces a nodes

existence to one another. In 802.11 radios, beacon frames are sent periodically by the MAC layer, these could be leveraged at the network layer for neighbourhood knowledge. The simplest form allows a node to know only his immediate, *1-hop* neighbours. By transmitting a list of all *1-hop* neighbours in a $HELLO$ message or piggy-backed on a data message, nodes can be made aware of their *2-hop* neighbours, at the cost of additional/larger messages. Naturally 3+hop neighbourhoods can also be maintained but these become prohibitively expensive and as such most algorithms use *1-hop* or *2-hop* neighbourhood knowledge.

Density

Throughout the literature several parameters are used to control the density of a MANET:

- Wireless range of a node
- Number of nodes
- Size of the simulation area.

People often use the average number of neighbours as parameter of density although it is important to recognise that this is actually a result value that is effected by wireless range, number of nodes, simulation area and mobility of nodes. The average number of neighbours reflects the density observed by nodes. Since much research provides only for a subset of these parameters we calculate global density as the average number of nodes within a circle of radius equal to the wireless range. This measure allows us to compare density with work that provides no density figure. This global density is subtly different from the average number of neighbours since a global density of 0.5 means that a randomly chosen circle of wireless range will contain a node with probability 0.5. On the other hand with average number of 0.5 neighbours, a randomly selected node will have an average of 0.5 neighbours.

Assuming symmetrical wireless ranges(r) that are identical for each of N nodes, we calculate network density as:

$$Density = \frac{N * \pi * r^2}{Areaof Simulation}$$

For example with a network of 64 nodes, a wireless range of $179.80m$, and a network area of a square kilometre ($1000000m^2$) we get a density of 6.5.

Work has been done to find the ideal node density for communication [83], they perform a number of simulations under GloMoSim in which they determine for a stationary network, 7 or 8 neighbours per node should be optimum though, as mobility increases, higher densities would increase throughput.

We examine considerably lower densities and develop a protocol that can perform well at these densities.

2.0.2 Traditional Routing

Recent research into routing both unicast (AODV [78, 80], DSR [42], DSDV[79]) and multicast (MAODV[84], CAMP[26], ODMRP[57], FGMP [17]) has focussed on applying traditional wired network solutions to the MANET environment. Where mobility and network partitions cause this to break down they each have recovery mechanisms but naturally if routes break down too often then the protocols perform poorly or even fail completely.

Under traditional routing protocols mobility causes substantial reduction in performance. Coverage is usually heavily affected as routes break more frequently with higher node mobility. This is often because recalculating a route requires flooding the network with a route request. So if every route breaks immediately, the protocol performs worse than flooding.

It has however been suggested that mobility could in fact increase the throughput capacity of Ad-Hoc networks, if one is willing to sacrifice for a larger latency [28]. If nodes are able to delay transmission to reduce the number of hops that are needed to reach a destination then the network is able to make fewer transmissions on the whole reducing congestion and allowing more useful throughput.

2.1 Reliability

The requirements for reliable broadcast have been traditionally described by three properties [32]

Validity If a correct process broadcasts a message m, then all correct processes eventually deliver m.

Agreement If a correct process delivers a message m, then all correct processes eventually deliver m.

Integrity For any message m, every correct process delivers m at most once, and only if some process broadcasts m.

Most research into reliable broadcast in MANET does not consider the possibility of incorrect processes. The only two examples we are currently aware of are Burmester [9] and Bhandari [6] who both examine reliable broadcast with Byzantine failure. Instead we define a process as correct if it is never permanently partitioned from the network, as in this case propagation of messages is doomed to failure.

There are two distinct approaches to achieving reliable broadcast and multicast in MANET, Deterministic and Probabilistic. The deterministic approach is favoured in traditional wired networks as it provides atomic guarantees of message delivery. Traditional Reliable Broadcast requires this deterministic approach to achieve. Several deterministic reliable broadcast/multicast protocols have been proposed for MANET [74], [27], [99].

Naturally this approach consumes considerable overhead resources to achieve reliability. The deterministic approach has been shown to scale poorly and a probabilistic approach of fighting fire with fire scales much better [29]. Research into deployment of reliable broadcast in large scale networks has also suggested that scalability of performance is more desirable than strict adhesion to these properties [49].

The probabilistic approach is therefore arguably more suitable for MANET; it provides guarantees that a broadcast will be successfully delivered with some probability P. These protocols are commonly simpler and use less resources. Naturally cases exist where stronger guarantees that may include timing may be required, however it seems unlikely these can be delivered in the extreme environment we propose without massive overhead that will reduce scalability considerably.

2.2 A Measure of Reliability for Probabilistic Broadcast in MANET

The traditional definition of reliability is impossible to achieve probabilistically. However Δ-Reliable Broadcast [25] is proposed as a good measure of how reliable a probabilistic protocol may be in MANET. Where Δ represents the allowed variability of the reliability.

ΔReliable Broadcast which is satisfies the same properties as before with probability *psi*. They redefine Agreement as follows.

Δ-Agreement If a correct process delivers message m, then eventually at least a fraction ρ of correct processes deliver m.

So the reliability measure of a protocol is defined by $\Delta = (\psi, \rho)(\psi, \rho \in [0, 1])$. This requires that a fraction ρ of nodes deliver m with probability ψ. Unfortunately once calculated this measure only applies to the exact parameters of a system in which it was calculated. If any parameter (mobility, density radio range etc.) were to change the reliability measure would be invalid, and in many cases wildly inaccurate. Nevertheless this could be a useful measure to be able to directly compare protocols across identical simulation parameters.

2.3 Improving the reliability of broadcast

Since flooding is a commonly used protocol used for broadcast itself and as a primitive by many other protocols it is important to optimise its performance. Flooding suffers from the broadcast storm problem [72], there are several methods in the literature that have been developed with which to combat the problem. The most effective of these limit contention and collision by reducing the number of nodes that rebroadcast the flooded message. A general approach used in most solutions follows. When a node $Node_i$ receives a new message m, set a timer $t(m)$. When that timer expires, decide whether or not to retransmit.

Timer

This timer determines the waiting time until the forwarding decision. Three goals that may be achieved in this waiting period are outlined here. The primary goal of the delay is to reduce likelihood of collision. Random delay is a common tool used in many protocols. A secondary goal may be to influence the timer based on expected coverage. If a node has a large number of neighbours, the delay might be shorter than nodes with fewer neighbours. This would improve the chances of higher area being covered by the next transmission. The third goal is simply to allow ample time for the decision process to complete.

Decision Process

The majority of decision processes are designed with the aim of reducing the probability of transmission if this is likely to be redundant. Many of the broadcast protocols we present later are named after the design of the decision process. The simplest of all decision processes is gossip. By retransmitting with probability P, the number of retransmissions can be reduced significantly. Naturally reducing the number of broadcasts will reduce collisions, if P is selected carefully this can enhance the reliability of the protocol.

2.4 Flooding for Multicast and Broadcast in MANET

Recent literature has suggested that flooding may itself be a better starting point for high coverage broadcast and multicast MANET routing than traditional methods. In a study comparing the performance of flooding, ODMRP and MAODV under both ns2 and GloMoSim [73], flooding is shown to outperform the multicast protocols especially when traffic load and number of sources increase. In fact flooding still achieves over 70% coverage when the number of senders and mobility are high when the multicast protocols show very poor performance.

Flooding is also proposed as a solution to MANET multicast by Ho et al in [37]. Ho illustrates that under extreme mobility even flooding is not sufficient. He suggests research into persistent flooding, where messages are stored and repeatedly flooded at intervals. He I identifies three key issues which should be addressed in future work:

1. **Packet State** : For robustness, some packet state is likely to be required, this information should be kept to minimum.

2. **Flooding Overhead** : It is non trivial to measure and quantify. Multiple flooding waves of the same packet may occur and high overheads may be incurred by re-flooding the packet.

3. **Collision Loss** : The broadcast nature of flooding leads to packet loss through collision, mechanisms for coping with this are needed.

This provides us with a strong motivation for our protocol, Encounter Gossip, a persistent form of flooding which has minimal packet state and we optimise to minimise overhead and collision loss. We introduce this fully in Chapter 3 and our performance metric, Redundant Broadcasts and optimisation using RAD in Chapter 4.

2.5 Taxonomy

We use Khelil's taxonomy for broadcast protocols in MANET; he identifies application sensitivity to delay and connectivity of network as two orthogonal values that can be used to provide a useful basis for taxonomy [50]. His classification of broadcast algorithms in connected networks combines the very thorough discussion of broadcasting techniques is given by Williams et al in [104], and also work by Yi et al [108] and Heissenbuttel et al [35]. We add to this several other related works and our own observations and discussions of the pros and cons of the broadcast protocols.

We follow this taxonomy that takes into consideration the MANET characteristics on the one hand and the application requirements on the other. Applications can be split broadly into those which are delay critical and those which are delay tolerant. MANET can be split into those which are mostly connected or those which are frequently partitioned.

Delay-critical applications broadcasting in partitioned networks (Q1 in Fig. 2.1) will only achieve low coverage as they may only deliver to the subset of nodes in their partition. Since high values of coverage are desired in broadcast, we conclude that solutions in Q1 are impossible, unless coverage within only the currently connected partition is acceptable.

As we noted earlier, Grossglauer provides foundations for using delay tolerance in connected networks (Q4) to achieve greater throughput. As far as we can establish, however no protocols have been developed that utilise this capability. The gain in network capacity comes at the cost of unpredictable end-to-end delay.

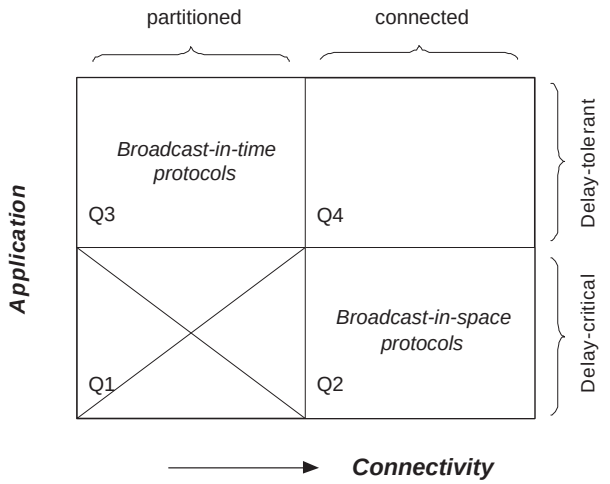

Figure 2.1: Taxonomy of related work

Khelil divides the current protocols from the literature into two classes; those that fit in Q2: Broadcast-in-space; and those that fit into Q3: Broadcast-in-time [50]. If a MANET is not partitioned, a broadcast will reach the entire space, Broadcast-in-space encompasses a large number of broadcast protocols which are designed for connect MANET, examples of these are in section 2.6. These are commonly based on flooding and apply various optimisations to increase coverage or reduce redundancy. These protocols are suitable for delay sensitive applications.

If a MANET is partitioned into one or more partitions, then a broadcast may only reach all nodes in all partitions given enough time for those nodes from separate partitions to propagate the broadcast. Broadcast-in-time protocols provide store and forward semantics. Messages are stored in nodes and propagated at opportunistic encounters , examples of these are in section 2.6.

Since mobility dictates the speed at which partitions are traversed transmission delay may be anything from seconds, to days or more, depending on the specific scenario. This class of protocols are appropriate for delay-tolerant applications in highly partitioned networks (Q3). Applications should tolerate delay values that allow for the *Ideal Minimum Propagation Time*(IMPT) of the MANET. Broadcast-in-time protocols perform well in highly partitioned networks. Unfortunately, performance of such protocols in connected MANET typically degrade due to higher message overhead. Our approach uses techniques from both classes and performs well in both connected and partitioned networks.

2.5.1 Pocket Switched Networking

A new term Pocket Switched Networks (PSN) has been coined to define the Broadcast-in-time type network that we are addressing where nodes are frequently disconnected for unknown periods of time

[15]. Work at Cambridge has examined the behaviour of PSN from earlier experiments in university campuses and their own data from a conference experiment to discern the trend of connectivity in PSN [15, 39]. They argue that the performance of some MANET routing algorithms is highly dependant on the mobility patterns that have been used under simulation and show that this has considerably different characteristics to those found in real world environments. Crowcroft's Pilgrim proposal suggests work on a new network stack to support PSN and human mobility [21], initial proposals are to use the algorithm we describe in Chapter 3. We further explore the implications of this work with experiments in Chapter 6

2.6 Broadcast-in-space Protocols

Broadcast is a necessary network primitive for both application level communication and routing in networks. In a MANET it is typically performed by flooding. This is a simple topology free operation in which each node simply rebroadcasts any message which is not a duplicate. In denser networks this is extremely wasteful and much of the literature has discussed ways to optimise flooding.

Whilst trying to reduce the redundancy in flooding we also want to increase its reliability. If one can increase the reliability of a flood at the network layer, a reduction in the number of times flooding is invoked at the network layer can occur. Such that an application that requires 99% reliability and as such might invoke a 90% reliable broadcast, would only need invoke a 99% reliable broadcast once.

In this section we provide descriptions of a number of protocols and their perceived strengths and weaknesses. The classification structure we use in shown in figure 2.2.

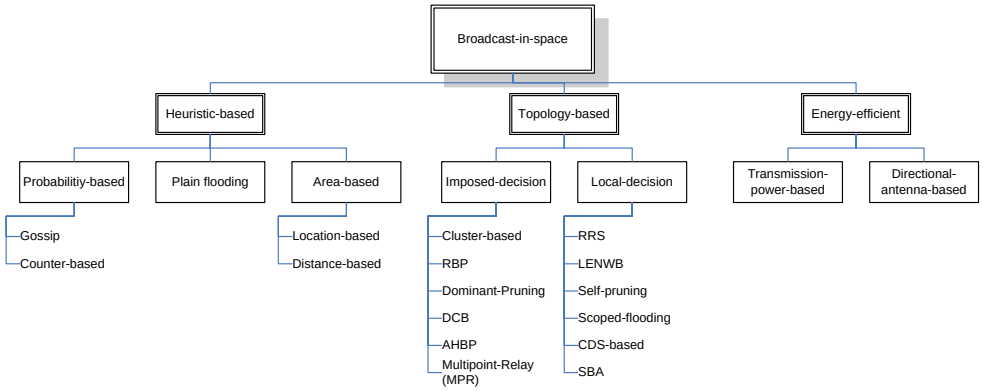

Figure 2.2: Taxonomy of Broadcast-in-space protocols

There are two further classes that all the following protocols fall into: Non-adaptive and Adaptive protocols. Naturally adaptive protocols adapt their behaviour to the environment of the MANET, where non-adaptive protocols do not.

2.7 Non-adaptive Protocols

2.7.1 Heuristic Based

Probability-Based

Gossiping and counter based protocols have been presented as two probabilistic approaches in the literature.

Haas et al use gossip to optimise Flooding[31]. They argue using percolation theory and then illustrate using simulation that the use of gossip with flooding gives coverage a bimodal distribution similar to that observed in bimodal multicast [7]. The algorithm $Gossip1(p, k)$ works as follows: For the first k hops the message is forwarded with probability 1, this helps avoid early extinction of the broadcast. For the remaining $hops > k$ the message is rebroadcast with probability p. They show that $Gossip1$ can be used to effectively broadcast messages with a good reduction in cost and as a benefit provide a probabilistic statistic to determine the reliability of their algorithm.

Gossip based ad-hoc routing, [59] A probabilistic approach based on *phase transition* is proposed by Sasson et al [86]. They show that the bimodal properties described by Haas et al, are only achievable when network conditions are ideal and network size is around 250 nodes, for lower sizes 25 and fewer nodes, the coverage behaviour is almost linear. Through experiments with speeds of $0 and 5 ms^{-1}$ the authors show that at very high densities, ($Density = 19$) coverage performs with a bell curve peaking at 100% coverage using propagation probability 0.1. In contrast at lower densities (D=2.5, 7.1) coverage is linearly proportional to propagation probability peaking at less than 60% even when propagation probability is 1.

Cartigny et al discuss an adaptive gossip protocol which includes 3 enhancements to gossip which modify the probability with which a node will transmit. A density aware scheme and one which prefers nodes further from the source (preferably Border Nodes) and a combination of these are discussed [12].

A counter based scheme [72, 104] uses a RAD during which the number of duplicate messages received are counted. The message is only retransmitted if the count is less than a specified threshold value otherwise the message is dropped. The key idea being that those nodes that receive most duplicates are unlikely to be able to reach any new nodes with a retransmission.

Area-Based

Area-based protocols estimate the additional area of coverage that could be achieved by a node retransmitting a message. If this is higher than a given threshold then the retransmission is allowed otherwise it is suppressed. In the literature, two area-based schemes have been proposed in [72]: Distance-based and location-based techniques.

A distance-based forwarding decision is made, using the distance of the receiving node to its neighbours from which it has received the message. The shorter this distance, the less likely the node will be able to provide any additional coverage. The closer to maximum wireless range the larger coverage is likely to be achieved. A threshold value somewhere between these extremes needs to be chosen. The authors of [72] proposed to use signal strength to approximate distances.

The location-based approach estimates the additional coverage area more precisely. Rather than using distance, it is proposed that each node know and transmit it's location. This would allow accurate calculation of additional area covered at the expense of larger hello messages containing location data, and the power and technology requirements of location devices (perhaps using GPS) on a mobile device, which is hard to realise in many MANET applications.

2.7.2 Topology-Based

We identify two sub-classes: The local-decision- based approaches and imposed-decision-based approaches. In local-decision-based approaches (reactive approaches), each node determines on its own, whether or not to propagate a broadcast message. However, in imposed-decision-based approaches (proactive approaches), the forwarding decision is imposed by other nodes such as the previous relay node or the cluster-head.

Local-Decision-Based

The basic idea of these approaches is that a node exploits neighbourhood connectivity and history of the nodes that the message has already visited, in order to decide on its own, if it is a forward node or not. The authors of [106] proposed a generic scheme that covers most existing local-decision-based approaches. The scheme is based on two conditions, namely on neighbourhood connectivity and history of the nodes already visited. Each node builds information about its neighbourhood by exchanging neighbourhood information with its 1-hop neighbours, by means of periodic HELLO messages. Information about a node's property, such as ID or node degree, and a list of nodes already visited is added to the broadcast messages. Based on this information a node decides whether or not to propagate a message.

The simplest local-decision-based method is flooding with *self-pruning* [60] also known as *neighbour coverage scheme* which uses 2-hop neighbourhood knowledge. The sender piggybacks a list of its 1-hop neighbours on each transmitted broadcast message and a receiving node only propagates the message immediately if it can cover some additional nodes to those of the sender.

The scalable broadcast algorithm (SBA) [77] uses the same forwarding strategy as the neighbour coverage scheme, with the following two main differences. First, nodes insert the list of their 1-hop neighbours to HELLO beacons and not to data messages. Secondly, nodes do not forward

immediately, but initiate an RAD. For each neighbour that propagates during the waiting period, the node re-calculates its additional coverage. When the RAD expires and the additional coverage set is not empty, the node is a forward node. In SBA, RAD is not constant, but it is adapted depending on a node's relative neighbour degree. Scoped flooding [73, 98] is a variant of the SBA protocol. Upon the expiration of RAD, a node propagates the message, if more than a fixed ratio of its neighbours are still not covered. The authors of [98] propose to use 15% as the ratio value.

Scoped flooding behaves like SBA for 0% ratio value. The lightweight and efficient network-wide broadcast (LENWB) [93] is similar to the neighbour coverage scheme, but nodes acquire 2-hop neighbourhood information by the periodic sending of HELLO beacons that contain the list of 1-hop neighbours. Upon receiving a broadcast message from a sender, the receiver computes the coverage of its 1-hop neighbours that received the message and have a higher node degree. Only if all receiver's neighbours are covered by higher degree nodes, is the forward cancelled.

Several of the local-decision-based approaches are based on connected dominating sets (CDS). One such algorithm, requires 2-hop neighbour information [107]. A node belongs to the dominating set if two unconnected neighbours exist. Only nodes that belong to the CDS forward the message. Unlike [107], in [92] 1-hop neighbour information is sufficient if nodes are aware of their positions in order to determine if two neighbours are connected. Under the assumption that each node knows its accurate position, connected dominating sets and the concept of planar subgraphs are used in Relative Neighbourhood Graph (RNG) relay subset flooding [11] to reduce the communication overhead for broadcast messages.

RNG relay subset flooding (RRS) builds a RNG traversal graph ([95])for the entire network. A RNG traversal graph is somewhere between a minimal spanning tree and a Delaunay Triangulation [58], thus it is fully connected, and has the benefits of both. This graph specifies whether nodes within communication are RNG neighbours with one another. Each node maintains knowledge of its own neighbours and RNG-neighbours and also those of its neighbours, thus keeping a complete 2-hop neighbourhood knowledge. RRS uses an interesting scheme when hearing a broadcast, which we shall call *RRS-bcast*. Using this information for each broadcast, a node decides to retransmit if he has RNG-neighbours that the transmitting node does not: ie he can reach new node(s), otherwise he drops the message. Effectively this means that for broadcasts that originate from different sources may have different forwarding nodes, thus distributing the cost of the broadcast more fairly. They additionally optimise the protocol with a variation on the random assessment delay (RAD [104]) before transmitting a broadcast. If during this period a duplicate is heard, then the comparison of neighbours is made again and retransmission only occurs if the node can reach new neighbours.

RRS is shown to achieve higher coverage than MPR [81] (see Imposed-Decision-Based Approaches) over a range of densities and scenarios. The densities chosen are however very high, from 10 to 60 and the effects of mobility are not considered.

RRS effectively reduces the cost of flooding the network but the cost of computing the RNG is not calculated. If mobility is introduced then the RNG graph would naturally need to be recalculated which would be expensive. The broadcasting technique *RRS-bcast* can however be used ignoring RNG altogether and may be a useful method for reducing redundancy.

Imposed-Decision-Based Approaches

The basic idea is that each node selects a subset of its 1-hop neighbours for forwarding the message, such that all 2-hop neighbours can be reached by this subset.

In multipoint relay (MPR) [81], nodes insert the list of their 1-hop neighbours into their HELLO beacons, so that nodes are aware of their 2-hop neighbourhood. The sending node selects forwarding nodes from its 1-hop neighbours, so that all 2-hop neighbours are covered by the set (the selection rule is defined in [81]). Nodes piggyback the forwarding list in their HELLO beacons. Only nodes in this list forward broadcast messages.

In dominant-pruning [60] also acquire 2-hop neighbour knowledge using HELLO beacons, and senders select the designated forwarders using the same MPR rule. Unlike MPR, receivers select the forwarding set depending on MPR selection rule and additionally depending on the knowledge of which neighbours have already been covered by the senders broadcast. The forwarding set is selected from the 1-hop neighbours that are not neighbours of the previous relay. The forwarding list of the same node may therefore differ from message to message. The forwarding list is piggybacked on the broadcast message. In [64], Lou and Wu present total dominant pruning and partial dominant pruning, two improvements that utilise neighbourhood information more efficiently. The ad hoc broadcast protocol (AHBP) [76] is similar to MPR, but piggybacks the forward designation onto the broadcast message. Nodes that receive a broadcast message from a node that is not listed as a neighbour, always forward the message. In cluster-based schemes, the decision is imposed by the cluster formation algorithm. The general idea of cluster-based schemes has been introduced by Li et al. in [72]. The proposed scheme in [72] assumes that clusters have been formed in the MANET and are maintained regularly by the underlying cluster formation algorithm. It also assumes that the cluster head's forwarding covers all nodes of that cluster. The cluster-based scheme proposes that only cluster heads and gateway nodes (nodes that can communicate with nodes from other clusters) forward the broadcast messages using any broadcast technique such as gossiping. Further cluster-based broadcast protocols have been defined in [54] [109].

In [65], the authors propose a broadcast algorithm, called double-covered broadcast (DCB) which uses 2-hop neighbourhood to create a CDS of the network, and requires each node to be covered by two broadcasts. The sender selects forwarding nodes in such a way that first the sender's 2-hop neighbours are covered and secondly the sender's 1-hop neighbours are either a forward node, or a

non-forward node but covered by at least two forwarding neighbours. The retransmissions of the forward nodes are received by the sender and serve as an acknowledgement. If the sender does not detect all its forwarding nodes' retransmissions, it will resend the packet until the maximum number of retries is reached. Simulation results show that DCB provides good performance for a broadcast operation under a high transmission error rate environment. In dense networks this reduces the number of broadcasts considerably, when compared to flooding, whilst also achieving very high coverage. Densities explored are 7.27 up to 24.24, which are relatively high. At the lowest of these densities the figures provided suggest that coverage will tail off sharply at lower densities. The high delivery degrades linearly when speed increases, although all tests are done with minimum speed of 0 so average speed will be much lower than the maximums shown.

RBP: Robust Broadcast Propagation in Wireless Networks addresses low power networks of non uniform density [90] [91] The key optimisation of this protocol is the use of local density information, and neighbours recognising 'important' links. In this protocol nodes listen for rebroadcasts from $1 - hop$ neighbourhood as implicit acknowledgements. A percentage of neighbours must respond or the broadcasting node will retry x times. Both the threshold percentage and retries x are a function of node density. If a node hears a broadcast twice from a node it will unicast an explicit Ack to the broadcaster. In the case that number of neighbours who have acked the message is greater than those who have not; the message is unicast to the non acking nodes. An optimization to this is also introduced which has nodes recognise links which are regularly first with transmitting messages and communicates to them that they have an 'important' special relationship. These important links means that more retries will be made if needed to ensure these nodes maintain them.

Whilst each flood is more costly with RBP, the effective cost of their method is cheaper than standard flooding where multiple floods are needed to achieve the same coverage.

Overall RBP is shown to achieve a coverage of 0.99 with a cost that is on average less than half that of increasing the number of flooding waves to achieve the same coverage.

It is likely however that this when mobility is introduced links will fail and neighbourhoods will change repeatedly. This will increase the number of retransmits and unicast messages and degrade the performance of the protocol. However the comparison against the cost of increasing flooding to achieve the same level of reliability is a very useful metric.

2.7.3 Energy-Efficient Approaches

The authors in [35] classify the existing power-efficient broadcast techniques into transmission-power-based approaches and direction-antenna approaches.

Transmission-Power-Based

Since radio propagation follows the inverse square law, it is imperative that to reduce the distance nodes transmit in order to reduce the power used. The inverse power law states that to double the transmission radius requires four times the power.

To increase this efficiency, these protocols adjust the radio range to realise an efficient network-wide broadcast. In [103], the authors proposed a broadcast incremental power (BIP) algorithm that constructs a tree starting from the source node and adds a node in each step, which is not yet included in the tree, but which can be reached with minimal additional power from one of the tree nodes. [14] considered the minimum energy broadcasting problem and proposed a localised protocol, where each node only requires the knowledge of its own position and those of its 1-hop neighbours. The algorithm presented in [46] constructs a static routing tree, which maximises network lifetime by accounting for residual battery energy at the nodes. The algorithm however, does not really maximise the possible network lifetime, if nodes are mobile. [102] presented a distributed topology control algorithm, which extracts network topologies that increase network lifetime by reducing the transmission power.

Direction-Antenna Approaches

Directional antennas are used to improve the performance of broadcasting by reducing interferences, contention, etc. It was shown in [101] that MAC protocols, which utilise directional antennas can improve the performance of broadcast traffic in ad hoc networks. In [38], each node is assumed to have a beam-width of 90° and packets are only forwarded in the 270° direction from that in which the packet arrived. If nodes are aware of their neighbourhood through HELLO messages, nodes may explicitly send the packet to nodes that are farthest from the current node. In [13], directional antennas are used to transmit broadcast packets to all neighbours in a connected planar subgraph of the complete network graph, namely the relative neighbourhood graph.

2.7.4 Discussion of Non-adaptive Protocols

The following discussion provides a summary of the findings of several works researching the performance of Broadcast-in-space broadcast protocols. Heuristic approaches are examined in [104]. Topological approaches are also examined in [104] and [108]. A comparison of self pruning techniques is covered in [22]. Power efficient approaches are compared in [45] and directional antenna protocols are examined in [47].

Non adaptive broadcast protocols are designed to apply to specific sets of network parameters. When they are tested within these ranges they show good coverage and efficiency. When network conditions deteriorate from the ideal, so the efficiency and coverage also fall. Heuristic protocols in

this class use fixed thresholds to reduce the number of redundant transmissions. In the case that the number of redundant transmissions increases, then the fixed heuristic may not be sufficient to quash the extra transmissions. On the contrary, if there are fewer than the expected number of redundant transmissions then too many transmissions may be suppressed and reduce coverage significantly. This sensitivity to their threshold value severely limits the applicability of such schemes.

Topology based protocols in this class, whether local or remote decision based, show high coverage and low redundancy for stable topologies. However in high mobility scenarios they suffer high overhead caused be frequent need to recalculate new topology views. Some of the topology-based protocols that make use of CDS or cluster techniques do not consider the propagation of the nodes that have been traversed. This could lead to nodes being abused and lead to early power supply depletion. In other cases these protocols may instead allow nodes to enter sleep states without affecting network operation, thus prolonging network operation. Naturally a careful balance between these needs to be achieved.

Energy efficient approaches may build efficient structures but these come with the high cost of computation. When this is used for a static network, many savings can be made in power used. In dynamic, mobile network this approach suffers considerable overhead caused by recalculating the structure repeatedly. When mobility is high, this overhead will make it impractical as the structure breaks too frequently. Our research does not limit nodes to being devices with special capabilities such as control over antenna direction or radio power and therefore transmission range. For these reasons we no longer consider energy efficient approaches in this thesis.

The most simple flooding improvement scheme is Gossiping. It is topology-independent and although gossiping does not consider the nodes previously visited, it does not abuse certain nodes by forwarding, because of its probabilistic nature.

2.8 Adaptive Broadcast-in-space Protocols

Some non-adaptive Broadcast-in-space techniques have been adapted to local MANET characteristics. The basic idea of adaptive topology-based approaches is to manage node mobility better, for the purpose of avoiding stale topology information [96, 105]. Adaptive heuristic-based protocols however, adapt the heuristics to the number of neighbours [96, 12].

Adaptive Topology-Based Protocols

In [105] authors proposed to use two different communication ranges in order to cope more efficiently with mobility. One range for the topology management (determination of forwarders) and another for the data transmission. They recommend selecting a shorter range for topology management and to adapt the difference between the two ranges to node mobility, which requires speed information.

They further proposed a mechanism to ensure consistency between the different views of different nodes on the network. In [96], the authors proposed one adaptive topology-based scheme, called the adaptive neighbour-coverage scheme. The authors adapted the neighbour-coverage scheme by dynamically adjusting the HELLO interval to node mobility reflected by neighbourhood variation, so that the required 2-hop topology information is more accurate. Despite these optimisations, the adaptive topology-based schemes still have the main drawback that neighbourhood information may be inaccurate in congested networks.

Adaptive Heuristic-Based Protocols

In [96] the authors also proposed two adaptive heuristic-based schemes, called adaptive counter-based (ACB) and adaptive location-based (ALB). By means of simulations, the authors derived the best appropriate counter-threshold and coverage-threshold for ACB and ALB respectively, as a function of the number of neighbours. The authors showed that these adaptive schemes outperform the non-adaptive schemes and recommend ACB if location information is unavailable and simplicity is required. Cartigny et al. [12] adapted the forwarding probability of gossiping to the local number of neighbours n and called their adaptive scheme stochastic flooding (STOCH-FLOOD). Nodes use the following forward probability: $p = min(1; 11/n)$.

Discussion of Adaptive Protocols

The performance of Adaptive heuristic and topology based protocols is higher than that of their non-adaptive counterparts. It is clear that the ability to modify the protocol used to the current network conditions improves the overall performance of the protocol. The ACB, ALB, adaptive neighbour-coverage and stochastic flooding support a broad range of node densities and speeds, however they show poor coverage in partitioned networks.

2.9 Broadcast-in-time Protocols

MANETs by their very nature will have nodes that are from time to time disconnected. A network may be separated into several disjoint partitions containing one or more nodes, ie. nodes in one partition are unreachable by nodes from another. As the density of a network falls and/or its mobility increases, the network will experience more partitions. In one extreme, a network of say three nodes, where $Node_1$ is permanently partitioned from $Node_2$ by a large distance, messages may only be passed between them by a carrier node, $Node_3$.

Example applications that use these properties fall into the category of Delay Tolerant Networks (DTN). An Approach to Interplanetary Internet [8] discusses a similar model to this, proposing a new

network stack to ignore the inherent temporal displacement that may have to take place to achieve such communication. Another example application and protocol stack for DTN is given in Data MULES [88], where *mules* (people, vehicles, animals) carry data from remote sensors back to access points. In this case, high coverage is no longer trivially achievable by one off transmissions as in simple flooding; we need a broadcast-in-time approach. The main research focus of Broadcast-in-time protocols is development of broadcasting strategies that effectively exploit the mobility of nodes to achieve high coverage, despite network partitioning.

The discovery of neighbours is a requirement for developing broadcast strategies for the broadcast-in-time approach. Periodic beacons must therefore be made announcing the node's presence. If one were to consider the amount of energy used by beaconing, there may be a considerable expenditure of energy. This necessary cost is generally considered to be acceptable in MANET. Reducing the cost of beaconing is an active area of research for Wireless Sensor Networks (WSN), for example [24] in which the authors propose an efficient node discovery mechanism.

Store and Forward methods are key to traversing partitions; after transmitting messages they are stored and retransmitted on encountering unknown nodes: This may be compared to the spread of an epidemic disease. Whilst gossip in wired networks is commonly termed an epidemic algorithm, it is the encounters that occur between nodes that are more similar to people meeting one another to spread an epidemic infection. When a node comes into contact with another it 'infects' it with the message and thus it also attempts to propagate the message. These epidemics may have any number of controlling factors, for example, limiting the lifetime of a message will reduce the number of times the message is transmitted at a possible reduction to the coverage achieved. These epidemic algorithms when used in MANET may also be described as opportunistic networking as they attempt to bridge these gaps between disconnected networks using any opportunity of new contact to further propagate messages.

For example Vahdat and Becker introduce an epidemic unicast protocol [97]. Their protocol uses a hop count transmitted along with each message to limit how many times the message will be propagated. Each message is stored in a buffer and on meeting a new node messages are exchanged via an anti entropy negotiation , and hop counts incremented. Once a message reaches its hop count threshold it is no longer propagated to intermediate nodes. A message with hop count 0 may thus only be delivered to its destination. They show through simulation that eventual delivery of packets is 100% successful. This is highly significant as it is similar to the technique we use in our broadcast protocol. The anti entropy session comprises of 3 parts: i) ADV advertisement which messages a node possesses. ii) REQ request for missing messages. iii) Transfer of requested messages. This trades off the benefit of reducing redundant message transmissions against the overhead of the history exchange and the likelihood that, at high relative speeds, the nodes will move out of range during it. In relying only on propagation histories, we avoid that overhead. Protocols which also use this

negotiation approach are described in section 2.9.2. A limited experimental comparison of the two approaches will be presented in section 5.8 by comparing our approach with Hypergossiping [51] which is also described in section 2.9.2.

Broadcast-in-time has thus emerged as a useful technique for dealing with network partitions and the recent work to employ it includes [71] which develops the Java Messaging Service (JMS) for the MANET environment.

We identify two approaches to Broadcast-in-time protocols and further subdivide them into adaptive and non-adaptive. The first approach is repetitive-flooding. A flooding phase is likely to stop before all nodes are reached. Repeating the message transmission over a period of time, which is referred to as hyperflooding [73] [98], can help to cope with network partitions. The second approach to cope with this situation is based on handshake procedures. Nodes use handshake mechanisms to 'notify' which messages they already received, so that other nodes carrying other messages can forward missed message to them. These strategies are grouped under the negotiation-based class. The classification structure we use in shown in figure 2.3.

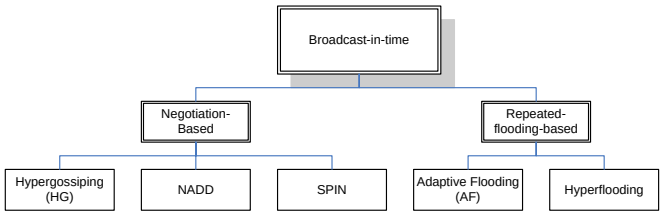

Figure 2.3: Taxonomy of Broadcast-in-time protocols

2.9.1 Repetitive Flooding

2.9.1.1 Non-adaptive protocols

Hyperflooding

In hyperflooding [73] [98], nodes store messages for a fixed time period and rebroadcast them on discovering encountering other nodes.

This solution relies on a simple partition join detection mechanism, which performs less well when in dense networks as it can create considerable broadcast storms. Furthermore, the caching strategy allows each node to shortly cache each newly received message independently from the time until next encounter of nodes which have yet to be covered. This makes the solution inefficient for highly partitioned and low mobility networks, where the caching time is not long enough until the encounter of new partitions.

Simulation results show that Hyperflooding is suitable for scenarios, where the network is partitioned but connectivity-in-time is achieved in a short time. These scenarios are a kind of grey-zone with respect to network connectivity. Hyperflooding shows a higher end-to-end delay than Broadcast-in-space algorithms, but not as high as negotiation-based protocols.

2.9.1.2 Adaptive protocols

Adaptive Flooding

Focusing on providing an adaptive broadcast protocol for MANET, [98] they call *Adaptive Flooding*(AF). AF aims to apply to a wide range of mobility speeds and as such comprises 3 distinct modes: *Scoped Flooding*, *Plain Flooding* and *Hyper Flooding*. *Scoped Flooding* attempts to reduce unnecessary broadcasts when mobility is low, it requires $2-hop$ neighbour knowledge. On receipt of a new message the a node compares its neighbourhood to that of the sender and if there is greater than 85% overlap then the rebroadcast is suppressed. *Hyper Flooding* works on top of *Plain Flooding* and requires $1-hop$ neighbourhood knowledge. All messages are buffered for a specified time (in simulations this is only 5 seconds) and when a new neighbour is detected all packets in the buffer are transmitted.

In order to switch among the three protocols *relative velocity* data (speed and direction) is exchanged in HELLO messages. The *relative velocity* of two nodes is calculated and the flooding mode switched according to *Table 2.9.1.2*.

Speed	AF Protocol
$< gt10ms^{-1}$	*Scoped Flooding*
$< 20ms^{-1}$	*Plain Flooding*
$\geq 20ms^{-1}$	*Hyper Flooding*

Table 2.1: Switching speeds for *Adaptive Flooding*

Performance of AF is compared against the three individual flooding modes. AF performs comparably well with the three modes achieving the low routing overhead of *Scoped Flooding* at low *relative speed* and the high coverage of *Hyper Flooding* at higher *relative speed*.

The highest coverage achieved by AF in their simulations is 0.90. We argue that this is too low to be considered high coverage. The protocol suffers from three distinct issues. The timeout of 5 seconds applied by AF is aggressive to reduce retransmissions and limit the effect of broadcast storms, however, this low timeout value means that nodes which are partitioned for longer than this will not be reached. Thus AF would perform poorly in low mobility partitioned networks. If this threshold were to be raised then many more retransmissions may take place increasing the potential coverage but would ultimately cause many more redundant transmissions, this would be especially damaging in high mobility connected networks. Using *relative velocity* as the mechanism for mode changing

does not take into account the density of a network. And moreover may be hard to calculate without providing additional technology for each node.

The encounter threshold that we propose in Chapter 3 successfully gets around these limitations; rather than limiting the number transmissions using a timer we count the number of encounters on which a message is retransmitted.

2.9.2 Negotiation-Based Protocols

The general idea behind these protocols relies on a three way handshake procedure. Nodes exchange advertisement messages (ADV), reply with data request messages (REQ), and finally exchange the appropriate DATA. These protocols are shown to be appropriate for highly partitioned and low mobility networks.

2.9.2.1 Non-Adaptive

In the literature we find the following schemes: SPIN-based protocols [34] and NADD [33].

- SPIN-Based Protocols: In [34], the authors presented Sensor Protocols for Information via Negotiation (SPIN). This protocol family is based on a three-way-handshake mechanism. When a mobile node discovers other mobile nodes, it advertises a summary of its messages. The listening nodes then request the messages which they are interested in. Finally, the advertising node sends the requested data. Noteworthy is that nodes store messages in local databases and do not purge them.

- NADD Protocol: In [33], the authors presented a Negotiation-based Ad hoc Data Dissemination protocol (NADD), a protocol for data dissemination in frequently partitioned MANETs. The protocol implements a three-way-handshake mechanism, in which a node advertises the IDs of a subset of locally stored messages to all its neighbours. Receivers of the advertisement message reply with a request message, where they indicate the IDs of the messages they have missed. The advertising node can then transmit the requested data. The advertising is triggered by the reception of the first copy of a message or by the discovery of a new neighbour. Since the number of cached messages becomes very large for large update rates, the authors discussed different advertising and selection strategies for the data to be advertised.

Negotiation-based protocols are robust against network partitioning since they significantly increase the delivery ratio in highly partitioned networks. By communicating with each other about the messages they still need to obtain, nodes are better able to cope with network partitioning. Due to the robustness of handshake mechanisms against network partitioning, we will utilise them to realise our generalised broadcasting technique.

The main shortcoming of the negotiation-based protocols is that they are tailored for highly partitioned networks and low mobility networks. Therefore, they are less efficient in connected or highly mobile MANETs, compared to the Broadcast-in-space protocols. For very dense networks the negotiation-based protocols lead to a higher message overhead and longer delivery delay. Highly mobile networks cause particular difficulty to these protocols since the contact between nodes may be too brief to complete negotiation, let alone transfer relevant messages. Negotiation-based protocols show high end-to-end delays (NADD shows delays up to some hours [33]). Therefore, these protocols are well suited to delay-tolerant applications. In contrast, they are unsuitable for delay-critical applications.

2.9.2.2 Adaptive

Hypergossiping

Hypergossiping (HG) provides a new adaptive protocol for broadcast in MANET [51, 50]. HG is a Broadcast-in-time approach to broadcast in MANET. It's goal to successfully allow broadcast to traverse partitions. HG utilises Gossiping within partition and Broadcast Repetition on detection of a new partition, this utilises negotiation similar to that in SPIN. A broadcast table history with entries consisting of message id, and lifetime is maintained. All messages are buffered for a specific for lifetime. Partitions are detected by exchange of Last Broadcast Received (LBR), a subset of the broadcast history, if LBRs differ significantly then a new partition is detected. On detection of a partition join, a negotiation process takes place. The encountering nodes will transmit a list of received broadcasts which allows each node to only retransmit those messages that the encountered node has not previously received. HG compares well against plain gossiping and under various densities. HG performs particularly well in low density highly partitioned low mobility networks. Without this negotiation phase HG could be considered similar to our approach with the key control being the lifetime of the message rather than our encounter threshold. This negotiation approach allows minimal transmissions to be made in a low mobility scenario, however it is precisely this technique that when used in high density or high mobility scenarios causes the performance of HG to degrade.

As mentioned earlier we provide a limited experimental comparison of our approach with Hypergossiping in section 5.8.

2.10 Improving Multicast with Gossip

Gossip may also be used to improve traditional multicast algorithms. One such protocol, Anonymous Gossip, is described here.

Anonymous Gossip

Anonymous Gossip [16](AG) utilises gossip as a method of improving the reliability of any on-demand multicast protocol. A gossip phase is added to any multicast protocol to increase the protocol's reliability. The authors add AG to MAODV although it should work for any on-demand multicast protocol. The AG phase is used after the unreliable multicast protocol phase to recover missed messages and is composed of 4 periodic rounds.

1. $Node_k$ selects a random group member $Node_l$.
2. $Node_k$ sends $Node_l$ a message history containing the id's of messages it is missing, and the sequence number of the next message it expects.
3. $Node_l$ checks if it requires or can offer any messages required by $Node_k$.
4. $Node_k$ and $Node_l$ exchange messages that are missing from one another's message history.

The choice of neighbour with which is entirely random, there is no need for any costly knowledge of group membership. A gossip message is sent to a random neighbour which then propagates this to another random neighbour: not the source of the gossip message. if the receiving node is a member of the multicast group it may choose whether to accept or re-propagate the gossip message. On acceptance a node will unicast a response back to the source of the gossip message. Propagation of the gossip message is along the multicast tree to prevent loops. The propagation is slightly biased so that nodes close to the source are chosen with high probability but that nodes far from the source are still chosen periodically; local gossip is good to avoid saturating the network with long distance gossip messages, occasional long distance gossips help resolve message loss within an entire locality. AG is a novel method of improving reliability in multicast within a MANET. AG is shown to greatly improve the performance of MAODV although 100% delivery is never achieved. The extra cost of achieving this improvement is not discussed in their work. Observations of the performance of AG improved MAODV show that performance degrades when speed of mobility increases from 99% when stationary to 90% $1ms^{-1}$ and 80% at $10ms^{-1}$.

2.11 Conclusions

Existing Broadcast-in-space techniques suffer from common drawbacks: they perform poorly in low density networks and fail to traverse partitions in disconnected networks. Several studies comparing the performance of the Broadcast-in-space techniques expose these limitations. Research dedicated to adapting these Broadcast-in-space techniques to local MANET properties has been conducted: adaptations based on the rate of change of neighbourhood are explored in [12] and [96], adaptations

based on size of neighbourhood are covered in [96]. These adaptive schemes improve performance over a broader range of network properties but still fail to meet the demands of partitioned networks.

Current Broadcast-in-time techniques perform well in partitioned networks but are less effective than Broadcast-in-space techniques in connected networks. The adaptive protocol AF addresses both connected and partitioned networks using a range of protocols from scoped flooding up to hyperflooding. This allows AF to perform well in connected networks, and perform reasonably in some partitioned networks. Unfortunately the timer based approach lets AF down in high density, high mobility networks, and also in low mobility partitioned networks.

Negotiation based protocols particularly Hypergossiping offer a intelligent approach to sparse, partitioned low mobility scenarios. Unfortunately it is the strength of their negotiation process that lets them down in both high mobility and high density scenarios.

Both AF and HG suffer from the fact that they rely on being able to provide a suitable lifetime for broadcast messages. A useful lifetime must be greater than the IMPT. The IMPT is infeasible to predict with accuracy, as it is entirely dependant on mobility, which may change at any point. If the lifetime chosen is too small and mobility is low, coverage will also be low as the broadcast times out before it covers all nodes. If the lifetime is too high this causes the protocols to continue propagating actions even after maximum coverage is reached: flooding protocols retransmit too many times causing more redundant transmissions; negotiation protocols enter the three way handshake negotiation more than necessary.

The strategies described here cover a range of different network scenarios. Both AF and HG use approaches that are closely related to our work however, there are no current protocols that can cope with both high and low mobility in partitioned networks. Our approach outlined in Chapter 3 specifically copes with both high and low speed mobility in disconnected networks as well as offering good performance in connected networks. We call our protocol Encounter Gossip.

To the best of our knowledge at this time the lowest density network that protocols have been tested in is 1.5[50]. We will show that our protocol performs with high coverage even in a density of 0.5; at this time we believe the lowest density network tested.

Chapter 3

Encounter Gossip

3.1 Introduction

We propose, and study, a family of protocols which preserve the topology-independent nature of flooding, while being able to achieve coverage levels arbitrarily close to 1, for any node density. Of course a specific high coverage cannot be guaranteed in any given instance, but can be expected with high probability. These protocols are based on a notion of 'encounter', and are controlled by an 'encounter threshold' parameter. The cost paid for a high coverage is an increase in the message traffic, since messages are broadcast more than once by each node. Under certain simplifying assumptions, it is shown that to achieve a coverage close to 1 in a network with n nodes, the total average number of broadcasts per message is on the order of $O(n \ln n)$. This is a moderate increase on the $O(n)$ broadcasts carried out in flooding. The propagation time of a message is on the order of $O(\ln n)$. Various aspects of the protocols' performance are examined by simulation.

The model, and the message propagation protocols, are described in Section 3.2. Some analytical results concerning the propagation time and the number of broadcasts are obtained in Section 3.3. The outcomes of a number of simulation experiments are presented in Section 3.4, while Section 3.5 summarises the results obtained and outlines avenues of further enquiry.

3.2 The model

The system under consideration consists of n mobile nodes which move within a given terrain. The nodes communicate with each other using wireless technology, but without any fixed network infrastructure support. That is, the nodes themselves are the sources as well as the forwarders of the message traffic, and thus form a mobile ad-hoc network. Each node has a unique identifier (MAC or IP address). It is assumed that nodes do not run out of power and do not fail (see section 7.7.1)fail; however, due to their mobility, they may become disconnected, and reconnected, as they move out of and into each other's wireless range. Thus, the structure of the network can change with time in an

unpredictable manner. For simplicity, assume that the wireless ranges of all nodes are equal and remain constant during the period of interest.

The movement of each node is governed by some 'mobility pattern', which controls its current speed and direction. It is assumed that the n nodes are statistically identical, i.e. the rules of their mobility patterns are the same, and any random variables involved have the same distributions for all nodes.

We shall define a protocol whose principal objective is to deliver a message, originating at any node, to all other nodes with high probability. A secondary objective is to minimise, as far as possible, the memory requirements at each node. In fact, what will be defined is not a single protocol, but a family of protocols depending on an integer parameter, τ, the number of encounters on which a node transmits each message.

