A HISTORY-BASED ENERGY-EFFICIENT ROUTING PROTOCOL FOR OPPORTUNISTIC NETWORKS

by

Khuram Khalid

B.Sc in Computer Science, Memorial University of Newfoundland, Canada, 2007

A Thesis

presented to Ryerson University

in partial fulfillment of the

requirements for the degree of

Master of Science

in the Program of

Computer Science

Toronto, Ontario, Canada, 2016

Author's Declaration

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

A HISTORY-BASED ENERGY-EFFICIENT ROUTING PROTOCOL FOR OPPORTUNISTIC NETWORKS

Master of Science
in Computer Science
Ryerson University

Abstract

In this thesis, a history-based energy-efficient routing protocol (called AEHBPR) for opportunistic networks (OppNets) is proposed, which saves the energy consumption by avoiding unnecessary packets transmission in the network and by clearing the buffer of nodes carrying the copies of the already delivered packets. The proposed AEHBPR protocol is evaluated using the Opportunistic NEtwork (ONE) simulator with both synthetic and real mobility traces, showing a superior performance compared to the History-Based Prediction for Routing (HBPR) protocol and AEProphet, in terms of average remaining energy, number of dead nodes, number of delivered messages, and overhead ratio, where AEProphet is the ProPHet routing protocol for OppNets on which the same energy-aware mechanism has been implemented.

Acknowledgments

Foremost, I would like to express my immeasurable appreciation and deepest gratitude to my supervisor Dr. Isaac Woungang, for his continuous support throughout my graduate studies. His invaluable guidance helped me throughout my research work, and for the completion of this thesis. I would also like to thank the DABNEL lab team, in particular, Dr. Glaucio Carvalho for being there when I needed help. My appreciation also goes to Dr. Sanjay K. Dhurandher for his fruitful advice throughout my studies, and to the Department of Computer Science at Ryerson University for providing me the useful resources toward the accomplishment of my degree.

I wish to express my love, and sincerity to my Mother for her continuous encouragement, support, and prayers throughout my life. I wish to thank my wife Narjis, for her continuous support and motivation, without whom this goal would have been hard to accomplish. I wish to thank my brother, sister, and friends for their continuous encouragement throughout my studies.

I cannot forget to express my love to my three-year-old daughter Eba, whose smiles and hugs have always been a source of stress relief after lengthy and exhaustive hours of study.

Contents

List of Figures

List of Tables

List of Abbreviations

HBPR	History Based Prediction Routing Protcol
AEHBPR	Energy-efficient HBPR
CMM	Custom Mobility Model
DTN	Delay Tolerant Networks
ONE	Opportunistic Network Environment
OppNets	Opportunistic Networks
Ack_M	Acknowledgment Message
Ack_Table	Acknowledgment Table
ProPHet	Probabilistic Routing Protocol using History of Encounters and Transitivity
AEProphet	Energy-efficient ProPHet

Chapter 1

Introduction

1.1 Motivation and Research Problem

In OppNets, the communication routes between the nodes are built dynamically on the fly, in a store-carry-and-forward fashion. Due to the highly mobile nature of the nodes and the uncertain mobility patterns, the disconnection of links among the nodes occurs frequently, yielding the so-called intermittent connectivity. The basic characteristics of OppNets include limited battery power, broken links, limited storage, to name a few [1]. Due of these facts, a packet delivery from source to destination is not necessary guaranteed. However, the applications of OppNets are diverse as they can be deployed in many real life situations such as disaster relief, military operations, wildlife monitoring, health care monitoring, to name a few [1], [2].

In OppNets, since an end-to-end path between the source and destination nodes may seldom exist, routing is a challenging task. Due to its highly mobile nature, a relay node can only be expected to opportunistically establish a connection with another node within a short period of time, then use it to pass along the message, hoping that this recipient node will do the same with another encountered node, until the message is eventually deliver to the desired destination. Packets drop is very frequent due to node's power failure, short range wireless communication, and node's mobility. One way to avoid packets drop is to let a node keep the packets in its buffer for a longer period of time, until a suitable next hop is found to forward them. In addition, Storing the packets at a node increases the probability of successfully delivering the message, but at the expense of delaying the delivery time [1], [3].

