



## **Terms and Conditions of Use of Digitised Theses from Trinity College Library Dublin**

### **Copyright statement**

All material supplied by Trinity College Library is protected by copyright (under the Copyright and Related Rights Act, 2000 as amended) and other relevant Intellectual Property Rights. By accessing and using a Digitised Thesis from Trinity College Library you acknowledge that all Intellectual Property Rights in any Works supplied are the sole and exclusive property of the copyright and/or other IPR holder. Specific copyright holders may not be explicitly identified. Use of materials from other sources within a thesis should not be construed as a claim over them.

A non-exclusive, non-transferable licence is hereby granted to those using or reproducing, in whole or in part, the material for valid purposes, providing the copyright owners are acknowledged using the normal conventions. Where specific permission to use material is required, this is identified and such permission must be sought from the copyright holder or agency cited.

### **Liability statement**

By using a Digitised Thesis, I accept that Trinity College Dublin bears no legal responsibility for the accuracy, legality or comprehensiveness of materials contained within the thesis, and that Trinity College Dublin accepts no liability for indirect, consequential, or incidental, damages or losses arising from use of the thesis for whatever reason. Information located in a thesis may be subject to specific use constraints, details of which may not be explicitly described. It is the responsibility of potential and actual users to be aware of such constraints and to abide by them. By making use of material from a digitised thesis, you accept these copyright and disclaimer provisions. Where it is brought to the attention of Trinity College Library that there may be a breach of copyright or other restraint, it is the policy to withdraw or take down access to a thesis while the issue is being resolved.

### **Access Agreement**

By using a Digitised Thesis from Trinity College Library you are bound by the following Terms & Conditions. Please read them carefully.

I have read and I understand the following statement: All material supplied via a Digitised Thesis from Trinity College Library is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of a thesis is not permitted, except that material may be duplicated by you for your research use or for educational purposes in electronic or print form providing the copyright owners are acknowledged using the normal conventions. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

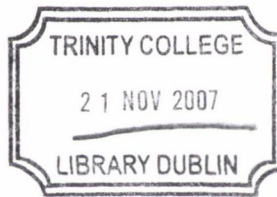
# **Parasitic Routing: Using Social Network Analysis for Routing in Disconnected Delay-Tolerant MANETs**

**Elizabeth M. Daly**

Distributed Systems Group,  
Department of Computer Science,  
Trinity College, University of Dublin.

A thesis submitted to the University of Dublin, Trinity College  
in fulfillment of the requirements for the degree of  
Doctor of Philosophy (Computer Science)

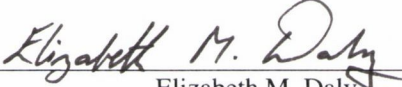
April 2007



THESIS  
8240

## Declaration

I, the undersigned, declare that this work has not previously been submitted to this or any other University, and that unless otherwise stated, it is entirely my own work. I agree that Trinity College Library may lend or copy this thesis upon request.

  
Elizabeth M. Daly

Dated: July 25, 2007



# Acknowledgements

I was working on the proof of one of my poems all the morning, and took out a comma. In the afternoon I put it back again.

~ Oscar Wilde

First, I would like to thank my supervisor Dr. Mads Haahr for his support and enthusiasm. Thanks for the insightful thoughts and guidance that have been the source of inspiration and encouragement over the course of my PhD. I would also like to thank the DSG academics, for asking the difficult questions. The search for answers lead me down paths that proved challenging, and yet fascinating. Thanks to IRCSET for their sponsorship of this research.

I would like to take the opportunity to express my gratitude to the Dartmouth Crawdad project for establishing a forum for researchers to share invaluable experimental data. I would, particularly, like to thank the contributors to the projects whose research is used in the evaluation of this thesis. From the Hagggle Project: James Scott, Richard Gass, Jon Crowcroft, Pan Hui, Christophe Diot and Augustin Chaintreau. From the MIT Reality Mining Project: Nathan Eagle and Alex Pentland. And finally, from the UMassDieselNet Project: John Burgess and Brian Levine.

I would like to thank everyone in DSG. I believe I will never again meet such an interesting group of people. Many thanks to all of you who have encouraged me, entertained me and made DSG such a wonderful place to work. Thank you so much to Tim for providing much needed last minute computing resources.

Thanks to my dear friends Anne-Marie, Elaine, Ivana and Elizabeth. To Anne-Marie, for letting me bend her ear and for always being on my side, no matter what. To Elaine, for always making me forget my worries and to see the positive side of any situation. To Ivana, for never

letting me take myself too seriously and for teaching me how to use commas. To Elizabeth, my long distance cheering section, even from an ocean away you have managed to be there for me.

I don't have the words to express my gratitude to my family. To my parents, you have supported me, loved me and most importantly believed in me. You have both set an example to me, that through hard work, dedication and above all enjoyment in what you do, there are no bounds to what can be achieved. To Paul, Nicholas and Alice, each of you have been rooting for me and shown your support in your own special ways.

And finally, to Dominik, you have been through this journey with me, rejoicing with me during the good, consoling me during the bad. Your friendship, love and unconditional support has been the source of my strength during these last few months. I know I will get the opportunity to return the favour.

**Elizabeth M. Daly**

*University of Dublin, Trinity College*

*April 2007*

# Contents

<b>Acknowledgements</b>	<b>v</b>
<b>List of Figures</b>	<b>xii</b>
<b>List of Tables</b>	<b>xv</b>
<b>Abstract</b>	<b>xvi</b>
<b>Chapter 1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 MANETs . . . . .	2
1.3 Delay-Tolerant Networks (DTN) . . . . .	3
1.4 Disconnected Delay-Tolerant MANETs (DDTM) . . . . .	5
1.5 Thesis Contributions . . . . .	6
1.6 Thesis Roadmap . . . . .	7
<b>Chapter 2 The Challenges of Disconnected Delay-Tolerant MANETs</b>	<b>8</b>
2.1 Wireless Communication . . . . .	9
2.1.1 Little or No Infrastructure . . . . .	9
2.1.2 Variable Node Population . . . . .	10
2.1.3 Limited Bandwidth . . . . .	11
2.1.4 High Loss Rate . . . . .	11
2.1.5 Presence of Asymmetric Links . . . . .	12
2.2 Mobility . . . . .	13
2.2.1 Limited Control Over Node Movements . . . . .	13



2.2.2	Limited Knowledge of Future Node Movements . . . . .	14
2.2.3	Disconnected Network Graph . . . . .	15
2.2.4	Highly Dynamic Topology . . . . .	15
2.2.5	Limited Topology Information . . . . .	16
2.2.6	Variable Link Duration . . . . .	16
2.3	Portability . . . . .	17
2.3.1	Limited Battery Power . . . . .	17
2.3.2	Limited Memory . . . . .	18
2.3.3	Limited Processing Power . . . . .	19
2.4	Second Tier Challenges . . . . .	19
2.4.1	Low Delivery Reliability . . . . .	20
2.4.2	High End-to-End Latency . . . . .	21
2.4.3	Poor Quality of Service Support . . . . .	21
2.5	Summary . . . . .	22
<b>Chapter 3 State of the Art</b>		<b>23</b>
3.1	Deterministic Approaches . . . . .	26
3.1.1	Space-Time Routing . . . . .	26
3.1.2	Tree Approach . . . . .	29
3.1.3	Modified Shortest Path Approaches . . . . .	31
3.1.4	Conclusion . . . . .	33
3.2	Epidemic-Based Approaches . . . . .	33
3.2.1	Classic Approach . . . . .	34
3.2.2	Limited Number of Copies . . . . .	36
3.2.3	2-Hop Approaches . . . . .	36
3.2.4	Conclusion . . . . .	38
3.3	History or Prediction-Based Approaches . . . . .	39
3.3.1	Contact-Based . . . . .	40
3.3.2	Location-Based . . . . .	43
3.3.3	Utility-Based . . . . .	46
3.3.4	Conclusion . . . . .	50

3.4	Approaches Based on Movement Control . . . . .	51
3.4.1	Conclusion . . . . .	53
3.5	Coding-Based Approaches . . . . .	53
3.5.1	Erasure-Coding . . . . .	53
3.5.2	Network Coding . . . . .	55
3.6	Discussion of DDTM routing Solutions . . . . .	56
<b>Chapter 4 Social Networks for Information Flow</b>		<b>58</b>
4.1	Network Centrality for Information Flow . . . . .	59
4.2	Strong Ties for Information Flow . . . . .	62
4.3	Tie Predictors . . . . .	65
4.4	Applications of SNA . . . . .	67
4.5	Conclusion . . . . .	68
<b>Chapter 5 Parasitic Routing for DDTMs</b>		<b>69</b>
5.1	Requirements . . . . .	69
5.2	Using Social Network Analysis for Routing in DDTMs . . . . .	72
5.2.1	Measuring Centrality for DDTMs . . . . .	72
5.2.2	Measuring Tie Strength for DDTMs . . . . .	73
5.2.3	Measuring Similarity for DDTMs . . . . .	76
5.2.4	Node Utility Calculation . . . . .	77
5.2.5	Social Network Analysis and Requirements . . . . .	78
5.3	Parasitic Routing . . . . .	79
5.3.1	Protocol Architectural Overview . . . . .	79
5.3.2	Neighbour Discovery Service . . . . .	80
5.3.3	Social Component . . . . .	81
5.3.4	Queue Manager . . . . .	82
5.3.5	Parasitic Routing Component . . . . .	84
5.3.5.1	Message Scheduling . . . . .	88
5.3.5.2	Replication . . . . .	89
5.4	Summary . . . . .	90

<b>Chapter 6 Implementation</b>	<b>92</b>
6.1 Trace-Based Simulator . . . . .	92
6.2 Architectural Overview . . . . .	93
6.3 Parasitic Routing Layer API . . . . .	96
6.4 Inter-Component Communication . . . . .	96
6.5 Neighbour Discovery Service . . . . .	97
6.6 Social Component . . . . .	98
6.7 Queue Manager . . . . .	100
6.8 Parasitic Routing Component . . . . .	102
6.8.1 Event Handling Algorithms . . . . .	102
6.8.1.1 Message Receipt Event . . . . .	103
6.8.1.2 Neighbour Discovery Event . . . . .	103
6.8.1.3 Receive Contact List Event . . . . .	104
6.8.1.4 Receive Summary Vector Event . . . . .	105
6.9 Protocol Message Format . . . . .	106
6.10 Summary . . . . .	108
<b>Chapter 7 Evaluation</b>	<b>109</b>
7.1 Evaluation Strategy . . . . .	109
7.2 Case Study: Huggle Data Set . . . . .	116
7.2.1 Intel Data Set . . . . .	117
7.2.1.1 Baseline . . . . .	117
7.2.1.2 Limited Bandwidth . . . . .	121
7.2.1.3 Limited Capacity . . . . .	122
7.2.2 Cambridge Data Set . . . . .	123
7.2.2.1 Baseline . . . . .	123
7.2.2.2 Limited Bandwidth . . . . .	126
7.2.2.3 Limited Capacity . . . . .	127
7.2.3 Infocom Data Set . . . . .	127
7.2.3.1 Baseline Performance Evaluation . . . . .	128
7.2.3.2 Limited Bandwidth . . . . .	130

7.2.3.3	Limited Capacity . . . . .	131
7.2.4	Discussion . . . . .	131
7.3	Case Study: MIT Reality Mining Data Set . . . . .	132
7.3.1	Baseline . . . . .	134
7.3.2	Limited Bandwidth . . . . .	138
7.3.3	Limited Capacity . . . . .	139
7.3.4	Discussion . . . . .	139
7.4	Case Study: UMassDieselNet . . . . .	140
7.4.1	Baseline . . . . .	141
7.4.2	Limited Bandwidth . . . . .	143
7.4.3	Limited Capacity . . . . .	144
7.4.4	Discussion . . . . .	145
7.5	Conclusion . . . . .	145
<b>Chapter 8</b>	<b>Conclusion</b>	<b>147</b>
8.1	Achievements . . . . .	147
8.2	Future Work . . . . .	149
8.3	Concluding Remarks . . . . .	150
<b>Appendix A: P</b>	<b>RoPHET Parameter Tuning</b>	<b>151</b>
<b>Bibliography</b>		<b>154</b>

# List of Figures

1.1	Multi-hop Routing in MANETs . . . . .	3
1.2	DTN Architecture using a DTN gateway to interconnect regions running dis- similar protocols . . . . .	4
2.1	Laptop technology improvements during the past decade. The logarithmic scale shows multiples of the technology's improvement since 1990 [108] . . . . .	17
2.2	Tiered Challenges . . . . .	19
3.1	Space-time routing (derived from [89]) . . . . .	27
3.2	Message Path [53] . . . . .	29
3.3	MoVe vector calculation derived from [70] . . . . .	44
3.4	Network Encoding [120] . . . . .	56
3.5	Disconnected Clusters . . . . .	57
4.1	Bank Wiring Room network sociocentric and egocentric betweenness [85] . . . . .	61
5.1	Parasitic Routing Component Architecture . . . . .	80
5.2	Social Component Data Structure . . . . .	82
5.3	Control Flow During Initialisation . . . . .	84
5.4	Parasitic Routing Protocol Information Flow . . . . .	85
5.5	Response to Message Receive Event . . . . .	86
5.6	Response to Neighbour Discovery Event . . . . .	87
5.7	Response to Contact Exchange Event . . . . .	87
5.8	Response to Summary Vector Event . . . . .	88
6.1	OSI Model . . . . .	94

6.2	Parasitic Routing in an IP Stack . . . . .	94
6.3	Architectural Component Overview . . . . .	95
6.4	Neighbour Discovery Service Class Diagram . . . . .	97
6.5	Social Component Class Diagram . . . . .	99
6.6	Queue Manager Class Diagram . . . . .	100
6.7	Parasitic Routing Class Diagram . . . . .	102
6.8	Protocol Message Format . . . . .	107
7.1	Baseline Protocol Performance for Intel Data . . . . .	117
7.2	Baseline Control Data Overhead for Intel Data . . . . .	120
7.3	Effect of Limited Bandwidth for Intel Data . . . . .	121
7.4	Effect of Buffer Capacity for Intel Data . . . . .	122
7.5	Baseline Protocol Performance for Cambridge Data . . . . .	123
7.6	Baseline Control Data Overhead for Cambridge Data . . . . .	126
7.7	Effect of Limited Bandwidth for Cambridge Data . . . . .	126
7.8	Effect of Buffer Capacity for Cambridge Data . . . . .	127
7.9	Baseline Protocol Performance for Infocom Data . . . . .	128
7.10	Baseline Control Data Overhead for Infocom Data . . . . .	130
7.11	Limited Bandwidth for Infocom Data . . . . .	131
7.12	Effect of Buffer Capacity for Infocom Data . . . . .	132
7.13	Betweenness Distribution . . . . .	133
7.14	Friendship network Eagle and Pentland [33] . . . . .	134
7.15	Baseline Protocol Performance for MIT Data . . . . .	135
7.16	Baseline Control Data Overhead for MIT Data . . . . .	137
7.17	Percentage of message delivered between a subset of nodes increasing based on increased Betweenness . . . . .	138
7.18	Effect of Limited Bandwidth for MIT Data . . . . .	139
7.19	Effect of Buffer Capacity for MIT Data . . . . .	140
7.20	Baseline Protocol Performance for UMassDieselNet Data . . . . .	141
7.21	Baseline Control Data Overhead for UMassDieselNet Data . . . . .	144
7.22	Limited Bandwidth Effect of Message Size for UMassDieselNet Data . . . . .	144

7.23 Effect of Buffer Capacity for UMassDieselNet Data . . . . . 145

# List of Tables

5.1	Overview of the Requirements . . . . .	70
5.2	Parasitic Utility Calculation . . . . .	77
5.3	Overview of the Requirements . . . . .	78
5.4	Requirements Satisfied by Social Network Analysis (SNA), Protocol Design (PD) and Parasitic Routing (PR) . . . . .	91
6.1	Example Trace File Extract . . . . .	93
7.1	Experimental Parameters . . . . .	115
7.2	Haggle Data Set . . . . .	116
7.3	Baseline Protocol Performance Statistics for Intel Data . . . . .	119
7.4	Baseline Protocol Performance Statistics for Cambridge Data Statistics . . . . .	125
7.5	Baseline Protocol Performance Statistics for Infocom Data . . . . .	129
7.6	Baseline Protocol Performance Statistics for MIT Data Statistics . . . . .	136
7.7	Baseline Protocol Performance Statistics for UMassDieselNet Data . . . . .	143
8.1	Intel Baseline PROPHET Tuning . . . . .	151
8.2	Cambridge Baseline PROPHET Tuning . . . . .	151
8.3	InfoCom Baseline PROPHET Tuning . . . . .	152
8.4	MIT Baseline PROPHET Tuning . . . . .	152
8.5	DeiselNet Baseline PROPHET Tuning . . . . .	152



# Abstract

Message delivery in sparse Mobile Ad hoc Networks (MANETs) is difficult due to the fact that the network graph is rarely (if ever) connected. As a consequence, a full end-to-end path between nodes may not exist at any given point in time. However, an end-to-end connectivity graph that is time-varying may exist to forward messages. The disconnected nature and the lack of end-to-end connectivity between nodes means that the communication must be delay-tolerant. In order to support message delivery in such disconnected delay-tolerant MANETs (DDTMs), each node acts a router, forwarding messages to other nodes. Messages are buffered using a store-and-forward mechanism where the data is physically carried through the time-varying network graph.

The disconnected network graph and dynamics of the network result in limited availability of information regarding the connectivity of the underlying network. The challenge is to identify potential paths without global knowledge of the present state of the graph and without knowledge of its future state. Consequently, the selection of a next hop as a potential carrier for messages is not an easy one.

This thesis proposes using social network analysis techniques in order to determine the underlying social structure of the network. This is motivated by the fact that information in regards to underlying social structure is less dynamic and time dependent than evaluating temporal connections. Using this social structure, nodes that are useful in connecting otherwise disconnected nodes may be identified. Consequently, if no information is available regarding the destination node, then central nodes may be used to route information to nodes that have a higher probability of encountering information about the destination node. Additionally, the social structure may be used to identify nodes that belong to similar groups as the destination node. Finally, social network analysis techniques may be used to evaluate social tie strength, which represents the strength of a connection between two nodes. Strong ties have a higher

probability of being available for information dissemination.

The thesis presents Parasitic Routing which uses a novel and practical routing metric to provide efficient message delivery in disconnected delay-tolerant MANETs. This metric is based on social analysis of a node's past interactions and consists of three locally evaluated components: a node's 'betweenness' centrality (calculated using ego networks), a node's social 'similarity' to the destination node and the strength of a node's relationship to the destination node. Parasitic Routing makes no assumption of prior knowledge of the nodes contained in the network or of the area covered by the network and requires no control over node movements. A simulation implementation of Parasitic Routing is evaluated using real world trace data capturing encounters between devices. The trace data consists of varying node populations and experimental time frames and include Bluetooth devices carried by people and 802.11 devices carried on buses enabling inter-vehicle communication. The simulation results show that in nearly all of the scenarios, Parasitic Routing achieves delivery performance close to the theoretical max (achievable by flooding) and that this performance is achieved with significantly less overhead than flooding. Additionally, when storage capacity is limited on nodes, Parasitic Routing outperforms flooding. The evaluation also shows that Parasitic Routing achieves superior delivery performance compared to the most influential state-of-the-art protocol for routing in DDTMs.



# Chapter 1

## Introduction

This thesis addresses the problem of providing communication in Mobile Ad hoc Networks (MANETs) that are disconnected. This is achieved through the design of a new routing protocol called Parasitic Routing, which relies on the use of a node's history of encounters in order to evaluate the utility of a node as a potential carrier of messages. In this way, data is physically carried from source to destination. The utility of a node is based on social analysis of a node's past interactions and consists of a node's locally determined centrality and the strength of the node's relationship with the destination node. Parasitic Routing makes no assumption of prior knowledge of the nodes contained in the network or of the area covered by the network, and it requires no control over node movements.

This introductory chapter discusses the motivation behind the Parasitic Routing protocol. The chapter introduces mobile ad hoc networks in general and delay-tolerant networks in particular in order to position the work in relation to these areas. Next, the thesis contributions are summarised before outlining a roadmap for the remainder of the thesis.

### 1.1 Motivation

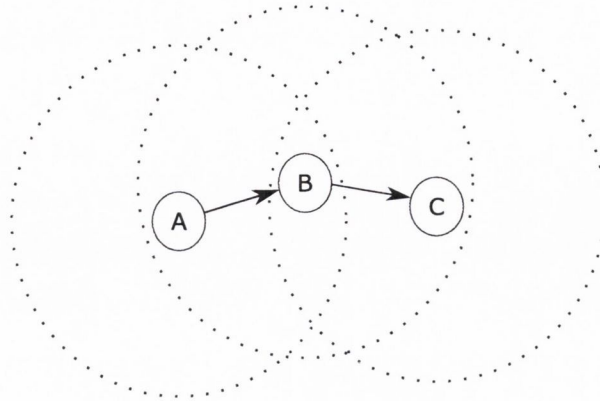
The increase of mobile computing and communication devices, such as cell phones, laptops and handheld digital devices such as Personal Digital Assistants (PDAs), means that wireless networks are increasingly the most convenient solution for interconnection in many usage scenarios. Since the early 2000s mobile devices are getting smaller, cheaper and more convenient to carry, with the ability to run applications and connect to network services. Currently, most

of the connections among wireless devices are achieved through fixed infrastructure service providers or private networks. For example, since the 1980s mobile phones have been connected by cellular networks, and the connection of laptops to the Internet via wireless access points has grown rapidly in popularity in the early 2000s [73]. Current developments, such as 3G and 4G phones, show little signs of change in this trend. While infrastructure-based networks provide an effective mechanism for mobile devices to get network service, setting up the necessary infrastructure can be time consuming and incurs potentially high costs. There are situations where networking connections are not available in a given geographic area, and providing connectivity and network services in these situations becomes a real challenge. Examples range from wildlife tracking and habitat monitoring sensor networks, military networks, inter-vehicle communication, disaster response networks, inter-planetary networks to nomadic community networks. For this reason, alternative ways to deliver services in disconnected environments have been emerging. Two such areas include MANETs which arose in the 1990s, and more recently delay-tolerant networks (DTNs) which were first introduced in 2001.

## 1.2 MANETs

MANETs were traditionally developed for tactical networks related to improving battlefield communication where the network cannot rely on access to a fixed communication infrastructure. A MANET is a dynamic wireless network with or without fixed infrastructure. Nodes may move freely and organise themselves arbitrarily; thus the network's wireless topology may change rapidly and unpredictably [29]. Each node may communicate directly with any node within transmission range. Communication beyond that range is achieved by using intermediate nodes to relay messages hop by hop. Such a route may include multiple hops, and therefore the resulting network may or may not be a multi-hop network. MANETs do not depend on centralised administration, rather each node acts as an independent router and typically also as an application node, generating and receiving application data. As such, network management is distributed across the nodes. Figure 1.1 shows an example of multi-hop routing. In the scenario, node A is out of direct communication range with node C, but can communicate with node C by using node B as an intermediary.

MANET routing protocols can be classified into two main categories: proactive routing pro-



**Fig. 1.1:** Multi-hop Routing in MANETs

protocols and reactive routing protocols. Proactive routing protocols attempt to maintain routing information for every pair of network nodes by actively propagating route updates. Reactive protocols establish a route to a destination only when needed. The source node initiates a route discovery process when the route is required, and once a route has been established it is maintained until either the destination becomes inaccessible or until the route is no longer used. Both categories of MANET routing protocols assume the existence of a full (possibly, multi-hop) route from source to destination at the time of sending, and if one is not available routing fails.

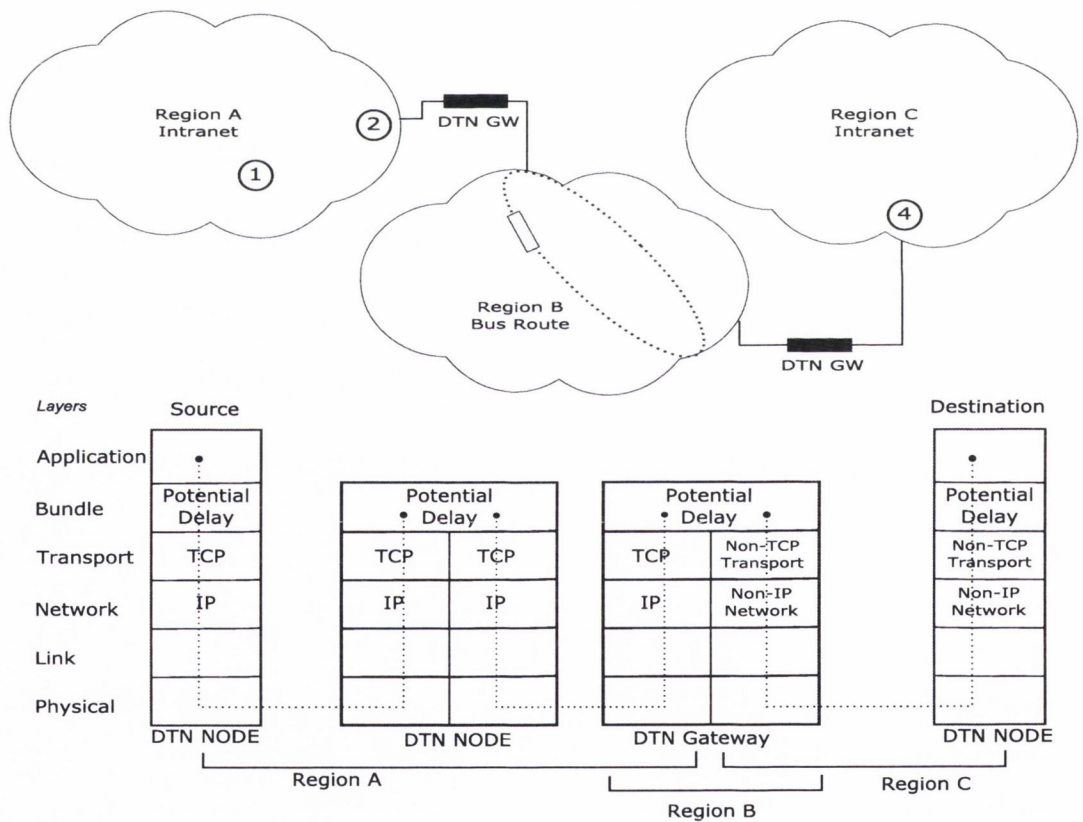
The assumptions of MANETs of complete routes available at the time of sending in the network makes them unsuitable for environments where disconnections are frequent and potentially long-term.

### 1.3 Delay-Tolerant Networks (DTN)

A delay-tolerant network (DTN) provides ‘interoperable communications with and among challenged environments’ [36]. A challenged network is defined as a network that has one or more of the following characteristics: high end-to-end path latency; end-to-end disconnection meaning a path between a node pair may never exist; limited resources or limited life expectancy either due to lack of battery power, such as in sensor networks, or node damage as may occur in battlefield deployments. Such networks may never have an end-to-end path from source to destination at a given time. The routing problem in DTNs can be described as ‘where mes-

sages are to be moved end-to-end across a connectivity graph that is time-varying but whose dynamics may be known in advance' [62].

DTNs stemmed from research to develop an interplanetary Internet (IPN). The Delay-Tolerant Network Research Group (DTNRRG) have proposed an architecture to support messaging in delay-tolerant applications. The architecture presented by Kevin Fall [36] consists of an overlay, called the bundle layer. A bundle is defined as a number of messages to be delivered together. DTN nodes implement the bundle layer which forms an overlay that employs persistent storage to overcome network interruptions. The bundle layer stores and forwards bundles between DTN nodes. The bundle layer is situated below the application layer and above the transport layer, thus allowing environment-specific underlying protocols.



**Fig. 1.2:** DTN Architecture using a DTN gateway to interconnect regions running dissimilar protocols

Figure 1.2 shows an example of a DTN configuration consisting of three regions. Region A is an intranet which is running TCP/IP and region B consists of a bus carrying a DTN Gateway

node (DTN GW). Region C is running a non-TCP/IP environment specific transport protocol. In order for the DTN Gateway to carry bundles between region A and region C, the DTN GW of region B must run multiple lower-layer protocols to allow the gateway to span two regions using different lower-layer protocols.

The key contribution of the architecture is the bundle layer, which facilitates hop-by-hop reliability and retransmission. It also implements a naming scheme that uses late binding, which means that the name is not mapped to an address until topologically near the target of communication. Each DTN node has a two-part name, consisting of a region ID which is known among all regions of the DTN and an entity ID which has a regional scope. Routing between regions is based only on region IDs, which are bound to their corresponding addresses throughout the DTN. Routing within regions is based only on entity IDs, which are bound to their corresponding address only within that region.

The bundle layer provides an overlay to hide disconnections and delays from the application layer and provide transparent communication between different regions. However, the problem of how to route within, and among the regions is still an open issue.

## **1.4 Disconnected Delay-Tolerant MANETs (DDTM)**

This thesis is concerned with supporting communication in disconnected MANETs with such a sparse population of nodes and so little (or no) fixed infrastructure that the network graph is rarely, if ever, connected. The networks considered are autonomous and do not depend on established infrastructure. Each node acts a router, forwarding messages to other nodes. Due to the disconnected network graph, a full end-to-end path to nodes may not exist at any given point in time. However, an end-to-end connectivity graph that is time-varying may exist to forward messages. The challenge for routing protocols in this type of environment is to achieve the best delivery ratio with the available information about the network. Messages are buffered using a store-and-forward mechanism, where the data is physically carried through the time-varying network graph. These challenged environments are characterised by their disconnected nature where continuous end-to-end connectivity cannot be assumed. As a result, they suffer from long or variable delay times, asymmetric data rates and high error rates. The disconnected nature and the lack of end-to-end connectivity between nodes means that the communication must



be delay-tolerant. We will refer to such a networks as disconnected delay-tolerant MANETs (DDTMs).

Routing protocols designed for fixed networks cannot be used in DDTMs as they depend on the assumptions of continuous bidirectional end-to-end paths, short round-trip times and low error rates. Routing protocols designed for MANETs do not work properly in such networks, since under these protocols, when packets arrive and no existing end-to-end paths for their destination can be found, these packets are simply dropped. The DTN architecture presented in [36] accounts for end-to-end disconnections and delays. However, the DTN architecture is mainly concerned with constructing an overlay connecting nodes that may be delay-tolerant and is not concerned with routing. Routing in delay-tolerant networks is still an open issue.

## 1.5 Thesis Contributions

Message delivery in sparse Mobile Ad hoc Networks (MANETs) is difficult due to the fact that the network graph is rarely (if ever) connected. As a consequence of the disconnected network graph, a full end-to-end path to nodes may not exist at any given point in time. However, an end-to-end connectivity graph that is time-varying may exist to forward messages. The challenge is to identify potential paths without global knowledge of the present state of the graph and without knowledge of its future state. Consequently, the selection of a next hop as a potential carrier for messages is not an easy one.

This thesis proposes using social network analysis techniques to identify such paths by determining the underlying social structure of the network. This is motivated by the fact that information in regards to underlying social structure is less dynamic and time dependent than evaluating temporal connections. Using this social structure, nodes that are useful in connecting otherwise disconnected nodes may be identified. Consequently, if no information is available regarding the destination node, then central nodes may be used to route information to nodes that have a higher probability of encountering information about the destination node. Additionally, the social structure may be used to identify nodes that belong to similar groups as the destination node. Finally, social network analysis techniques can be exploited to evaluate social tie strength, which represents the strength of a connection between two nodes. Strong ties have a higher probability of being available for information dissemination.

The work presented in this thesis focuses primarily on the investigation of techniques based on social network analysis in the design and implementation of a routing protocol for DDTMs. The contributions can be summarised as two primary and one secondary contributions to the state of the art in the area of DDTM routing:

- A set of metrics for capturing the underlying time-varying network structure in disconnected delay-tolerant MANETs where global topology information is unavailable. These metrics are derived from a translation of social network analysis techniques for capturing the behaviour of nodes.
- A protocol, called Parasitic Routing, that supports message delivery in DDTMs by utilising these metrics to provide a complete solution for delay-tolerant message delivery in disconnected networks. An implementation and evaluation validates the Parasitic Routing protocol using real world trace data capturing encounters between devices.
- A secondary contribution is an analysis of the challenges associated with the specific mobile computing environment constituted by disconnected delay-tolerant MANETs.

## 1.6 Thesis Roadmap

The remainder of this thesis follows the following structure:

**Chapter 2** defines the challenges associated with disconnected delay-tolerant MANETs and an analysis of the mobile computing environment.

**Chapter 3** presents the state of the art in routing in disconnected networks.

**Chapter 4** presents social network analysis techniques and explains how they may be applied to delay-tolerant networks.

**Chapter 5** presents the Parasitic Routing protocol design.

**Chapter 6** presents an simulation implementation of the design of the Parasitic Routing protocol from the previous chapter.

**Chapter 7** evaluates the work described in the thesis using real-world trace data.

**Chapter 8** concludes and discusses future work.

# **Chapter 2**

## **The Challenges of Disconnected Delay-Tolerant MANETs**

The challenges associated with mobile computing are not new. A comprehensive overview was given as far back as 1994 by Forman et al. [40]. This paper identified the challenges of mobile computing which are classified into three main categories each stemming from an inherent property of mobile computing. These categories are wireless communication, mobility and portability. We build on this format and divide the challenges of DDTMs into the same three groups. The issues identified in wireless communication such as low bandwidth, disconnections and high bandwidth variability are problematic and further exacerbated in DDTMs by little or no infrastructure, variable node population and lossy links. DDTMs additionally face challenges of mobility without the presence of servers, as originally contemplated by Forman et al., where each node must act as a router. Mobility is frequent and uncontrolled resulting in a highly dynamic topology and disconnected network graph. However, the challenges of portability remain the same, where battery power, memory and processing power are limited.

In this chapter the main underlying challenges related to DDTMs are divided into wireless communication, mobility and portability. These challenges combined result in aggregated challenges. As such we encompass these issues identified by Forman et al. and classify them as first tier challenges. Forman's classifications are then extended to include the challenges that occur due to the collective first tier challenges, by adding a classification of second tier challenges.

This chapter discusses these challenges. Section 2.1 discusses the issues relating to communication in the wireless environment. Section 2.2 examines the consequences of mobility.

Section 2.3 investigates the pressures that portability places on the network and section 2.4 presents the combined issues that are identified as second tier challenges. Finally, the chapter is summarised.

## **2.1 Wireless Communication**

Communication in DDTMs is primarily wireless. It is well known that the wireless communication environment presents many more challenges than a wired one. Forman et al. observe that this is due to the surrounding environment which interacts with the signal which may result in blocking of signal paths and introduces noise and echoes [40]. As a result, bandwidth is limited and suffers from lossy links. Additionally, the varying capabilities of nodes result in asymmetric links.

The node movement also affects the communication and means that disconnections are the norm rather than the exception. As a result, the traditional solutions trying to prevent disconnections such as choosing alternative links, performing route recovery or changing transmission range are unsuitable for these environments. Forman et al. note that in such circumstances it is important for a mobile device to operate as a stand-alone device, and utilise links only when they become available. As such, in order to exploit connections to their full potential, the problems associated with wireless communication must be addressed.

### **2.1.1 Little or No Infrastructure**

As discussed in section 1.2 a MANET is a dynamic wireless network with or without fixed infrastructure. MANETs have traditionally been based on the cellular networks, such as GSM, and relied on good infrastructure support, in which mobile devices communicate with access points or base stations connected to fixed network infrastructure. Though infrastructure should be used if available, it is an unrealistic assumption for all mobile environments. One example where fixed infrastructure is unavailable is disaster relief. Nodes are placed in the network in response to an emergency and must self-organise without the presence of infrastructure and centralised coordination.

The lack of infrastructure means that control and management of the network is distributed

across the nodes. Communication to out of range nodes is accomplished through intermediate nodes. MANET routing protocols such as Ad hoc On Demand Distance Vector (AODV) [98], Dynamic Source Routing (DSR) [64] and Destination-Sequenced Distance-Vector (DSDV) [99] function without the presence of infrastructure. These protocols have proved effective in infrastructureless environments when node mobility is limited and node population is limited. However, performance varies greatly in the presence of high mobility and increasing node population [21, 114]. Additionally, the above protocols assume the existence of a full path from source to destination at the time of sending. However, this may not be the case in DDTMs, resulting in a failure to deliver data.

What can be learned from these protocols is that reactive routing protocols outperform proactive routing protocols in high mobility scenarios [63]. This is due to reactive protocols focusing only on needed connectivity and not complete connectivity. A protocol suitable for highly dynamic environments should only focus on connectivity as it is needed.

### **2.1.2 Variable Node Population**

The disconnected nature of these networks along with the limited topology information available, means that the node population is unknown. Chlamtac et al. point out that the majority of popular network management algorithms were designed for either fixed or relatively small wireless networks [27]. However, there are a large number of MANET environments that involve tens of thousands of nodes such as in sensor networks. As such, handling of varying node population is critical to the successful deployment of these networks.

Increased node population may result in excessive routing overhead. Large populations of stationary nodes may be overcome using hierarchical routing, where nodes are divided into clusters with one node communicating to other nodes on behalf of the cluster. However, in mobile environments where nodes come and go, hierarchies must be continuously updated rendering them inefficient. Location information may be used to reduce routing information propagation. For example, Location-Aided Routing (LAR) first searches a limited area for a route before resorting to searching the entire network [69].

### **2.1.3 Limited Bandwidth**

The wireless bandwidth available for mobile devices is much more limited than for stationary nodes. Toh states that wireless links have significantly lower capacity than wired links, due to effects of multiple access, multi-path fading, noise, and signal interference [113]. This means that wireless links may have a throughput less than the radio's maximum transmission capacity. This is not just due to the communication capabilities of the node but also due to the problem of finite battery power which is further discussed in section 2.3.1. Chlamtac et al. state that experiments have shown that networking activities consume approximately 10% of overall power consumption in laptops and as much as 50% in handheld devices [27].

Though wireless bandwidth has improved exponentially since the 1990's as can be seen from figure 2.1, bandwidth improvements remain low in contrast with the developments of CPU and storage capacity. Toh observes that if a routing protocol incurs excessive control traffic, the available network bandwidth will be consumed by control traffic which can impact performance [113]. In such environments where routing is performed through opportunistic links, the limited bandwidth may result in a bottleneck where the contact cannot be used to the full before severed.

The result of not utilising bandwidth effectively will result in missed communication opportunities. It will also result in a reduction in the battery life of the node. Methods for maximising wireless communication include data compression and protocol design to use communication between nodes sparingly and effectively.

### **2.1.4 High Loss Rate**

Wireless links suffer from a much higher loss rate than wired links. Ramanathan and Steenstrup observe that wireless links tend to have lower and less predictable quality than wired links, and their quality is highly sensitive to environmental conditions such as the terrain, surrounding noise and the distance between nodes [100]. Packet loss may also occur due to disconnections caused by node mobility. A related problem is that transport protocols such as TCP anticipate that packet loss is caused by congestion as it was designed to work over wired links [59]. As a result the TCP congestion protocol may tell the sending node to reduce the sending rate, assuming the loss is due to congestion. This causes a decrease in performance. The transmission

protocol needs to distinguish the nature of the error in order to determine the most appropriate action. Ad hoc TCP (ATCP) utilises Explicit Congestion Notification (ECN) messages to explicitly notify the sender of network congestion [83]. On receipt of an ECN message, congestion control is invoked. If loss occurs and the ECN flag is not set, ATCP assumes that loss is due to errors and retransmits the lost packet, thus differentiating between loss due to congestion and lossy links.

### **2.1.5 Presence of Asymmetric Links**

An asymmetric link is when communication between two nodes has a higher capacity in one direction than in the other. This can occur for a number of reasons such as heterogeneity in transmission capabilities such as bandwidth and transmission power. Additionally, it may be caused by the use of directional antennas or the existence of background noise. Extreme cases of link asymmetry can result in unidirectional links when information can be sent and received in one direction but not the other.

Conventional network and transport protocols assume bidirectional communication in order to function. The performance of TCP in asymmetric links, for example, is reduced as the throughput is regulated through the flow of acknowledgments, meaning the capacity is limited to the capacity of the slower direction. Additionally a number of acknowledgments may arrive at once causing bursty traffic in the forward path known as ACK compression [124]. Additionally, asymmetric conditions may result in inaccurate round trip time estimations.

The conservative solution employed by MANET routing protocols such as AODV is to detect asymmetric links and avoid using them. Alternatively, communication is enabled by tunneling through intermediate nodes. However, in sparse node population environments this may prove to be impractical as it cannot be assumed that there will be other nodes within communication range to provide tunneling. However, Sterbenz et al. observe that the usefulness of such networks may require the use of highly asymmetric and possible unidirectional links and as such must be supported [109].

For example, a number of techniques have been proposed for improving TCP performance in asymmetric links. TCP header compression can be used to reduce the size of ACK packets [60]. However, as stated in [8], this solution alone is ineffective due to the per-packet over-

head, which is independent of packet size. ACK filtering attempts to minimise the amount of acknowledgments in the lower capacity link by suppressing acknowledgments waiting at lower capacity node and then discarding them when new ACKs are to be queued [9]. ACK reconstruction can be used in conjunction with ACK filtering to smooth out the appearance of ACKs thus reducing the effect on TCP of infrequent acknowledgments on the sender's side [8]. ACK scheduling prioritises ACKs over data on the constrained link in order to minimise the time ACKs spend queued behind upstream data packets in an attempt to protect the sender from starvation [8]. The ECN mechanism discussed in section 2.1.4 also aids in dealing with asymmetric links by explicitly notifying the sender when congestion occurs, signaling the sender to slow down the transmission rate. Balakrishnan et al. in [8] propose sender adaptation where the sending node does not just consider the number of acknowledgments received to determine the sending rate, but also the amount of data that has been acknowledged.

The aim of the above techniques is to reduce the amount of data that needs to be sent along the lower capacity link, while preventing the reduced traffic from adversely affecting the transmission rate along the higher capacity link.

## **2.2 Mobility**

The ability of nodes to change location frequently and possibly be constantly in motion is the source of many of the challenges in DDTM environments. The high mobility combined with a sparse node population results in disconnected network graph with a highly dynamic topology. This is compounded by lack of control over node movements with limited knowledge of future node movements. These characteristics combined result in limited topology information available.

### **2.2.1 Limited Control Over Node Movements**

The environments considered consist of nodes that may be attached to other objects such as vehicles, animals or people. Given these assumptions, it is not generally possible to assume the existence of any control or influence over node movements. The assumption of control over node movements would result in a limitation to the usefulness of the network, which can



only be used in specific environments where the network nodes have an influence over their trajectories.

However, even though node movements cannot be controlled, nodes tend to exhibit some degree of regularity in mobility patterns [112]. As such, mobility prediction techniques may be used to estimate how nodes are expected to move and therefore aid routing decisions.