Node i ($i = 1, 2, \ldots, n$) advertises its presence by broadcasting, at regular intervals, a signal carrying its identifier and saying, essentially, 'hello, this is node i'. It also listens for similar signals from other nodes and maintains a list, $\{j_1, j_2, \ldots, j_k\}$, of the nodes, other than itself, that it can hear. That list is called the 'current neighbourhood' of node i. At any moment in time, any current neighbourhood may be empty, or it may contain any number of other nodes.

The current neighbourhood of node i changes when a node which was in it, say j_1, moves out of range, or when a node which was not in it, say j_{k+1}, moves into range. The latter event is called an 'encounter'; that is, node i is said to encounter node j_{k+1}. Note that, since 'hello' signals are not assumed to be synchronised among the nodes, if node i encounters node j, node j does not necessarily encounter node i at the same time. Also note that, if node j leaves the current neighbourhood of node i and at some later point enters it again, then that entry constitutes an encounter. Nodes do not maintain a history of their current neighbourhoods, in order to keep their memory requirements low.

Now consider a message propagation protocol where each node behaves as follows:

1. Upon receiving or originating a new message, m, store it, together with an associated counter, $c(m)$, which is set to zero. Add the forwarding node to the current neighbourhood, unless already present. If the current neighbourhood contains nodes other than the sending one, broadcast m and increment $c(m)$ by 1.

2. At every encounter thereafter, if $c(m) \leq \tau$, broadcast m and increment $c(m)$ by 1.

3. When $c(m) = \tau + 1$, remove m from memory (but keep its sequence number in order to remember that it has been handled).

Thus, every node receiving a message broadcasts it at $\tau + 1$ consecutive encounters (one of which may be the message arrival), and then discards it. There are no acknowledgements. The integer τ is

called the 'encounter threshold'. The above protocol will be referred to as 'Encounter Gossip', and will be denoted $EG(\tau)$, in order to make explicit the dependence on encounter threshold τ.

When $\tau = 0$, the $EG(0)$ protocol behaves like flooding (except that the broadcast is delayed until the next encounter if the current neighbourhood contains only the sender). At the other extreme, if $\tau = \infty$, we have an $EG(\infty)$ protocol whereby messages are kept forever and broadcast at every encounter. Assuming that the mobility pattern is such that every node eventually encounters every other node, $EG(\infty)$ achieves coverage 1. Of course, $EG(\infty)$ is not a practical option, but we shall see in Section 3 that it can provide some useful insights.

It should be pointed out that $EG(\tau)$ trades memory capacity and probability of reaching all nodes against message traffic. Because past histories are not kept and exchanged, messages may be sent again to nodes who have already received them. By increasing the value of τ, the coverage can be made to approach 1, at the cost of having to store more messages for longer periods, and making more broadcasts.

The performance measures of interest are:

(i) The average response time of $EG(\tau)$, defined as the interval between the arrival (origin) of a message and the moment when no node can propagate it further.

(ii) The average propagation time of a message, defined as the interval between its arrival and the moment when either all nodes have received it, or no node can propagate it further.

(iii) The coverage of a message, i.e. the fraction of nodes that have received it by the end of its propagation time.

All of these performance measures are stated in terms of averages. However, the simulation results reported in Section 4 provide some indication of the corresponding variances, by repeating each experiment 10 times with different random number streams. For example, observing a coverage of 1 implies that *all 10 runs* achieved a coverage of 1.

It is important to be able to choose the value of τ so as to achieve high coverage, without unduly increasing the response and propagation times. This question will be addressed in the following sections.

3.3 Analytical approximation

In this section, we concentrate on evaluating the ability of $EG(\tau)$ to achieve high coverage, In order to make the model tractable, we assume the following:

- The overheads of collision resolution are negligible.

- Hello signals are sent and monitored at the MAC level; the information necessary to maintain the neighbourhood list is obtained at no extra cost to the higher level protocol.

- Encounters last long enough for a message to be received, i.e. the processing and propagation times of hello and broadcast messages are small enough for the encountered node to remain in the range of the encountering node.

These assumptions will not be required in the simulation experiments.

Consider an idealised system with n mobile nodes who never cease to propagate the messages they receive (∞-propagation). Let T be the random variable representing a message propagation time, i.e., the interval between the origin of a message at some node, and the first instant thereafter at which all nodes have received it. If messages are not discarded, and every node eventually encounters every other node, T is finite with probability 1. It is then of interest to estimate its average value, $E(T)$. That quantity will also be used in choosing a suitable value for τ, when designing a practicable $EG(\tau)$ protocol.

An estimate for $E(T)$ will be obtained under the following simplifying assumptions:

(a) Each node experiences encounters at intervals which are exponentially distributed with mean ξ.

(b) At each encounter, a node meets one other node.

(c) The node encountered is equally likely to be any of the other nodes; that is, the probability that node i will next encounter node j, $j \neq i$, is equal to $1/(n-1)$, regardless of past history.

Assumption (a) can be justified by remarking that the interval until the next encounter experienced by a given node — say node 1 — is the smallest of the intervals until its next encounters with node 2, node 3, ..., node n. Some of these intervals may in fact be of length 0 with a positive probability. Nevertheless, it is reasonable (e.g., see [70]) to assume that the interval until the first of many random occurrences is approximately exponentially distributed. The value of ξ depends on the density of nodes, on the speed with which they move, and on the mobility pattern. It may be difficult to determine ξ analytically, but in practice it can be estimated by monitoring the system and taking measurements.

Assumption (b) is deliberately pessimistic, in order to give the estimate the character of an upper bound. If a node encounters more than one other node at the same time, then the propagation will proceed faster. In fact, it will be seen in the experiments that at high densities this assumption is *very* pessimistic.

Assumption (c) is loosely based on the fact that all nodes are statistically identical, and move independently of each other. If the starting positions of the nodes are uniformly distributed, the assumption is justifiable at the first encounter, although it may well be violated in subsequent ones.

However, this assumption provides the simplification necessary for analytical tractability. Its effect on the performance measures will be evaluated in the simulation experiments.

Let $X = \{X(t)\,;\, t \geq 0\}$ be the Markov process whose state at any given time is the number of nodes that have already received the message. The initial state of X is $X(0) = 1$ (only the originating node has received it; again, this is a pessimistic simplification since the neighbourhood of the originating node may in fact contain other nodes). The random variable T is the first passage time of X from state 1 to state n.

Suppose that X is in state k, i.e. k nodes have received the message and $n-k$ have not. If any of the former k nodes encounters any of the latter $n-k$, the process will jump to state $k+1$. Since each node experiences encounters at rate $1/\xi$, and the probability of encountering any other node is $1/(n-1)$, the transition rate of X from state k to state $k+1$, $r_{k,k+1}$, is equal to

$$r_k = \left[\frac{k}{\xi}\right]\left[\frac{n-k}{n-1}\right] \,. \tag{3.1}$$

In other words, the average time that X remains in state k is

$$\frac{1}{r_k} = \frac{(n-1)\xi}{k(n-k)} \,. \tag{3.2}$$

Hence, the average first passage time from state 1 to state n is given by

$$E(T) = (n-1)\xi \sum_{k=1}^{n-1} \frac{1}{k(n-k)} \,. \tag{3.3}$$

This last expression can be simplified by rewriting the terms under the summation sign in the form

$$\frac{1}{k(n-k)} = \frac{1}{n}\left[\frac{1}{k} + \frac{1}{n-k}\right] \,.$$

The two resulting sums are in fact identical. Therefore,

$$E(T) = \frac{2(n-1)\xi}{n} \sum_{k=1}^{n-1} \frac{1}{k} = \frac{2(n-1)\xi H_{n-1}}{n} \,, \tag{3.4}$$

where H_n is the nth harmonic number. When n is large, the latter is approximately equal to

$$H_n \approx \ln n + \gamma \,,$$

where $\gamma = 0.5772...$ is Euler-Mascheroni's number. Also, when n is large, $(n-1)/n \approx 1$ and $\ln(n-1) \approx \ln n$.

We have thus arrived at the following estimate, valid under assumptions (a), (b) and (c):

Proposition 1 *In a large mobile network where messages are not discarded, the average propagation period for a message is approximately equal to*

$$E(T) \approx 2\xi(\ln n + \gamma) \ . \tag{3.5}$$

An immediate corollary of Proposition 1 is that, during the propagation period T, the originating node experiences an average of $2(\ln n + \gamma)$ encounters. Other nodes, who receive the message later on, tend to experience fewer encounters. Thus, choosing the encounter threshold, τ, to have the value

$$\tau = 2\lceil \ln n + \gamma \rceil \ , \tag{3.6}$$

should ensure that, when the protocol terminates, most nodes will have received the message. This suggestion will be tested experimentally.

Note 1. An attractive aspect of equation (3.6) is that the only parameter appearing in it is the number of nodes, n. The mobility pattern and the node density do not matter, as long as assumptions (a), (b) and (c) are satisfied reasonably well. However, for some mobility patterns those assumptions, and in particular (c), may be difficult to satisfy.

Note 2. Since, under the $EG(\tau)$ protocol, every node that receives a message broadcasts it $\tau + 1$ times, the total number of broadcasts per message is on the order of $O(n(\tau + 1))$. Hence, if τ is chosen according to (3.6), the total number of broadcasts per message is on the order of $O(n \ln n)$.

3.4 Experimental results

A number of simulation experiments were carried out, aimed at evaluating the effect of various parameters on the performance of τ-propagation. The following factors were kept fixed:

The terrain is a square of dimensions (1000 m) $\times$ (1000 m). The number of nodes is kept fixed at $n = 64$. The node density (defined as the average number of nodes within a circle of radius equal to the wireless range) is varied by altering the wireless range. Two values for the density are used: 0.5 and 6.5.

The interval between 'hello' signals for each node is 25 ms.

The mobility pattern is 'Random Waypoint': Initially, the nodes are distributed uniformly on the square; thereafter, each node chooses a random destination (also uniformly distributed on the square) and moves towards it at a given speed; upon reaching the destination, the node pauses for a given interval (1 ms in our case), selects a new random destination and so on.

Manhattan Grid. The area is covered by a square grid of 'North-South' and 'East-West' paths, at 40 m spacing. Initially, nodes are distributed regularly at the first n intersections of the grid. Thereafter, they move along the paths at a fixed speed. Whenever a node reaches an intersection,

it chooses one of the four available directions with equal probability (at the edges of the grid the number of possible directions is reduced appropriately). We have used an implementation of the Manhattan Grid mobility pattern provided by the University of Oregon's Network Research Group [75]. The first 1000 seconds of mobility are discarded, in order to remove initial bias.

The speed, node density and encounter threshold were varied and the performance measures — average response time, average propagation time and coverage — were evaluated. Each run starts at time 0 with a message originating at node 1, and terminates when no node can propagate the message further. For each set of parameter values, the simulation ran 25 times, with different random number seeds, and the performance observations were averaged.

We begin by presenting figures for the Random waypoint model.

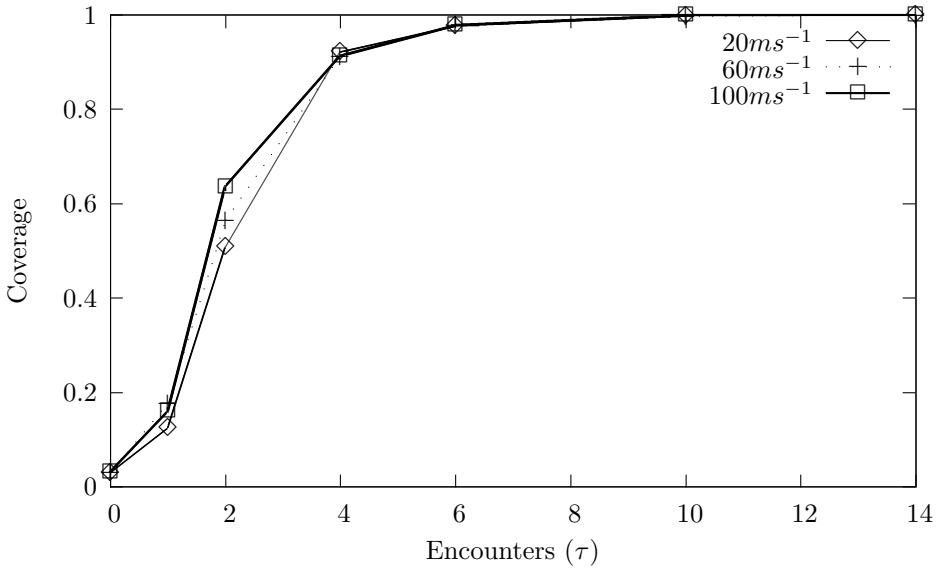

Figure 3.1: Encounters vs Coverage RANDOM-WAYPOINT Density = 0.5

Figures 3.1 – 3.3 show the coverage achieved as a function of the encounter threshold, τ, for node densities ranging between 0.5 and 6.5, and speeds ranging between 20 ms^{-1} and 100 ms^{-1} (these values are not intended to represent any realistic application; they are chosen merely as illustration). In fact, only the density has a significant effect on the coverage function; the node speed is, on the whole, immaterial. The figures quantify the extent to which the coverage can be improved by increasing τ: at low densities, where flooding performs poorly ($\tau = 0$), the improvement is considerable; at high densities, flooding performs well and the gain of increasing τ is correspondingly smaller.

Consider the analytical predictions concerning τ. For these 64 nodes, the encounter threshold given by equation (3.6) is $\tau = 10$, and the figures indicate that they do, indeed, achieve coverages close to 1. In fact, when the density is high, the threshold provided by equation (3.6) is rather

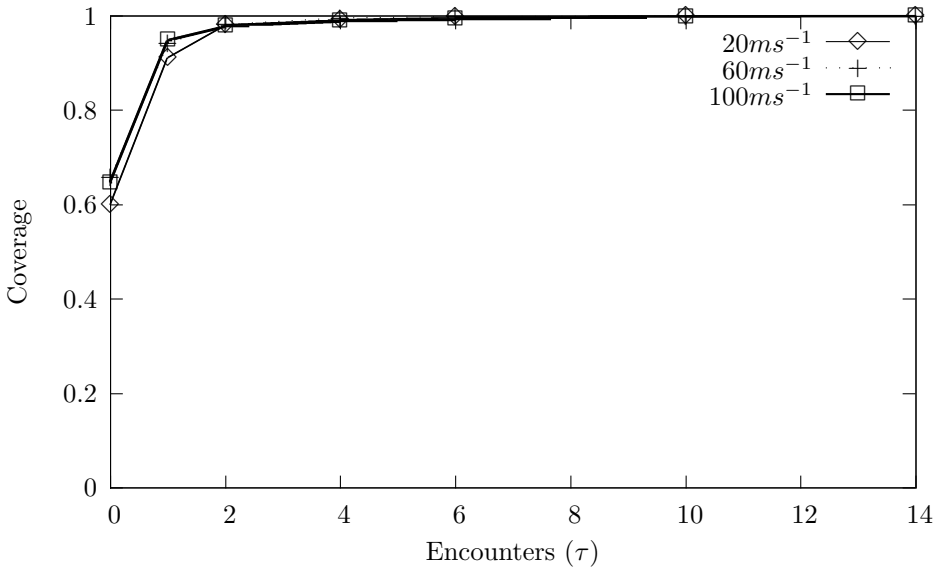

Figure 3.2: Encounters vs Coverage RANDOM-WAYPOINT Density = 3.5

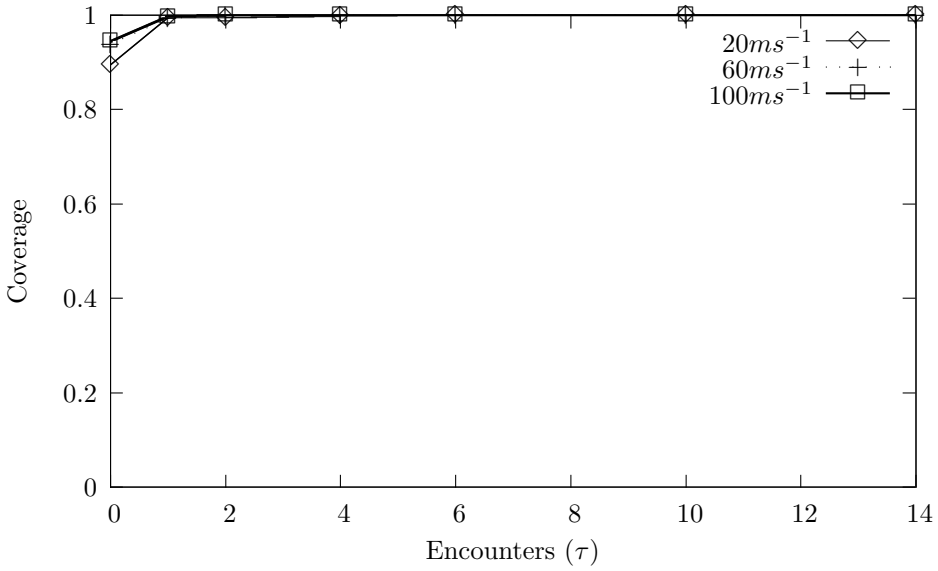

Figure 3.3: Encounters vs Coverage RANDOM-WAYPOINT Density = 6.5

conservative. This is because, for those densities, assumption (b) in Section 3 is too pessimistic.

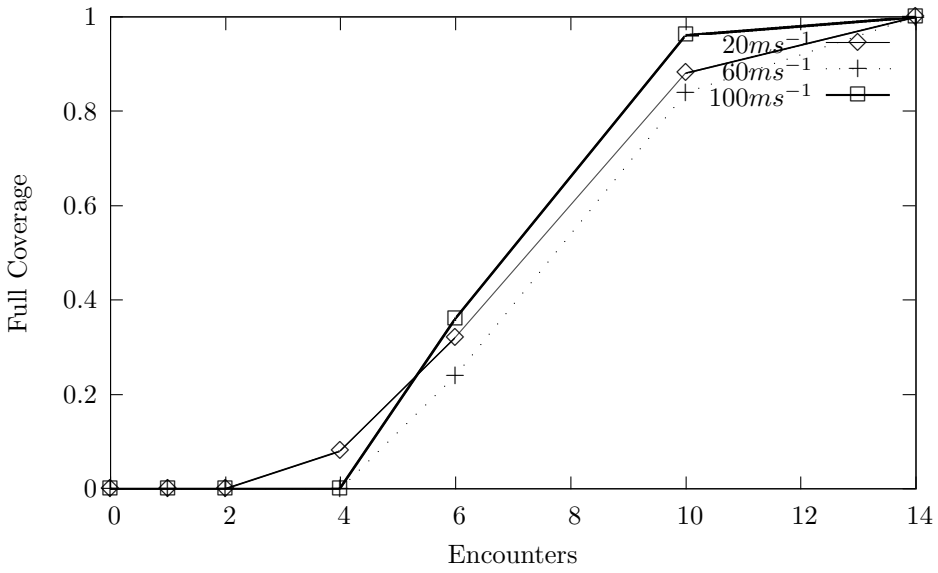

Figure 3.4: Encounters vs Full Coverage RANDOM-WAYPOINT Density = 0.5

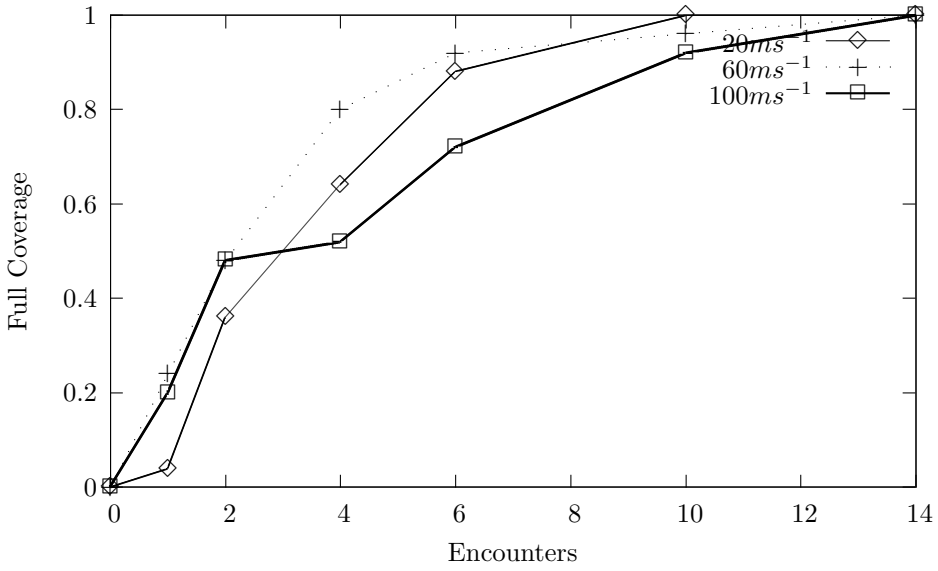

Figure 3.5: Encounters vs Full Coverage RANDOM-WAYPOINT Density = 3.5

Figures 3.4 – 3.6 estimate the probability of achieving full coverage.

This is calculated by the following:

$$for \text{ each } run_i \; p_i = \begin{cases} 1 & if \text{ full coverage is achieved} \\ 0 & otherwise \end{cases}$$

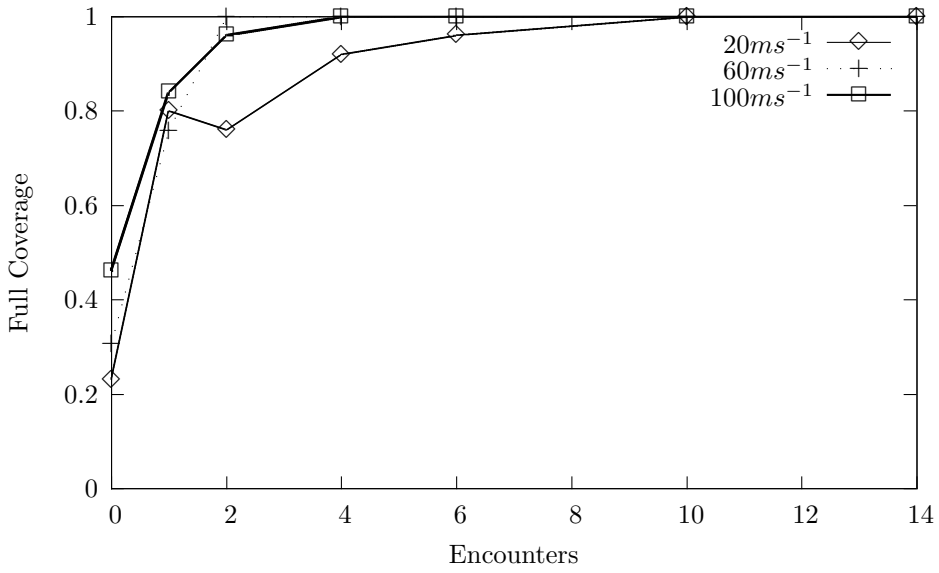

Figure 3.6: Encounters vs Full Coverage RANDOM-WAYPOINT Density = 6.5

$$P = \frac{\sum_{i=1}^{n} p_i}{n}$$

The non-monotonicity of some of these plots is due to random fluctuations in the simulated data. A single delivery failure in 64 will cause p_i to become 0 and thus P to drop by 1/25. Nevertheless this is a useful measure to compare against our predicted τ of 10. When density is 6.5 it is clear that at the higher speeds a τ of 4 is sufficient to achieve coverage of 1 with probability 1. But at the lower speed of $20ms^{-1}$ $\tau = 10$ is required to achieve a probability of 1. With the lower densities 3.5 1nd 0.5, a $/tau$ of 10 are needed respectively to achieve probability 1.

Figures 3.7 – 3.9 show the average response time and the average propagation time as functions of τ, for densities 0.5, 3.5 and 6.5 with node speeds 20 ms^{-1}, 60 ms^{-1} and 100 ms^{-1}.

The effect of mobility speed on propagation and response times is clear. An increase in speed greatly reduces the time taken to deliver the message and terminate the algorithm. It is important to note that at a density of 0.5 the network is heavily partitioned, possibly into 64 islands of only 1 node. In this case it is s apparent that the increase in speed allows nodes to meet more frequently and therefore reduce propagation and response times.

A noteworthy aspect of these figures is that, while the response time keeps increasing with τ (as expected), the propagation time increases up to a point ($\tau = 5$), and then decreases. To explain that behaviour compare Full Coverage probability and timing figures 3.4 and 3.7 at speed $20ms^{-}1$, note that when the threshold is 2 or less, the probability of $coverage = 1$ is 0 and therefore the propagation time is equal to the response time. When the threshold is 4 or more, a probability coverage 1 is greater than 0, and propagation completes, before nodes have stopped broadcasting.

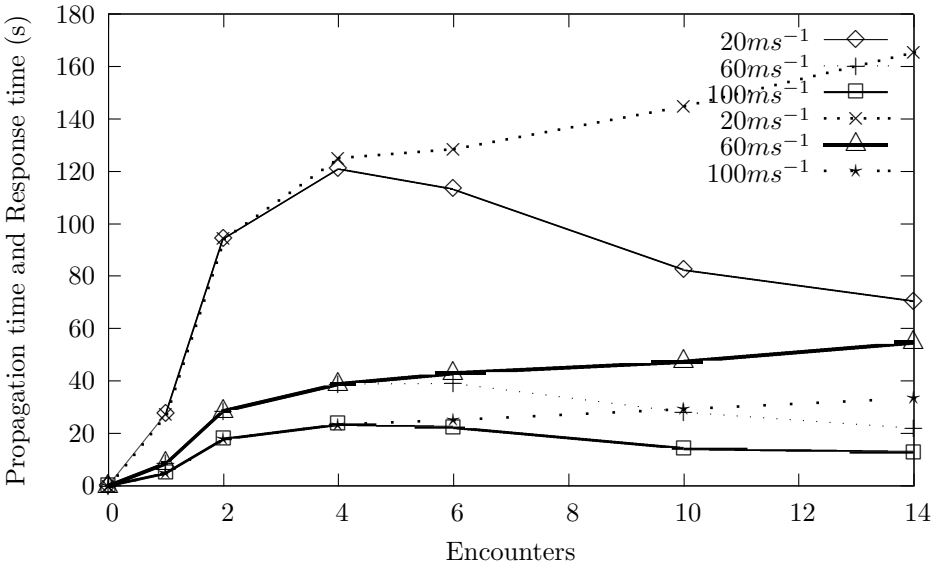

Figure 3.7: Encounters vs Propagation time and Response time at Density = 0.5

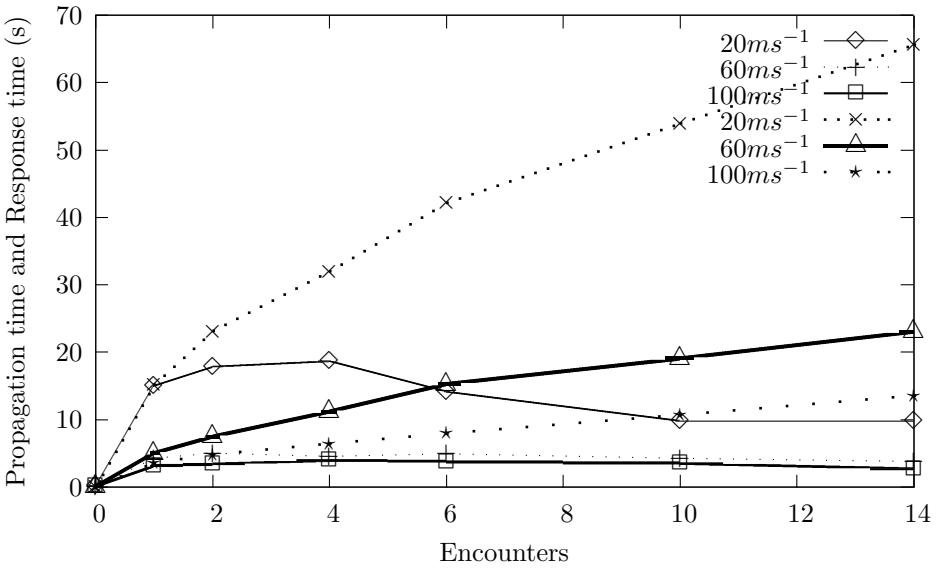

Figure 3.8: Encounters vs Propagation time and Response time at Density = 3.5

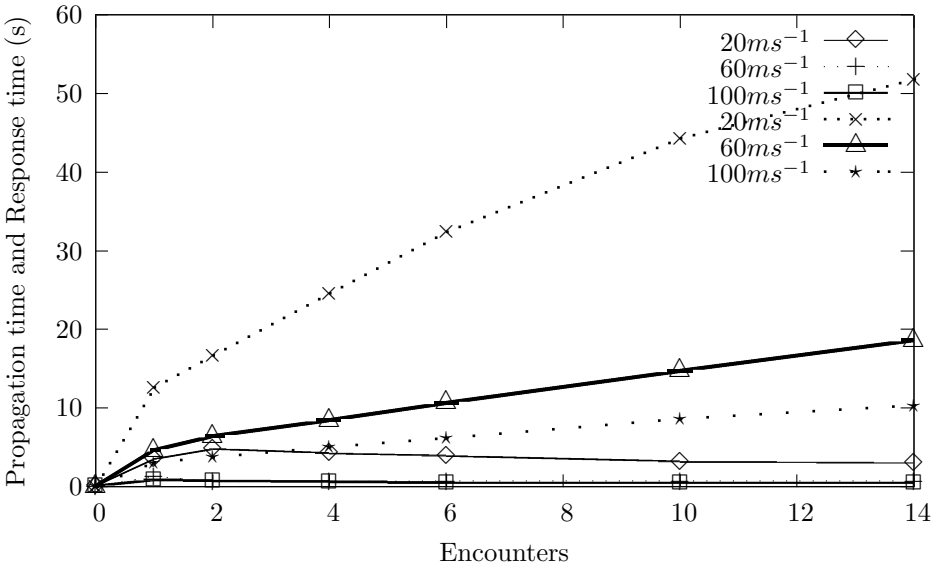

Figure 3.9: Encounters vs Propagation time and Response time at Density = 6.5

Moreover, further increases in τ tend to speed up the propagation, but prolong the response time.

At higher densities the split in timings occurs earlier. At density = 3.5 propagation and response times diverge at τ=1. At density = 6.5 the propagation and response times are split from τ=0.

The observed average intervals between encounters for density 3.5 and speeds 20 ms^{-1}, 60 ms^{-1} and 100 ms^{-1}, are $\xi = 0.96$, $\xi = 0.40$ and $\xi = 0.29$, respectively. According to equation (3.5), the corresponding limiting average propagation times (for $\tau = \infty$) should be 12.9, 5.4 and 3.9, respectively. These values agree quite well with the propagation times reached at $\tau = 14$.

We now present the same set of figures with mobility controlled by the Manhattan grid model.

Coverage for the Manhattan grid model (figures 3.10–3.12) follow a similar trend to those of the random waypoint, higher densities achieve greater coverage. For each density compared against its Random waypoint counterpart it is clear however that higher τ values are needed to achieve the same coverage. At densities 0.5,3.5 and 6.6 τ=14, 6 and 4 respectively achieve full coverage whereas in random waypoint model τ=10, 4, 2 respectively are sufficient. The restricted movement provided by gridding the mobility appears to make nodes less likely to encounter one anther.

The performance in probability of full coverage is shown in figures 3.13 - 3.15, it differs less from the random waypoint plots than the coverage did. The trends followed again appear to be similar to the random waypoint plots with higher τ required to reach the same full coverage probability. Note, however that at densities of 0.5 and 3.5, under the Manhattan grid model, P(full coverage) never reaches 1.

Propagation and response time for manhattan model are presented in figures 3.16 ... 3.18. The same features appear on these plots. Propagation and response time increase in unison until

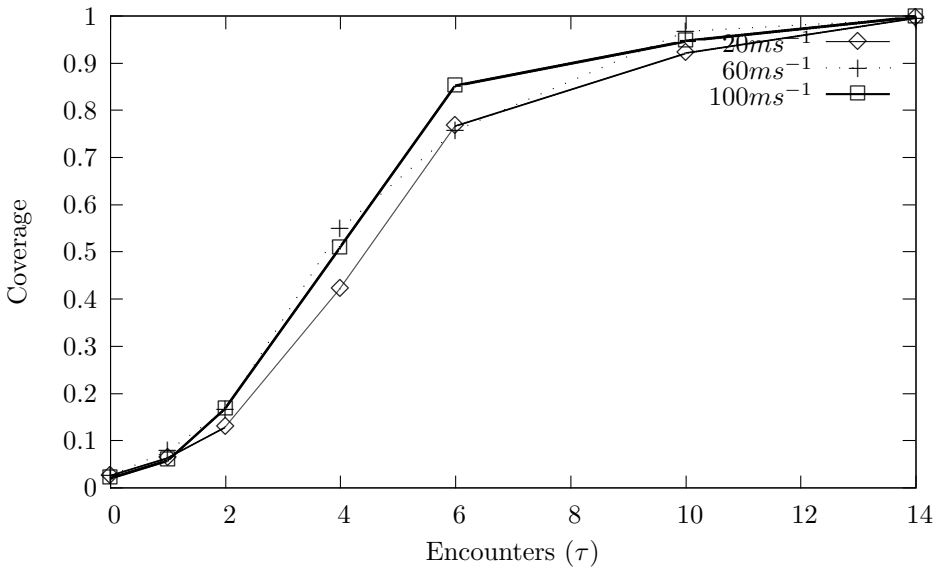

Figure 3.10: Encounters vs Coverage MANHATTAN Density = 0.5

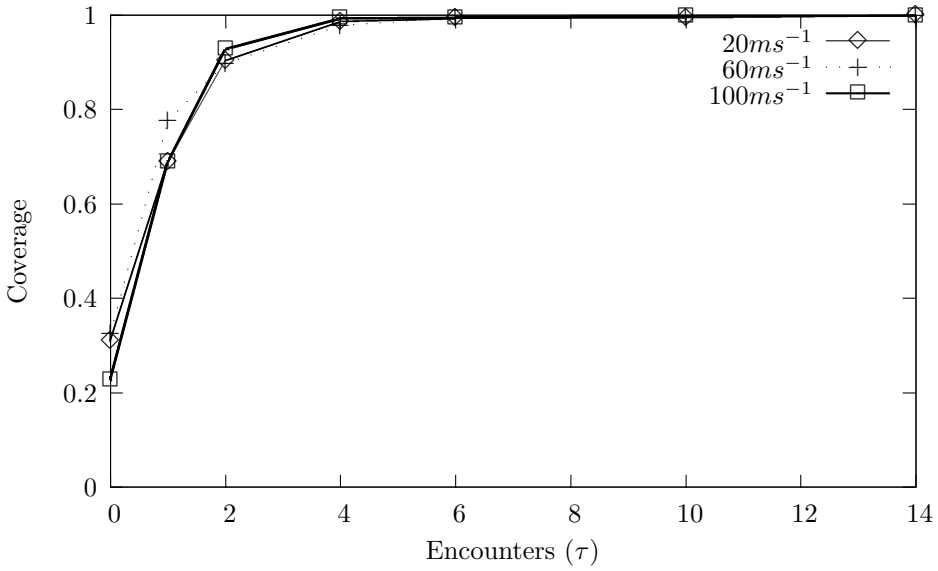

Figure 3.11: Encounters vs Coverage MANHATTAN Density = 3.5

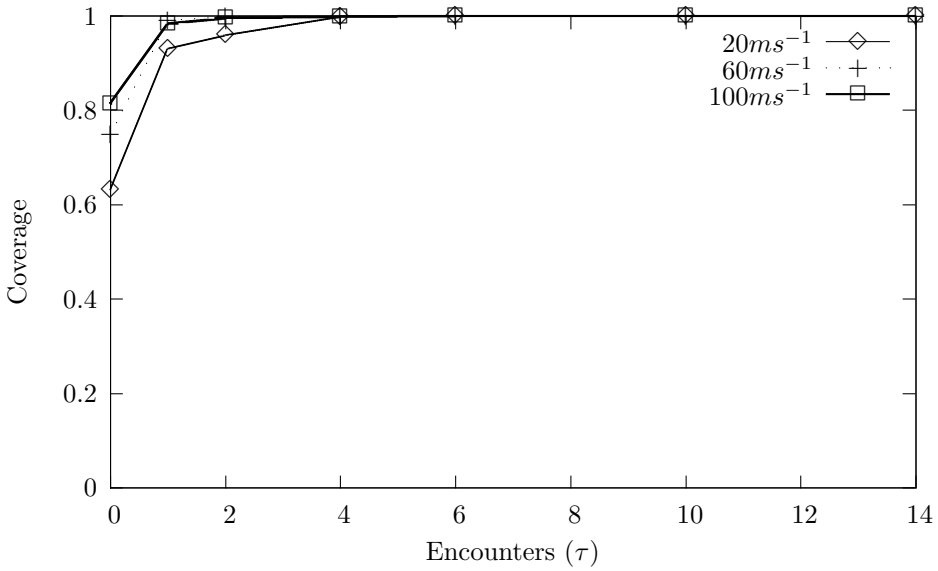

Figure 3.12: Encounters vs Coverage MANHATTAN Density = 6.5

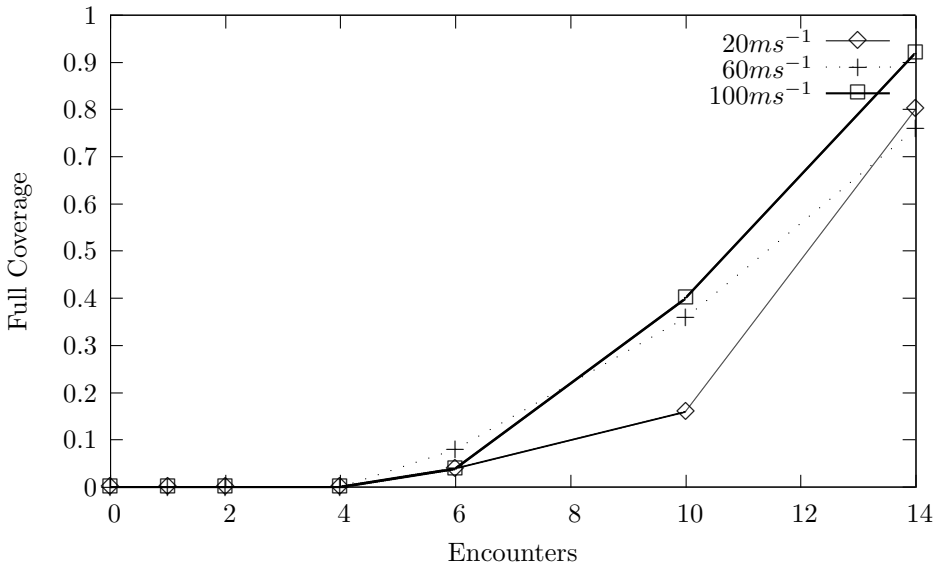

Figure 3.13: Encounters vs Full Coverage MANHATTAN Density = 0.5

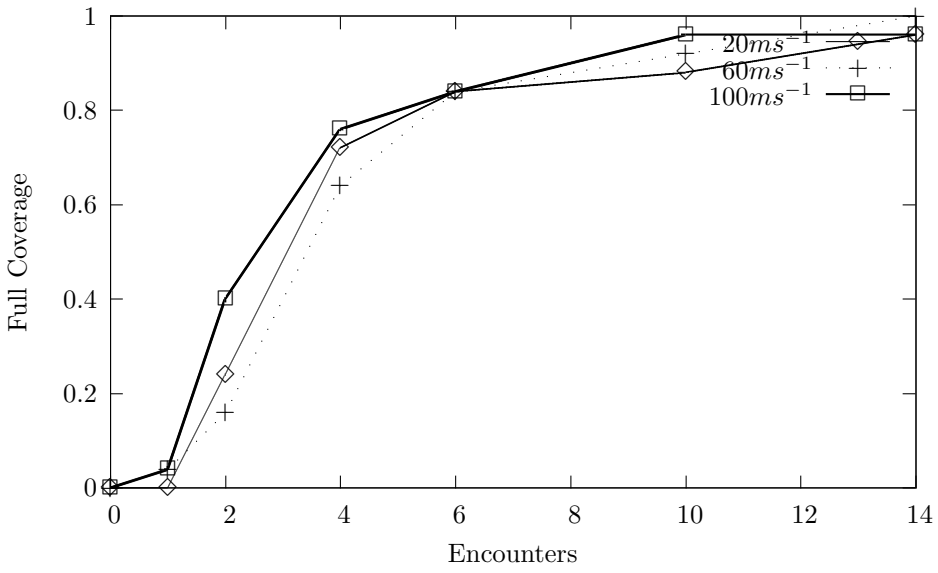

Figure 3.14: Encounters vs Full Coverage MANHATTAN Density = 3.5

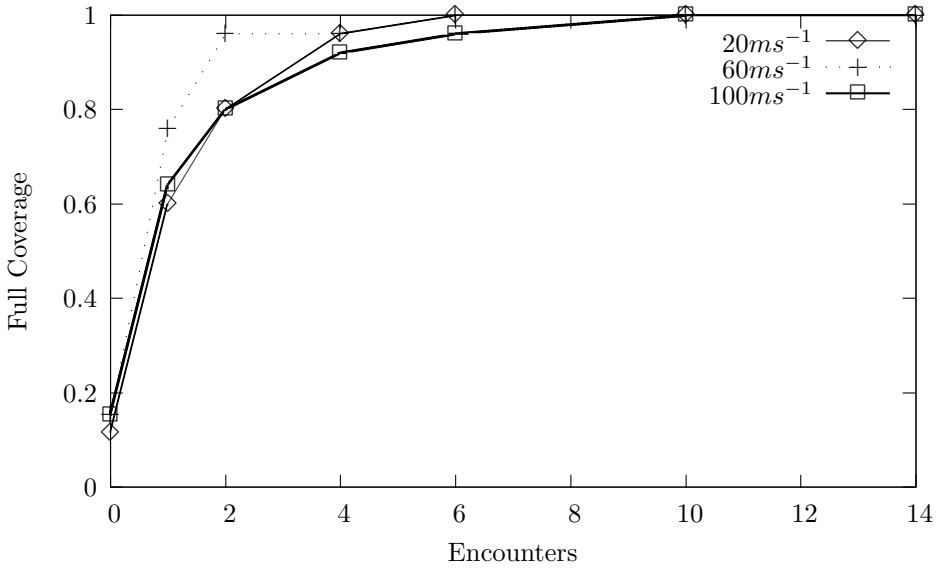

Figure 3.15: Encounters vs Full Coverage MANHATTAN Density = 6.5

P(full coverage)> 0 then propagation time begins to decrease whilst response time continues its rise. At all densities and τ the response and propagation times are considerably higher than those achieved under random waypoint. In figure 3.16 we see that the response time at τ=14, $20ms^{-}1$ is 490s, this is just more than double the response time of the same set of simulations under random waypoint. On a more careful examination of the figures this trend is followed across all densities, τ values and speeds. The transition between random waypoint and manhattan grid models causes the response and propagation times to approximately double.

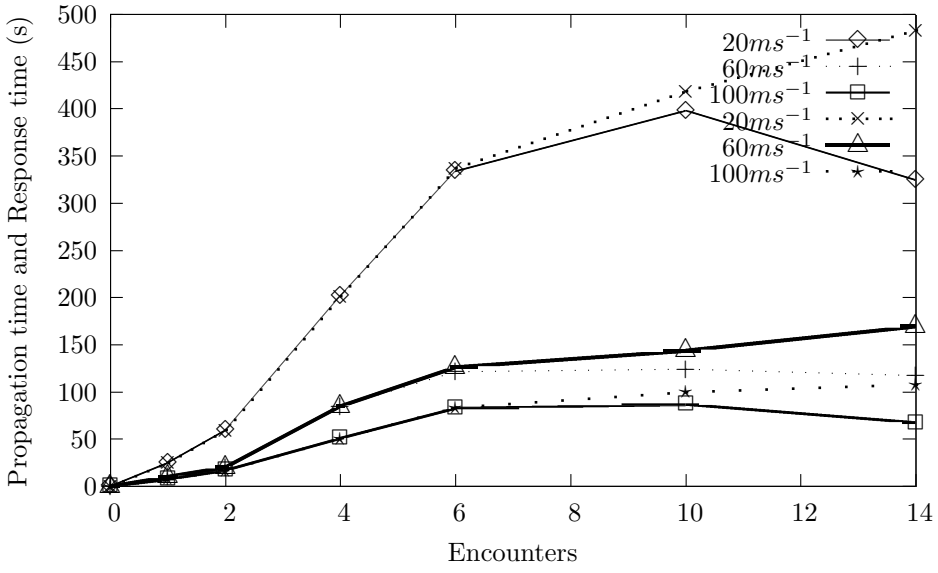

Figure 3.16: Encounters vs Propagation time and Response time at Density = 0.5

The process of propagating a message among the nodes in a network where the speed (60 ms^{-1}) and threshold (τ = 14) are fixed, while the density is varied in the range 0.5 – 6.5, is illustrated in figure 3.19. The graphs show how the rate of propagation changes as more and more nodes are covered. At high densities, it takes longer to cover the last 5% of the nodes than the first 95%. This phenomenon is due to the fact that some nodes on the periphery of the terrain can be relatively more difficult to reach than the others. It is less pronounced at lower densities, but is still in evidence: the last 20% of the nodes take about as long to cover as the first 80%.

3.5 Conclusions

The main contributions of this chapter can be summarised as follows:

1. Introduction of Encounter Gossip, the τ-propagation family (Section 2).

2. Mobility-independent estimate for the value of τ that achieves high coverage (equation (3.6)).

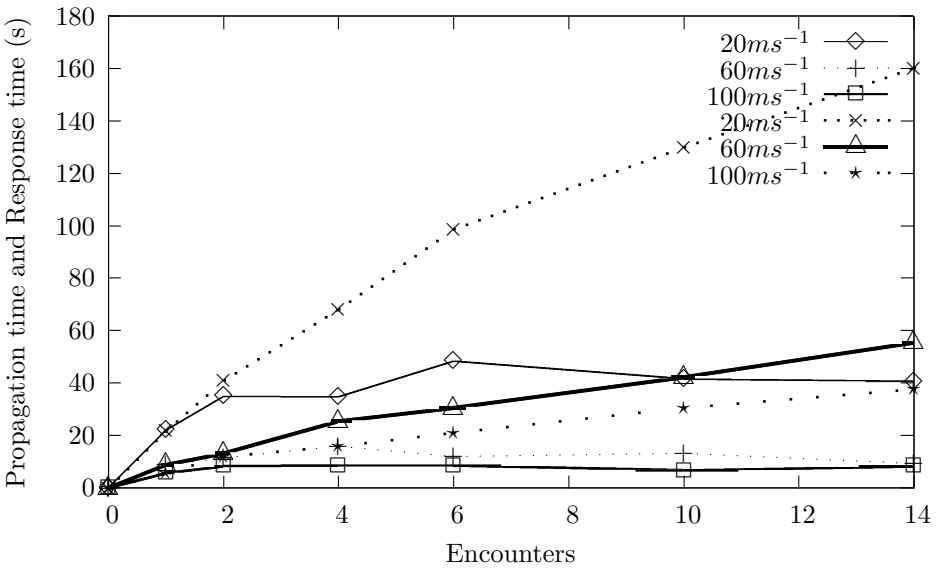

Figure 3.17: Encounters vs Propagation time and Response time at Density = 3.5

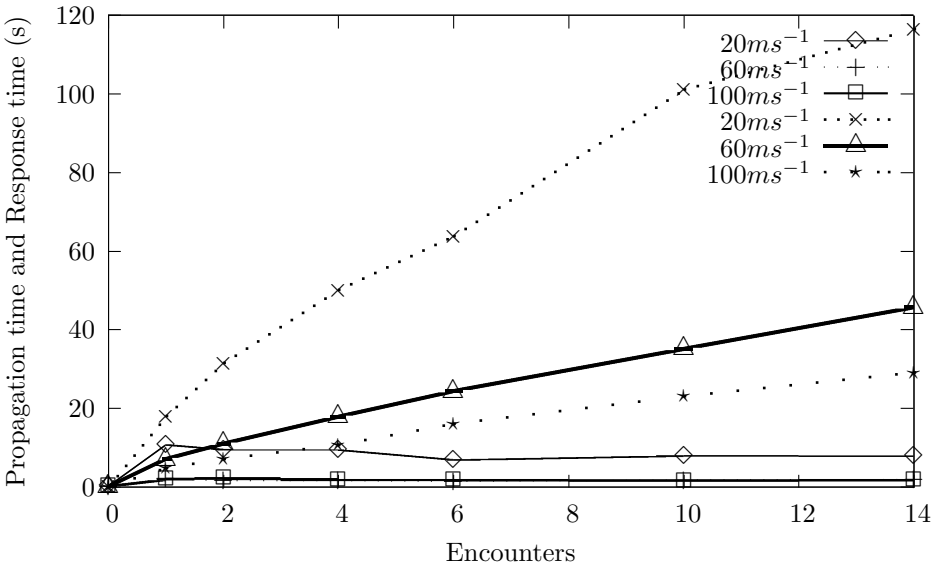

Figure 3.18: Encounters vs Propagation time and Response time at Density = 6.5

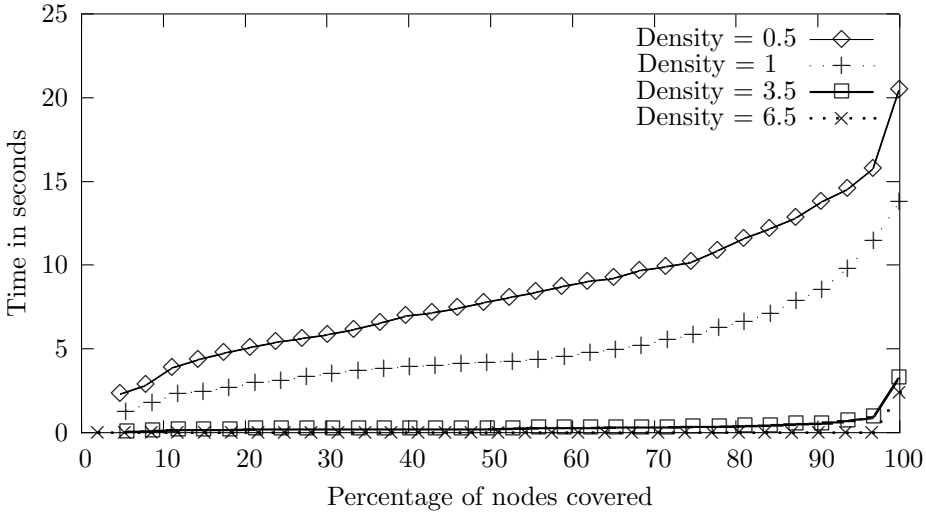

Figure 3.19: Process of propagation: speed = 60 ms^{-1}; $\tau = 14$

3. Quantitative performance results obtained by experimentation (Section 4).

Under the Random Waypoint mobility model, $EG(\tau)$ achieves high coverage with $\tau = 10$, the required τ for 64 nodes. When density is high this may be causing many unnecessary transmissions. In the next chapter we will address the optimisation of EG.