In OppNets, the energy consumption of nodes is a critical issue which occur frequently when a node sends and receive a packet or when one packet is sent many times by a node to multiple nodes [4]. In order to save the node's battery life, and thereby increase the chance of successful packets delivery, it is desirable to (1) avoid unnecessary transmissions of packets which have already been delivered to the destination, and (2) to route the packets only through those nodes that have enough remaining battery power, in such a way that the delivery ratio of the packets is maximized.

In this thesis, following those objectives, the recently proposed HBPR protocol for OppNets [3] is redesigned by incorporating some energy-related constraints, yielding the so-called energy-efficient HBPR (AEHBPR). The performance of AEHBPR is evaluated using the Opportunistic Network (ONE) simulator [5] and compared against that of the HBPR [3] and AEProphet routing protocols, in terms of average remaining energy, number of dead nodes, number of delivered messages, and overhead ratio, under varying number of nodes, message size, and message generation interval; where AEProphet is the ProPHet routing protocol [6] on which the same energy-aware mechanism has been implemented.

1.2 Approach

For the design of the proposed AEHBPR protocol, the same assumptions that were utilized for HBPR [3] prevail, and new energy-related factors have been introduced as follows:

- When a message has been delivered to its destination, there still many copies of it moving around in the OppNet. This unnecessary transmission consumes a lot of energy in the sending and receiving of messages by the nodes. These extra copies of the delivered messages occupy unnecessary spaces in the buffer of the nodes and may also cause congestion in the network, thus degrade the network performance since the overhead ratio may greatly increase. A one-hop acknowledgment mechanism is introduced to remedy to this situation, which consists of removing copies of an already delivered message from the buffer of other nodes carrying the same message, and ensuring that no extra copy of the same message is generated and transmitted in the network.

  Typically, when a message reaches its destination, an acknowledgment message (Ack_M) containing (Message ID, Source ID, Destination ID) is sent from the destination node to the last encounter node that had send the message to destination. Both nodes update their acknowledgment tables (Ack_Table) with the Ack_M information, and then remove the message from their buffer to prevent its further relaying. Afterwards, these two nodes flood the Ack_Table information to all nodes in the network. Any node carrying the already delivered message that come in contact with the node having the updated Ack_Table updates its table by removing a copy of that message in its buffer. This process continues until all copies of the message have been removed. This process is referred to as one-hop acknowledgment because only one acknowledgment message Ack_M is sent from the destination to the last relay node, and the rest of the Ack_M flooding is done by the exchange of the Ack_Table information among other nodes.

- A minimum energy threshold (in Joules) is set so that whenever the energy level of an encountered relay node (which is not the destination) is checked and found to be less than that threshold, the message will not be sent to that node. This threshold is set by the network system administrator. In our work, its value has been set through simulation experiments.

- If the next hop is the destination node, the message is forwarded directly to it without computing the utility metric as was done in HBPR [3].

1.3 Thesis Contributions

The contributions of this thesis are as follows:

- Design of a history-based energy-efficient routing protocol for opportunistic networks (called AEHBPR) which relies on the introduction of a one-hop acknowledgment mechanism.
- Design of AEProphet, the ProPHet routing protocol for OppNets, on which the same energy-aware mechanism has been implemented.
- Performance analysis of the proposed routing protocols using the ONE simulator [5], showing that AEHBPR outperforms HBPR, and AEProphet in terms of predefined performance metrics.

1.4 Thesis Outline

The thesis is organized as follows:

- **Chapter 1** introduces the subject, motivation, and contributions of our research.
- **Chapter 2** presents some background information and related works.
- **Chapter 3** describes our proposed AEHBPR in depth.
- **Chapter 4** describes the performance analysis of the proposed routing algorithms.
- **Chapter 5** concludes our work and highlights some future work.

Chapter 2

Background and Related Works

2.1 Background

This section introduces OppNet, its main characteristics and routing challenges in such networks.