### **2.2.2 Limited Knowledge of Future Node Movements**

As previously mentioned nodes may move frequently or be in constant motion. Though resources exist to determine current location such as GPS, location beacons and so forth, the knowledge of future node movements may be limited. In certain circumstances assumptions may be made, for example using bus route time tables or predefined trajectories, however this may not always be the case. Mobility prediction techniques may be used to predict node movements.

The Mobile Motion Prediction (MMP) algorithm makes use of the user's movement history [82]. Movements are considered to be a combination of random and regular movements and are matched using a Markov chain model made up of movement circles and movement tracks. The main drawback of the MMP algorithm is its high sensitivity to random movements. Any movement that cannot be classified by the simple mobility patterns defined is classified as random movement. Prediction performance of MMP decreases linearly with the increase in the random factor [82]. Su et al. predict the Link Expiration Time (LET) between any two nodes in an ad hoc network by making use of the co-ordinates of the nodes, their speeds and direction of motion [110]. Chellapa et al. assume that the area is divided into cells and mobility prediction is on predicting the next cell a node will visit based on their current location within that cell [25]. However, the assumption that the area is pre-divided into cells imposes restriction on the usefulness.

However, a key challenge of highly dynamic environments is to adapt to node movements. Though some knowledge may be available in regards to future node movements, this does not guarantee that nodes will not diverge from these trajectories. As such, any information available cannot be considered static or guaranteed. Additionally, any mobility prediction method used must be reactive and adaptable, using node movements to further improve prediction ca-

pabilities.

As such, it is necessary then to assume that information available is probabilistic and may change over the course of the message traversing through the network. This means that forwarding decisions must be locally determined based on the current information available.

### **2.2.3 Disconnected Network Graph**

The characteristics of DDTM environments mean that the network graph is rarely if ever connected. The result is that two nodes wishing to communicate may never have a full end-to-end path to each other at any given point in time.

The majority of classic MANET routing protocols such as AODV [98], DSR [64], DSDV [99] and LAR [69] make the assumption that the network graph is fully connected and fail to route messages if there is not a complete route from source to destination at the time of sending. Kevin Fall in [38] states that interruption is an characteristic of the operating environment that must be accommodated by any architecture. The solution is to assume that only asymmetric communication is possible.

Epidemic routing presented in [115] disseminates information between nodes by exchanging summary vectors of messages held by each node. Nodes compare summary vectors and exchange previously unseen messages thus propagating messages across a disconnected network. Though promising as a solution, messages are distributed to each node in the network which is relatively wasteful of valuable resources.

### **2.2.4 Highly Dynamic Topology**

In MANETs nodes may move freely and organise themselves arbitrarily, thus the network's topology may change rapidly and unpredictably. Unlike fixed infrastructure networks where link failures are relatively rare, McDonald et al. observe that the rate of link failure due to node mobility is the primary obstacle to routing in ad hoc networks [88]. High mobility causes rapid changes in link connectivity meaning that topology information cannot be distributed throughout the network, as it would result in a prohibitively large amount of routing updates that would consume the available bandwidth. Additionally, the dynamic nature may mean that any topology information may become stale after a short period of time making it ineffective

for use in forwarding decisions.

As a result, the knowledge of complete paths will not be available in the network. The solution is to ensure that forwarding decisions are based on local knowledge, exploiting new links as they become available.

### **2.2.5 Limited Topology Information**

Whereas sections 2.2.1 and 2.2.2 dealt with a node's knowledge and control of its own behaviour, this section deals with a node's knowledge of its surroundings. As mentioned in section 2.2.4 the network will be highly dynamic and as a result the topological information of the network available may be extremely limited. Due to high mobility and limited transmission range traditional MANET routing protocols cannot react quickly to the rapidly changing topology and may also cause excessive overhead of large update messages which may become stale very quickly, ultimately degrading the performance of the network. As a result, nodes may have little if any information regarding the number of other nodes and the location and links between them. This means that at no time can the complete network topology be known by any individual node. It is also to be assumed that based on the information available a complete route to a given destination may not be available. The result is that each node must form a local view of the network topology and forwarding decisions made accordingly.

### **2.2.6 Variable Link Duration**

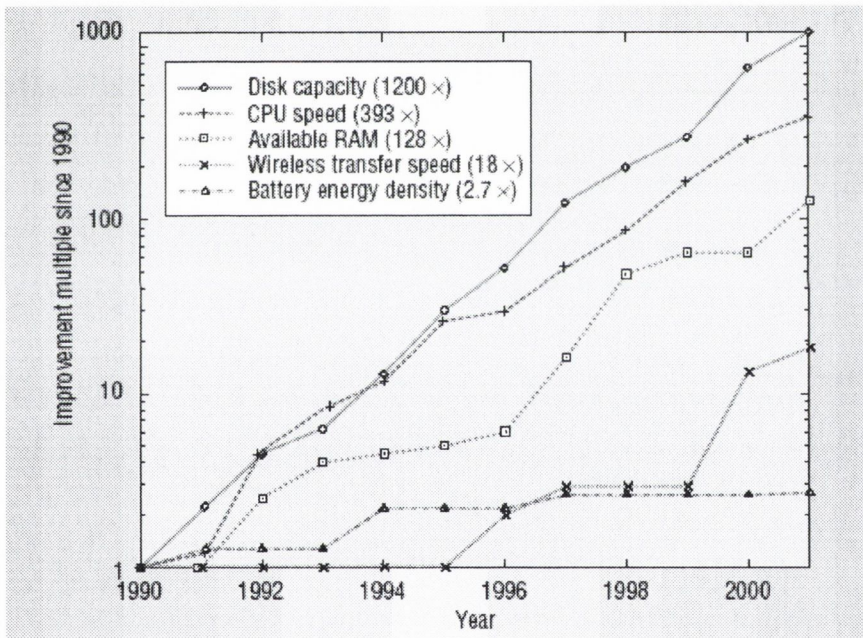
Link duration defines the amount of time a link between two nodes remains useful for transmission. The frequent movements of nodes and the assumption that node movements are not influenced means that the duration of connections between nodes may vary greatly. Nodes may be in contact for a long period of time or it may be only a passing connection. The anticipated length of the connection may have an influence on the forwarding strategy used. A short link duration may mean that not all data destined for that node may be transferred within the limited connection duration.

A number of techniques exist to estimate the stability of links. Su et al. define Link Expiration Time (LET) using location information to predict the duration of the link in order to anticipate a disconnection [110]. Routelifetime Assessment Based Routing (RABR) is based

on analysing signal strength. It tries to predict the time when the received signal strength falls below a critical threshold by measuring the average change in received signal strength [6].

## 2.3 Portability

Small portable mobile devices are heavily constrained due to pressures of size and weight. As a design trade-off for portability mobile nodes have access to limited resources. Though resources in mobile nodes are increasing, the fact remains that mobile nodes are much more resource limited than wired nodes, therefore, resource conservation remains a vital underlying challenge for such networks.



**Fig. 2.1:** Laptop technology improvements during the past decade. The logarithmic scale shows multiples of the technology's improvement since 1990 [108]

### 2.3.1 Limited Battery Power

Forman et al. observe that battery power is finite and represents one of the greatest constraints in designing algorithms for mobile devices. Goldsmith et al. point out that energy constraints are not inherent in all ad hoc networks, as some nodes may have access to a power source [48].

However, many if not most, mobile ad hoc nodes will be powered with batteries with a limited lifetime. As can be seen in figure 2.1 the improvement of battery power over the last decade is the least improved of node resources, meaning it will continue to be a constraint for mobile nodes.

Wireless communication generally consumes significant battery power and Chlamtac et al. in [27] observe that in such MANETs each node is acting as both an end system and a router at the same time placing additional energy demands on nodes. Additionally, Goldsmith et al. in [48] discuss that energy constraints impact both hardware operation and the signal transmission associated in hardware operation, thus having a direct impact on other resources available on the mobile node such as processing power and signal transmission.

If power-saving techniques are not employed the depletion of batteries in nodes greatly reduces the performance of the overall network. If a node no longer has power, then it can no longer participate in the network in forwarding/storing data and data that is stored on the device is no longer accessible.

General solutions are to reduce wireless communication through efficient transmission in order to increase the operating time of the mobile node and to enable power management techniques at the hardware level.

### **2.3.2 Limited Memory**

Forman et al observe in [40] that storage capacity on portable computers is limited by physical size and power requirements. Though there have been improvements as seen in figure 2.1 these values represent laptop specifications which may have significantly more storage capacity due to their size, as such memory still needs to be considered limited.

Limited storage capacity poses a problem as there may be a large amount of data generated by the application, and due to the limited connectivity and available bandwidth of the device it may have to store this data for extended periods. Additionally, information used during routing protocol need to be held in memory by the device consuming this limited resource.

As the storage capacity is finite the potential to run out of memory is high, resulting in loss of data. Common solutions to this problem include data compression, limiting the routing information required and reducing the memory footprint of the application.

### 2.3.3 Limited Processing Power

Though mobile devices still have lower processing power than most stationary devices, the rate of improvement of CPU capabilities is far greater than improvements in available battery power and bandwidth as can be seen in figure 2.1. CPU utilisation requires battery power but it requires significantly less than wireless transmission, and as battery power and bandwidth are the more scarce resource it can be contended that expending additional CPU resources in order to reduce transmission is desirable.

## 2.4 Second Tier Challenges

The previous section discussed in detail the primary challenges associated with DDTMs. For- man et al.'s classification is extended to include the combined challenges of the wireless envi- ronment, mobility and portability result in low delivery reliability, high end-to-end latency and poor QoS support. These aggregated challenges are referred to as second tier challenges in or- der to highlight that they occur due to the combination of first tier challenges. This relationship is highlighted in figure 2.2.

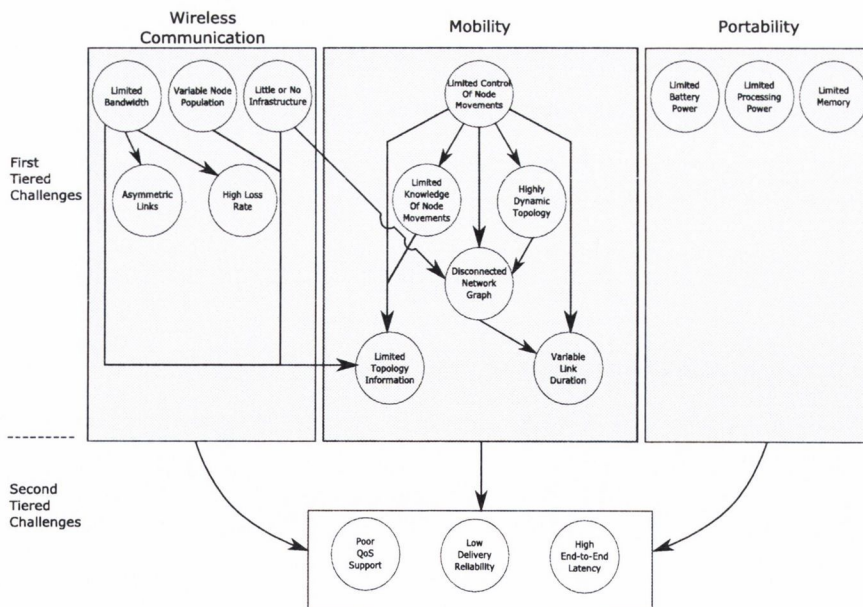


Fig. 2.2: Tiered Challenges

### 2.4.1 Low Delivery Reliability

Delivery reliability is defined as the fraction of data generated within the network that is eventually delivered to the destination. Delivery reliability is low in DDTMs due to limited resources (section 2.3), asymmetric links (section 2.1.5) and a disconnected network graph (2.2.3). On this topic, Fall states that ‘for networks which lack any return channel at all, any form of reliability will be at best probabilistic’ [36]. Delivery guarantees cannot be provided because resource constraints on nodes may cause an intermediate node to drop messages. However, attempts can be made to improve reliability.

It is the task of any routing substrate to maximise the probability of the delivery of data to the network destination. Conventional reliable delivery techniques use end-to-end acknowledgments to verify that data has been delivered and resend any data that has not been delivered within a designated time. The highly disconnected and end-to-end latency of these networks makes this solution difficult and unsuitable. Possible solutions to improve delivery reliability include flooding, replication, hop-to-hop acknowledgments and custody transfer. Flooding consists of forwarding the message to all nodes met on the network which is excessive, sending many more messages than necessary. This solution consumes considerable bandwidth and memory resources and as a result can eventually degrade the performance of the system. Replication involves duplicating the message and distributing it to several nodes, but typically not all. Different replication strategies may be used, such as limiting the number of copies allowed to exist in the network. Thus consuming less bandwidth and memory, leading to better scalability. However, sensible replication strategies must be used in order to conserve resources. Custody transfer proposed in [38] essentially changes the notion of reliability from end-to-end reliability to hop-by-hop reliability whereby a node receiving a message assumes responsibility for the eventual delivery. This is not necessarily employed at each hop, potentially utilising nodes with added resources such as persistent data storage or longer battery life time. This mechanism has the advantage of not wasting node resources while taking advantage of nodes with more extensive resources. However, once custody has been transferred if the node were to fail unexpectedly, reliability would be lost.

### **2.4.2 High End-to-End Latency**

End-to-end latency is high in DDTMs. This is due to the disconnected nature resulting from mobility (section 2.2) and the variable link quality due to the wireless environment (section 2.1). End-to-end latency is dependent on the time varying graph. While a route may be available from source to destination when considering the time varying graph, the time considered by the graph may be anything from minutes, hours, days and even weeks. This problem is further exacerbated by the limitations of the connectivity of the network. Though a node may establish a connection with a suitable node for delivery of the data, the limited bandwidth and variable link duration, could result in the data not being transferred during that connection. As a result, the data may have to remain on the intermediate node until another opportunity to transfer the data arises. As such, standard mechanisms in network management cannot be employed, such as the use of acknowledgments to clear out buffers, the provision of reinforcement learning for stable links and identification of shortest paths. The latency to be expected depends on the deployment environment and the application. Support should ideally handle varying delivery latencies.

### **2.4.3 Poor Quality of Service Support**

Quality of Service (QoS) aims to provide consistent, predictable data delivery. However, low delivery reliability and high end-to-end latency means that DDTMs have poor support for QoS. Typically QoS mechanisms involve raising the priority of a flow or limiting the priority of another. While traditional QoS cannot be provided where absolute performance requirements are facilitated, Brunner et al. point out that some facility for an application to indicate its intentions associated with the data may be important [19]. They observe that this is different from traditional QoS which provides absolute performance requirements. The QoS in DDTMs more involves giving a hint to nodes on the network as to how to treat the data.

Cerf et al. have defined priority classes for data in DTNs [22]. The class priority system is based on the U.S. postal service where there are generally no strong guarantees of timely delivery, however, there are some forms of class of service. The priority classes are defined as bulk, normal and expedited. Bulk messages are not forwarded until all other classes of messages for the destination have been forwarded. Normal messages are shipped before bulk



messages, but only after any expedited messages for the destination node. This mechanism is a simple method for ranking messages in order to give priority to more important message.

## **2.5 Summary**

This chapter outlined the challenges associated with DDTMs. Using the format defined by Forman et al. new challenges that are specific to the environment of DDTMs have been identified. The consequences of three primary characteristics of DDTMs have been discussed: wireless communication, mobility and portability. Wireless communication results in a challenging communication environment where pair-wise communication is not simple. Mobility causes a highly dynamic network where information in regards to the network is extremely limited and volatile. Portability has the consequence of limited resources and limited longevity.

Additionally, Forman et al.'s classification has been extended by identifying a new classification of first and second tier challenges. First tier challenges fall into the classification outlined by Forman et al., second tier challenges are a result of the combination of first tier challenges. The second tier challenges discussed are poor delivery reliability, high end-to-end latency and poor quality of service support which are problematic in traditional computing. The inter-relationships between these challenges have also been outlined.

# Chapter 3

## State of the Art

In MANETs nodes can only communicate directly with other nodes that are within range. As such, communication with nodes out of range is achieved through repeatedly forwarding messages to intermediate nodes until reaching the destination. In networks where nodes are constantly moving, the decision of which node to forward messages to is difficult, since the network's topology is constantly changing. Additionally, in sparse networks an end-to-end path may never exist between nodes at a single point in time. Finding a solution to this problem has been an area of much research spawning a number of different terms.

**Delay-tolerant networks (DTN)** - This term was first coined by Vint Cerf et al. in an Internet draft in relation to interplanetary research [23]. The authors determined that such a network should not assume a direct or indirect path to the destination at the time of sending and a store-and-forward mechanism should be used to route data.

**Disruption-tolerant networks** - Upon the creation of the Delay-tolerant Networking Research Group, DARPA announced a similar program referred to as Disruption-tolerant networks also referred to as DTN [3]. The term disruption-tolerant networks is used by DARPA specifically, to describe military tactical communication environments.

**Partially connected networks** - Vahdat and Becker used this term to describe MANETs where the presence of a connected end-to-end path from source to destination may not be assumed [115]. A partially connected network is a fragmented network, i.e. a net-

work where groups of nodes are interconnected with each other, but disconnected from the rest of the network.

**Intermittently connected networks** - Vint Cerf et al. broadly classified contacts in DTNs into persistent and intermittent contacts [23]. A contact is a time-window when data can be exchanged between two nodes within range. Any contact that is not persistent and always available is therefore an intermittent contact. Intermittently connected networks (ICN) generally defines any network where intermittent contacts are utilised in order to forward data [79].

**Opportunistic routing** - Opportunistic networks are a subclass of DTNs. Vint Cerf et al. defined different types of intermittent contacts where a contact could be scheduled, opportunistic or predicted [23]. ‘Opportunistic contacts are those that are not scheduled, but rather present themselves unexpectedly’. An opportunistic network is a network that exploits opportunistic contacts in order to forward data to an encountered node.

**Space-time routing** - Space-time routing determines a route over a network graph that is disconnected at any given point in time [89]. When the future movements at specific times are known, a space-time graph may be constructed in order to determine a time-dependent route. Space-time routing can be classified as a routing mechanism suitable for a subclass of DTNs where contacts are scheduled.

**Challenged environments** - Challenged environments is a term used to describe the type of network environments that are characterised by end-to-end latency, bandwidth limitations, limited node longevity, resource limitations and end-to-end disconnections between nodes [37]. DTN protocols are designed to function in challenged environments.

**Extreme networks** - Extreme networks is a term used to describe networks that operate in challenged environments [35].

All of the above terms describe a network that may never be completely connected. Evidently, there is a large amount of overlap in these terms but the problem of how to route messages in

a disconnected delay-tolerant network remains the same. The disconnected nature means that message forwarding generally works using the ‘store-carry-forward’ paradigm. The idea originated from telecommunications research where switches used the ‘store-and-forward’ method. This method involved a switch fully receiving a packet and temporarily storing it while verifying the contents before forwarding it. The paradigm is extended in disconnected networks with the use of ‘store-carry-forward’ where a node receives a message, stores it and physically carries it to a different location before the message is forwarded. The ‘store-carry-forward’ mechanism loosely describes the solutions presented in the remainder of this section. Although the mechanisms used are similar, they differ in determining when and to whom the message is forwarded.

Current approaches to determine forwarding decisions can be broadly divided into two categories: deterministic and stochastic [125]. Deterministic methods solve the problem of how to route in a disconnected delay-tolerant network by making the assumption that the dynamics of the network are not unpredictable but deterministic and known a priori [53, 62, 89]. An example of a network with deterministic dynamics are orbiting satellites, where the time where a satellite may be within range is completely dependent on the orbit of the satellite. Another possible deterministic network is a factory consisting of robots that move in a predetermined manner. As such, the routing problem becomes a matter of determining a path over a connectivity graph that is time-varying but whose dynamics are known in advance.

This assumption greatly simplifies the problem by total predictability. However, in real life environments, the dynamics may not be so accurately predictable and known in advance. For this reason, stochastic methods have been proposed which attempt to cope with the randomness that may exist in the network, where the next state of the environment is not fully determined by the previous state. The term stochastic describes a system involving chance or probability. If the time-evolving topology is stochastic, routing can be achieved by moving the message closer to the destination one hop at a time. Various approaches exist, for example epidemic-based solutions blindly inject a number of copies of the message into the network [10, 31, 51, 93, 96, 102, 105, 115]. History or prediction-based solutions attempt to improve on blind forwarding by employing some form of ‘probability to deliver’ metric in order to preferentially forward to nodes deemed most likely to deliver. These metrics are based on either contact history [20, 65, 67, 80, 111], location information [70, 71] or utility metrics [26, 87, 106]. Some

projects go further in using nodes to deliver data by assuming control or influence over node movements [74, 126]. Additional research has been done utilising coding-based techniques to encode messages before being transferred in the network in order to provide redundancy or combining multiple messages into a single message [61, 75].

This section reviews routing approaches in disconnected networks. Deterministic solutions are reviewed in section 3.1. The various stochastic solutions are presented in the remaining sections. In this section only routing protocols that provide a store-carry-forward mechanism are considered. In particular, protocols that assume end-to-end connectivity are not considered, as these solutions are unsuitable for DDTM environments as discussed in section 1.4.

## 3.1 Deterministic Approaches

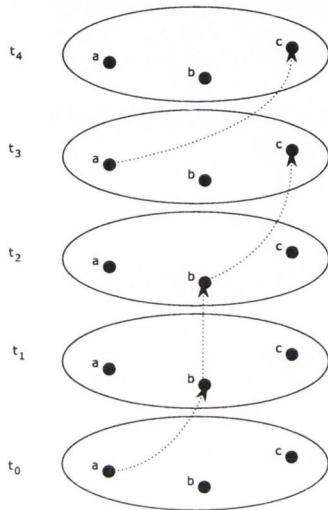
The term deterministic describes a system whose time evolution can be predicted exactly. If a complete future topology of the network is deterministic and known in advance the forwarding path can be scheduled ahead of time. Zhang has classified deterministic methods based on the underlying graph theoretic technique used in each solution [125].

### 3.1.1 Space-Time Routing

Merugu et al. achieve routing over a disconnected path by constructing space-time routing tables where the next node in the path is selected based on current as well as future neighbours [89]. The destination and the arrival time of the message are both used to determine the next hop in order to minimise end-to-end delivery delay.

The dynamic network is modeled as a space-time graph. The space-time graph is constructed as follows. Assume that node movements and timing information is known in advance over a finite time horizon  $T$  or an infinite time horizon if the node moves in a cyclic manner. The graph can be represented as  $G(t) = (V, E(t))$  where  $V$  is the set of nodes and  $E(t)$  represents the links between these nodes at time  $t$ .  $E(t)$  can also be represented as  $L_{ij}(t)$  which denotes a link between nodes  $i$  and  $j$  at time  $t$ . Using the location and time information of each node, it can be determined that a connection exists between node  $i$  and  $j$  if the distance between the physical location  $P_i(t)$  of node  $i$  at time  $t$  and the physical location  $P_j(t)$  of node  $j$  at time

$t$  is less than the transmission radius. A layered graph is then constructed, where each layer corresponds to a specific time  $t$ . Using this information the routing tables are constructed for each node and the path with the minimum end-to-end delivery delay is computed.



(a) Routing space-time graph for nodes a, b and c over a fixed time period T

Destination Node	Next Hop			
	$t_0$	$t_1$	$t_2$	$t_3$
Node C	Node B	Wait	Wait	Node C
Node B	Node B	Wait	Wait	Wait

(b) Next Hop Routing table of Node A

**Fig. 3.1:** Space-time routing (derived from [89])

Figure 3.1 a) shows a time-space network graph for *node a*, *node b* and *node c*. The next hop is selected as a function of destination and time. Figure 3.1 b) shows the routing table of *node a*. In order to route a message from *node a* to *node c*, two paths are available, if the message arrives at *node a* at time  $t_0$  the message could be forwarded to *node b* at time  $t_0$  arriving at time  $t_1$ . *Node b* carries the message until  $t_2$  when a link exists to *node c* at which time the message is forwarded to *node c*. If the message arrives after  $t_0$ , *node a* must carry the message until time  $t_3$  at which time a direct connection exists with *node c*.

The time expanded graph is based on Ford and Fulkerson's time-expanded networks, where the network is viewed as a static network with one copy of the node set at each time step building a time layer [39]. Using this technique the network flow over time problem is translated into a static network flow, and standard graph theory is applied to compute the shortest path for a source to destination.

## Discussion

The authors do not make any assumption of controlling node movements, however, the strong assumption is that complete knowledge of node future trajectories are available and reliable. Based on this assumption, communication is achieved in a completely disconnected network graph. The graph topology may be highly dynamic, as long as it is known in advance. However, any deviation from the predicted topology will result in broken paths. Routing tables are constructed once in an initialisation phase with all the trajectory information available. Once a message has been scheduled for transmission at a given time slot, the slot is considered full and further transmissions to that node are scheduled on the next available time slot. The result is that the protocol does not take advantage of long links in order to transmit the maximum amount of data queued. Additionally, it is not clear that they take into account the possibility that the data scheduled for transmission at a given time slot may not be fully transferred before the link is severed.

Routes between each node pair at each time step is calculated at initialisation, there is no exchange of routing information, or routing requests. All transmissions and therefore resources are utilised to transfer data, rather than consumed with control data. Because all routing decisions are made in advance, the only resources during the network's operation is node memory for storing the routing 'script'. Additionally, only one copy of the message exists on the network at any given time, and the solution therefore avoids needlessly retransmitting the message, as well as the need for multiple nodes to store the message at a given time. The computation of the routing tables at initialisation is significant, but occurs only once. Following initialisation, route calculation is a basic routing table look-up. As a result the protocol performs well in regard to resources.

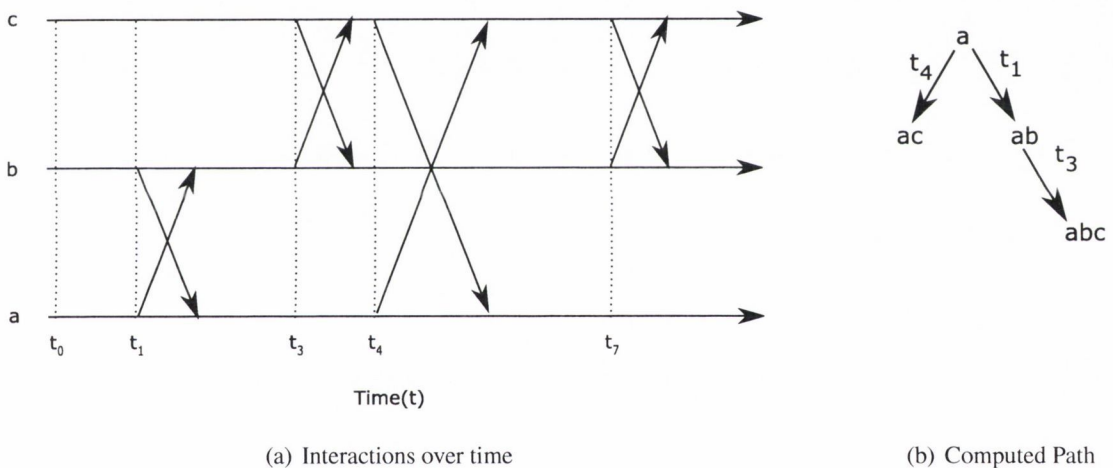
Assuming the node movements are exactly known and do not deviate from the defined trajectory and timing information then delivery reliability is high and the end-to-end latency is minimal. However, the repercussions following deviation from the predefined trajectory could significantly degrade the solution's performance. A delay of a single node could cause large queues as the node currently carrying the message waits for another possible connection. Additionally, if the node is delayed once, unless there is a means for the node to catch up and return to the predefined trajectory, the delay would result in the routing tables of all nodes that use this node as an intermediate node to be inaccurate, thus reducing reliability and significantly

increasing end-to-end delays. As such the solution is unadaptive.

Though this solution requires no additional infrastructure, it will not cope with a changing node population as each node and its trajectory information is known in advance. As a result, a new node joining the network may not participate, and a node unexpectedly leaving the network would result in the failure of any path that includes that node.

### 3.1.2 Tree Approach

Handorean et al. model the connectivity of a disconnected network by mapping the sequence of connectivity between nodes to a directed graph [53]. A message path between any two nodes is defined as a sequence of hops that a message can follow over time from the source node to the destination node.



**Fig. 3.2:** Message Path [53]

The mobility of each node is captured in what is referred to as a ‘characteristic profile’. The ‘characteristic profile’ of each node is known in the system and mapped as  $m_p(t)$  where  $m$  is the physical location of node  $p$  at time  $t$ . Using this information, the connectivity between two nodes may be determined if the distance between two nodes’ physical location at a given time is less than the transmission radius of the nodes. Figure 3.2 shows a simple example of the future interactions between *node a*, *node b* and *node c*. In order to route a message from *node a* to *node c*, a directed graph is constructed, with the source node as the tree root. From this graph, the number of hops and the estimated arrival time for the message can be computed.



A list of visited nodes is maintained while computing each path in order to prevent loops. The directed graph is constructed each time a node wants to send a message to a destination node.

An extension to the algorithm relaxes the assumption that each node's 'characteristic profile' is globally known throughout the network. Upon meeting, nodes exchange information in regards to their own mobility and the mobility of other nodes met. The complete route is calculated by the source node before sending. However, if an intermediate node receiving the message has a better route, then the intermediate node makes the local decision of adjusting the message path accordingly. If a complete path is unknown to the source node at the time of sending, Epidemic Routing is used [115]. Epidemic Routing is further discussed and explained in section 3.2.

## **Discussion**

No control over node movements is assumed, however, nodes must have complete knowledge of their future movements. The topology information is either globally known, or when only partial knowledge is available a local view of the topology is constructed. As such, communication is achieved across a completely disconnected network graph. However, the assumption is that knowledge of future node movements is perfect, thus any deviation from the future trajectory information will result in path failures.

Route calculation must be performed each time a message is sent. The route calculation is quite complex and therefore requires a lot of processing power. The characteristic profile of each node must be stored locally imposing significant memory constraints on the device. The assumption is that the trajectory information may be stored in the form of a repeated equation, or a repeating sequence of movements. But if the movement of nodes is not repeating, then the characteristic profile becomes prohibitively large and as such the protocol cannot be used. If global topology information is assumed, bandwidth and battery power are conserved because the message is only forwarded along one path. However, if a complete path is unknown at the time of sending, the protocol resorts to Epidemic Routing [115]. Epidemic Routing is costly in terms of resources which is discussed further in section 3.2. Alternatively, if we do not assume global knowledge, nodes exchange their characteristic profiles and the characteristic profiles of all nodes encountered. These characteristic profiles could be potentially quite large, and as such may consume a large amount of bandwidth upon two nodes meeting. As such the protocol has

high memory, processing power and bandwidth requirements if global topology information is not assumed.

Assuming complete mobility information is available and accurate, the algorithm calculates the optimum path and as a result delivery reliability is high. However, ‘characteristic profiles’ contain explicit location and timing information, and therefore clock synchronisation is required. Clock synchronisation is difficult to guarantee in a highly distributed environment. Errors in clock synchronisation will result in inaccurate mobility information and may result in path failures, thus reducing delivery reliability. Additionally, if the link between two nodes along a path is already being used for a transmission, the message may be queued for a long period of time before it is able to be transmitted, thus reducing the usefulness of calculating the optimum path. Additionally, the size of data to be delivered in an encounter between nodes is not considered even though the duration of node encounters is known. As a result, there is no guarantee that the entire message will be delivered in the duration of the connection. As such deviations from the predefined trajectory, clock synchronisation errors and contention for transmission slots will affect delivery reliability.

Similar to Merugu et al. (section 3.1.1) this solution makes no assumption of the existence of infrastructure. The protocol provides greater flexibility compared to Merugu et al. in that new nodes may join the network by meeting a node and exchanging characteristic profiles. However, if a node leaves the network unexpectedly and is an intermediate node along the route path, then the route becomes invalid. As a result, the system does not cope well with unexpected departures from the network.

### **3.1.3 Modified Shortest Path Approaches**

Jain et al. present a number of algorithms for routing in delay-tolerant networks [62]. They assume the existence of one or all of the following information in the form of oracles that have specific knowledge about the network: ‘contact summary oracle’, ‘contact oracle’, ‘queuing oracle’ and ‘traffic demand oracle’. The ‘contact summary oracle’ contains the average waiting time until a given connection becomes available. A contact is defined as an opportunity for two nodes to exchange data at a given time. The ‘contact oracle’ contains exact information regarding contacts that will occur between two nodes at any point in time. The ‘queuing oracle’

knows the buffer occupancy of any node at any given time. The ‘traffic demand oracle’ contains information about the present or future traffic inserted into the network at any given time. All the oracle information is calculated in advance and different oracles are made available to nodes for a number of different scenarios. The intention is to determine how much information needs to be available to nodes in the network in order to achieve high delivery performance, load balancing etc.

Routing with partial knowledge is achieved through assigning costs to edges, where edges are links between nodes as they become available. This information is used to construct a shortest-path problem, where the shortest path is the path with the minimum cost associated with it. Assuming only the contact summary oracle is available, the information is time-invariant. Therefore, the Dijkstra’s shortest path algorithm may be applied using the average waiting time to calculate the cost of a route.

The contact oracle information is time-variant. As such, Dijkstra’s algorithm is modified to take into account the time-varying information of the start time  $T$  and the time the message will arrive at each node in order to calculate the cost. This variation of Dijkstra’s algorithm is applied where the cost of the edge is computed under two different scenarios. The first utilises the contact oracle and local queuing information in order to route around congestion at the first node along the path. The second utilises the contact oracle and the queuing oracle where global queuing information is available and edges along the computed path are reserved to ensure that messages are transmitted over the assigned edge to avoid missing scheduled contacts due to congestion. Finally, the authors consider the scenario where all the oracles are available. In this case, the optimal route is calculated to minimise the average delay in the network using a linear programming formulation.

## **Discussion**

This work makes the assumption that global knowledge of node contacts, queue size and traffic demands are available. The authors acknowledge that the implementation of these knowledge oracles in a real world situation would be impractical. But the approach is still useful in order to analyse performance results of scenarios where different knowledge oracles are available. These results are then used to identify which information is most useful in calculating routes in order to achieve the best delivery performance. The analysis extends on the previously

presented work in section 3.1.1 and section 3.1.2, where the route is not calculated purely based on computing a path given mobility information, but also using queue size and traffic demands in order to take into account that the optimum path may be congested in a realistic network environment. Results analysis determined that in environments where contacts are plentiful the benefits of full knowledge are minimal. However, when resources are limited (such as limited contacts, bandwidth and storage) then the knowledge oracles become useful. Under these circumstances the contact oracle and the use of local queuing information proved valuable. Though promising, the problem still remains that node behaviour in this network is assumed to be deterministic, where node contacts are predictable and known in advance.

### 3.1.4 Conclusion

Deterministic methods solve the problem of how to route through a disconnected delay-tolerant network by making the assumption that the dynamics of the network are completely predictable. As such, the routing problem becomes a matter of determining a path over a connectivity graph that is time-varying but whose dynamics are known in advance. These solutions however, are limited to environments where such an assumption may hold. Examples of such environments include interplanetary communication, where the orbit of satellites are predictable and cyclic and the use in a factory where the robots are used and the movement patterns are predefined and known in advance.

The assumption of deterministic behaviour means that these solutions perform well in term of delivery reliability as discussed in section 2.4.1. However, the environments that discussed in section 2.2 do not necessarily have deterministically predictable or globally known movement patterns. As such these solutions are unsuitable for DDTMs.

## 3.2 Epidemic-Based Approaches

Epidemic algorithms were first used to guarantee consistency in distributed databases in 1987 [32]. They adopt the terminology of modeling infection diseases used in epidemiology literature. The ‘simple epidemic’ is one in which individuals are either *susceptible* or *infective*. An individual is *susceptible* to a disease if it is prone to it and *infective* describes an individual

that is infected with the disease. When  $n$  individuals are initially susceptible, one individual is inserted that is *infective*. The infective individual upon meeting a *susceptible* individual will infect them. Eventually, all  $n$  individuals will become *infective*. This is known as the *susceptible-infective-susceptible* (SIS) model [54].

### 3.2.1 Classic Approach

One of the first protocols to solve the problem of routing in disconnected environments was epidemic routing for partially connected networks proposed by Vahdat and Becker [115]. Their epidemic algorithm is a method for distributing messages in a dynamic disconnected network [32]. Epidemic Routing relies on nodes referred to as carriers coming into contact with other connected portions of the network through node mobility. Messages spread like a disease epidemic through the network as nodes meet and infect each other with messages. The aim is to deliver a message to a particular host with a high probability and to minimise the message delivery latency.

Each host maintains a buffer consisting of messages it has generated along with messages it has received from other nodes. Each node stores a summary vector representing the messages held by the node. When two nodes come within communication range, they exchange their summary vectors, each node examines the other's summary vector in order to determine which messages held by the other node it has not yet received. In turn, each node then requests copies of the messages not stored in its buffer. This algorithm effectively results in flooding the message throughout the network. Given sufficient buffer space and time, this protocol guarantees eventual delivery. In order to limit the number of copies of the message contained in the network, a maximum hop count is assigned, meaning a message with a hop count of one will only be delivered if the carrier comes directly into contact with the destination. Higher priority messages may be assigned a large hop count, meaning a large number of copies may be distributed throughout the network in order to reduce delivery time.

As an example of Epidemic Routing in a real environment, Small and Haas have presented the Shared Wireless Infostation Model (SWIM) in order to deliver sensor data held by mobile devices carried by whales [105]. The mobile devices attached to whales collect data which is stored on the device. Upon meeting another whale, Epidemic Routing is employed to exchange

sensor information. When a whale passes by an Infostation, the information held by the mobile device is uploaded.

Davis et al. explored using Epidemic Routing among Wearable Computers attached to people [31]. Vahdat's work assumed infinite resources are available, the authors extend this work and apply Epidemic Routing to situations where resources are limited. In this case, buffers will become full and messages must be dropped. They propose four types of drop strategies: Drop-Random (DRA), Drop-Least-Recently-Received (DLR), Drop-Oldest (DOA) and Drop-Least-Encountered (DLE). Results revealed that dropping packets destined for nodes that have been encountered less in relation to other nodes (DLE) leads to the best results.

## **Discussion**

Epidemic Routing supports eventual delivery of messages with minimal assumptions regarding the underlying topology and connectivity of the underlying network. Only pair-wise connectivity is required to ensure eventual message delivery. It succeeds in routing where classic MANET routing protocols would fail. The flexibility of the protocol means that the networks can cope with nodes joining and leaving efficiently.

However, nodes are required to have a large buffer size to ensure eventual delivery. According to [115] in order to ensure eventual delivery the buffer size on a subset of the nodes must be at least equal to the expected number of messages in transit at any given time. If resources such as buffer space and bandwidth were unlimited, this approach would achieve best possible performance as it would find the shortest delay path possible from source to destination. However, as discussed in section 2.3 resources in mobile devices are limited. Limited resources mean that delivery reliability is reduced, as messages are dropped when buffers are full. The work of Davis et al. shows that intelligent drop strategies may be used to reduce the adverse effects of limited resources on delivery reliability [31]. However, delivery performance of Epidemic Routing with limited resources is still less than that where infinite resources are assumed, even with an intelligent drop strategy.

Additionally, this scheme is wasteful of energy and suffers from contention for network resources such as bandwidth and buffer space, which can significantly degrade the performance of the network as has been noted in [96, 79]. Two general approaches have been taken when trying to reduce the overhead associated with epidemic routing. The first is to limit the number

of copies of the message allowed in the network as reviewed in section 3.2.2. The second is to minimise the number of hops as discussed in section 3.2.3.

### 3.2.2 Limited Number of Copies

One simple modification was proposed by Ni et al. to significantly reduce contention and resource consumption [96]. When a node receives a message through Epidemic Routing the node forwards the message with a probability  $p$ . When  $p = 1$  the scheme is equivalent to flooding. The results showed that in a dense population of nodes, a small value of  $p$  is sufficient, however, a larger value of  $p$  is needed if the node population is sparse.

Spyropoulos et al. attempt to tune the number of copies based on the population introducing ‘Spray and Wait’, a technique that sprays (distributes) a number of copies into the network, and then waits till one of these nodes meets the destination [107]. A message is assigned a replication number  $n$ . Upon encountering a node that does not hold a copy of the message the node forwards the message with a replication number of  $n/2$ , keeping a value of  $n/2$  for itself. The node continues forwarding messages until  $n = 1$ , meaning the node can now only perform a direct transmission to the destination node. The value of  $n$  is tuned based on the estimated node population. The estimated population is derived from sampling the time it takes for a node to meet one node  $T1$  and the time taken to meet the next node  $T2$ .

#### Discussion

Limiting the number of copies reduces the memory requirements and bandwidth consumption associated with Epidemic Routing. However, work of Spyropoulos et al. is based on the premise that node movements are similar to that of a random walk. As such each node has an equal probability of meeting every other node, and therefore, as long as the number of message copies is proportional to the node population, the message has a high probability of being delivered. However, in real environments nodes do not move according to a random walk mobility model and as such delivery performance may be greatly reduced.

### 3.2.3 2-Hop Approaches

While the previous section presented solutions that limited the number of message copies in the network, a number of solutions attempt to reduce resource consumption by limiting the

number of hops a message may take. Grossglauser and Tse limit the number of hops to two, however, an infinite number of copies is allowed [51]. Essentially, the source node distributes the message to all other nodes it meets. These nodes then become relays, and upon meeting the destination node, the message is delivered. In the theoretical network the authors have used for modeling the approach, the nodes have infinite buffers and move randomly in the network.

Beaufour et al. propose to use Epidemic Routing in order to distribute data in a disconnected network consisting of a set of static sensor nodes, static information display nodes, and a large number of mobile smart-tags [10]. Similarly, the Pollen project consists of a set of PDAs held by moving people and a set of static nodes located at places of interest or on devices [47]. When a PDA comes into contact with a node, epidemic forwarding is used and the two nodes exchange previously unknown packets, thus distributing information throughout the network.

DataMULES designed for sensor networks assumes three entities in the system: sensors, Mobile Ubiquitous LAN Extensions (MULEs) and base stations [102]. MULEs move around randomly collecting data from sensors. When a MULE passes by a base station, it uploads the data. There is no attempt to identify location patterns, and the assumption is that the MULEs have a high memory capacity, since they must carry all data until they randomly pass a base station.

The DakNet project developed a wireless network for rural areas in India [97]. Mobile Access Points (MAPs) are used to deliver data to wireless-enabled village kiosks. The MAP is mounted on a bus or motorcycle and travels in a circuit through villages. When a MAP comes within range of a village kiosk, it uses a wireless connection to deliver and collect data. When the MAP passes by a Hub with Internet access, it uploads the collected data and receives new data for the village kiosks. Though it is possible to have more than one MAP in the network, MAPs do not communicate or exchange data, and so provide a rather basic pick-up and drop-off service. MAPs are assumed to have very high buffer capacity in order to store all data until a Hub is reached. This is a very simple mechanism where information is physically carried the whole distance in a circuit around the villages. It should be noted that these are mechanisms for distributing data rather than a routing protocol, because the routing decision is a simple transmission from a static node to a mobile node.