The differences visible between Manhattan grid and random waypoint models are of considerable interest. The Manhattan grid model provides a much more difficult propagation scenario for Encounter Gossip to tackle. We have shown however that increasing τ can overcome these difficulties.

A more adaptable family of propagation protocols may be designed by introducing a FIFO buffer for messages. Messages would be kept in the buffer, and re-broadcast, until either they are displaced by new messages or they reach an encounter threshold. The number of times a message is broadcast by a node would then change dynamically in response to changing conditions. That number could also be adjusted by keeping track of repeated receptions of the same message. A time-out interval can be introduced, to force the discarding of a message if the node does not experience a sufficient number of encounters. In addition, the encounter threshold may be controlled by the number of nodes already encountered, and possibly by the mobility pattern. All these are worthy topics for future research.

Chapter 4

Optimisation of Encounter Gossip

4.1 Introduction

Encounter Gossip is a family of protocols which preserve the topology-independent nature of flooding, while being able to achieve a coverage close to 1 even at low densities, was proposed in Chapter 3. These protocols, called *Encounter Gossip*, are controlled by an 'encounter threshold' parameter, τ, specifying the number of times a node is required to broadcast a given message. The larger the value of τ, the higher the coverage achieved, but also the higher the propagation overhead. Indeed, in a network with n nodes employing Encounter Gossip with parameter τ, the average number of redundant transmissions (a broadcast is redundant if it does not enlarge the set of nodes that have already received the message) is roughly proportional to $n\tau$.

This chapter presents the optimisation of Encounter Gossip using several different techniques. Modifications of the τ-propagation protocol are suggested, aimed at reducing the number of redundant transmissions without significantly lowering the achieved coverage. Some of these use random timers; others keep (partial) information about the history of the propagation process. The proposed approaches are evaluated and compared, for different network configurations.

The stochastic processes modelling message propagation in a MANET under the proposed protocols do not, in general, lend themselves to mathematical analysis. That is why the protocol evaluations and comparisons are performed by simulation, using the GloMoSim package.

We describe an additional performance measure, redundant transmissions in Section 4.2. The modifications using timers are presented in Section 4.3, while those employing history information are described in Section 4.4. The empirical results of the simulation experiments are displayed in Section 4.5, while Section 4.6 gives a summary and outlines avenues of further enquiry.

4.2 Redundant Broadcasts under EG τ

A broadcast is said to be 'useful' if it enlarges the set of nodes that have received m, i.e. if there is at least one node in the current neighbourhood for whom m is new. A broadcast which is not useful is 'redundant' (all nodes receiving that broadcast have already received m).

Since every useful broadcast adds at least one node to those that have received m, there can be at most $n-1$ useful broadcasts associated with a given message. On the other hand, if all nodes receive the message, a total of $n(\tau+1)$ broadcasts are made under EG(τ). Therefore, when the coverage is 1, at least $n\tau+1$ broadcasts are redundant. Reducing that number while still achieving a high coverage is an important objective.

The general idea of the approaches proposed here is to suppress a broadcast of m mandated by the EG(τ) protocol, if that broadcast is judged to have little additional effect on the propagation of m, and to treat the suppressed broadcast as though it had been carried out. Consequently, a node may end up doing fewer than τ broadcasts while the coverage remains largely unaffected.

Two types of broadcast-suppression mechanisms are introduced. In the first, *timer* based approach, node i sets a random timeout interval following an encounter and decides whether or not to suppress the broadcast depending on the events it observes during that period.

The second approach is *history* based. Each node maintains a local list of nodes that are known to have received m or that could have received m. When node i experiences an encounter, it suppresses its broadcast if the encountered node is already in the local list.

There are several variants within each of the above mechanisms.

It is worth pointing out that the two approaches are operationally independent of each other. The timer based control is concerned with the events that node i observes *soon after* an encounter, whereas the history based approach relies on information available *at the time* of an encounter. Therefore the two mechanisms can be combined to operate together, enabling the suppression of a broadcast if either of them recommends it.

4.3 Timer Based Optimization

One technique that is used to reduce redundant transmissions is the Random Assessment Delay (RAD). Having decided to broadcast a message as a result of an encounter (not as an originator), a node waits for a random period of time, called the 'RAD interval'. An encounter which occurs during a RAD interval does not generate a new RAD interval. If, during a RAD interval, a node hears another broadcast of the same message, then the planned broadcast is suppressed (see [104]). The rationale is that if several nodes in a given neighbourhood are in possession of the message and decide to broadcast it, one of them will do so first (the one with the shortest RAD interval), and

then the others can keep quiet.

Remember that under Encounter Gossip a node ceases to transmit message m when the number of encounters, recorded in $c(m)$, reaches the value τ. Now, the addition of RAD may affect the way $c(m)$ is incremented. During a RAD interval associated with message m, the node counts the number of transmissions of m that it hears. Denote that number by r. If $r = 0$ at the end of the RAD interval, then the node transmits m and increments $c(m)$ by 1. If $r > 0$, there are two possibilities:

1. Do not transmit m and increase $c(m)$ by 1;
2. Do not transmit m and increase $c(m)$ by r.

Both of these policies were studied and it was found that their performance is similar, with policy 2 performing marginally better than policy 1 (in the sense that it achieves a slightly larger reduction in redundant transmissions, without significant adverse effect on the coverage). So, in order to avoid duplication, policy 1 will not be considered further; the RAD results presented in Section 5 concern policy 2 only.

It would be worth further investigation to consider mobility scenarios where nodes tend to cluster together. In such clusters, local density is higher than the norm and so policy 2 will provide higher reductions of transmissions.

4.3.1 α-*reduction*

Suppose that node i has already received and perhaps broadcast message m, and now hears it broadcast by a node j present in its current neighbourhood (i.e., node i is not experiencing an encounter). This can happen because node j has an encounter with another node, k, which is outside i's neighbourhood. Node i thereby learns that a node carrying the same message, outside the current neighbourhood, is passing within 2 radii of it. One can argue that, in the light of this information, node i should not proceed to make the number of broadcasts required by EG(τ), but should increment its counter $c(m)$ by an amount reflecting the density of nodes in this region. It is proposed, therefore, that on hearing a broadcast of a message already held, by a node already present in the neighbourhood, node i should increment $c(m)$ by a fraction, α, of the number of nodes in its current neighbourhood (truncated to an integer).

Incrementing $c(m)$ causes the number of future broadcasts to be reduced. This policy will be referred to as α-*reduction*. Although it does not use a timer, we include it here because it has a similar character to a timer-based policy: its decisions are triggered by events that occur *after* an encounter. The α-*reduction* policy can be operated on its own, or in conjunction with other broadcast suppression policies such as RAD.

After some experimentation with different values of α, the value $\alpha = 0.39$ was found to perform well over a range of parameters. That number happens to be (approximately) the minimum area of

intersection of two equal circles whose centers are within each other's radii, as a fraction of the area of one of them.

4.4 History Based Optimization

Suppose that each node has a local cache where it can store the *id*s of all nodes it has encountered since first receiving (or originating) message m. It would be reasonable to assume that all nodes on that list have received m (that assumption is wrong only if a broadcast message failed to reach some of the neighbouring nodes). Therefore, if node i encounters node j and discovers that j is already on its list of receivers, it can suppress the broadcast required by EG(τ) and increment $c(m)$ by 1. This policy will be referred to as *encounter history* reduction, or EH.

A more comprehensive history of nodes that are presumed to have received a given message can be maintained by passing cache information at encounter events. Any node broadcasting m can piggy-back its current list of *id*s onto the message being broadcast; the receiving nodes would then merge that list with their own. Thus, the current list kept by node i contains not only the nodes to whom i has broadcast m, but also nodes about which it has been told that they have received m. Again, if node i encounters node j and finds that j is already on its list of receivers, it suppresses its broadcast and increments $c(m)$ by 1.

This policy is called *propagation history* reduction, or PH. For PH to be scalable, the size of the piggy-backed information needs to be kept small. This can be achieved by limiting the number, k, of *id*s that are piggy-backing onto a message. If, at the time of a broadcast, a node's cache contains more than k *id*s, k of them are selected for inclusion in m. The selection criterion may be, for example, FIFO, LIFO, or random.

Clearly, there is a trade-off between the size, k, of the list that may be attached onto a message, and the efficiency of the propagation protocol. The larger the value of k, the fewer redundant messages will be sent, but also the larger each broadcast will be, and hence the smaller the fraction of 'essential' information transmitted per broadcast. That trade-off is not studied to any great extent here. The experiments in Section 5 assume that $k = n$, and thus provide an upper bound on the achievable reduction of redundant transmissions.

4.4.1 Broadcast count reduction

A simple way of associating history information with a message m is to attach to m a count, $b(m)$, of the number of times it has been broadcast. Whenever any node broadcasts m, $b(m)$ is incremented by 1. Of course, copies of the same message being passed on by different nodes may experience different numbers of broadcasts; their values of $b(m)$ would then be different. If a node receives a copy with a higher value of $b(m)$ than the one it already holds, then $b(m)$ is set to the new value.

The Encounter Gossip protocol can be modified by replacing the local node encounter counts, $c(m)$, and the node threshold, τ, with the message broadcast counts, $b(m)$, and a message threshold, β. Any node which receives, or broadcasts, a message m whose broadcast count has reached the threshold, $b(m) = \beta$, would stop broadcasting m, even though it may later receive a copy with a lower count.

This version of the protocol will be referred to as *broadcast count* reduction, or BC. Since the information attached to a message consists of a single integer, rather than a list that may grow with n, BC has the advantage of being scalable, against the disadvantage of using a rather limited kind of history information.

The trade-offs involved in choosing the value of β are similar to those concerning τ: the larger the value of β, the higher the coverage, but also the greater the number of redundant messages. These trade-offs will be examined empirically, but it would also be useful to propose a heuristic value for β that may be expected to perform well. Such a heuristic is suggested by the following very crude argument.

Suppose that the system evolves in discrete time, and assume that all encounters are useful, synchronised and involve one new node each. In other words, a message originates at time 0 at some node, i; at time 1, node i encounters node j; at time 2, node i encounters node k and node j encounters node l; etc. Under this optimistic scenario, the number of nodes that have received the message by time t grows roughly like 2^t. Conversely, when all n nodes have received the message, roughly $\log_2 n$ time steps have elapsed; that is also the value reached by the broadcast counters.

In practice, things are not so regular, so the value of β should be more conservative. The heuristic proposed is to use some small multiple of the above estimate, e.g. $\beta = 2 \log_2 n$ or $\beta = 3 \log_2 n$.

Broadcast count reduction may also be operated in conjunction, rather that instead of, Encounter Gossip: a node would stop broadcasting m as soon as either the encounter count or the broadcast count reaches the relevant threshold, i.e. $c(m) = \tau$ or $b(m) = \beta$.

4.5 Experimental Results

In order to evaluate the performance of the proposed modifications to EG(τ), several sets of simulations were run using the GloMoSim tool. The following parameters were kept constant throughout the simulations unless explicitly specified.

The terrain is a square of dimensions (1000 m) $\times$ (1000 m).

The number of nodes is kept fixed at $n = 64$. The node density (defined as the average number of nodes within a circle of radius equal to the wireless range) is varied by altering the wireless range. Three values for the density are used: 0.5, 3.5 and 6.5.

The interval between 'hello' signals for each node is 250 ms. Timeout interval after which a node

is removed from a neighbourhood is 1000 ms

Different node speeds were simulated, ranging from 2 ms^{-1} to 40 ms^{-1}. It was observed that the speed has little effect on either coverage or redundant transmissions. We present both speeds for our timer based optimisations (RAD and α-*reduction*) under Random Waypoint, to illustrate the similarities. The remaining figures presented are only for a node speed of 2 ms^{-1}.

Experiments were carried out with two different mobility patterns: 'Random Waypoint' and 'Manhattan Grid'. These work as follows:

Random Waypoint. Initially, the nodes are randomly positioned with a uniform distribution on the square; thereafter, each node chooses a random destination (also uniformly distributed on the square) and moves towards it at the given speed; upon reaching the destination, the node pauses for a given interval (0 ms in our case), selects a new random destination and so on. The first 1000 seconds of mobility are discarded, in order to skip the naturally occurring initial clustering phase (see [10]).

Manhattan Grid. The area is covered by a square grid of 'North-South' and 'East-West' paths, at 40 m spacing. Initially, nodes are distributed regularly at the first n intersections of the grid. Thereafter, they move along the paths at a fixed speed. Whenever a node reaches an intersection, it chooses one of the four available directions with equal probability (at the edges of the grid the number of possible directions is reduced appropriately). We have used an implementation of the Manhattan Grid mobility pattern provided by the University of Oregon's Network Research Group [75]. The first 1000 seconds of mobility are discarded, in order to remove initial bias.

The performance measures are the coverage (fraction of nodes that received the message), and the average number of redundant transmissions per node (total number of redundant transmissions divided by the number of nodes that received the message). Each run starts at time 0 with a message originating at node 1, and terminates when no node can propagate the message further. For each set of parameter values, the simulation ran 50 times, with different random number seeds, and the performance observations were averaged.

The experimental results are grouped according to the policies that are being compared, and also according to the mobility pattern used. Two figures are produced for each group, showing the coverage achieved, and the number of redundant transmissions per node, as functions of the encounter threshold (or broadcast threshold in the case of the BC policy).

Figures 4.1 ... 4.4 illustrate the effects of RAD and α-reduction policies under Random Waypoint mobility, for two different node densities (and a joint application of RAD and α-reduction). The label 'RAD' here applies to a RAD period distributed uniformly on the interval (0,100) ms with speeds of $2ms^{-1}$ and $40ms^{-1}$. The RAD period was distributed uniformly on the interval (0,100) ms. The numbers of redundant transmissions *per node* are plotted as functions of the encounter threshold.

At speeds of $2ms^{-1}$, in figures 4.1 and 4.2 we observe that at the low density of 0.5, the original

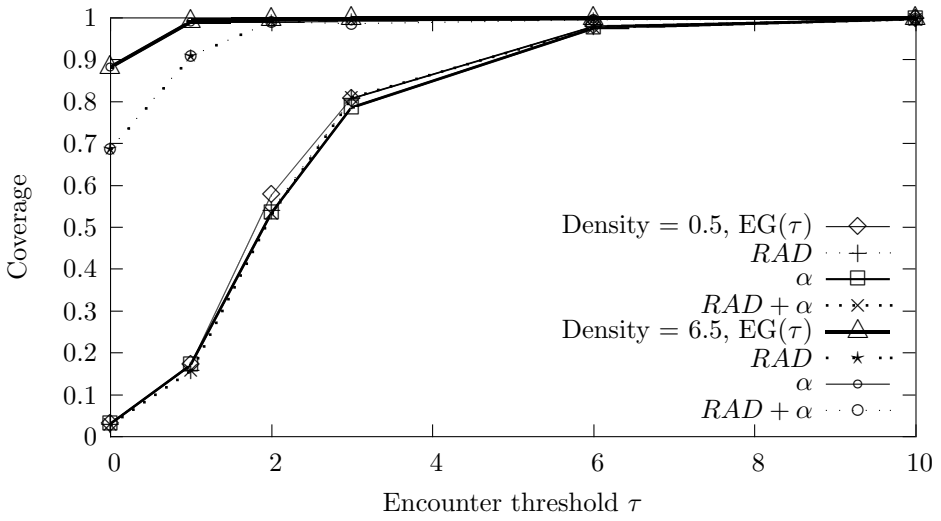

Figure 4.1: Coverage: RAD and α-reduction policies; Random Waypoint; $2ms^{-1}$

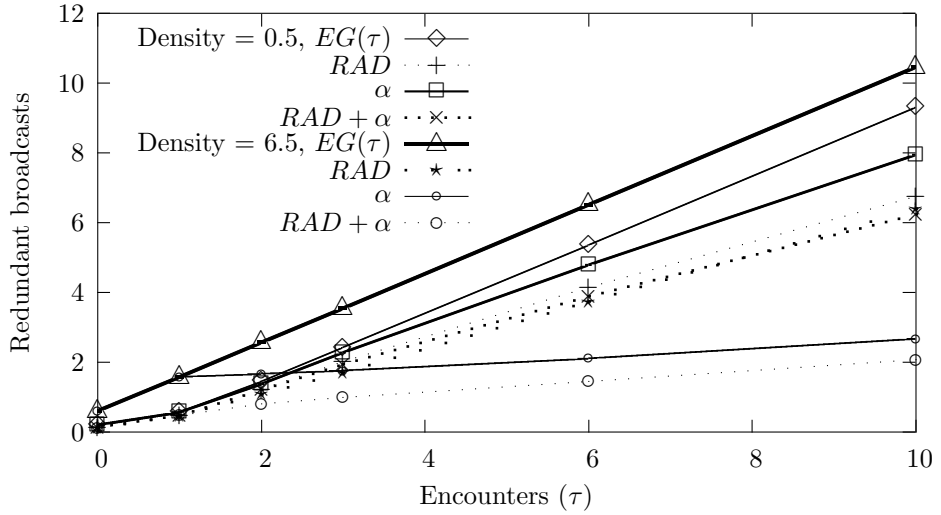

Figure 4.2: Redundant Broadcasts: RAD and α-reduction policies; Random Waypoint; $2ms^{-1}$

$EG(\tau)$ protocol and the RAD and α-reduction modifications achieve very similar coverage levels. At that density, an encounter threshold of 9 or 10 is required in order to achieve full coverage. The corresponding savings in redundant transmissions achieved by the RAD and α-reduction policies are approximately 25% and 15% respectively. The joint application of RAD and α-reduction yields a small additional improvement.

On the other hand, when the density is at the high level of 6.5, an encounter threshold of 2 or

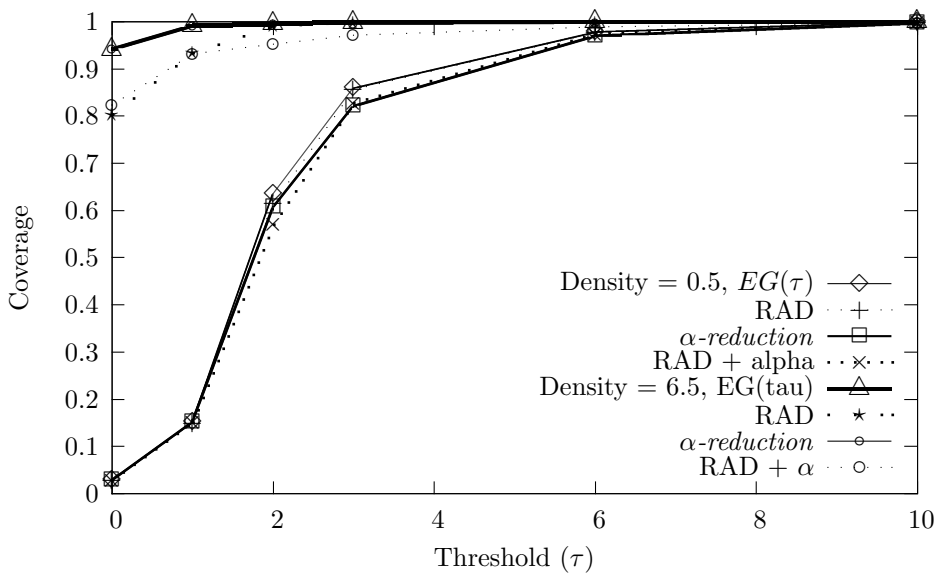

Figure 4.3: Coverage: RAD and α-reduction policies; Random Waypoint; $40ms^{-1}$

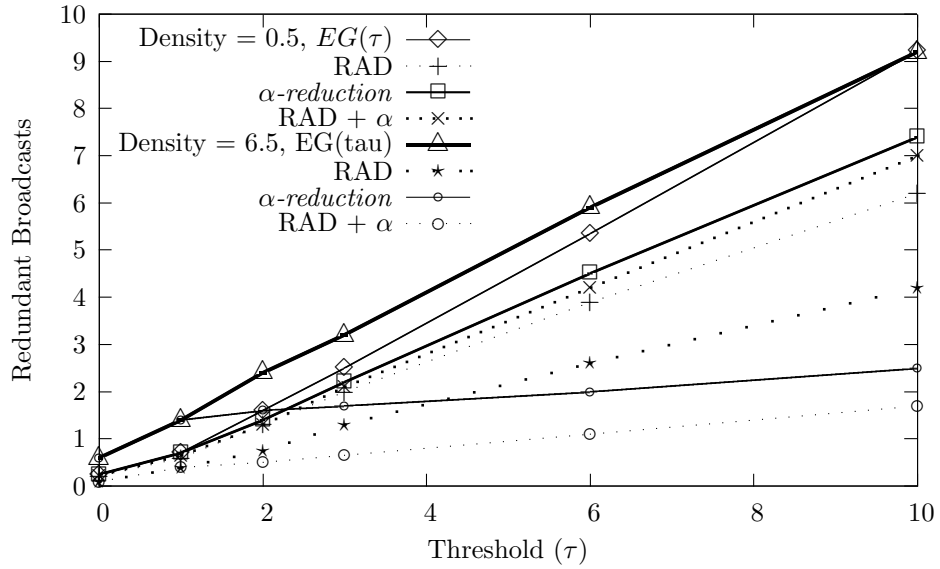

Figure 4.4: Redundant broadcasts: RAD and α-reduction policies; Random Waypoint; $40ms^{-1}$

3 suffices to achieve full coverage. Some coverage is lost by the RAD policy, but almost none is lost by the α-reduction policy. The RAD policy starts off achieving bigger savings in redundant transmissions than the α-reduction policy, but becomes poorer at higher thresholds. When $\tau = 3$ (where all policies provide full coverage), both RAD and α-reduction achieve approximately 50% reduction in redundant transmissions, however, when combined they increase the saving to about 70%. A notable feature of figure 4.2 is that the average number of redundant transmissions per node

for the α-reduction policy at high density is almost independent of the value of τ.

Examining higher speed ($40ms^{-1}$) figures 4.3...4.4, we can see that for all densities and protocol optimisations, there is little difference from the lower speeds ($2ms^{-1}$). What we see is a marginal coverage increase for all plots, and a marginally lower redundancy at each point. This trend is typical across all the optimisation techniques we demonstrate, we omit the high speed figures for the other optimisations.

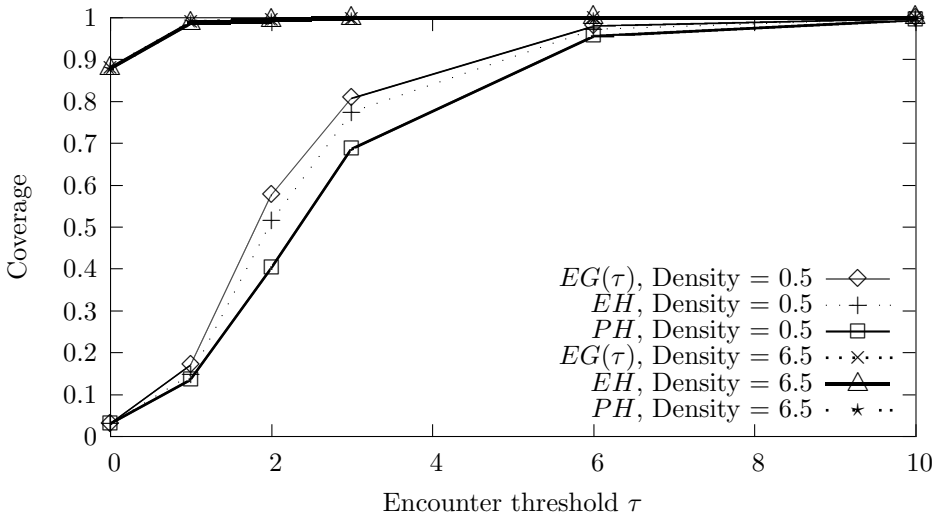

Figure 4.5: Coverage: encounter and propagation histories; Random Waypoint

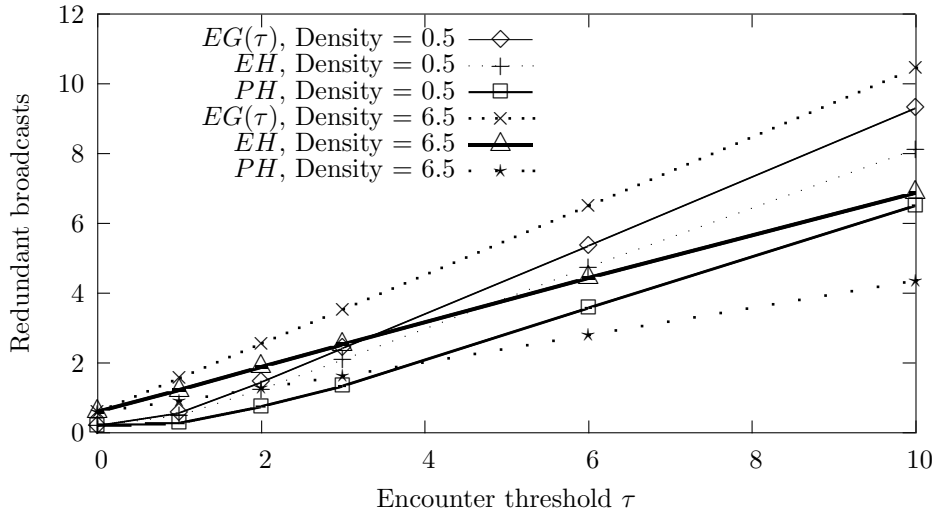

Figure 4.6: Redundant broadcasts: encounter and propagation histories; Random Waypoint

Figures 4.5 and 4.6 compare the performance of the Encounter History (EH) and Propagation History (PH) policies with that of the original Encounter Gossip protocol. In this experiment, there is no limit on the number of node ids that can be attached to a message (i.e., $k = n$). Thus, any benefits in redundant transmissions achieved by the PH policy should be set against the extra overhead of broadcasting longer messages than necessary.

When the node density is low, both the EH and PH policies achieve lower coverage than $EG(\tau)$ (PH being consistently worse than EH). However, those differences are noticeable only if the threshold τ is in any case insufficient; all three policies achieve full coverage at about the same threshold level, in this case $\tau = 10$. For that value of τ, the EH policy reduces the average number of redundant transmissions by just over 10%, while the PH policy reduces them by approximately 30%.

When the node density is high, the three policies are almost indistinguishable in their coverage. For threshold values that achieve full coverage ($\tau = 2$ or $\tau = 3$), the EH policy reduces the redundant transmissions by about 50%, while the reduction achieved by the PH policy is close to 70% (but remember the comment about the extra overhead involved).

The next group of experiments, illustrated in figures 4.7 and 4.8, examine the performance of the Broadcast Count reduction policy (BC), on its own and in conjunction with an encounter threshold. The coverage and redundant transmissions are plotted against the BC threshold, β. A heuristic value of $\beta = 2\log_2 n$ or $\beta = 3\log_2 n$ was suggested in Section 4.1. In the case of $n = 64$, that means $\beta = 12$ or $\beta = 24$.

Figure 4.7 shows that either heuristic is fine at high density, but even the larger one is not quite sufficient when the density is low; then an almost full coverage is achieved with $\beta = 35$. Introducing an encounter threshold of 10 in addition to the broadcast threshold makes no appreciable difference to the coverage.

According to figure 4.8, using a broadcast threshold sufficient to achieve full coverage at low density, produces an average of about 6.5 redundant transmissions per node (the addition of an encounter threshold reduces that number to slightly under 6). This should be compared with the corresponding number of more than 9 redundant transmissions per node in the absence of a reduction policy (figures 4.2 and 4.6). The gain is on the order of 25-30%, which is at least as good as the one achieved by the timer and history policies. At high node density, the BC policy achieves full coverage for values of β as low as 6, when there are about 2 redundant transmissions per node. At the point where $EG(\tau)$ achieves full coverage ($\tau = 2$ or 3), there are just over 3 redundant transmissions per node: a gain of about 35%. This is somewhat lower than the gain achieved by the other reduction policies.

The experiments described so far were also carried out in the context of the Manhattan Grid mobility model. The corresponding results are displayed in figures 4.9 – 4.14. The following is a summary of those results, and the way they compare with the Random Waypoint model.

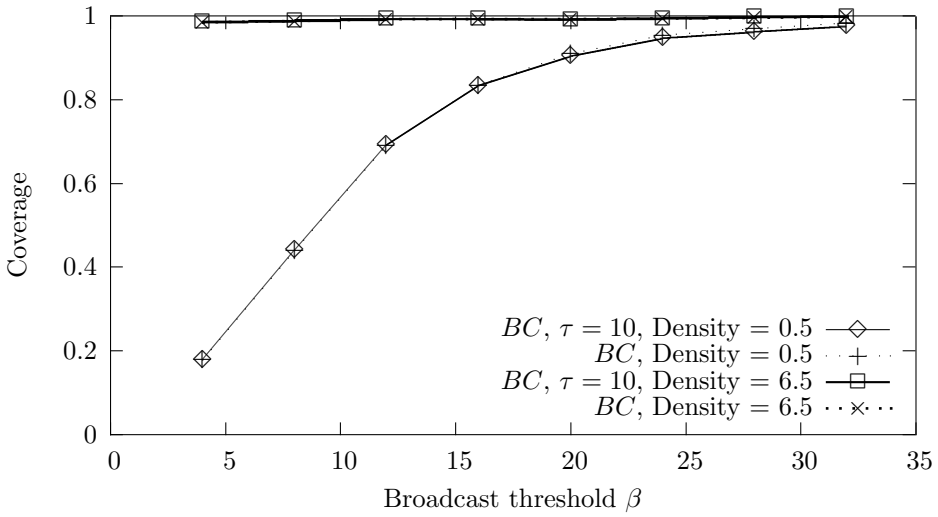

Figure 4.7: Coverage: broadcast count reduction; Random Waypoint

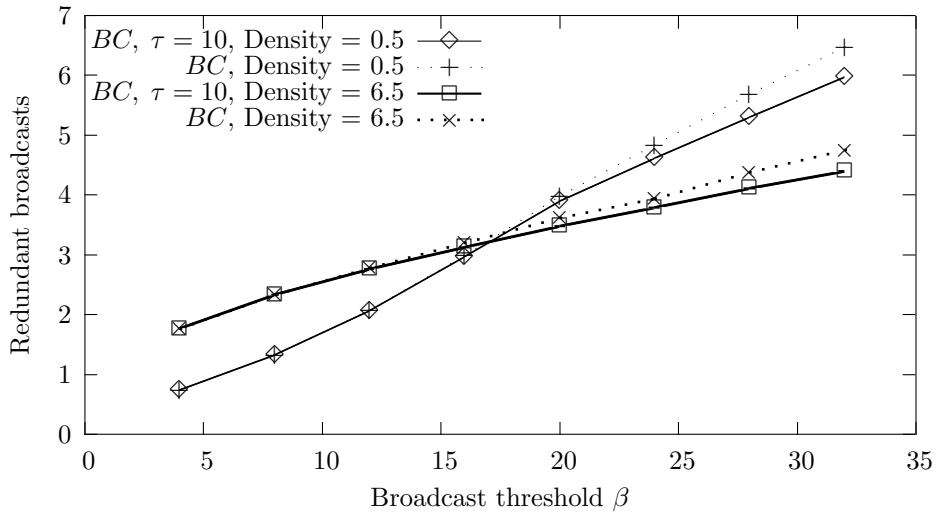

Figure 4.8: Redundant broadcasts: broadcast count reduction; Random Waypoint

Low density

1. Full coverage is considerably harder to achieve for the Manhattan Grid than for Random Waypoint. The protocol $EG(\tau)$ requires thresholds $\tau \geq 18$. This is due to the fact that, at every encounter, a node is more likely to meet nodes it has already met before, rather than new ones.

2. The loss of coverage due to the application of any broadcast reduction policy is greater for the

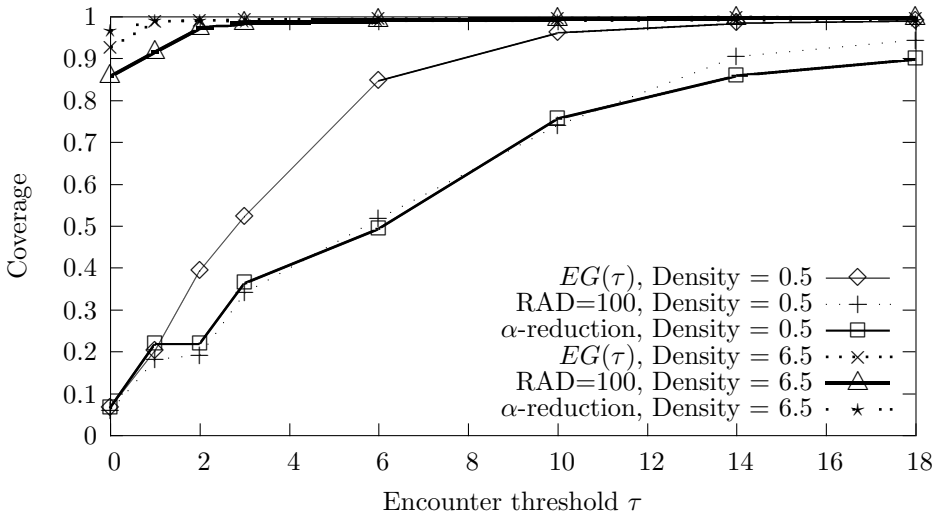

Figure 4.9: Coverage: RAD and α-reduction policies; Manhattan Grid Model

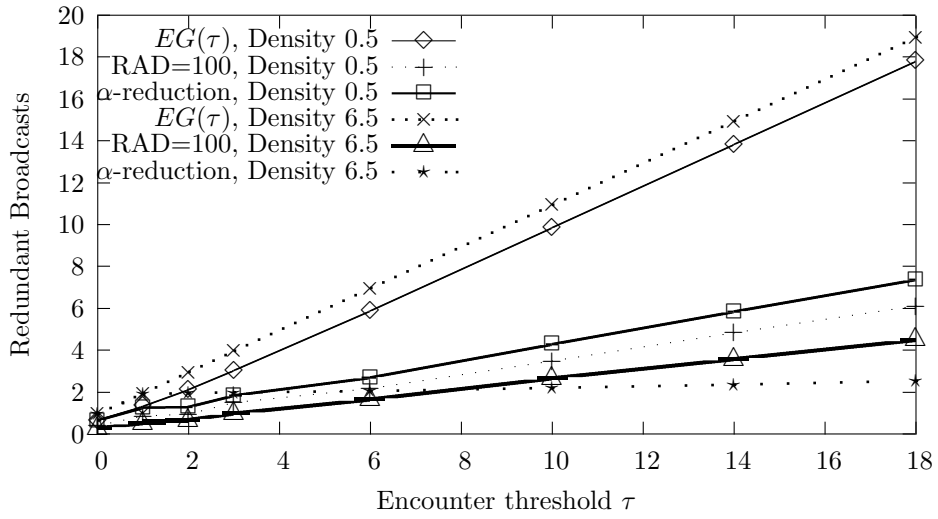

Figure 4.10: Redundant broadcasts: RAD and α-reduction policies; Manhattan Grid Model

Manhattan Grid than for Random Waypoint.

3. The gains achievable in the average number of redundant transmissions are larger (about 60-70%) for the Manhattan Grid than for Random Waypoint (see comment in item 1).

4. As before, the gains achieved by the RAD, α-reduction, EH and PH policies are quite similar.

5. The Broadcast Count policy needs larger thresholds ($\beta > 80$) to achieve high coverage. However, the gains it yields in redundant transmissions are also higher than for Random Waypoint. The addition of an encounter threshold makes a much bigger difference now.

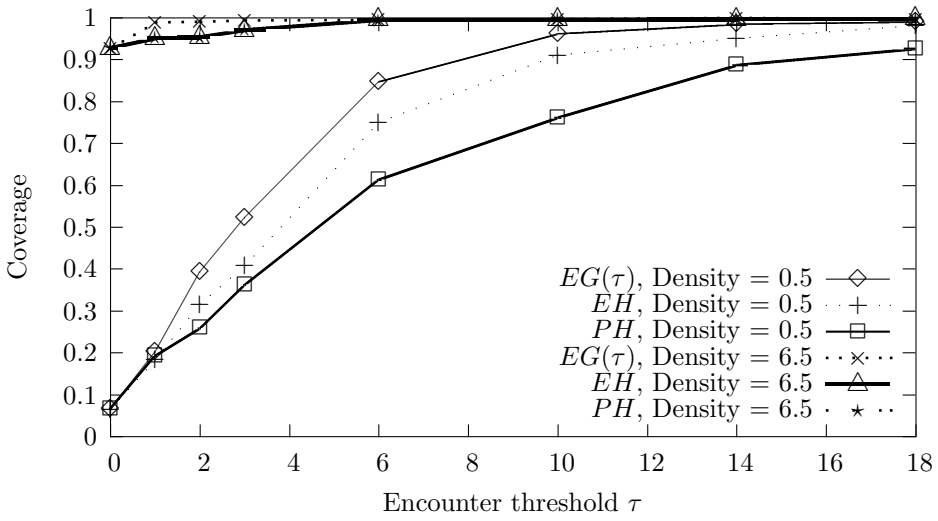

Figure 4.11: Coverage: encounter and propagation histories; Manhattan Grid Model

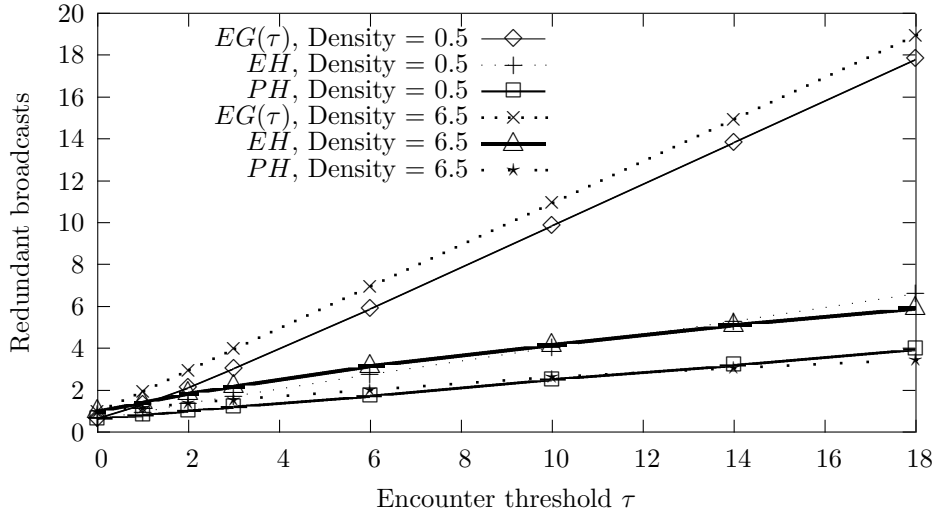

Figure 4.12: Redundant broadcasts: encounter and propagation histories; Manhattan Grid Model

High density

1. All policies achieve high coverage with a few broadcasts per node. A little coverage is lost by the RAD, EH and PH policies.

2. For encounter threshold values that achieve full coverage ($\tau \approx 6$), the RAD, α-reduction, EH and PH policies achieve reductions in redundant transmissions on the order of 60-70%.

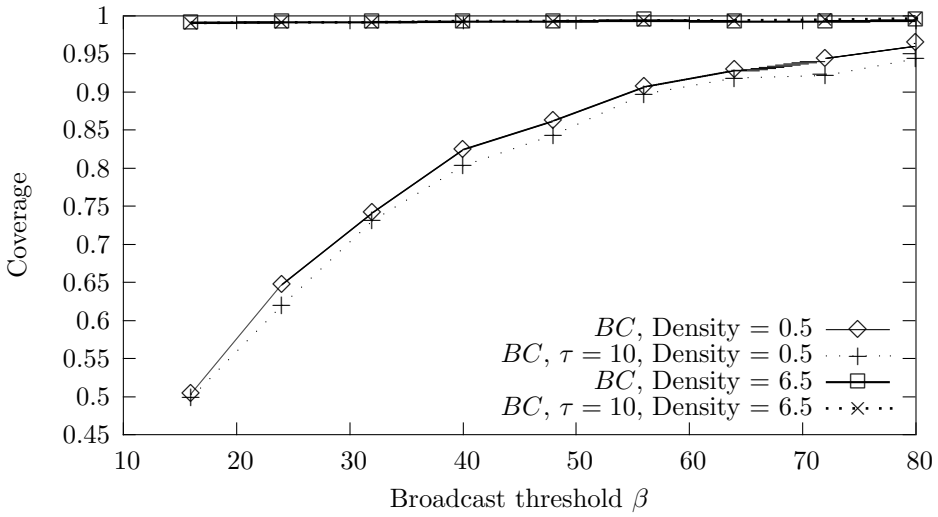

Figure 4.13: Coverage: BC reduction; Manhattan Grid Model

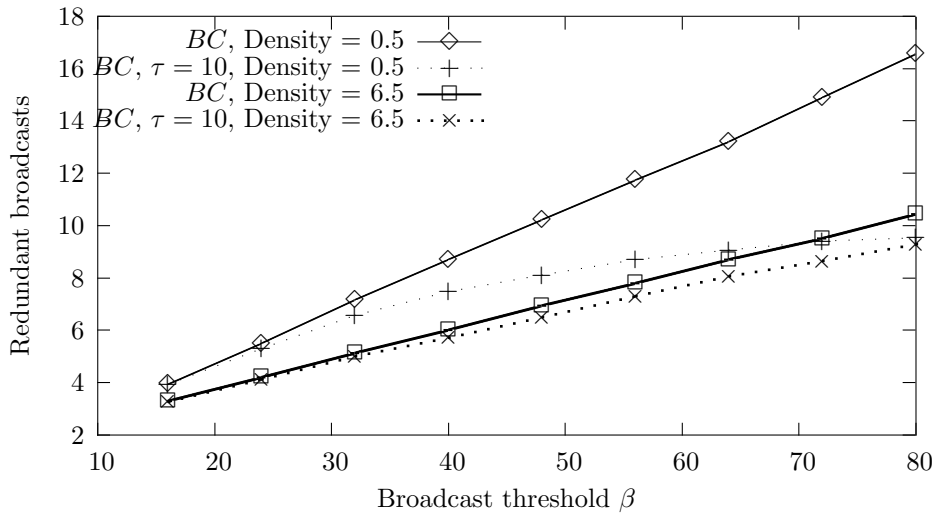

Figure 4.14: Redundant broadcasts: BC reduction; Manhattan Grid Model

3. For broadcast threshold values that achieve full coverage ($\beta \approx 16$), the BC reduction policy performs less well than the others.

4.6 Conclusions

Attempting to reduce the number of redundant transmissions used in the propagation of a message is clearly a worthwhile effort. The gains are perhaps not very important when the node density is high, since only a few broadcasts per node are then enough to achieve full coverage. However, when the density is low, one needs to set a large threshold on the number of broadcasts per node in order to achieve a satisfactory coverage. Under those conditions, when many of the broadcasts made are redundant, even a modest percentage reduction of the latter is significant in absolute terms.

Our experiment have shown that, at low node density, the average number of redundant transmissions per node can be reduced by about 30% in the Random Waypoint mobility model, and by twice that amount in the Manhattan Grid model.

An interesting observation is that, at the threshold levels that are necessary to achieve high coverage, the simple policies — RAD, *α-reduction* and Broadcast Count reduction — perform no worse than the ones employing caches and message lists (EH and PH). In fact, the *α-reduction* policy can be used with a high threshold over a range of densities, and still produce large savings in redundant transmissions.

Combinations of policies have been investigated to a limited extent. For example, combining RAD with *α-reduction* shows promising results. A more complete study of different combinations will be a topic for future research.

Chapter 5

Multiple Messages

5.1 Introduction

Encounter Gossip as described in Chapter 3 is a single message protocol and is simulated with only one message in the network at any time. In a real network there would naturally be many messages propagated by many nodes using more bandwidth. Buffers in each node will need to carry more than one message at a time and contention for resources will occur. In this chapter we address some of the issues that arise when more than one propagation, of messages originating at different nodes at different times, may overlap. Specifically we identify and address two problems termed as *Encounter Redundancy* and *Departure Redundancy*. We introduce a set of algorithms that mitigate these redundancies. The efficacy of addressing these problems is then examined using simulation. Furthermore we examine the performance of Encounter Gossip as a high coverage broadcast protocol and discuss the feasibility of achieving a stricter requirement of source order delivery.

The next Section 5.2 of this chapter refreshes us on the workings of Encounter Gossip. Section 5.3 describes Encounter Redundancy and Departure Redundancy, specific problems that can arise when multiple messages are in the network. We present a solution to this problem in Section 5.4.1 and compare the performance of the original and modified algorithms in Section 5.6. We discuss how source order can be applied to Encounter Gossip in Section 5.7 and finally we summarise our results and findings in Section 5.9.

5.2 Encounter Gossip Again

Recall that for a single message scenario the protocol behaves as follows (see chapter 3).

1. Upon receiving or originating a new message, m, store it, together with an associated counter, $c(m)$, which is set to zero. Add the sending node to the current neighbourhood, unless already present. If the current neighbourhood contains nodes other than the sending one, broadcast m_i.

2. At every encounter thereafter, if $c(m) \leq \tau$, broadcast m and increment $c(m)$ by 1.

3. When $c(m) = \tau + 1$, remove m from memory (but keep its sequence number in order to remember that it has been handled).

To achieve such multiple propagations, each node must maintain a buffer of all messages that it has received, together with the corresponding counts indicating how many times each message has been broadcast. A simple generalisation of the $EG(\tau)$ protocol, where each node can keep track of up to M messages in the process of propagation, has the following structure.

1. Upon receiving or originating a new message, m_i, if there is room in the local buffer, store m_i in it, together with an associated counter, $c(m_i)$; the latter is set to zero. Add the sending node to the current neighbourhood, unless already present. If the current neighbourhood contains nodes other than the sending one, broadcast m_i and increment $c(m_i)$ by 1. If the buffer is full when m_i arrives, it is rejected.

2. At every encounter thereafter, for each buffered message, m_i, if $c(m_i) \leq \tau$, broadcast m_i and increment $c(m_i)$ by 1.

3. When $c(m_i) = \tau + 1$, remove m_i from the buffer (but keep its sequence number in order to remember that it has been handled).

This protocol, the Original Algorithm will be referred to as 'Simple Encounter Gossip with Multiple Messages' and will be denoted by $SEG(\tau, M)$. For the purposes of this study, M is chosen sufficiently large so that messages are never rejected.

5.3 The Problem

We present two scenarios that illustrate our problem. The first describes the redundancy when an encounter occurs during buffer transmission, we call this Encounter Redundancy. The second scenario describes the redundancy caused by an encountered node departing before an entire buffer of messages is transmitted, we call this Departure Redundancy. Both of these cause more transmissions than necessary to propagate a buffer of messages to a new node. Any extra transmissions made during the operation of a protocol increase the likelihood of collision and thus reduce the coverage achieved.

5.3.1 Encounter Redundancy

Broadcasting several messages one after another may take quite a long time. If, during that period, nodes are likely to join or leave the current neighbourhood, the efficiency of $SEG(\tau, M)$ can be

seriously impaired. Suppose, for example, that node 1 encounters node 2 and starts broadcasting k messages, $m_1, m_2, \ldots, m_k$, that are currently in its buffer. During the transmission of m_i, node 2 leaves the neighbourhood. Then the transmissions of $m_{i+1}, m_{i+2}, \ldots, m_k$ are unnecessary and it would be better to suppress them.

The following provides an example which demonstrates how and when Encounter Redundancy Occurs, and what its effects are for EG.

- Node $Node_1$ has a message buffer containing n messages $m_1 \ \ldots \ m_n$
- $Node_1$ encounters $Node_2$
- $Node_1$ schedules transmission of $m_1 \ \ldots \ m_n$
- Before transmission of m_i, $Node_1$ encounters $Node_3$
- $Node_1$ schedules transmission of $m_1 \ \ldots \ m_n$

This means that:

i $m_1 \ \ldots \ m_n$ are transmitted twice to $Node_2$ due to the two schedules.

ii $m_i \ \ldots \ m_n$ are transmitted to $Node_3$ due to the first schedule during which it arrives.

iii $m_i \ \ldots \ m_n$ are transmitted a second time to $Node_3$ during the second transmission. This is called *encounter redundancy*.

There are two distinct redundancies here: a)Transmission of $m_1 \ \ldots \ m_{i-1}$ twice and b) Transmission of $m_i \ \ldots \ m_n$ twice. The arrival of $Node_3$ is an unforseen event. If it were known before any transmission schedules were begun that $Node_3$ would arrive, transmission of messages $m_1 \ \ldots \ m_{i-1}$ could have been delayed and thus only transmitted each message once; this would avoid redundancy a) altogether. Since this information is unavailable this redundancy a) is unavoidable. Once $Node_3$ arrives the following messages are transmitted: $m_i \ \ldots \ m_n$, then $m_1 \ \ldots \ m_{i-1}$, then $m_i \ \ldots \ m_n$ again. Our solution will avoid redundancy b) by not transmitting $m_i \ \ldots \ m_n$ twice.

Observation: Any increase in transmissions increases the likelihood of collision. Collision is undetectable in our scenario and thus this redundancy reduces coverage by increasing the likelihood that transmissions will fail to be delivered.

5.3.2 Departure Redundancy

Alternatively, suppose that during the transmission of m_i, node 3 joins the neighbourhood. This is a new encounter, so $SEG(\tau, M)$ schedules another broadcast of all k messages (assuming that their counters have not exceeded τ), to begin as soon as the current schedule completes. The

resulting retransmission of messages $m_1, m_2, \ldots, m_i$ is necessary, because node 3 has not received them. However, the repeated transmission of $m_{i+1}, m_{i+2}, \ldots, m_k$ is unnecessary; besides wasting power, it hastens the termination of the protocol and hence reduces the probability of full coverage.

The following provides an example which demonstrates how and when Encounter Redundancy Occurs, and what its effects are for EG.

- Node $Node_1$ again has a message buffer containing n messages $m_1 \ \ldots \ m_n$
- $Node_1$ encounters $Node_2$
- $Node_1$ begins transmission of $m_1 \ \ldots \ m_n$
- After transmission of $m_{i-1}, 1 < i \leq n$, but before transmission of m_i, $Node_2$ leaves the neighbourhood.

In this case messages $m_i \ \ldots \ m_n$ are transmitted redundantly. This problem is particularly damaging as it means the encounter threshold counter will be incremented unnecessarily $m_i \ \ldots \ m_n$ and thus reach τ and stop broadcasting early.

Upon detecting the departure of node $Node_2$ we should cancel transmission of messages $m_i \ \ldots \ m_n$.

Observation: This redundancy reduces coverage by two means:

- Redundancy increases the likelihood of transmission collisions.
- Encounter threshold counter affected causing early termination of the protocol.

[Note] At high speeds, encounters happen more frequently, which helps to cause Encounter Redundancy. Also high speed encounters are likely to be of a lower duration, which would tend to cause more frequent Departure Redundancy. We expect low density networks to suffer more than high density cases as they are more dependant on encounters for coverage.

5.4 Towards a Solution

Another problem caused by premature departures is that messages near the tail of a $FIFO$ buffer are less likely to be propagated successfully (because the target node leaves before their turn comes), than those near the head. However, that problem is easily cured by using a 'circular' buffer instead of a $FIFO$ one. An integer indicating the index of the currently transmitted message is incremented by 1 (modulo M) after each transmission. A newly received or locally generated message is inserted immediately to the left of the current index (modulo M). See figure 5.1.

- $Node_1$ again has a message buffer containing

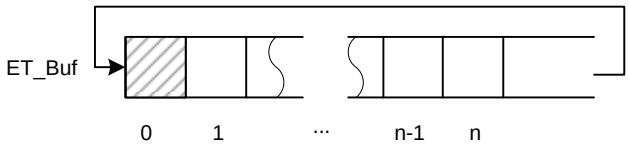

Figure 5.1: Encounter Transmission Buffer ET_{buf}

- n messages $m_1 \ \ldots \ m_n$
- $Node_1$ encounters $Node_2$
- $Node_1$ begins transmission of $m_i \ \ldots \ m_n$
- Before Transmission of m_i, $Node_1$ receives a new locally sourced message from transport layer.
- $Node_1$ transmit's m
- m is added to message buffer in position $m_n + 1$

If a circular buffer is used, $Node_1$ will therefore transmit m twice to $Node_2$ once as a blind flood, and once again during the encounter. To avoid this whilst using a cyclic buffer it is important to:

- Insert any incoming messages between m_{i-1} and m_i ie. in the position before the pointer.
- For each node encountered, record which message should be the last one sent.

5.4.1 The Solution

In order to eliminate the redundant transmissions which may result from arrivals or departures during broadcasts, each node maintains, in addition to the list of nodes in the current neighbourhood, a sublist of 'destination' nodes: these are nodes that have still not received all the messages in the circular buffer. For each target node, the index of the 'last-due' message, i.e. the last message that it needs to receive, is recorded. Step 2 of the $SEG(\tau, M)$ protocol is modified as follows:

2. At every encounter thereafter, add the newly encountered nodes to the target list, with last-due index equal to the current index - 1 (modulo M). Then, for each buffered message, m_i, if $c(m_i) \leq \tau$ *and* the target list is not empty: broadcast m_i; increment $c(m_i)$ by 1; remove from the target list all nodes whose last-due index is equal to i, as well as all nodes that have meanwhile left the neighbourhood.

A node is deemed to have left the neighbourhood if no beacon has been received from it during a given interval of time. The latter is normally chosen to be larger than the inter-beacon interval, in order to include possible network delays.

The modified protocol will be referred to as 'Encounter Gossip with Multiple Messages' and will be denoted by $EG(\tau, M)$.

5.5 Detailed Description

In this section we present and expand our proposed solution to minimise both Encounter and Departure Redundancy. We assume a buffer size sufficient to contain all messages transmitted in our scenarios. This is in effect an infinite buffer. The implications of this and possible e recovery mechanisms are briefly discussed in 7.7. The following list describes the main actions and data structures.

Data Structures

Encounter Transmission Buffer ET_{buf} (figure 5.1) is a circular double linked list containing the set of messages that need to be transmitted on encounters.

Neigh The nodes in the neighbourhood of a node.

Encounter Transmission Destinations ET_{dest} are the destination nodes: The subset of $Neigh$ to which we have not transmitted every message; they have not been present through an entire cycle of transmissions of ET_{buf}.

Neighbourhood Timeout We set neighbour time out t_i which is current time t plus a neighbourhood timeout value β.

Pointer to the *current* message. Thus $ETbuf[current]$ contains the current message to be transmitted next.

Message Each $ET_{buf}[i]$ contains message m, and a counter $c(m)$ which is set to zero.

Phantom Message The buffer also contains a single phantom message, with $id = 0$ which marks the start/end of the buffer. Thus when 'empty' the buffer contains 1 phantom message such that $phantom.next = phantom$ and $pahntom.index = 0$. This is required in case a nodes arrives when the current message is the only message in the buffer.

Actions

Receive Message

On receipt of a new message, it is given an incremental *id*, and inserted between *current.pevious* and *current* in ET_{buf}.

Encounter

- When a node $Node_k$ is encountered: it is added to $Neigh$ and ET_{dest}.
- $Node_i.finish$ is set to $ET_{buf}[current].previous$. Thus ET_{dest} can now be more precisely described as the subset of all nodes currently in $Neigh$ to which $ET_{buf}[Node_i.finish]$ has not yet been transmitted.
- If a transmission cycle is not in progress then begin Transmission.
- If a beacon from $Node_k$ is not received before $t \geq t_i$, (i.e. within neighbourhood timeout β), then $Node_k$ is removed from $Neigh$ and ET_{dest} if it is a member.

Drop Message

A message in ET_{buf} is discarded, if that message has been transmitted τ times.

Transmit Messages

- All messages in ET_{buf} are transmitted cyclicly commencing from $ET_{buf}[current]$. followed by $ET_{buf}[i].next$.
- If $ET_{buf}[Node_i.finish]$ is transmitted then $Node_k$ is removed from ET_{dest} and is now only a member of $Neigh$, the set of all neighbours. $Node_k$ is now a normal neighbour.
- If ET_{dest} is empty ($ET_{dest} = \{\}$) or ET_{buf} contains only the phantom message ($ET_{buf} = \{phantom\}$) terminate.

5.5.1 Algorithms

The following pseudo code defines four algorithms which form our *Modified Protocol*. Algorithm 1, *SetUp* defines the data structures we set up for the protocol as described in 5.4.1. After set up each algorithm will be run in a separate thread when its entry requirements are met. Algorithm 2, *ReceiveMessage* is run when messages are originated or received from another node. Algorithm 3, *BeaconReceived* is run when a beacon is received, it also initiates *Encounter*. Algorithm 4, $NeighbourhoodTimeout$ removes neighbours from ET_{dest} and $Neigh$ when timeout β expires. Algorithm 5, *Encounter* applies the modified multiple message policy to the transmissions.
This set of algorithms shall form the $ModifiedProtocol$.

$Neigh \leftarrow \{\}$; $Phantom.id \leftarrow 0$;
$Phantom.prev \leftarrow Phantom$;
$Phantom.next \leftarrow Phantom$;
$ET_{dest} \leftarrow \{\}$;
$ET_{buf} \leftarrow \{Phantom\}$;
$ET_{buf}.size \leftarrow 0$;
$ET_{buf}.nextId \leftarrow 1$;
$ET_{buf}.current \leftarrow Phantom$;

Algorithm 1: SetUp

On originating or receiving a message m
do
  if $(Neigh.size > 0)$; // If we have neighbours, transmit message
  then
    Transmit message;
  // Add message to ET_{buf}
  $m.id \leftarrow ET_{buf}.nextId$;
  $m.count \leftarrow 0$ $ET_{buf}.nextId \leftarrow ET_{buf}.nextId + 1$;
  $current.prev.next \leftarrow m$; // Insert m before $current$
  $m.next \leftarrow current$;
  $m.prev \leftarrow current.prev$;
  $current.prev \leftarrow m$;

Algorithm 2: ReceiveMessage

On receiving a beacon from $Node_l$
do
  set t_i to neighbourhood timeout β + current time t;
  if $Node_j \notin Neigh$; // This is a new encounter
  then
    $Node_j.finish \leftarrow current.previous$;
    $ET_{dest} \leftarrow ET_{dest} \cup \{Node_j\}$; // Add to $Neigh$ and ET_{dest}
    $Neigh \leftarrow Neigh \cup \{Node_j\}$;
    if *Encounter is not running* **then**
      Start *Encounter* (Algorithm 5).

Algorithm 3: BeaconReceived

On neighbourhood timeout t_i
do
  if $t > t_i$; // No beacon received from $Node_l$ within the last β
  then
    ; // Remove $Node_l$ from Neighbour hood, and Encounter Destinations
    if $Node_j \in ET_{dest}$
    then
      $ET_{dest} \leftarrow ET_{dest} - \{Node_j\}$;
    $Neigh \leftarrow Neigh - \{Node_j\}$;

Algorithm 4: NeighbourhoodTimeout

```
On Encounter  while ET_dest ≠ { }
do
    if ET_buf.size = 0
    then
        ET_dest ← {};                          // Empty the Encounter destination buffer
        return ;                                                  // no messages to transmit
    m ← ET_buf.current;
    if m.id ≠ 0;                                                  // If not the Phantom message
    then
        Transmit (m);
        m.count ← m_{count+ET_buf.size};
    foreach  Node_l in ET_dest
    do
        if Node_j.finish = m.id ;  // All messages in ET_buf have been transmitted to
        Node_l
        then
            ET_dest ← ET_dest − {Node_j} ;                  // Node_l is now a simple neighbour
            if ET_dest = {} then
                return; ;                                  // No more nodes in encounter so end
    current ← m.next;
    if m.count ≥ τ
    then
    ET_buf ← ET_buf − {m};
    m.prev.next ← m.next;
    m.next.prev ← m.prev;                                          // remove hole in buffer
```

Algorithm 5: Encounter

5.5.2 Example

Using the scenarios from 5.3 we show how the modified protocol addresses the problems of Encounter Redundancy and Departure Redundancy. Both scenarios share an identical beginning. We describe this in detail next, up until the differences in scenario which give the examples of Encounter or Departure Redundancy.

5.5.2.1 Scenario Set Up

We follow the interactions $Node_1$ experiences as it follows the $ModifiedProtocol$. We shall use the value $\tau = 3$.

Algorithm 1, SetUp

$Node_1$ has no neighbours so, $Neigh$ and ET_{dest} are the empty set $\{\}$. $Node_1$ has no messages yet so $Phantom$ message, with $id = 0$, is the only contents of ET_{buf} contains only the $Phantom$ message, its $size$ is set to 0 and $nextId$ is 1;

Node $Node_1$ **has a message buffer containing** n **messages** $m_1 \ \ldots \ m_n$

Algorithm 2, ReceiveMessage

$Node_1$ originates (receives from transport layer) n messages. Since $Node_1$ has no neighbours, suppress transmission. Add the message m to ET_{buf}. Message m is given $id = ET_{buf}.nextId$, and $ET_{buf}.nextId$ is incremented. Each message is inserted sequentially between $current.prev$ and $current.next$. Hence we have $ET_{buf} = Phantom(id = 0), m_1 \ \ldots \ m_n$.

$Node_1$ **encounters** $Node_2$

Algorithm 3, BeaconReceived

$Node_1$ receives a beacon from $Node_2$ and so executes algorithm 3, BeaconReceived. A timeout t_2 is set to current time t plus the specified neighbourhood timeout β. Since $Node_2$ is not a current member of the neighbourhood $Neigh$ we set $Node_2.finish$ to $current.previous$, which is m_n and add it to $Neigh$ and ET_{dest} and thus both are equal to $Node_2$. Since we are now encountering a node we start algorithm 5, $Encounter$.

$Node_1$ **schedules transmission of** $m_1 \ \ldots \ m_n$

Algorithm 5, Encounter

Since ET_{dest} is non empty we can enter the while loop. Each of the following iterations iterates around the circular buffer, until all messages have been transmitted to all nodes in ET_{dest}, i.e. until $ET_{dest} = \{\}$.

iteration 1

ET_{buf} contains several messages so we begin processing the *current* message m in ET_{buf}. The current message has $m.id = 0$, it is the phantom message so we do not transmit it. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: they do not. We move our pointer to *current* to *current.next* and since the *count* value of *phantom* is never increased it is always lower than τ it is never removed.

iteration 2 ... $(i-1)$

ET_{buf} still contains several messages so we begin processing the new *current* message m. This message has $id > 0$ so we transmit the message and increment its counter $m.count$ by the number of nodes that are being encountered $ET_{dest}.size = 1$, thus $m.count+ = 1$, leaving $m.count = 1$. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: they do not. We move *current* to point at *current.next* and we confirm that $m.count < \tau$ and so end the loop.

If a lower τ was used then this message could have reached it's transmission threshold and be discarded from ET_{buf}. We would then remove the hole in the buffer by creating a link between $m.prev$ and $m.next$.

[Note] $Node_1$ has transmitted $m_{1\ ...\ (i-1)}$ once to $Node_2$.

5.5.2.2 Encounter Redundancy

This scenario shows how the modification overcomes the problem of Encounter Redundancy that we introduced in Section 5.3. The following actions continue occur immediately after the Set Up from the previous section 5.5.2.1.

Before transmission of m_i**,** $Node_1$ **encounters** $Node_3$

Algorithm 3, BeaconReceived

$Node_1$ receives a beacon from $Node_3$ and so executes algorithm 3, BeaconReceived. A timeout t_2 is set to current time t plus the specified neighbourhood timeout β. Since $Node_3$ is not a member of our neighbourhood $Neigh$ we set $Node_3.finish$ to *current.previous*, which is $m_i - 1$ and add it to $Neigh$ and ET_{dest} and thus both are equal to $Node_2, Node_3$. *Since we are already in an encounter there we do not start algorithm 5, Encounter.*

$Node_1$ **schedules transmission of** $m_1\ \ldots\ m_n$

Algorithm 5, Encounter Continued ...

iteration $i\ \ldots\ n$

ET_{buf} still contains messages so we begin processing the new *current* message m. This

message has $id > 0$ so we transmit the message and increment its counter $m.count$ by the number of nodes that are being encountered $ET_{dest}.size = 2$, thus $m.count+ = 2$, leaving $m.count = 2$. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: they do not. We move *current* to point at *current.next*. We check and find that $m.count$ is not less than τ and end the loop.

[Note]

$Node_1$ has transmitted $m_{i\ \ldots\ n}$ once to $Node_2$ and $Node_3$.

iteration $n + 1$

We have now traversed the buffer to the end and are back at the start of the buffer. ET_{buf} still contains messages so we begin processing the *current* message m in ET_{buf}. The current message has $m.id = 0$, it is the phantom message so we do not transmit it. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: $Node_2.finish = 0$ so we remove $Node_2$ from ET_{dest}, thus $ET_{dest} = Node_3$. We move our pointer to *current* to *current.next* and since the *count* value of *phantom* is never increased it is always lower than τ it is never removed.

iteration $n + 2\ \ldots\ n + (i - 2)$

ET_{buf} still contains messages so we begin processing the new *current* message m. This message has $id > 0$ so we transmit the message and increment its counter $m.count$ by the number of nodes that are being encountered: $ET_{dest}.size = 1$, thus $m.count+ = 1$, leaving $m.count = 2$. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: they do not. We move *current* to point at *current.next*. We check and find that $m.count$ is not less than τ and end the loop.

iteration $n + i - 1$

ET_{buf} still contains messages so we begin processing the *current* message m in ET_{buf}. The current message has $m.id > 0$, so we transmit the message and increment its counter $m.count$ by the number of nodes that are being encountered: $ET_{dest}.size = 1$, thus $m.count+ = 1$, leaving $m.count = 2$. We check if any nodes in ET_{dest} have $m.id$ as their $finish$ value: $Node_3.finish = i - 1$ so we remove $Node_3$ from ET_{dest}, since $ET_{dest} = \{\}$.

iteration $n + i - 1$

ET_{dest} is now empty; there are no neighbours who are active on the current encounter. So we end the encounter.

[Note] Since iteration $n + 1\ \ldots\ n + i$ $Node_1$ has transmitted $m_{1\ \ldots\ i-1}$ once to $Node_3$ (and unavoidably to $Node_2$ for a second time).

This means that messages $m_1\ \ldots\ m_{i-1}$ are transmitted only once to $Node_3$ (twice to $Node_2$, unavoidably) and that $m_i\ \ldots\ m_n$ are transmitted only once to both$Node_2$ and

$Node_3$.

This shows that we avoid the redundant second transmission of messages $m_i \ \ldots \ m_n$.

5.5.2.3 Departure Redundancy

This scenario shows how the modification overcomes the problem of Encounter Redundancy that we introduced in Section 5.3. The following actions continue occur immediately after the Set Up from the earlier section 5.5.2.1.

before transmission of m_i, $Node_2$ leaves the neighbourhood.

Algorithm 4, NeighbourhoodTimeout

$Node_1$ has not received a beacon from $Node_2$ within the last β thus current time t is greater than t_2. $Node_2$ is removed from both ET_{dest} and $Neigh$.

Algorithm 5, Encounter Continued ...

iteration i

Since ET_{dest} is now the empty set $\{\}$, we end the encounter algorithm.

[**Note**] We make no more transmissions.

This means only messages $m_1 \ \ldots \ m_{i-1}$ are transmitted to $Node_2$. Thus messages $m_i \ \ldots \ m_n$ do not have their $m.count$ incremented and as such the transmission of these messages should not be adversely affected. An added bonus is that m_i is now the *current* message and will be transmitted first on the next encounter.

We have successfully avoided redundant transmission of $m_i \ \ldots \ m_n$.

5.6 Experimental results

A number of GloMoSim simulation experiments were carried out, aimed at evaluating the effect of various parameters on the performance of τ-propagation under the original and modified algorithms. The following factors were kept fixed:

The terrain is a square of dimensions $(1000\ m) \times (1000\ m)$. The number of nodes is kept fixed at $n = 64$. The node density (defined as the average number of nodes within a circle of radius equal to the wireless range) is varied by altering the wireless range. The interval between 'hello' signals for each node is 25 ms. The buffer in each node is capable of containing all messages sent in this scenario. The mobility pattern used is 'Random Waypoint'.

The speed, node density and encounter threshold were varied and performance of coverage was evaluated.

Each run starts with 100 messages originating from each node, and terminates when no node can propagate the messages further. For each set of parameter values, the simulation ran 25 times, with different random number seeds, and the performance observations were averaged.

Table 5.1 shows the improvement in coverage by the modified algorithm. This clearly shows that the largest percentage increase is with density 0.5, $\tau = 1$ at $100ms^{-1}$. An increase of 105.72%.

Encounters (τ)		0	1	2	4	6	10	14
Density	Speed							
0.5	2	0.00	-39.45	-2.71	-0.20	-0.01	0.01	0.00
	20	0.00	85.94	23.67	2.01	0.30	-0.01	-0.01
	60	0.00	91.58	53.55	6.67	1.36	0.00	-0.02
	100	0.00	105.72	54.70	7.04	1.47	0.03	-0.03
3.5	2	0.00	-0.86	-0.07	-0.03	-0.03	-0.01	0.00
	20	0.00	11.65	3.13	0.76	0.27	0.05	0.01
	60	0.00	6.35	1.64	0.43	0.12	0.01	0.00
	100	0.00	12.36	3.41	0.66	0.19	0.02	0.00
6.5	2	0.00	-0.39	-0.12	-0.03	-0.01	-0.01	0.00
	20	0.00	1.40	0.24	0.06	0.02	0.00	0.00
	60	0.00	4.43	0.65	0.18	0.06	0.00	0.00
	100	0.00	4.68	0.49	0.15	0.06	0.01	0.00

Table 5.1: Percentage Improvement in Coverage achieved by Modified over Original Algorithm

Figures 5.2 – 5.11 show the coverage achieved as a function of the encounter threshold, τ, for node densities ranging between 0.5 and 6.5, and speeds ranging between 2 ms^{-1} and 100 ms^{-1}.

The coverage achieved with the original protocol at low density (density = 0.5) can be seen in figure 5.2. As we might expect, there is a drop in coverage as speed increases. This is more pronounced where the encounter threshold τ is low and coverage even at low speed is below 1. The largest difference between modified and original algorithms in this instance is with encounter threshold $\tau = 1$, at $100ms^{-1}$ a difference of 0.35 (105.72%). This is caused by the Encounter and Departure Redundancy as described in Section 5.3. When speed is low we experience a low encounter rate with neighbours remaining longer within one another's neighbourhood. Under these conditions it is unlikely that one encounter be interrupted by another, thus Encounter Redundancy is less prominent. It is also unlikely that encounter's will be brief enough for nodes to leave before an entire buffer of messages is transmitted and so Departure Redundancy is minimal. When speed is high, the rate of encounters increases, and the duration of neighbourhood membership decreases. The increase in encounter rate increases the likelihood of Encounter Redundancy as encounters are more likely to interrupt one another. The decrease in neighbourhood membership duration increases the likelihood of Departure Redundancy since it is more likely that a neighbour will depart from a neighbour before

an entire buffer is transmitted.

Comparing figures 5.2 and 5.3 we can see that with the modified algorithm there is an improvement at all speeds and a large improvement in coverage for the higher speeds such that all speeds now achieve a similar high coverage result even at encounter threshold $\tau = 2$.

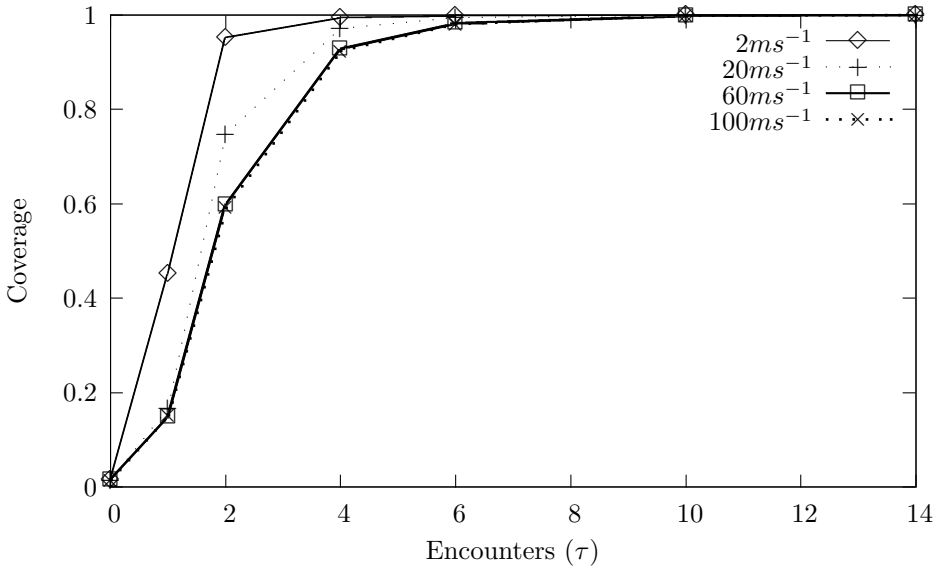

Figure 5.2: Coverage vs Encounters [Original algorithm] Density = 0.5

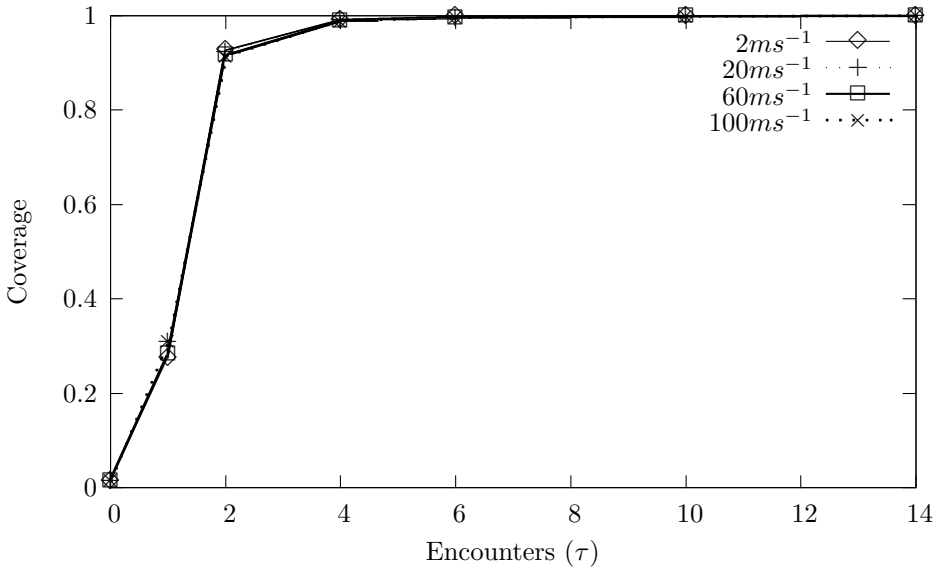

Figure 5.3: Coverage vs Encounters [Modified algorithm] Density = 0.5

Figures 5.4 and 5.5 provide a zoomed in scale of the top 0.95 coverage of each of the previous

figures. This allows us to see clearly that all speeds perform more similarly to one another in the modified algorithm and that after $\tau \geq 4$ all achieve greater than 0.98 coverage. By $\tau \geq 6$ all achieve higher than 0.995.

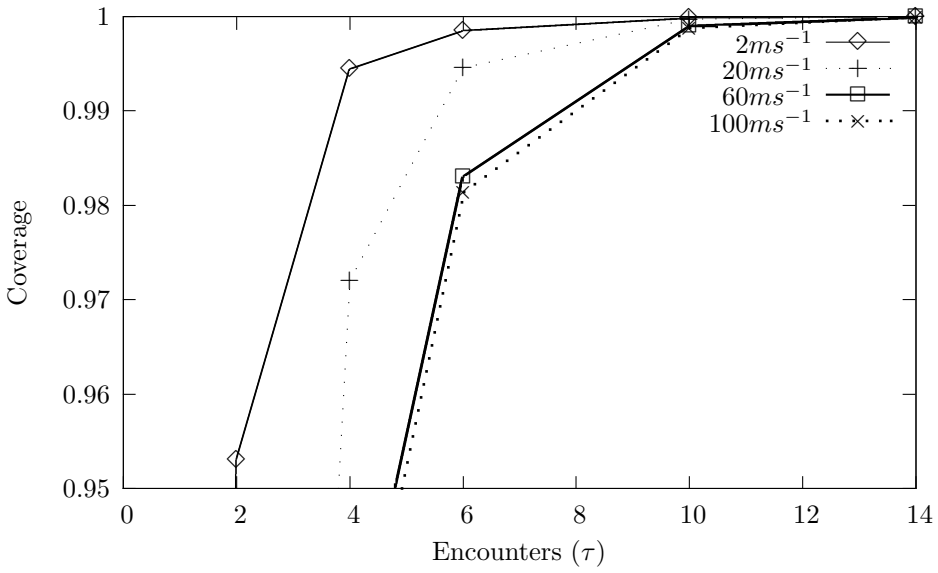

Figure 5.4: Encounters vs Coverage [Original algorithm] Density = 0.5

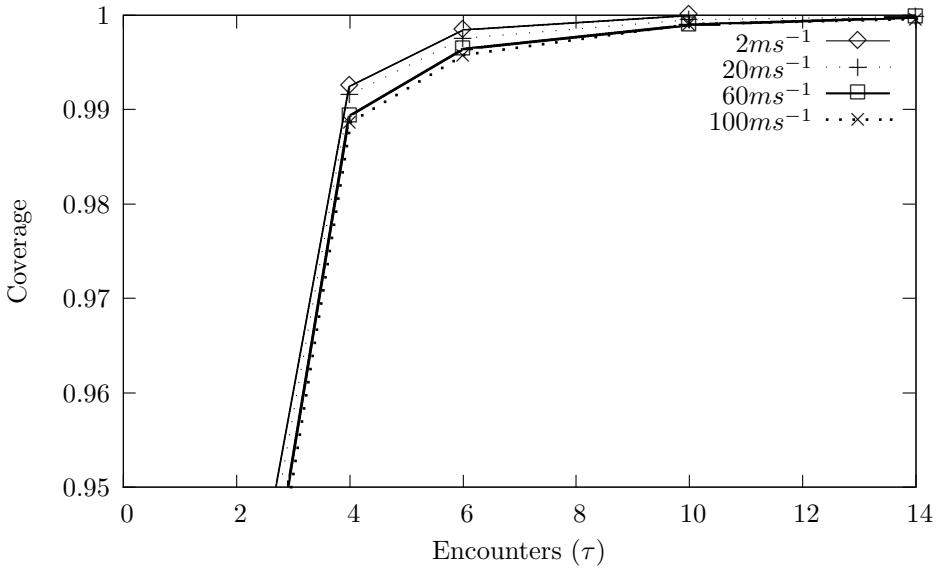

Figure 5.5: Encounters vs Coverage [Modified algorithm] Density = 0.5

Similarly in medium densities (density = 3.5) figures 5.6 and 5.7 we can see the same improvement in coverage although to a lesser extent. At encounter threshold $\tau = 1$ we see a recovery of the lost 0.15 (12.36%) coverage.

The zoomed figures for these, 5.8 and 5.9 show how all high speed mobilities match those of the low speed under the modified algorithm. Also that when $\tau \geq 4$ coverage is above 0.999.

In our high density scenario (density = 6.5) figures 5.10 and 5.11 show coverage gains but to an even lesser extent; at encounter threshold $\tau = 1$ we see a difference of only 0.1 (4.68%). Our zoomed figures 5.12 and 5.13 again show more clearly this performance. It is clear that the problems defined in 5.3 have most impact at high speed and low density and our modified algorithm is effective.

When density is high, the encounter process accounts for only a small proportion of coverage achieved. The lower the density is the more encounters are needed to propagate messages.

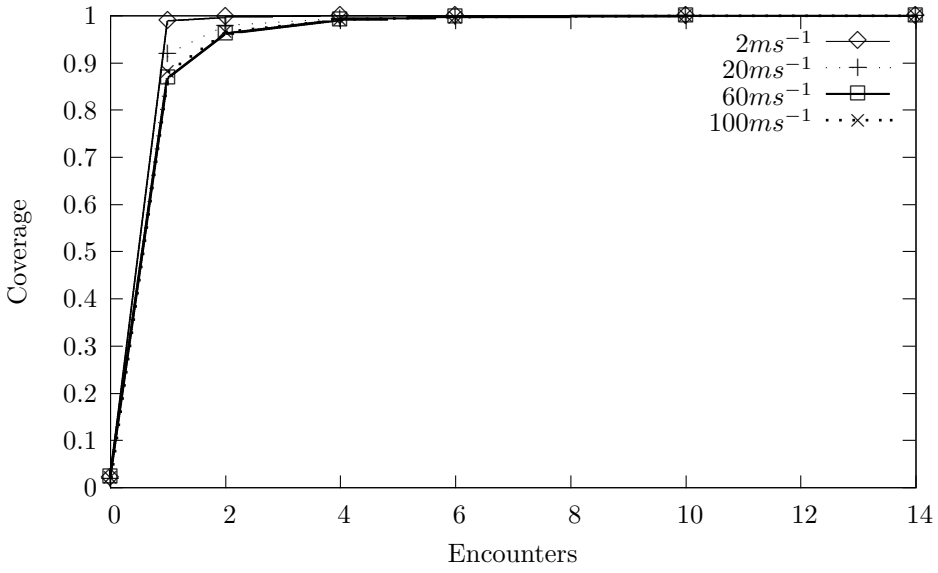

Figure 5.6: Coverage vs Encounters [Original algorithm] Density = 3.5

We also evaluate the number of redundant transmissions made to see how well our modified algorithm performs. We recap on the definition of a redundant transmission here, originally stated in 4.2. A broadcast is said to be 'useful' if it enlarges the set of nodes that have received m, i.e. if there is at least one node in the current neighbourhood for whom m is new. A broadcast which is not useful is 'redundant' (all nodes receiving that broadcast have already received m).

Figures 5.14 and 5.15 display redundant transmissions for the under the low density ($D = 0.5$) scenario. They show that we have a similar linear relationship that we have seen before in Chapter4 Section 4.5. The modified algorithm shows a reduction of approximately 1 redundant transmission when $\tau \geq 2$.

For the medium density scenario ($D = 3.5$) figures 5.16 and 5.17, the same linear pattern is observed and again a reduction of approximately 1 redundant transmission is visible in the modified

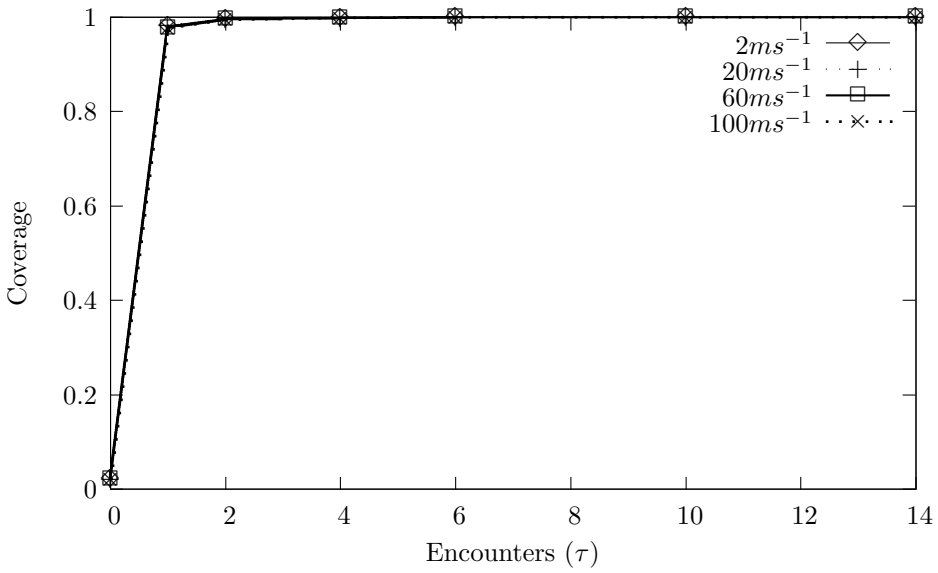

Figure 5.7: Coverage vs Encounters [Modified algorithm] Density = 3.5

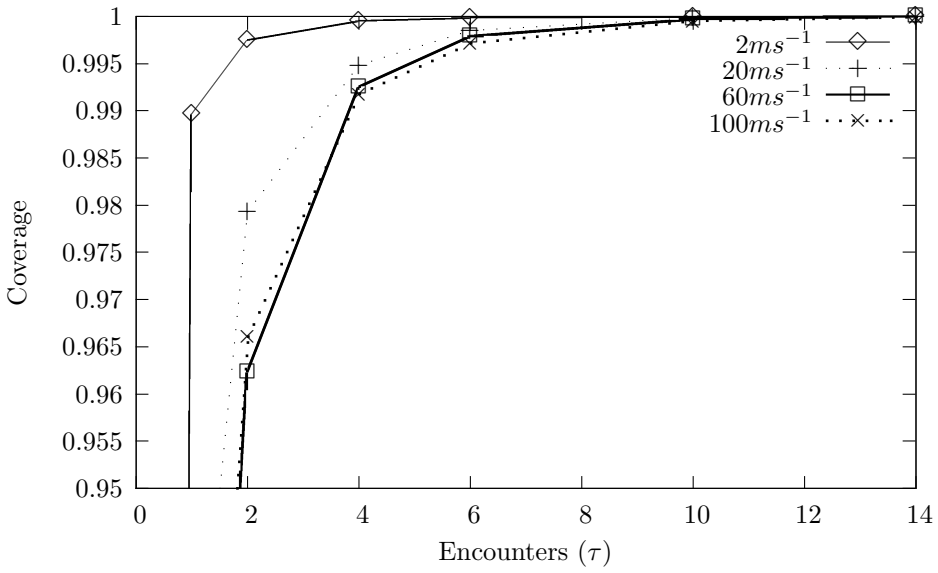

Figure 5.8: Encounters vs Coverage [Original algorithm] Density = 3.5

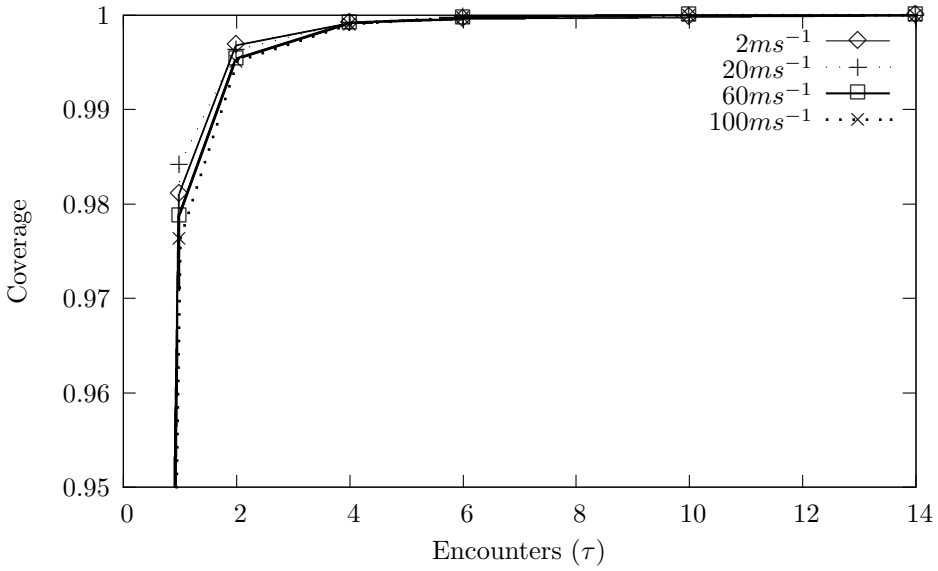

Figure 5.9: Encounters vs Coverage [Modified algorithm] Density = 3.5

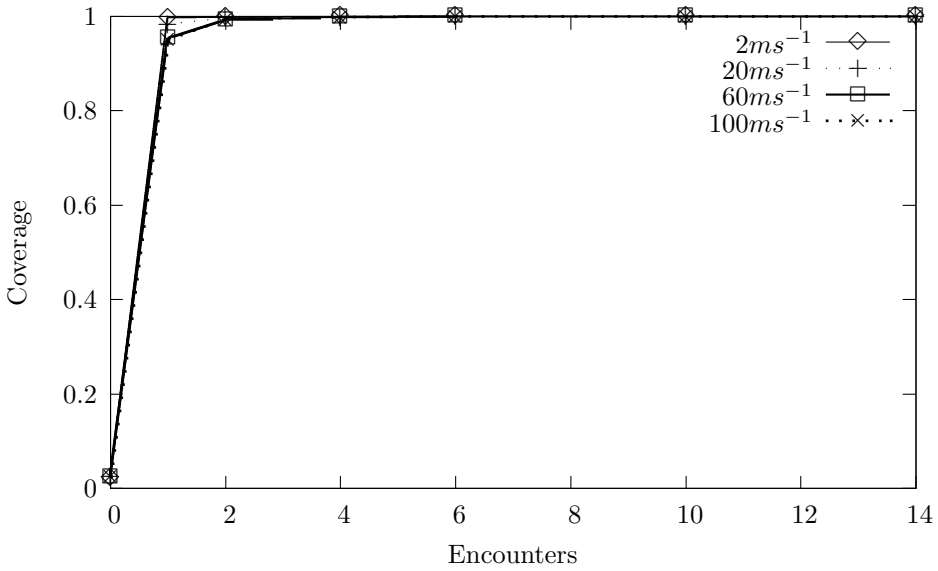

Figure 5.10: Coverage vs Encounters [Original algorithm] Density = 6.5

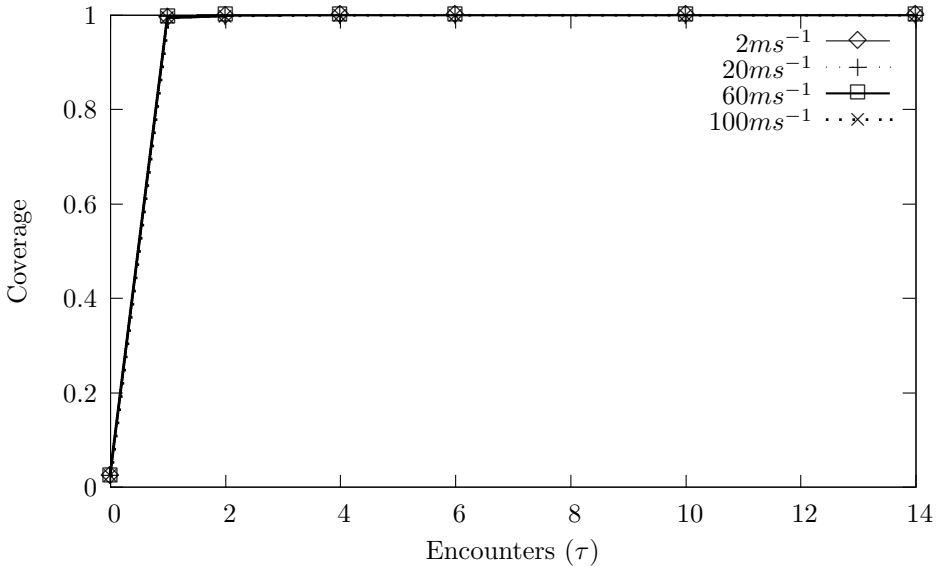

Figure 5.11: Coverage vs Encounters [Original algorithm] Density = 6.5

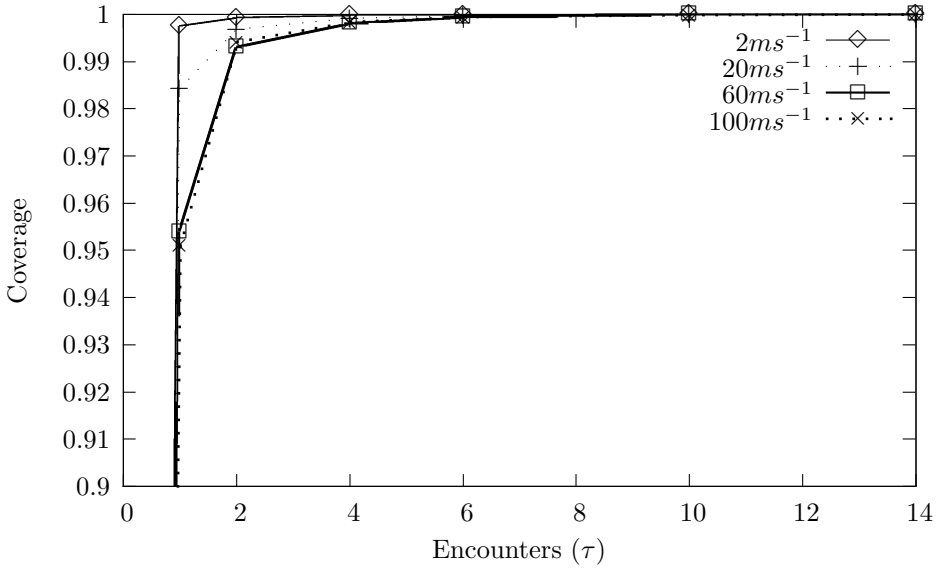

Figure 5.12: Encounters vs Coverage [Original algorithm] Density = 6.5

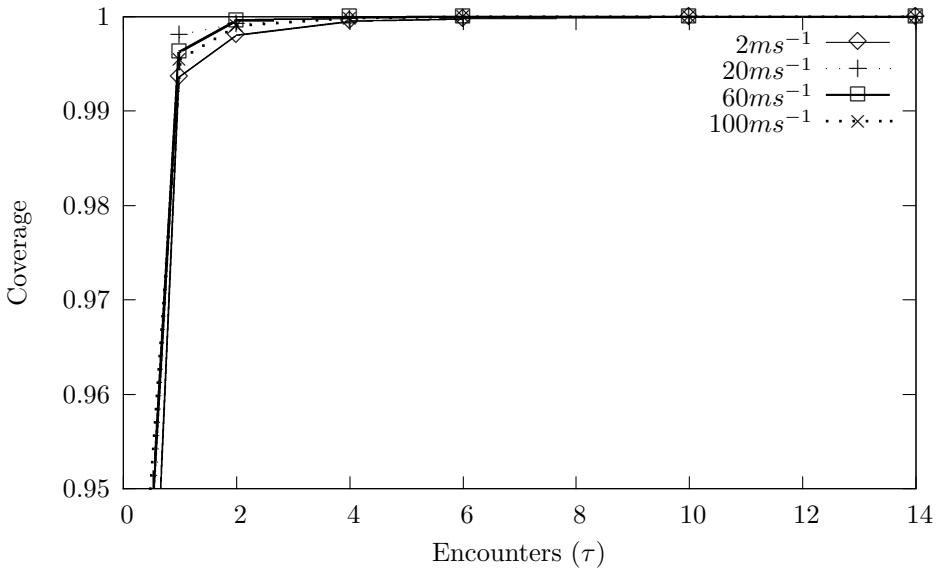

Figure 5.13: Encounters vs Coverage [Modified algorithm] Density = 6.5

algorithm.

Similarly in the high density scenario ($D = 6.5$) figures 5.18 and 5.19, the linear pattern is again observed and reduction of redundant transmissions is visible in the modified algorithm.