Nain et al. present the Message Relay Protocol (MRP) [93]. MRP runs in conjunction with the DSDV MANET routing protocol [99]. When DSDV fails to find a route for a packet, the



packet is passed on to the MRP layer. The MRP layer then broadcasts the packet locally to the node's neighbours, who become relay nodes for the packet. If the node currently has no neighbours, the packet is repeatedly broadcast until a number of nodes have acknowledged the receipt of the packet. Each node carries the packet until a route is discovered less than  $d$  hops away, using DSDV as the underlying routing protocol. This protocol is designed to enhance the delivery performance of DSDV rather than serve as a routing solution for completely disconnected networks. The use of DSDV as an underlying routing protocol means that it is primarily useful for a network that is mostly connected, with a small number of disconnected nodes in the network.

### **Discussion**

Limiting the number of hops reduces the resources consumed compared to Epidemic Routing. However, this has the affect that a node may have to carry the message for a long period of time, as a result buffers may become full before the message has been forwarded and as a result may be dropped from the network.

The above protocols are either based on nodes moving according to a random movement model [51, 102] or under the assumption that static nodes exist at locations nodes frequently visit [10, 97]. As with the solutions presented in section 3.2.2, the selection of which node to carry data is unimportant since each node has an equal probability of delivering the data.

There are no real routing decisions made in [10, 47, 97, 102]. The systems offer a simple exchange of information between stationary nodes and mobile nodes. They essentially carry data through disconnected parts of the network where the mobile nodes act as a virtual backbone.

### **3.2.4 Conclusion**

Though Epidemic-based routing solutions provide an effective method for routing in disconnected environments, they are very resource intensive. The biggest drawback of epidemic solutions is the high consumption of bandwidth, and as a result there is a high consumption of battery power. Furthermore, the propagation of messages to a high population of the nodes causes high memory usage. There are few or no calculations in terms of forwarding, as a result very little processing power is required.

Epidemic solutions adapt well to the wireless environment, assuming no existing infrastructure and a highly variable node population, where any joining node may contribute to the network. The 2-hop solutions have the disadvantage that an unexpected failure or departure from the network of an intermediate relay node carrying the data will result in the message being lost. Assuming the use a sufficiently high degree of replication, packet loss is overcome by using redundancy, although this comes at a price of heavy bandwidth consumption.

In Epidemic solutions node mobility is considered an asset rather than a disadvantage and no assumption in regards to control over node movements or knowledge of future movements are made. The network may be highly dynamic and disconnected and no topology information is required. However, the information is blindly forwarded to nodes without any effort to guide the message towards its destination. In the case of classic Epidemic Routing this guarantees eventual delivery since every node carries a copy of the message. However, in solutions presented in sections 3.2.2 and 3.2.3, carriers or relay nodes are selected randomly. But since the majority of the protocols are evaluated based on random mobility this does not have an adverse affect on performance since each node has an equal probability of being able to deliver the data. In a real environment, where nodes do not move in a random mobility model, the probability of a node meeting a given node or visiting a given location is not an even distribution. In a study of tracing user movements by logging the access points a user visits throughout a campus Hsu and Helmy found that none of the users visited more that 40% of the access points on the campus [56]. Consequently, the delivery performance of may be greatly affected in a realistic environment.

The metrics of delivery reliability and end-to-end latency are highly dependent upon the level of replication used. In the extreme scenario of classic Epidemic Routing [115], data is distributed to all nodes, and assuming infinite buffer space, the optimum path is always found. Quality of Service requirements can be used in order to tune the replication number, assigning a lower value to less important data, and a high replication value for high priority data.

### **3.3 History or Prediction-Based Approaches**

History or prediction-based techniques attempt to improve on epidemic routing by not blindly forwarding messages between nodes. Instead intermediate nodes estimate the probability of

eventually meeting the destination. Based on these estimated probabilities, nodes determine whether to forward the message to a given node, or wait until a more appropriate node becomes available. These probabilities are based on either contact history, location information or utility metrics.

### 3.3.1 Contact-Based

Lindgren et al. proposed Probabilistic Routing Protocol using History of Encounters and Transitivity (PRoPHET) [80]. PRoPHET uses a probability metric referred to as delivery predictability,  $P(a, b)$ , at *node a* for each known destination *node b*. Upon two nodes meeting, this information is used to determine which message to forward to the other node, and which messages to request from the other node. The delivery predictability is based on the frequency of nodes encountered, and is updated every time a node encounter occurs. The delivery predictability held by *node a* for *node b* is given below where  $P_{init} \in [0, 1]$ , is an input initialisation parameter:

$$P_{(a,b)} = P_{(a,b)old} + (1 - P_{(a,b)old}) \times P_{init} \quad (3.1)$$

If a pair of nodes have not encountered each other in a while, then they are less likely to be good candidates to forward the message to. Consequently, this metric is then decreased over time using a decaying factor. The decay predictability  $P_{(a,b)}$  is given below, where  $\gamma \in [0, 1]$ , is the aging constant and  $k$  is the number of time units elapsed since the last encounter.

$$P_{(a,b)} = P_{(a,b)old} \times \gamma^k \quad (3.2)$$

The appropriate time unit used differs depending on the application and the expected delays in the network. The delivery predictability is not based solely on direct encounters, but also on indirect encounters. If *node a* frequently meets *node b* and *node b* frequently meets *node c*, *node a* has an indirect delivery predictability to *node c* through *node b*. The transitive delivery predictability is shown below where  $\beta \in [0, 1]$  is a scaling constant that determines how large an impact transitivity should have on delivery predictability.

$$P_{(a,c)} = P_{(a,c)old} + (1 - P_{(a,c)old}) \times P_{(a,b)} \times P_{(b,c)} \times \beta \quad (3.3)$$

Similarly to epidemic routing, when two nodes meet they exchange a summary vector containing a list of messages they are carrying, along with the current node's delivery predictability for each message. The delivery predictability is used to determine which messages should be exchanged where each message is forwarded to the node holding the higher delivery predictability value. This method has shown to be promising with simulation results showing similar delivery performance to Epidemic Routing and improved performance when the queue size is limited.

ZebraNet was designed to collect sensor data from collars carried by zebras [67]. A mobile base station is located on a vehicle, which is driven around the area on a daily basis. The history-based protocol defines the likelihood of a node being in range of the base station by assigning each node a hierarchy level based on its past success rate at transferring data to the base station. The higher the hierarchy level, the higher the probability that the node will come into range with the base station. Each node requests the hierarchy level of all other nodes within range and transfers its data to the responding node with the highest hierarchy level. When a node comes within range of the base station its hierarchy level is raised. Conversely, when a node is out of range of the base station the hierarchy level decays.

Tan et al. present an algorithm for shortest path routing where they calculate a delivery probability based on past node encounters in order to calculate the cost of the route [111]. The probability of a link between node  $i$  and  $j$  is given by:

$$P_{i,j} = \frac{T_{i,j}}{T_W} \quad (3.4)$$

where  $T_{i,j}$  is the amount of time that a link has existed between node  $i$  and node  $j$  over a previous time frame  $T_W$ . This probability is stored for each link and upon meeting, nodes exchange probabilities along with a sequence number in order to replace old probabilities with new. On receiving a link probability update message, a node updates its local table and uses Dijkstra's shortest path algorithm to calculate the *expected path length* to all other nodes in the network, where the weight of each link is the reciprocal of  $P_{i,j}$ . The smaller the *expected path length*, the higher the probability of delivery. In order to route a message from *node a* to *node d*, *node a* upon meeting *node b* compares the *expected path length* held by *node b* with its own. If the *expected path length* of *node b* is smaller, then the message is transferred to *node b*.

Jones et al. [66] explore a distributed version of the contact summary oracle presented in

[62] discussed in section 3.1.3. Where global knowledge is assumed in [62] to calculate the average expected delay, Jones et al. use previously observed contacts. The average expected delay is referred to as the *minimum estimated expected delay* (MEED) and is based on the amount of time a link has existed over a sliding window of time. The MEED is calculated by each node and stored in a routing table. Routing information is distributed in an epidemic fashion where upon meeting two nodes exchange a summary vector of their routing table as well as updated values of their routing information and the information received by other nodes. With the use of this information a path to the destination can be calculated using Dijkstra's shortest path algorithm as done in [62]. The route is recalculated at each intermediate node, in order to make use of the most recent information.

Burgess et al. present MaxProp where each node is ranked with a *delivery likelihood*, a probability based on contact history [20]. Initially the *delivery likelihood* of node  $i$  for each node  $j$  is given by:

$$f_j^i = \frac{1}{|s| - 1} \quad (3.5)$$

where  $s$  is the total number of nodes encountered so far. Each time node  $i$  encounters node  $j$  this *delivery likelihood* is incremented by 1. Then all the probabilities are normalised in order to sum to a total of 1 which is referred to as incremental averaging.

$$f_j^i = \frac{f_j^{i, old}}{\sum_{j=1}^s f_j^i} \quad (3.6)$$

The result is that infrequently encountered nodes obtain lower *delivery likelihood* probabilities over time. Messages stored in the buffer of the node are ranked according to their *delivery likelihood*. Acknowledgments are sent to all nodes upon delivery and delivered messages are then deleted from the buffers. Whenever two nodes meet, messages are exchanged with priority given to messages that have not traveled far in the network, followed by the remaining messages in order of highest *delivery likelihood*. Priority is given to messages that have not traveled far because the authors observed that transmitting in order of *delivery likelihood* favours the forwarding of messages with a high *delivery likelihood*, while others never get propagated. If the connection lasts long enough all messages are transmitted, and therefore similar to Epidemic

Routing. However, the authors cater for the reality that connections may not last long enough in order to complete a full transfer of data between nodes. As such, information is exchanged in order of importance based on the probability that the encountered node will be able to deliver the message.

## Discussion

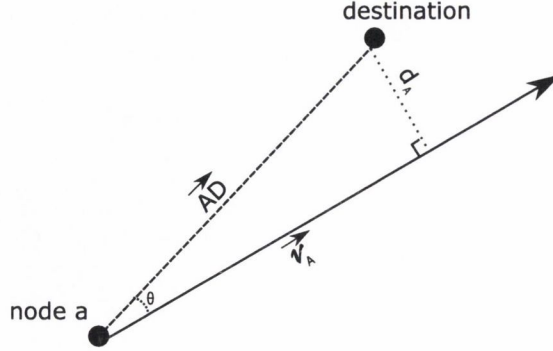
Encounter-based schemes improve upon Epidemic-based schemes in that messages are not blindly forwarded to any encountered nodes. In a real environment nodes do not meet randomly and therefore encounter-based schemes attempt to predict which node is most likely to encounter the destination. With the exception of [20] the above solutions assume a limited number of copies of each message on the network meaning resource consumption is much reduced compared to epidemic-based schemes. Consequently, when resources are limited encounter-based schemes perform better than epidemic-based schemes in realistic movement scenarios. Additionally, encounter-based schemes have the same benefit of Epidemic-based schemes, in that they require no global knowledge in the network, unlike deterministic solutions.

### 3.3.2 Location-Based

Lebrun et al. in [70] propose a location-based delay-tolerant routing scheme to deliver data to static destinations, whose position is globally known through mobile vehicles equipped with GPS receivers. Nodes transmit periodic HELLO messages and upon receiving a HELLO the receiving node responds with its trajectory information. The sending node then compares the responding node's trajectory with its own.

The trajectory information is used to predict if the node is moving towards or away from the destination, and the closest the node will be to the destination in the future if the node continues along the current trajectory in a straight line. In figure 3.3  $\vec{v}_c$  is the motion vector of node  $a$ . The second vector,  $\vec{AD}$  is drawn from the node to the destination. The angle between the two vectors is given by:

$$\theta = \cos^{-1}\left(\frac{\vec{AD} \cdot \vec{v}_c}{|\vec{AD}||v_c|}\right) \quad (3.7)$$



**Fig. 3.3:** MoVe vector calculation derived from [70]

The closest predicted distance  $d_a$  is given by  $\sin(\theta) \times \overline{AD}$ . If  $\theta < 90$ , the node is moving towards the destination, otherwise it is moving away. A node forwards the message to another node if the responding node is heading towards the destination, and the predicted distance from the destination is less than the distance from the current node to the destination.

Leguay et al. introduce a virtual coordinate system where the node coordinates are composed of a set of probabilities, each representing the chance that a node will be found in a specific location [71, 72]. The aim is to forward the message to the node that has the most similar mobility profile. When two nodes meet, they exchange mobility profiles and the euclidean distance is used to compute similarity between the encountered node and the destination node. If the encountered node's mobility profile is closer to the destination node's mobility profile than that of the current node, then the message is forwarded. All nodes are aware of their own and other node's mobility patterns.

Similarly, Ghosh et al. propose exploiting the fact that nodes tend to move between a small set of locations, which they refer to as 'hubs' [46]. The movement between these hubs form 'sociological orbits'. A list of 'hubs' specific to each users movement profile is assumed to be available to each node on the network in the form of a 'probabilistic orbit' which defines the probability with which a given node will visit a given hub. Messages destined for a specific node are routed towards one of these user specific 'hubs'. The delivery probability of a node delivering a message to a specific node is based on a combination of: the probability  $P_{n_i h_j}^t$  that node  $n_i$  will travel to hub  $h_j$ , the next hub  $h(n_i)$  the node will visit which is assumed to be known according to the probabilistic orbit profile and the probability  $P_{n_i n_k}^c(h_j)$  that node  $n_i$  will have contact with destination node  $n_j$  at hub  $h_j$ . The delivery probability  $P_{n_i n_h}^d$ , of node

$n_i$  for hub  $h_j$  is given by:

$$P_{n_i h_i}^d = \max(P_{n_i n_k}^t, \max(P_{n_i n_k}^c h(n_i)) * P_{n_k n_j}^t)$$

The sending node distributes  $k/2$  copies of the message to the  $k/2$  nodes that have a higher probability of visiting the most visited hub of the destination node, and  $k/2$  copies to the  $k/2$  neighbours that have a higher probability of visiting the second most visited hub of the destination node, where  $k$  is a replication parameter. In this solution the mobility profiles are assumed to be available to all other nodes. In this manner the notion of routing is changed from routing to a specific node, but to a location the node is considered likely to visit.

## Discussion

Routing in disconnected networks using location-based schemes is not as well researched as that of encounter-based schemes. This is primarily due to the difficulty in capturing location data in an efficient manner. The use of simple velocity and direction information as used in [70] is useful as a short term predictor to the general destination of a node, but under realistic mobility scenarios it cannot provide a good indicator for long term future locations.

The solution proposed by Leguay et al. provides mechanism for routing messages to nodes with the most similar mobility profile [71, 72]. A problem that may occur with such a scheme is where nodes may visit similar locations but at different times, meaning the opportunity to exchange data may not occur. As such routing purely based on similar location profiles may have limited usefulness depending on the mobility of the nodes in the network. Ghosh et al. provide a similar notion of routing based on locations nodes frequent [46]. Both solutions simplify the problem of capturing location information by assuming key locations to each user are pre-identified. Additionally they assume that this location information is globally available to all nodes.

In other work by Ghosh et al. the problem of identifying these hubs is explored by logging user visits to buildings on a university campus [45]. They found that hub identification was possible by analysing the frequency distribution of these visits. However, the process takes approximately a week to generate a hub distribution list, and following that the hub must be updated approximately once every two weeks. Though promising, the time required to identify specific locations and distributing them dynamically throughout the network, remains an open



issue. Due to the training period of mobility profiles and the distribution of these mobility profiles hub lists, these solutions are most appropriate for long term networks where the node population is reasonably static. As nodes that are part of the network for a short time will be unable to participate without a mobility profile that is known locally, and by other nodes on the network.

### 3.3.3 Utility-Based

Chen et al. propose a utility-based scheme in order to move the message to a node *closer* than the current node carrying the message [26]. The utility of a node reflects the probability that the node will meet the destination before the message becomes invalid. Each node calculates their own utility for a given message on demand. Nodes repeatedly initiate a probe phase after a given timeout, which is referred to as the rediscovery interval. During the probe phase the node broadcasts a request containing the destination id, the time to live of the message and the utility of the requesting node. The receiving node then calculates its own utility and if it is higher than that of the requesting node, the encountered node sends a response with its own utility value. The message is then transferred to the node encountered with the highest utility value.

The utility calculation is based on five components:

- List of nodes most recently encountered

$$U_{MRN} = \left(1 - \frac{CurrentTime - TimeLastNoticed_D}{TimeOut_m}\right) \times 100 \quad (3.8)$$

- where the  $TimeLastNotice_D$  is the time of the last encounter of the destination node  $D$  and the  $TimeOut_m$  is the time after which the message  $m$  is no longer valid in the system.

- List of nodes most frequently encountered

$$U_{MFN} = \left(1 - \frac{CurrentTime - FirstTimeNoticed_D}{NumTimesNoticed_D \times TimeOut_m}\right) \times 100 \quad (3.9)$$

- Future plan of the node

- This utility component is only available if an explicit calendar is available on the node, where information can be extracted as to the next expected meeting time of

node  $D$ . Alternatively, the node may hold a list of nodes which are expected to be met in a given time frame. If such information is available the calendar utility value is given as:

$$U_{CAL} = \left(1 - \frac{NextMeetingTime_D - CurrentTime}{TimeOut_m}\right) \times 100 \quad (3.10)$$

- The power level

$$U_{power} = \left(1 - \frac{TimeOut_m}{EstimatedRemainingPower}\right) \times 100 \quad (3.11)$$

- The rediscovery interval

- The rediscovery interval reflects the frequency with which a node initiates a probe phase searching for a next hop. This frequency is dependent on the environment, where the frequency is increased when new encounters become more frequent. The premise is to forward messages to more active nodes that are currently forming new contacts.

$$U_{RDI} = \frac{MIN_{RDI}}{RDI} \times 100 \quad (3.12)$$

- Where the  $MIN_{RDI}$  is the minimum  $RDI$  for that node and  $RDI$  is the current rediscovery interval for that node.

The utility value is calculated by combining the above utility components. Each component is assigned a different weight depending on the application area in order to give higher weight to the components deemed most useful for the given application domain.

Musolesi et al. present context aware routing for sensor networks (SCAR) and choose multiple carriers among neighbours of the data source based on their history of encounters, mobility and resources [87]. Each node evaluates their rate of change of connectivity, their history of encountering the destination node and their battery resources. These context attributes are combined in order to determine a delivery probability.

The first context attribute considered is the change degree of connectivity  $\hat{U}_{cdc}(s_i)$  which is the number of connections and disconnections that a node experienced over the last time period  $[t - 1, t]$ , which is normalised by the number of hosts that have been in reach during that time period. This attribute attempts to measure the relative mobility, and hence the probability that a node will meet different nodes in a given time period.

$$U_{cdc}(s_i) = \frac{|N_{i_{t-1}} \cup N_{i_t}| - |N_{i_{t-1}} \cap N_{i_t}|}{|N_{i_{t-1}} \cup N_{i_t}|} \quad (3.13)$$

Where  $N_{i_t}$  is the number of reachable hosts at time  $t$ . The next attribute considered is the history of the node encountering the destination which they refer to as the co-location attribute  $\hat{U}_{coloc}(s_i)$ . The value is high if a node has recently encountered the destination. The co-location attribute measures the percentage of time that two nodes have been in reach. It is calculated by periodically running a Kalman filtering process, assuming that the value is 1 if the node is currently in reach or 0 if not. The resultant predicted values will be in the range  $[0, 1]$  reflecting an estimation of the probability of being in reach of the node in the future.

The final attribute is the estimation of the future battery level of the node  $\hat{U}_{bat}(s_i)$ . The value 1 corresponds to a full battery, and 0 corresponds to an empty one. The calculation of  $\hat{U}_{coloc}(s_i)$  and  $\hat{U}_{bat}(s_i)$  is achieved using a Kalman filter in order to predict the future values.

The context attributes considered are combined to present the best trade-off among the varying attributes to determine the best carrier. By considering the three attributes and their utility functions, they formulate the problem as a multiple criteria decision problem with three goals, where  $U(s_i)$  represents the utility of node  $s_i$ . Weights are assigned to each attribute in order to reflect the relevance of each goal where  $w_{cdc}$ ,  $w_{coloc}$  and  $w_{bat}$  are the significance weights.

$$U(s_i) = w_{cdc}\hat{U}_{cdc}(s_i) + w_{coloc}\hat{U}_{coloc}(s_i) + w_{bat}\hat{U}_{bat}(s_i) \quad (3.14)$$

Upon encountering other nodes, the messages are transferred to nodes with a greater probability of delivering to the destination node. Probability calculations are performed locally and nodes only exchange data in regards to their delivery probabilities and available buffer space. Nodes maintain a list of neighbouring nodes, including themselves, in the order of their delivery probability. The data is replicated to the  $R$  nodes with the highest delivery probability, or  $R - 1$  if the current node is included in the list of  $R$  nodes. The replica of the data is sent to the node with the highest probability value and marked as master copy. The other replicas are distributed to the remaining nodes and marked as backup copies. The backup copies are deleted when the node buffer is full, but the master copy is only to be deleted when it is delivered to the destination. The value of  $R$  is left up to the developers to determine and is based on priority,

where higher priority data is given a higher value of  $R$ , thus increasing delivery reliability and decreasing end-to-end latency.

Spyropoulos et al. [106] use an analysis of random walk combined with a set of timers recording the time since all known nodes are encountered. Based on a random walk model they utilise timers to predict hitting times, which is the estimated time when a node will arrive at a given location or meet a given node. This information is then used to compute a utility value. They propose a hybrid approach called ‘seek and focus’ where the message is initially forwarded until it meets a node with a utility value above a certain threshold. From there on, the message is routed by transferring the message to the node with a higher utility value. The premise behind the ‘seek and focus’ hybrid approach is that the source node may have to wait a long time until it finds a next hop with a higher utility, and therefore by using the random walk initially this slow-start phase is avoided. The simulation results show that the end-to-end delivery delay is higher using the hybrid approach compared to a pure utility-based approach, however, the total number of message delivered is improved. It should be noted that the simulation utilises a random walk mobility model for all nodes, which is the basis for the utility metric. Consequently, it is unclear how well this utility metric will translate in a more realistic movement scenario.

## **Discussion**

As can be seen from the above related work, utility-based forwarding approaches attempt to take into account more than encounters and location information. They also take into account available node resources such as battery power and available buffer space. As such, many aspects of a node are considered to determine the nodes potential usefulness in relation to the delivery of a given message. A good example where this could prove beneficial is if a node has low battery power remaining, but has a high probability of delivering a message, the usefulness of forwarding the message to the node is significantly reduced. Additionally, utility may prove useful if there are a number of ‘super-nodes’ in the network with renewable energy and very high memory capacity. Though such nodes may not have a high chance of meeting the destination node, the message has a high chance of being stored on the network without being lost until a suitable intermediate node can be found.

### 3.3.4 Conclusion

History and prediction-based schemes are highly useful in reducing the number of copies of a message on the network required to ensure successful delivery. This is accomplished by forwarding messages only to nodes that provide a good chance of delivering the message. As a result, the load on resources is distributed throughout the network rather than consumed by all nodes holding and forwarding each message .

History and prediction-based schemes assume the existence of no infrastructure and attempt to reduce the amount of bandwidth required. The variable node population means that new nodes may take a while before they build up a useful history to determine a delivery probability. In the case of utility-based schemes, message loss through node failure as a result of battery depletion can be reduced by taking into account the battery power remaining before forwarding a message to a node.

History and prediction-based schemes assume no control over node movements. Assumptions are made in regards to node future movements based on the fact that history and prediction-based schemes assume that a node does not move randomly. As a result, encounter and movement patterns that have been seen before will most likely occur again. The network graph can be highly disconnected and dynamic, and no topology information is assumed other than that observed by the node. As a result, history and prediction-based schemes are well suited to address the challenges of mobility as discussed in section 2.2.

Battery power and buffer space is used more sparingly than in epidemic-based schemes. However, depending on the node population, memory is required to store history and prediction information, which is not the case for epidemic routing. The storage capacity required can be reduced by potentially only storing information about nodes that are deemed most useful, where nodes that are rarely met being assumed to be equivalent to a node never met.

Delivery reliability and end-to-end latency is inevitably lower than that of epidemic routing which finds the optimal path. However, this is a trade-off in order to provide better node longevity and more efficient buffer usage.

### 3.4 Approaches Based on Movement Control

The previously discussed approaches assume no control over node movements and passively wait for connections between nodes to arise. Some recent research proposes exploiting and controlling node mobility in order to provide guaranteed connections. For example Li and Rus aim to guarantee message transmission in minimal time [74]. By allowing mobile hosts to actively modify their trajectories to transmit messages. Information about the motion of the destination host is used to determine how the message can be sent by the co-operation of intermediate hosts. The authors assume an ad hoc network where the trajectory of each node is approximately known.

A trajectory is computed for sending a message from A to B by asking intermediate nodes to change their trajectory in order to complete a routing path between nodes A and B. The aim is to minimise the trajectory modifications while delivering the message with the minimum end-to-end delay. The protocol is an application layer protocol.

There are two versions of the protocol presented. The first assumes information about the motions and locations of all nodes is known to all hosts. The second does not assume that movements of the nodes is known. Nodes inform other nodes of their current position.

When a host needs to transmit a message, it computes the optimum relay path which is a sequence of intermediate hosts that can relay the message to the destination. The optimum relay path is computed by first calculating the amount of time required for the sending node to move within communication range of the destination node, which is marked as the current optimum path. Next, the neighbouring node that requires the least amount of time to move within communication range is selected as an intermediate node. The time cost of each path is now recomputed using the intermediate node as a relay, if the new path is less than the current optimum path then it is replaced with the indirect path. This continues until an optimum path has been computed for all destination nodes.

In the second scenario it is not assumed that movements of the nodes is known. Instead, nodes inform other nodes of their current position. Communication is achieved by constructing a minimum spanning tree which contains the shortest edges in the graph that provide full connectivity in the graph. Each host has the responsibility of updating its location by informing all the hosts connected to it in the minimum spanning tree. This work assumes that the network is

almost fully connected. It is not clear how the minimum spanning tree is determined or what happens if no such minimum spanning tree is found. Additionally their protocol considers the propagation of only one message (end-to-end) each time, as it has not been defined how to deal with multiple, contradicting trajectory modification requests.

Zhao et al. propose Message Ferrying for data delivery in sparse networks [126]. Message Ferrying is a proactive approach that utilises a set of special mobile nodes called message ferries to provide communication for nodes in the network. A message ferry moves around the deployment area and carries data between nodes. By inserting designated message ferries into the network they introduce non-randomness in the movement of nodes and exploit this non-randomness to help deliver data. They present two schemes, node-initiated and message ferry-initiated. In the message ferry initiated scheme, ferries travel through the network along a known route. Nodes can move towards a message ferry at a designated time to exchange messages. The message ferry route is calculated using the solution to the traveling salesman problem to compute the shortest route using the average delay time as the weighting rather than the shortest route. In the node-initiated scheme a node may initiate communication by sending a packet to notify a message ferry using a long range radio; the message ferry then sets a trajectory in order to intercept the node and exchange data. This protocol assumes that mobile nodes can control their movements, and that there are a number of designated message ferries to travel around the area.

The Message Ferrying solution involves one message ferry providing communication between nodes. Zhao et al. went on to explore the use of multiple message ferries in a network [127]. Message ferries communicate with both nodes and each other, either directly or through the use of stationary relay nodes, in order to deliver data. The multiple data ferry routes are computed in order to allow communication between each pair of nodes with the minimum possible delay and optimum load balancing. Nodes are assigned to specific ferries

It is important to note that in both Message Ferrying schemes, the complete node population and their locations needs to be known to the message ferries in order to compute the route. Essentially the problem of connecting a relatively static disconnected network graph is solved through the insertion of deterministic movement of a mobile message ferries.

### 3.4.1 Conclusion

The use of controlled node movements in order to delivery data limits the application areas for the solution. It could be argued that the presence of specific nodes, such as in [126] is essentially a form of infrastructure. The existence of a base-station is replaced by the existence of mobile nodes that act as a mobile base station, visiting the vicinity of each node. Both solutions discussed in this section, assume the knowledge of the complete node population meaning nodes may not join and leave unexpectedly. These assumptions mean that bandwidth usage is limited and efficient.

The assumption of control of node movements also includes the assumption of knowledge of future node movements. The network graph is disconnected, however, it is not highly dynamic. Li and Rus communicate node location changes throughout the network meaning that if nodes are highly mobile and dynamic the network will fail [74] . Similarly, the Message Ferrying project assumes that the general location of nodes are known to the message ferries in order to guarantee that the message ferry will be able to visit all nodes [126]. The assumption the node locations are known and that node movements are infrequent limits the environments that these solutions apply to. In a highly dynamic mobile environment these solutions cannot provide message delivery.

## 3.5 Coding-Based Approaches

Coding-based techniques utilise message encoding techniques before forwarding messages. Erasure-coding techniques provides redundancy by encoding the contents of a message over a number of different packets. Network coding techniques combine multiple messages into a single message in order to reduce transmission overhead.

### 3.5.1 Erasure-Coding

Erasure-coding based schemes provide redundancy by encoding a single message into a large number of code blocks, such that if a fraction  $\frac{1}{r}$  or more of the code blocks are received, then the message can be reconstructed at the destination. This technique has been primarily proposed for multicast or broadcast protocols, where retransmission can be avoided in the case of nodes



experiencing packet loss [91]. Erasure-coding differ from replication schemes where entire copies of a message are simultaneously propagated over multiple paths.

Wang et al. propose the use of erasure-coding for opportunistic networks [117]. A message  $M$  is encoded into  $k$  code blocks, where  $k$  is a constant. These blocks are then equally distributed among the first  $kr$  nodes met, where  $r$  is the replication factor. To compare it to simple replication if  $k = 1$  then this scheme is replication with  $r$  copies of the message. The code blocks are distributed to the first  $kr$  nodes met, these code blocks are then carried by each node until they meet the destination. When the destination receives  $\frac{1}{r}$  of the code blocks the message  $M$  can be reconstructed. The aim is to utilise more nodes to carry portions of the data making the network is more robust to failures or bad routing choices. However, it also means that a node must wait longer in order to meet a sufficient number of nodes to carry the code blocks. This mechanism has been shown to converge to a constant average delay, however, it does not achieve minimum delay compared to replication or history-based schemes.

Liao et al. build upon this method by taking into account a node's probability to deliver when distributing the code blocks [75]. The probability to deliver value is a simple average contact frequency (ACF), which is defined as the number of encounters between node  $i$  and node  $j$  within a given time frame. The source node encodes the message into  $kr$  blocks. Upon encountering another node  $B$ , if the current node  $A$  holds more than  $k$  code blocks, the code blocks are distributed between the nodes in proportion to the probability of a node to deliver where the number of code blocks  $m_B$  to be transferred to node  $B$  from source node  $A$  is given by:

$$m_B = m_A \frac{T_{B,D}}{T_{A,D} + T_{B,D}} \quad (3.15)$$

where  $m_A$  is the number of code blocks held by node  $A$  and  $T_{A,D}$  and  $T_{B,D}$  is the probability to deliver to destination node  $D$  held by node  $A$  and node  $B$  respectively. When the current node is holding  $k$  or less code blocks, and the encountered node has a higher probability to deliver, then all the code blocks are transferred to the encountered node.

Jain et al. perform a theoretical analysis of using erasure-coding in DTNs under varying path reliability [61]. They assume that paths and the path reliability of each path are globally known. Path failures are then simulated and the results of the analysis show, that when path reliability is low in the network, then simple replication performs better than erasure-coding. When path

reliability is approximately 60% the benefits of erasure-coding are only evident when there are many paths available on which to split the code blocks. Finally, when path reliability is high both erasure-coding and replication perform well with erasure-coding outperforming replication with near perfect results. These results show that erasure-coding, though potentially improving delivery performance, the benefit is very much dependent on the underlying network and the path reliability. In DDTMs it is quite unlikely that there will be many, high probability paths to a destination, as such erasure-coding techniques may not be beneficial.

### Discussion

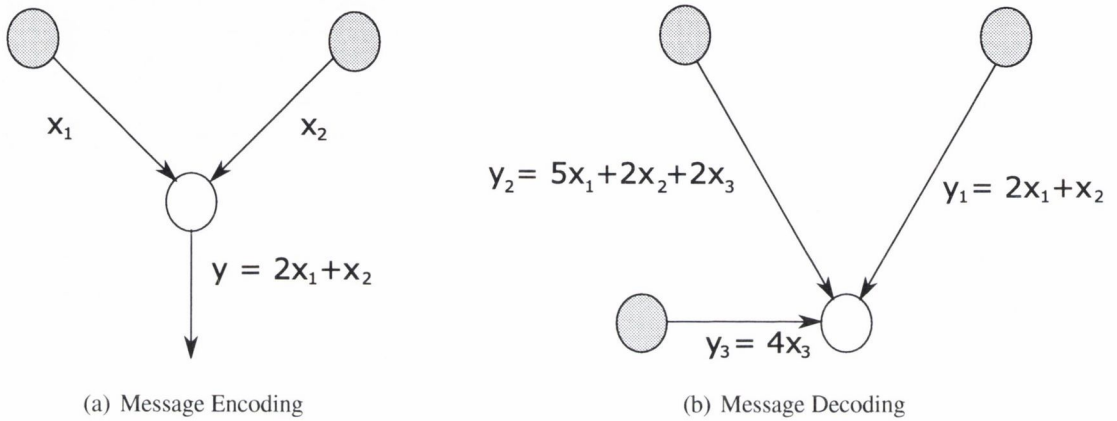
One of the advantages of erasure coding techniques by distributing code blocks over a large number of relays is that sending only a fraction of the code blocks over each relay allows control of the routing overhead in terms of bytes transmitted. However, this scheme results in a higher number of transmissions per messages, and establishing a connection is costly in terms of battery power. As such, erasure-coding techniques may result in heavy power consumption in the network.

### 3.5.2 Network Coding

Network coding is a mechanism proposed to improve throughput utilisation of a given network topology. Network coding originated in order to maximise the data sent in a single transmission. Ahlswede et al. showed that intermediate nodes in a network mix information from different flows in order to achieve broadcast capacity [7]. As an example, assume node A and node C wish to communicate through node B. Node A sends a message  $M_{a,c}$  to node B and node C sends a message  $M_{c,a}$  to node B. Node B then combines these two messages and instead of individually forwarding the messages, it broadcasts the combined message  $M_{a,c} \oplus M_{c,a}$ . Node A has the message  $M_{a,c}$  and thus can decode the message  $M_{c,a}$  from the combined encoded message, node C can do likewise. The aim is to utilise broadcast in order to be able to deliver different messages to different receivers with one transmission.

Widmer et al. propose using network coding in extreme networks [121]. Nodes encode the contents of several messages they receive into a single message which is then broadcast to neighbouring nodes with a probability of  $p$ . Figure 3.4 a) shows an intermediate node, upon receiving message  $x_1$  and  $x_2$  encoding both messages into a single message  $y$  using an

encoding vector  $g = (2, 1)$ . A node forwarding an encoded message includes the encoding vector  $g$ . Figure 3.4 b) shows a destination node receiving three encoded messages. When enough packets are received at the destination node, the original message can be used to decode the message using Gaussian elimination.



**Fig. 3.4:** Network Encoding [120]

### Discussion

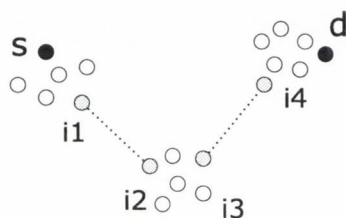
The result is that an encoded version of all messages are eventually delivered to all nodes. This mechanism ensures high delivery reliability and low latency, however, it results in high buffer occupancy and high bandwidth consumption. Additionally, this mechanism makes encoded data segments indistinguishable from each other, and though this solution is appropriate for blind forwarding, it is unsuitable for applications where stale data needs to be removed. If an application is only interested in up-to-date data there needs to be a mechanism for removing obsolete data. This can only be done by a node after decoding, which is time and resource consuming. It is not possible to decode partial sets of data and the limited storage available in nodes may prevent it from saving a full set of data.

## 3.6 Discussion of DDTM routing Solutions

Deterministic solutions have limited application areas due the strong assumptions of global knowledge and deterministic behaviour. The stochastic solutions presented in the this chapter can be classified into two general categories: blind forwarding and metric-based. Metric-based

schemes are based on previous encounters, location information or a node utility. Blind forwarding schemes require more resources. In order to provide reliability of delivery they either require a great deal of replication which results in heavy bandwidth, battery or memory consumption, or require nodes to carry data for a great deal of time until they come within range of an appropriate delivery node. Metric-based schemes have the benefit that they require less resources and generally require less replication in order to achieve delivery.

A number of contact-based solutions for routing in disconnected networks are based on a node's observed encounters where data is routed to nodes with the highest 'probability to deliver' to a destination node. Such metrics are typically based on either direct or indirect observed encounters [20, 80, 68]. These schemes are well suited to dynamic networks where topology information is limited.



**Fig. 3.5:** Disconnected Clusters

However, some networks may consist of cliques where metrics based on direct or indirect encounters may not find a suitable carrier for the message. Consider three disconnected clusters in figure 3.5. Source node  $s$  wishes to send a message to destination node  $d$ . However, node  $s$  is involved in a highly cliquish cluster in which none of the nodes have directly or indirectly met destination node  $d$ . This makes the decision of selecting a node to forward data difficult. The three clusters are linked by bridging ties from  $i1$  to  $i2$  and from  $i3$  to  $i4$ . A path exists between the three clusters using intermediate nodes  $i1$ ,  $i2$ ,  $i3$  and  $i4$  which form bridges between the three clusters. Weak acquaintance ties of  $i1$ - $i2$  and  $i3$ - $i4$ , illustrated by the dashed lines, become a crucial bridge between the three tightly connected groups, and these groups would not be connected if not for the existence of these weak ties. Motivated by this observation the next chapter explores social network analysis theory in order to identify such 'bridges' and the identification of nodes that reside within the same cluster as the destination node, along with evaluating the strength of node relationships.

## Chapter 4

### Social Networks for Information Flow

In a disconnected environment, data must be forwarded using node encounters in order to deliver data to a destination. The problem of message delivery in disconnected delay-tolerant networks can be modeled as the flow of information over a dynamic network graph with time-varying links. The majority of protocols discussed in chapter 3 have been evaluated using homogeneous movement patterns of nodes where each node is generally as good a candidate to encounter the destination node as another. In a real world environment this is not the case. Current research supports the observation that encounters between nodes in real environments do not occur randomly [58] and that nodes do not have an equal probability of encountering a set of nodes. In fact, one study by Hsu and Helmy observed that nodes never encountered more than 50% of the overall population [56]. As a consequence, not all nodes are equally likely to encounter each other, and nodes need to assess the probability that they will encounter the destination node. Additionally, Hsu and Helmy performed an analysis on real world encounters based on network traffic traces of different university campus wireless networks [55]. Their analysis found that node encounters are sufficient to build a connected relationship graph, which is a small world graph. Therefore, social analysis techniques are promising for estimating the social structure of node encounters in a number of classes of disconnected delay-tolerant MANETs.

Social networks exhibit the small world phenomenon which comes from the observation that individuals are often linked by a short chain of acquaintances. The classic example is Milgrams' 1967 experiment, where 60 letters were sent to various people located in Nebraska to be delivered to a stockbroker located in Boston [90]. The letters could only be forwarded to

someone whom the current letter holder knew by first name and who was assumed to be more likely than the current holder to know the person to whom the letters were addressed. The results showed that the median chain length of intermediate letter holders was approximately 6, giving rise to the notion of ‘six degrees of separation’. Milgram’s experiment showed that the characteristic path length in the real world can be short. Of particular interest, however, is that the participants did not send on the letters to the next participant randomly, but sent the letter to a person they perceived might be a good carrier for the message based on their own local information. In order to harness the benefits of a small world networks for the purposes of message delivery, a mechanism for intelligently selecting good carriers based on local information must be explored. In DDTMs we wish to exploit the underlying social network structure in order to provide information flow from source to destination [30]. The remainder of this chapter reviews network theory that may be applied to social networks along with social network analysis techniques. These techniques have yet to be applied to the context of routing in DDTMs. Social network analysis is the study of relationships between entities, and on the patterns and implications of these relationships. Graphs may be used to represent the relational structure of social networks in a natural manner. Each of the nodes may be represented by a vertex of a graph. Relationships between nodes may be represented as edges of the graph.

## **4.1 Network Centrality for Information Flow**

Centrality in graph theory and network analysis is a quantification of the relative importance of a vertex within the graph (for example, how important a person is within a social network). The centrality of a node in a network is a measure of the structural importance of the node, typically, a central node has a stronger capability of connecting other network members. There are several ways to measure centrality. Three widely used centrality measures are Freeman’s degree, closeness, and betweenness measures [41, 42].

‘Degree’ centrality is measured as the number of direct ties that involve a given node [42]. A node with high degree centrality maintains contacts with numerous other network nodes. Such nodes can be seen as popular nodes with large numbers of links to others. As such, a central node occupies a structural position (network location) that may act as a conduit for information exchange. In contrast, peripheral nodes maintain few or no relations and thus are located at the

margins of the network. Degree centrality for a given node  $p_i$  where  $a(p_i, p_k) = 1$ , if a direct link exists between  $p_i$  and  $p_k$  is calculated as:

$$C_D(p_i) = \sum_{k=1}^N a(p_i, p_k) \quad (4.1)$$

‘Closeness’ centrality measures the reciprocal of the mean geodesic distance  $d(p_i, p_k)$ , which is the shortest path between a node  $p_i$  and all other reachable nodes [42]. Closeness centrality can be regarded as a measure of how long it will take information to spread from a given node to other nodes in the network [95]. Closeness centrality for a given node, where  $N$  is the number of nodes in the network, is calculated as:

$$C_C(p_i) = \frac{N - 1}{\sum_{k=1}^N d(p_i, p_k)} \quad (4.2)$$

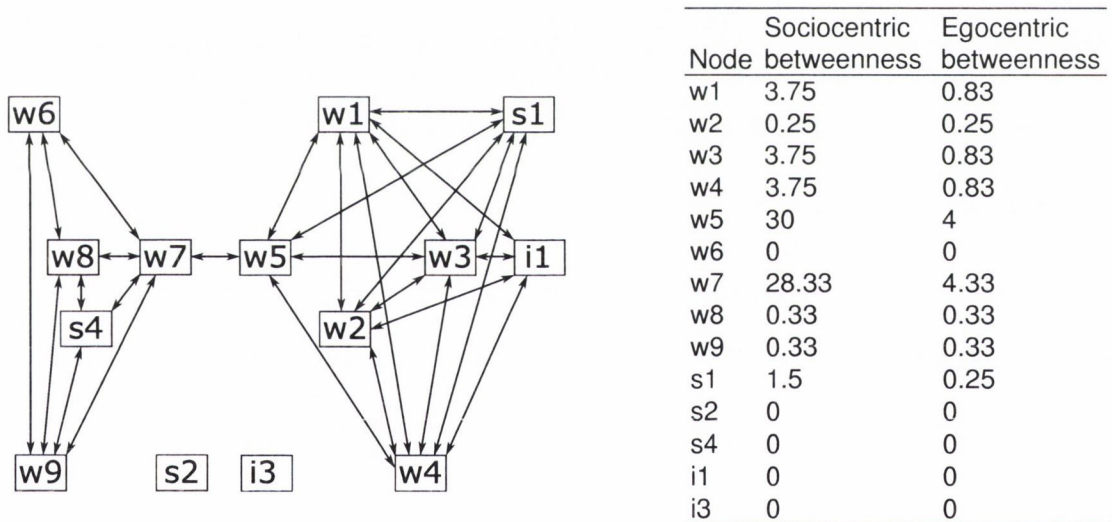
‘Betweenness’ centrality measures the extent to which a node lies on the geodesic paths linking other nodes [41, 42]. Betweenness centrality can be regarded as a measure of the extent to which a node has control over information flowing between others [95]. A node with a high betweenness centrality has a capacity to facilitate interactions between the nodes that it links. In our case, it can be regarded as how much a node can facilitate communication to other nodes in the network. Betweenness centrality, where  $g_{jk}$  is the total number of geodesic paths linking  $p_j$  and  $p_k$ , and  $g_{jk}(p_i)$  is the number of those geodesic paths that include  $p_i$  is calculated as:

$$C_B(p_i) = \sum_{j=1}^N \sum_{k=1}^{j-1} \frac{g_{jk}(p_i)}{g_{jk}} \quad (4.3)$$

Borgatti analyses centrality measures for flow processes in network graphs [14]. A number of different flow processes are considered, such as package delivery, gossip and infection. He then analyses each centrality measure in order to evaluate the appropriateness of each measure for different flow processes. His analysis showed that betweenness centrality and closeness centrality were the most appropriate metrics for message transfer that can be modeled as a package delivery.