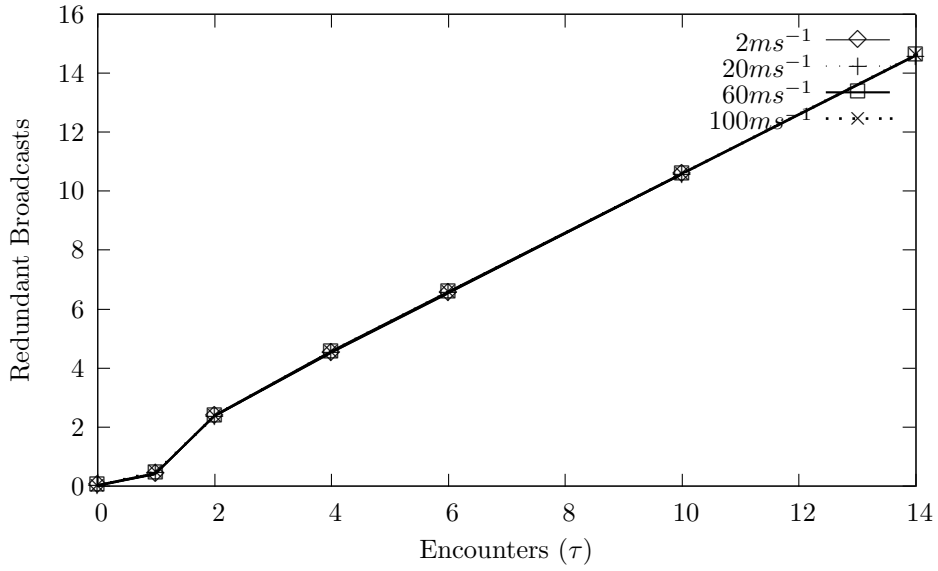

Figure 5.14: Encounters vs Redundant Broadcasts [Original algorithm] Density = 0.5

The reduction of around 1 redundant transmission across the simulation parameters equates to a relatively large reduction when scale is considered. Redundant broadcasts are measured per message, per node. In this simulation each of 64 nodes, propagate 100 messages each. So a reduction of 1

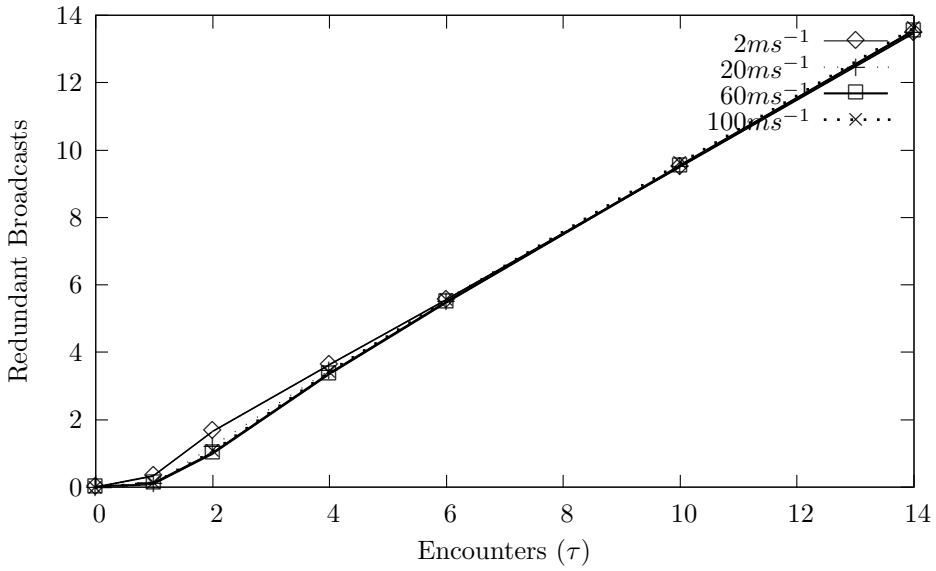

Figure 5.15: Encounters vs Redundant Broadcasts [Modified algorithm] Density = 0.5

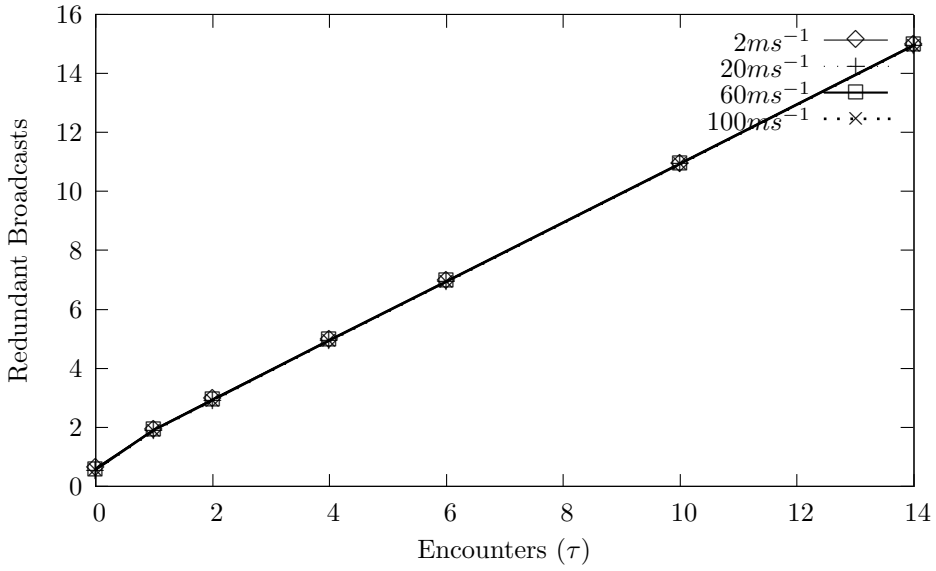

Figure 5.16: Encounters vs Redundant Broadcasts [Original algorithm] Density = 3.5

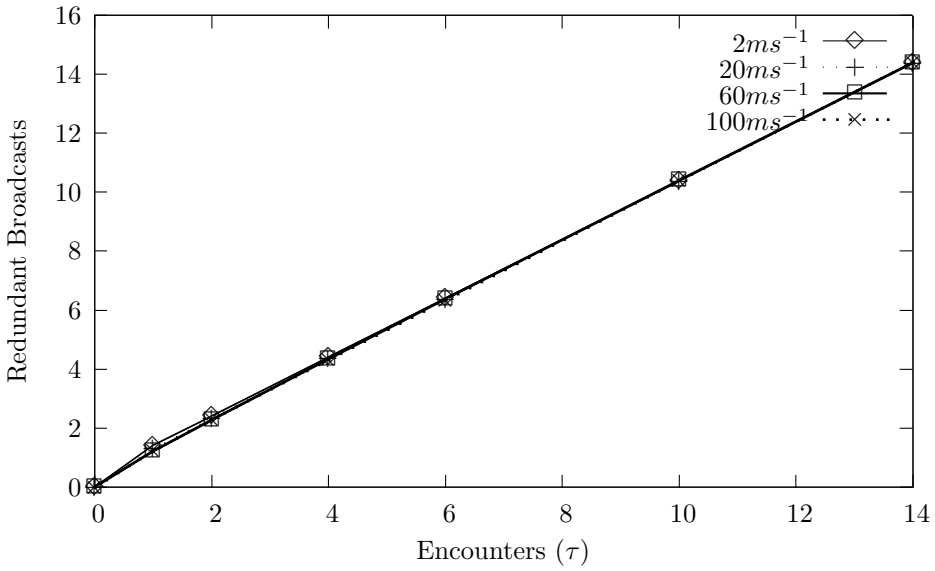

Figure 5.17: Encounters vs Redundant Broadcasts [Modified algorithm] Density = 3.5

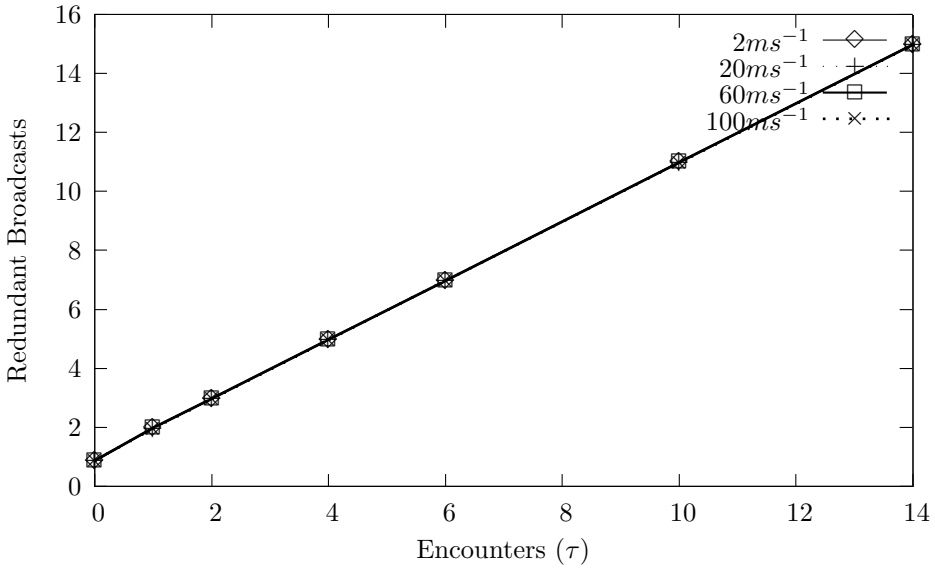

Figure 5.18: Encounters vs Redundant Broadcasts [Original algorithm] Density = 6.5

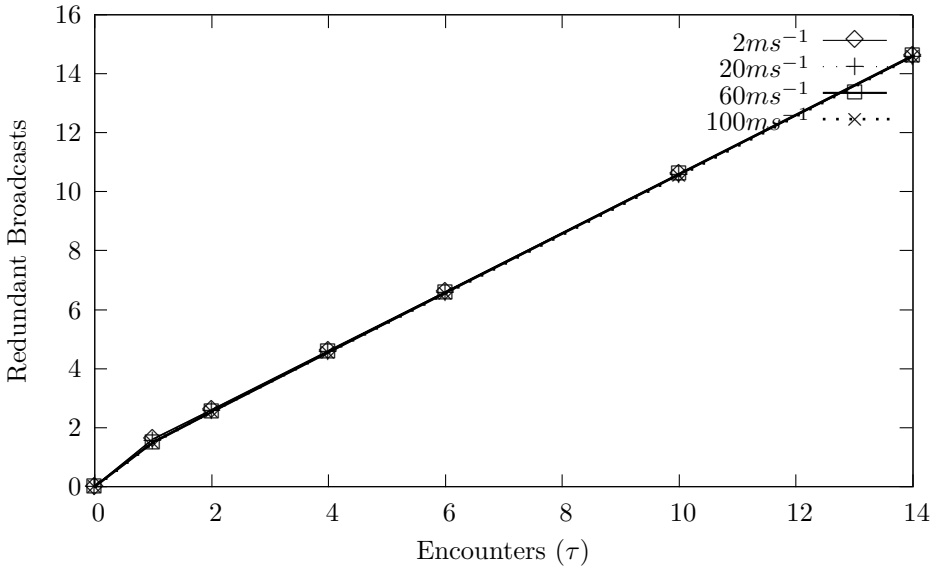

Figure 5.19: Encounters vs Redundant Broadcasts [Modified algorithm] Density = 6.5

redundant transmission reduces transmissions made in the network by 6400. We saw the benefit of this reduction in the increase of coverage shown in figures 5.2 ... 5.13.

5.7 Ordered Delivery

To achieve FIFO order, each message must be uniquely identified by its source and sequence number. FIFO delivery is simple to achieve. If process p receives and delivers all consecutive messages $m_0 \ \ldots \ m_{i-1}$ and as such is waiting to receive message m_i so that it may be delivered. Instead p receives messages $m_j \ \ldots \ m_k$ where $i < j < k$ it simply buffers messages $m_j \ \ldots \ m_k$ until message m_i is received, before continuing consecutive delivery. If coverage is complete(i.e. 1), it is sufficient to buffer all received messages ordering them by sequence number and delivering them is trivial.

We can reevaluate our earlier experiment from Section 5.6 in light of this to establish whether FIFO delivery is achievable under Encounter Gossip. We calculate a new performance measure *FIFO Coverage*. This is the probability that a random node receives all 100 messages from a random source node.

$$for\ Node_k \text{ receiving from } Node_l \qquad p_{ij} = \begin{cases} 1 & if\ Node_k \text{ received all messages from } Node_l \\ 0 & otherwise \end{cases}$$

$$\text{let } n = \text{number of nodes}$$

$$for \text{ each } run_k \qquad P_k = \frac{\sum_{i=1}^{n} \sum_{j=1, j \neq i}^{n} p_{ij}}{n(n-1)}$$

$$\text{let } N = \text{number of simulation runs}$$

$$\text{FIFO Coverage} = \frac{\sum_{k=1}^{N} P_k}{N}$$

Thus a *FIFO Coverage* of 0.75 would indicate that over all 25 runs, an average of 75% of the nodes achieved a coverage of 1 and thus could achieve FIFO delivery. Table 5.2 shows the FIFO Order Coverage improvement by the modified algorithm.

Figures 5.20 ... 5.25 show the performance of Encounter Gossip with regard to our new performance measure. We have included both original and modified versions of our algorithm in the figures so that the improvements achieved earlier be made more apparent. It can be seen that the FIFO Order figures show a larger difference between the performance of the original and modified algorithms visible than our earlier experiments. It is quite a visible improvement even at lower speeds. In the low density case (Density = 0.5), figures 5.20 and 5.21 we can see that at $\tau = 4, 100ms^{-}1$ there is an improvement of 0.75 (4592%!) in FIFO Order Coverage. The modified algorithm shows that at all speeds we achieve FIFO Order Coverage near 1 at $\tau \geq 10$, whereas in the original algorithm, $\tau = 14$ would have been necessary. This improvement is caused by the removal of Encounter and Departure Redundancy. A node who was to receive 99 messages from every node would have a fifo coverage of 0 if the missing message is not also received. The delivery of the last few messages is key here to achieving FIFO Order Coverage. Thus when the redundancies are removed, a more dramatic result can be observed than in standard coverage. In figures 5.22 and 5.23 we observe the medium density case (Density = 3.5), again see the vast improvement when moving from original to the modified algorithm. In this case the largest improvement is this time when $\tau = 2$, again at high speeds with $speed = 100ms^{-}1$, we see an improvement of 0.88 and at $60ms^{-1}$, give 0.89. Although the largest percentage increase was at $\tau = 1$ $speed = 20ms^{-}1$ with and increase of 4234.62%. The modified algorithm shows we would need $\tau \geq 4$ to get a coverage near 1, whereas using the original algorithm $\tau \geq 12$ was required. In figures 5.24 and 5.25 we see the high density case (Density = 6.5), improvement when moving from original to the modified algorithm is again visible. The largest improvement is at $speed = 100ms^{-}1$,when $\tau = 1$ we see an improvement of 0.88 (5035.71%). The modified algorithm shows we would need only $\tau \geq 2$ to get a coverage near 1, whereas using the original algorithm $\tau \geq 10$ was required.

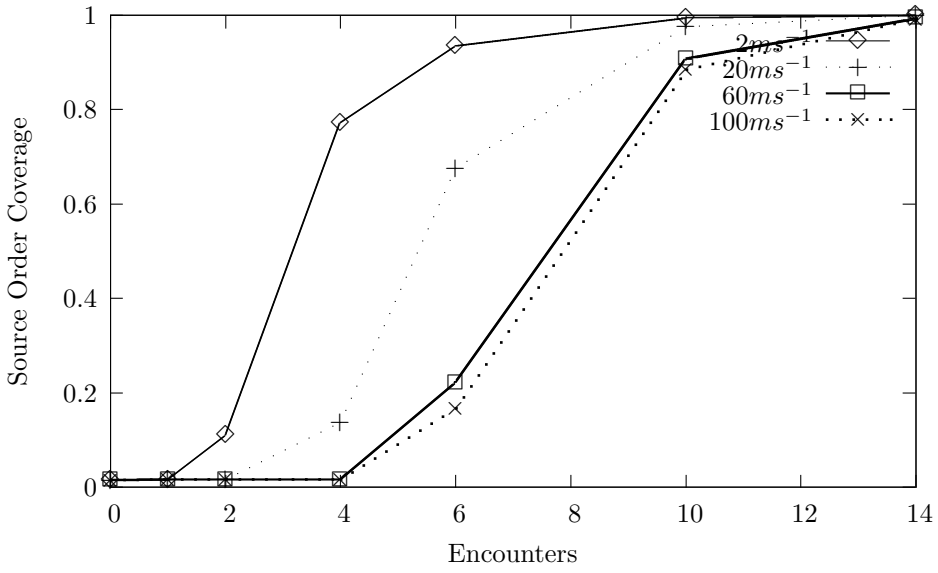

Figure 5.20: FIFO Coverage vs Encounters [Original algorithm] Density = 0.5

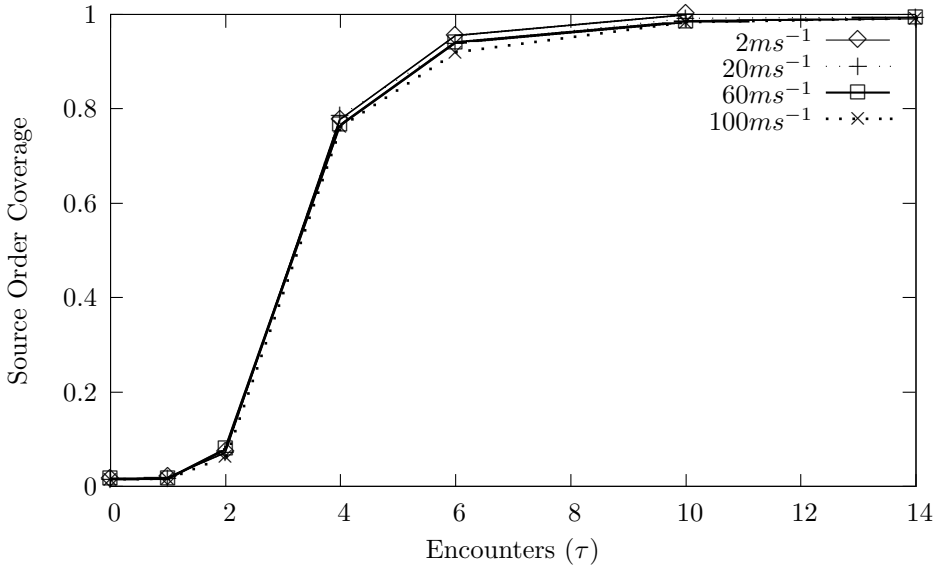

Figure 5.21: FIFO Coverage vs Encounters [Modified algorithm] Density = 0.5

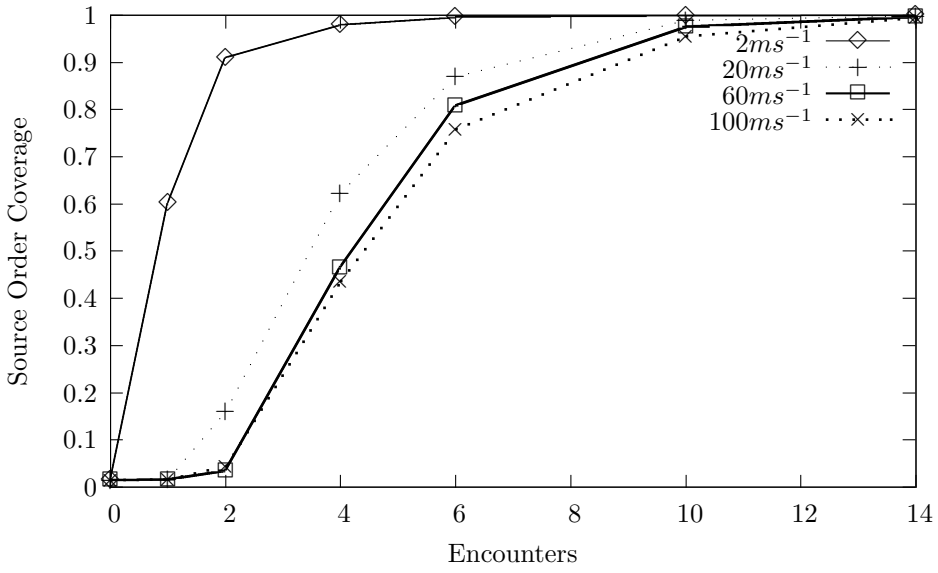

Figure 5.22: FIFO Coverage vs Encounters [Original algorithm] Density = 3.5

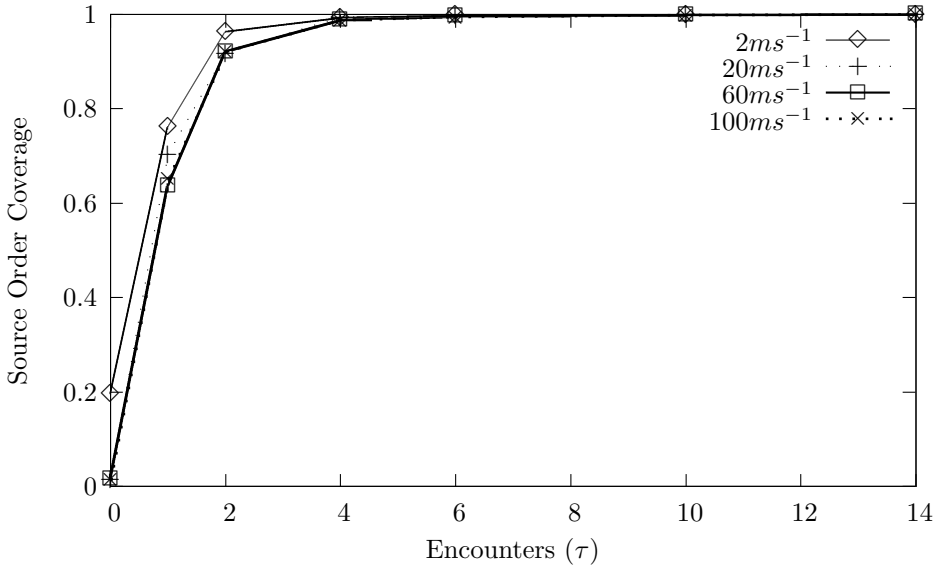

Figure 5.23: FIFO Coverage vs Encounters [Modified algorithm] Density = 3.5

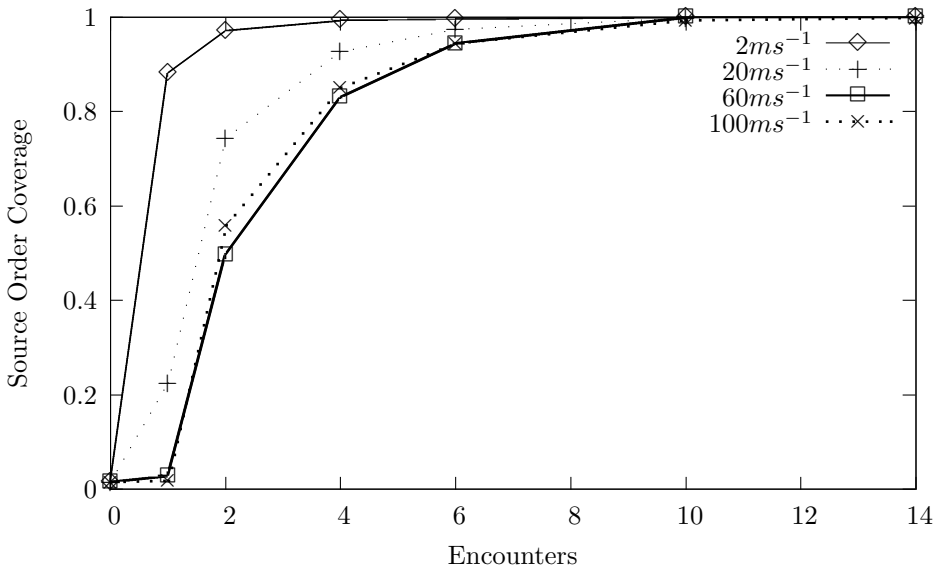

Figure 5.24: FIFO Coverage vs Encounters [Original algorithm] Density = 6.5

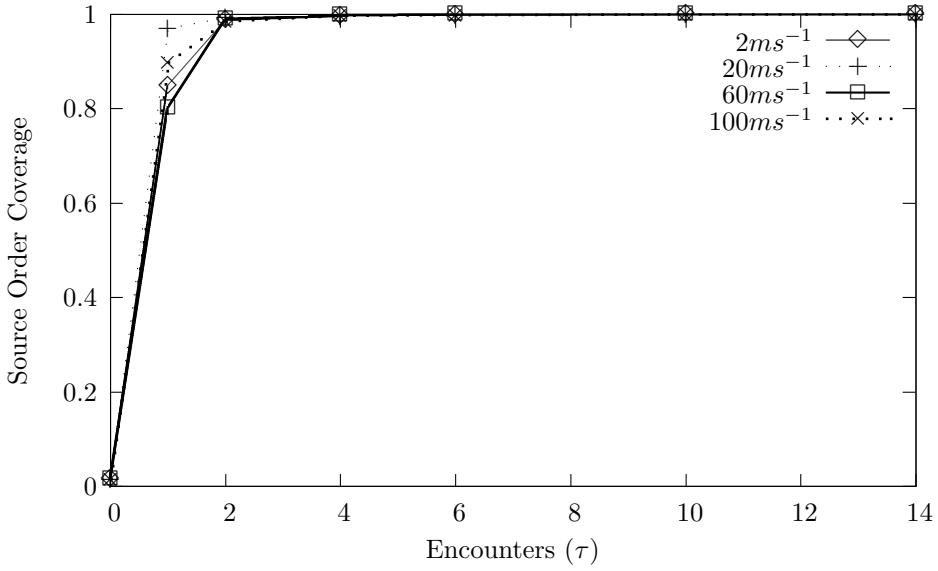

Figure 5.25: FIFO Coverage vs Encounters [Modified algorithm] Density = 6.5

Encounter		0	1	2	4	6	10	14
Density	Speed							
0.5	2	0.00	20.00	-34.29	0.73	2.27	0.50	0.00
	20	0.00	0.00	364.00	471.82	38.82	1.67	-0.50
	60	0.00	0.00	400.00	4611.54	326.06	8.61	0.00
	100	0.00	0.00	304.00	4592.31	451.31	11.31	0.06
3.5	2	0.00	26.24	5.77	1.40	0.19	-0.01	-0.08
	20	0.00	4234.62	471.98	59.58	14.74	1.07	0.13
	60	0.00	3819.23	2484.21	112.50	23.28	2.24	0.25
	100	0.00	4080.00	1947.22	125.86	31.05	4.52	0.57
6.5	2	0.00	-3.69	1.67	0.31	0.31	0.00	-0.06
	20	0.00	331.11	33.61	7.56	2.57	0.19	0.06
	60	0.00	2753.33	99.25	20.17	5.96	0.06	0.00
	100	0.00	5035.71	76.29	17.27	5.76	0.69	0.13

Table 5.2: Percentage Improvement in FIFO Order Coverage achieved by Modified over Original Algorithm

5.7.1 Message Recovery

We have identified that under the specified simulation conditions for densities $0.5, 3.5, 6.5$ we can achieve FIFO Order Coverage near 1 by applying $\tau = 10, \tau = 4, \tau = 2$ respectively. We argue that a simple recovery mechanism may be able to retrieve any missing messages in this case. A node should simply be able to request the missing message from a neighbour or at the next encounter and receive it with high probability.

Looking at our coverage and FIFO Order coverage figures for the modified algorithm we see very high average results. Let us examine a single run from our experiments. At density 0.5 with speed $2ms^{-1}$, $\tau = 10$ we choose the poorest result from the 25 runs. The coverage achieved on this run was 0.98 whilst the FIFO Order coverage was only 0.91. Using these values, a random node $Node_i$ should on average only be missing 2% of all messages. The probability of another node picked at random, $Node_j$ having all the missing messages for one node is the same as the probability of FIFO Order Coverage, in this case $P = 0.91$. The probability of a random $Node_j$ having just one message that $Node_i$ is missing is $P \geq 0.91 \leq 0.98$. Thus we have demonstrated that it is likely that a simple request from a random node will retrieve missing messages with high probability.

5.8 Comparison with Hypergossiping

In this section we compare the performance of Encounter Gossip with that of Hypergossiping [51], which is in many ways similar. On detecting an encounter, Hypergossiping establishes the need for message transmissions by first exchanging information on recently received messages. These exchanges are piggybacked onto successive hello beacons and are not counted in transmission cost. Hypergossiping thus attempts to save transmissions at the cost of delays before transmitting.

We use the experimental data reported in [51], figures 11 (a) and (b). We perform the same experiments as in that paper and compare the results on coverage achieved and transmissions carried out. Hypergossiping parameters not relevant to our protocol, such as message lifetime, are ignored. The Encounter threshold value τ is chosen as indicated in section 3.

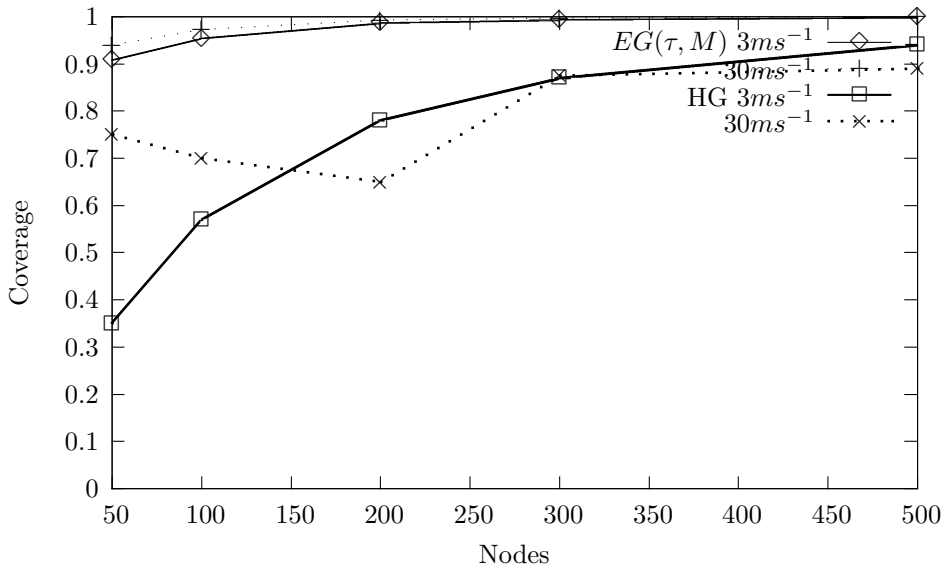

Figure 5.26: Coverage vs Nodes

Figure 5.26 shows the coverage achieved by Encounter Gossip and Hypergossiping for two different node speeds, and different node densities. We observe that in all corresponding cases, Encounter Gossip provides significantly higher coverage.

The improved performance in terms of coverage is of course paid for by higher number of transmissions. This is illustrated in figure 5.27, which plots the average number of transmissions per node under the two protocols, for different node densities. At low densities, Encounter Gossip carries out 5-6 times more transmissions than Hypergossiping, whereas that factor comes down to about 2 at high densities.

5.9 Summary and Further Work

In this chapter we have extended our simulation of Encounter Gossip to include multiple messages. We have identified a serious performance issue which this introduces and proposed modifications to the algorithm. These modified algorithms have then been simulated to evaluate the performance coverage before and after modification of the algorithms. We have shown how the redundancy affects coverage directly. We have demonstrated that a considerable improvement in coverage can be achieved when using our modified algorithm, particularly when comparing high speeds and low densities. We have

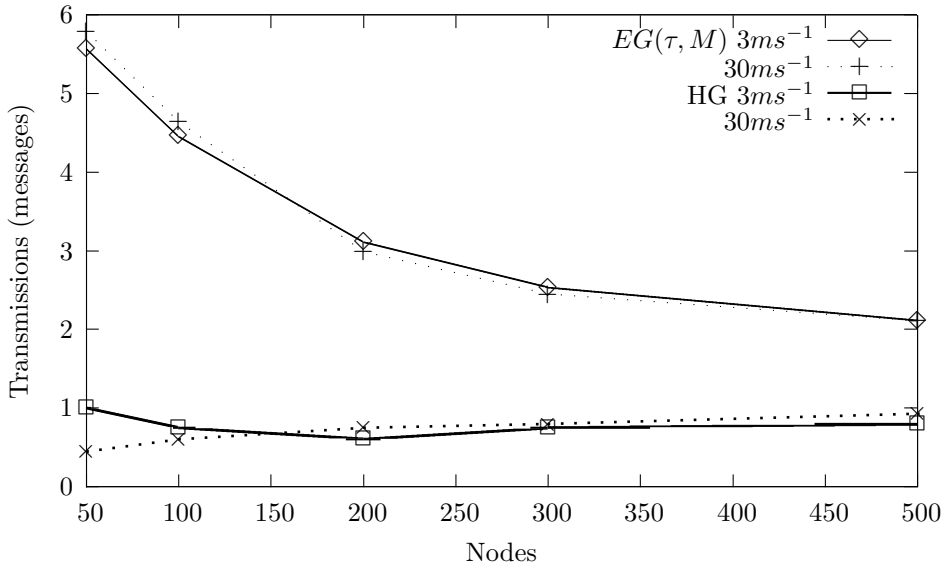

Figure 5.27: Transmissions vs Nodes

investigated a new performance measure to evaluate the possibility of achieving FIFO Order Reliable Broadcast and found that at $Density = 0.5, 3.5, 6.5$,for $\tau = 10, 4, 2$ respectively we can achieve FIFO Order Coverage approaching 1. We have discussed how Encounter Gossip may be made more reliable by introducing a simple message recovery mechanism, that we argue will be effective in retrieving all messages from a random node with high probability. The implementation and investigation into this recovery mechanism is left as a topic for further work. A limited comparison with Hypergossiping is performed and shows that for a range of node densities Encounter Gossip provides higher coverage than Hypergossiping at a moderate cost of higher repeated transmissions.

Chapter 6

Proof of Concept

6.1 Introduction

In previous chapters we have presented Encounter Gossip and its optimisations. We examined its performance using simulation experiments which showed favourable results. Let us now examine the assumptions behind our simulation models.

There are six common assumptions made in simulation of wireless networks, as presented by Kotz et al [52, 53]. We recap on these assumptions here, and show which of these are applicable to our earlier simulation models.

1. **The world is flat**
   Clearly in a real network nodes may be anywhere in 3D space, not merely in a 2D plane. Thus nodes close to one another horizontally may be separated vertically.

2. **Radio transmission is circular**
   Radio range in the real world is far from regular. In fact it is neither circular nor convex and often non contiguous.

3. **All nodes have equal transmission range**
   Even identical nodes will in reality seldom have the same radio range as atmospheric conditions and node placement and orientation may heavily effect the range. Also as batteries discharge radio range decreases.

4. **Communication is Symmetric**
   Naturally in the real world, if transmission ranges are not equal then a $Node_i$ that can receive transmissions from $Node_j$ does not imply that $Node_j$ can receive transmission from $Node_i$.

5. **If a node hears a transmission, it hears it perfectly**
   In the real world transmission success is not a binary process where transmissions are successful when in range and not when out of range. There is no sudden drop off as radio range is reached.

6. **Signal Strength is a simple function of distance**

   The reality is that whilst transmission success reduces with distance under a power law function, the environment can provide obstacles that create obstructions, reflections diffusion and scattering of signals.

Our GloMoSim simulations used assumptions 1,2,3,4 and 6 from this list. The TWO-RAY radio model used in our GloMoSim simulation provides a more realistic propagation scenario, where reflection from the ground is also considered thus assumption 5 is not made. This however still does not include occlusion or reflection by any obstacles so communication is still likely to be more symmetric than in reality.

We now introduce real world problems, describe how these are revealed in our experiment and our attempts at overcoming them.

In the real world, problems exist that these simulation assumptions abstract away. When compared to our GloMoSim Simulations a real world experiment would help address 3 distinct issues:

Fading and transient network links [18]:

Once a connection has been established between two nodes, even without mobility, the ability to transmit between two nodes is not constant. Successful transmission of a packet over a wireless network link is probabilistic at best, as such when a connection between nodes is made (in our case an encounter is experienced) it is not possible to determine the quality or duration of that link. As such, data may be sent along an unstable channel whilst assuming that transmission success rate will be quite high.

Communication grey zones [67]:

A $Node_i$ that is able to receive a transmission from another $Node_j$is not necessarily able to transmit a successful reply. Three factors that contribute to the forming of communication grey zones when using Encounter Gossip are.

i **Small Packet Size**

Hello packets are considerably smaller than data packets and are therefore much more likely to be successfully transmitted. Thus a successful Hello transmission may lead to an encounter when there is no possibility of transmitting data.

ii **Fluctuating Links**

As mentioned above, links may be transient and fade in and out, this is especially true at the boundary of a node's wireless range. Thus a detected encounter may lead to a failed data transmission.

iii **Asymmetric, non circular radio range**

Since radio range is neither circular or identical among nodes, nodes may experience

encounters whilst they are not within transmission range of one another: $Node_i$ may encounter $Node_j$ due to $Node_j$ successful Hello transmission. However if $Node_i$'s range is smaller than $Node_j$'s range $Node_i$ will be unable to transmit data to $Node_j$.

Realistic mobility In our simulations, all nodes move using either Random Waypoint, or Manhattan Grid mobility models. The Random Waypoint model in particular has been heavily criticised for its inability to provide steady state [110, 5]. The Random Waypoint Model is a very unrealistic mobility pattern, especially when contrasted with natural human mobility. Paths chosen by nodes under the Random Waypoint are extremely jagged and at a constant speed between destinations. The Manhattan Grid whilst marginally more realistic, still uses a random choice of path. Humans tend instead to move from place to place on a specified route that is usually one of the shortest, fastest or most interesting available.

Along side these we made three simplifying assumptions to enable us to make our analytical approximation in section 3.3. However in our real world experiment these may be re-examined:

(a) **Each node experiences encounters at intervals which are exponentially distributed with mean ξ.**

In our real world experiment we cannot guarantee this exponential distribution, in fact in section 6.8 we show that in our experiment the distribution is closer to a power law.

(b) **At each encounter, a node meets one other node.**

In our experiment nodes may easily meet multiple nodes at once simply by moving out from behind an obstruction.

(c) **The node encountered is equally likely to be any of the other nodes; that is, the probability that node i will next encounter node j, $j \neq i$, is equal to $1/(n-1)$, regardless of past history.**

In our experiment it is much more likely that nodes located near one another meet more frequently as their movements tend to include similar destinations.

With a real world experiment we can examine how Encounter Gossip performs without these assumptions.

In this chapter we present a proof of concept implementation of Encounter Gossip to show the performance of the protocol under real world conditions. We conduct an experiment in a realistic environment, Claremont Tower, a confined office tower block with several physical obstacles that stand in the way of encounters occurring. We examine some of the physical limitations of the encounter based approach and present simple solutions. We show that Encounter Gossip achieves a

modest coverage even within such a difficult environment. Furthermore we gather data from the experiment to create a real mobility pattern. We use this real mobility pattern within simulation to explore performance across an ideal radio network. These performance measures are a useful measure as to what effect the protocol performs versus the technology.

The design of the implementation is described next in Section 6.2 including language and platform choices. We discuss issues regarding encounter detection in Section 6.3 and present a simple approach to tackling these issues. The experiment itself is documented in Section6.4 and subsequently the results from this are presented in 6.5. We then perform a simulation using newly acquired data and perform further comparison and discussion in Section 6.7. Finally a summary of our findings is presented in Section 6.9.

6.2 Implementation

6.2.1 Language Choice

Platform independent design is important as ideally we would like to be able to test the protocol over a large number of devices; this would then be a heterogeneous environment to allow more devices at our disposal to be included in our experiment. Whilst Encounter Gossip is a routing protocol and should therefore be placed in the network layer in the protocol stack, this would require implementation to be hardware specific. Instead we implement Encounter Gossip using Java at the application layer. This allows our code to run on any device capable of running a Java virtual machine.

6.2.2 Hardware

A total of 18 PDAs and 1 Laptop were used to run the experiment. The laptop was an IBM Thinkpad T30, 2GHz, 512MB RAM with internal Intel wireless mini PCI card (for 802.11b) running ubuntu 6.06. The PDAs were HP iPAQ 5550, 400MHz, 128MB RAM, 48MB ROM, with on board 802.11b running Familiar 8.4 [30] using JamVM[66]. Familiar is a full featured Linux distribution designed for iPAQ and similar devices. JamVM is a small implementation JVM which conforms to JVM specification 2 [62]. At the time of the experiment JamVM did not include the Java 1.5 libraries for concurrency so we used the backport-util-concurrent [56] which are based on Doug Lea's Concurrent Programming in Java [55].

6.3 Encounter Detection

As we mentioned in the introduction to this chapter, radio transmission between nodes is asymmetric. This subjects transmission to a range of problems including fading transient links and communication grey zones. Approaches have been suggested to overcome these problems including artificially limiting the wireless range of devices so that only when a signal is of sufficient strength should it be deemed to be heard. Alternatively transmitting Hello messages at a lower power would mean that when hear the receiver would more likely be within data transmission range. Implementing either of these solutions requires low level access to the device, something that is not easy to achieve in Java, and impossible to achieve platform independently.

Since fading transient links and communication grey zones are most prevalent at the border of transmission range, they are especially problematic when using an encounter based approach to broadcast. This is because they cause encounters during which data may not be successfully transmitted. Along with this the transiency of radio link means that a node may receive Hello messages intermittently. In the worst case this would cause multiple repeated encounters and cause any messages in the node's buffer to be retransmitted redundantly. Worse still these extra encounters would cause message counters $c(m)$ to reach threshold τ too rapidly and thus terminate protocol for those messages early.

We implement a simple solution of requiring H multiple consecutive beacon transmissions to be received to determine whether a node is a neighbour or not. In pairing with this is a neighbourhood timeout T during which at least 1 beacon must be received otherwise the neighbour will timeout and leave the neighbourhood.

Initial tests of the encounter system indeed found that nodes on the boundaries of one another's wireless range would fluctuate in and out of range and thus generate frequent 'superficial' encounters. Devices were noted for having asymmetric ranges, depending on positioning of the devices with respect to distance and interposing obstacles such as walls, floors etc. It was possible for example for $Node_k$ to detect $Node_l$'s presence without reciprocal detection.

This would naturally lead to lower coverage being achieved by this node. Experiments that varied H, the number of consecutive beacons and T, the neighbourhood timeout were conducted. Two nodes were used. $Node_1$ was brought from outside of wireless range towards $Node_2$ at a speed of 0.5m every 10 seconds. Once an encounter was detected by $Node_1$ all movement was stopped and the number of encounters detected was counted on each device for a period of 2 minutes. As H was increased, the nodes needed to be brought closer to one another to achieve an encounter. As T was increased the number of encounters experienced reduced and length of encounter increased. Once parameters H and T were found that reduced the number of encounters to 1, we extended the time that we measured over to 15 minutes. We found that there were several encounters were

observed, caused no doubt by fading transient links. To combat these we increased the T, the neighbourhood timeout until these were removed. We repeated this experiment with several different pair of nodes $Node_3$ and $Node_4$ which gave different required H and T values. By averaging the results we were able to find a working solution that produced minimal boundary fluctuation between pairs of devices with the ability to still detect encounters and departures. The H parameter also had the benefit of reducing the effect of asynchronous wireless ranges and therefore hopefully reducing communication grey zones. We found good values to be beacon interval 500ms, consecutive beacons $H = 5$, neighbourhood timeout $T = 2500ms$.

6.4 Real World Experiment

We perform an experiment to test the performance of Encounter Gossip under difficult, real world conditions. We use an environment that is particularly disruptive to communications by wireless, and in particular 802.11b communication. Many walls in Claremont Tower are extremely thick, most contain steel reinforcement. On top of this, every office has a white board which contains a steel sheet inside covering a large proportion of one wall which makes signal transmission through that wall even less likely. In fact the floors in Claremont Tower appear to allow wireless transmissions to pass easier than the walls. A recent survey concluded that a minimum of 30 and optimally 45 high power base stations would be required to provide coverage of the area [89]. For this experiment we use only 19 devices 18 of these are PDAs with comparatively low power radios. This gives us a low density of nodes in the network, this is discussed further in Section 6.7.2.

6.4.1 Set Up

The experiment was run, in Claremont Tower, which houses the school of computing science at Newcastle University. Volunteers were requested from the school, from the number that responded a selection was made to maximise the spread of the nodes whilst remaining based in the tower complex. Figure 6.1 shows the layout of the Claremont Tower and figure 6.2 illustrates which rooms held users with PDAs on each day of experiment. Table 6.2 can be cross referenced to see which PDA users were based in which room. To prevent user interaction with the device, each PDA was sealed in a cardboard box with a cut out for the arial, and a belt loop fastening. Each user was requested to carry the PDA with them at all times during the experiment.

The experiment was run twice, on consecutive work days to observe different mobility scenarios. On each day the following occurred. Users collected a PDA each from D8.18 at 9:20am and dispersed about their usual daily routine. The office location of each user was recorded against the PDA taken. This is no guarantee that the user would spend their time in or near that room. The PDAs were returned after 1pm when their batteries had been depleted. PDA battery life was measured in earlier

tests to range between 2.5 and 3 hours.

We use our formula from Chapter 3 in Section 3.3 to suggest a good value for our threshold τ. With 19 nodes this suggests an encounter threshold value of $\tau = 7$. Giving consideration to the low density distribution of the nodes and the obstacles they must overcome and low mobility speeds we shall increase this modestly to $\tau = 8$.

Each device was set to send 6 messages. One after every 10 minutes for half an hour. The first at 9:30 and the last at 10:20. This should subsequently allow at least 2 hours to distribute the messages.

A summary of the experiment set up can be seen in table 6.1.

Each PDA would collect statistics on messages sent and received including timings for encountering other nodes. The following details are recorded by each node.

Encounter Node number of device encountered, time of occurrence.

Departure Node number of device departed, time of occurrence.

Originate Message number, time of creation.

Send Message number, message transmission count, time of transmission.

Receive Message number, time of transmission.

This data is used to calculate the following performance figures which are averaged across the set of all devices.

Coverage the average fraction of messages that are received by the end of the experiment.

Redundant Broadcasts The number of a broadcasts made, per message, that do not enlarge the set of nodes that have received the message.

Propagation Time Average time for each message to reach its last destination.

Response Time Average time taken for the protocol to terminate.

Encounter History A representation of a nodes view of the encounters it experienced over the experiment. These are visible in full in Appendix A.

6.5 Experiment Results

The performance measure of interest are again coverage, redundant transmissions, response time and propagation time. In this case however propagation time is the time taken to reach the maximum coverage achieved, since on no occasion was a coverage of 1 achieved.

Equipment	IBM Thinkpad T30	1
	iPAQ 5550	20, 15(Thurs), 14(Fri)
Settings	Messages	6 (per node)
	Message interval	5 Mins
	Beacon Interval	500ms
	Neighbourhood Timeout	2500ms
	Neighbourhood Threshold	5 beacons
	Threshold τ	8

Table 6.1: Table of Experiment Settings

PDA	Thursday	Friday		PDA	Thursday	Friday
(laptop) 0	D8.18	D8.18		10	T7.06	D8.18
1	T7.11	T7.06		11	T7.12	D8.01
2	T8.14	T7.11		12	T7.07	T10.02
3	T8.25	T8.09		13	D8.04	T8.25
4	T8.25	D8.03		14	T7.14	T7.14
5	T7.08	T10.02		15	D8.18	-
6	T7.12a	T8.14				
7	T8.09	T8.18		T/D X.Y	X=Floor	Y=Room
8	T7.05	D8.04			$T = Tower$	$D = Daysh$
9	T9.18	T1002				

Table 6.2: Table of PDA User's room allocation

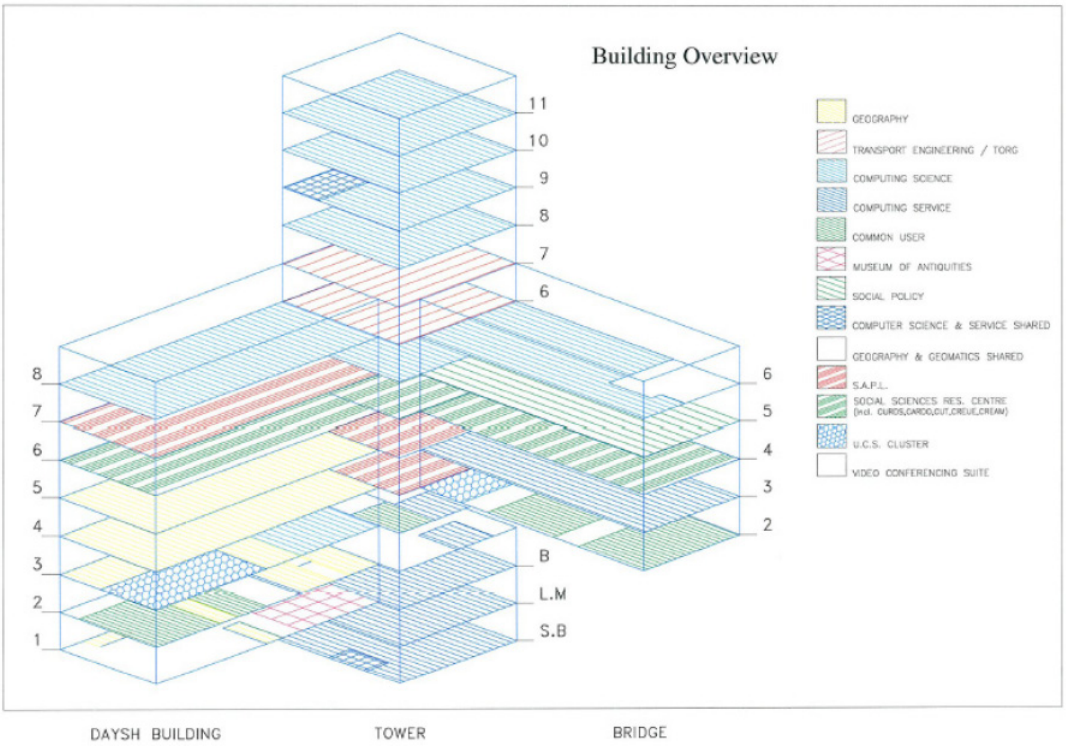

Figure 6.1: Isometric Plan of Claremont Tower Buildings

Figure 6.2: Floor Plans Floor Claremont Tower Buildings

Along side these figures we record the *Encounter History* of each node. The EH of for $Node_k$ comprises of a list of encounter and departure times for each $Node_l (l = 0 \ldots N, l \neq k)$ that have been in the neighbour hood of $Node_k$.

We came across two issues when running the experiment.

1. During the experiment 3 iPAQs failed as their battery was unable to power them for longer than half an hour.

2. On Thursday 2 users and on Friday 3 users, left the experimental area and did not return during the experiment.

These nodes are treated as cooperative non members and as such are excluded from performance figures and statistics.

6.6 Experimental Performance

6.6.1 Coverage

The average coverage observed on Thursday's experiment was 0.64 whilst on Friday 0.59 was achieved. In the following figures we illustrate coverage in two different ways: *Senders Perspective* and *Receivers Perspective*. These two performance measures will give us insight into how each node performed. *Senders Perspective Coverage* is a measure of how well a node propagated its messages to other nodes. *Receivers Perspective Coverage* is a measure of how well a node received messages from other nodes. We will define these in the next section.

The average *Receiver Perspective Coverage* across all nodes is identical to the average *Senders Perspective Coverage* across all nodes. In an ideal network we would find that *Senders Perspective Coverage* and *Receiver Perspective Coverage* would also be the same for each node. However due to the asymmetry of radio ranges we should find that some nodes were more effective at sending data and others more suited to receiving data. Comparing *Receiver Perspective Coverage* and *Senders Perspective Coverage* will allow us to draw conclusions on the asymmetry of radio ranges in this experiment.

6.6.2 Senders Perspective Coverage

Senders Perspective Coverage is a measure of how well a node propagated its messages to other nodes. We use the term SPC to determine the average senders perspective coverage per node, and *spc* to define the average senders perspective coverage per message. They are calculated as follows.

$$n = \text{number of messages a node sends}$$

N = number of Nodes

$$\text{P}(Node_k \text{ succesfully propagates } m_i \text{ to } Node_l) = P_{kil}$$

$$for\ Node_k \text{ sending message } m_i \text{ to } Node_l \qquad P_{kil} = \begin{cases} 1 & if\ Node_l \text{ received } m_i \text{ from } Node_k \\ 0 & otherwise \end{cases} \tag{6.1}$$

$$spc = \text{P}(Node_k \text{ successfully propagates } m_i \text{ to a randomly chosen } Node) = \frac{\sum_{l=1,l\neq k}^{N} P_{kil}}{N-1} \tag{6.2}$$

$$SPC = \text{P}(Node_k \text{ successfully propagates a random message to a random node}) = \frac{\sum_{i=1}^{n} spc}{n} \tag{6.3}$$

Senders Perspective coverage per message is calculated in equation 6.2, averaged over all messages for a single node is equation 6.3. The probability of a randomly chosen node successfully propagating a randomly chosen message to a randomly chosen node is the average SPC across all nodes which is defined in equation 6.4.

$$AverageSPC = \text{P}(\text{A random } Node \text{ successfully propagates a random } m \text{ to a random } Node) = \frac{\sum_{k=1}^{N} SPC}{N} \tag{6.4}$$

6.6.3 Receivers Perspective Coverage

Receivers Perspective Coverage, RPC is a measure of how well a node received messages from other nodes. We define rpc as the receivers perspective coverage per message and RPC as the receivers perspective coverage per node. The equation for rpc is almost identical to that for spc since for a single message node pairing the probability of successfully receiving is equal to the probability of successfully propagating. Thus we use P_{kil} from equation 6.1 in the calculation of rpc. Note that all that has changed is k and l are substituted for one another.

$$rpc = \text{P}(Node_l \text{ successfully receives } m_i \text{ from a randomly chosen } Node) = \frac{\sum_{k=1,k\neq l}^{N} P_{kil}}{N-1} \tag{6.5}$$

$$RPC = \mathrm{P}(Node_k \text{ successfully recieves a random message from a random node}) = \frac{\sum_{i=1}^{n} rpc}{n} \quad (6.6)$$

The average RPC is calculated in Equation 6.6 using P_{kil} from Equation 6.1:

$$AverageRPC = \mathrm{P}(Node_l \text{ receives a random } m \text{ from a random } Node) = \frac{\sum_{l=1}^{N} RPC}{N} \quad (6.7)$$

6.6.4 Results and Analysis

We will first present figures that illustrate *Senders Perspective Coverage*. This performance measure allows us to establish how well encounter propagation performed at distributing a particular $Node$'s messages throughout the network. We consider the messages from a particular source node. The coverage for each message is calculated as the average coverage over all nodes except the source.

We plot a stacked bar chart. The total height of the bar represents the SPC as calculated in equation 6.3. Each bar is broken into 6 sub-bars one for each message, each representing spc calculated in equation 6.2. If a single message has coverage $spc = 1$, the sub-bar representing this will show its maximum height of 1/6, a single message coverage of $spc = 0.5$ will have a bar of height 0.5/6. Thus when all 6 messages have coverage of $spc = 1$, therefore sub-bar height of 1/6 the total height of the bar will be 1 and so the total average coverage $SPC = 1$.

Secondly we will plot the *Receivers Perspective Coverage*, RPC. This performance measure allows us to establish how effective encounter propagation was at delivering all messages to particular nodes. In these figures the bar height represents RPC, and each sub bar represents rpc.

Thursday

We begin by looking at the SPC for Thursday in figure 6.3 The average SPC observed by nodes in Thursday's experiment is 0.62. The division of node and message coverage is as follows. $Node_0$ achieved high coverage on all 6 messages giving $Node_0$ an average coverage of $SPC = 0.97$. $Node_{14}$ achieved only $SPC = 0.03$ only propagating m_0. $Node_4$ has low coverage for all messages except that message 2 has nearly full coverage. The remaining nodes achieved good coverage on most messages, averaging coverage of $SPC \approx 0.69$. It seems only natural that $Node_0$ (the laptop), achieve a higher coverage seeing as it has a transmission power much greater than that of the PDAs.

We now examine the RPC for Thursday's experiment in figure 6.4. $Node_{14}$ now achieves RPC of 0.67 and $Node_4$ has RPC of 0.63. The remaining nodes are distributed around the range $0.4 \ldots 0.8$. Interestingly $Node_0$ now only shows a coverage of $RPC = 0.37$.

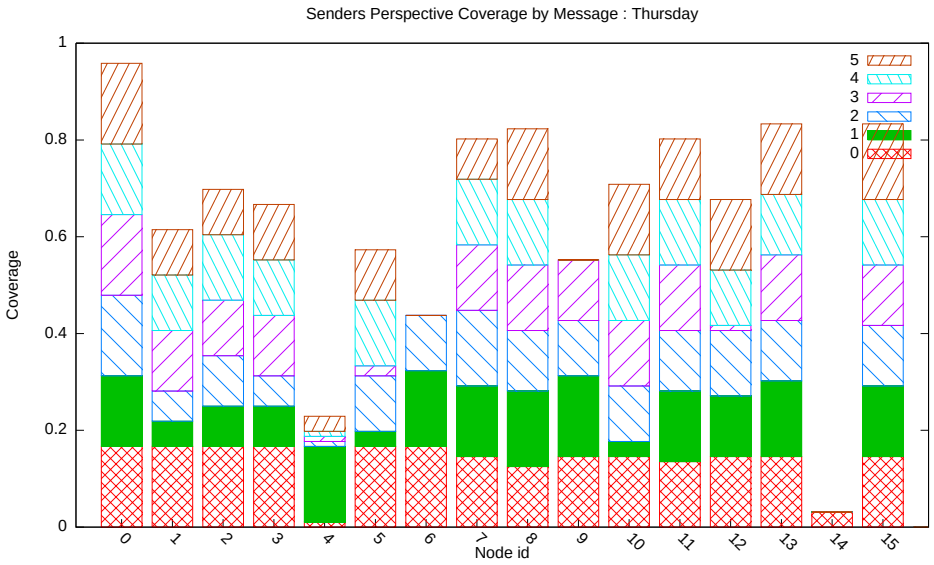

Figure 6.3: Senders Perspective Coverage by node, per message: Thursday

Discussion

Notice the main differences between the two figures. $Node_0$ has considerably lower RPC than SPC, a fall of 0.6 from $SPC = 0.97$ to $RPC = 0.37$. On the other hand, node 14 has improved coverage by 0.64 from $SPC = 0.03$ to $RPC = 0.67$ and $Node_4$ from $SPC = 0.23$ to $RPC = 0.37$. To summarise the variation in coverage. $Nodes$ 1, 4, 9, 10 and 14 show a reasonable increase whilst nodes 0, 2, 6 11 and 13 show a decrease. The remaining nodes 3, 5, 7, 8, 15 only show a small change in coverage. The reduction observed from SPC to RPC observed on $Node_0$ is likely due to with asynchronous wireless range. Recall that $Node_0$is an IBM Thinkpad T30 laptop. This model has large radio antenna along either edge of its screen. From earlier experiments it is obvious that the range of communication by the laptop is much higher than the PDAs. Thus beacons from PDAs will only reach the Laptop once within the laptop's transmission range. This means that a when the laptop receives a beacon from a new $Node$ and therefore detects an encounter, the encountered $Node$ is inside the transmission range of the laptop. The laptop is therefore is highly likely to succeed in transmission of any messages it carries. Conversely, PDAs may receive beacons from the laptop whilst the laptop is outside of their transmission range and thus when they broadcast messages on encountering the laptop they are less likely to reach it. This is an example of a communication grey zone, the PDA is able to detect the laptop, but is unable to transmit data to it. The increasing and decreasing performance coverage of PDA nodes may be related to this.

Although examination of encounter histories for each node has not allowed us to conclusively prove or deny this theory. We propose that the PDAs have similar wireless range to one another

but that the asymmetry in wireless range is exacerbated by the environmental conditions thus creating transient and fading network links as nodes move past obstacles that generate reflections or block signals altogether. Placing a wireless transmitter/receiver next to a large metal surface like a whiteboard may effectively block out radio communication from the other side of the white board but also increase the reception from signals on the same side as the device. The encounter histories of all nodes can be seen in appendix A.

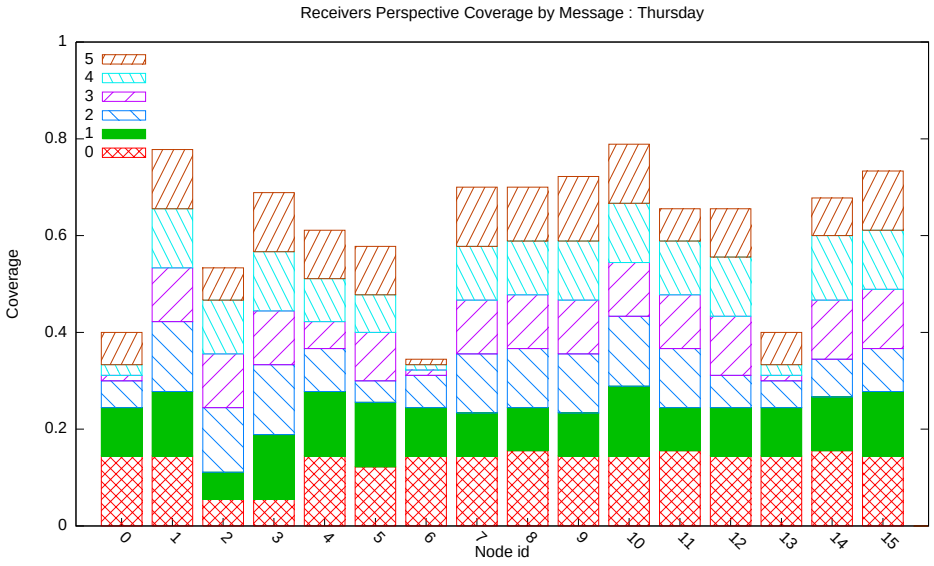

Figure 6.4: *Receivers Perspective Coverage*: Thursday

Friday

In Friday's Experiment *Senders Perspective Coverage* in figure 6.5 is a little lower. An average SPC of 0.59. $Node_0$ this time only achieves a coverage of $SPC = 0.69$. The highest coverage of $SPC = 0.83$ is achieved by Node 5. All other nodes show a coverage range between $0.3 \ldots 0.7$.

The *Receivers Perspective Coverage* for Friday's experiment is graphed in figure 6.6. Node 0 shows a similar reduction, RPC is 0.42. Most other nodes again show a coverage range of $RPC = 0.3 \ldots 0.6$.

Discussion

Again we examine the major differences between sending and receiving coverage graphs. Fridays differences show the expected decrease in performance from SPC to RPC by $Node_0$ (the laptop) a fall from $SPC = 0.69$ to $RPC = 0.42$. We see the high performing $Node_5$'s coverage fall from $SPC = 0.83$ to $RPC = 0.61$. Summarising these performances again yields three groups of devices. Those that increase: *Nodes* 1, 4, 9, 12 and 14 and a set that decrease *Nodes* 0, 2, 5, 6, 7, and 13.

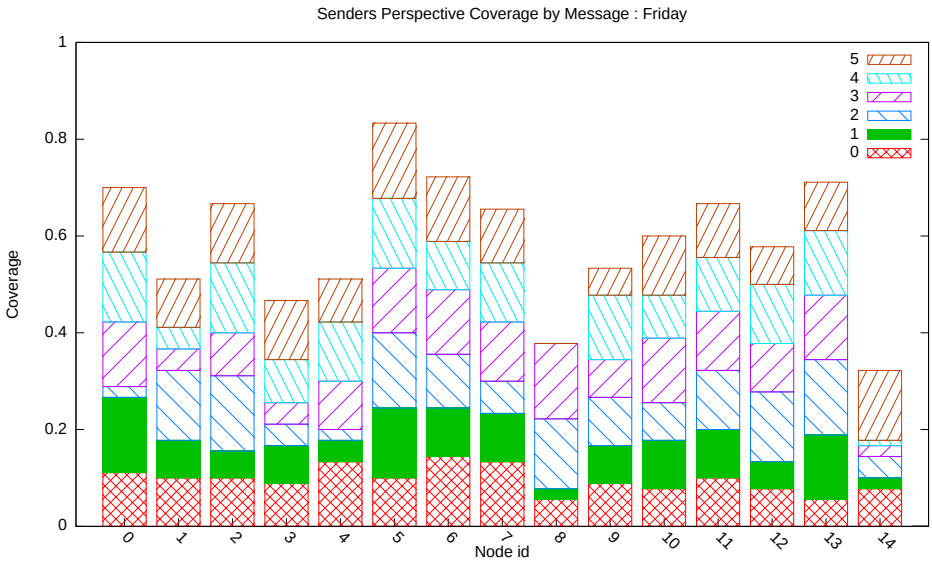

Figure 6.5: *Senders Perspective Coverage*: Friday

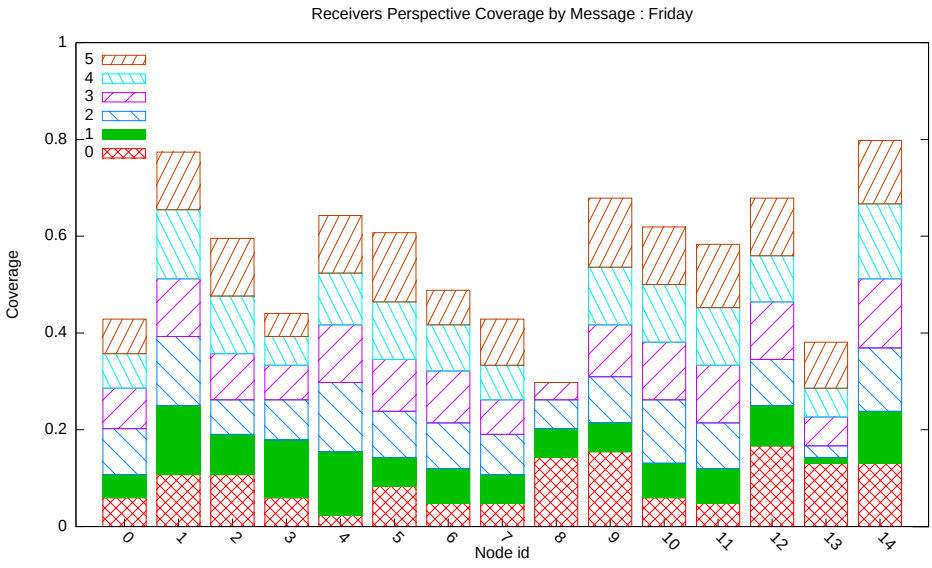

Figure 6.6: *Receivers Perspective Coverage*: Friday

Those that are relatively stable 3, 6, 8, 10, 11.

So here we examine the three distinct groups of results, those that increase dramatically in coverage SPC to RPC, those that show small differences and tose that show large reduction of coverage from SPC to RPC. Examining the encounter histories which are in Appendix A we see a pattern emerge. Those that have a large increase from SPC to RPC tend to have frequent short encounters as exemplified by $Node_{14}$ on Thursday as shown in figure 6.7. Those that show little change have both long and short encounters; Figure 6.8 shows $Node_5$ on Thursday with such a pattern. Where we see the large decrease from SPC to RPC we see much longer encounters taking place, such is the case with $Nodes$ 0 and 13 on both Thursday and Friday. Here we show three Encounter History of $Node_0$ on Thursday in figure 6.9 which is representative of the category.

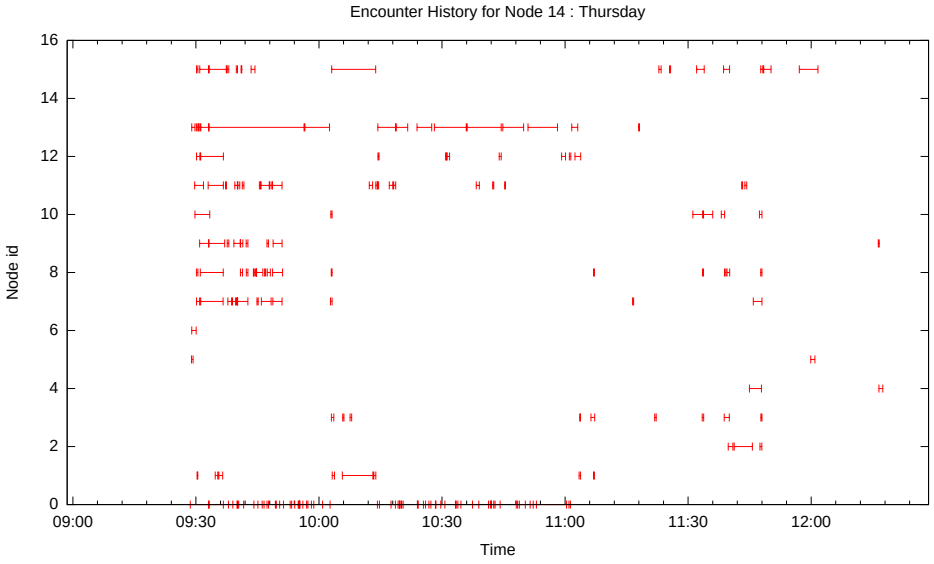

Figure 6.7: Message History for node 14, (Thursday)

The difference between the coverage of Thursday and Friday's experiments is only 0.05. It seems likely that this is caused by natural variation in mobility of users. Examination of the locations in which nodes were based shows that on Thursday we have a single node on floor 9. On Friday we have no nodes on floor 9. This would given more separation and thus fewer encounters between nodes on floor 10 and 8, and could cause reduced performance.

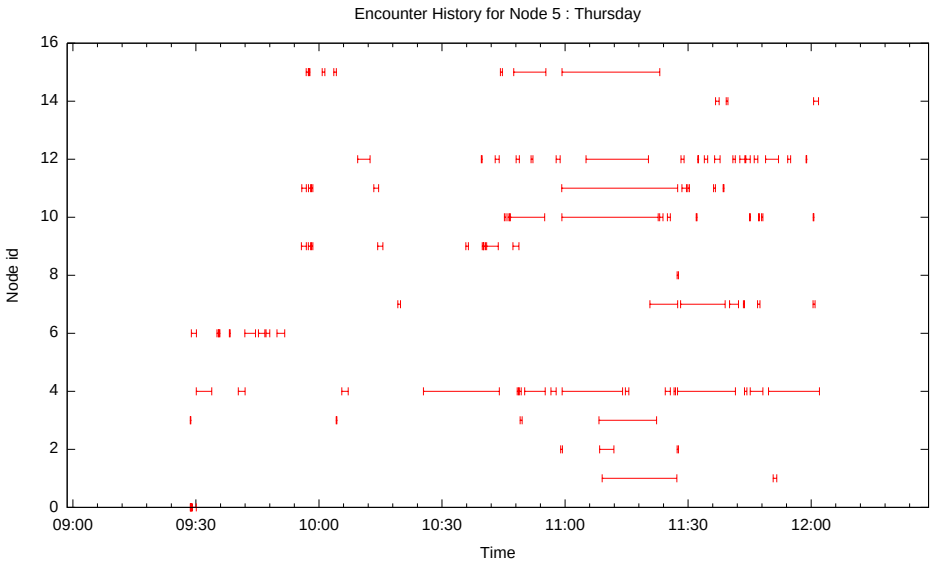

Figure 6.8: Message History for node 5, (Thursday)

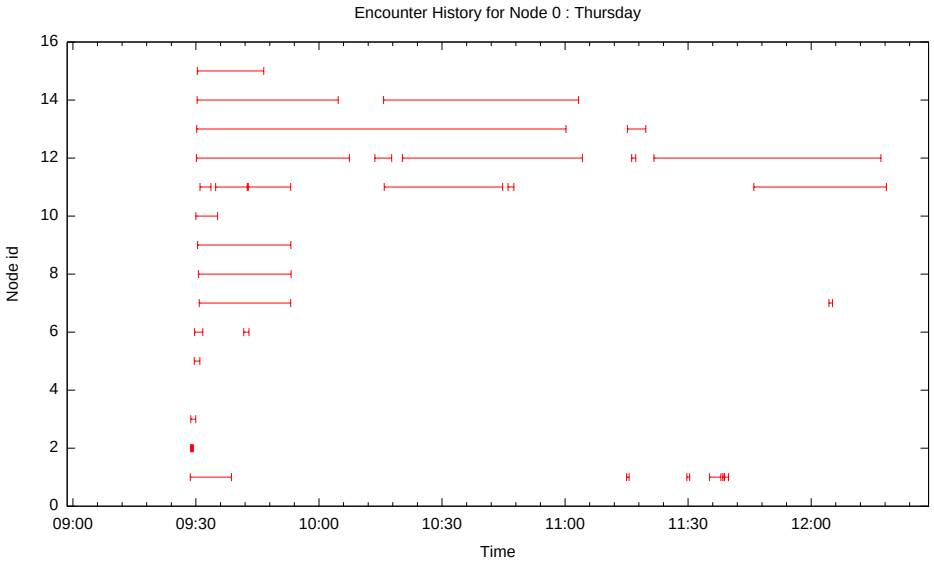

Figure 6.9: Message History for node 0, Thursday

6.7 Simulated Performance

Using Encounter History for a Mobility Pattern

In a simulation, nodes use mobility patterns to generate a series of time linked coordinates. As simulation time progresses the simulator steps each node through its coordinates. Each time a node makes a transmission, the simulator decides if communication is possible between any pair of nodes.

To do this it compares the coordinates of communicating nodes using a radio model to determine whether or not two nodes are able to communicate. In our next set of simulations we can dispense with the radio model as we have a set of Encounter Histories which tell us exactly when our nodes were able to communicate with one another. As simulation time increases the encounter and departure events provide a snapshot of which nodes are able to communicate with one another. In order to capture figures of theoretical maximum performance, we remove the effects of radio asymmetry from the simulation. To do this we make each encounter symmetrical. For example compare encounter histories for *Nodes* 0 and 14 on Thursday, figures 6.9 and 6.7. We can see that $Node_0$ observes two long lived encounters with $Node_{14}$, whereas $Node_{14}$ observes many very short encounters with $Node_0$ during the same periods. For our simulation we use the union of these encounter histories, so now both $Node_0$ and $Node_{14}$ would see two long encounters. Compare *Nodes* 5 and 14, $Node_5$ encounters $Node_{14}$ for three brief occasions toward the end of the experiment. $Node_{14}$ on the other hand encounters $Node_5$ briefly at the start and toward the end of the experiment. To take the union of these encounters both nodes would now see the same set of encounters: one at the start and three towards the end.

6.7.1 The Simulation

To further investigate the performance of *Encounter Gossip* we can use our newly acquired *Encounter History* as replacement mobility pattern for each *Node* in a simulation. We set up an ideal world simulation to ascertain what the maximum coverage that could be achieved under this mobility scenario with a range of different encounter thresholds. In this simulation radio ranges are not used, the *Encounter History* data from the experiment is used as a direct indication of when nodes came into contact with one another. Removing the beacon and encounter detection and replacing message transmission with immediate passing of messages. *Note: An encounter observed by only one Node in the* Encounter History *will be observed by* both *Nodes in the simulation.* During an encounter, communication is assumed to be collision free in both directions. This in effect simulates an ideal MAC and PHY layer in which a nodes neighbours are known to it, and transmission is instantaneous and error free. As such we are effectively using a real world mobility model.

Figures 6.10-6.17 show the performance of the protocol under ideal simulation using the mobility data obtained from the experiment.

6.7.1.1 Coverage

Figures 6.10-6.11 show the coverage achieved under simulation. Both graphs show a now familiar curve that increases rapidly as the number of encounters is increased from 0 to 4 and flattens to almost horizontal by the time 8 encounters are reached. Table 6.3 allows us to compare performance figures at a glance. Thursday shows a coverage of 0.92 at 8 encounters, where Friday shows 0.88 a

Performance Measure		Thursday	Friday
Coverage	Observed	0.64	0.59
	Simulated	0.92	0.88
Redundant Broadcasts	Observed	4.73	4.95
	Simulated	6.79	6.34
Response Time (s)	Observed	19340	6903
	Simulated	8335	3385
Propagation Time (s)	Observed	15974	4136
	Simulated	4970	1283

Table 6.3: Table of performance measures at $\tau = 8$, comparing Observed and Simulated Values

difference of 0.4 which is comparable to that shown in our experimental performance: Thursday was 0.64, Friday 0.59 a difference of 0.05.

Comparing simulated coverage to the real world experiment coverage we see an increase in the simulation of 0.28 for Thursday and 0.28 for Friday over the observed coverage. This indicates the effect of the assumptions made in the simulation, error free, symmetric transmission of messages. It also allows us to see the maximum coverage gain that could be achieved in the current scenarios if an ideal MAC, PHY and encounter detection were developed and used in the real world.

It is also worth noting that coverage only reaches 0.94 on Thursday and 0.93 on Friday even at 16 encounters. This suggests that in this scenario it is very unlikely that coverage could ever reach 1 even with infinite encounters as the network is never totally connected to deliver or receive certain messages.

In this environment it is difficult to calculate an estimate for the simulation area as it is irregular 3D and theoretically infinite, as users were not physically restricted to the buildings. It is also impossible to calculate a wireless coverage for our nodes since the location of the node will severely alter the range and shape of coverage achievable. We can however compare these results with those from our initial GloMoSim simulations in Section 3.4 on page 36. We saw that with a density of 0.5 a coverage of 0.99 was achieved at 8 encounters and that coverage of 1 was achieved by 12. This suggests that the density of the real world experimental network was considerably lower than this. We propose that the mobility of the nodes is the key factor here; some nodes may be isolated from others for a majority of the experiment. As we noted earlier when discussing the coverage losses shown between Thursday and Friday, missing nodes on floor 9 would have helped to cause longer partitions in the network. In fact it is clear that there are distinct groupings of nodes that would likely encounter one another frequently. Those close to one another on any particular floor. And those close to one another perpendicularly. The separation of the Daysh Building 8th floor from the Claremont Tower by one floor and a considerable horizontal span containing external walls would have helped to create another partition.

The simulated performance of our algorithm is much higher than that of our actual experiment

using the same encounter data. This illustrates the effects of several obstacles:

1. Collisions and other radio effects cause large message loss.

2. Encounter detection scheme and knowledge of neighbours is not ideal (ie. Communication Grey Zones not fully compensated for).

We feel it is less likely that collision is the major cause of such drops in coverage. Assuming all 21 nodes were present and correct, each node beacons infrequently, every 500ms and with 250ms Jitter. Only 6 messages are initiated by each od N nodes, 1 every 10 minutes. At a maximum $6N$ messages can be in the network, with a maximum of $N^*\tau$ transmissions of each message. Thus the maximum number of messages in transmit over the entire experiment is $6 * N^2 * \tau$. That is only 126 messages for a sum total of 21168 transmissions as a maximum. Over the period of the experiment which is approximately 3 hours that is only 2.3 messages per second, hardly enough to saturate the network. Although transmissions are restricted mainly to encounters and so transmission would be bursty.

It is more likely that the encounter detection played a larger part. It is probable that the radio coverage of each node was not symmetrical due to differences in power at each node and interference and reflections of transmission caused by the physical environment. It would be worth exploring the encounter detection mechanism further and should be investigated in future work. In fact design of MAC and PHY layers are separate research topics in their own right.

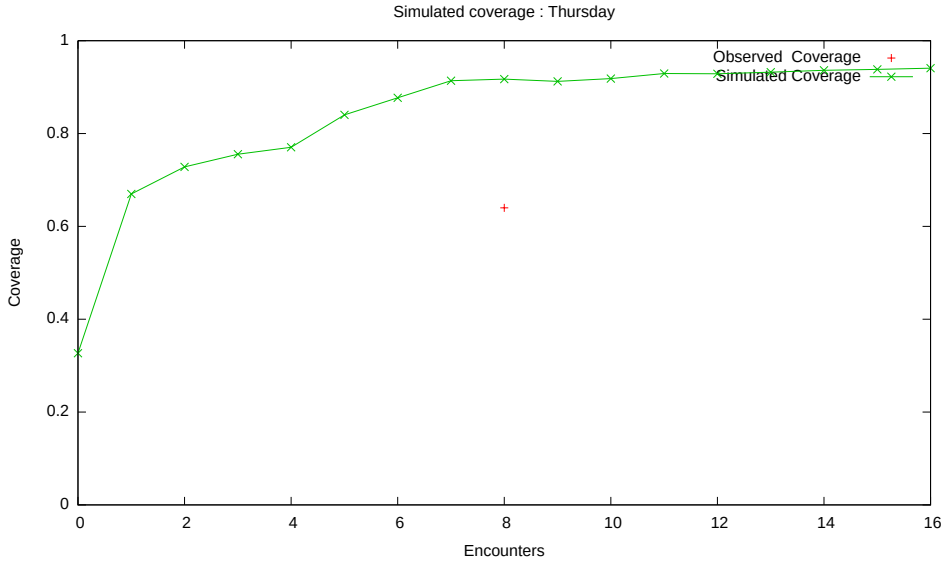

Figure 6.10: Simulated Coverage: Thursday

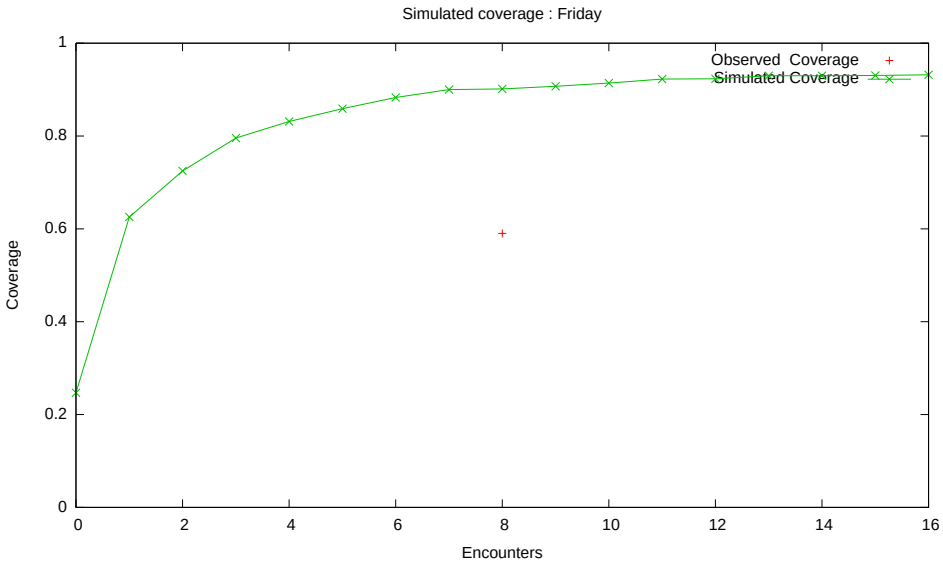

Figure 6.11: Simulated Coverage: Friday

6.7.2 Transmission Power, Network Density and Encounter Frequency

In this section we will examine the scenario to show how challenging the environment is when compared with our earlier simulations. In earlier GloMoSim simulations in Chapters 3 . . . 5 we have used network density as a parameter to vary through several experiments. It was found that a higher density produced higher coverage, even with lower encounter threshold. As we have mentioned earlier, it is impractical to calculate the density of our real network. Instead, we can observe the average number of neighbours a given device has over the period of the experiment. For comparison, we use an earlier vanilla Encounter Gossip GloMoSim scenario. We test speeds of $1ms{-}1$ and $2ms^{-1}$ which are reasonable for human mobility speeds. To calculate the average number of neighbours for $Node_k$ we sum the duration of all encounters between $Node_k$ and $Nodes_{j...n}$ $(i \neq j)$ and divide this by the duration of the simulation. To get an average figure for all nodes we sum the resultant value for each node and divide by total number of nodes.

Figure 6.12 plots network density against the number of neighbours. There is a linear relationship between density and number of neighbours and that the speed of movement alters the gradient, a higher speed results in a higher gradient. It follows then that higher average neighbourhood sizes are proportional to higher densities and should equally produce higher coverage. Also a higher speed causes a higher neighbourhood size.

It is impossible for us to calculate the actual mobility speeds of the nodes in our real world experiment, but since our protocol relies on encounters occurring at a reasonable rate, we can calculate the encounter frequency from our encounter histories: Averaging the sum of all encounter

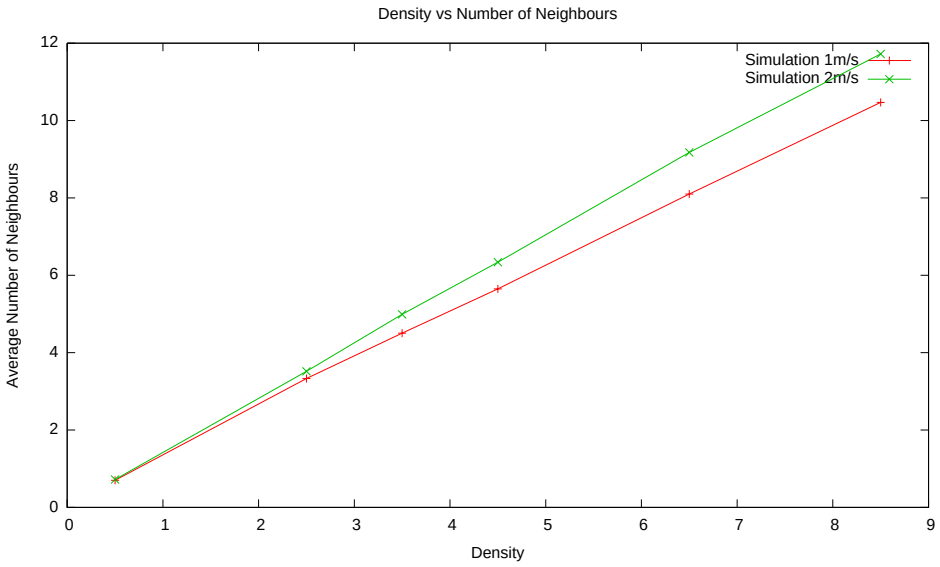

Figure 6.12: Density vs number of Neighbours

intervals over the whole experiment time. By calculating this performance metric for both real world and the simulation above we can compare our real scenario against simulation parameters. Figure 6.13 shows the relationship between Encounter Frequency and average number of neighbours. Again this plots GloMoSim data for speeds $1ms{-}1$ and $2ms^{-1}$ along with these we also plot the points for our observations from our real world experiments. We see that this time the frequency of encounters increases faster than linear, as the average number of neighbours increases. The encounter frequency achieved by our real world experiments is lower than both simulation results at these average neighbours. This is indeed a challenging environment.

6.7.2.1 Redundant Broadcasts

The number of redundant transmissions made per node per message can be seen in Figures 6.14-6.15. Each shows a linear increase in redundancy, Thursday reaches 7.1 redundant transmissions at 8 encounters and Friday reaches 5.2. This is similar to earlier GloMoSim results shown in 4.5 page 53. We also plot the observed experiment results here for comparison. They are 4.56 and 4.72 respectively, these values are lower than the simulated values, but this is to be expected as coverage was also lower thus fewer messages were being transmitted by fewer nodes. Redundancy schemes proposed in 4.5 would likely have little effect here as the density is so low. It would be prudent to examine other ways for reducing redundancy.

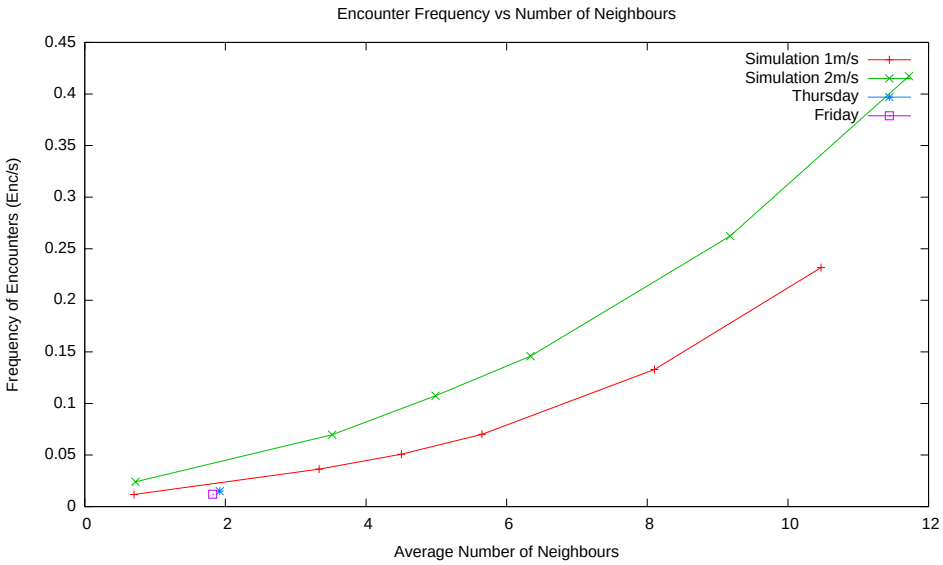

Figure 6.13: Encounter Frequency vs Number of Neighbours

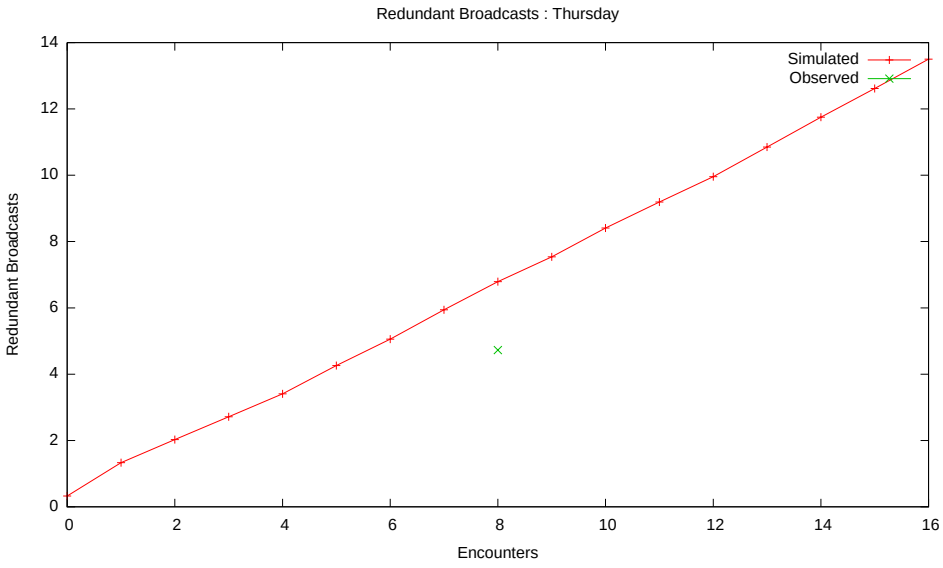

Figure 6.14: Redundant Broadcasts per Node vs Encounters: Thursday

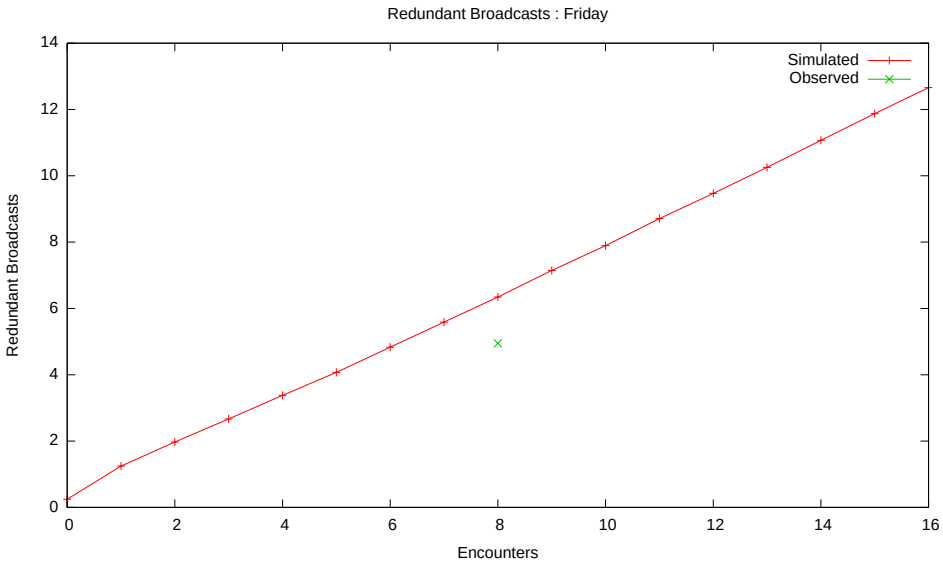

Figure 6.15: Redundant Broadcasts per Node vs Encounters: Friday

6.7.2.2 Timing

The timing graphs are depicted in figures 6.16 and 6.17. Propagation time being the time taken to reach maximum propagation of the message. Response time is the time taken to stop transmitting the message. These both show that the time taken to achieve the given coverage is the approximately same after the increase from 0-2 encounters. In Thursday's case propagation time is $4970 seconds$ (1hr, 23mins). On Friday this drops to $1283 seconds$ (18mins), note that this corresponds to the lower coverage of Friday's experiment, a coverage of 0.88 rather than Thursday's 0.98. The response time in each case increase approximately in linear fashion after the initial increase. Thursday reaches 8335seconds (2 hr 19 mins) at 8 encounters, where Friday reaches 3385 seconds (56mins). The experimental results are also plotted here, they are higher than the simulated results again due to asymmetric message transmission.

We can compare these figures to the timing figures in Section 3.4. Those show that with a density of 0.5 and mobility of $20 ms^{-1}$ that at 8 encounters propagation and response times are between $350 seconds$ and $400 seconds$. This is lower than those observed in our experiment due to a neighbourhood size, speed and therefore lower density.

Another very interesting observation is that the time to reach maximum propagation remains virtually constant in both cases. It would be expected that this time fall as you increase τ. One possible explanation for this is that the encounters may be more common between the same pairs of nodes. That it is more likely that a node encounter another node that it has already encountered than one at random. If this were the case then counter c would reach the encounter threshold τ early,

transmitting the message to fewer new nodes than it would if each were a random node.

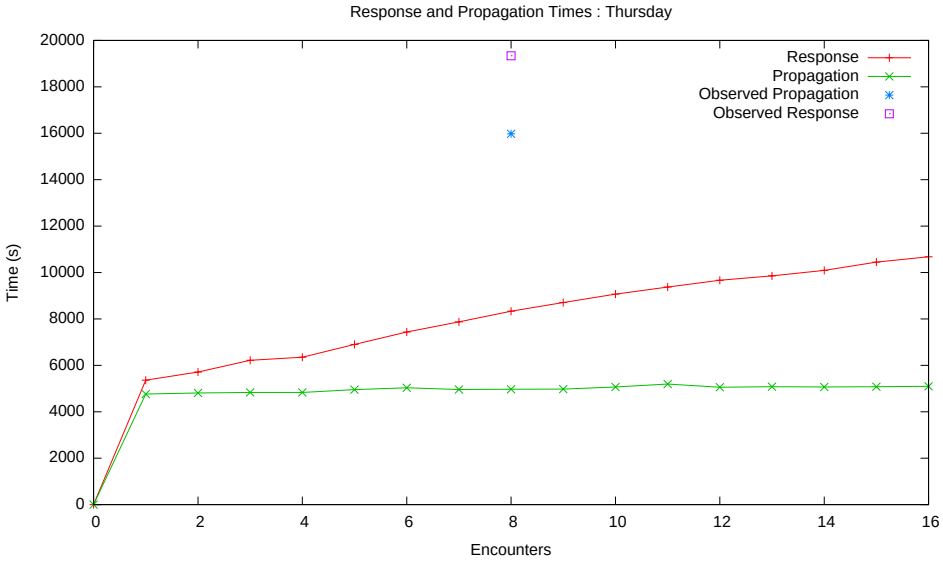

Figure 6.16: Timing graphs: Thursday

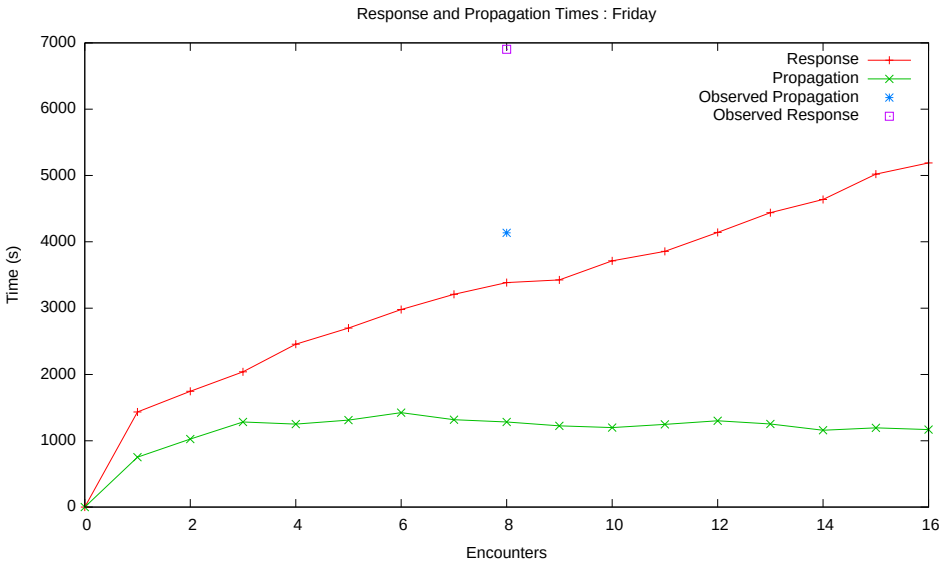

Figure 6.17: Timing Graphs: Friday

6.8 Real World Mobility

The generation of mobility models for simulation of MANETs has been researched for many years, Camp et al provide a summary of many of these in [10]. The goal of mobility models developed,

has typically been to evaluate the performance of routing protocols. We have investigated the performance of EG with Random Way Point (RWP) and Manhattan Grid (MG) models in earlier chapters. It is clear however from the simulations in section 6.7.1 that EG performs differently with this instance of real world mobility.

A lack of realism is inherent in randomly generated patterns such as RWP and MG. To introduce realism, different aspects of mobility have been investigated. A new mobility model using buildings as obstacles to mobility and radio transmission is proposed in [40]: using Voronoi graphs they derive paths upon which nodes may move. In [15, 39, 2, 68, 36], research into the inter-encounter distribution of real world mobility traces and the meaning of this for real world opportunistic networking is carried out. A common property of many mobility models found in the literature is that inter-encounter interval is distributed exponentially. We use this to make our analytical approximation for τ tractable, in Section 3.3.

In [15] four mobility traces are analysed: two from self run experiments, one from Dartmouth [36] and one from UC San Diego [68]. It is argued that for all 4 data sets, the tail distribution of inter encounter time is distributed with a power law of low coefficient; 0.6 or 0.28 data set dependant. The work goes on to examine the expected transmission delay an algorithm should experience. If distribution of encounter intervals is distributed with a power law with coefficient cf:

- $cf < 1$ no stateless forwarding algorithm can achieve transmission delay with finite expectation.
- $1 < cf < 2$ stateless algorithms that propagate sufficient copies of the message may succeed
- $cf > 2$ any stateless algorithm converges.