Freeman’s centrality metrics are based on analysis of a complete and bounded network, which is sometimes referred to as a sociocentric network. These metrics become difficult to evaluate in networks with a large node population because they require complete knowledge of the network topology. For this reason the concept of ‘ego networks’ has been introduced.

Ego networks can be defined as a network consisting of a single actor (ego) together with the actors they are connected to (alters) and all the links among those alters. Consequently, ego network analysis can be performed locally by individual nodes without complete knowledge of the entire network. Marsden introduces centrality measures calculated using ego networks and compares these to Freeman’s centrality measures of a sociocentric network [85]. Degree centrality can easily be measured for an ego network where it is a simple count of the number of contacts. Closeness centrality is uninformative in an ego network, since by definition an ego network only considers nodes directly related to the ego node, consequently by definition the hop distance from the ego node to all other nodes in the ego network is 1. On the other hand, betweenness centrality in ego networks has shown to be quite a good measure when compared to that of the sociocentric measure. Marsden calculates the egocentric and the sociocentric betweenness centrality measure for the network shown in figure 4.1.



**Fig. 4.1:** Bank Wiring Room network sociocentric and egocentric betweenness [85]

The betweenness centrality  $C_B(p_i)$  based on the egocentric measures does not correspond perfectly to that based on sociocentric measures. However, it can be seen that the ranking of nodes based on the two types of betweenness are identical in this network. This means that two nodes may compare their own locally calculated betweenness value, and the node with the higher betweenness value can be determined. In effect, the betweenness value captures the extent to which a node connects nodes that are themselves not directly connected. For example,



in the network shown in figure 4.1,  $w_9$  has no connection with  $w_4$ . The node with the highest betweenness value connected to  $w_9$  is  $w_7$ , so if a message is forwarded to  $w_7$ , the message can then be forwarded to  $w_5$  which has a direct connection with  $w_4$ . In this way, betweenness centrality may be used to forward messages in a network. Marsden compared sociocentric and egocentric betweenness for 15 other sample networks and found that the two values correlate well in all scenarios.

The correlation is also supported by Everett and Borgatti who also identified a correlation of ego network betweenness and betweenness in a complete network [34]. They observe that it is common practice to normalise centrality scores in order to be able to compare measures with different networks consisting of varying population size. However, normalisation assumes the knowledge of a maximum centrality score which is not the case in ego networks. Consequently, they recommend not normalising betweenness scores. Additionally, the larger the number of contacts a node has, the higher the probability the contacts are between nodes outside of the ego network view. Normalisation would counter this effect as the size of an ego's ego network is a valuable piece of information.

## **Discussion**

Routing based on betweenness centrality provides a mechanism for information to flow from source to destination in a social network. However, routing based on centrality alone presents a number of drawbacks. Yan et al. analysed routing in complex networks and found that routing based on centrality alone causes central nodes to suffer severe traffic congestion as the number of accumulated packets increases with time, because the capacities of the nodes for delivering packets are limited [122]. Additionally, centrality does not take into account the time-varying nature of the links in the network and the availability of a link. In terms of information flow, a link that is available is one that is 'activated' for information flow.

## **4.2 Strong Ties for Information Flow**

The previous section discussion of information flow based on centrality measures does not take into account the strength of the links between nodes. In terms of graph theory, where the links in the network are time-varying, a link to a central node may not be highly available. Brown and

Reingen explored information flow in word-of-mouth networks and observe that it is unlikely that each contact representing potential sources of information have an equal probability of being activated for the flow of information [18]. They hypothesize that tie strength is a good measure of whether a tie will be activated, since strong ties are typically more readily available and result in more frequent interactions through which the transfer of information may arise. In a network where a person's contacts consisted of both strong and weak tie contacts, Brown and Reingen found that strong ties were more likely to be activated for information flow when compared to weak ties.

Tie strength is a quantifiable property that characterises the link between two nodes. The notion of tie strength was first introduced by Granovetter in 1973. Granovetter suggested that the strength of a relationship is dependent on four components: the frequency of contact, the length or history of the relationship, contact duration, and the number of transactions. Granovetter defined tie strength as: 'the amount of time, the emotional intensity, the intimacy (mutual confiding), and the reciprocal services which characterise a tie' [49]. Marsden and Campbell extended upon these measures and also proposed a measure based on the depth of a relationship referred to as the 'multiple social context' indicator. Lin et al. proposed using the recency of a contact to measure tie strength [77]. The tie strength indicators are defined as follows:

**Frequency Indicator** - Granovetter observes that 'the more frequently persons interact with one another, the stronger their sentiments of friendship for one another are apt to be' [49]. This metric was also explored in [49, 86, 11, 12, 78]

**Intimacy/Closeness Indicator** - This metric corresponds to Granovetter's definition of the time invested into a social contact as a measure for a social tie [12, 49, 86]. A contact with which a great deal of time has been spent can be deemed an important contact.

**Long Period of Time (Longevity) Indicator** - This metric corresponds to Granovetter's definition of the time commitment into a social contact as a measure for a social tie [12, 49, 86]. A contact with which a person has interacted over a longer period of time may be more important than a newly formed contact.

**Reciprocity Indicator** - Reciprocity is based on the notion that a valuable contact is one that is reciprocated and seen by both members of the relationship to exist. Granovetter discusses the social example with the absence of a substantial relationship, for example a 'nodding'

relationship between people living on the same street [12, 49]. He points out that this sort of relationship may be useful to distinguish from the absence of any relationship.

**Recency Indicator** - Important contacts should have interacted with a user recently [77]. This relates to Granovetter's amount of time component and investing in the relationship, where a strong relationship needs investment of time to maintain the intimacy.

**Multiple Social Context Indicator** - Marsden and Campbell discuss using the breadth of topics discussed by friends as a measure to represent the intimacy of a contact [12, 86].

**Mutual Confiding (Trust) Indicator** - The mutual confiding indicator can be used as a measure of trust in a contact [49, 86].

## Discussion

Routing based on tie strength in network terms is routing based on the most available links. A combination of the tie strength indicators can be used for information flow to determine which contact has the strongest social relationship to a destination. In this manner, messages can be forwarded through links possessing the strongest relationship, as a link representing a strong relationship more likely will be activated for information flow than a weak link with no relationship with the destination. These social measures lend themselves a local view of a network graph as they are based solely on observed link events and require no global knowledge of the network.

However, Granovetter argued the utility of using weak ties for information flow in social networks [49]. He emphasised that weak ties lead to information dissemination *between* groups. He introduced the concept of 'bridges', observing that

information can reach a larger number of people, and traverse a greater social distance when passed through weak ties rather than strong ties ... those who are weakly tied are more likely to move in circles different from our own and will thus have access to information different from that which we receive [49].

Consequently, it is important to identify contacts that may act as potential bridges. Betweenness centrality is a mechanism for identifying such bridges. Granovetter differentiates between the usefulness of weak and strong ties, 'weak ties provide people with access to information

and resources beyond those available in their own social circle; but strong ties have greater motivation to be of assistance and are typically more easily available.’ As a result, routing based on a combination of strong ties and identified bridges is a promising trade-off between the two solutions.

### 4.3 Tie Predictors

Marsden and Campbell distinguished between indicators and predictors [86]. Tie strength evaluates already existing connections whereas predictors use information from the past to predict likely future connections. Granovetter argues that strong tie networks exhibit a tendency towards transitivity, meaning that there is a heightened probability of two people being acquainted, if they have one or more other acquaintances in common [49]. In literature this phenomenon is called ‘clustering’. Watts and Strogatz showed that real-world networks exhibit strong clustering or network transitivity [118]. A network is said to show ‘clustering’ if the probability of two nodes being connected by a link is higher when the nodes in question have a common neighbour.

Newman demonstrated this by analysing the time evolution of scientific collaborations and observing that the use of examining neighbours, in this case co-authors of authors, could help predict future collaborations [94]. From this analysis, Newman determined that the probability of two individuals collaborating increases as the number  $m$  of their previous mutual co-authors increases. A pair of scientists who have five mutual previous collaborators, for instance, are about twice as likely to collaborate as a pair with only two, and about 200 times as likely as a pair with none. Additionally Newman, determined that the probability of collaboration increases with the number of times one has collaborated before, which shows that past collaborations are a good indicator of future ones.

Liben-Nowell and Kleinberg explored this theory by the following common neighbour metric in order to predict future collaborations on an author database [76]. The probability of a future collaboration  $P(x, y)$  between authors  $x$  and  $y$ , where  $N(x)$  and  $N(y)$  are the set of neighbours of author  $x$  and  $y$  respectively, is calculated by:

$$P(x, y) = |N(x) \cap N(y)| \tag{4.4}$$

Their results strongly supported this argument and showed that links were predicted by a factor of up to 47 improvement compared to that of random prediction. The *common neighbour* measure in equation 4.4 measures purely the similarity between two entities. Liben-Nowell also explored using Jaccard's coefficient which attempts to take into account not just similarity but also dissimilarity. The Jaccard coefficient is defined as the size of the intersection divided by the size of the union of the sample sets:

$$P(x, y) = \frac{|N(x) \cap N(y)|}{|N(x) \cup N(y)|} \quad (4.5)$$

Adamic and Adar performed an analysis to predict relationships between individuals by analysing user home pages on the world wide web (WWW) [5]. The authors computed features of the pages and also took into account the ingoing and outgoing links of the page, and defined the similarity between two pages by counting the number of common features, assigning greater importance to rare features than frequently seen features. In the case of neighbours, Liben-Nowell utilised this metric which refines a simple count of neighbours by weighting rarer neighbours more heavily than common neighbours. The probability, where  $N(z)$  is the number of neighbours held by  $z$ , is then given by:

$$P(x, y) = \sum_{z \in N(x) \cap N(y)} \frac{1}{\log|N(z)|} \quad (4.6)$$

All three metrics performed well compared with random prediction. Liben-Nowell explored a number of different ranking techniques based on information retrieval research, but generally found that common neighbour, the Jaccard's coefficient and the Adamic and Adar technique were sufficient, and performed equally as well, if not better than the other techniques.

## Discussion

Centrality and tie strength are based on the analysis of a static network graph whose link availability are time-varying. However, in the case of DDTMs the network graph is not static; it evolves over time. Tie predictors can be used in order to predict the evolution of the graph and evaluate the probability of future links occurring. Tie predictors may be used not only to reinforce already existing contacts but to anticipate contacts that may evolve over time.

## 4.4 Applications of SNA

Social network analysis techniques have been applied to a number of different areas. Several research initiatives have looked to harness these techniques for a range of computer-related problems. The following section highlights a number of them.

Whittaker et al. explored classifying user email contacts based on email logs. They asked participants to classify all extracted contacts into important and unimportant contacts [119]. They then evaluated a number of different social ties in order to identify how well they correlated to the participants' classification as important or unimportant contacts. Contacts were measured on frequency, reciprocity, longevity and recency. The authors found that reciprocity and longevity were the two highest predictors. Recency also was quite good with frequency being a weaker but still significant predictor.

Multiple social context has been used by Eagle and Pentland who recorded nine months of mobile phone users data recording other phones met, cellular towers seen and static bluetooth devices [33]. Based on past observed encounters, given the time of day and that the user is at work, Bayes rule was applied in order to predict the probability of encountering a specific individual. They compared relationships defined by users as friends to the daily proximity relationship and found that the networks share a similar structure indicating that proximity is a good indication of a relationship.

Zhang and Ackerman analysed strategies for searching for experts in a social network using an Enron email data set [123]. The authors generated a query from a random person using the email database. They evaluated searching based on a number of different algorithms including Breadth First Search (broadcast query to all neighbours), Random Walk Search (randomly select neighbour from list), Best Connected Search (send query to the neighbour with the highest out-degree), Weak Tie Search (send query to the neighbour who received the fewest emails from the current user) and Strong Tie Search (send the query to the user who received the most emails from the current user). This analysis can be seen as comparing the utility of strong ties, weak ties and centrality. The Best Connected Search achieve maximum performance with minimal path length. Weak Ties Search performed slightly better than Strong Ties search with slightly lower path lengths. They additionally found the Best Connected Search was the most resilient to path failure where they removed weak ties from the network and surprisingly also

when they removed central nodes.

The application of tie strength indicators in DDTM has been explored in limited terms. As discussed in section 3.3, the frequency indicator has been used in several routing solutions for disconnected networks [20, 26, 67, 80]. Intimacy/closeness in terms of the amount of time spent with a node has also been used in disconnected networks [66, 111]. Choudhury and Pentland explored the use of wearable sensors in order to detect underlying social patterns based on the frequency and duration of a contact by measuring conversations between users based on microphones and proximity sensors [28]. They found that the use of frequency lead to the discovery of a larger number of weaker ties, whereas duration identified less, but stronger ties. Consequently, the use of more indicators may prove more useful for evaluating ties strength.

## 4.5 Conclusion

We propose that information flow in a network graph whose links are time-varying can be achieved using a combination of centrality, strong ties and tie prediction. Social networks may consist of a number of highly disconnected cliques where neither the source node nor any of its contacts has any direct relationship with the destination. In this case, relying on strong ties would prove futile, and therefore weak ties to more connected nodes may be exploited.

Centrality has shown to be useful for path finding in static networks, however, the limitation of link capacities causes congestion. Additionally, centrality does not account for the time varying nature of link availability. Tie strength may be used to overcome this problem by identifying links that have a higher probability of availability. Tie strength evaluates existing links in a time varying network but does not account for the dynamic evolution of the network over time. Tie predictors may be used to aid in predicting future links that may arise. As such, we propose that the combination centrality, tie strengths and tie predictors are highly useful in routing based on local information when the underlying network exhibits a social structure.

# Chapter 5

## Parasitic Routing for DDTMs

This chapter presents the Parasitic Routing protocol, which is a protocol for delay-tolerant routing in disconnected MANETs. The goal of Parasitic Routing is to provide efficient message delivery in a highly dynamic disconnected delay-tolerant MANET. The routing protocol supports autonomous forwarding decisions by individual nodes to achieve message delivery in such challenging environments. The inspiration for the approach stems from social networks and the observation that many mobile networks exhibit small world properties. Chapter 4 gave an overview of social network analysis techniques and how these apply to information flow in a dynamic network graph whose links are time-varying. In this chapter these techniques are exploited in the protocol design. The result is a routing protocol that support message delivery without global knowledge and with no assumptions in regards to the control, or knowledge of future node movements.

Section 5.1 discusses the requirements for Parasitic Routing. Section 5.2 shows the application of the of social network analysis techniques discussed in chapter 4 as forwarding metrics for DDTMs. Section 5.3 presents the Parasitic Routing protocol design, beginning with a high level overview followed by detailed descriptions of the Parasitic Routing components. Finally, section 5.4 explains how the requirements are satisfied by the Parasitic Routing protocol.

### 5.1 Requirements

Chapter 2 presented the challenges associated with routing in DDTMs. These challenges can be used to derive design requirement for the Parasitic Routing protocol.



#	Requirement	Challenges			
		Mobility	Portability	Wireless Communication	Second Tier
R1	Support for autonomous forwarding decisions	•			
R2	Support for uncontrolled and unknown mobility	•			
R3	Support for localised autonomous topology discovery	•		•	
R4	Limit resource consumption		•	•	
R5	Support for unreliable wireless communication			•	
R6	Support for delay-tolerant message delivery	•			•
R7	Support for different quality of service requirements				•

**Table 5.1:** Overview of the Requirements

**R1: Support for autonomous forwarding decisions.** Each node must act as a router and make forwarding decisions in the absence of any centralised administration. Nodes must autonomously determine whether to forward a message to an encountered node, or continue buffering it. Due to the mobile environment resulting in a disconnected network graph as discussed in section 2.2.3, an end-to-end route from source to destination may never have a full end-to-end path to each other at any given point in time. Consequently, the decision must be based on locally available information and estimate which of the encountered nodes will be the better carrier for a specific message.

**R2: Support for uncontrolled and unknown mobility.** The consequences of node mobility constitute one of the most important and challenging attributes of MANETs. In the context of DDTMs, nodes may be attached to a number of different physically moving objects such as cars, people or animals. Consequently, and as discussed in sections 2.2.1 and 2.2.2, no assumptions can be made with regards to control over node movements or knowledge of future movements.

- R3: Support for localised autonomous topology discovery.** Due to the dynamic nature of the mobile environment as discussed in sections 2.2.4 and 2.2.5, limited topology information is available and its utility is volatile. As a consequence, an attempt is not made to analyse current connections to derive the topology of the network but instead use social network analysis in order to identify topological ‘bridges’ and long-term relationships between nodes in the network, which are not volatile and whose status will change slowly.
- R4: Limit resource consumption.** Resource consumption must be limited wherever possible. Memory concerns discussed in section 2.3.2 have the consequence that memory resources must be conserved by restricting the maximum number of messages a given node may buffer. When the maximum buffer size has been reached messages must be dropped. Battery power must be conserved whenever possible, as power concerns, discussed in section 2.3.1, remain one of the biggest problems for mobile nodes. Additionally, limited bandwidth resulting from wireless communication as discussed in section 2.1.3 means that transmissions must be used conservatively. The result of limited transmissions also aids in reducing power consumption as wireless communication is costly in terms of battery power.
- R5: Support for unreliable wireless communication environments.** Communication in a wireless environment means that one hop message delivery is not guaranteed. This may be due to lossy links or the fact that a node has moved out of range as discussed in section 2.1.4. Consequently, hop-by-hop acknowledgments must be used to determine if an encountered node has fully received forwarded messages. Additionally, the presence of asymmetric links, as discussed in section 2.1.5, means that communication may be possible in one direction but not the other. Despite their obvious inferior utility to symmetric encounters, such asymmetric encounters may still prove useful for learning about the social structure of the network.
- R6: Support for delay-tolerant message delivery.** In disconnected networks, a direct path from source to destination is assumed to be unavailable due to the disconnected network graph, as discussed in section 2.2.3. Therefore nodes must buffer messages they receive from the application layer and forward them to encountered nodes. Poor forwarding de-

cisions result in low delivery reliability, as discussed in section 2.4.1. Consequently, forwarding decisions must be made to maximise the probability of delivery, this is achieved by forwarding data to nodes that provide a higher probability of encountering the destination node. This requirement is the most important differences between DDTMs and classic MANETs.

**R7: Support for different quality of service requirements.** Application messages may have different quality of service requirements, as discussed in section 2.4.3. For example, some data may be time-dependent and lose its value after a given timeout. Others may be less time-dependent but may require stronger guarantee of delivery. Therefore support must be in place to allow applications to define differing quality of service requirements which must be utilised in order to provide differing levels of service.

The requirements imposed by the DDTM environment must be satisfied by the Parasitic Routing protocol design in order to provide sufficient message delivery support in such an environment. These requirements are used to drive the protocol design and are discussed throughout the chapter.

## 5.2 Using Social Network Analysis for Routing in DDTMs

Chapter 4 presented social network analysis for information flow in a dynamic network graph whose links are time-varying. This section describes how these metrics may be applied to information flow in DDTMs. Each metric is translated from a traditional social network analysis metric into a metric that is meaningful in a DDTM context. Additionally, the requirements of those presented in section 5.1 met by each metric is discussed.

### 5.2.1 Measuring Centrality for DDTMs

Betweenness centrality is calculated using an ego network representation as discussed in section 4.1, where a contact is a node which the ego node has encountered. When two nodes meet, they exchange a list of contacts. A contact is defined as a node that has been directly encountered by the node. The received contact list is used to update each node's local ego network. Mathematically, node contacts can be represented by an adjacency matrix  $A$ , which

is an  $n \times n$  symmetric matrix, where  $n$  is the number of contacts a given node has encountered. The adjacency matrix has elements:

$$A_{i,j} = \begin{cases} 1 & \text{if there is a contact between } i \text{ and } j \\ 0 & \text{otherwise} \end{cases} \quad (5.1)$$

Contacts are considered to be bidirectional, so if a contact exists between  $i$  and  $j$  then there is also a contact between  $j$  and  $i$ . The betweenness centrality is calculated by computing the number of nodes that are indirectly connected through the ego node. The betweenness centrality of the ego node is the sum of the reciprocals of the entries of  $A'$  where  $A'$  is equal to  $A^2 [1 - A]_{i,j}$  [34] where  $i,j$  are the row,column matrix entries respectively. A node's betweenness utility is given by:

$$Bet = \sum \frac{1}{A'_{i,j}} \quad (5.2)$$

Since the matrix is symmetrical, only the non-zero entries above the diagonal need to be considered. When a new node is encountered, the new node sends a list of nodes it has encountered. The ego node makes a new entry in the  $n \times n$  matrix. As an ego network only considers the contacts between nodes that the ego has directly encountered, only the entries for contacts in common between the ego node and the newly encountered node are inserted into the matrix. The betweenness centrality aids in satisfying two of the seven requirements. Betweenness centrality locally and autonomously measures the utility of a node in connecting other nodes in the network topology, satisfying R3. This utility can then be used by a node to locally determine its utility for carrying messages, which aids in satisfying R1. Additionally, the ego network may be stored as a sparsely populated matrix of 0's and 1's which is efficient in terms of memory resources, which is desirable for meeting requirement R4.

### 5.2.2 Measuring Tie Strength for DDTMs

Measuring tie strength will be an aggregation of a selection of indicators based on those discussed in section 4.2. One issue of concern is to ensure that each metric is on a similar scale. In order to accomplish this, an evidence based strategy is used to evaluate whether each measure supports or contradicts the presence of a strong tie. The evidence is represented as a tuple  $(s, c)$  where  $s$  is the supporting evidence and  $c$  is the contradicting evidence. Given two pieces of

evidence  $(s, c)$  and  $(s', c')$  the tuples may be compared where  $(s, c) < (s', c')$  if  $(s/c) < (s'/c')$  [50]. Consequently, the trust in a piece of evidence is measured as a ratio of supporting and contradicting evidence. Therefore as long as the ratio increases and evidence is at least as numerous as tuple  $(s', c')$  then it can be determined that  $(s', c')$  expresses more evidence in favour of the proposition. Below elaborates on specific tie strength indicators, representing them as a ratio of supporting and contradicting evidence and bring them into the context of DDTMs.

**Frequency Indicator** - In the case of DDTMs, the frequency indicator may be based on the frequency with which a node is encountered. The supporting evidence of a strong tie strength is defined as the total number of times node  $n$  has encountered node  $m$ . The contradicting evidence is defined as the amount of encounters node  $n$  has observed where node  $m$  was not the encountered node. The frequency indicator  $FI_n(m)$ , where  $f(m)$  is the number of times node  $n$  encountered node  $m$  and  $F(N)$  is the total amount of encounters node  $n$  has observed, is given by:

$$FI_n(m) = \frac{f(m)}{F(N) - f(m)} \quad (5.3)$$

**Intimacy/Closeness Indicator** - In the case of DDTMs, the duration indicator can be based on the amount of time the node has spent connected to a given node. The supporting evidence of intimacy/closeness is defined as a measure of how much time node  $n$  has been connected to node  $m$ . The contradicting evidence is defined as the total amount of time node  $n$  has been connected to nodes in the network where node  $m$  was not the encountered node. If  $d(m)$  is the total amount of time node  $n$  has been connected to node  $m$  and  $D(N)$  is the total amount of time node  $n$  has been connected across all encountered nodes, then the intimacy/closeness indicator can be expressed as:

$$ICI_n(m) = \frac{d(m)}{D(N) - d(m)} \quad (5.4)$$

**Long period of time (Longevity) Indicator** - In the case of DDTMs, the longevity indicator can be based on the length of time a node has known a given node. The evidence supporting the longevity indicator is the total amount of time node  $n$  has known node  $m$ . The contradicting evidence is the amount of time node  $n$  has been in the network and not known node  $m$ . If  $l(m)$  is the amount of time node  $n$  has known node  $n$  and  $L(n)$  is the total amount of time node  $n$  has been a part of the network, the longevity indicator is given as:

$$LI_n(m) = \frac{l(m)}{L(n) - l(m)} \quad (5.5)$$

**Reciprocity Indicator** - In the context of DDTMs, the equivalent of a ‘nodding’ relationship may be the reception of a ‘hello’ message broadcast by a passing node. The passing node may not receive a reply and therefore remains unaware of the contact. This ‘nodding’ relationship is captured by the frequency indicator. In order to differentiate between this ‘nodding’ relationship and one that is reciprocated, the supporting evidence is defined as the number of times node  $n$  has observed a reciprocated encounter with node  $m$ . The contradicting evidence is the number of times node  $n$  encountered node  $m$  which did not result in a reciprocated encounter in which a full two-way ‘hello’ exchange occurred. The reciprocity indicator, where  $r(m)$  is the total number of reciprocated encounters and  $f(m)$  is the total number of times node  $n$  encountered node  $m$ , is given as:

$$RI_n(m) = \frac{r(m)}{f(m) - r(m)} \quad (5.6)$$

**Recency Indicator** - The recency indicator is based on how recently node  $n$  has encountered node  $m$ . According to social network analysis theory a strong tie must be maintained in order to stay strong which is captured by the recency indicator. The supporting evidence  $rec(m)$ , is how recently node  $n$  last encountered node  $m$ . This is defined as the length of time between node  $n$  encountered node  $m$  and the time node  $n$  has been on the network. The contradicting evidence is the amount of time node  $n$  has been on the network since it has last seen node  $m$ . The recency indicator, where  $L(n)$  is the total amount of time node  $n$  has been a part of the network, is given as:

$$RecI_n(m) = \frac{rec(m)}{L(n) - rec(m)} \quad (5.7)$$

**Multiple Social Context Indicator** - In the case of DDTMs, multiple social context could correspond to nodes that encounter each other in various locations. For example a contact experienced in the workplace that is also seen outside of the workplace suggests a stronger relationship than one in which contacts only occur in the office. The multiple social context indicator evaluates how many different locations node  $n$  has encountered node  $m$ . The multiple social contact indicator is defined where  $c(m)$  is the total number of locations node  $n$  has

encountered node  $m$  normalised by the total number of places node  $n$  has visited  $C(N)$  where node  $n$  was not encountered.

$$MSCI_n(m) = \frac{c(m)}{C(N) - c(m)} \quad (5.8)$$

**Mutual confiding (trust) Indicator** - In the case of DDTMs, trust could be based on the number of times a successful message exchange has occurred between two nodes. This measure differentiates between nodes that may form brief contacts which do not last long enough to provide a successful exchange and nodes that provide an opportunity for full message exchange. In this case, it is not a measure of the strength of a relationship but rather a measure of the usefulness of the relationship.

Tie strength is a combination of a selection of indicators and the above metrics are all combined in order to evaluate an overall single tie strength measure. A strong tie should, ideally, be high in all measures. Consequently, the tie strength of node  $n$  for an encountered node  $m$ , where  $t$  is the total number of social indicators and  $w_k$  is the weighted value of each perspective indicator, is given by:

$$Tie_n(m) = \sum_{k=1}^t w_k I_n(m) \quad (5.9)$$

### 5.2.3 Measuring Similarity for DDTMs

Node similarity is calculated using the  $n \times n$  matrix discussed in section 4.3. The number of common neighbours between the current node  $i$  and destination node  $j$  can be calculated as the sum of the total overlapping contacts as represented in the  $n \times n$  matrix. This only allows for the calculation of similarity for nodes that have been met directly, but during the exchange of the node's contact list, information can be obtained in regards to nodes that have yet to be encountered. As discussed in section 4.3, the number of common neighbours may be used for ranking known contacts but also for predicting future contacts. Hence, a list of indirect encounters is maintained in a separate  $n \times m$  matrix, where  $n$  is the number of nodes that have been met directly and  $m$  is the number of nodes that have not directly been encountered, but

may be indirectly accessible through a direct contact. The similarity calculation, where  $N_n$  and  $N_e$  are the set of contacts held by node  $n$  and  $e$  respectively, is given as follows:

$$Sim(n, e) = \left| N_n \cap N_e \right| \quad (5.10)$$

#### 5.2.4 Node Utility Calculation

The Parasitic utility captures the overall improvement a node represents when compared to an encountered node across all measures presented in sections 5.2.1, 5.2.2 and 5.2.3.

Selecting which node represents the best carrier for the message becomes a multiple attribute decision problem across all measure, where the aim is to select the node that provides the maximum utility for carrying the message. This is achieved using a pairwise comparison matrix on the normalised relative weights of the attributes.

Attribute	Node1	Node2	Sum	Node1 Normalised	Node2 Normalised
BetUtility	$bet1$	$bet2$	$bet1 + bet2$	$bet1/betSum$	$bet2/betSum$
SimUtility	$sim1$	$sim2$	$sim1 + sim2$	$sim1/simSum$	$sim2/simSum$
TieUtility	$tie1$	$tie2$	$tie1 + tie2$	$tie1/tieSum$	$tie2/tieSum$
				$ParasiticUtility1$	$ParasiticUtility2$

**Table 5.2:** Parasitic Utility Calculation

The tie strength utility  $TieUtil_n$ , the betweenness utility  $BetUtil_n$  and the the similarity utility  $SimUtil_n$  of node  $n$  for delivering a message to destination node  $d$  compared to node  $m$  are given by:

$$TieUtil_n(d) = \frac{Tie_n(d)}{Tie_n(d) + Tie_m(d)} \quad (5.11)$$

$$SimUtil_n(d) = \frac{Sim_n(d)}{Sim_n(d) + Sim_m(d)} \quad (5.12)$$

$$BetUtil_n = \frac{Bet_n}{Bet_n + Bet_m} \quad (5.13)$$

The  $ParasiticUtil_n(d)$  is given by combining the normalised relative weights of the attributes, where  $U = \{TieUtil, SimUtil, BetUtil\}$ :



$$ParasiticUtil_n(d) = \sum_{i=1}^{u \in U} \gamma_i u_n(d) \quad (5.14)$$

The parameter  $\gamma$  is the relative weight for a given utility in  $U$ . Consequently, these parameters allow for the adjustment of the relative importance of each attribute utility value.

### 5.2.5 Social Network Analysis and Requirements

The use of social network analysis techniques satisfies a number of the requirements for routing in DDTMs.

#	Requirement	Satisfied by SNA
R1	Support for autonomous forwarding decisions	•
R2	Support for uncontrolled and unknown mobility	•
R3	Support for localised autonomous topology discovery	•
R4	Limit resource consumption	
R5	Support for unreliable wireless communication environment	
R6	Support for delay-tolerant message delivery	•
R7	Support for different quality of service requirements	

**Table 5.3:** Overview of the Requirements

The utility of a node is calculated based on locally evaluated information compared with that of an encountered node. Consequently, each node can make local and autonomous forwarding decisions in order to forward a message from source to destination, which satisfies R1. The metrics outlined for the parasitic utility calculation are based on observed interactions between nodes and require no further assumptions about past or future node movements, satisfying R2. Additionally, the metrics locally capture the underlying dynamic network graph whose links are time-varying, which satisfies requirement R3. As has been shown in chapter 4, information flow based on these social analysis metrics is possible. The message forwarding problem in delay-tolerant networks can be modeled as an information flow problem in a dynamic network graph whose links are time-varying. Consequently, these techniques provide support for R6. The remainder of this chapter describes the Parasitic Routing protocol which provides support for the remaining requirements.

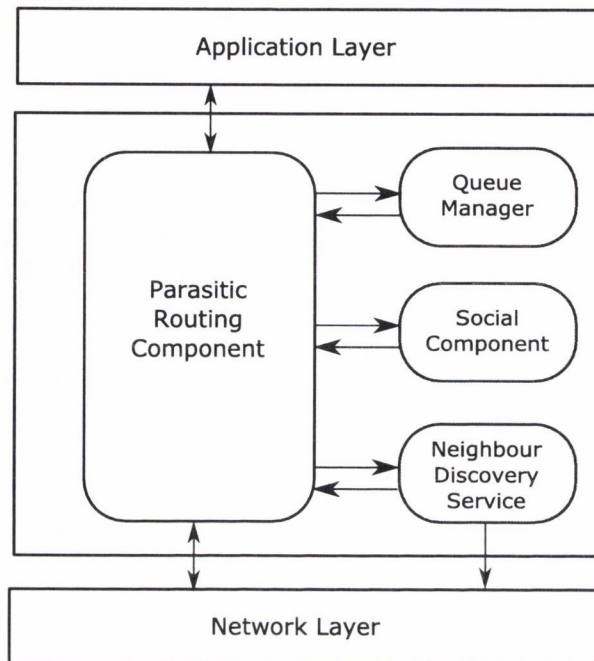
## 5.3 Parasitic Routing

The Parasitic Routing protocol is based on the concept of social network analysis. Instead of blindly forwarding messages, as in Epidemic Routing [115], the Parasitic Routing component forwards message to an encountered node that has a higher probability of being able to deliver the message to a given destination. This probability is based on a combination of social analysis metrics and available resources on the routing node, such as buffer space and battery power. When two nodes meet, they first exchange a list of contacts they have previously encountered. This information is used to update the local betweenness value as described in section 5.2.1. Each node in turn then sends a Summary Vector, which contains a list of destination nodes which they are currently carrying messages for, along with their betweenness centrality metric, available resource information and the social utility values of similarity and tie strength for each entry in the Summary Vector. Upon receipt of a Summary Vector, the node locally determines its own utility values for each destination and sends a Summary Vector response. Based on this information, the receiving node determines whether to forward messages to the encountered node or to continue buffering them.

### 5.3.1 Protocol Architectural Overview

Parasitic Routing is a reactive routing protocol that exploits mobile node encounters as they occur in order to route data from source to destination. The model assumes that each mobile node has a Parasitic Routing layer, which is situated between the application layer and the network layer. Figure 5.1 shows a logical overview of the Parasitic Routing layer. No bootstrapping is required to join the network, and once a joining node encounters an existing node, it is eligible to start transferring messages. Node mobility means that IP addresses may not be used as node identifiers because they may change as nodes move around the network. Consequently, it is assumed that each node has a unique DTN routable node identifier. Like all MANET protocols, the model requires each participating node to contribute resources in order aid the propagation of messages.

The Parasitic Routing layer is composed of four components: the Parasitic Routing component, the Queue Managers, the Social Component and a Neighbour Discovery Service. The following sections present the individual components in turn.



**Fig. 5.1:** Parasitic Routing Component Architecture

### 5.3.2 Neighbour Discovery Service

The Parasitic Routing component needs to be notified when a new node comes within transmission range, and when old nodes disappear. The Neighbour Discovery Service keeps track of one-hop asymmetric neighbours using periodic broadcasts of ‘hello’ messages. Receipt of another node’s ‘hello’ message indicates a one-hop neighbour relationship between those two nodes. The Neighbour Discovery Service maintains a list of its active neighbours. At each ‘hello’ interval, the heartbeat value of each neighbour is decremented. When a ‘hello’ message is received from an active neighbour the heartbeat value is reset. When a given number of ‘hello’ timeouts have occurred and a ‘hello’ message has not been received, then the neighbour is assumed to have been lost and the Parasitic Routing component is notified. Upon receipt of a ‘hello’ message, the sending node is determined to be a new encounter, if it is not already in the neighbour discovery list of currently available nodes. The Parasitic Routing component is notified of this event.

Neighbour discovery in MANETs is typically very expensive in terms of battery power. In order to avoid missing potential connections nodes must continuously scan for new neighbours.

Typical MANET neighbour discovery solutions use a fixed timeout range where a node scans for new neighbours. For example, the timeout will be anywhere within the range specified by a minimum and a maximum timeout interval. The exact timeout within this range is uniformly randomly selected in order to avoid synchronisation of node discovery resulting in collision of several ‘hello’ messages originating from different nodes.

However, this constant scanning for new neighbours results in quickly draining the battery. In order to address R4 and conserve resources, an adaptive neighbour discovery timeout interval is used. The onus of neighbour discovery is put on nodes currently buffering a number of messages ( $MQS$ ). If a node is currently not carrying any messages, the time out is set to the maximum. If the node is carrying a large number of messages the timeout range is selected proportional to the number of message the node is carrying in its buffer.

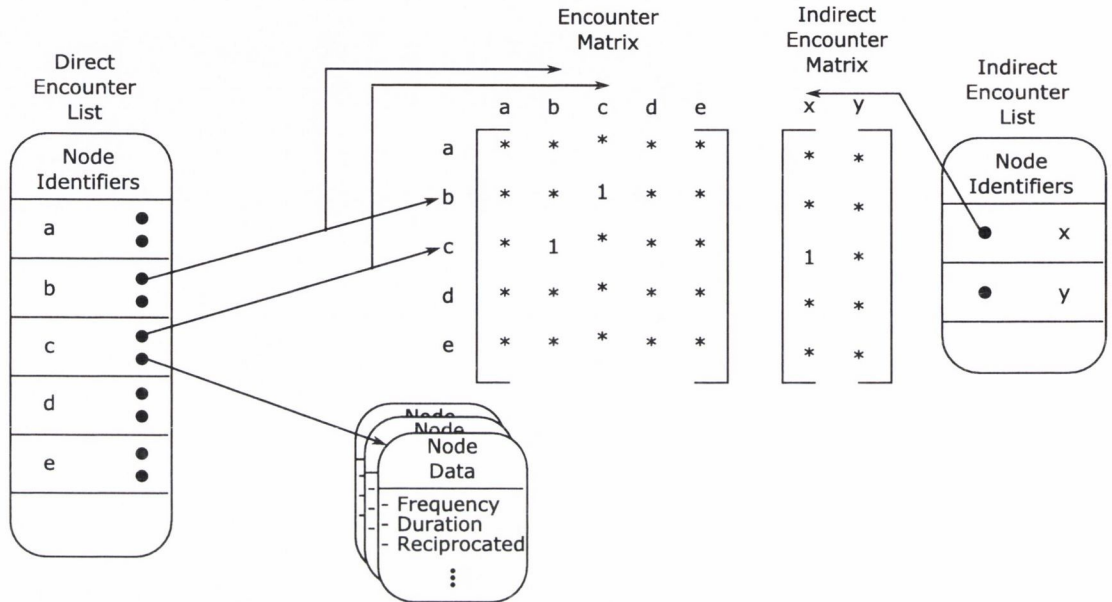
$$T = \begin{cases} Max & \text{if } MQS \text{ is } 0 \\ Min + \left(\frac{Max-Min}{MQS}\right) & \text{otherwise} \end{cases} \quad (5.15)$$

As a consequence, the larger the queue size, the shorter the discovery timeout, whereby the timeout is bounded to  $Max$ , if the message queue size is 0 and converges to  $Min$  if the  $MQS$  is large. The premise behind this design decision is to take advantage of every possible opportunity to transfer messages if a node is carrying a number of messages. This requirement is less prevalent for nodes that are not carrying any messages. However, it is still desirable to discover a node early enough in order to allow for an exchange of neighbour information.

### 5.3.3 Social Component

The Social Component is responsible for analysing observed encounters in order to evaluate three metrics: tie strength, social similarity and betweenness centrality, as described in section 5.2. The Social Component maintains a direct encounter list and an indirect encounter list, as shown in figure 5.2. The direct encounter list consists of node identifiers of nodes that have been directly encountered by the node. Each entry has a pointer to the row, column entry in the encounter matrix, which is used to calculate the node’s ego network betweenness value. Additionally, the entry maintains a pointer to the statistics related to the encountered node, which are required to calculate tie strength. The indirect encounter list has a pointer to the

column index of the indirect encounter matrix. The similarity of two nodes is calculated by obtaining the column entries of each corresponding node. In the example below, node *b* and node *x* have a similarity value of 1 as they share a common contact in node *c*.



**Fig. 5.2:** Social Component Data Structure

The direct encounter matrix and the indirect encounter matrix are sparsely populated matrices consisting of 0's and 1's. As a consequence, it is possible to store the matrix in compressed form [52, 104]. In this fashion, the structure of data storage of the Social Component addresses R4 where limited storage requirements is desirable for forwarding information.

### 5.3.4 Queue Manager

Parasitic Routing is a store-carry-forward routing mechanism. Consequently, messages must be buffered on a node until an appropriate carrier is found or the destination node is encountered. The Queue Manager is responsible for buffering the messages currently held by the node. In order to address R4, the Queue Manager must control the length of the queue and potentially which messages occupy it, by selecting which packets should be dropped when appropriate.

When the Parasitic Routing component receives a message, either from the application layer or from an encountered node, the message is inserted into the queue. Due to limited resources,

nodes may need to drop messages. A node must define a queuing policy to determine how its message queue should be handled. This section defines some basic queuing policies that could be deployed or combined to form more sophisticated schemes.

If the queue is full, then the Queue Manager determines which message may be dropped. There are a number of different options in regards to drop policy design.

**FIFO** - Handle the queue in a First-In-First-Out order. The message that was first entered into the queue is the first message to be dropped.

**TTL** - Drop shortest Time-To-Live first. Each message has a timeout value specifying when it is no longer meaningful to the application and should be deleted. This could be envisioned for applications where the utility of message delivery is time-dependent.

**DLU** - Drop Least Utility first. The Queue Manager drops the message that the currently carrying node has the least utility for delivering.

**DLP** - Drop Lowest Priority. Each message is assigned a priority level by the application layer in order to allow for differential quality of service support. When a buffer is full, lowest priority messages are dropped first.

**FQ** - Fair Queuing. In order to provide fair buffer management for all nodes on the network, drop a number of messages where either the source or the destination of the messages constitute the highest proportion of messages currently in the queue.