Taking these findings into account and also motivated by social networking theories, new mobility patterns have been developed [69, 20]. Evaluating our protocols against these new mobility models would be interesting future work.

We plot the inter encounter intervals experienced by nodes in our experiments in figures 6.18. . .6.19. In both experiments we find that the tail of the distribution of inter encounter intervals is heavier than that of the calculated exponential distribution. An approximate power law distribution with $cf = 2$ provides a much better fit. We have shown therefore that EG provides good coverage even under a real world experiment with a power law distribution. In the light of this distribution it may be worth evaluating encounter gossip by simulation using a mobility model that generates encounters with a power law distribution.

Chaintreau also plots the encounter durations experienced in his experiment sets and we provide ours here for completeness in figure 6.20.

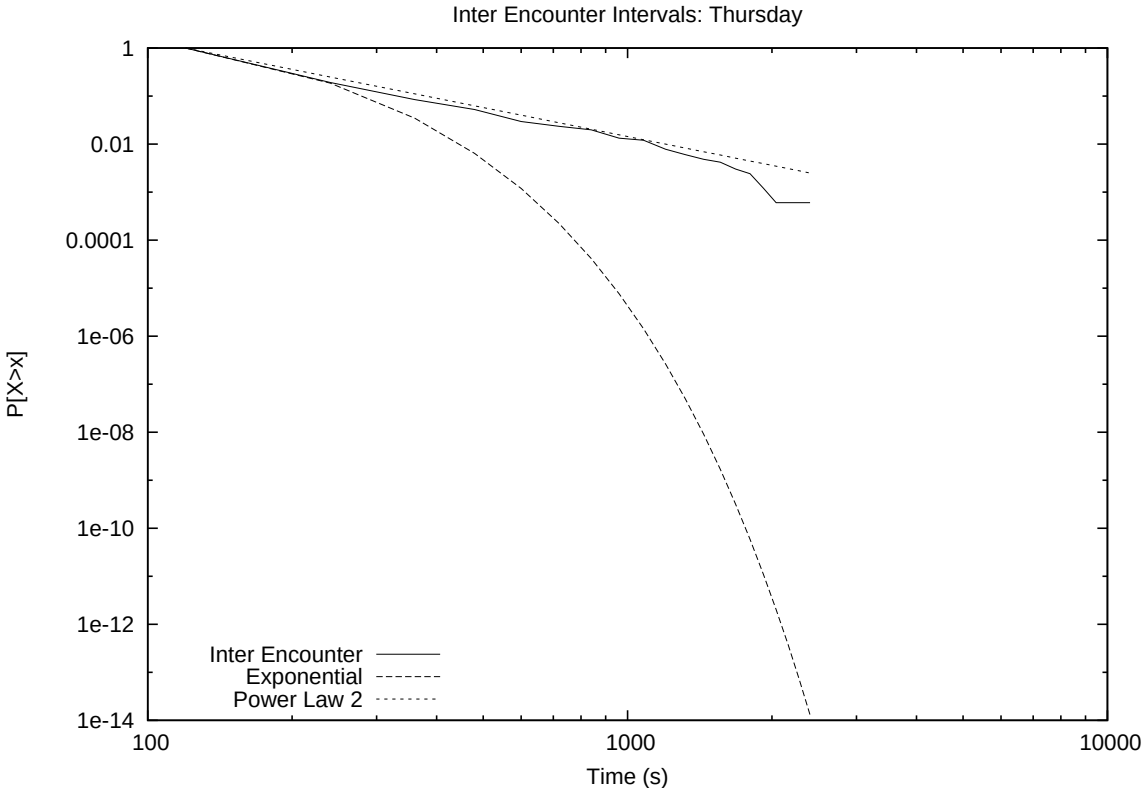

Figure 6.18: Inter-encounter intervals: Thursday

6.9 Summary and further work

We have developed a Java implementation of Encounter Gossip and performed two experiments within a realistic environment, the Claremont Tower Complex, a group of buildings with poor wireless transmission qualities. Results have shown that the protocol works under this difficult scenario. Whilst performance was not as high as we might hope there two areas to investigate are identified as to the cause of the sub ideal performance: message collision and encounter detection system. We have used *Encounter History* data gathered from the experiment to simulate the protocol under a real mobility pattern (identical to that of the actual experiments) and review the performance that could be achieved with perfect message transmission. These simulations have shown similar performance to that of earlier GloMoSim runs although the maximum coverage achieved was never above 0.92. It has been established that the protocol generates a similar number of redundant transmissions as found in simulation and so implementation of some of the optimisation techniques discussed in Chapter 4 would be necessary.

It would make an interesting extension to perform an experiment to discover how isolated groups of nodes were and to discover what partitions actually exist over the network.

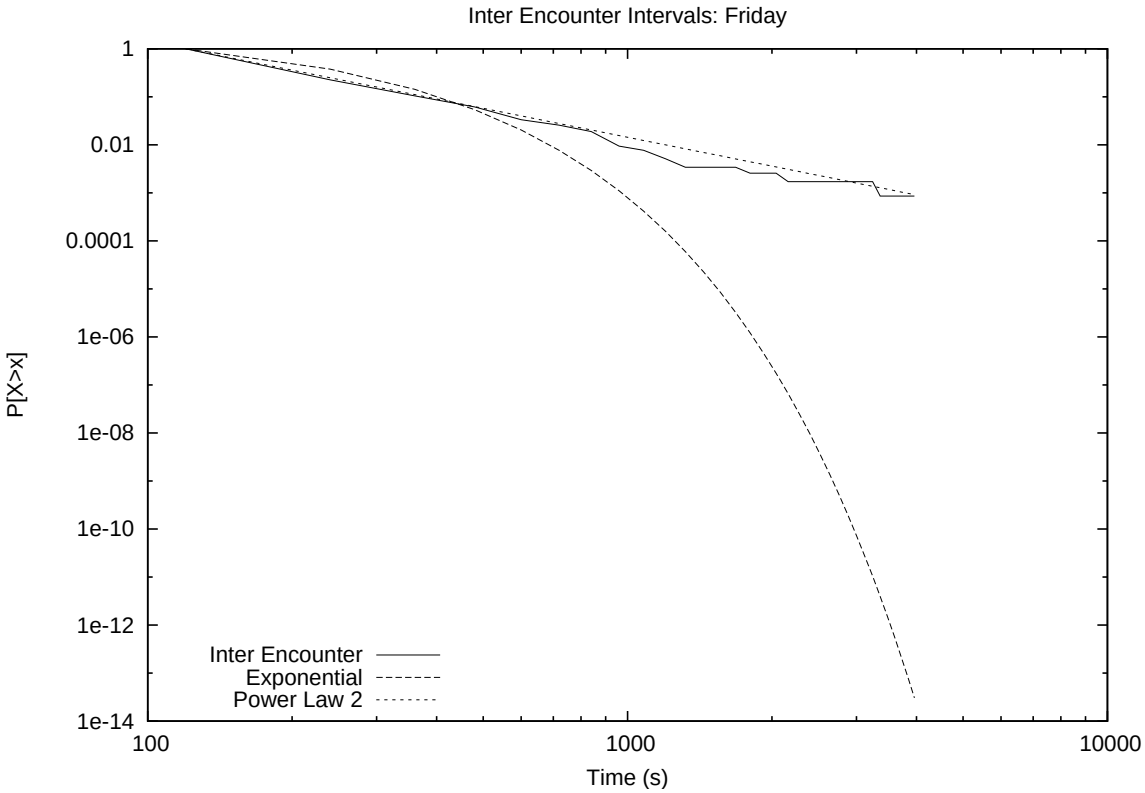

Figure 6.19: Inter-encounter intervals: Friday

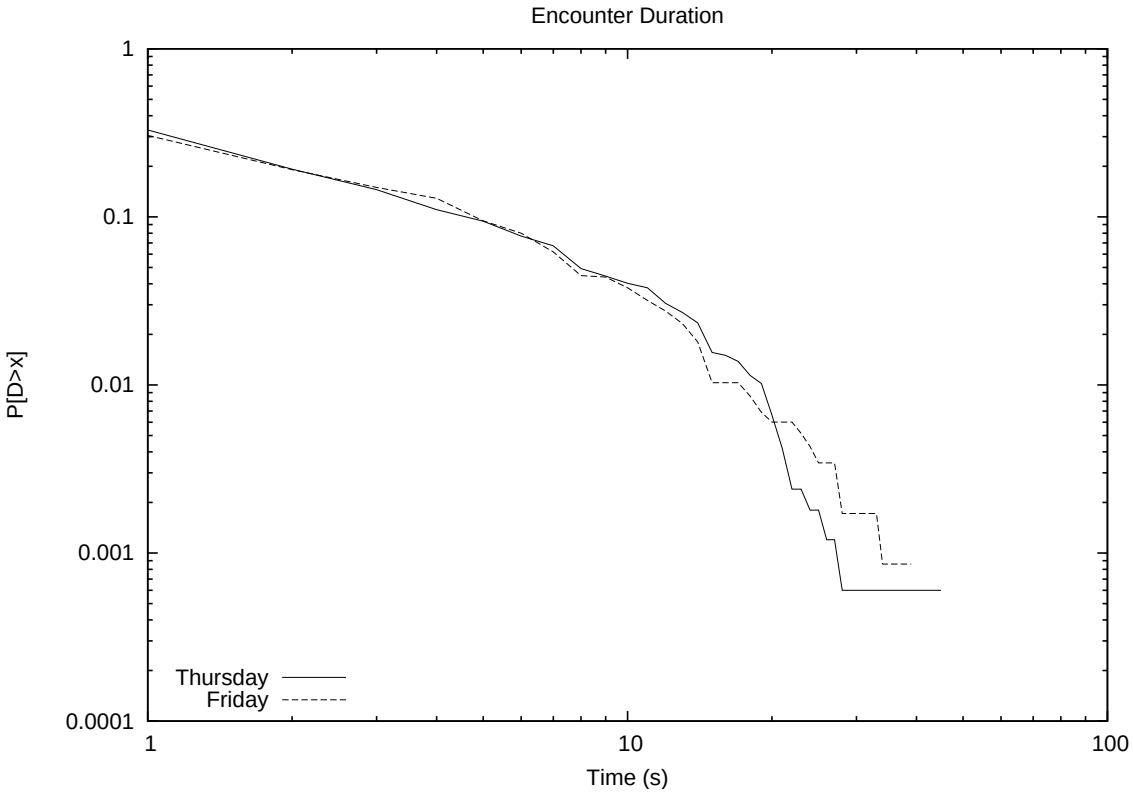

Figure 6.20: Encounter Durations

Chapter 7

Conclusions

During the course of this thesis we examined the existing Broadcast protocols for MANET, proposed our own solution and tested it through simulation and experiments. Whilst there are other protocols that attempt to provide good coverage in the *Broadcast-in-time* scenario, there are areas where they all fall down. Both AF (Adaptive Flooding) and HG(Hyper Gossiping) [98], [51] perform poorly if the lifetime of a broadcast not adequately set since they cannot predict the *Ideal Minimum Propagation Time* (IMPT): if the lifetime is too low then coverage will be poor; if the lifetime is too high for the mobility of the network then many redundant transmissions will be made. We introduced Encounter Gossip (EG), a protocol based around encounter threshold propagation. This protocol provides high coverage in *Broadcast-in-space* scenarios no matter the speed of mobility. As density increases and we move into *Broadcast-in-time* scenarios EG is equally successful.

The main contributions of this thesis can be summarised as follows:

1. Introduction of Encounter Gossip, a family of encounter propagation protocols (Chapter 3).
2. Mobility and density independent estimate for the value of τ that is expected to achieve high coverage (Chapter 3, equation (3.6)).
3. Improvements to the protocol aimed at reducing the number of redundant transmissions (Chapter 4). In particular, a combination of RAD and α-*reduction* is both simple and efficient, and hence is worth implementing.
4. An efficient generalisation to multiple propagations in parallel (Chapter 5).
5. An implementation and validation of the protocol in a real-world environment (Chapter 6).
6. Quantitative performance results obtained by simulation (Sections 3.4, 4.5, 5.6, 6.7).

7.1 Encounter Gossip

We have introduced EG, the τ-propagation family of Broadcast protocols. Moreover we provide a Mobility-independent estimate for the value of τ that achieves high coverage, arbitrarily close to 1. We have shown using GloMoSim simulation under both Manhattan Grid and Random Waypoint mobility at densities ranging from 0.5 to 6.5 with speeds of $2ms^{-1} \ldots 100ms^{-1}$, EG can achieve a coverage approaching 1 with an appropriate τ. Whilst the Manhattan Grid model provides more of a challenge we have shown that increasing τ is sufficient to overcome this challenge. Considering our analytical predictions concerning τ. For 64 nodes, the encounter threshold given by equation 3.6 is $\tau = 10$, and the figures indicate that they do, indeed, achieve coverage close to 1. In fact, when the density is high, the threshold provided by equation 3.6 is rather conservative.

7.2 Optimisation

We have optimised EG to reduce the cost of high coverage. Attempting to reduce the number of redundant transmissions used in the propagation of a message is clearly a worthwhile effort. Using RAD and other techniques from the literature we are able to achieve a significant reduction in redundancy. Our key achievement in this, is the introduction of our *α-reduction* policy. It is by far the most effective of those tested and when combined with RAD achieves further reduction.

A real world network may have areas of high density and areas of low density. It would be ideal therefore to have a single τ that could be used in all densities. To achieve high coverage in low density networks a large τ is required which in turn produces more redundant transmissions. Our experiments have shown that, at low node density, the average number of redundant transmissions per node can be reduced by about 30% in the Random Waypoint mobility model, and by twice that amount in the Manhattan Grid model. Experiments with this same high τ at high density show that by using RAD plus *α-reduction* policies we can achieve a 75% reduction in redundant transmissions, down to almost the same level of redundancy that would be created with a much lower τ whilst maintaining the high level of coverage.

An interesting observation is that, at the threshold levels that are necessary to achieve high coverage, the simple policies — RAD, *α-reduction* and Broadcast Count reduction — perform no worse than the ones employing caches and message lists (EH and PH). In fact, the *α-reduction* policy can be used with a high threshold over a range of densities, and still produce large savings in redundant transmissions.

7.3 Multiple Messages

In examining the operation of EG using multiple Broadcasts we have identified two serious performance issues: Encounter Redundancy and Departure Redundancy. We have introduced modifications to the algorithm to overcome these issues. We have demonstrated that considerable improvements in coverage can be achieved, particularly when comparing high speeds and low densities. We have also examined a new performance measure to evaluate the possibility of achieving FIFO Order Reliable Broadcast without recovery mechanism and found that at $Density = 0.5, 3.5, 6.5$ for $\tau = 10, 4, 2$ respectively we can achieve FIFO Order Coverage approaching 1.

7.4 Encounter Gossip in the real world

The examination of the performance of any Broadcast protocol would not be complete without some testing in a real environment. We have therefore developed and tested a Java implementation of Encounter Gossip and performed two experiments within a novel realistic environment, the Claremont Tower Complex, a university office block complex with poor wireless transmission qualities. Results have shown that the protocol achieves good coverage even in this difficult scenario. Whilst performance was not as high as we might hope there two areas to investigate are identified as to the cause of the sub ideal performance: message collision and encounter detection system. Using *Encounter History* data gathered from the experiment we were also able to simulate the protocol under a real mobility pattern (identical to that of the actual experiments) and compare the maximum performance that could be achieved with that of the real world experiments. These simulations have shown similar performance to that of earlier GloMoSim runs although the maximum coverage achieved was never above 0.89. This itself is an indicator that the environment was particularly hostile as this is the maximum any protocol could have achieved. It has been established that the protocol generates a similar number of redundant transmissions as found in simulation and so implementation of the optimisation techniques discussed in Chapter 4 would be necessary.

7.5 Evaluation of τ estimate

The effectiveness of our estimate for τ from equation 3.6, can now be examined with respect to all experiments. We here display the formula once more for simplicity. For our simulations in chapters 3, 4 and 5 we use 64 nodes which gives us an the following.

$$\tau = 2\lceil \ln n + \gamma \rceil$$

$$\tau = 2\lceil \ln 64 + 0.5772 \ldots \rceil$$

$\tau = 10$

In chapter 3, where we make initial investigations into the performance of EG, we find that the figures indicate that they do, indeed, achieve coverage close to 1. In fact, when the density is high, the threshold is rather conservative.

In chapter 4 we show that we can reduce the number of redundant transmissions significantly in higher density networks. Specifically we show that using *α-reduction* we can achieve a low, almost constant, level of redundancy in high densities regardless of *tau*: with *density* = 6.5, at $\tau = 2$, redundant transmissions ≈ 1.5; at $\tau = 10$, redundant transmissions ≈ 2.5. This means that a single τ can be used effectively across all densities with minimal additional redundancy.

In chapter 5 we show that when multiple broadcasts are made in parallel we can achieve the same high coverage with $\tau = 10$. We have also show that this is sufficient to achieve FIFO delivery for all densities.

We made a final experimental evaluation of EG in chapter 6. Here we show that for 19 nodes using $\tau = 8$ we achieved a coverage of 0.64 and 0.59 for Thursday and Friday's experiments respectively. Moreover, we find that in our ideal network, real mobility simulation (using the captured real world mobility) that coverage reaches it's maximum coverage of 0.92, and 0.88 respectively at $\tau = 8$, see figures 6.10 and 6.11. From this point increasing τ has negligible improvement on coverage even at $\tau = 16$. It is therefore unlikely a higher coverage than this is possible, which indicates that our experimental coverage results provide approximately 70% of the maximum possible coverage in a real world scenario.

7.6 Take Home Message

- **Encounter:** On discovering a new node retransmit broadcast message m, increment $c(m)$.
- **τ:** When $c(m) > \tau$ terminate.
- **Estimate τ:** $\tau = 2\lceil \ln n + \gamma \rceil$

Encounter Gossip provides a high coverage broadcast solution for highly partitioned ad-hoc networks. Simple optimisations proposed here automatically come into effect when density is high and allow the protocol to provide high coverage with minimal redundancy. EG can also provide FIFO delivery in a cost effective manner. An additional attractive feature is that the deployment of EG does not require prior knowledge of network topology, density, speed or mobility pattern; only the number of nodes.

7.7 Further Work

Our encounter threshold propagation gets around issues with using lifetimes to limit the propagation of broadcast in MANET. However, if mobility may be such that even this is not sufficient, and the threshold expires before the broadcast is complete there will be lower coverage. One strategy for helping to deal with this could be to maintain a limited recent neighbourhood memory, in order to reduce the effect of encountering the same nodes repeatedly. In order to recover from anything that may cause loss of coverage a method of requesting messages should also be investigated. One possible strategy is outlined below.

The data obtained from the real world experiment may provide us with yet more information from further investigation. We should examine at the clustering and node degree of the mobility traces to establish its effect on Encounter Gossip's performance.

At present we have not investigated the behaviour of our protocol when messages are too great in number for all to be buffered. Naturally a FIFO approach to this would lead older messages to expire whilst they still may be relevant to currently un-encountered partitions. This extinction of messages would also make robust FIFO delivery more problematic. Here we outline an approach to share responsibility for messages across the network. When making its final transmission of a broadcast a node inserts a *keeping* flag into the message which is initially holds the value 1. All nodes hearing this, tag their copy of the message with this information. When they come to their final transmission they also add the *keeping* flag incrementing the value by 1. We would experiment with a threshold value at which point to not transmit a *keeping* flag, the higher the threshold, the more nodes would keep each message. When buffers become full nodes may delete messages which it has not decided to keep, with priority of deletion then going to those with the highest *keeping* flag value. The necessary recovery of missing messages introduces additional delay in propagation, the effects of different recovery policies should be assessed with this in mind.

This distribution of responsibility for messages lends itself to helping recover lost messages. Since messages will be around for longer than a single buffer may contain them, recovery attempts may still be successful after a longer period. This recovery method and evaluation of these enhancements will be well worth investigating in the future.

In this thesis we have shown EG's high coverage in a range of scenarios. We have provided a limited comparison with Hypergossiping showing that for a range of node densities Encounter Gossip provides higher coverage than Hypergossiping at a moderate cost of higher repeated transmissions. Yet it would be prudent to provide a more in depth study comparing against other algorithms in the *Broadcast-in-time* scenario. It would be worth performing simulations to compare and contrast the effective differences of performance between EG and protocols such as AF and HG using real world mobility traces. These protocols have not been tested in the real world and we would have liked to

evaluate and compared their performance under the relatively new Obstacle Mobility Model [41] and mobility models based on social networks theory [69, 20]. Recent work also provides new ways of generating realistic mobility traces which may be used [87].

7.7.1 Node Failure

We do not claim EG is a fault tolerant protocol. However it does exhibit some resilience to node failure. We present the following worst case scenario for Encounter Gossip protocol. Where the first node has failed to propagate it's message on all but the final transmission:

- $Node_0$ of N nodes EG broadcasts a message m.
- $Node_0$ has made τ broadcasts; one broadcast still to make.
- $EncNodes_0 = \{Node_i | Node_0$ has transmitted m to $Node_i\}$
- $|EncNodes_0| = \tau$
- $\forall Node_i \in EncNodes_0$, $Node_i$ has crashed.
- $Node_0$ transmits for the final time to $Node_j$.

On first sight this may look like a bad scenario for EG. However since $Node_j$ *has* received the message it will now perform τ transmissions on encounters. This is similar to $Node_j$ initiating a message in a network of size $N - \tau$. Thus 2 nodes have the message with a lower effective network size. Therefore the current τ will likely be as or more effective at reaching the remaining nodes. Further investigation into these properties would be interesting.

Bibliography

[1] I.F. Akyildiz and I.H. Kasimoglu. Wireless sensor and actor networks: research challenges. *Ad Hoc Networks*, 2(4):351–367, 2004.

[2] M. Balazinska and P. Castro. Characterizing Mobility and Network Usage in a Corporate Wireless Local-Area Network. *Proceedings of MobiSys 2003*, pages 303–316, 2003.

[3] C. Becker, M. Bauer, and J. Hähner. Usenet-on-the-fly: supporting locality of information in spontaneous networking environments. *CSCW 2002 Workshop on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments, New Orleans, USA*, 2002.

[4] P. Bellavista, A. Corradi, and E. Magistretti. REDMAN: An optimistic replication middleware for read-only resources in dense MANETs. *Pervasive and Mobile Computing*, 1(3):279–310, 2005.

[5] C. Bettstetter, G. Resta, and P. Santi. The node distribution of the random waypoint mobility model for wireless ad hoc networks. *IEEE Trans. Mobile Computing*, 2(3):257–269, July–September 2003.

[6] V. Bhandari and N.H. Vaidya. On reliable broadcast in a radio network. *Proceedings of the twenty-fourth annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, pages 138–147, 2005.

[7] K.P. Birman, M. Hayden, O. Ozkasap, Z. Xiao, M. Budiu, and Y. Minsky. Bimodal multicast. *ACM Transactions on Computer Systems (TOCS)*, 17(2):41–88, 1999.

[8] S. Burleigh, A. Hooke, L. Torgerson, K. Fall, V. Cerf, B. Durst, K. Scott, and H. Weiss. Delay-tolerant networking: an approach to interplanetary Internet. *Communications Magazine, IEEE*, 41(6):128–136, 2003.

[9] M. Burmester, T.V. Le, and A. Yasinsac. Adaptive gossip protocols: Managing security and redundancy in dense ad hoc networks. *Ad Hoc Networks*, 5(3):313–323, 2007.

[10] T. Camp, J. Boleng, and V. Davies. A survey of mobility models for ad hoc network research. *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, 2(5):483–502, 2002.

[11] J. Cartigny, F. Ingelrest, and D. Simplot. RNG relay subset flooding protocols in mobile ad-hoc networks. *International Journal of Foundations of Computer Science*, 14(2):253–265, 2003.

[12] J. Cartigny and D. Simplot. Border Node Retransmission Based Probabilistic Broadcast Protocols in Ad-Hoc Networks. *Telecommunication Systems*, 22(1):189–204, 2003.

[13] J. Cartigny, D. Simplot, and I. Stojmenovic. Localised energy efficient broadcast for wireless networks with directional antennas. *Proceedings of the Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET2002)*, 2002.

[14] J. Cartigny, D. Simplot, and I. Stojmenovic. Localised minimum-energy broadcasting in ad-hoc networks. *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 3, 2003.

[15] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, and J. Scott. Pocket switched networks: Real-world mobility and its consequences for opportunistic forwarding. *University of Cambridge Computer Laboratory, Tech. Rep. UCAM-CL-TR-617, Feb*, 2005.

[16] R. Chandra, V. Ramasubramanian, and K. Birman. Anonymous Gossip: improving multicast reliability in mobile ad-hocnetworks. *Distributed Computing Systems, 2001. 21st International Conference on.*, pages 275–283, 2001.

[17] C. Chiang and M. Gerla. On-demand multicast in mobile wireless networks. *Proceedings of IEEE ICNP*, pages 260–270, 1998.

[18] K.W. Chin, J. Judge, A. Williams, and R. Kermode. Implementation experience with MANET routing protocols. *ACM SIGCOMM Computer Communication Review*, 32(5):49–59, 2002.

[19] I. Chlamtac, M. Conti, and J.J.N. Liu. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks*, 1(1):13–64, 2003.

[20] P. Costa, M. Mascolo, C. Musolesi, and G.P. Picco. Socially-aware Routing for Publish-Subscribe in Delay-tolerant Mobile Ad Hoc Networks. *IEEE Journal of Selected Areas of Communications*, 2008.

[21] J. Crowcroft, C. Diot, M. Liberatore, and M. Pias. Pilgrim-A Communication Paradigm for Ad-hoc Communications. *Unpublished*, 2004.

[22] F. Dai and J. Wu. Performance analysis of broadcast protocols in ad hoc networks based on self-pruning. *Parallel and Distributed Systems, IEEE Transactions on*, 15(11):1027–1040, 2004.

[23] KV Davis, R.S. Co, and F. Wayne. JTRS-an open, distributed-object computing software radioarchitecture. In *Digital Avionics Systems Conference, 1999. Proceedings. 18th*, volume 2, 1999.

[24] Vladimir Dyo and Cecilia Mascolo. Efficient Node Discovery in Mobile Wireless Sensor Networks. *ACM/IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS08),Santorini. Greece.*, 2008.

[25] P.T. Eugster, R. Guerraoui, and P. Kouznetsov. Δ-reliable broadcast: A probabilistic measure of broadcast reliability. *Proceedings of the 24th IEEE International Conference on Distributed Computing Systems (ICDCS 2004)*, pages 24–26.

[26] JJ Garcia-Luna-Aceves and E.L. Madruga. A Multicast Routing Protocol for Ad-Hoc Networks. *Proceedings of IEEE INFOCOM*, 99:784–792, 1999.

[27] T. Gopalsamy, M. Singhal, D. Panda, and P. Sadayappan. A reliable multicast algorithm for mobile ad hoc networks. *Proceedings of the 22nd International Conference on Distributed Computing Systems, 2002.*, pages 563–570, 2002.

[28] M. Grossglauser and D.N.C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 10(4):477–486, 2002.

[29] I. Gupta, K.P. Birman, and R. van Renesse. Fighting fire with fire: using randomised gossip to combat stochastic scalability limits. *Quality and Reliability Engineering International*, 18(3):165–184, 2002.

[30] A. Guy and R. Nelson. The familiar project. *2006 at: http://familiar.handhelds.org.*

[31] Z.J. Haas, J.Y. Halpern, and L. Li. Gossip-based ad hoc routing. *IEEE/ACM Transactions on Networking (TON)*, 14(3):479–491, 2006.

[32] V. Hadzilacos and S. Tueg. *Fault-Tolerant Broadcasts and Related Problems*, pages 330–331. Addison-Wesley, 1993.

[33] J. Hahner, C. Becker, and K. Rothermel. A protocol for data dissemination in frequently partitioned mobile ad hoc networks. *Proceedings of IEEE ISCC 2003, Eighth International Symposium on Computers and Communication.*, pages 633–640, 2003.

[34] W.R. Heinzelman, J. Kulik, and H. Balakrishnan. Adaptive protocols for information dissemination in wireless sensor networks. *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 174–185, 1999.

[35] M. Heissenbuttel, T. Braun, M. Walchli, and T. Bernoulli. Optimized stateless broadcasting in wireless multi-hop networks. *IEEE Infocom 2006*, pages 23–29.

[36] T. Henderson, D. Kotz, and I. Abyzov. The changing usage of a mature campus-wide wireless network. *Proceedings of the 10th annual international conference on Mobile computing and networking*, pages 187–201, 2004.

[37] Christopher Ho, Katia Obraczka, Gene Tsudik, and Kumar Viswanath. Flooding for reliable multicast in multi-hop ad hoc networks. In *Proceedings of the 3rd International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications*, pages 64–71, Seattle, WA, 1999.

[38] C. Hu, Y. Hong, and Urbana Hou, J. On Mitigating the Broadcast Storm Problem with Directional Antennas. *ICC '03. IEEE International Conference on Communications*, 51:104 – 110, 2003.

[39] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, and C. Diot. Pocket switched networks and human mobility in conference environments. *Applications, Technologies, Architectures, and Protocols for Computer Communication*, pages 244–251, 2005.

[40] A. Jardosh, E. Belding-Royer, K. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks, 2003.

[41] A. Jardosh, E.M. Belding-Royer, K.C. Almeroth, and S. Suri. Towards realistic mobility models for mobile ad hoc networks. *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 217–229, 2003.

[42] D.B. Johnson and D.A. Maltz. Dynamic source routing in ad hoc wireless networks. *Mobile Computing*, 353(153-181):152, 1996.

[43] P. Juang, H. Oki, Y. Wang, M. Martonosi, L.S. Peh, and D. Rubenstein. Energy-Efficient Computing for Wildlife Tracking: Design Tradeoffs and Early Experiences with ZebraNet. In *Architectural Support for Programming Languages and Operating Systems: Proceedings of the 10 th international conference on Architectural support for programming languages and operating systems*, volume 5, pages 96–107, 2002.

[44] K. Kanchanasut, A. Tunpan, M.A. Awal, D.K. Das, T. Wongsaardsakul, and Y. Tsuchimoto. A Multimedia Communication System for Collaborative Emergency Response Operation in Disaster-affected Areas. Technical report, Interlab Technical Report TR 2007-1.

[45] I. Kang and R. Poovendran. A comparison of power-efficient broadcast routing algorithms. *Global Telecommunications Conference, 2003. GLOBECOM'03. IEEE*, 1, 2003.

[46] I. Kang and R. Poovendran. Maximizing static network lifetime of wireless broadcast ad hoc networks. *Communications, 2003. ICC'03. IEEE International Conference on*, 3, 2003.

[47] I. Kang and R. Poovendran. Power-efficient broadcast routing in adhoc networks using directional antennas: technology dependence and convergence issues. *University of Washington, Washington, USA, Tech. Rep. UWEETR-2003-0015, July*, 2003.

[48] G. Karumanchi, S. Muralidharan, and R. Prakash. Information dissemination in partitionable mobile ad hoc networks. *Reliable Distributed Systems, 1999. Proceedings of the 18th IEEE Symposium on*, pages 4–13, 1999.

[49] A.M. Kermarrec, L. Massoulie, and A.J. Ganesh. Probabilistic reliable dissemination in large-scale systems. *IEEE Transactions on Parallel and Distributed Systems*, 14(3):248–258, 2003.

[50] A. Khelil. A Generalised Broadcasting Technique for Mobile Ad Hoc Networks. *PhD Thesis, University of Stuttgart, Department of Computer Science, Distributed Systems Group*, 2007.

[51] A. Khelil, P.J. Marrón, C. Becker, and K. Rothermel. Hypergossiping: A generalised broadcast strategy for mobile ad hoc networks. *Ad Hoc Networks*, 5(5):531–546, 2007.

[52] D. Kotz, C. Newport, R.S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. *Proceedings of the 7th ACM international symposium on Modeling, analysis and simulation of wireless and mobile systems*, pages 78–82, 2004.

[53] D. Kotz, C. Newport, R.S. Gray, J. Liu, Y. Yuan, and C. Elliott. Experimental evaluation of wireless simulation assumptions. *Technical Report TR2004-507*, 2004.

[54] T.J. Kwon and M. Gerla. Efficient flooding with Passive Clustering (PC) in ad hoc networks. *ACM SIGCOMM Computer Communication Review*, 32(1):44–56, 2002.

[55] D. Lea and D. Lea. *Concurrent Programming in Java (tm): Design Principles and Patterns*. Addison-Wesley Professional, 2000.

[56] JSR166 Lea, D. backport-util-concurrent. *2006, http://dcl.mathcs.emory.edu/util/backport-util-concurrent/*.

[57] S.J. Lee, M. Gerla, and C.C. Chiang. On-demand multicast routing protocol. *Wireless Communications and Networking Conference, 1999. WCNC. 1999 IEEE*, pages 1298–1302, 1999.

[58] BA Lewis and JS Robinson. Triangulation of planar regions with applications. *The Computer Journal*, 21(4):324–332, 1978.

[59] L. Li, J. Halpern, and Z. Haas. Gossip-based ad hoc routing, 2002.

[60] H. Lim and C. Kim. Multicast tree construction and flooding in wireless ad hoc networks. *Proceedings of the ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems (MSWIM)*, pages 61–68, 2000.

[61] C. Lindemann and O.P. Waldhorst. Epidemic Dissemination of Presence Information in Mobile Instant Messaging Systems. *Kommunikation in Verteilten Systemen (Kivs) 2005: 14. Itg/Gi-Fachtagung Kommunikation in Verteilten Systemen (Kivs 2005) Kaiserslautern, 28. Februar-3. Marz 2005*, 2005.

[62] T. Lindholm and F. Yellin. *Java Virtual Machine Specification.* Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1999.

[63] J. Liu, D. Sacchetti, F. Sailhan, and V. Issarny. Group management for mobile Ad Hoc networks: design, implementation and experiment. *Proceedings of the 6th international conference on Mobile data management*, pages 192–199, 2005.

[64] W. Lou and J. Wu. On reducing broadcast redundancy in ad hoc wireless networks. *System Sciences, 2003. Proceedings of the 36th Annual Hawaii International Conference on*, page 10, 2003.

[65] W. Lou and J. Wu. Double-covered broadcast (DCB): a simple reliable broadcast algorithm in MANETs. *INFOCOM 2004. Twenty-third AnnualJoint Conference of the IEEE Computer and Communications Societies*, 3, 2004.

[66] R. Lougher. JamVM. *2006 at: http://jamvm.sourceforge.net/.*

[67] H. Lundgren, E. Nordströ, and C. Tschudin. Coping with communication gray zones in IEEE 802.11 b based ad hoc networks. *Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*, pages 49–55, 2002.

[68] M. McNett and G.M. Voelker. Access and mobility of wireless PDA users. *ACM SIGMOBILE Mobile Computing and Communications Review*, 9(2):40–55, 2005.

[69] Cecilia Mascolo Mirco Musolesi. Designing Mobility Models based on Social Networks Theory. *ACM Mobile Computing and Communications Review*, 2007.

[70] I. Mitrani. *Probabilistic Modelling.* Cambridge University Press, 1998.

[71] M. Musolesi, C. Mascolo, and S. Hailes. EMMA: Epidemic Messaging Middleware for Ad hoc networks. *Personal and Ubiquitous Computing*, 10(1):28–36, 2006.

[72] S.Y. Ni, Y.C. Tseng, Y.S. Chen, and J.P. Sheu. The broadcast storm problem in a mobile ad hoc network. *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, 1999.

[73] K. Obraczka, G. Tsudik, and K. Viswanath. Pushing the limits of multicast in Ad hoc networks. *21 st IEEE International Conference on Distributed Computing Systems*, 1:719–722, 2001.

[74] E. Pagani and G.P. Rossi. Reliable broadcast in mobile multihop packet networks. *Proceedings of the 3rd annual ACM/IEEE international conference on Mobile computing and networking*, pages 34–42, 1997.

[75] M. Pandey and D. Zappala. The Effects of Mobility on Multicast Routing in Ad Hoc Networks. 2004.

[76] W. Peng and X. Lu. AHBP: An efficient broadcast protocol for mobile ad hoc networks. Journal of Science and Technology. *Beijing, China*, 2002.

[77] W. Peng and X.C. Lu. On the reduction of broadcast redundancy in mobile ad hoc networks. *Proceedings of the 1st ACM international symposium on Mobile ad hoc networking & computing*, pages 129–130, 2000.

[78] C. Perkins, E. Belding-Royer, and S. Das. RFC3561: Ad hoc On-Demand Distance Vector (AODV) Routing. *Internet RFCs*, 2003.

[79] C.E. Perkins and P. Bhagwat. Highly dynamic Destination-Sequenced Distance-Vector routing (DSDV) for mobile computers. *Proceedings of the conference on Communications architectures, protocols and applications*, pages 234–244, 1994.

[80] C.E. Perkins and E.M. Royer. Ad-hoc on-demand distance vector routing. *Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications*, 2:90–100, 1999.

[81] Amir Qayyum, Laurent Viennot, and Anis Laouiti. Multipoint relaying: An efficient technique for flooding in mobile wireless networks. Technical Report Research Report RR-3898, INRIA, February 2000.

[82] D. Reichardt, M. Miglietta, L. Moretti, P. Morsink, and W. Schulz. CarTALK 2000: safe and comfortable driving based upon inter-vehicle-communication. In *Intelligent Vehicle Symposium, 2002. IEEE*, volume 2, 2002.

[83] EM Royer, PM Melliar-Smith, and LE Moser. An analysis of the optimum node density for ad hoc mobile networks. *Communications, 2001. ICC 2001. IEEE International Conference on*, 3, 2001.

[84] E.M. Royer and C.E. Perkins. Multicast Ad hoc On-Demand Distance Vector (MAODV) Routing. *draft-ietf-manet-maodv-00, July*, 700, 2000.

[85] R. Ruppe, S. Griswald, P. Walsh, and R. Martin. Near Term Digital Radio (NTDR) system. In *MILCOM 97 Proceedings*, volume 3, 1997.

[86] Y. Sasson, D. Cavin, and A. Schiper. Probabilistic broadcast for flooding in wireless mobile ad hoc networks. *Wireless Communications and Networking, 2003. WCNC 2003. 2003 IEEE*, 2, 2003.

[87] S. Scellato, C. Mascolo, M. Musolesi, and V. Latora. Epcast: Controlled Dissemination in Human-based Wireless Networks by means of Epidemic Spreading Models. *Bio-Inspired Computing and Communication*, 2008.

[88] RC Shah, S. Roy, S. Jain, and W. Brunette. Data MULEs: modeling a three-tier architecture for sparse sensor networks. *Sensor Network Protocols and Applications, 2003. Proceedings of the First IEEE. 2003 IEEE International Workshop on*, pages 30–41, 2003.

[89] T. Smith. Wlan in Claremont Tower. *Computing Officer Support Materials, Computing Science, University of Newcastle upon Tyne*, 2006.

[90] F. Stann, J. Heidemann, R. Shroff, and M.Z. Murtaza. RBP: robust broadcast propagation in wireless networks. *Proceedings of the 4th international conference on Embedded networked sensor systems*, pages 85–98, 2006.

[91] Fred Stann, John Heidemann, Rajesh Shroff, and Muhammad Zaki Murtaza. RBP: Reliable broadcast propagation in wireless networks. Technical Report ISI-TR-2005-608, USC/Information Sciences Institute, November 2005.

[92] I. Stojmenovic, M. Seddigh, and J. Zunic. Dominating sets and neighbour elimination-based broadcasting algorithms in wireless networks. *IEEE Transactions on Parallel and Distributed Systems*, 13(1):14–25, 2002.

[93] J. Sucec and I. Marsic. An efficient distributed network-wide broadcast algorithm for mobile ad hoc networks. *Rutgers University, CAIP Technical Report*, 248, 2000.

[94] J. Tian, P.J. Marron, and K. Rothermel. Location-based hierarchical data aggregation for vehicular ad hoc networks. In *Proceedings of Communication in Distributed Systems*. Springer, 2005.

[95] G. Toussaint. The relative neighbourhood graph of a finite planar set. *Pattern Recognition*, 12(4):261–268, 1980.

[96] Y.C. Tseng, S.Y. Ni, and E.Y. Shih. Adaptive approaches to relieving broadcast storms in a wireless multihop mobile ad hoc network. *Computers, IEEE Transactions on*, 52(5):545–557, 2003.

[97] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks, April 2000.

[98] K. Viswanath and K. Obraczka. An adaptive approach to group communications in multi hop ad hoc networks. *Proceedings of IEEE ISCC 2002, Seventh International Symposium on Computers and Communications.*, pages 559–566, 2002.

[99] E. Vollset and P. Ezhilchelvan. Enabling reliable many-to-many communication in ad-hoc pervasive environments. *Pervasive Computing and Communications Workshops, 2005. PerCom 2005 Workshops. Third IEEE International Conference on*, pages 65–69, 2005.

[100] E. Vollset and P.D. Ezhilchelvan. Design and Performance-Study of Crash-Tolerant Protocols for Broadcasting and Reaching Consensus in MANETs. *Proceedings of the 24th Symposium on Reliable Distributed Systems (SRDS)*, pages 166–175, 2005.

[101] Y. Wang and JJ Garcia-Luna-Aceves. Broadcast Traffic in Ad Hoc Networks with Directional Antennas. *Proceedings of IEEE Global Telecommunications Conference (Globecom '03)*, pages 210–215, 2003.

[102] R. Wattenhofer, L. Li, P. Bahl, and Y.M. Wang. Distributed topology control for power efficient operation inmultihop wireless ad hoc networks. *Proceedings of IEEE INFOCOM 2001. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies.*, 3, 2001.

[103] JE Wieselthier, GD Nguyen, and A. Ephremides. On the construction of energy-efficient broadcast and multicasttrees in wireless networks. *Proceedings of IEEE INFOCOM 2000. Nineteenth Annual Joint Conference of the IEEE Computer and Communications Societies.*, 2, 2000.

[104] B. Williams and T. Camp. Comparison of broadcasting techniques for mobile ad hoc networks. In *Proceedings of the ACM International Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, pages 194–205, 2002.

[105] H. Wu, R. Fujimoto, R. Guensler, and M. Hunter. MDDV: a mobility-centric data dissemination algorithm for vehicular networks. *Proceedings of the first ACM workshop on Vehicular ad hoc networks*, pages 47–56, 2004.

[106] J. Wu and F. Dai. A generic distributed broadcast scheme in ad hoc wireless networks. *Computers, IEEE Transactions on*, 53(10):1343–1354, 2004.

[107] J. Wu and H. Li. On calculating connected dominating set for efficient routing in ad hoc wireless networks. *Proceedings of the 3rd international workshop on Discrete algorithms and methods for mobile computing and communications*, pages 7–14, 1999.

[108] Y. Yi, M. Gerla, and T.J. Kwon. Efficient flooding in ad hoc networks: a comparative performance study. *Communications, 2003. ICC'03. IEEE International Conference on*, 2, 2003.

[109] Y. Yi, M. Gerla, T.J. Kwon, and T. Technologies. Efficient Flooding in Ad Hoc Networks Using On-demand(Passive) Cluster Formation. *Proceedings of 2nd annual Mediterranean Ad Hoc Networking Workshop (Med-hoc-Net '03)*, 2003.

[110] J. Yoon, M. Liu, and B. Noble. Random waypoint considered harmful. *INFOCOM '03. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies. IEEE*, 2, 2003.

Appendix A

Appendix

A.1 Encounter Histories

A.1.1 Thursday

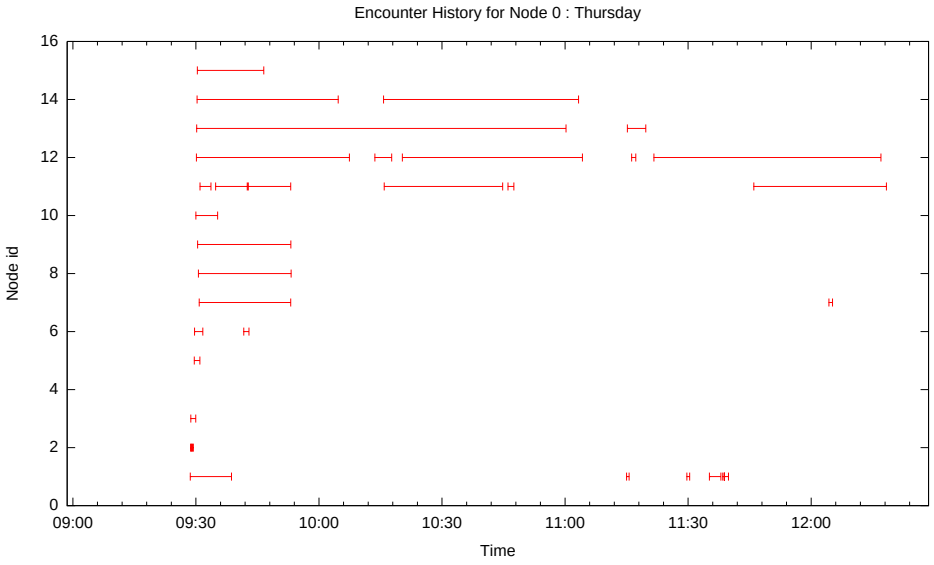

Figure A.1: Message History for node 0

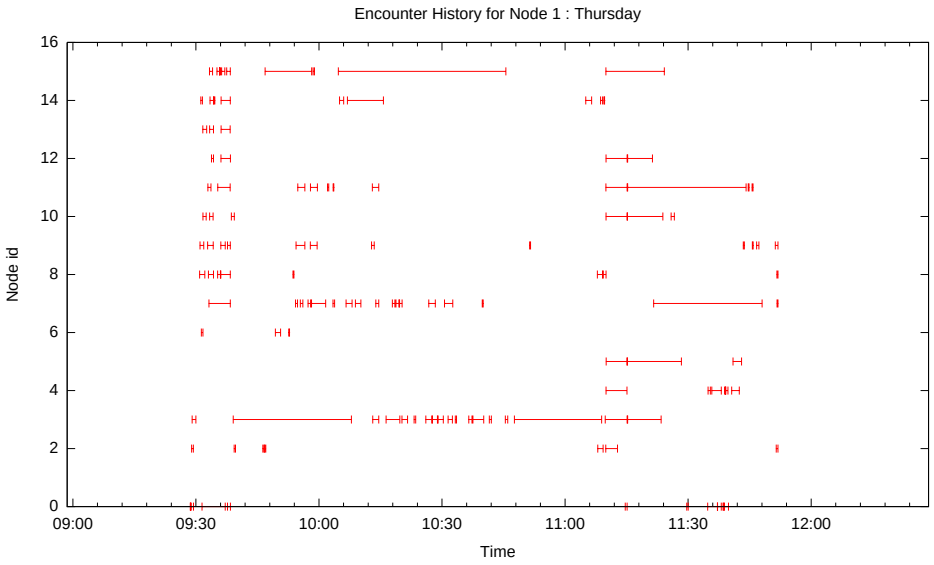

Figure A.2: Message History for node 1

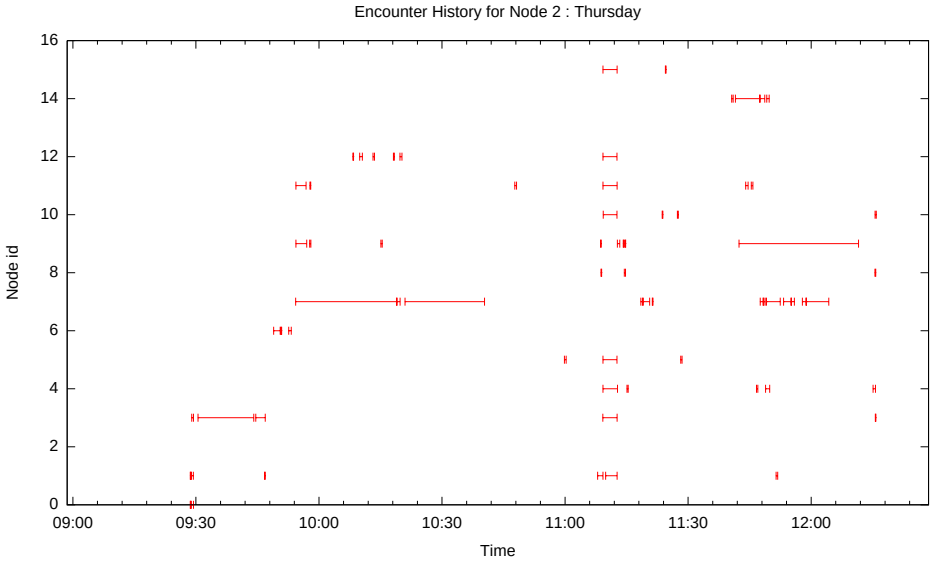

Figure A.3: Message History for node 2

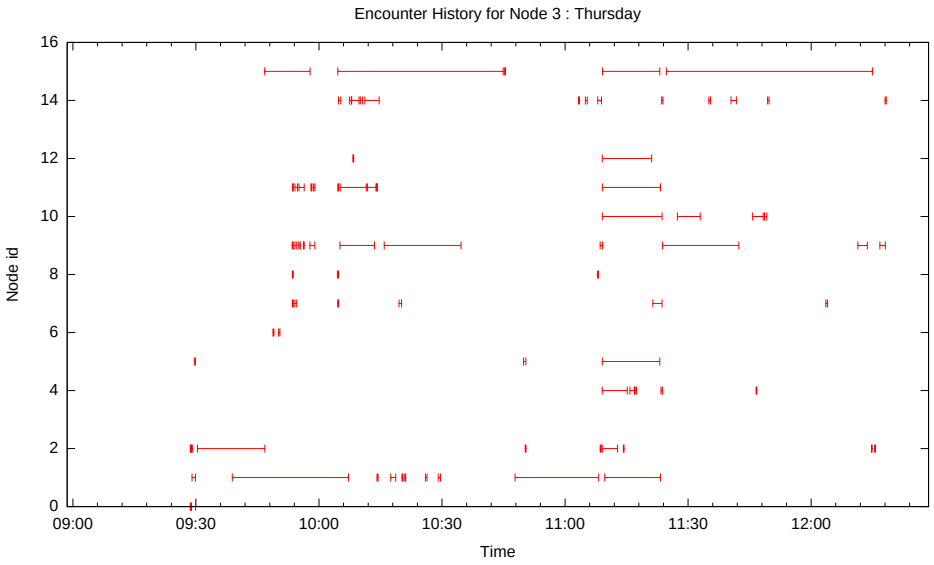

Figure A.4: Message History for node 3

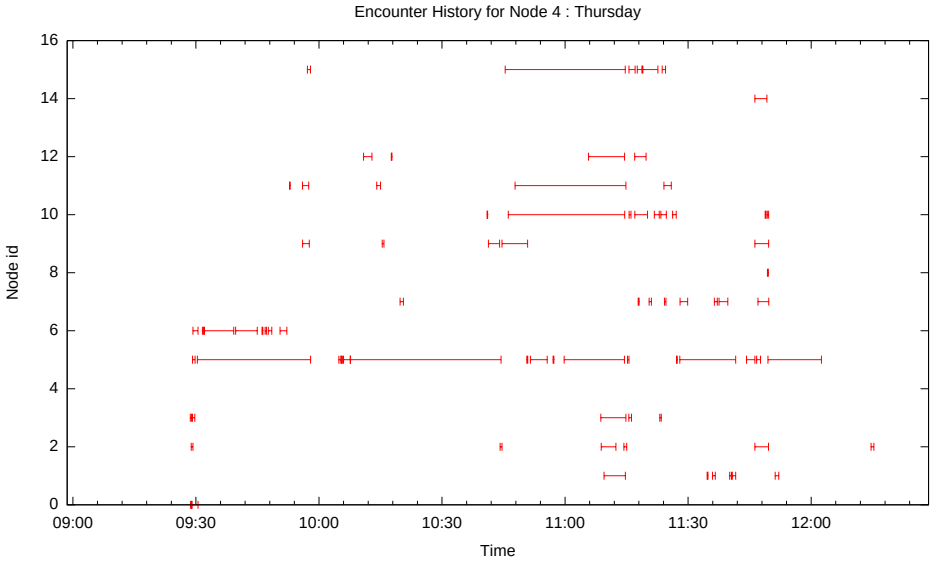

Figure A.5: Message History for node 4

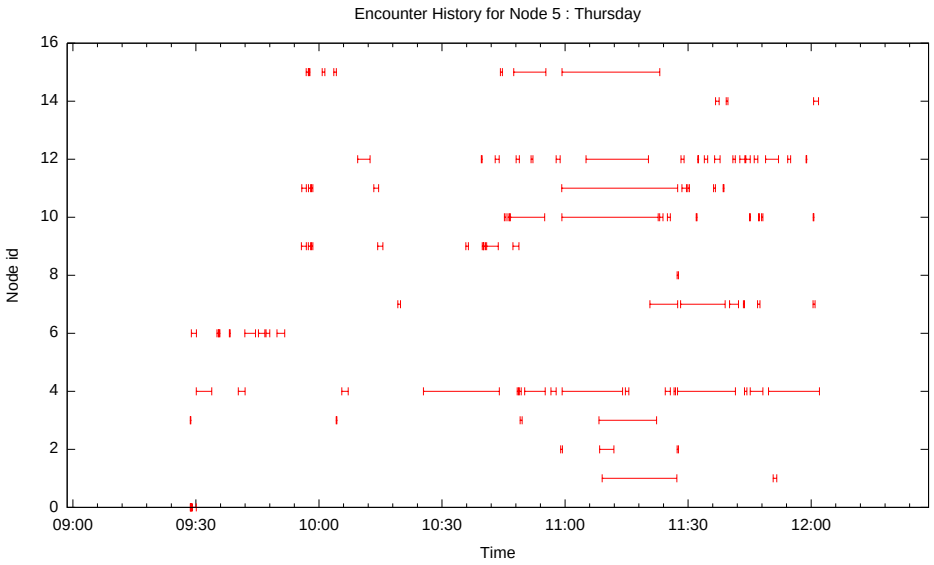

Figure A.6: Message History for node 5

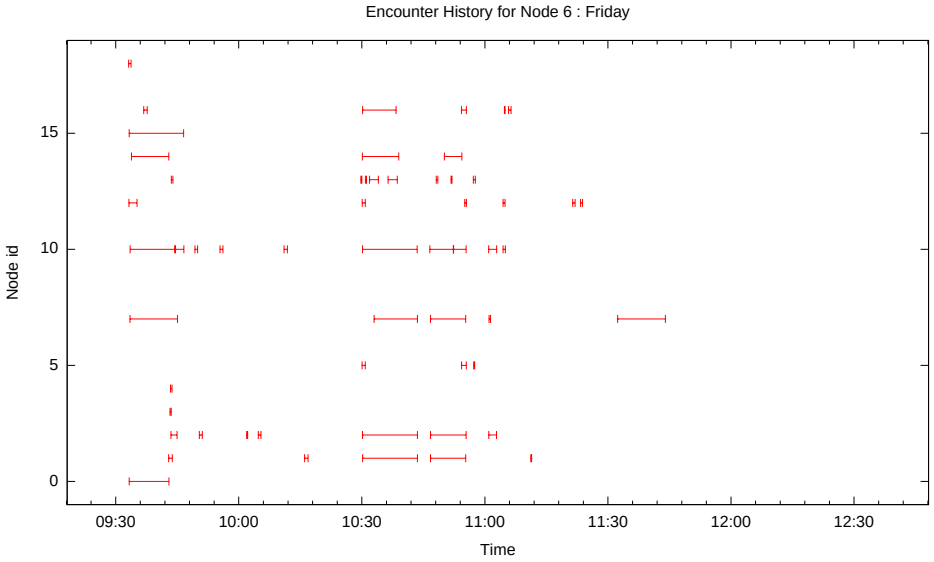

Figure A.7: Message History for node 6

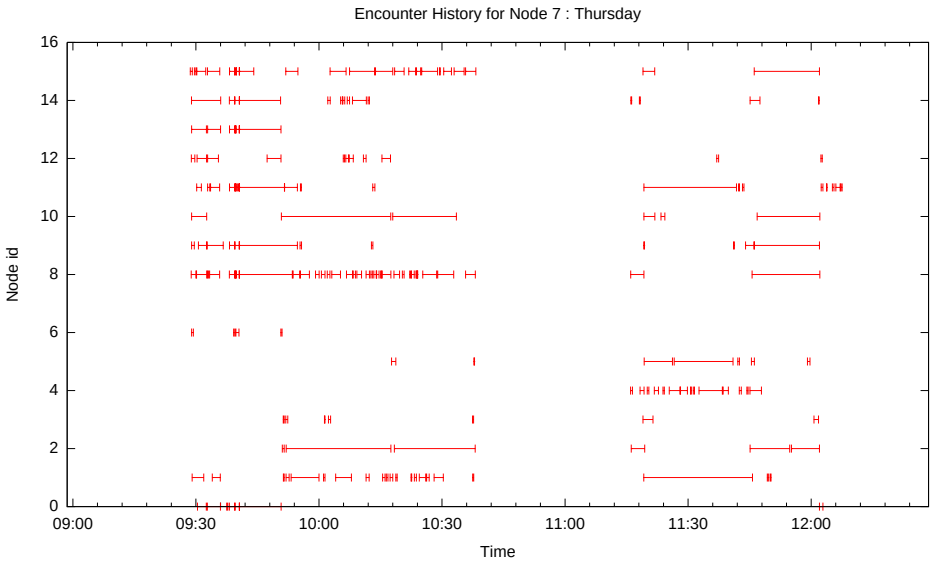

Figure A.8: Message History for node 7

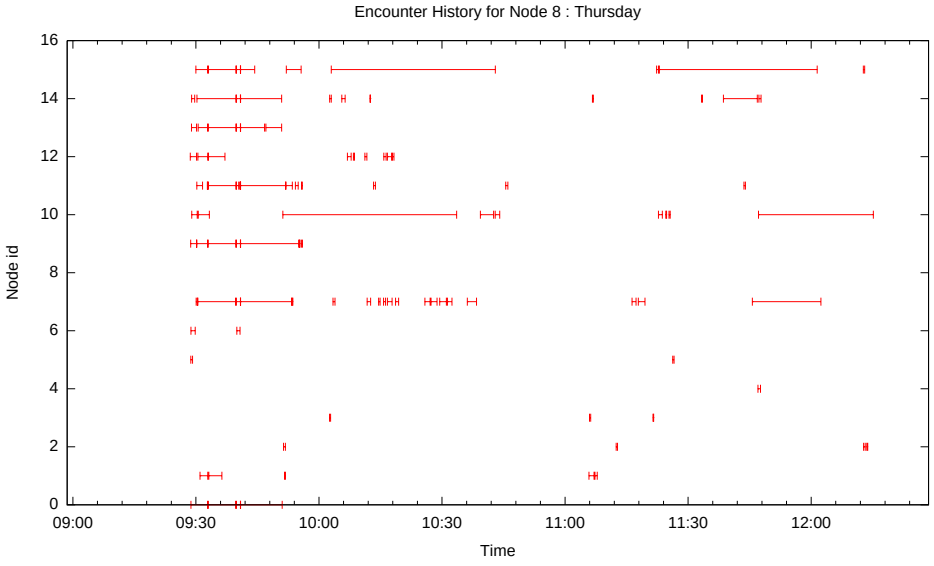

Figure A.9: Message History for node 8

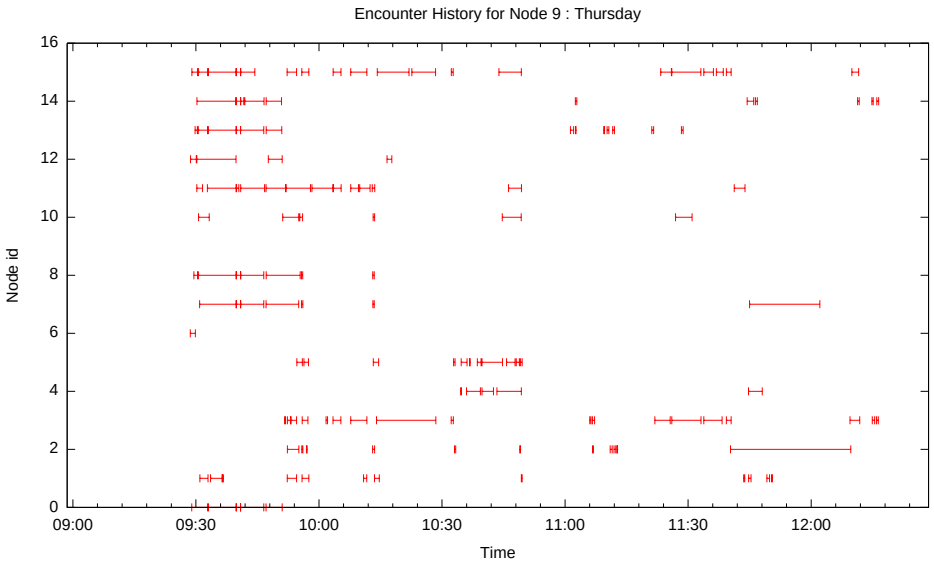

Figure A.10: Message History for node 9

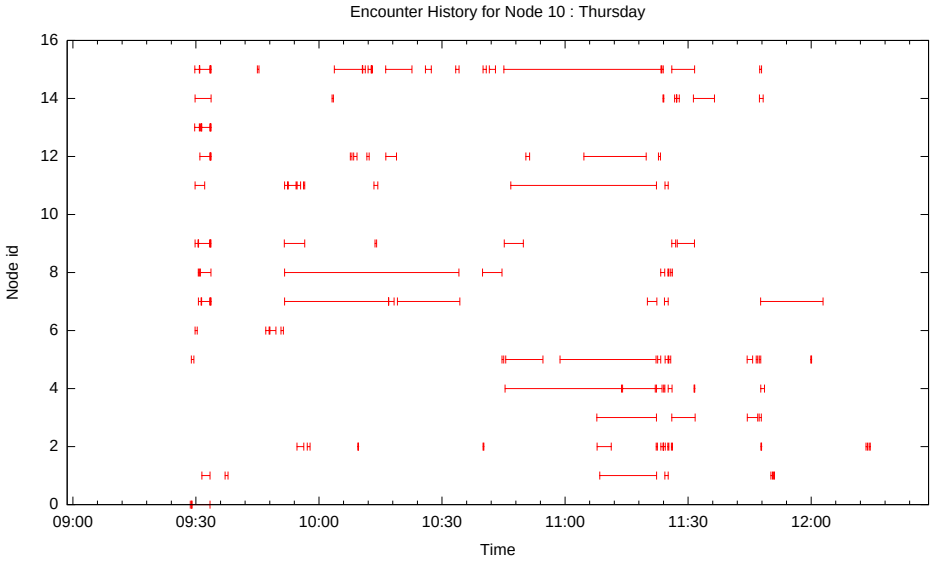

Figure A.11: Message History for node 10

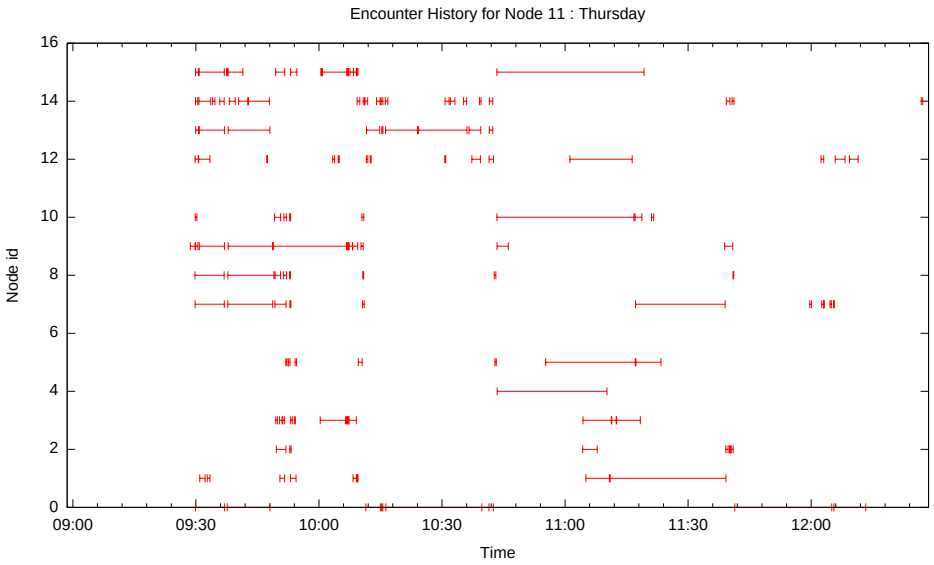

Figure A.12: Message History for node 11

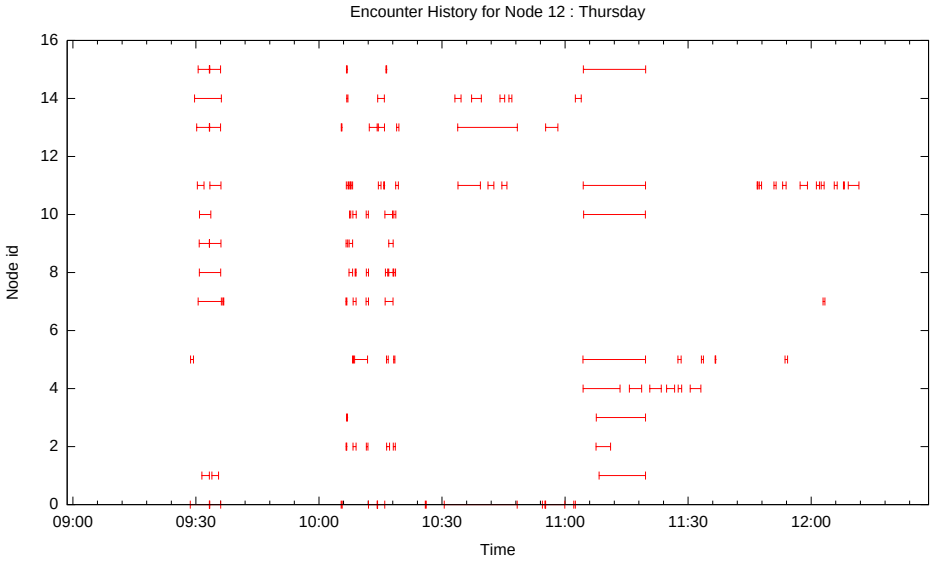

Figure A.13: Message History for node 12

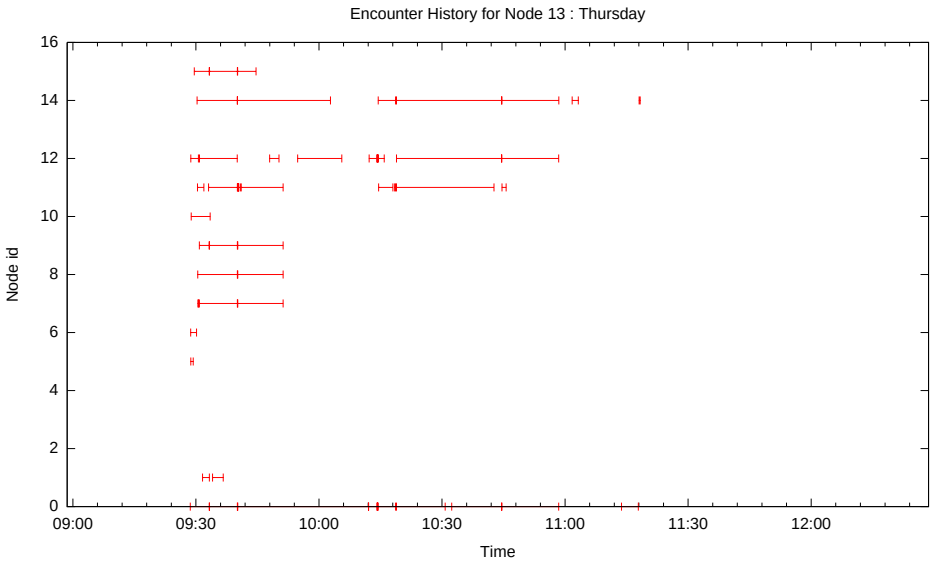

Figure A.14: Message History for node 13

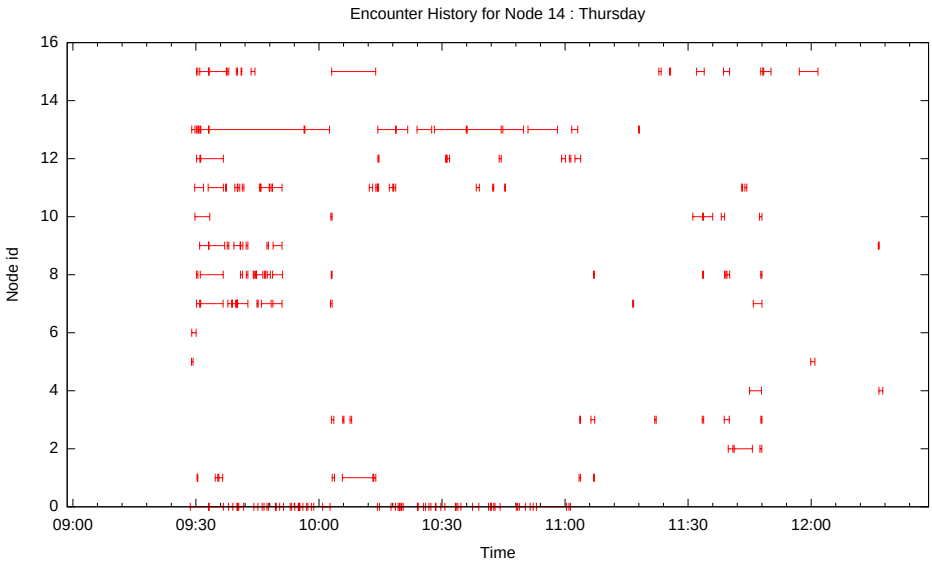

Figure A.15: Message History for node 14

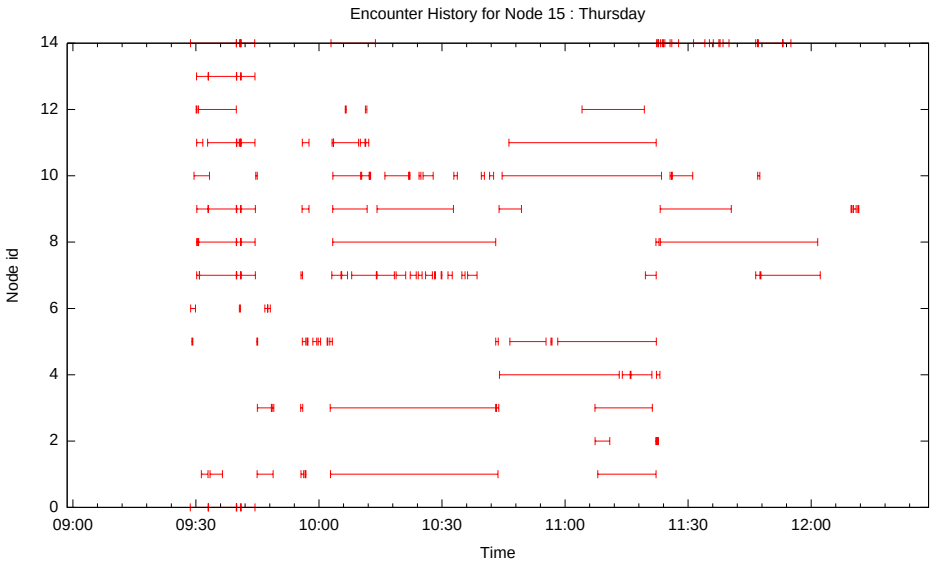

Figure A.16: Message History for node 15

A.1.2 Friday

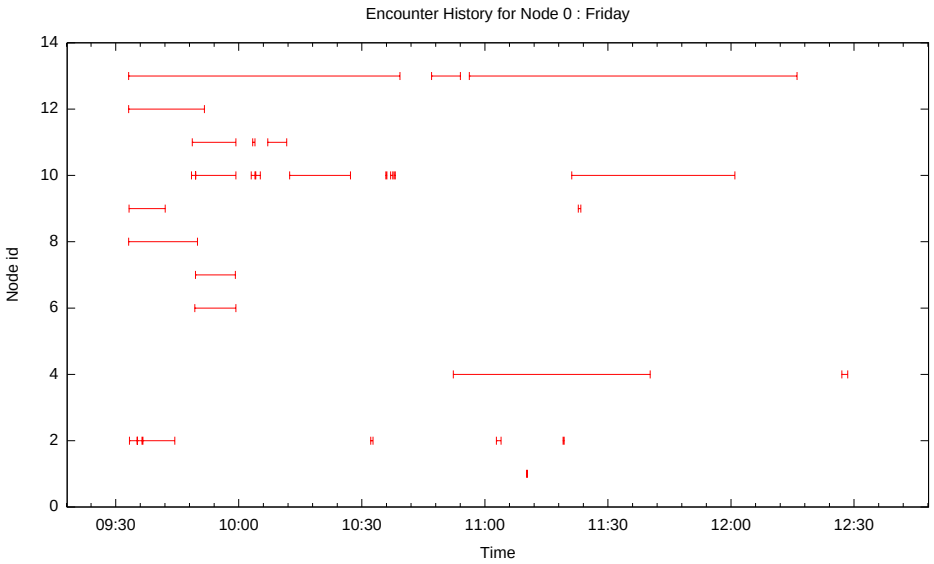

Figure A.17: Message History for node 0

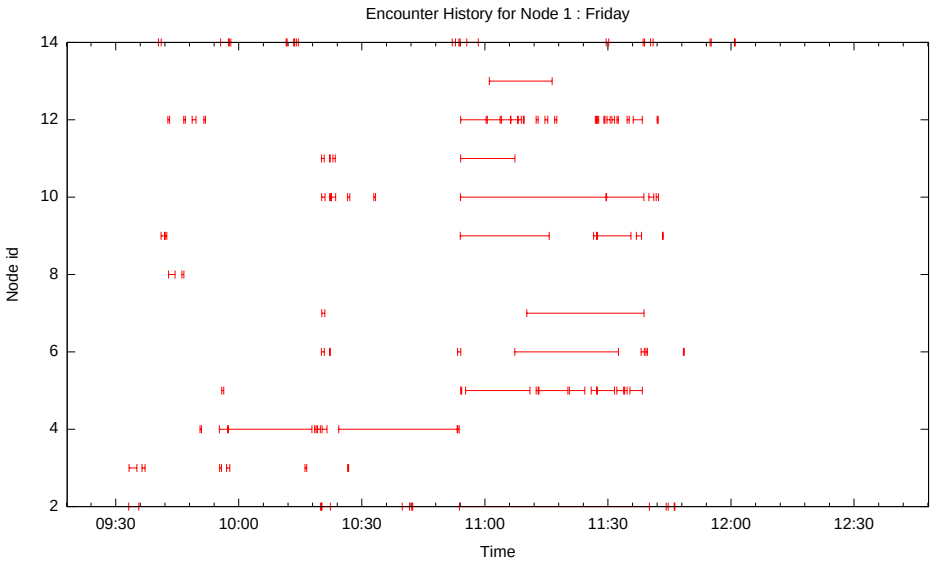

Figure A.18: Message History for node 1

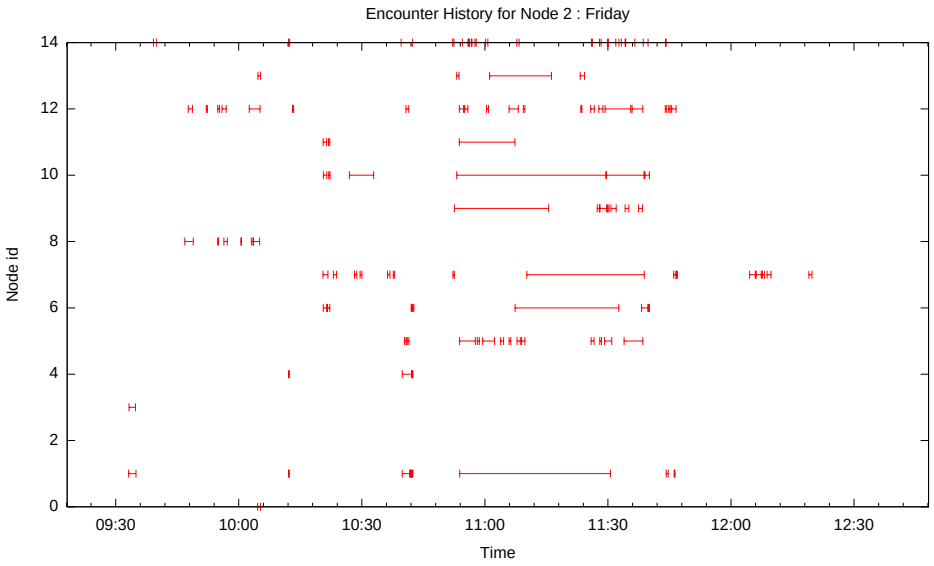

Figure A.19: Message History for node 2

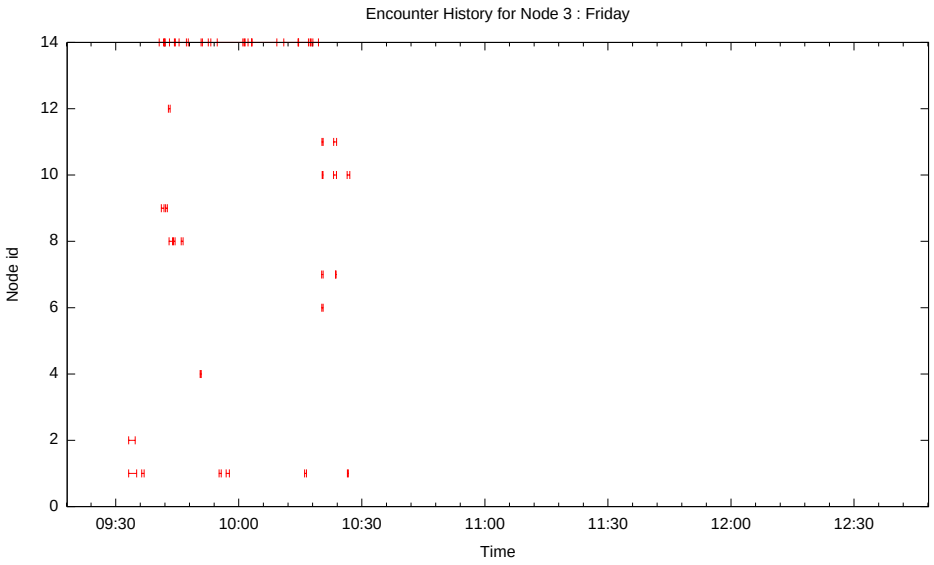

Figure A.20: Message History for node 3

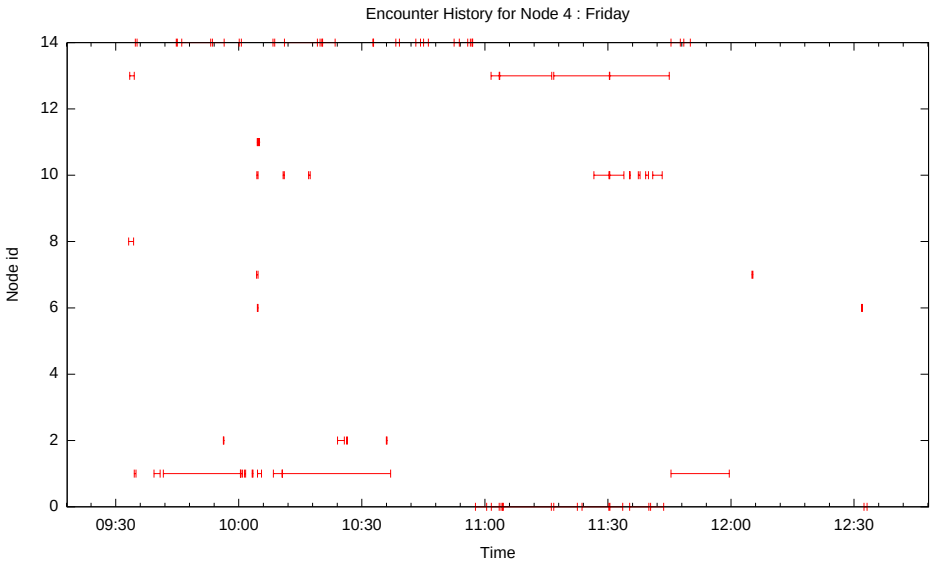

Figure A.21: Message History for node 4

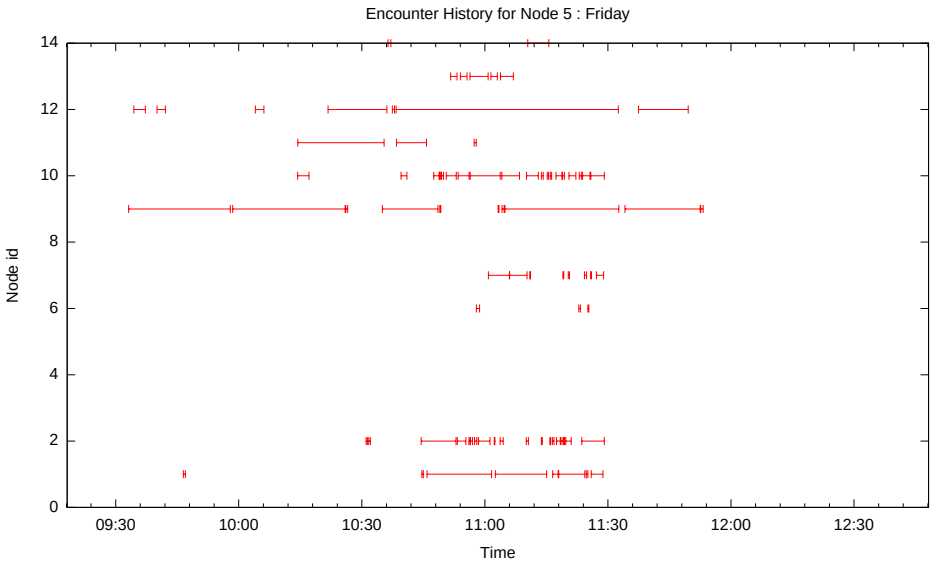

Figure A.22: Message History for node 5

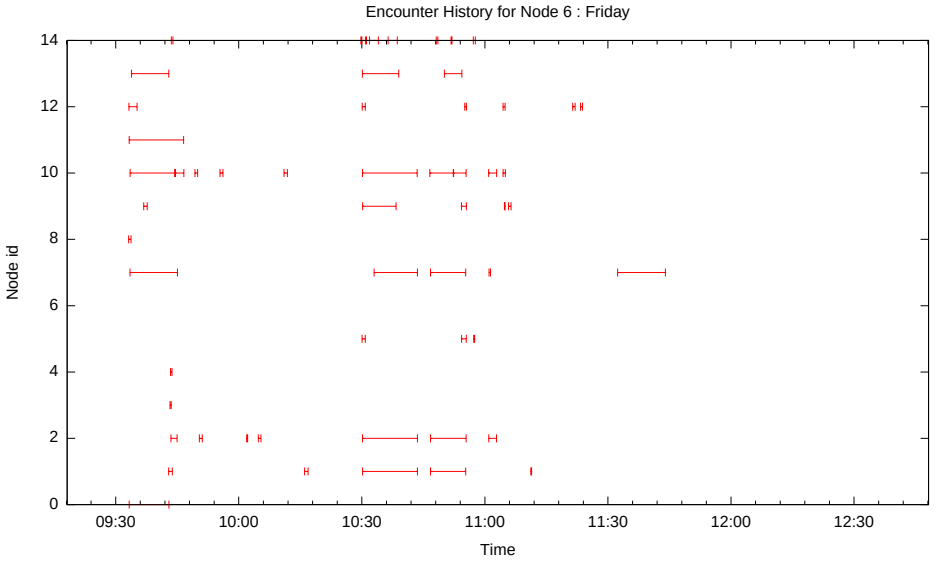

Figure A.23: Message History for node 6

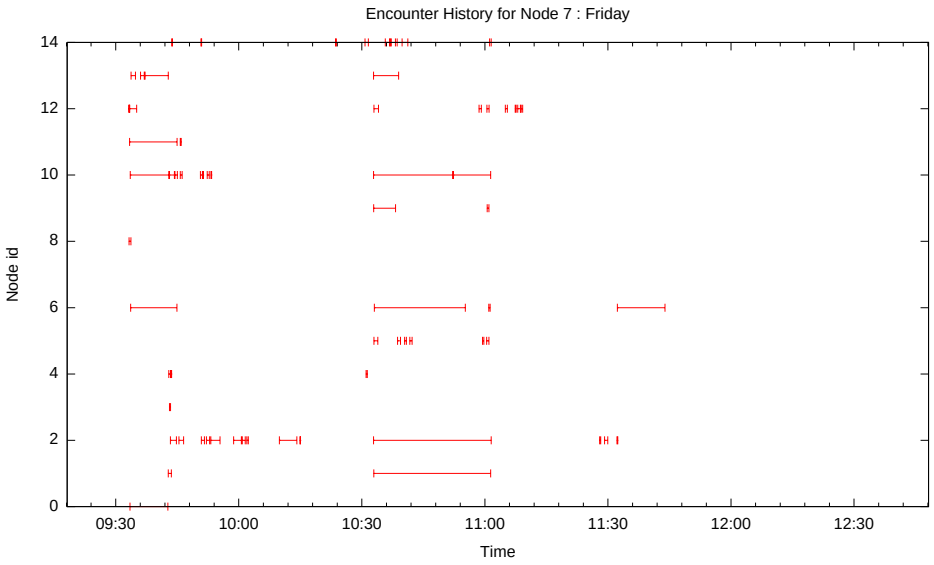

Figure A.24: Message History for node 7

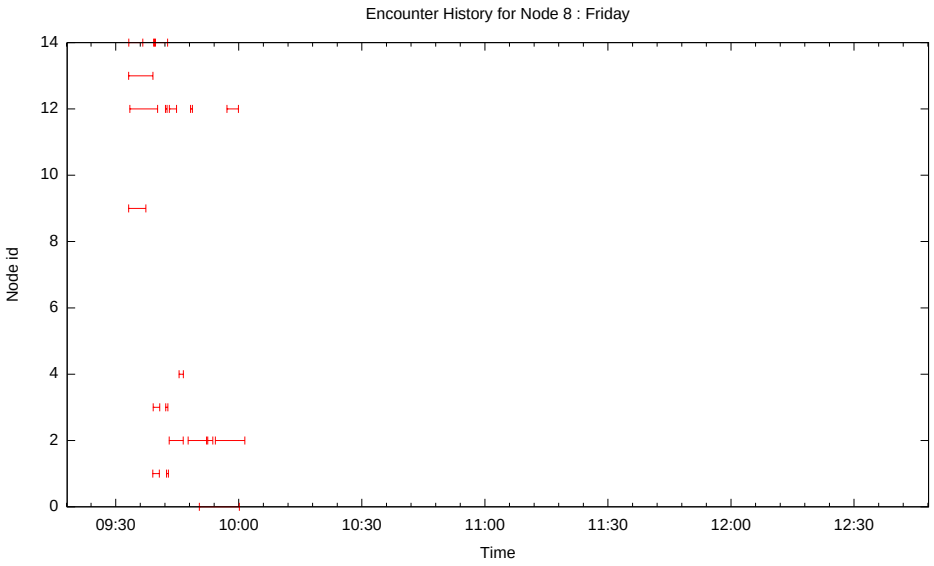

Figure A.25: Message History for node 8

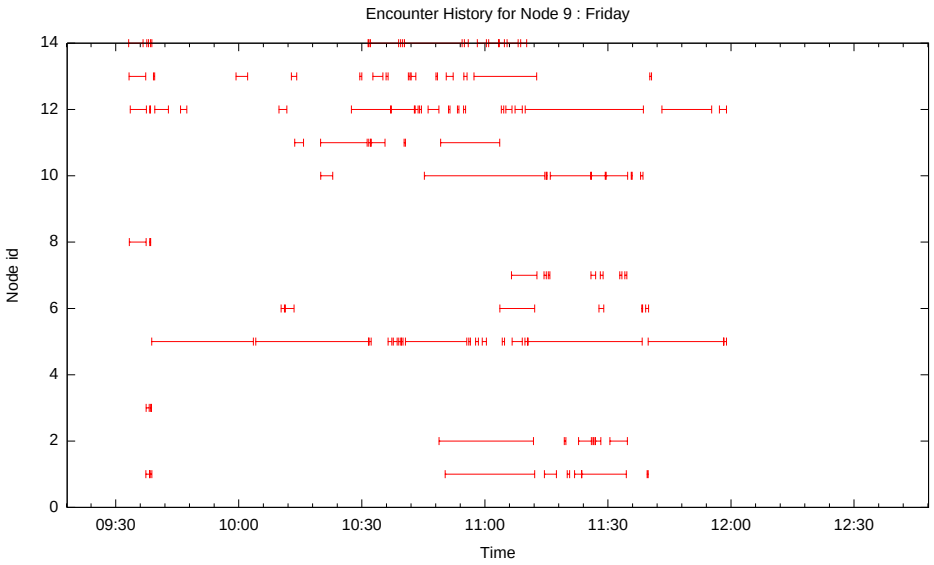

Figure A.26: Message History for node 9

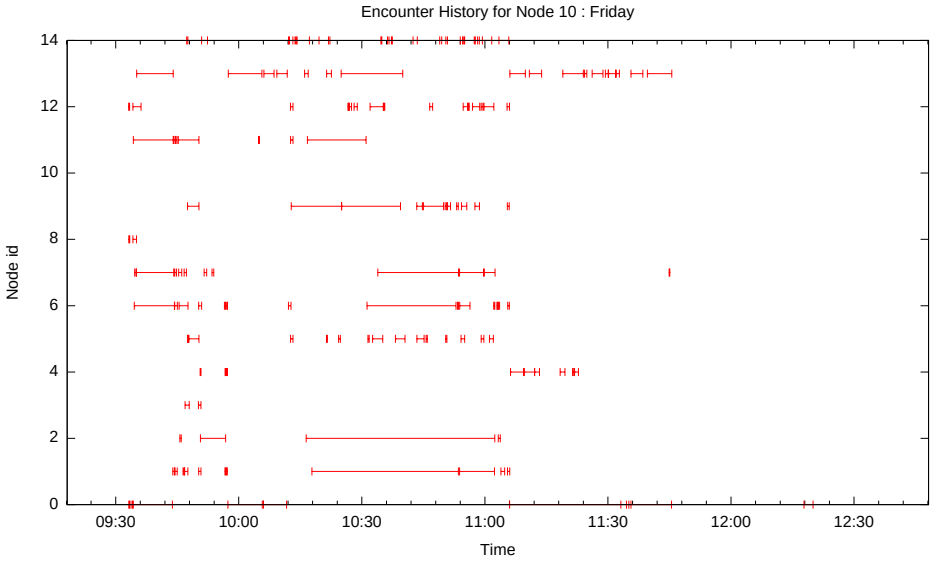

Figure A.27: Message History for node 10

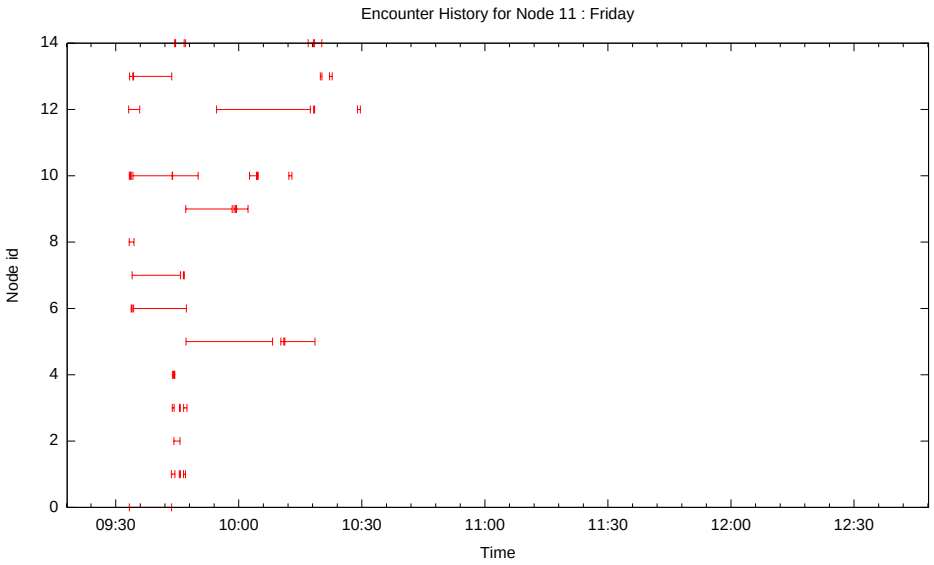

Figure A.28: Message History for node 11

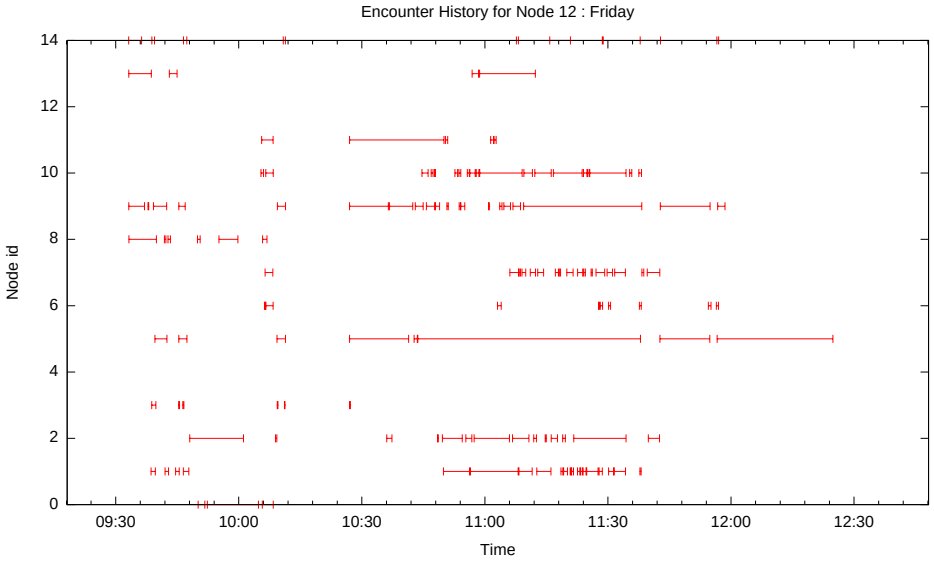

Figure A.29: Message History for node 12

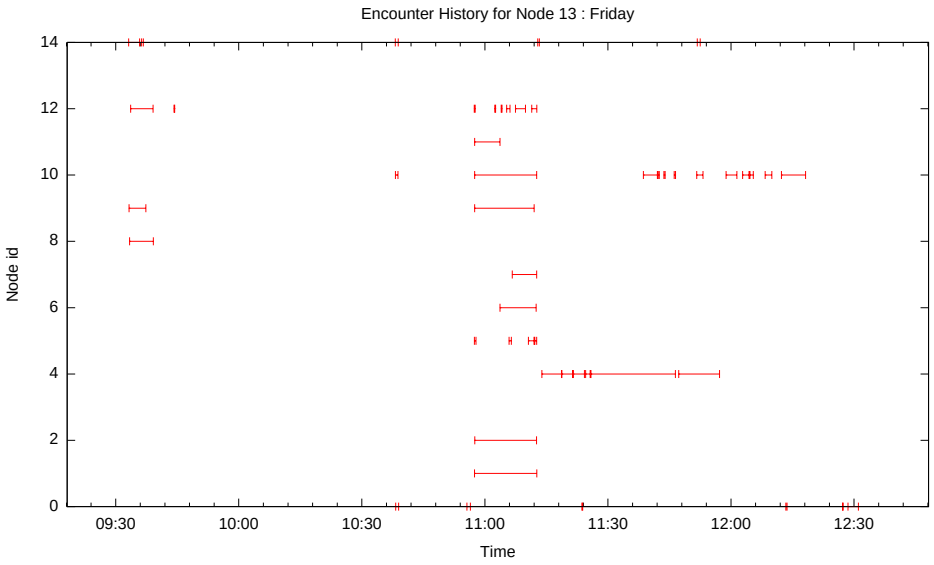

Figure A.30: Message History for node 13

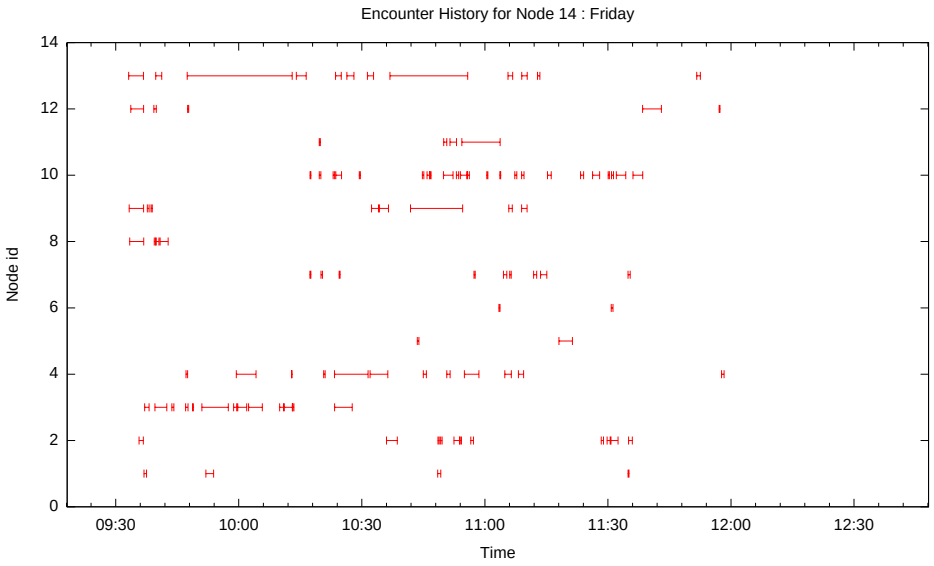

Figure A.31: Message History for node 14