More than one queuing policy may be combined, where the first policy is used primarily and the second policy is used only if there are two messages with the same eviction priority as determined by the primary eviction policy. Dropping messages only when needed results in the queue being full a large proportion at a time. Consequently, the Queue Manager may need to employ active queue management techniques [44]. This may be achieved by examining the queue and dropping any packets containing a timeout value that has expired when the queue length reaches a certain threshold. The DLP drop strategy provides support for R7 to accommodate varying QoS requirements based on priority.

The Queue Manager is also responsible for returning a list of destination nodes for which messages are currently held in the queue to the Parasitic Component. Additionally, the Queue

Manager is responsible for returning a set of messages for a given destination node. The messages for a given destination must be returned to the Parasitic Routing component in order of priority determined by the queuing policy.

Buffer space is limited and represents a valuable resource in the mobile environment. Therefore, in order to address R3, the available buffer space becomes a node attribute *ResUtil*. This attribute is incorporated into the *ParasiticUtil* value given in section 5.2.4. Consequently, the  $ParasiticUtil_n(d)$  where  $U = \{TieUtil, SimUtil, BetUtil, ResUtil\}$  and  $\gamma$  is the relative weight for a given utility in  $U$ , is given by :

$$ParasiticUtil_n(d) = \sum_{i=1}^{u \in U} \gamma_i u_n(d) \quad (5.16)$$

As a result, if two nodes have similar Parasitic social metrics, the message is forwarded to the node with the most available buffer space. However, the resource utility must only be used as a tie break if two nodes are similar across all other utilities.

### 5.3.5 Parasitic Routing Component

The Parasitic Routing component is responsible for initialising the Queue Manager, the Social Component and the Neighbour Discovery Service as shown in a UML 2.0 activity diagram in figure 5.3.

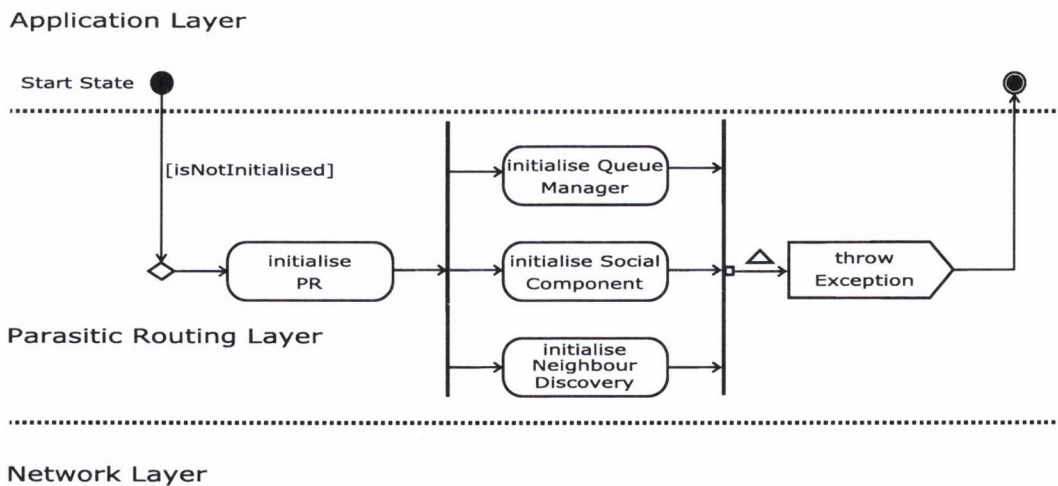
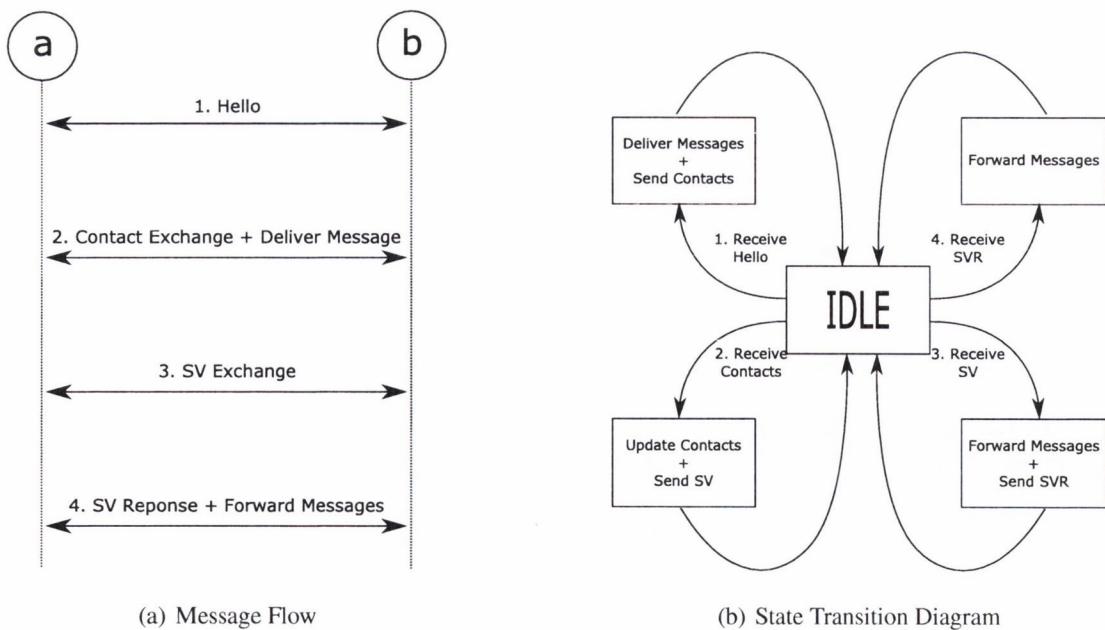


Fig. 5.3: Control Flow During Initialisation

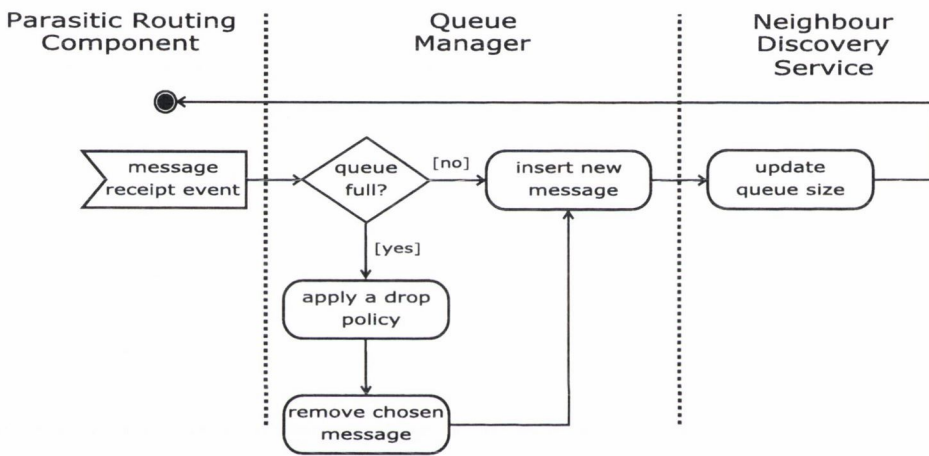
The component is event-driven in design and processes events from the application layer and the network layer. Figure 5.4 shows an overview of the protocol information flow. When two nodes *a* and *b* meet by exchanging hello messages, they first deliver messages destined for each and exchange contact information. The two nodes then exchange Summary Vectors containing a list of message destination nodes that are currently held in the Queue Manager along with the Parasitic metric information. If node *b* has a higher Parasitic utility value for a given destination node *c*, then node *a* immediately forwards all messages destined for node *c* to node *b*. If node *a* has a higher Parasitic utility for a given destination node *d*, then the destination node *d* is added to a Summary Vector response message. Upon node *b* receiving a Summary Vector response, the requested messages for node *d* are forwarded from node *b* to node *a*. The remainder of this section gives a more detailed account of how message receipt events are processed. It is important to note that a Parasitic node never misses an opportunity to deliver data or to forward data to a node with a higher utility. This is to address R5 and the fact that in DDTMs the duration of an encounter may be limited, and to take full advantage of even short encounters. Additionally, forwarded messages are not deleted from the message queue until these messages have been acknowledged by the next-hop node. Acknowledgments are piggybacked onto other messages in order to reduce overhead.



**Fig. 5.4:** Parasitic Routing Protocol Information Flow



**Message Receive Event:** Upon receipt of a message from the application layer or from an encountered node, the Parasitic Routing component inserts the message into the Queue Manager and updates the queue size value of the Neighbour Discovery Service as shown in figure 5.5.



**Fig. 5.5:** Response to Message Receive Event

**Neighbour Discovery Event:** Upon notification of a neighbour discovery event, the Parasitic Routing component first notifies the Social Component that a node encounter has taken place. It then checks the Queue Manager to see if there are any messages currently held for the encountered node. If this is the case, it then sends these messages through the network layer to the encountered node. The Parasitic Routing component then gets a list of contacts representing encountered nodes from the Social Component and sends this to the encountered node as shown in figure 5.6.

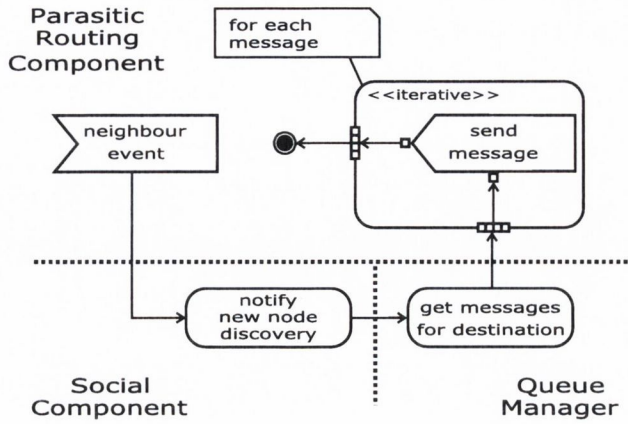


Fig. 5.6: Response to Neighbour Discovery Event

**Contact Message Exchange Event:** Upon receipt of a contact list, the Parasitic Routing component notifies the Social Component of the contact list which is used to update the betweenness representation matrix along with the indirect encounters matrix. Additionally, since a complete message exchange has occurred between the current node and the encountered node, the Parasitic Routing component notifies the Social Component that a reciprocated encounter has occurred. The Parasitic Routing component then compiles a Summary Vector message containing the betweenness value, the available queue resources, the similarity and the tie strength for each node in the Summary Vector as shown in figure 5.7 .

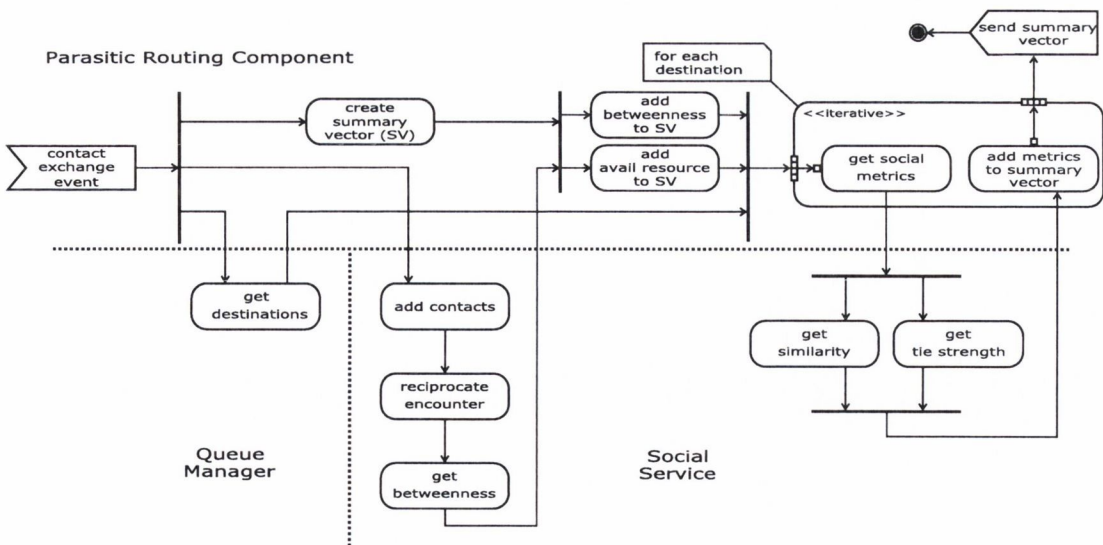


Fig. 5.7: Response to Contact Exchange Event

**Summary Vector Received Event:** Upon receipt of a Summary Vector, the Parasitic Routing component compares its own Parasitic utility with that of the encountered node. If the encountered node has a higher Parasitic utility value, then messages for the given destination node are forwarded to the encountered node. If the current node has a higher Parasitic utility value, then the node adds this information to a Summary Vector Response as shown in figure 5.8.

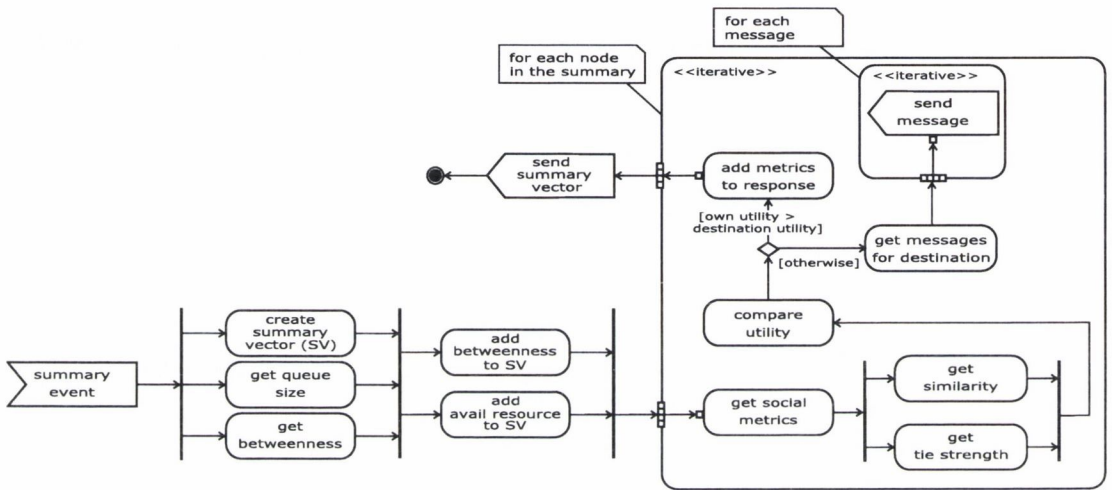


Fig. 5.8: Response to Summary Vector Event

### 5.3.5.1 Message Scheduling

In order to address R7, the Queue Manager is responsible for returning messages to the Parasitic Routing component for a destination node. These message may be returned in order of priority such as to ensure higher priority messages are delivered before those with a lower priority. Nodes forward messages upon receipt of a Summary Vector in the order in which the destinations appear in the Summary Vector or Summary Vector response. The Parasitic Routing component may order the destination nodes in a Summary Vector or Summary Vector response in the order of message importance based on a scheduling policy. There are a number of different options in regards to scheduling policy design.

**FHU** - Forward Highest Utility first. Messages are forwarded in descending order of the Parasitic utility the encountered node holds for the message destination node. Consequently, messages that the encountered node provides the highest utility for are forwarded first.

**FHP** - Forward Highest Priority. Each message is assigned a priority level by the application layer in order to allow for differential quality of service support. Messages are forwarding in descending order of priority.

**FTTL** - Forward shortest Time-To-Live first. Each message has a timeout value specifying when it is no longer meaningful. In order to provide these messages with the best chance of arriving at the destination node before this timeout, messages may be forwarded in ascending order of the TTL value.

**FFQ** - Fair Forwarding. In order to provide fair opportunity for messages for all nodes on the network, a proportion of messages for each node are forwarded first before the remaining messages of a source or destination are forwarded.

Messages are forwarded in the order dictated by the chosen forwarding policy. Consequently, if the encounter does not last long enough to exchange all messages the those that are deemed the most important are delivered first.

### 5.3.5.2 Replication

As discussed in chapter 3, replication is a standard way to improve the probability of successful message delivery. Consequently, in order to further satisfy R7, replication may be used for high priority messages where either the delivery time is of importance or the success of delivery of a high priority. A maximum replication number may be assigned to the Parasitic Routing component. When a message of high priority is received from the application layer, the maximum replication number is assigned to the message. Upon forwarding a message to an encountered node, the replication number is divided between the two nodes. This division is dependent on the Parasitic utility value of each node which is a value between 0 and 1 therefore the division of the replication number  $R_a$  and  $R_b$  for destination  $d$  between node  $a$  and node  $b$  is given by:

$$R_a = \lfloor R_{cur} \times ParasiticUtil_a(d) \rfloor \quad (5.17)$$

$$R_b = R_{cur} - R_a \quad (5.18)$$

Consequently, the node with the higher utility value receives a higher replication value. Replication improves the probability of message delivery, however, this comes at the cost of increased resource consumption. If replication is used, in order to avoid sending messages to nodes that are already carrying the message the Summary Vector must include a list of message identifiers it is currently carrying for each destination node.

## 5.4 Summary

This chapter has presented the Parasitic Routing protocol. The protocol requirements were defined in section 5.1. The mapping of social network analysis techniques that may be applied to DDTMs were described in section 5.2. The details of the Parasitic Routing protocol design were given in section 5.3.

Section 5.2.5 identified the requirements satisfied by utilising social network analysis to provide routing in DDTMs. Table 5.4 summarises the requirements satisfied by social network analysis techniques (the SNA column), protocol design (the PD column) and the overall resulting Parasitic Routing protocol (the PR column). The protocol design reduces resource consumption through the reduction of overhead associated with neighbour discovery, as presented in section 5.3.2. Due to the nature of wireless communication in mobile environments, communications may not last long enough in order to transfer all control information and exchange messages. The protocol design addresses this by first ensuring messages destined for the encountered node are delivered before any control data is exchanged, as described in section 5.3.5. Additionally, messages are forwarded to encountered nodes based on a message scheduling policy, as described in section 5.3.5.1. Consequently, if the encounter only lasts long enough for a small number of messages to be exchanged, those messages represent the most important messages as defined by the message scheduling policy.

#	Requirement	SNA	PD	PR
R1	Support for autonomous forwarding decisions	•		•
R2	Support for uncontrolled and unknown mobility	•		•
R3	Support for localised autonomous topology discovery	•		•
R4	Limit resource consumption		•	•
R5	Support for unreliable wireless communication environment		•	•
R6	Support for delay-tolerant message delivery	•		•
R7	Support for different quality of service requirements		•	•

**Table 5.4:** Requirements Satisfied by Social Network Analysis (SNA), Protocol Design (PD) and Parasitic Routing (PR)

# Chapter 6

## Implementation

This chapter describes a simulator-based implementation used for evaluating the Parasitic Routing protocol as presented in chapter 5. The motivation behind the choice of implementation in a simulator is first discussed. The main architectural components and their interfaces are described in detail. The chapter is structured as follows: Section 6.1 describes the trace-based simulator. Section 6.2 provides a high-level component overview of the Parasitic Routing layer. The next section defines the Parasitic Routing API, followed by an overview of the inter-component communication. The following sections 6.5 to 6.8 detail the application programming interfaces of the respective components.

### 6.1 Trace-Based Simulator

A trace-based simulator was implemented in order to provide a framework to evaluate Parasitic Routing under a range of conditions using real trace data of node encounters. The simulator was written in Java and replays encounters between nodes using trace files. Consequently, the simulation is driven by encounter events.

Trace files are fed into the simulator as inputs. Each encounter logged in the trace file is treated as a discrete encounter event. Events are ordered chronologically as shown in table 6.1. These encounters are treated differently depending on the level of detail available in the trace data. The example shown in table 6.1 captures the start time and end times of each encounter and the node identifiers of the nodes involved. If trace data is available for all nodes included in the experiment, then it is possible to differentiate between encounters where one

device recognised the presence of another device, but the encountered device remains unaware of the encounter. If this is the case, the encounter is inserted into an event queue and is only processed if both devices recorded the encounter. An example of a bi-directional encounter is shown in events 38 and 39 of table 6.1. As can be seen, start time and end time may differ due to clock synchronisation problems. To accommodate this, the encounter duration is taken to be the average of the two recorded encounter durations. If the two nodes differ in the recorded encounter duration by more than 15 minutes, then an error is assumed and the lower of the two values is used as the encounter duration.

Event Id	Node1	Node2	start time	end time
35	29	68	2004-08-01 14:04:51	2004-08-01 14:10:04
36	68	31	2004-08-01 14:05:20	2004-08-01 14:09:04
37	29	21	2004-08-01 14:21:14	2004-08-01 14:21:14
38	83	86	2004-08-01 15:50:14	2004-08-01 18:48:41
39	86	83	2004-08-01 15:58:32	2004-08-01 18:42:36

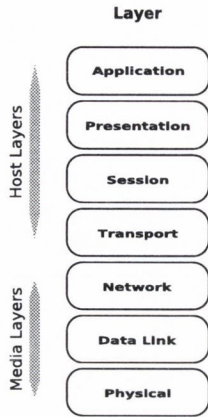
**Table 6.1:** Example Trace File Extract

The trace file encounters are used to simulate a node encounter where the simulator initiates communication between the two devices. It is possible for events to occur around the same time, however, the chronological ordering of encounters artificially imposes event ordering. As a consequence, when messages are exchanged between nodes, the time that this occurs is logged, meaning a message that arrived during an encounter may not be forwarded until after this encounter has ended. As a result, messages exchanged between nodes 29 and 68 during event 35 may not be forwarded to node 31 during event 36. The details of the individual trace data are discussed in more detail in the evaluation in chapter 7.

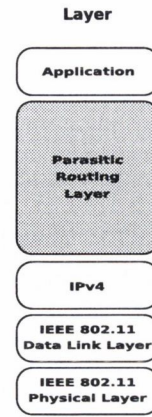
## 6.2 Architectural Overview

The Parasitic Routing layer sits between the mobile application and the underlying network layer. Although figure 6.2 shows an implementation above 802.11, Bluetooth or others can be





**Fig. 6.1:** OSI Model



**Fig. 6.2:** Parastic Routing in an IP Stack

substituted. Figure 6.3 depicts a high-level component diagram of the Parastic Routing Layer. The layer is divided into four individual components: Parastic Routing, Neighbour Discovery, Social, and Queue Manager components.

While the Parastic Routing component provides the infrastructure and the hooks into the application and network layers, the remaining three components are responsible for specialised functionality provided to the Parastic Routing component. The primary pattern used in the design of the Parastic Routing layer is the Observer pattern [43]. The motivation for applying the observer pattern is based on the observation that proper encapsulation mandates loose coupling of objects that manage the same data, while data only managed in objects internally should be highly cohesive [101]. Intuitively, coupling and cohesion are attributes that measure the degree of interdependence among objects and within objects, respectively [16]. In the context of the observer pattern, the subject is represented by the Parastic Routing component and the observer is represented by concrete Listener classes (implemented in the respective components) that declare interest in a particular Event triggered within the Parastic Routing component. When a particular Event is triggered, the Parastic Routing component notifies the registered listeners. Each registered listener gets the chance to handle the event and update the internal state of the respective component. This design has the benefit that the Parastic Routing component does not have to call interfaces of Neighbour Discovery, Social, and Queue Manager components. This reduces coupling and maintains a high degree of extensibility which is one of the desired features of this design. New functionality can be easily added by implementing

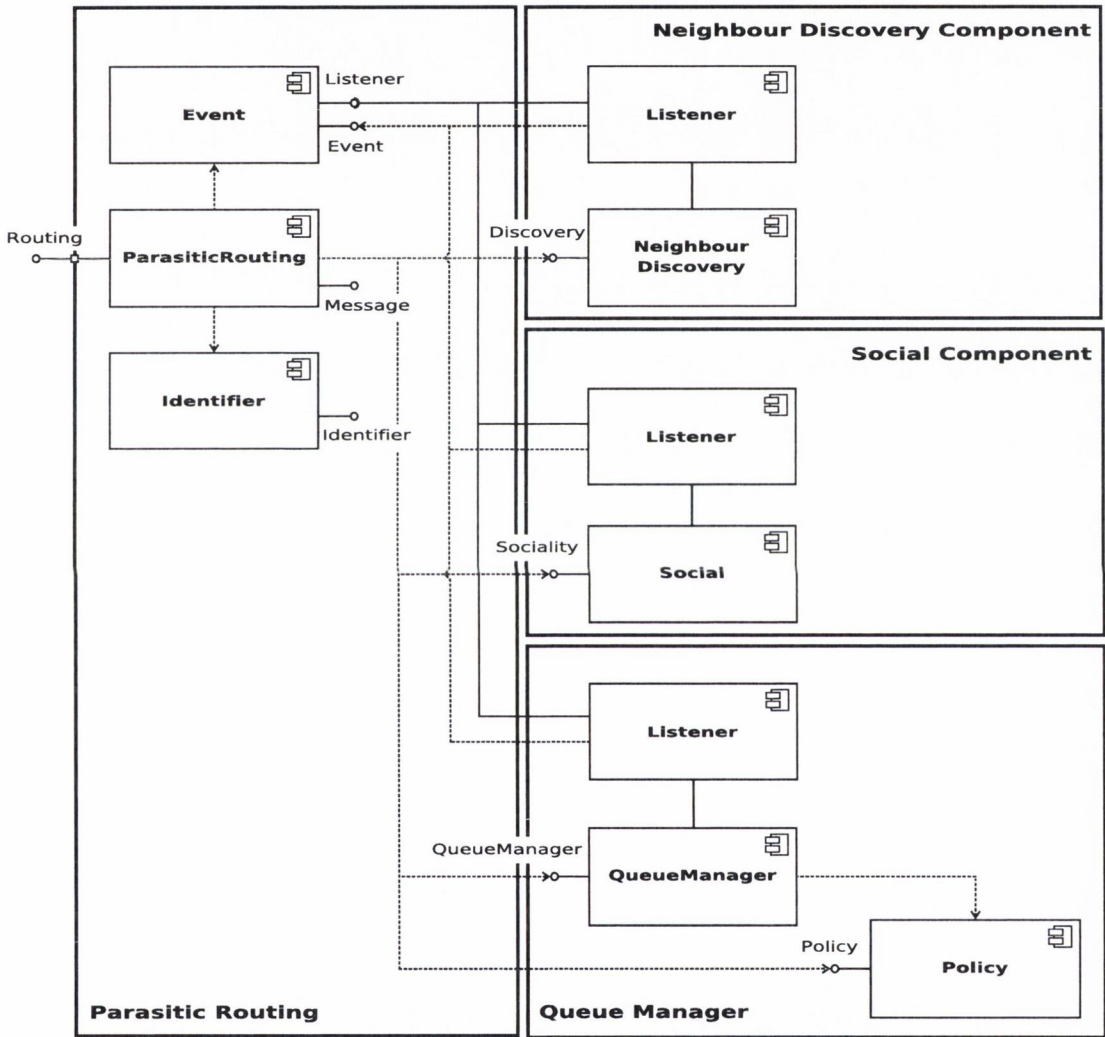


Fig. 6.3: Architectural Component Overview

specialised listeners and registering those with the Parasitic Routing component.

## 6.3 Parasitic Routing Layer API

The Parasitic Routing component exposes only two methods to the application layer:

- `sendMessage(Destination, Message, TTL, Priority)` is called when the application layer has a message to be delivered to a remote destination node. The message is then passed down to the Parasitic Routing layer.
- `receive(Message)` the callback is raised when a message has been delivered to the Parasitic Routing component where the current node is the destination. The Parasitic Routing layer then delivers the message to the application layer.

We assume the underlying network layer exposes two methods to the Parasitic Routing component:

- `sendIpMsg(Message, Destination)` is called by the Parasitic Routing component and passes an IP message to the networking layer. The network layer then assumes responsibility for delivering the message to the next hop destination node.
- `receive(Message, SrcIpAddress, lastHopMacAddress, dstIpAddress, Priority, TTL)` is a callback method which passes a received IP message from the networking layer to the Parasitic Routing component.

## 6.4 Inter-Component Communication

Inter-component communication is achieved using the observer pattern, where components register their interest in specific events. The events are defined as follows:

- `HelloEvent`: the node has received a hello message from an encountered node.
- `ContactEvent`: the node has received a contact exchange message from a neighbouring node.

- `SummaryEvent`: the node has received a Summary Vector, or a Summary Vector Response.
- `MessageEvent`: the node has received a new message either from the application layer or from a neighbouring node.
- `MessageAckEvent`: the node has received an acknowledgment for a message forwarded to an encountered node.

These events mirror the abstract events discussed in section 5.3.5. In this way, the relevant components are notified when events occur and act accordingly. The following sections describe the handling of these events by each component.

## 6.5 Neighbour Discovery Service

This section describes the Neighbour Discovery Service derived from the design discussed in 5.3.2. As stated earlier, a loosely coupled relationship with the Parasitic Routing component is achieved using the observer pattern.

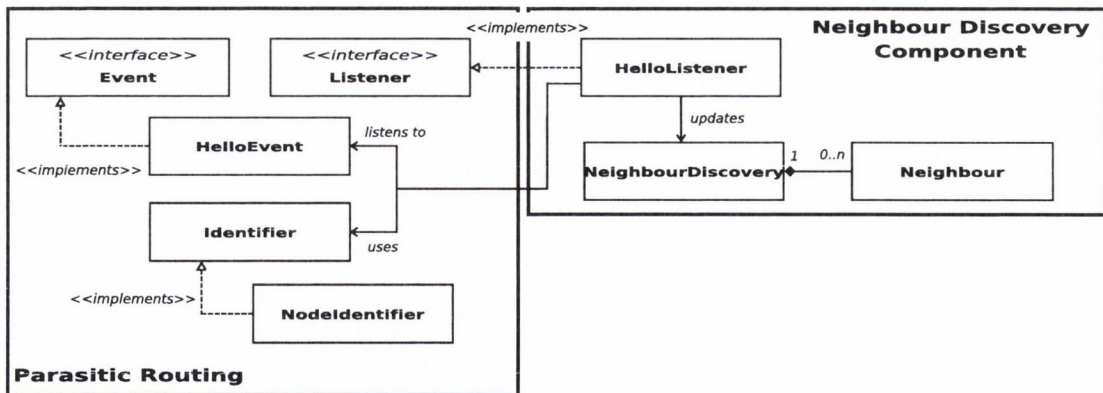


Fig. 6.4: Neighbour Discovery Service Class Diagram

Figure 6.4 shows the class diagram of the neighbour discovery service and its associations to classes contained in the Parasitic Routing component. The responsibilities of the neighbour discovery service is to maintain a list of neighbours within radio transmission range. Consequently, a `HelloListener` is registered with the Parasitic Routing component to be notified

when a `HelloEvent` is raised. In this case, the `HelloListener` adapts the number of `Neighbours` and updates their heartbeat values accordingly. A neighbour is identified using the `Identifier` class. The sub-class `NodeIdentifier` is a specialisation that represents a node with its IP address and MAC address.

Given these interactions, the neighbour discovery interface is defined as follows:

- `neighbourDiscovered(Identifier)` is a callback raised to notify the neighbour discovery service that a new node has been discovered. The neighbour is added to the list of current neighbours.
- `neighbourLost(Identifier)` is a callback raised to notify the neighbour discovery service that a neighbour has been lost, at this point the neighbour is removed from the current neighbours list.
- `setQueueSize(int)` is a method called by the Parasitic Routing Component. The Parasitic Routing component is responsible for notifying the neighbour discovery service of the size of the message queue in order to determine the neighbour discovery timeout interval.

## 6.6 Social Component

This section outlines the classes and application programming interface of the Social Component. Similar to the Neighbour Discovery Component described in section 6.5, the observer pattern is represented by the concrete listener implementations, `HelloListener`, `ContactListener`, and `ReciprocateListener` which listen to the `HelloEvent` and `ContactEvent`.

The social component, shown in figure 6.5, is responsible for associating social metrics with a node. The relevant information to identify a node is given by the `NodeIdentifier` class. In line with the design given for the social component in section 5.3.3, the application programming interface for the Social Component class exposes the following methods:

- `encounterNode(NodeIdentifier)` is called by the Parasitic Routing component upon the discovery of a new neighbour in order to update encounter metrics for this node.

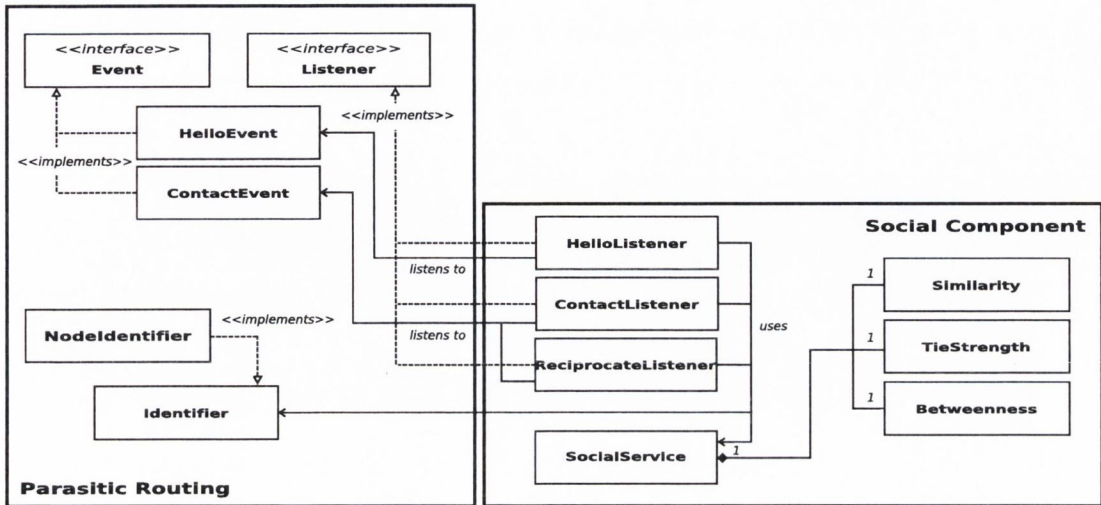


Fig. 6.5: Social Component Class Diagram

- `reciprocatedEncounter(NodeIdentifier)` is called by the `ReciprocateListener` when a reciprocated message exchange has occurred between the current node and the encountered node in order to update reciprocated metrics for this node. The message exchange originates from the `ContactEvent` triggered in the Parasitic Routing component. When this occurs the the
- `updateTimeConnected(NodeIdentifier, Timestamp)` is called by the Parasitic Routing component when notified by the Neighbour Discovery Service that a neighbour has been lost. The `Timestamp` is the amount of time that has passed since the Neighbour Discovery Service registered this node as a neighbour and when the Neighbour Discovery Service determines that the neighbour has been lost.
- `getTieStrength(NodeIdentifier)` is called by the Parasitic Routing component to determine the tie strength utility of the node for a given message.
- `getSimilarity(NodeIdentifier)` is called by the Parasitic Routing component to determine the similarity utility of the node for a given message.
- `getBetweenness()` is called by the Parasitic Routing component to determine the betweenness utility of the current node.

- `getContacts()` is called by the Parasitic Routing component in order to generate a contacts message to be sent to an encountered node.

## 6.7 Queue Manager

This section deals with the application programming interface and internal behaviour of the Queue Manager component based on the design described in 5.3.4. Section 5.3.4 discussed the motivation behind queue management and presented a few basic queuing policies, including FIFO (first-in first-out), TTL (time-to-live), DLU (drop least utility first), DLP (drop lowest priority first), and FQ (fair queuing). Only these basic queuing policies were considered, although the design of the Queue Manager component allows for possible extensions to the queuing strategies.

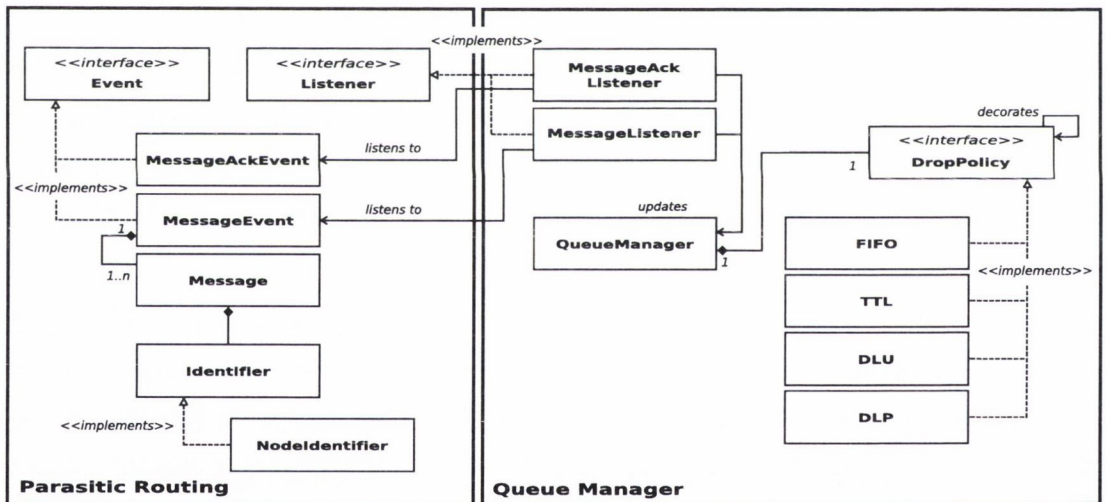


Fig. 6.6: Queue Manager Class Diagram

Figure 6.6 shows the class diagram of the Queue Manager component. Like the previously discussed components, the queue manager employs the observer pattern in order to register an interest in `MessageEvents`. The `MessageListener` implements the behaviour to be executed on the arrival of new messages. This listener class is registered by the `QueueManager` with the Parasitic Routing component. Similarly, the `MessageAckListener` listens for message acknowledgments or, more specifically, to the `MessageAckEvent` triggered by the Parasitic Routing component.

The queue manager interface is as follows:

- `insert(Destination, Message, TTL, Priority)` is called by the `MessageListener` when a message is received and the `MessageListener` is notified by the Parasitic Routing component. The message is then inserted into the queue.
- `delete(MessageID)` this method is called by the `MessageAckListener` upon the receipt of an acknowledgment. Next hop message delivery is not guaranteed, as a result messages are not deleted from the message queue until an acknowledgment has been received from the encountered node.
- `getDestinationList()` is called by the Parasitic Routing component to compile a summary vector message to be sent to an encountered node. The summary vector contains a list of destination nodes the current node is carrying messages for along with the Parasitic utility values.
- `getMessagesForDest(NodeIdentifier)` is called by the Parasitic Routing component either when a destination node is encountered that messages are carried for or when it is determined that an encountered node is a better carrier for messages for a given destination.
- `getQueueSize()` is called by the Parasitic Routing component in order to determine how many messages are currently stored in the message queue.

The Queue Manager component incorporates a Strategy pattern [43] in order to encapsulate the queuing policies (`DropPolicy`) into separate classes and allow those to be interchanged transparently. The strategy `DropPolicy` interface declares methods common to all drop policies. The family of concrete strategies (drop policies) considered in this section are `FIFO`, `TTL`, `DLU`, and `DLP`. The strategy is used in the so-called context object (here the `QueueManager`). The `QueueManager` maintains a reference to the `DropPolicy` object, which is configured with a concrete drop policy implementation. Furthermore, several drop policies can be combined through the Decorator pattern [43]. The decoration has the advantage of avoiding class explosion, if inheritance was used to add responsibilities to individual objects. Also, it offers flexibility to instantiate Parasitic Routing to match requirements specific to the application domain.



## 6.8 Parasitic Routing Component

This section provides the algorithmic details of the component based on the design described in 5.3.5. Figure 6.7 outlines the classes of the Parasitic Routing component which is the primary component that handles the interaction between the constituent components of the Neighbour Discovery Service, the Social Component and the Queue Manager. These interactions are discussed in the following event handling algorithms.

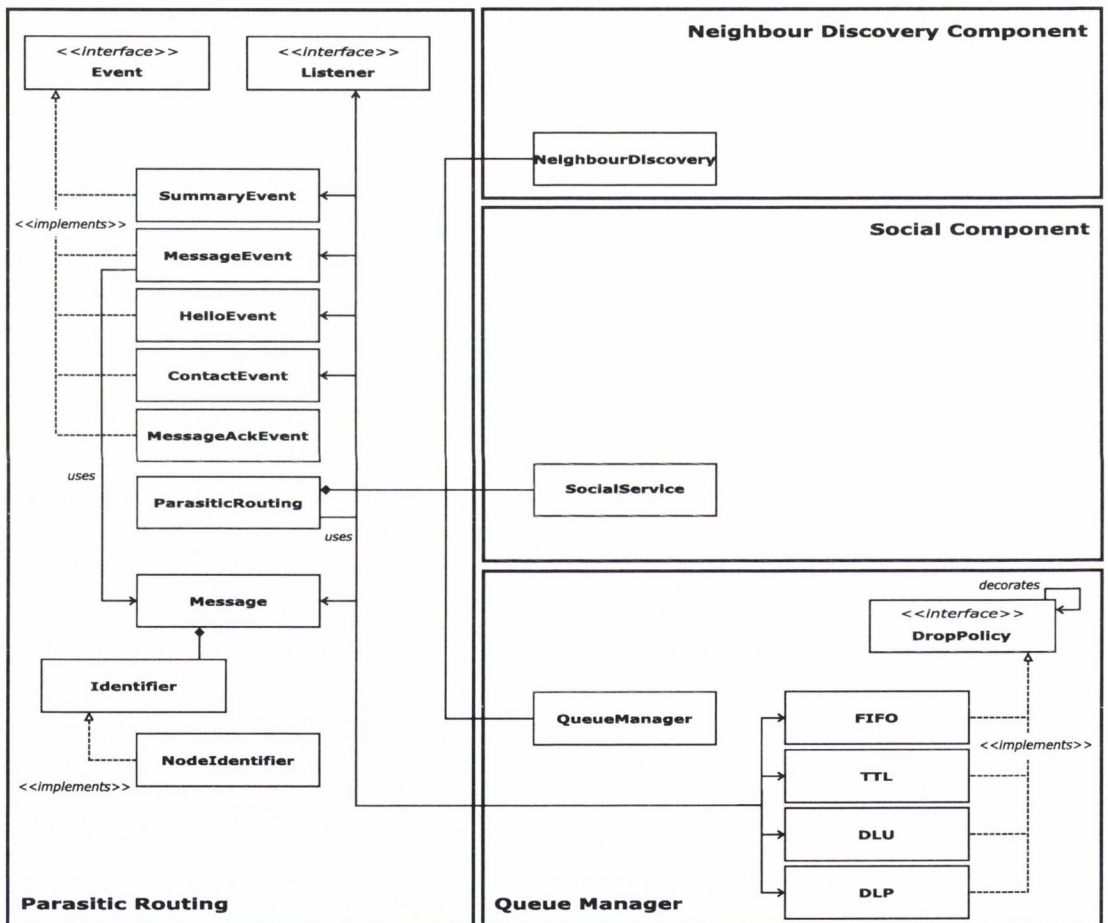


Fig. 6.7: Parasitic Routing Class Diagram

### 6.8.1 Event Handling Algorithms

The Parasitic Routing component is an event-based component where listeners have to be registered to declare interest in particular events. This allows the Parasitic Routing component to

be fairly independent of status updates in sub-components, i.e., the Neighbour Discovery Component, the Social Component, and the Queue Manager component. The events were described in section 5.3.5. Their handling by the Parasitic Routing component is described below.

#### **6.8.1.1 Message Receipt Event**

Upon receipt of a message from the application layer or from an encountered node, the Parasitic Routing component notifies the QueueManager and the neighbour discovery service in order to insert the new message and update the queue size respectively as shown in algorithm 1.

---

**Algorithm 1** MessageReceiptEvent

---

- 1: *notify(QueueManager, Message)*
  - 2: *notify(NeighbourDiscoveryService, QueueSize)*
- 

#### **6.8.1.2 Neighbour Discovery Event**

Algorithm 2 depicts the behaviour of the Parasitic Routing component when a neighbour discovery event occurred. Upon notification of a neighbour discovery event, the Parasitic Routing component must first notify the Social Component that a node encounter has taken place. It then checks the Queue Manager to see if there are any messages currently held for the encountered node. If this is the case, it then sends these messages through the network layer to the encountered node. The Parasitic Routing component then gets a list of contacts representing encountered nodes for the Social Component and then sends this to the encountered node.

---

**Algorithm 2** NeighbourDiscoveryEvent

---

```
1: notify(SocialComponent, NodeIdentifier)
2: msgsList ← QueueManager.getMessagesForDest(NodeIdentifier)
3: for all m ∈ msgsList do
4:   sendIpMsg(m)
5: end for
6: contact ← SocialComponent.getContacts()
7: create contactMessage cm
8: sentIpMsg(cm)
```

---

### 6.8.1.3 Receive Contact List Event

Upon receipt of a contact list, the Parasitic Routing component notifies the Social Component of the contact list, which subsequently is used to update the betweenness representation matrix along with the indirect encounters matrix. Additionally, since a complete message exchange has occurred between the current node and the encountered node, the Parasitic Routing component notifies the Social Component that a reciprocated encounter has occurred. It then compiles a summary vector message containing the betweenness value, the message queue size and then the similarity and tie strength for each node in the summary vector. The individual steps are formalised in the algorithm 3.

---

**Algorithm 3** ReceiveContactListEvent

---

```
1: # ContactMessage represents the list of contacts of encountered node
2: contactList ← ContactMessage.getContactList()
3: notify(SocialComponent, NodeIdentifier, contactList)
4: notify(SocialComponent, NodeIdentifier)
5: create summaryVectorMessage svm
6: svm.betweenness ← SocialComponent.getBetweenness()
7: svm.queueSize ← QueueManager.getQueueSize()
8: destList ← QueueManager.getDestinationList()
9: for all d ∈ destList do
10: create summaryVectorElement sve
11: sve.dest ← d
12: sve.sim ← SocialComponent.getSimilarity(d.dest)
13: sve.tie ← SocialComponent.getTieStrength(d.dest)
14: svm.add(sve)
15: end for
16: sendIpMessage(svm, destination)
```

---

#### 6.8.1.4 Receive Summary Vector Event

Upon receipt of a Summary Vector the Parasitic Routing component compares its own Parasitic utility with that of the encountered node. If the encountered node has a higher Parasitic utility value then messages for the given destination node are forwarded to the encountered node. If the current node has a higher Parasitic utility value then the node adds this information to a Summary Vector Response as outlined in algorithm 4.

---

**Algorithm 4** ReceiveSummaryVectorEvent

---

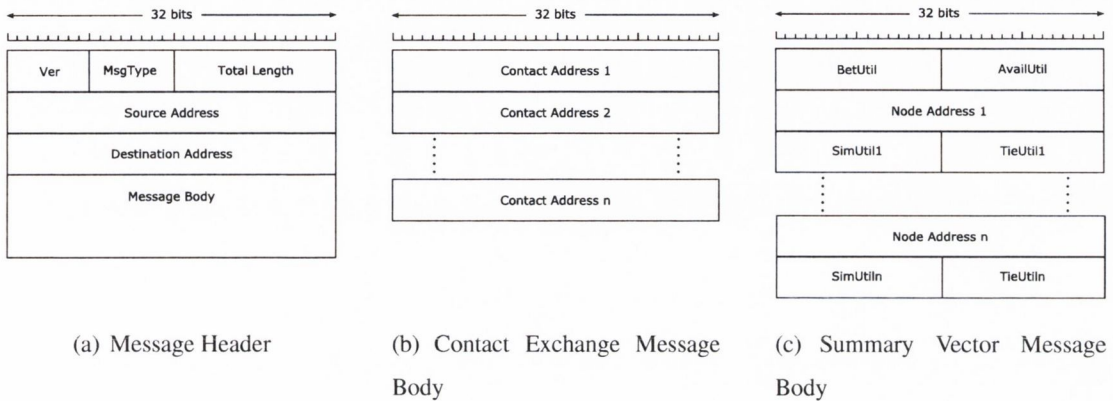
```
1: # SummaryVectorMessage represents the list of nodes the encountered node is carrying
   messages for, along with the utility values
2: encounteredBetweenness ← SummaryVectorMessage.getBetweenness()
3: encounteredQueueSize ← SummaryVectorMessage.getQueueSize()
4: create summaryVectorResponseMessage svrm
5: svrm.betweenness ← SocialComponent.getBetweenness()
6: svrm.queueSize ← QueueManager.getQueueSize()
7: elementList ← SummaryVectorMessage.getContactList()
8: for all  $e \in$  elementList do
9:   if compareUtility( $e.dest$ , encounteredBetweenness,  $e.similarity$ ,  $e.tieStrength$ , encounteredQueueSize) == 1 then
10:    create summaryVectorResponseElement sve
11:    sve.sim ← SocialComponent.getSimilarity( $e.dest$ )
12:    sve.tie ← SocialComponent.getTieStrength( $e.dest$ )
13:    svrm.add(sve)
14:   else
15:    msgsList ← QueueManager.getMessagesForDest( $e.dest$ )
16:    for all  $m \in$  msgsList do
17:      sendIpMsg( $m$ )
18:    end for
19:   end for
20: sendIpMessage(svrm, destination)
```

---

## 6.9 Protocol Message Format

Figure 6.8 shows the protocol message formats. The message header is shown in figure 6.8 a) which includes the protocol version number, the message type, the length of the message and source, destination node addresses followed by the message body. The MsgType field

identifies the format of the message body. The message body may be a contact exchange, a summary vector request or response or a message transfer as shown in figure 6.8 b) and c) respectively. In the case of a ‘hello’ message, the message body is empty.



**Fig. 6.8:** Protocol Message Format

- **Ver** This identifies the protocol version number
- **MsgType** The MsgType field identifies the type of message that is in the body of the message.
- **Total Length** This value gives the total length of the message
- **Source Address** DTN routable identifier of sending node
- **Destination Address** DTN routable identifier of message destination node
- **BetUtil** the Betweenness utility of the sending node
- **AvailUtil** the available resources utility of the sending node
- **Node Address 1** the DTN routable identifier of a node the sending node is carrying a message for
- **SimUtil1** The similarity utility of the sending node for the node identified by Node Address 1
- **TieUtil1** The tie strength utility of the sending node for the node identified by Node Address 1

## **6.10 Summary**

This chapter has presented a simulator implementation of the Parasitic Routing protocol as defined by the protocol design in chapter 5. The trace-based simulator can be used for evaluation purposes to determine the utility of the metrics described in chapter 4 and is used to assess the performance of the protocol in chapter 7. The implementation of Parasitic Routing uses the Observer pattern in the software design, and is implemented in Java. The implementation details of the Parasitic Routing components in the form of the Neighbour Discovery Service, the Queue Manager, the Social Component have been described, along with the key algorithms. Additionally, the protocol specific message formats have been given.

# Chapter 7

## Evaluation

This chapter evaluates Parasitic Routing for message delivery in disconnected delay-tolerant mobile ad hoc networks (DDTMs). First, the evaluation plan is presented and the evaluation criteria are defined. The related protocols that will be used for comparison to Parasitic Routing are then described, along with the evaluation parameters. The evaluation is based on five case studies using trace data consisting of real world device interactions. Each case study is discussed in turn and the performance of the respective protocols assessed. Finally, the chapter concludes.

### 7.1 Evaluation Strategy

The objective of the experiment is to establish how well Parasitic Routing has addressed the requirements as presented in section 5.1. The key aim is to provide delay-tolerant message delivery in an environment that is disconnected and where node movements are uncontrolled and unknown. Each node must operate autonomously, making forwarding decisions with no global knowledge as to the underlying network. Additionally, the protocol must achieve message delivery under realistic scenarios where resources are limited.

#### Protocols for Comparison

In order to evaluate the performance of Parasitic Routing versus other protocols, two related protocols were implemented which are also designed for message delivery in disconnected



networks. Epidemic Routing as presented in section 3.2 represents the only protocol that guarantees message delivery in disconnected networks, even though it assumes infinite bandwidth and node storage requiring no global knowledge [115]. As a result, Epidemic Routing represents a good baseline in terms of message delivery because it always finds the shortest path from source to destination, given that one exists. The second protocol is a version of the PRoPHET protocol as reviewed in section 3.3, which was implemented according to the design presented in [80, 81]. PRoPHET encapsulates a number of solutions based on past encounters and requires no global knowledge and makes no assumptions with regards to node movements. Additionally, PRoPHET is the only routing protocol that has been submitted as an internet draft to the DTNRG [2]. PRoPHET constitutes the primary currently accepted solution to routing in disconnected delay-tolerant networks where node movements are unknown and uncontrolled.

## Protocol Parameters

Epidemic Routing requires no parameter tuning. When a node encounter occurs, nodes exchange a summary vector containing a list of message identifiers and the destination node of each message currently stored in the node's buffer. Nodes then exchange messages not contained in the encountered node's summary vector.

PRoPHET has four parameters. The default PRoPHET parameters as recommended in [80] were used. However, one parameter that should be noted is the time elapsed unit  $\kappa$  used to age the contact probabilities. The appropriate time unit used differs depending on the application and the expected delays in the network. Therefore, this parameter was tuned for each simulation and the value that delivered the optimal results was selected. For completeness, the results from simulations used to tune the parameters are presented in Appendix 8.3.

The Parasitic Routing utility is the combination of three attribute utilities: *TieUtil*, *SimUtil*, *BetUtil*, as discussed in section 5.2.4. When node storage resources are limited the *ResUtil* attribute is used as a tie break between nodes with equal *ParasiticUtil* values. Each attribute is assigned equal weight. *TieUtil* is based on four indicators as presented in 5.2.2: frequency, intimacy/closeness, reciprocated and recency. Each tie strength indicator is assigned equal weight.

Two versions of Parasitic Routing and PRoPHET are evaluated: a single-copy version and a

multi-copy version. In the single-copy strategy, when two nodes meet, messages are exchanged between nodes such that messages are forwarded to the node with the highest utility. The node that has forwarded the message must then delete the message from its message queue. In the multi-copy strategy, replication is used where messages are assigned a replication value  $R$ . When two nodes meet, if the replication value  $R > 1$  then a message copy is made. The value of  $R$  is divided between the two nodes, as discussed in section 5.3.5.2. If  $R = 1$  then the forwarding becomes a single-copy strategy. A replication value of  $R = 4$  is used in the experiments presented here.

## Evaluation Criteria

The performance of each protocol is evaluated under the following criteria, which are common metrics for comparing routing protocols:

- **Total Number of Messages Delivered:** The ultimate goal of the protocol is to achieve message delivery in a disconnected mobile environment where no global knowledge is available and where node movements are unknown and uncontrolled. This metric is measured as the total number of messages that arrive at the messages' destination. It is expected that with the assumption of infinite bandwidth and unlimited node storage, Epidemic Routing will achieve very high delivery performance, as the redundancy of replicated messages means that a route will always be found if one exists. Single-copy Parasitic Routing and PROPHET do not have the benefit of redundancy and consequently, message delivery is expected to be lower. Multi-copy Parasitic Routing and PROPHET include a degree of redundancy, although on a much lower scale than that of Epidemic Routing.
- **Average End-to-End Delay of Delivered Messages:** End-to-End delay is an important concern in DDTMs. Long end-to-end delays mean messages must occupy valuable buffer space for longer, and consequently a low end-to-end delay is desirable. Additionally, low end-to-end delays are generally desirable in communication environments. The end-to-end delay is measured by logging the time (maintained globally by the simulator) at which the message was generated by the source node and the time at which it was delivered at the destination node. The average end-to-end delay is the total delay across

all messages normalised by the total number of messages delivered. Epidemic Routing is expected to perform well, as it is guaranteed to find the shortest route from source to destination. As with message delivery, because Parasitic Routing and PROPHET include limited redundancy the average end-to-end delay is expected to be greater.

- **Average Number of Hops per Delivered Message:** The average number of hops taken by a message in order to reach the destination is important for resource consumption in the network. Wireless communication is costly in terms of battery power, and as a result minimising the number of hops also minimises the battery power expended in forwarding messages. This metric is measured by incrementing the hop count of a message every time it is forwarded. When the message is delivered, the number of hops taken by that message is added to a global overall number of hops. The average number of hops per message is the total value normalised by the total number of messages delivered. Epidemic Routing will find the optimal path in terms of time, even if the shortest route requires more hops. However, it is still expected that the average number of hops for Epidemic Routing will be close to optimal. We place no limit on the number of hops a message may take, and consequently Parasitic Routing and PROPHET may require a greater number of hops to reach the destination.
- **Total Number of Forwards:** The number of forwards represents the overhead in the network in terms of how many times a message forward occurs in the network. This metric is measured by incrementing a counter every time a node exchanges a payload message with an encountered node. This metric excludes any control data messages and messages that are delivered to the destination node. Epidemic Routing is expected to perform poorly in this metric, as each node forwards each message to an encountered node currently not carrying the message. Single-copy PROPHET and Parasitic Routing are expected to lead to a much lower number of forwards as only a single copy of each message exists in the network. Multi-copy PROPHET and Parasitic Routing are expected to incur an increased number of forwards when compared to the single-copy strategy. However, due the relatively low replication value, they are expected to show a much lower number of forwards than Epidemic Routing.
- **Control Data Overhead:** Protocol control overhead can be costly in highly mobile

MANETs. If the protocol overhead is great then it can significantly affect the utility of the protocol. This metric is measured by keeping an overall total of the amount of data in bytes exchanged between nodes on the network. In the case of Epidemic Routing, the only control data exchanged is a summary vector including a list of message identifiers along with the destination node. However, as each node eventually will carry a copy of each message contained in the network, the size of the summary vector will grow with time. For each encounter, PRoPHET first exchanges a list of all nodes that have been encountered directly and indirectly along with a delivery predictability for each destination node. Parasitic Routing first exchanges a list of all nodes that have been directly encountered, followed by a summary vector containing a list of destination nodes for which it is currently carrying messages along with the Parasitic Routing Utility metrics. The PRoPHET probabilities are a single float, whereas the Parasitic Routing utility includes two values for each destination node, tie strength and similarity. As a result, it is expected that Parasitic Routing will show an additional overhead when compared to PRoPHET. In the case of multi-copy Parasitic Routing and PRoPHET, in order to prevent messages from being routed to nodes already carrying the message, nodes must exchange a summary vector containing a list of message identifiers currently held on the node, similar to that of Epidemic Routing. Consequently, the control data required for multi-copy Parasitic Routing and PRoPHET is expected to be greater when compared to the single-copy strategies.

## **Simulation Setup**

Parasitic Routing is based on the non-random movement and encounter patterns of nodes in real world scenarios. The value of the validation is therefore highly dependent on realistic movement models for simulation. In the Random Waypoint mobility model [17], which is widely used in MANET evaluation scenarios, a node randomly selects a destination within the network and randomly selects a speed within a defined range. The node then travels to the selected location at the given speed. Once the node reaches the destination, it remains there for a given pause time (also randomly determined) and then repeats the process by again selecting a random destination and speed. The movement of the nodes is in a bounded space, and the

density of nodes at the center of the simulation area tends to grow indefinitely.

In response to the growing need for more realistic mobility models, a number of new models have been proposed based on emulating social structure [15, 92]. Both of these mobility models are relatively new and therefore their validity has not been verified. As a result, Parasitic Routing is evaluated using real world trace data which is known to capture realistic node encounter patterns. All of the traces include data regarding encounters between electronic devices and the time at which these encounters took place. Movement information is not available, and for this reason popular simulators such as ns2 are unsuitable. Consequently, the trace data is used to create an event-based simulation as described in section 6.1. The traces include a number of different network scenarios, ranging from node population from 30-228, and include long and short term interactions [1]. Bluetooth devices are used in data sets consisting of encounters between devices carried by people. Additionally, the final data set includes 30 buses communicating using 802.11.

For each data set, four experiments were conducted, comparing the performance of each protocol.

- **Baseline Comparison:** The first test is designed to highlight the utility of each protocol without real world constraints of limited resources and limited bandwidth. It is assumed that each encounter that lasts longer than zero seconds is sufficient to transmit all control data, exchange appropriate messages and deliver all messages destined for the encountered node. Additionally, there is no limit to the number of messages a node may carry. The motivation behind this experiment is to evaluate purely the routing strategy over a graph whose links are time-varying. Additionally, because the network graph is disconnected, the existence of a path from source to destination is not guaranteed. If one such path exists, Epidemic Routing will find it and consequently, its results represent a baseline for the maximum possible number of messages that could be delivered.
- **Limited Bandwidth - Transfer Speed:** This experimental setup is similar to the previous one, except in this case a fixed message size of 1K is assumed and delivery performance is evaluated as a function of transfer speed with the parameter ranging from 33Kbps to 1000Kbps. These ranges were selected to reflect the varying transmission capabilities of Bluetooth and 802.11 [57]. In this case, Parasitic Routing and PROPHET

Message Size	Transfer Speed	Capacity
1K	33Kbps	1MB
10K	50Kbps	5MB
50K	100Kbps	10MB
100K	500Kbps	30MB
–	1000Kbps	50MB

**Table 7.1:** Experimental Parameters

employ the priority scheduling policy FHU, as discussed in section 5.3.5.1, where messages are forwarded in descending order of the node’s utility for delivering to each destination.

- **Limited Bandwidth - Message Size:** In this experiment, the total number of bytes that may be transferred during an encounter is restricted based on the encounter duration and assuming a fixed transfer speed. At each step in each protocol flow (message delivery, exchange control data, and message exchange), the size of each communication in bytes is calculated. If there are sufficient bytes remaining to complete the step, then the protocol continues. If there are no bytes remaining, then the simulator terminates the encounter. This experiment is repeated for various message sizes ranging from 1K to 100K, and assumes a fixed transfer speed of 100Kbps.
- **Limited Capacity:** In order to evaluate how each protocol performs under the realistic assumption that device memory capacity may be limited, the above experiment is repeated assuming a message size of 1K and a fixed transfer speed of 100Kbps with the additional constraint that the buffer space is limited. If a node must drop a message, Epidemic employs a FIFO drop strategy. In the case of Parasitic Routing and PROPHET, the DLU drop strategy is used in which a node drops messages that it has the lowest probability to deliver as discussed in section 5.3.4.

Table 7.1 summarises the experiment parameters. The following sections presents the details of each data set and the result of the simulation experiments and their analysis.

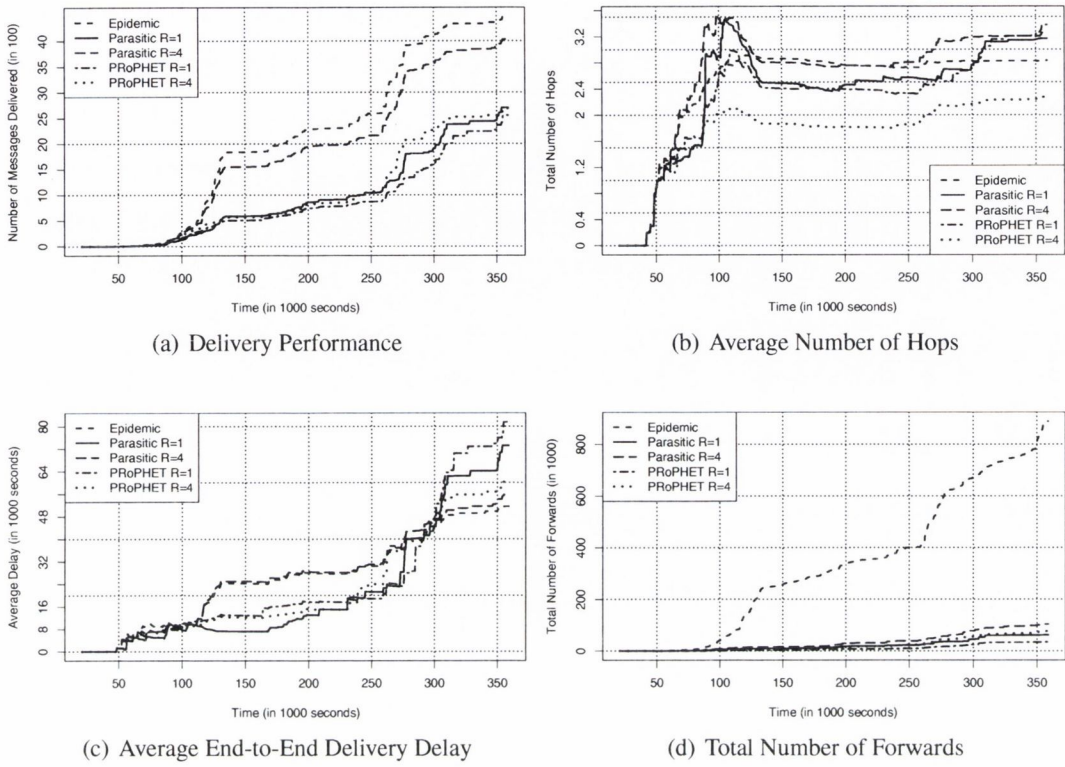
	Intel	Cambridge	Infocom
Participants	8	12	41
Total Devices	128	223	264
Total Encounters	2,264	6,736	28,250
Average Encounter Duration	885 secs	572 secs	231 secs
Duration	3 days	5 days	3 days

**Table 7.2:** Hagggle Data Set

## 7.2 Case Study: Hagggle Data Set

Cambridge University and Intel Research conducted three experiments of Bluetooth encounters using 8 participants carrying iMote devices [24, 58]. Encounters between Bluetooth devices were logged. Log entries include: the encounter start time, the deviceID, the deviceID of the encountered node and the duration of the encounter. Logs are only available for the participating nodes, but the data set also includes encounters between participants carrying the iMote devices and external Bluetooth contacts. These include anyone who had an active Bluetooth device in the vicinity of the iMote carriers.

The experiments lasted between three to five days and only included a small number of participants. As a result, all encountered devices are used as inputs for the simulator described in section 6.1, and each device represents a possible sender, receiver and carrier of data. In order to provide PROPHET and Parasitic Routing an opportunity to gather information about the nodes within the network, 15% of the simulation duration is used as a warm up phase. After the warm up phase, each node on the network generates messages to uniformly randomly selected destination nodes at a rate of 20 messages per hour from the start of the message sending phase when the sending node first reappears on the network until the last time the sending node appears on the network during the sending phase. The sending phase lasts for 70% of the simulation duration, allowing an additional 15% at the end in order to allow messages that have been generated time to be delivered. Table 7.2 summarises the experiment details for each data set.



**Fig. 7.1:** Baseline Protocol Performance for Intel Data

## 7.2.1 Intel Data Set

The first experiment included eight researchers and interns working at Intel Research in Cambridge and lasted three days [24]. A total of 128 nodes, including the eight participants, were encountered for the duration of the experiment.

### 7.2.1.1 Baseline

The following experiment determines the baseline performance of each protocol under the conditions of unlimited bandwidth and memory resources. The simulations were run for all three protocols 10 times with different random number seeds. In this case, it is assumed that each Bluetooth encounter provides an opportunity to exchange all routing data and all messages. Additionally, there is no limitation on the number of messages a single node may store. Figure 7.1 graphs the results for this simulation, and table 7.3 shows the statistics with the average and standard deviation across all runs of the simulation. As shown in figure 7.1 a) it is clear that



Epidemic Routing outperforms both Parasitic Routing and PRoPHET under scenarios with unlimited resource and infinite bandwidth. Single-copy Parasitic Routing delivers fewer messages than Epidemic but shows improvement when compared to Single-copy PRoPHET. Multi-copy Parasitic Routing shows significant improvement with the addition of replication resulting in an improvement of nearly 50% than the single-copy strategy. Multi-copy PRoPHET shows much less improvement with the addition of replication, achieving results comparable to single-copy Parasitic Routing. All protocols show a number of plateaus where no messages are delivered before the 150 mark and at the 200 mark. These plateaus represent night time when the devices are not within range of other devices.

Figure 7.1 b) shows the average number of hops taken by the delivered messages. All protocols result in a small number of hops of around 3. Single-copy and multi-copy PRoPHET show the largest and smallest number of average hops respectively. Single-copy and multi-copy Parasitic Routing show very similar average hop values, meaning the path lengths found by single-copy Parasitic Routing were close to that achieved by multi-copy Parasitic. Epidemic achieves short paths from the source to destination of just below an average of 3. Parasitic Routing shows a distinct increase in the average number of hops after the 300 mark, which coincides with a large number of messages being delivered. This indicates that these messages required a larger number of hops in order to be delivered, thus raising the average.

Figure 7.1 c) shows the average message end-to-end delay. As expected, Epidemic Routing shows the lowest average end-to-end delay. Single-copy PRoPHET shows the highest overall delay. Single-copy Parasitic Routing shows similar delays. Both multi-copy Parasitic Routing and PRoPHET show reduced delays with multi-copy Parasitic Routing resulting in low delays similar to that of Epidemic Routing. All protocols show a significant increase in delay around the 250 mark, which coincides with a steep increase in the number of messages delivered, meaning these delivered messages required a longer amount of time to be delivered, thus increasing the average end-to-end delay.

The true disadvantage of Epidemic Routing becomes clear when examining the total number of forwards as shown in figure 7.1 d). Epidemic Routing continues to forward messages throughout the network as time goes on and as a result incurs a great deal of overhead. Single-copy Parasitic and PRoPHET only have a single copy of each message on the network, resulting in a significantly lower number of forwards. Multi-copy Parasitic Routing and PRoPHET as

	Message Delivery		Average Number of Hops		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	4470.20	64.44		2.83	0.01
Prophet R=4	2688.60	54.04		2.27	0.02
Parasitic R=4	4030.20	67.39		3.25	0.02
Prophet	2552.89	54.89		3.38	0.06
Parasitic	2704.40	64.88		3.17	0.04

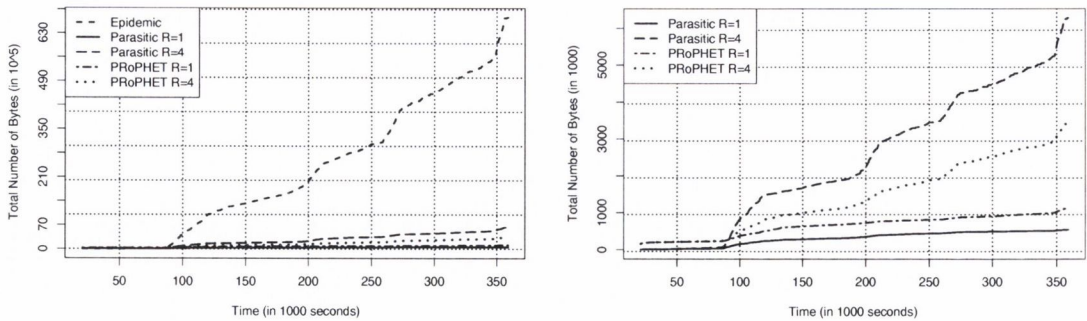
  

	Average End-to-End Delay		Total Number of Forwards		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	51514.35	1063.61		889047.10	4358.01
Prophet R=4	60200.47	1372.33		76585.00	1317.87
Parasitic R=4	55660.54	1110.22		103818.90	267.41
Prophet	81526.66	2857.29		34452.89	499.21
Parasitic	73152.45	1804.72		61800.60	231.92

**Table 7.3:** Baseline Protocol Performance Statistics for Intel Data

expected show an increase in the number of forwards with the use of replication. However, when compared to that of Epidemic Routing, multi-copy Parasitic Routing results in approximately 98% reduction in number of forwards. Single-copy and multi-copy PROPHET results in the least number of forwards, but results should be viewed in the context that PROPHET delivered less messages overall.

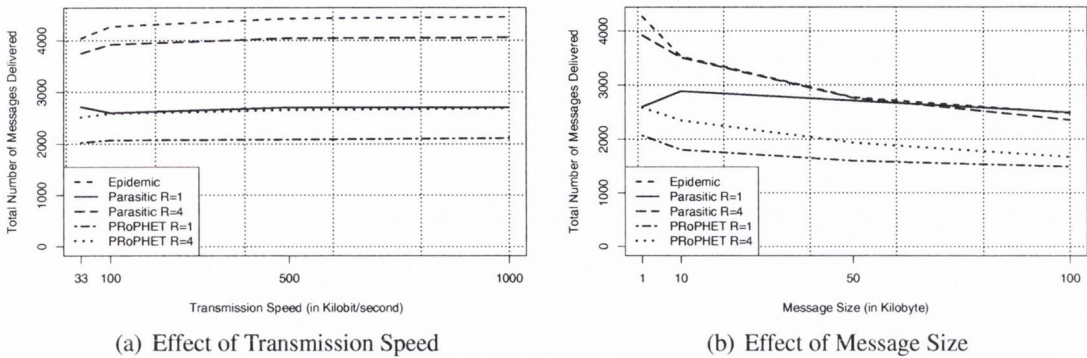
Figure 7.2 a) shows the protocol control data overhead in bytes. The overhead resulting from Epidemic Routing is so large it makes it difficult to differentiate between the overhead generated by Parasitic Routing and PROPHET. This is because the summary vector must contain a list of all messages the node is currently carrying, and as the number of messages on the network increases, so does the control data. Single-copy Parasitic Routing and PROPHET generate significantly lower overhead due to the fact that nodes exchange information about message destination rather than explicit message identifiers. As a result, there is an upper limit on the amount of bytes required. This upper bound depends on the node population. Figure 7.2 b) shows the control data overhead for Parasitic Routing and PROPHET. Single-copy Parasitic



**Fig. 7.2:** Baseline Control Data Overhead for Intel Data

Routing surprisingly shows a lower amount of control data required. The reason for this is that the upper bounds in terms of neighbour exchange is lower for Parasitic Routing than PRoPHET. PRoPHET exchanges data in regards to all nodes that have been encountered directly and indirectly, meaning that if all nodes have information about all other nodes on the network, then the control data required is directly proportional to the node population. In the case of Parasitic Routing however, the information exchange is limited to the number of nodes directly encountered and the number of destination nodes it is currently carrying messages for. In the case of the Intel data, the maximum number of nodes a single node encountered was 84, and the average was approximately 33. Consequently, on average nodes exchange encounter information for only 33 nodes, whereas PRoPHET may result in information exchange including up to 128 nodes. Multi-copy Parasitic Routing and PRoPHET show an increase in control data due to the necessary exchange of message identifiers, thus reducing the benefit of exchanging only routing data, as is the case for the single-copy strategy. However, the overhead is still significantly lower than that of Epidemic Routing due to the fact that nodes do not carry a copy of every message in the network.

From this experiment, we can determine that single-copy and multi-copy Parasitic Routing show higher message delivery when compared to that of PRoPHET. Multi-copy Parasitic Routing achieves delivery performance similar to that of Epidemic Routing with short path lengths and low end-to-end delay. The use of replication comes at the cost of an increased number of forwards and an increase in control data. However, when compared to that of Epidemic Routing, these metrics are still relatively low in terms of overhead.

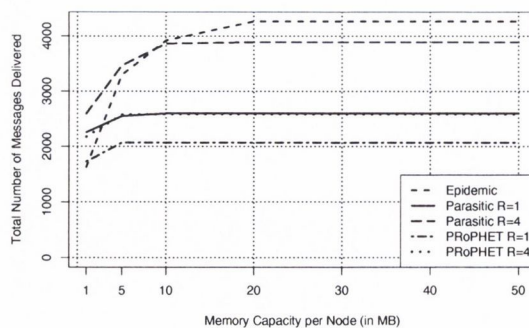


**Fig. 7.3:** Effect of Limited Bandwidth for Intel Data

### 7.2.1.2 Limited Bandwidth

The baseline tests show the protocol performance under idealised circumstances where bandwidth is unlimited and memory capacity is infinite. In order to evaluate the performance of the protocols in a more realistic scenario, the transfer speed is varied starting with 33Kbps up to 1000Kbps bandwidth capabilities. Each message is 1K in size. As can be seen in figure 7.3 a) the performance of Epidemic and PRoPHET shows a small decrease in performance transmission speeds of 33Kbps. Interestingly, single-copy Parasitic Routing shows an increase in performance at 33Kbps. This is an unforeseen side effect of the message scheduling policy where messages are forwarded to an encountered node in decreasing order of the Parasitic utility of the encountered node. Consequently, the messages that do not get a chance to be forwarded due to reduced bandwidth must remain on the current node for a longer period of time than with high transfer speeds. As a result, the message waits until a node with a higher utility is encountered. Multi-copy PRoPHET achieves delivery performance similar to that of single-copy Parasitic Routing. Multi-copy Parasitic Routing shows delivery performance similar to that of Epidemic Routing. These results suggest that for this data set, encounters generally lasted long enough to transmit most of the data, meaning bandwidth is not a severe bottleneck when the message size is small.

In order to evaluate performance with limited bandwidth as a function of message size, the experiment is repeated with fixed transfer speeds of 100Kbps. As shown in figure 7.3 b) the performance of Epidemic Routing decreases significantly when the message size increases to 100K. Multi-copy Parasitic Routing shows virtually identical performance. Interestingly,

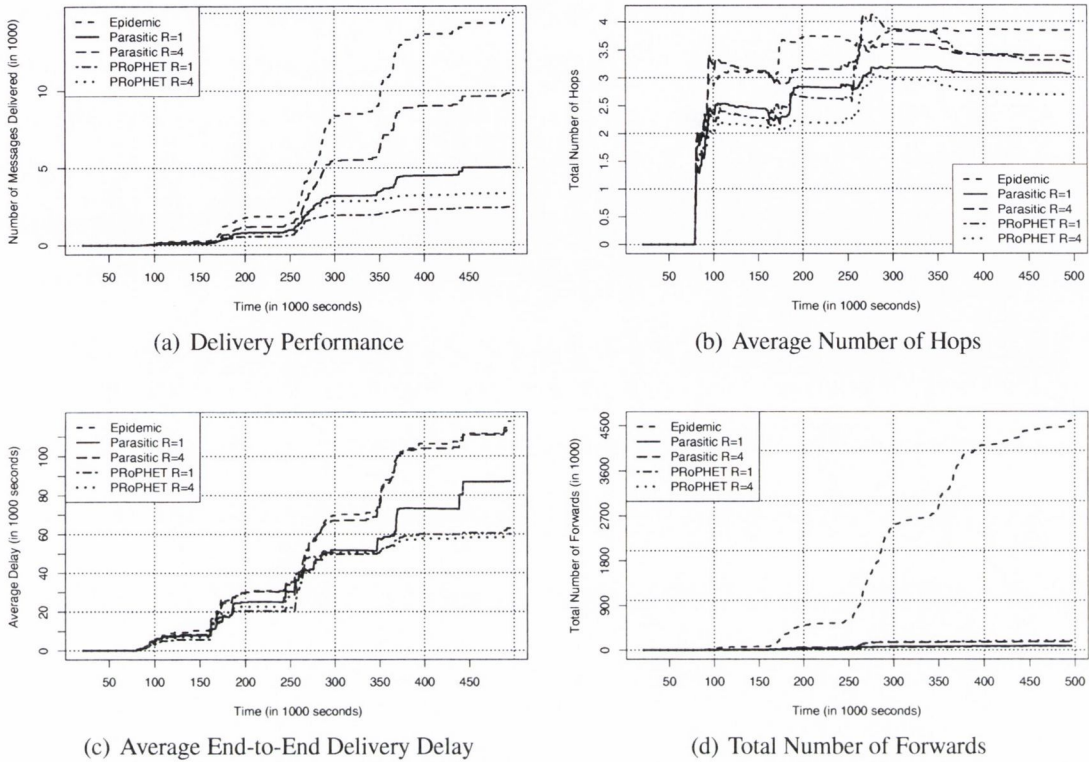


**Fig. 7.4:** Effect of Buffer Capacity for Intel Data

single-copy Parasitic Routing actually outperforms Epidemic Routing and multi-copy Parasitic Routing when the message size is 100K. This illustrates that the benefit of replication is reduced when the opportunity to transfer replicated messages is limited. Single-copy and multi-copy PRoPHET result in the lowest overall message delivery. As the message size increases, the single-copy and multi-copy strategies converge in performance. PRoPHET shows a drop in performance but it is a less significant drop than that of Epidemic Routing.

### 7.2.1.3 Limited Capacity

The above evaluation limits the node transmission capabilities, but retains the assumption that node buffer space is unlimited. In order to evaluate performance under limited memory space capabilities, we perform a set of experiments in which the message queue capacity is varied from 1MB to 50MB. The message size and transmission speed are fixed at 1K and 100Kbps respectively. For Epidemic Routing, a FIFO drop strategy is used. In the case of Parasitic Routing and PRoPHET, a DLU strategy is used. Figure 7.4 shows the results. The performance of Epidemic Routing is greatly affected at buffer capacity of 1MB. In this case, both single-copy and multi-copy Parasitic Routing and PRoPHET outperform Epidemic Routing, and Parasitic Routing achieves the best results. Multi-copy Parasitic Routing continues to outperform Epidemic Routing until the capacity increases to 10MB at which time they achieve the same results. Epidemic Routing requires a buffer size of 20MB in order to perform optimally. Single-copy Parasitic Routing and PRoPHET only require 5MB before limited capacity no longer becomes a limitation on delivery performance. Multi-copy PRoPHET achieves results similar to that of single-copy Parasitic Routing. Multi-copy Parasitic Routing requires 10MB before the perfor-



**Fig. 7.5:** Baseline Protocol Performance for Cambridge Data

mance remains steady. Consequently, it can be determined that Parasitic Routing outperforms Epidemic Routing when node capacity is very low.

## 7.2.2 Cambridge Data Set

The second data set included twelve doctoral students and faculty from Cambridge University Computer Lab [24]. The experiment lasted 5 days, and a total of 223 devices, including the participants, were encountered during the experiment.

### 7.2.2.1 Baseline

Figure 7.5 graphs the results from the baseline performance tests, and table 7.6 shows the statistics. Plateaus are again evident across all protocols where message delivery remains level. There are three such plateaus around the 200, 300 and 400 marks are again night time where devices are inactive. The first night occurred before the message sending phase commenced and

the 200, 300 and 400 represent the 2nd, 3rd and 4th night of the experiment respectively. As expected, Epidemic Routing shows superior delivery performance compared to that of P<sub>Ro</sub>PHET and Parasitic Routing as shown in figure 7.5 a). Single-copy Parasitic Routing outperforms both single-copy and multi-copy P<sub>Ro</sub>PHET. Single-copy Parasitic Routing achieves improved delivery performance of 100% when compared to single-copy P<sub>Ro</sub>PHET. More dramatically, multi-copy Parasitic Routing achieved improved delivery performance of 300% when compared to that of multi-copy P<sub>Ro</sub>PHET.

Figure 7.5 b) shows the average number of hops. As with the previous experiment, all protocols achieve message delivery in a relatively small number of hops of around 3-4. Epidemic Routing results in the highest average. It could be assumed that this is due to the higher number of messages delivered by Epidemic Routing resulting in a number of messages that required a larger number of hops in order to reach the destination. However, upon inspection of the message delivery graph, the average number of hops remains level for Epidemic Routing even after a large proportion of the messages are delivered. This can be explained by the fact that Epidemic Routing finds the optimum path in terms of delivery delay rather than the shortest path. Multi-copy P<sub>Ro</sub>PHET results in the least number of hops. Interestingly, multi-copy Parasitic Routing results in a higher average when compared to single-copy Parasitic Routing, but this increase around the 250 mark coincides with a sharp increase in message delivery. After this time, the average number of hops starts to decrease.

Figure 7.5 c) shows the average message delay. Epidemic Routing shows the highest overall delay, but it can be noted that the delay increases most sharply around the 350 mark. Upon inspection of the delivery graph 7.5 a) this increase can be explained by the increase in total messages delivered, representing the fact that the messages delivered during this time phase were unable to be delivered in a shorter time. Approximately 30% of the encountered nodes do not appear in the network until after the 300 mark. P<sub>Ro</sub>PHET shows the lowest overall delivery delay due to the fact that it delivered a smaller proportion of the total messages. Parasitic Routing results in an average end-to-end delay between that of P<sub>Ro</sub>PHET and Epidemic Routing. Similar to the average hop count, multi-copy Parasitic Routing shows an increase in average end-to-end delay when compared to single-copy Parasitic Routing. The increase in delay is clearly related to the increase in message delivery, and follows a trend almost identical to that of Epidemic Routing.

	Message Delivery		Average Number of Hops		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	14821.80	73.08	3.84	0.01	10.00
Prophet R=4	3372.30	44.80	2.67	0.05	10.00
Parasitic R=4	9814.40	96.69	3.38	0.02	10.00
Prophet	2460.30	64.18	3.27	0.15	10.00
Parasitic	5044.20	55.55	3.06	0.03	10.00

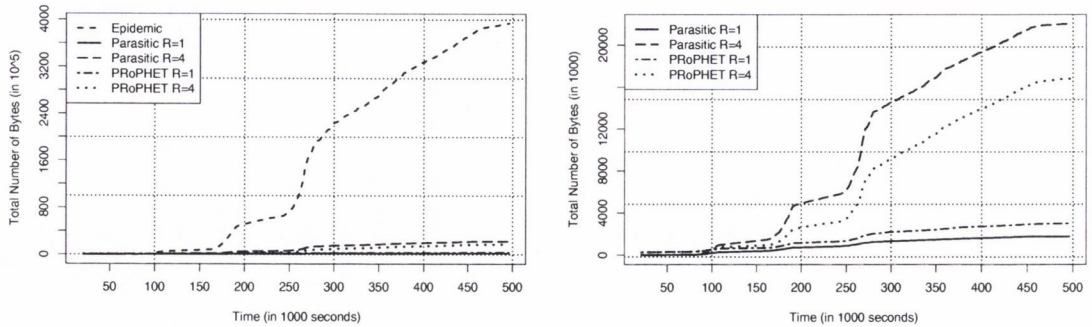
	Average End-to-End Delay		Total Number of Forwards		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	117615.90	321.11	4587653.30	21254.75	10.00
Prophet R=4	60089.97	603.84	188636.70	2870.29	10.00
Parasitic R=4	113903.65	595.10	168261.60	337.07	10.00
Prophet	63046.79	1390.96	73814.10	1606.37	10.00
Parasitic	87153.66	865.35	82251.30	165.86	10.00

**Table 7.4:** Baseline Protocol Performance Statistics for Cambridge Data Statistics

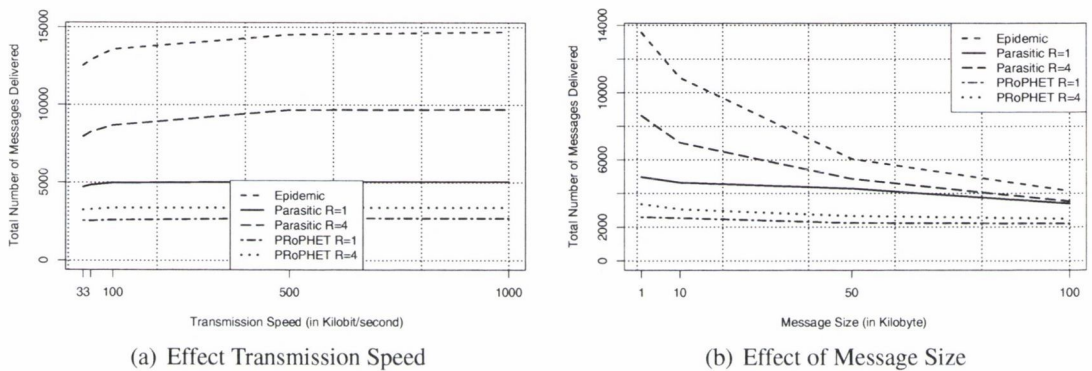
Figure 7.5 d) shows the total number of message forwards throughout the simulation. As with the previous data set, Epidemic Routing clearly results in an exponentially large number of forwards, which is to be expected with a flooding protocol. Single-copy Parasitic Routing and PRoPHET result in a relatively low number of forwards. This value increases for multi-copy Parasitic Routing and PRoPHET. Unlike the previous data set, multi-copy PRoPHET results in a higher number of forwards than that of multi-copy Parasitic Routing. As a result, it can be determined that this value is dependent on the dynamics of the underlying network rather than a deterministic side effect of the protocol.

Figure 7.6 shows the total overhead associated with the control data for each protocol. Epidemic Routing as expected increases dramatically as the number of messages on the network increases. The overhead associated with PRoPHET and Parasitic Routing is shown in figure 7.6. Both single-copy Parasitic Routing and PRoPHET protocols increase at approximately the same rate, however, single-copy Parasitic Routing results in the lowest number of bytes. Multi-copy Parasitic Routing results in a larger amount of control data when compared to multi-copy





**Fig. 7.6:** Baseline Control Data Overhead for Cambridge Data



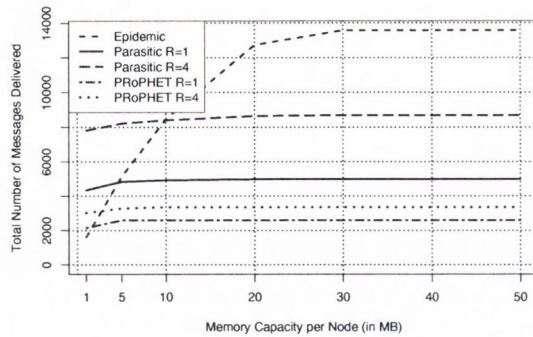
**Fig. 7.7:** Effect of Limited Bandwidth for Cambridge Data

PRoPHET, however, both protocols follow a similar trend.

### 7.2.2.2 Limited Bandwidth

Figure 7.7 a) shows the results of limited bandwidth tests on varying transmission speeds. Single-copy Parasitic Routing shows a slight drop in performance at transfer speeds of 33 Kbps and 50 Kbps, however, at speeds of 100 Kbps and above the performance remains constant. PRoPHET produces steady results at all transfer speeds. Epidemic Routing, as expected is the most affected under limited transfer speed capabilities, requiring transfer speeds of 500 Kbps in order to achieve full delivery. Multi-copy Parasitic Routing is similarly affected.

Figure 7.7 b) shows the results of limited bandwidth with varying message size assuming a fixed transfer speed of 100 Kbps. Single-copy Parasitic Routing shows a slight decline in performance as the message size approaches 100K. PRoPHET shows a similar but less pronounced decline. As expected, Epidemic Routing shows the worst degradation in performance,



**Fig. 7.8:** Effect of Buffer Capacity for Cambridge Data

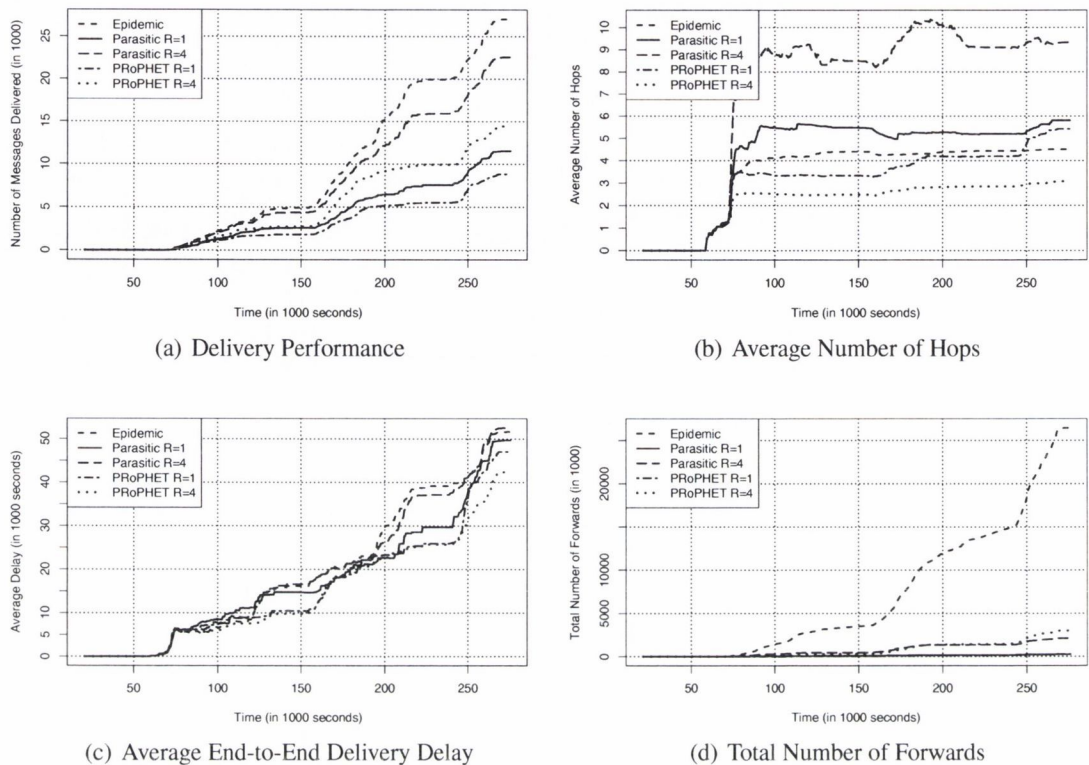
but it still outperforms both Parasitic Routing and PRoPHET. Multi-copy Parasitic Routing shows similar degradation. As the message size approaches 100K, the delivery performance of Epidemic Routing and Parasitic Routing start to converge.

### 7.2.2.3 Limited Capacity

Figure 7.8 shows the results of varying the message queue capacity assuming a fixed message size of 1K and a fixed transfer speed of 100 Kbps. Similar to the Intel data set, Epidemic Routing achieves the lowest overall message delivery at a buffer size of 1MB and performs the same as single-copy Parasitic Routing at a buffer size of 5MB. Multi-copy Parasitic Routing outperforms Epidemic routing until the buffer size reaches 10MB at which time the results are identical. Both single-copy and multi-copy PRoPHET show the lower message delivery and reach the maximum performance at buffer size 5MB.

### 7.2.3 Infocom Data Set

The third experiment collected data of encounters using iMotes equipped with Bluetooth distributed among conference attendees at the IEEE 2005 Infocom conference [58]. The participants were chosen in order to represent a range of different groups belonging to different organisations. Participants were asked to carry the devices with them for the duration of the conference. The experiment lasted 3 days and encounters between 264 devices were recorded.



**Fig. 7.9:** Baseline Protocol Performance for Infocom Data

### 7.2.3.1 Baseline Performance Evaluation

Figure 7.9 shows the baseline performance for the Infocom data set. Figure 7.9 a) shows the overall message delivery performance. Epidemic Routing shows significant improvement after the 150 mark. This can be explained by the fact that approximately 34% of the node population appear for the first time after this time frame. At this point, Parasitic Routing and PRoPHET start to build up encounter information in regards to these nodes. This explains why Parasitic Routing starts to out-perform PRoPHET after the 150 mark, because messages destined for nodes that have yet to be encountered are already routed to more central nodes which are more likely to gather encounter information more quickly about the previously unseen nodes. Multi-copy Parasitic Routing achieves message delivery close to Epidemic and outperforms PRoPHET.

Figure 7.9 b) shows the average number of hops of delivered messages. Epidemic Routing results in hop lengths of approximately 4. Single-copy Parasitic Routing and PRoPHET achieve

	Message Delivery		Average Number of Hops		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	27052.70	84.66	4.50	0.01	10.00
Prophet R=4	14465.90	86.49	3.08	0.01	10.00
Parasitic R=4	22575.10	82.81	9.33	0.05	10.00
Prophet	8828.22	184.18	5.43	0.17	10.00
Parasitic	11517.50	95.51	5.82	0.03	10.00

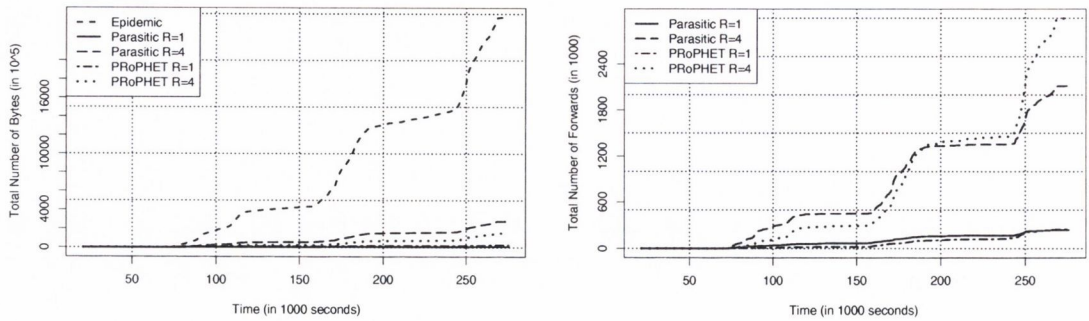
  

	Average End-to-End Delay		Total Number of Forwards		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	51619.43	203.55	26437449.40	67467.75	10.00
Prophet R=4	42514.64	234.32	2999041.90	12546.18	10.00
Parasitic R=4	52536.07	298.14	2119212.40	3761.09	10.00
Prophet	47025.95	1685.28	248830.67	8239.56	10.00
Parasitic	49671.18	501.64	240261.80	553.26	10.00

**Table 7.5:** Baseline Protocol Performance Statistics for Infocom Data

similar results of between 5 and 6 hops. However, PROPHET results in the lowest number of hops. Multi-copy PROPHET results in the lowest number of hops. In contrast, multi-copy Parasitic Routing shows significant increase, resulting in hops of approximately 9. This increase is due an increased path length of the additional messages delivered by multi-copy Parasitic Routing that remained undelivered for single-copy Parasitic Routing. Figure 7.9 c) shows the average end-to-end delay. Even with the increased path lengths used by multi-copy Parasitic Routing, the average end-to-end delay is similar to that of Epidemic Routing. The sharpest increase occurs for Epidemic after the 150 mark which is most likely when messages are delivered to nodes previously unseen nodes on the network. Parasitic Routing shows a similar increase. PROPHET shows the lowest average delay, however, the fact that it shows delays lower than that of Epidemic Routing illustrates that the increase in message delay shown by Epidemic is caused by the additional messages delivered, which required a greater amount of time before it was possible to deliver these messages.

Figure 7.9 c) shows the average message delay. Similarly to the Intel and Cambridge data



**Fig. 7.10:** Baseline Control Data Overhead for Infocom Data

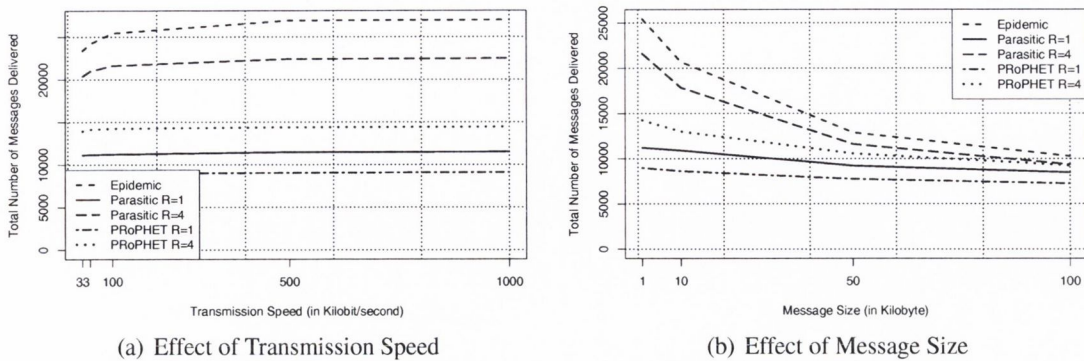
sets, Epidemic Routing shows the greatest average delay due to the higher proportion of messages delivered. The sharpest increase occurs after for Epidemic after the 150 mark which is most likely due to message delivery of messages destined for the previously unseen on the network. Parasitic Routing shows a similar increase just before the 250 mark which coincides with an increase in message delivery. Consequently, this increase also coincides with the messages delivery to nodes that have yet to be encountered.

Figure 7.9 d) shows the total number of messages forwarded in the network. As expected, Epidemic Routing results in the highest number of total forwards. Parasitic routing and PRoPHET result in a similar number of forwards when comparing the single-copy and multi-copy strategies. PRoPHET results in a larger number of forwards than Parasitic Routing.

Figure 7.10 shows the total control data associated with all protocols. The overhead of Epidemic Routing increases dramatically as the number of messages on the network also increases. As seen with the other data sets, single-copy Parasitic Routing and PRoPHET result in a similar amount of overhead. Multi-copy PRoPHET begins to show a climb in control data around the 250 mark. Multi-copy Parasitic Routing shows a much lower increase. This can be explained by the large number of messages delivered at this time, as a result, the size of the summary vectors exchanged between nodes decreases. As the number of messages on the network that remain undelivered by multi-copy PRoPHET, so to does the overhead.

### 7.2.3.2 Limited Bandwidth

Figure 7.11 a) shows the effect of varying the data transfer speed between nodes. All protocols show a slight decrease in performance at transfer speeds of 33Kbps. As with the Intel data set,



**Fig. 7.11:** Limited Bandwidth for Infocom Data

transfer speed has little effect on performance of all protocols, suggesting that for the Infocom data set contacts generally last long enough for complete messages transfer to take place when the message size is small.

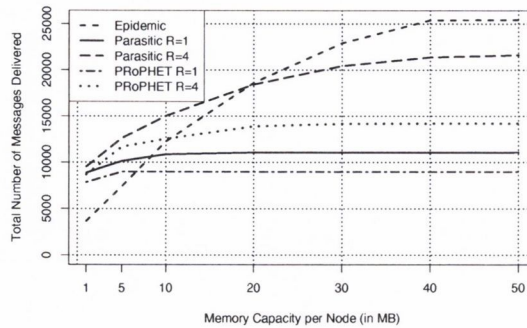
Figure 7.11 b) shows overall message delivery for varying message sizes. As expected, Epidemic Routing shows the worst performance degradation as the message size increases. Multi-copy Parasitic Routing shows similar degradation. Single-copy Parasitic Routing and PRoPHET are much less affected by increases in message size, though both show a slight decrease.

### 7.2.3.3 Limited Capacity

Figure 7.12 shows the results of limiting the message buffer space available for each node in the network. Parasitic Routing and PRoPHET outperform Epidemic at 1MB and 5MB. In this case, Epidemic Routing requires 40MB of buffer space in order to achieve maximum results. PRoPHET reaches optimum performance at 5MB where as Parasitic Routing requires 10MB before limited buffer space is no longer a limitation.

## 7.2.4 Discussion

In all three data sets Parasitic Routing outperformed PRoPHET in message delivery. With the benefit of replication, multi-copy Parasitic Routing achieved delivery performance close to that of Epidemic in two of the three data sets. The average number of hops was low in two of the three data sets. In the case of the Infocom data set, Parasitic Routing resulted in a higher than



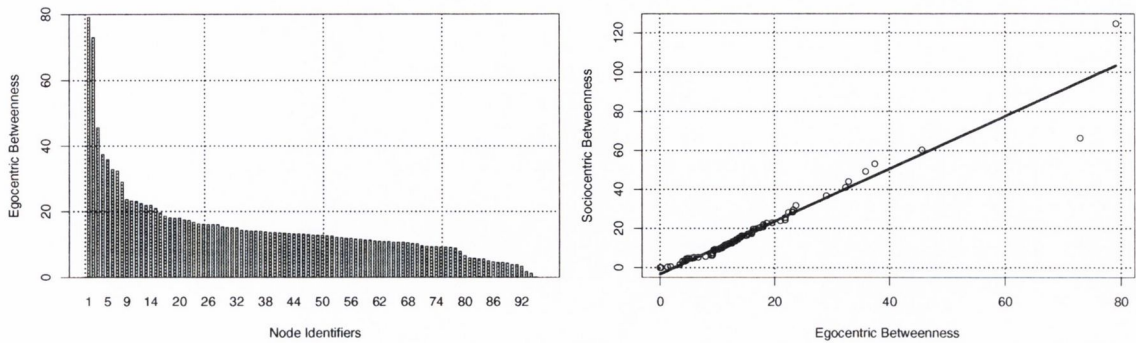
**Fig. 7.12:** Effect of Buffer Capacity for Infocom Data

average number of hops. However, this did not affect the end-to-end delay resulting in delays similar to that of Epidemic. The high delivery performance was achieved with significantly less overhead than Epidemic Routing. Additionally, Parasitic Routing outperformed Epidemic Routing when node storage capacity is low, which highlights that Parasitic Routing requires less storage capacity in order to provide good delivery performance.

The Huggle data set provides a useful environment to explore routing using Bluetooth encounters, however, there are a number of limitations of the data set. The participating number of nodes included in the experiment is quite small and the duration of each experiment is relatively short. Though external devices are recorded and may be used as potential contacts, both sides of the encounter are unavailable, so it is unclear whether both devices detected the encounter. Consequently, it is impossible to distinguish between asymmetric and symmetric encounters. Additionally, it is difficult to derive much information regarding the underlying structure of the network in order to validate how well Parasitic Routing protocol manages to infer the underlying network structure. The next section presents a case study where such information is available.

### 7.3 Case Study: MIT Reality Mining Data Set

One of the largest studies based on Bluetooth encounters is that of MIT Reality Mining project [4, 33]. The study consisted of 100 users carrying Nokia 6600 smart phones over the course of nine months. The researchers collected information using call logs, Bluetooth devices in proximity, cell tower IDs, application usage and phone status. Bluetooth sightings are used



(a) Betweenness Distribution of nodes from MIT trace (b) Scatterplot of sociocentric vs. egocentric betweenness for MIT trace

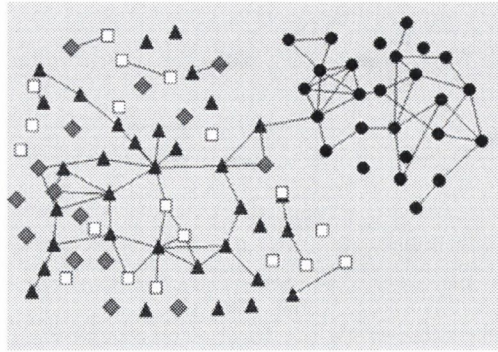
**Fig. 7.13:** Betweenness Distribution

in order to identify direct contacts between nodes where data transfer could have taken place. The MIT Reality Mining data set provides node encounters in a realistic social network. As a result, this data set may be used in order to evaluate to what extent Parasitic Routing is able to determine the underlying network structure based on local information.

Figure 7.13 a) shows the distribution of ego betweenness values calculated for all nodes at the end of the simulation. As can be seen, there are two primary nodes that have a high ego betweenness. These nodes may represent highly social people who serve to link some of the less sociable individuals. Interestingly, this corresponds well to the social structure shown in figure 7.14 which the MIT Reality Mining team derived from interviews with the participants as to who they spent time with, both in the workplace and out of the workplace, and who they would consider to be in their circle of friends [4, 33]. The authors observed two distinct cliques with two nodes linking them.

In order to validate how accurately the locally calculated ego betweenness value correlates to a global view of the network, the UCINET software for Social Network Analysis was used to calculate the sociocentric betweenness value using global information of the topology [13]. Figure 7.13 b) shows a scatterplot of the egocentric betweenness vs. the sociocentric betweenness which exhibit a very close correlation. The two outlier points represent the two most central nodes, and although the egocentric value does not directly map to the sociocentric value in these cases, the relative ranking of the two nodes is identical. Pearson's correlation of egocentric and sociocentric betweenness was used in order to evaluate the quality of the cor-





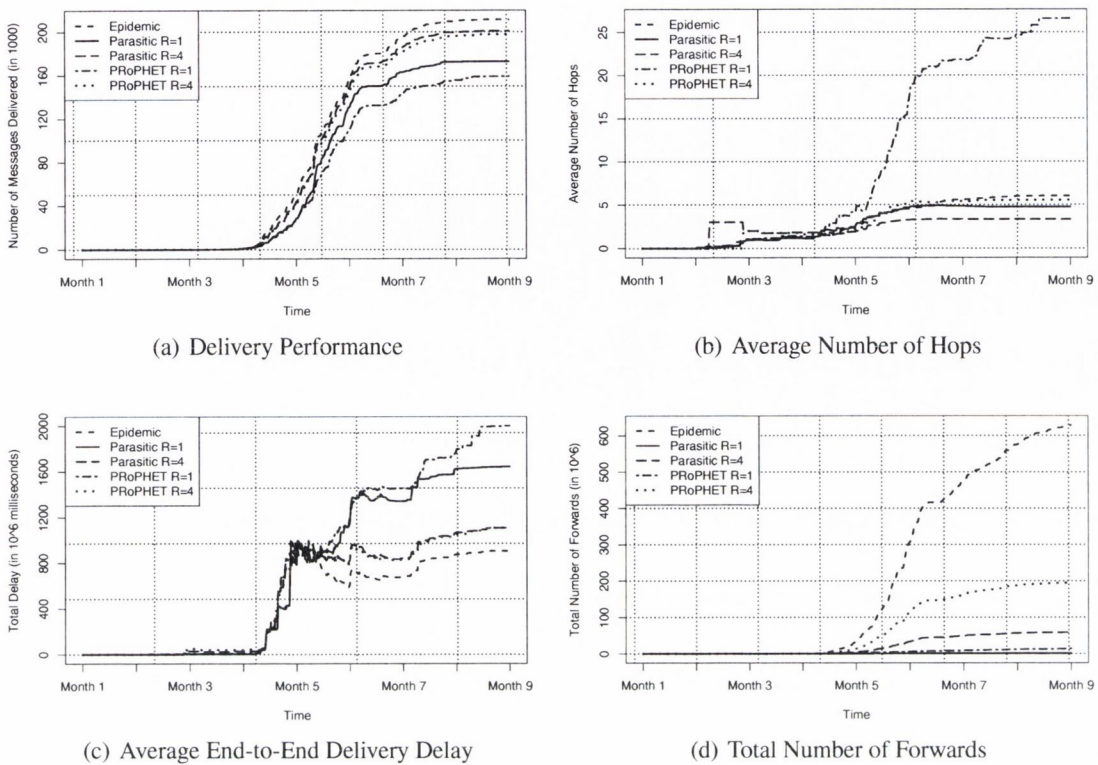
**Fig. 7.14:** Friendship network Eagle and Pentland [33]

relation. The Pearson's correlation value is 0.971 and thus egocentric betweenness reflects the comparative centrality of nodes for this network very well.

In the MIT Reality Mining data set, the encounters of all participants is available. Consequently, it is possible to differentiate between symmetric and asymmetric encounters where both nodes observed the encounter. All Bluetooth encounters between participants are used to generate a trace-based simulation, as discussed in section 6.1. An asymmetric encounter is one that is not reciprocated by the encountered node. In this case the encounter is recorded by the observing node, but the encountered node remains unaware of the encounter. It is assumed that data transfer may only take place when both nodes have observed the encounter. Due to clock synchronisation problems, there is some discrepancy in the observed contact duration. When this is the case, the average of the two duration values is used to determine the duration of the contact and evaluate how much data may be exchanged. In tests and unless otherwise stated, nodes generate two messages per day the node appears on the network, after an initial 15% warm up phase. Message destinations are selected uniformly randomly and the tests are repeated 10 times with different random seeds.

### 7.3.1 Baseline

The results of the baseline simulations of all protocols using the MIT data set assuming infinite bandwidth and buffer capacity are shown in figure 7.15. It appears from the graphs that no messages are delivered until the end of month three. However, this is not the case and is a consequence of the large amount of messages generated for this simulation along with the



**Fig. 7.15:** Baseline Protocol Performance for MIT Data

scale of the graph.

Epidemic Routing outperforms both Parasitic Routing and PRoPHET in the total number of messages delivered as shown in figure 7.15 a). Single-copy Parasitic Routing outperforms single-copy PRoPHET particularly towards the end of the the simulation. Multi-copy Parasitic Routing and multi-copy Parasitic Routing achieve similar delivery performance.

Figure 7.15 b) shows the average number of hops. The messages delivered by single-copy and multi-copy Parasitic Routing are delivered in an average of 5 hops which is slightly below that of Epidemic Routing which results in paths of approximately 6 hops. Single-copy PRoPHET results in some very long delivery paths of approximately 25 hops. However, this result improves significantly for multi-copy PRoPHET.

Figure 7.15 c) shows the average end-to-end delay of messages delivered for each protocol. Due to the long delivery paths taken by single-copy PRoPHET, the average end-to-end delay is the highest. Single-copy Parasitic Routing results in an average end-to-end delay lower than single-copy PRoPHET but significantly higher compared to Epidemic Routing, which achieves

	Message Delivery		Average Number of Hops		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	211259.89	185.63	6.01	0.02	10.00
Prophet R=4	197609.00	70.71	5.54	0.51	10.00
Parasitic R=4	200769.40	180.26	3.33	0.14	10.00
Prophet	158986.40	4352.34	26.57	1.90	10.00
Parasitic	172684.33	175.91	4.74	0.66	10.00

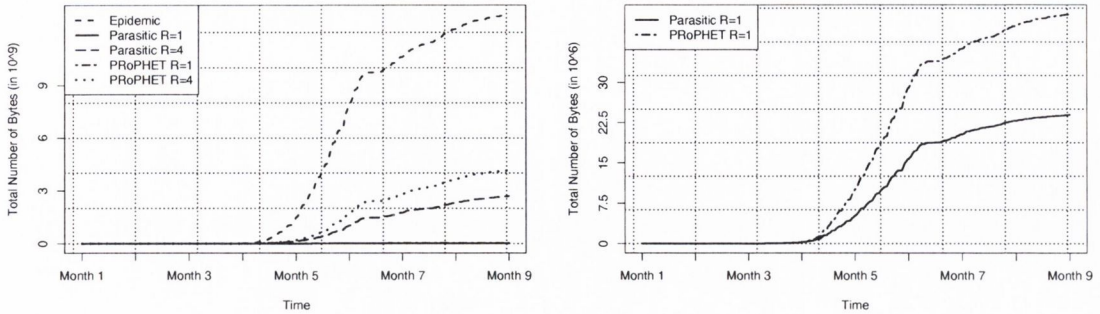
	Average End-to-End Delay		Total Number of Forwards		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	931711976.61	3843519.38	628872893.89	786611.81	10.00
Prophet R=4	1141111476.29	4123549.21	194659421.00	413615.65	10.00
Parasitic R=4	1141050444.54	3925576.00	58599461.00	411446.94	10.00
Prophet	2062033746.93	182955678.7	13834633.30	827231.29	10.00
Parasitic	1695042771.34	351740409.7	944439.10	205723.86	10.00

**Table 7.6:** Baseline Protocol Performance Statistics for MIT Data Statistics

the lowest end-to-end delivery delay. Multi-copy Parasitic Routing and PRoPHET improve upon delay performance when compared to the single-copy strategy. Both protocols result in almost identical delays.

Figure 7.15 d) shows the total number of forwards of messages on the network. As expected, Epidemic Routing results in significantly more forwards than Parasitic Routing and PRoPHET. Single-copy Parasitic Routing and PRoPHET result in the smallest number of forwards. Multi-copy Parasitic Routing causes an increase in the number of forwards, but the increase is much lower when compared to multi-copy PRoPHET. PRoPHET results in more forwards when compared to that of Parasitic Routing, which is primarily due to the long paths taken by messages in order to reach the destination.

The control data overhead generated by each protocol is shown in figure 7.16. As the number of messages on the network increases so too does the size of the summary vector which causes a very high amount of overhead to the point that the overhead generated by single-copy Parasitic Routing and PRoPHET is unintelligible on the graph. Figure 7.16 shows the con-



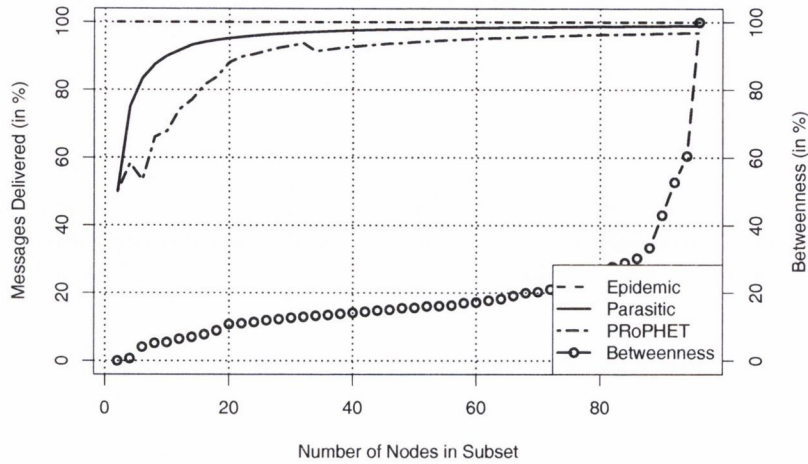
**Fig. 7.16:** Baseline Control Data Overhead for MIT Data

control data overhead of single-copy Parasitic and PRoPHET. Both multi-copy and single-copy PRoPHET shows an increase in overhead when compared to Parasitic Routing. This is caused by exchanging probability data for all nodes encountered directly and indirectly, whereas Parasitic Routing exchanges only data about directly encountered nodes and probabilities of nodes for which the routing node is currently carrying messages.

## Delivery Performance Between Least Connected Nodes

Given the fact that the MIT Reality Mining data set reflects a real social network structure, this enables an evaluation of message delivery between nodes based on the betweenness centrality of each node. Consequently, the baseline test is repeated where each node included in the sending and receiving subset generates a single message for each other node. The first subset includes only the least connected nodes as defined by the betweenness distribution shown in figure 7.13 a). Each subsequent simulation increases the subset of nodes including the next two in terms of increasing betweenness.

Figure 7.17 shows the delivery performance of Epidemic Routing, PRoPHET and Parasitic Routing for each simulation. The betweenness series shows the normalised maximum betweenness value of the subset of nodes. As can be seen from the figure, the performance of PRoPHET increases as the betweenness value of the subset of nodes increases. When considering message delivery between the 20 least connected nodes, the percentage of message delivery for PRoPHET is low. In contrast, Parasitic Routing performs better with an improvement as high as 25 percentage points when considering the 6 least connected nodes. This is achieved by routing from disconnected nodes to a more central node in order to find a good carrier for



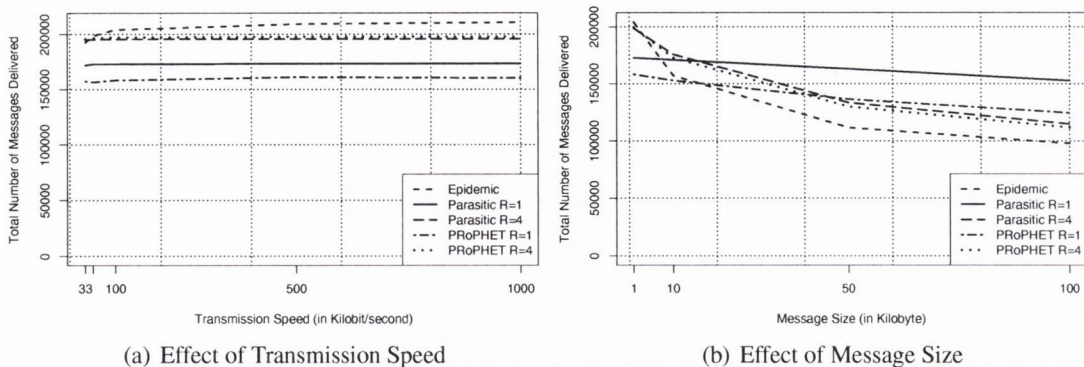
**Fig. 7.17:** Percentage of message delivered between a subset of nodes increasing based on increased Betweenness

the message.

### 7.3.2 Limited Bandwidth

Figure 7.18 b) shows the effect of varying the data transfer speed. Epidemic Routing shows a drop in performance at transmission speeds of below 100Kbps, after which, the decreased performance is relatively small and reaches full capabilities at 500Kbps. Parasitic Routing and PRoPHET show little change as the transmission speed increases. Consequently, the performance of all the protocols are relatively unaffected when the message size is small.

Figure 7.18 b) shows the message delivery performance assuming a transmission speed of 100Kbps with message sizes ranging from 1K to 100K. Epidemic Routing performs similarly to multi-copy Parasitic Routing and PRoPHET at a message size of 1K. The performance of Epidemic Routing steadily decreases as the message size increases above 1K. Multi-copy Parasitic Routing and PRoPHET show a similar declining trend although the degradation in performance is less severe. Single-copy Parasitic Routing and PRoPHET are less affected by the increase in message size, and both protocols outperform the multi-copy strategies at message sizes of 100K. Single-copy Parasitic Routing outperforms all protocols when the message size increases to 50K. However, both protocols and single-copy PRoPHET outperform Parasitic Routing for message sizes of 1K and 10K. The poor performance of Epidemic Routing



**Fig. 7.18:** Effect of Limited Bandwidth for MIT Data

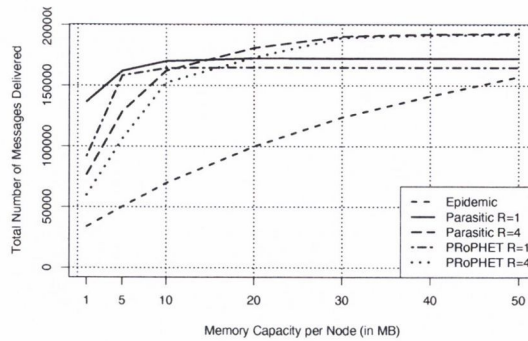
and the multi-copy strategies as message size increases is due to the fact that nodes may not exchange all messages within a single encounter. The benefit of a single copy message strategy is clear in this experiment as single-copy Parasitic Routing and PRoPHET both decrease with the same rate, however, the decrease in performance is significantly less than that of Epidemic.

### 7.3.3 Limited Capacity

Figure 7.19 shows the results of varying the message queue capacity assuming a fixed message size of 1K and a fix transfer speed of 100 Kbps. Epidemic Routing is significantly affected, achieving delivery performance less than either of the single-copy strategies. Single-copy Parasitic Routing and PRoPHET show decreased below buffer size of less than 10MB. Limited capacity is no longer a concern for single-copy PRoPHET at buffer capacity of 10MB, single-copy Parasitic requires 20MB. Miltie-copy Parasitic Routing and PRoPHET show a marked decrease in performance at buffer capacity below 10MB. However, this improves at buffer capacity of 20MB, after which it improves slowly and does not reach full performance at the maximum capacity of 50MB. Consequently, for this data set due to the long duration of the experiment and the large amount of messages generated, buffer capacity is a limitation for all replication strategies.

### 7.3.4 Discussion

The MIT Reality Mining data set provided a large-scale long-term representation of the social interactions of devices carried by people in the real world. This data has enabled the evaluation



**Fig. 7.19:** Effect of Buffer Capacity for MIT Data

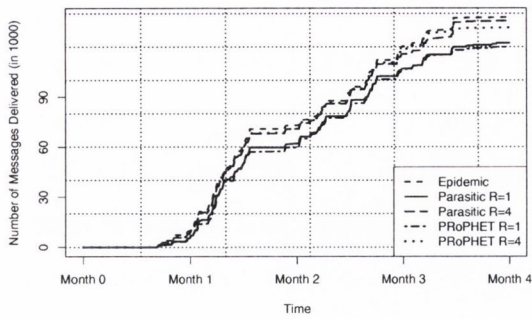
of the extent to which Parasitic Routing is able to capture an underlying social structure of the network. Additionally, by routing to the least connected nodes in the network, it has highlighted the benefit of using routing based on a combination of social ties and betweenness centrality where Parasitic Routing achieve marked improvement when compared to PRoPHET.

## 7.4 Case Study: UMassDieselNet

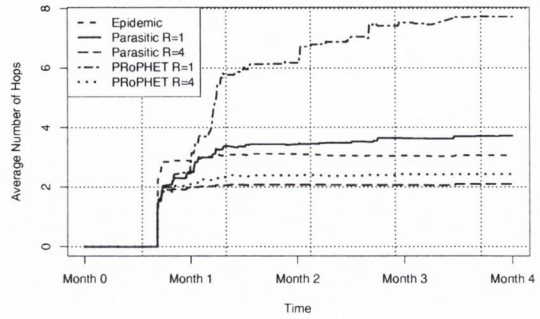
The previous data sets have been based encounters between devices carried by people. There are a number of advantages for this in the context of message delivery in disconnected networks. People move slowly and general walking speeds are in the range of 1-6mph. As a consequence, even low transfer speeds can enable data exchange of up to 1MB in an encounter lasting 10 seconds [57]. One application area of interest is that of inter-vehicle communication. The UMassDieselNet data consists of encounters between 30 city buses using 802.11b access points attached to the buses during weekdays over the course of four months in Amherst, Massachusetts [20]. Buses are assigned routes each day and therefore the same bus may not necessarily always travel the same route. The data set consists of the encounters between buses, each containing: the start time, the bus identifier recording the encounter and the current route it is traveling, the encountered bus identifier and the route that the encountered bus is traveling on<sup>1</sup>. Additionally, the authors recorded the amount of data in bytes that is received by the encountered node. The amount of data that was exchanged between buses was asymmetric.

The data was parsed in order to differentiate between asymmetric encounters and recipro-

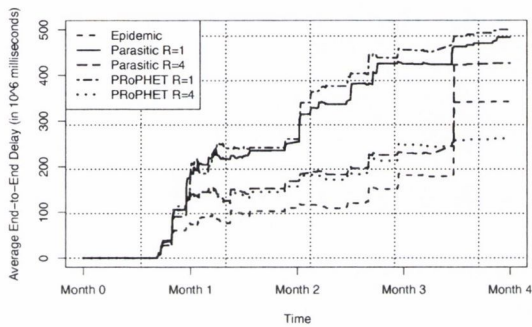
<sup>1</sup>GPS readings were also recorded, however, the authors state that this data is inconsistent and that use of this information is discouraged [20].



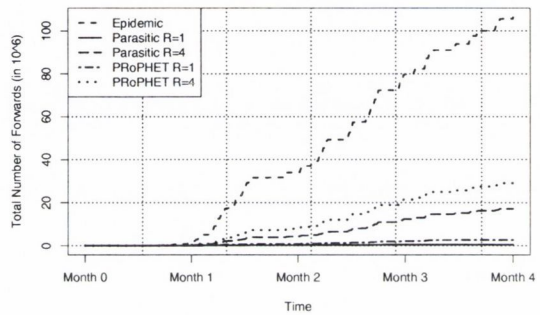
(a) Delivery Performance



(b) Average Number of Hops



(c) Average End-to-End Delivery Delay



(d) Total Number of Forwards

**Fig. 7.20:** Baseline Protocol Performance for UMassDieselNet Data

cated symmetric encounters between buses where more than zero bytes were exchanged in both directions. As with the MIT Reality Mining Data, an asymmetric encounter is recorded by the observing node, while the encountered node remains unaware of the encounter. Messages are only transferred during reciprocated encounters. The resulting data set was used to generate a trace-based simulation where the total number of bytes to be exchanged by encountered nodes is known. A message generation rate is assumed of 20 messages per hour the bus appears on the network after an initial 15% warm up period. The destination node for each message is uniformly randomly selected. Each simulation is repeated 10 times with different random number seeds.

### 7.4.1 Baseline

The baseline performance comparison for the UMassDieselNet data set is shown in figure 7.20. Epidemic Routing achieves the best delivery performance as shown in figure 7.20 a). Single-



copy Parasitic Routing achieves slightly improved performance when compared to single-copy P<sub>Ro</sub>PHET, but the overall message delivery is comparable. Upon analysis of the interactions between the buses in the data set, it turned out that buses may be assigned a different route each day. As a result, the majority of the buses encountered a similar number of buses over the course of the experiment. Given the low number of routes (12), the emergence of an underlying social network is reduced. Multi-copy Parasitic Routing achieves delivery performance similar to Epidemic Routing. Multi-copy P<sub>Ro</sub>PHET shows an improved delivery also, but again lower than that of multi-copy Parasitic Routing.

The average number of hops of delivered messages is shown in figure 7.20 b). Epidemic Routing and Parasitic Routing achieve similar average number of hops of approximately 3. Single-copy P<sub>Ro</sub>PHET results in a larger average number of hops, however, this improves for multi-copy P<sub>Ro</sub>PHET.

The average end-to-end delay in figure 7.20 c) shows all protocols show a sharp increase during the 3rd month which corresponds to a rise in message delivery within the same time frame. Parasitic Routing and P<sub>Ro</sub>PHET show a similar trend in increased message delay with P<sub>Ro</sub>PHET resulting in the highest delivery delay. Single-copy and multi-copy Parasitic Routing show similar delays at the end of the simulation. In contrast, P<sub>Ro</sub>PHET single copy and multi-copy result in the largest and the lowest average end-to-end respectively. This shows that Parasitic Routing finds similarly short paths to the destination node, with or without replication.

The total number of forwards on the network is shown in figure 7.20 d). As expected, the number of forwards caused by Epidemic Routing increases dramatically as the number of messages on the network increases. Single-copy P<sub>Ro</sub>PHET and Parasitic result in a low number of forwards when compared to Epidemic Routing. As expected, multi-copy Parasitic Routing and P<sub>Ro</sub>PHET result in an increased number of forwards, however, multi-copy Parasitic routing results in less than multi-copy P<sub>Ro</sub>PHET.

Figure 7.21 shows the control data overhead. Similar to previous experiments, Epidemic Routing causes the most amount of overhead. The control data caused by single-copy Parasitic Routing and P<sub>Ro</sub>PHET is much less pronounced than in the previous data sets. This is explained by the fact that the average number of buses directly encountered during the course of the experiment is a relatively high proportion of the population (23 out of 30 nodes). Consequently, the difference in overhead associated with exchanging encounter information of directly en-

	Message Delivery		Average Number of Hops		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	137650.70	197.96	3.06	0.02	10.00
Prophet R=4	131479.44	196.93	2.42	0.01	10.00
Parasitic R=4	135524.90	182.14	2.10	0.01	10.00
Prophet	119895.10	2489.20	7.72	1.45	10.00
Parasitic	122332.80	329.84	3.71	0.10	10.00

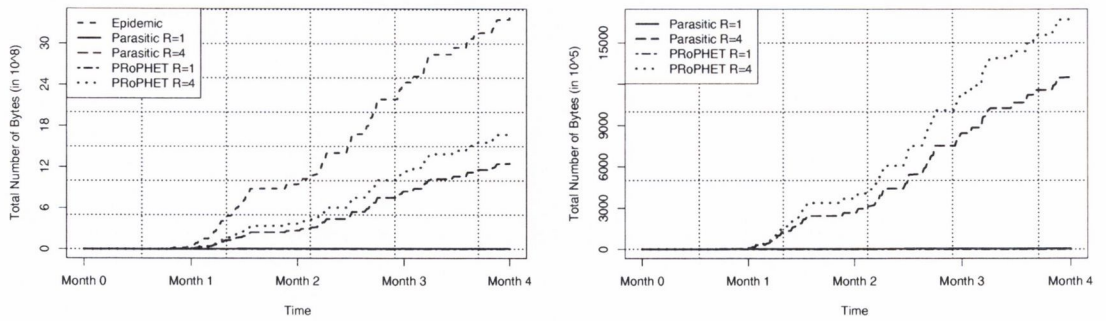
	Average End-to-End Delay		Total Number of Forwards		
	Mean	StdDev	Mean	StdDev	Sample Size
Epidemic	352525880.34	2036482.60	106369061.30	261172.14	10.00
Prophet R=4	269188490.56	988141.94	29204589.44	40729.07	10.00
Parasitic R=4	439125118.79	1971991.29	17173308.50	25918.17	10.00
Prophet	515010513.71	51470031.09	2647045.10	782147.43	10.00
Parasitic	497707113.87	14856902.72	466842.10	13046.75	10.00

**Table 7.7:** Baseline Protocol Performance Statistics for UMassDieselNet Data

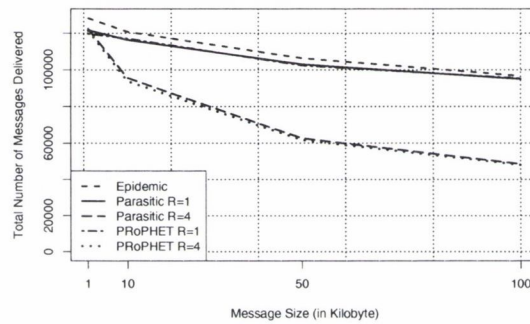
countered nodes, and nodes encountered both directly and indirectly is significantly reduced. Multi-copy Parasitic Routing and PRoPHET result in lower overhead when compared to Epidemic Routing, but show an increased trend when compared to the other data sets.

## 7.4.2 Limited Bandwidth

The UMassDieselNet data set contains explicit information as to the amount of bytes that were actually exchanged during each encounter, but the actual duration of the encounters are unknown. Consequently, the evaluation of varying transfer speeds is not carried out for this data set. The varying message size evaluation is shown in figure 7.22. Epidemic Routing achieves the best delivery at messages size of 1K. PRoPHET and Parasitic Routing show similar delivery for small message sizes. Interestingly, the delivery performance of multi-copy Parasitic Routing and PRoPHET both show a significant decline in performance as the message size increases. In this case, encounter duration were not long enough to exchange all data when limited replication is used. Epidemic is less affected, mainly due to the small node population



**Fig. 7.21:** Baseline Control Data Overhead for UMassDieselNet Data

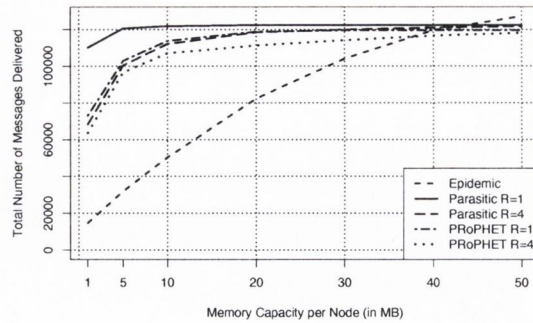


**Fig. 7.22:** Limited Bandwidth Effect of Message Size for UMassDieselNet Data

and the fact that its the heavy use of replication means the chances of at least a subset of nodes managing to transfer the data is sufficient.

### 7.4.3 Limited Capacity

Figure 7.23 shows the delivery performance under the circumstances of limited message queue capacity. Single-copy and multi-copy Parasitic Routing achieve the best overall results for buffer capacity less than 40MB. At buffer capacity of 1MB there is a decrease in performance, however, from sizes of 5MB onwards the performance is steady. Single-copy PRoPHET starts to improve at 10MB but takes until a buffer capacity of 20MB until performance is no longer affected. Epidemic Routing performs the worst overall, only outperforming Parasitic Routing and PRoPHET with a buffer capacity of 50MB. This supports the suspicion from the previous tests, that only a limited subset of nodes manage to deliver the data. When the buffer capacity is limited, the subset of nodes can no longer carry all messages and consequently, performance is severely degraded. Multi-copy Parasitic Routing and PRoPHET are similarly affect, however,



**Fig. 7.23:** Effect of Buffer Capacity for UMassDieselNet Data

the drop in performance is much less severe and improves at a more steady rate.

#### 7.4.4 Discussion

The UMassDieselNet data set has been used to evaluate performance of Parasitic Routing for VANETs (vehicle mobile ad hoc networks). The delivery performance was similar to that of PRoPHET for this type of network. This emphasises that even though Parasitic Routing may be used for such applications, for Parasitic Routing to achieve its best performance, an underlying social structure is preferable. One possible solution would be to ensure that the same Parasitic device is located on a bus following the same routes or subset of routes each day. If this was the case then a social structure would emerge where certain buses on specific routes are more likely to meet buses on a specific route.

### 7.5 Conclusion

This chapter has evaluated Parasitic Routing in terms of message delivery, protocol overhead and ability to achieve message delivery under the restrictions of limited bandwidth and storage capacity. Parasitic Routing has been evaluated using short term data ranging from 3 - 5 days and for long term data ranging from 4 - 9 months. Additionally, it has been evaluated for node population sizes ranging from 30 - 228. It has been demonstrated that Parasitic Routing achieves message delivery in social networks between devices carried by people, and in VANETs where the devices are stored on buses.

In the majority of the baseline experiments multi-copy Parasitic Routing has achieved deliv-

ery performance similar to that of Epidemic Routing. Parasitic Routing results in significantly less overhead. Additionally, the average number of hops and overall delay is comparable to Epidemic Routing. Single-copy Parasitic Routing outperforms though not significantly for the UMassDieselNet data set where the social structure is less prevalent. Additionally, single-copy Parasitic Routing finds short paths lengths comparable to Epidemic Routing when compared to PROPHET which shows varying results. Single-copy PROPHET shows a lower average end-to-end delay than Parasitic Routing. However, this results should be viewed in the context that PROPHET delivered less messages overall. Multi-copy Parasitic Routing shows marked improvement in message delivery with the additional benefit of replication in all three of Haggie data sets. Multi-copy PROPHET on the other hand shows a much lower improvement across these data sets. In the case of the MIT data set and UMassDieselNet the difference in performance of multi-copy Parasitic Routing and PROPHET is less significant. However, in both data sets the message delivery achieved by the single-copy strategies is already close to that of the theoretical maximum represented by Epidemic Routing.

When bandwidth is limited all protocol showed limited degradation in performance at low transmission speeds when the message size is small. However, in the MIT dataset single-copy Parasitic Routing outperforms Epidemic routing for transmission speed of less than 100 Kbps. All protocols show a reduction in performance when bandwidth is limited and the messages sizes approach 100K. Single-copy Parasitic Routing is less affected in general, this is a result of reduced control data overhead. As a result, the decision to use replication should take into account factors such as the expected encounter duration and message size.

Under the constraint of limited node storage space, Parasitic Routing outperforms Epidemic Routing at low capacities. Single-copy Parasitic Routing and PROPHET are the least affect, which is expected of single-copy strategies as the lack of redundancy reduces contention for storage resources. Multi-copy Parasitic Routing and PROPHET are more affected than the single-copy strategies but results in less contention than Epidemic Routing.

# Chapter 8

## Conclusion

The research presented in this thesis has explored the use of social network analysis techniques for routing in disconnected delay-tolerant MANETs. More specifically, it has focused on evaluating the utility of a node based on social analysis of the node's past interactions. This chapter summarises the achievements of the work and reviews related research issues that remain open for future work.

### 8.1 Achievements

In chapter 1, it was noted that the motivation for the work presented arose from the observation that while infrastructure-based networks provide an effective mechanism for communication between mobile devices, there are situations where the required networking connections are not available in a given geographic area, and providing the needed connectivity and network services in these situations becomes a real challenge. On this basis, chapter 2 mapped out the challenges associated with routing in disconnected mobile networks without the existence of such infrastructure. These challenges extended those identified by Forman et al. [40] related to communication in the wireless environment and the consequences of node mobility, along with the pressures that the portability of the devices places on the network.

Chapter 3 reviewed current research in the area of routing in DDTMs. From this examination, it was determined that metric-based stochastic solutions are well suited to dynamic networks where topology information is limited. Metric-based schemes evaluate the utility of a node to provide message delivery and these metrics are typically based on either direct or

indirect observed encounters. Though promising, the utility of a node is typically limited to the horizon of their encounters, and the encounters of those they have encountered.

Based on this observation, chapter 4 reviewed social analysis techniques in the context of information flow. The use of social network analysis was motivated by the fact that information in regards to underlying social structure is less dynamic and time dependent than information about temporal connections. Using this social structure, nodes that are useful in connecting otherwise disconnected nodes can be identified. Consequently, if no information is available regarding the destination node, then central nodes can be used to route information to nodes that have a higher probability of encountering information about the destination node. Additionally, the social structure can be used to identify nodes that belong to similar groups as the destination node. Finally, social network analysis techniques can be used to evaluate social tie strength, which represents the strength of a connection between two nodes. In particular, strong ties have a higher probability of being available for information dissemination.

The Parasitic Routing protocol described in chapter 5 leverages the social metrics identified in chapter 4 by translating them to the area of DDTMs. To the best of our knowledge, it is the first time such metrics have been applied to the context of DDTMs. The implementation of the Parasitic Routing protocol as described in chapter 6 was evaluated in chapter 7 using real world trace data. The simulation results showed that, in nearly all scenarios, Parasitic Routing achieves delivery performance close to the theoretical maximum achieved by Epidemic Routing and that performance is achieved with significantly less overhead than Epidemic Routing. Additionally, when storage capacity is limited on nodes Parasitic Routing outperforms Epidemic Routing. The evaluation also demonstrates that Parasitic Routing achieves superior delivery performance compared to the most influential of the related protocols documented in the literature.

The achievements can be summarised as two primary and one secondary contributions to the state of the art in the area of DDTM routing:

- A set of metrics for capturing the underlying time-varying network structure in disconnected delay-tolerant MANETs where global topology information is unavailable. These metrics are derived from a translation of social network analysis techniques for capturing the behaviour of nodes.

- A protocol, called Parasitic Routing, that supports message delivery in DDTMs by utilising these metrics to provide a complete solution for delay-tolerant message delivery in disconnected networks. An implementation and evaluation validates the Parasitic Routing protocol using real world trace data capturing encounters between devices.
- A secondary contribution is an analysis of the challenges associated with the specific mobile computing environment constituted by disconnected delay-tolerant MANETs.

## 8.2 Future Work

As is so often the case with research, there are some issues that remain open for possible future work. This section identifies four such areas.

The implementation presented in chapter 6 and evaluated in chapter 7 does not utilise all of the tie strength indicators discussed in chapter 4. The multiple social context indicator measures the range of locations two nodes have encountered each other. The premise being that a strong tie is most likely encountered in multiple geographic contexts. Evaluation of this indicator was not possible due to the lack of location information in the trace data presented in chapter 7. Location information could be used to augment the forwarding decisions. For example if node  $a$  frequently meets node  $b$  at location  $x$ , the probability the node  $a$  is expected to visit location  $x$  could infer a higher probability of encountering node  $b$  than if node  $a$  has a low probability of visiting location  $x$  in the near future.

Additionally, the concept of trust has not been addressed, however, social network analysis techniques lend themselves well to the area of trust [103]. Shi et al. propose that when transmitting covert or sensitive information all social ties may not be used and the route may be restricted to strong ties. Consequently, depending on the level of sensitivity of a message, a threshold may be placed on where the message is only forwarded to nodes with which the current node has a strong tie strength.

The current implementation uses replication as long as the replication value is above 1. The potential exists to refine this replication decision based on identifying messages that have a lower delivery probability. Analysis would need to be performed to identify the circumstances under which message were delivered using replication, that remained undelivered using



a single-copy strategy. It could be that message originating from nodes with no direct relationship with the destination node require a higher replication value than those that originate on nodes that have a strong link to the destination node.

The evaluation scenarios in chapter 7 consist of encounters between people, and the encounters between vehicles. An additional application area of interest is that of wildlife tracking. Lusseau and Newman applied social analysis techniques developed for human analysis to a group of bottlenose dolphins [84]. Their research showed that dolphin social structure exhibits communities and preferential companionship. As a consequence, Parasitic Routing could potentially be applied to wildlife tracking.

### **8.3 Concluding Remarks**

The work in this thesis can be seen as part of an interesting trend of multidisciplinary research where the translation of ideas between domains is becoming more frequent. The problem of information flow in social networks, which was used here, has also been explored in areas ranging from market research identifying influential social ties in word-of-mouth referral networks [18] to identifying whether a startup business will succeed or fail based on the social resources available [116]. The concept of ‘small-worlds’ is a source of a vast amount of research, given the desirable structural properties such as exhibiting short path lengths. However, this does not guarantee that short paths will be found. One issue that needs to be considered is that the participants in Milgram’s experiment did not randomly select the next recipient of the message, but used locally available, locally evaluated criteria in the decision process. In order to harness the power of ‘small-world’ graphs to their full extent deeper analysis of the decision process is necessary. The research presented here makes a concerted effort to measure properties that may be used in the decision process in a localised autonomous manner with promising results. The lack of global knowledge is characteristic of many existing distributed systems. While this thesis has applied these properties to the specific area of DDTMs, the results presented here may have wider applicability.

# Appendix A

## PRoPHET Parameter Tuning

The default PRoPHET parameters as recommended in Lindgren et al. [80] were used. However, one parameter that required some tuning is the time elapsed unit  $\kappa$  used to age the contact probabilities. The appropriate time unit used differs depending on the application and the expected delays in the network. Therefore, each baseline simulation was run once using various time units for one random seed. The results of these simulations are summarised below and the optimum value highlighted.

Aging Unit	Message Delivery	Ave Hops	Ave Delay	Total Forwards
1 minute	2253	4.57	63216.63	57241
10 minutes	2544	3.91	73231.52	39880
30 minutes	2504	3.65	81186.73	38311
<b>1 hour</b>	<b>2564</b>	<b>3.33</b>	<b>38311</b>	<b>33814</b>

**Table 8.1:** Intel Baseline PRoPHET Tuning

Aging Unit	Message Delivery	Ave Hops	Ave Delay	Total Forwards
1 minute	2370	4.60	62034.55	88796
<b>10 minutes</b>	<b>2419</b>	<b>3.45</b>	<b>62719.50</b>	<b>74586</b>
30 minutes	2153	2.42	60086.45	51268
1 hour	2386	2.49	62609.76	28445

**Table 8.2:** Cambridge Baseline PRoPHET Tuning

Aging Unit	Message Delivery	Ave Hops	Ave Delay	Total Forwards
1 minute	7978	8.18	42934.20	441003
10 minutes	8723	5.80	44316.04	279687
30 minutes	8362	5.46	45211.23	247432
<b>1 hour</b>	<b>8921</b>	<b>5.39</b>	<b>48963.78</b>	<b>249180</b>

**Table 8.3:** InfoCom Baseline PROPHET Tuning

Aging Unit	Message Delivery	Ave Hops	Ave Delay	Total Forwards
1 minute	133315	31.88	1.87E9	15898329
10 minutes	145163	38.12	2.61E9	16950678
<b>30 minutes</b>	<b>165842</b>	<b>28.22</b>	<b>2.17E9</b>	<b>13637660</b>
1 hour	161649	24.52	2.04E9	12854060

**Table 8.4:** MIT Baseline PROPHET Tuning

Aging Unit	Message Delivery	Ave Hops	Ave Delay	Total Forwards
1 minute	120334	8.77	4.95 E8	2840271
10 minutes	120479	10.28	6.16 E8	3047256
<b>30 minutes</b>	<b>120735</b>	<b>8.42</b>	<b>5.12 E8</b>	<b>2916220</b>
1 hour	120381	7.33	5.29 E8	2593729

**Table 8.5:** DeiselNet Baseline PROPHET Tuning



## Bibliography

- [1] A community resource for archiving wireless data at dartmouth: <http://crawdad.cs.dartmouth.edu/>. viewed on april 29th 2007.
- [2] Delay tolerant networking research group: <http://www.dtnrg.org/>. viewed on april 29th 2007.
- [3] Disruption tolerant networks program: <http://www.darpa.mil/ato/solicit/dtn/>. viewed on april 29th 2007.
- [4] MIT media lab: Reality mining <http://reality.media.mit.edu/>. viewed on april 29th 2007.
- [5] L. A. Adamic and E. Adar. Friends and neighbors on the web. *Social Networks*, 25(3):211–230, July 2003.
- [6] S. Agarwal, A. Ahija, J. P. Singh, and R. Shorey. Routelifetime assessment based routing (RABR) protocol for mobile ad-hoc networks. In *Proceedings of the IEEE International Conference on Communications, 2000. ICC 2000*, volume 3, pages 1697–1701. IEEE Press, June 2000.
- [7] R. Ahlswede, Ning Cai, S. Y. R. Li, and R. W. Yeung. Network information flow. *IEEE Transactions on Information Theory*, 46(4):1204–1216, July 2000.
- [8] H. Balakrishnan and V. N. Padmanabhan. How network asymmetry affects TCP. *IEEE Communications Magazine*, 39(4):60–67, April 2001.
- [9] Chadi Barakat and Eitan Altman. On ack filtering on a slow reverse channel. *International Journal of Satellite Communications and Networking*, 21(3):80–92, April 2003.

- [10] Allan Beaufour, Martin Leopold, and Philippe Bonnet. Smart-tag based data dissemination. In *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, pages 68–77, New York, NY, USA, September 2002. ACM Press.
- [11] Mario Benassi, Arent Greve, and Julia Harkola. Looking for a network organization: The case of GESTO. *Journal of Market-Focused Management*, 4(3):205–229, October 1999.
- [12] Philip Blumstein and Peter Kollock. Personal relationships. *Annual Review of Sociology*, 14:467–490, 1988.
- [13] S. P. Borgatti, M. G. Everett, and L. C. Freeman. UCINET 6 for Windows: Software for social network analysis, 2002.
- [14] Stephen P. Borgatti. Centrality and network flow. *Social Networks*, 27(1):55–71, January 2005.
- [15] Vincent Borrel, Franck Legendre, Marcelo, and Serge Fdida. SIMPS: Using sociology for personal mobility, 2006.
- [16] L. C. Briand, S. Morasca, and V. R. Basili. Property-based software engineering measurement. *Software Engineering, IEEE Transactions on*, 22(1):68–86, 1996.
- [17] Josh Broch, David A. Maltz, David B. Johnson, Yih C. Hu, and Jorjeta Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the 4th annual ACM/IEEE international conference on Mobile computing and Networking*, pages 85–97, New York, NY, USA, October 1998. ACM Press.
- [18] Jacqueline J. Brown and Peter H. Reingen. Social ties and word-of-mouth referral behavior. *The Journal of Consumer Research*, 14(3):350–362, December 1987.
- [19] Marcus Brunner, Lars Eggert, Kevin Fall, Jörg Ott, and Lars Wolf. Dagstuhl seminar on disruption tolerant networking. *ACM SIGCOMM Computer Communication Review*, 35(3):69–72, July 2005.

- [20] John Burgess, Brian Gallagher, David Jensen, and B. N. Levine. Maxprop: Routing for vehicle-based disruption-tolerant networking. In *Proceedings of the IEEE INFOCOM 2005. 24th Annual Joint Conference of the IEEE Computer and Communications Societies*. IEEE Press, March 2006.
- [21] 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, September 2002.
- [22] V. Cerf, S. Burleigh, A. Hooke, L. Torgerson, R. Durst, K. Scott, K. Fall, and H. Weiss. Delay-tolerant network architecture. internet draft, September 2006.
- [23] Vinton G. Cerf, Scott C. Burleigh, Adrian J. Hooke, Leigh Torgerson, Robert C. Durst, Keith L. Scott, Kevin Fall, Eric J. Travis, and Howard S. Weiss. Delay-tolerant network architecture: The evolving interplanetary internet, April 2002.
- [24] Augustin Chaintreau, Pan Hui, Jon Crowcroft, Christophe Diot, Richard Gass, and James Scott. Impact of human mobility on the design of opportunistic forwarding algorithms. In *Proceedings of IEEE INFOCOM 2006*, 2006.
- [25] R. Chellapa, A. Jennings, and N. Shenoy. A comparative study of mobility prediction in fixed wireless networks and mobile ad hoc networks. In *Proceedings of the IEEE International Conference on Communications, 2003. ICC '03*, volume 2, pages 891–895. IEEE Press, May 2003.
- [26] Xiangchuan Chen and Amy L. Murphy. Enabling disconnected transitive communication in mobile ad hoc networks. In *Proceedings of the Workshop on Principles of Mobile Computing (POMC), co-located with PODC*, pages 21–27, August 2001.
- [27] Imrich Chlamtac, Marco Conti, and Jennefer. Mobile ad hoc networking: imperatives and challenges. *Ad Hoc Networks*, 1(1):13–64, July 2003.
- [28] Tanzeem Choudhury and Alex Pentland. The sociometer: A wearable device for understanding human networks. In *Computer Supported Cooperative Work - Workshop*

- on Ad hoc Communications and Collaboration in Ubiquitous Computing Environments*, November 2002.
- [29] S. Corson and J. Macker. Mobile ad hoc networking (MANET): Routing protocol performance issues and evaluation considerations, rfc 2501, January 1999.
- [30] Elizabeth M. Daly and Mads Haahr. Social network analysis for routing in disconnected delay-tolerant MANETs. In *Proceedings of the 8th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2007, Montréal, Québec, Canada, September 9-14, 2007*, September 2007.
- [31] J. A. Davis, A. H. Fagg, and B. N. Levine. Wearable computers as packet transport mechanisms in highly-partitioned ad-hoc networks. In *Proceedings of the Fifth International Symposium on Wearable Computers, 2001*, pages 141–148. IEEE Press, October 2001.
- [32] Alan Demers, Dan Greene, Carl Hauser, Wes Irish, John Larson, Scott Shenker, Howard Sturgis, Dan Swinehart, and Doug Terry. Epidemic algorithms for replicated database maintenance. In *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, pages 1–12, New York, NY, USA, August 1987. ACM Press.
- [33] Nathan Eagle and Alex Pentland. Reality mining: sensing complex social systems. *Personal and Ubiquitous Computing*, 10(4):255–268, May 2005.
- [34] Martin Everett and Stephen P. Borgatti. Ego network betweenness. *Social Networks*, 27(1):31–38, January 2005.
- [35] Kevin Fall. Delay-tolerant networking for extreme environments, November 2001.
- [36] Kevin Fall. A delay-tolerant network architecture for challenged internets. In *Proceedings of the 2003 conference on Applications, technologies, architectures, and protocols for computer communications*, pages 27–34. ACM Press, August 2003.
- [37] Kevin Fall. A delay-tolerant network architecture for challenged internets. Technical report, Intel Research, Berkeley, 2003.



- [38] Kevin Fall. Messaging in difficult environments. Technical report, Intel Research Berkeley, December 2004.
- [39] L. R. Ford and D. R. Fulkerson. *Flows in Networks (Rand Corporation Research Studies Series)*. Princeton University Press, NJ, June 1962.
- [40] George H. Forman and John Zahorjan. The challenges of mobile computing. *Computer*, 27(4):38–47, April 1994.
- [41] Linton C. Freeman. A set of measures of centrality based on betweenness. *Sociometry*, 40(1):35–41, March 1977.
- [42] Linton C. Freeman. Centrality in social networks conceptual clarification. *Social networks*, 1(3):215–239, 1978-1979.
- [43] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. *Design Patterns*. Addison-Wesley Professional, January 1995.
- [44] P. Gevros, J. Crowcroft, P. Kirstein, and S. Bhatti. Congestion control mechanisms and the best effort service model. *IEEE Network*, 15(3):16–26, May 2001.
- [45] Joy Ghosh, Matthew J. Beal, Hung Q. Ngo, and Chunming Qiao. On profiling mobility and predicting locations of wireless users. In *REALMAN '06: Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 55–62, New York, NY, USA, 2006. ACM Press.
- [46] Joy Ghosh, Hung Q. Ngo, and Chunming Qiao. Mobility profile based routing within intermittently connected mobile ad hoc networks (ICMAN). In *IWCMC '06: Proceeding of the 2006 international conference on Communications and mobile computing*, pages 551–556, New York, NY, USA, 2006. ACM Press.
- [47] Natalie Glance, Dave Snowdon, and Jean-Luc Meunier. Pollen: using people as a communication medium. *Computer Networks*, 35(4):429–442, March 2001.
- [48] A. J. Goldsmith and S. B. Wicker. Design challenges for energy-constrained ad hoc wireless networks. *IEEE Wireless Communications*, [see also *IEEE Personal Communications*], 9(4):8–27, August 2002.

- [49] Mark S. Granovetter. The strength of weak ties. *The American Journal of Sociology*, 78(6):1360–1380, May 1973.
- [50] Elizabeth L. Gray. *A Trust-Based Reputation Management System*. PhD thesis, April 2006.
- [51] Matthias Grossglauser and David N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Trans. Netw.*, 10(4):477–486, August 2002.
- [52] Geir Gundersen and Trond Steihaug. Data structures in java for matrix computations. *Concurrency and Computation: Practice and Experience*, 16(8):799–815, 2004.
- [53] Radu Handorean, Christopher Gill, and Gruia-Catalin Roman. Accommodating transient connectivity in ad hoc and mobile settings. In Alois Ferscha and Friedemann Mattern, editors, *Proceedings of the Second International Conference, Pervasives 2004, Linz/Vienna, Austria, April 21-23, 2004*, volume 3001 of *Lecture Notes in Computer Science*, pages 305–322. Springer-Verlag, March 2004.
- [54] Herbert W. Hethcote and P. Driessche. An SIS epidemic model with variable population size and a delay. *Journal of Mathematical Biology*, 34(2):177–194, January 1995.
- [55] W. Hsu and A. Helmy. On nodal encounter patterns in wireless LAN traces. In *Proceedings of the 4th International Symposium on Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks, 2006*, pages 1–10. IEEE Press, April 2006.
- [56] Wei-Jen Hsu and Ahmed Helmy. Impact: Investigation of mobile-user patterns across university campuses using wlan trace analysis, August 2005.
- [57] P. Hui, A. Chaintreau, R. Gass, J. Scott, J. Crowcroft, and C. Diot. Pocket switched networking: Challenges, feasibility, and implementation issues. In *Proceedings of the Workshop on Autonomic Communications*, 2005.
- [58] Pan Hui, Augustin Chaintreau, James Scott, Richard Gass, Jon Crowcroft, and Christophe Diot. Pocket switched networks and human mobility in conference environments. In *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 244–251, New York, NY, USA, August 2005. ACM Press.

- [59] V. Jacobson. Modified TCP congestion avoidance algorithm. *end2end-interest mailing list*, April 1990.
- [60] V. Jacobson. Rfc1144: Compressing TCP/IP headers for low-speed serial links, February 1990.
- [61] Sushant Jain, Michael Demmer, Rabin Patra, and Kevin Fall. Using redundancy to cope with failures in a delay tolerant network. In *Proceedings of the 2005 conference on Applications, technologies, architectures, and protocols for computer communications*, volume 35, pages 109–120, New York, NY, USA, August 2005. ACM Press.
- [62] Sushant Jain, Kevin Fall, and Rabin Patra. Routing in a delay tolerant network. *SIGCOMM Comput. Commun. Rev.*, 34(4):145–158, October 2004.
- [63] Per Johansson, Tony Larsson, Nicklas Hedman, Bartosz Mielczarek, and Mikael Degermark. Scenario-based performance analysis of routing protocols for mobile ad-hoc networks. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 195–206, New York, NY, USA, August 1999. ACM Press.
- [64] David B. Johnson and David A. Maltz. *Dynamic source routing in ad-hoc wireless networks*, volume 353 of *The Springer International Series in Engineering and Computer Science*, pages 153–181. Kluwer Academic Publishers, 1996.
- [65] Evan P. C. Jones, Lily Li, Jakub K. Schmidtke, and Paul A. S. Ward. Practical routing in delay-tolerant networks. *IEEE Transactions On Mobile Computing*.
- [66] Evan P. C. Jones, Lily Li, and Paul A. S. Ward. Practical routing in delay-tolerant networks. In *Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 237–243, New York, NY, USA, August 2005. ACM Press.
- [67] Philo Juang, Hidekazu Oki, Yong Wang, Margaret Martonosi, Li S. Peh, and Daniel Rubenstein. Energy-efficient computing for wildlife tracking: design tradeoffs and early experiences with ZebraNet. In *Proceedings of the 10th international conference on Architectural support for programming languages and operating systems*, volume 37, pages 96–107, New York, NY, USA, October 2002. ACM Press.

- [68] A. Khelil, P. J. Marron, and K. Rothermel. Contact-based mobility metrics for delay-tolerant ad hoc networking. pages 435–444, 2005.
- [69] Young-Bae Ko and Nitin H. Vaidya. Location-aided routing (LAR) in mobile ad hoc networks. *Wireless Networks*, 6(4):307–321, September 2000.
- [70] J. Lebrun, Chen-Nee Chuah, D. Ghosal, and Michael Zhang. Knowledge-based opportunistic forwarding in vehicular wireless ad hoc networks. In *Proceedings of the IEEE 61st Conference on Vehicular Technology, 2005. VTC 2005-Spring*, volume 4, pages 2289–2293. IEEE Press, May 2005.
- [71] Jérémie Leguay, Timur Friedman, and Vania Conan. DTN routing in a mobility pattern space. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 276–283, New York, NY, USA, August 2005. ACM Press.
- [72] Jérémie Leguay, Timur Friedman, and Vania Conan. Evaluating mobility pattern space routing for DTNs. In *INFOCOM 2005. Proceedings of the 24th IEEE Annual Joint Conference of the IEEE Computer and Communications Societies*. IEEE Press, April 2006.
- [73] W. Lehr and L. Mcknight. Wireless internet access: 3G vs. WIFI ? *Telecommunication Policy*, 27:351–370, 2002.
- [74] Qun Li and Daniela Rus. Sending messages to mobile users in disconnected ad-hoc wireless networks. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 44–55, New York, NY, USA, August 2000. ACM Press.
- [75] Yong Liao, Kun Tan, Zhensheng Zhang, and Lixin Gao. Estimation based erasure-coding routing in delay tolerant networks. In *IWCMC '06: Proceeding of the 2006 international conference on Communications and mobile computing*, pages 557–562, New York, NY, USA, July 2006. ACM Press.
- [76] David Liben-Nowell and Jon Kleinberg. The link prediction problem for social networks. In *CIKM '03: Proceedings of the twelfth international conference on Infor-*

- mation and knowledge management*, pages 556–559, New York, NY, USA, November 2003. ACM Press.
- [77] Nan Lin, Paul Dayton, and Peter Reenwald. Analyzing the instrumental use of relations in the context of social structure. *Sociological Methods and Research*, 7(2):149–166, 1978.
- [78] Nan Lin, John C. Vaughn, and Walter M. Ensel. Social resources and occupational status attainment. *Social Forces*, 59(4):1163–1181, June 1981.
- [79] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. *ACM SIGMOBILE Mobile Computing and Communications Review*, 7(3):19–20, July 2003.
- [80] Anders Lindgren, Avri Doria, and Olov Schelén. Probabilistic routing in intermittently connected networks. In *Proceedings of the First International Workshop, SAPIR 2004, Fortaleza, Brazil, August 1-6, 2004*, volume 3126 of *Lecture Notes in Computer Science*, pages 239–254. Springer-Verlag GmbH, August 2004.
- [81] Anders Lindgren and Kaustubh. Evaluation of queueing policies and forwarding strategies for routing in intermittently connected networks. In *Proceedings of the The First International Conference on COMMunication System softWARE and MiddlewaRE (COM-SWARE 2006)*, January 2006.
- [82] G. Y. Liu and G. Q. Maguire. A predictive mobility management algorithm for wireless mobile computing and communications. In *Proceedings of ICUPC '95 - 4th IEEE International Conference on Universal Personal Communications*, pages 268–272. IEEE Press, November 1995.
- [83] J. Liu and S. Singh. ATCP: TCP for mobile ad hoc networks. *IEEE Journal on Selected Areas in Communications*, 19(7):1300–1315, July 2001.
- [84] David Lusseau and M. E. J. Newman. Identifying the role that individual animals play in their social network, Mar 2004.

- [85] P. V. Marsden. Egocentric and sociocentric measures of network centrality. *Social networks*, 24(4):407–422, October 2002.
- [86] Peter V. Marsden and Karen E. Campbell. Measuring tie strength. *Social Forces*, 63(2):482–501, December 1984.
- [87] Cecilia Mascolo and Mirco Musolesi. SCAR: Context-aware adaptive routing in delay tolerant mobile sensor networks. In *IWCMC '06: Proceeding of the 2006 international conference on Communications and mobile computing*, pages 533–538, New York, NY, USA, July 2006. ACM Press.
- [88] A. B. McDonald and T. F. Znati. A mobility-based framework for adaptive clustering in wireless adhoc networks. *IEEE Journal on Selected Areas in Communications*, 17(8):1466–1487, August 1999.
- [89] Shashidhar Merugu, Mostafa Ammar, and Ellen Zegura. Routing in space and time in networks with predictable mobility. Technical report, July 2004.
- [90] Stanley Milgram. The small world problem. *Psychology Today*, 2:60–67, May 1967.
- [91] M. Mitzenmacher. Digital fountains: a survey and look forward. In *Proceedings of the IEEE Information Theory Workshop, 2004*, pages 271–276. IEEE Press, October 2004.
- [92] Mirco Musolesi and Cecilia Mascolo. A community based mobility model for ad hoc network research. In *REALMAN '06: Proceedings of the second international workshop on Multi-hop ad hoc networks: from theory to reality*, pages 31–38, New York, NY, USA, 2006. ACM Press.
- [93] Delphine Nain, Noshirwan Petigara, and Hari Balakrishnan. Integrated routing and storage for messaging applications in mobile ad hoc networks. *Mobile Networks and Applications*, 9(6):595–604, December 2004.
- [94] M. E. J. Newman. Clustering and preferential attachment in growing networks. *Physical Review E*, 64(2):025102+, July 2001.
- [95] M. E. J. Newman. A measure of betweenness centrality based on random walks. *Social networks*, 27(1):39–54, January 2005.

- [96] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. In *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, pages 151–162, New York, NY, USA, August 1999. ACM Press.
- [97] A. Pentland, R. Fletcher, and A. Hasson. Daknet: rethinking connectivity in developing nations. *Computer*, 37(1):78–83, January 2004.
- [98] C. E. Perkins and E. M. Royer. Ad-hoc on-demand distance vector routing. In *WMCSA '99: Proceedings of the Second IEEE Workshop on Mobile Computing Systems and Applications*, pages 90–100. IEEE Press, February 1999.
- [99] Charles E. Perkins and Pravin Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *SIGCOMM '94: Proceedings of the conference on Communications architectures, protocols and applications*, volume 24, pages 234–244. ACM Press, October 1994.
- [100] S. Ramanathan and Martha Steenstrup. A survey of routing techniques for mobile communications networks. *Mobile Networks and Applications*, 1(2):98–104, October 1996.
- [101] Arthur J. Riel. *Object-Oriented Design Heuristics*. Addison-Wesley, 1st edition, April 1996.
- [102] R. C. Shah, S. Roy, S. Jain, and W. Brunette. Data mules: modeling a three-tier architecture for sparse sensor networks. In *Proceedings of the First IEEE International Workshop on Sensor Network Protocols and Applications, 2003*, pages 30–41. IEEE Press, May 2003.
- [103] Xiaolin Shi, Lada A. Adamic, and Martin J. Strauss. Networks of strong ties. *Physica A: Statistical Mechanics and its Applications*, 378(1):33–47, May 2007.
- [104] F. S. Smailbegovic, G. N. Gaydadjiev, and S. Vassiliadis. Sparse matrix storage format. In *Proceedings of the 16th Annual Workshop on Circuits, Systems and Signal Processing, ProRisc 2005*, pages 445–448, November 2005.

- [105] Tara Small and Zygmunt J. Haas. The shared wireless infostation model: a new ad hoc networking paradigm (or where there is a whale, there is a way). In *MobiHoc '03: Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, pages 233–244, New York, NY, USA, June 2003. ACM Press.
- [106] T. Spyropoulos, K. Psounis, and C. S. Raghavendra. Single-copy routing in intermittently connected mobile networks. In *Proceedings of the First Annual IEEE Communications Society Conference on Sensor and Ad Hoc Communications and Networks, 2004. IEEE SECON 2004*, pages 235–244. IEEE Press, October 2004.
- [107] Thrasyvoulos Spyropoulos, Konstantinos Psounis, and Cauligi S. Raghavendra. Spray and wait: an efficient routing scheme for intermittently connected mobile networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 252–259, New York, NY, USA, August 2005. ACM Press.
- [108] T. Starmer. Thick clients for personal wireless devices. *Computer*, 35(1):133–135, January 2002.
- [109] James P. G. Sterbenz, Rajesh Krishnan, Regina R. Hain, Alden W. Jackson, David Levin, Ram Ramanathan, and John Zao. Survivable mobile wireless networks: issues, challenges, and research directions. In *WiSE '02: Proceedings of the 3rd ACM workshop on Wireless security*, pages 31–40, New York, NY, USA, September 2002. ACM Press.
- [110] William Su, Sung-Ju Lee, and Mario Gerla. Mobility prediction and routing in ad hoc wireless networks. *International Journal of Network Management*, 11(1):3–30, January 2001.
- [111] Kun Tan, Qian Zhang, and Wenwu Zhu. Shortest path routing in partially connected ad hoc networks. In *Proceedings of the IEEE Global Telecommunications Conference, 2003. GLOBECOM '03*, volume 2, pages 1038–1042. IEEE Press, December 2003.
- [112] Diane Tang and Mary Baker. Analysis of a local-area wireless network. In *MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 1–10, New York, NY, USA, August 2000. ACM Press.



- [113] C. Toh. Maximum battery life routing to support ubiquitous mobile computing in wireless ad hoc networks. *IEEE Communications Magazine*, 39(6):138–147, June 2001.
- [114] J. Tsumochi, K. Masayama, H. Uehara, and M. Yokoyama. Impact of mobility metric on routing protocols for mobile ad hoc networks. In *Proceedings of the IEEE Pacific Rim Conference on Communications, Computers and signal Processing, 2003. PACRIM. 2003*, volume 1, pages 322–325. IEEE Press, August 2003.
- [115] A. Vahdat and D. Becker. Epidemic routing for partially connected ad hoc networks. Technical report, April 2000.
- [116] Gordon Walker, Bruce Kogut, and Weijian Shan. Social capital, structural holes and the formation of an industry network. *Organization Science*, 8(2):109–125, 1997.
- [117] Yong Wang, Sushant Jain, Margaret Martonosi, and Kevin Fall. Erasure-coding based routing for opportunistic networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 229–236, New York, NY, USA, August 2005. ACM Press.
- [118] D. J. Watts and S. H. Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, June 1998.
- [119] Steve Whittaker, Quentin Jones, and Loren Terveen. Contact management: identifying contacts to support long-term communication. In *CSCW '02: Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 216–225, New York, NY, USA, November 2002. ACM Press.
- [120] Jörg Widmer. Network coding for efficient communication in wireless networks, November 2005.
- [121] Jörg Widmer and Jean-Yves Le Boudec. Network coding for efficient communication in extreme networks. In *WDTN '05: Proceeding of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, pages 284–291, New York, NY, USA, August 2005. ACM Press.

- [122] Gang Yan, Tao Zhou, Bo Hu, Zhong Q. Fu, and Bing H. Wang. Efficient routing on complex networks. *Physical Review E (Statistical, Nonlinear, and Soft Matter Physics)*, 73(4), April 2006.
- [123] Jun Zhang and Mark S. Ackerman. Searching for expertise in social networks: a simulation of potential strategies. In *GROUP '05: Proceedings of the 2005 international ACM SIGGROUP conference on Supporting group work*, pages 71–80, New York, NY, USA, November 2005. ACM Press.
- [124] Lixia Zhang, Scott Shenker, and David D. Clark. Observations on the dynamics of a congestion control algorithm: the effects of two-way traffic. *ACM SIGCOMM Computer Communication Review*, 21(4):133–147, September 1991.
- [125] Z. Zhang. Routing in intermittently connected mobile ad hoc networks and delay tolerant networks: Overview and challenges. *IEEE Communications Surveys & Tutorials*, 8(1):24–37, 2006.
- [126] W. Zhao, M. Ammar, and E. Zegura. A message ferrying approach for data delivery in sparse mobile ad hoc networks. In *MobiHoc '04: Proceedings of the 5th ACM international symposium on Mobile ad hoc networking and computing*, pages 187–198, New York, NY, USA, May 2004. ACM Press.
- [127] W. Zhao, M. Ammar, and E. Zegura. Controlling the mobility of multiple data transport ferries in a delay-tolerant network. In *INFOCOM 2005. Proceedings of the 24th Annual Joint Conference of the IEEE Computer and Communications Societies*, volume 2, pages 1407–1418. IEEE Press, March 2005.