2.1.1 Opportunistic Networks

Traditional ad hoc networks and systems have been designed to provide a one-size-fits-all basis for the deployment of all types of applications. As such, they may not necessary accommodate certain types of specialized applications such as Emergency Preparedness and Response (EPR).

To overcome this deficiency, the design of the so-called specialized ad hoc networks and systems (SAHNS) [7] have been advocated as possible solutions. Opportunistic networks (OppNets) is category of SAHNS, which can be viewed as a pragmatic evolution of the generic Mobile Ad Hoc Networks (MANETs) paradigm.

An OppNet [8] can be considered as a kind of challenged mobile multi-hop ad hoc network, characterized by prolonged disconnections, partitions, unpredictable and unstable topology, asymmetric data rates, long or variable delay, to name a few. This results in a paradigm shift for the design of network services in the sense that OppNets are quite different from legacy MANETs [9] because the network is disconnected as a rule rather than as an exception to deal with.

In an OppNet, nodes are mobile (they can be pedestrian, vehicles, people equipped with smart devices, to name a few) or fixed devices. The nodes are expected to discover each other using all kinds of communication media; for instance, Bluetooth, WiFi, RFID, cellular technologies, satellite link, point of access toward fixed Internet, to name a few. The network itself is made of several network partitions (so-called regions) and nodes in these regions can be interconnected with each other by means of a store-carry-and-forward message switching mechanism following a new protocol layer (called bundle layer), which is implemented on top of heterogeneous region-specific lower layers [10], [11]. In this sense, OppNets are also considered as a subclass of delay tolerant networks (DTNs) [12]. Typically, the goal of OppNets is to be able to simultaneously exploit and leverage the resources of the above-mentioned separate network regions according to the needs of specific application tasks.

2.1.2 Main Characteristics and Requirements of Opportunistic Networks

In OppNets [8], due to the highly mobile nature of nodes and the uncertainty of mobility patterns, intermittent connectivity frequently occur. Therefore, some of the main features that have driven the design of routing protocols for OppNet include:

- *Contact opportunity*: Because of the dynamics of wireless channels and node's mobility, a contact between two nodes is initiated at an unpredicted time. Due to this, such contacts must be exploited opportunistically for exchanging the messages. The contact capacity i.e. the amount of data that can be transferred between two nodes that come in contact with each other, should also be considered as design criterion.

- *Storage limitations*: To avoid/reduce the dropping of packets, it is necessary that enough storage be available at each intermediate node to store all messages for an unpredictable period of time, i.e. until the next contact occurs. Consequently, the routing and replication strategies must account for the storage constraint. One way to achieve this is by implementing a buffer management strategy such as the one proposed in [13] if the node storage capability is limited.

- *Cooperation level*: In an OppNet, some of the nodes may be required to share their own resources (memory, bandwidth, battery power, to name a few) with other nodes without having in return any direct compensation for doing so. In such situation, a strategy based on reciprocal altruism may not suffice to guarantee some form of cooperation enforcement between the nodes. In this sense, few works in the literature have considered the necessity of using incentives mechanisms [14] to boost the routing and data forwarding in OppNets.

- *Mobility modelling*: As OppNets are a type of intermittently connected networks, human users carrying mobile devices are often considered as the typical architecture. In that setting, mobility can be exploited by attempting to retrieve the inherent user interests, habits, social features, to name a few, for the purpose of simulating and evaluating various OppNet scenarios. In this sense, human mobility modelling is an important aspect to be considered when designing content dissemination schemes for OppNets. It should be noted that the characterization of mobility is often context-specific and it is difficult to perceive how the environment of analysis and the tracked mobile devices can have an influence on the observed mobility patterns. This has led to several human mobility issues, generally classified into three categories: mobility models and traces (both synthetic and realistic), mobility prediction techniques, and mobility characteristics. A comprehensive description of them is provided in [15].

- *Energy efficiency*: In an OppNet, node connectivity is considered transient. As such, nodes commit solely to human mobility for message delivery. In doing so, energy consumption at a node does not occur only through transmission and reception of packets, but also through processing. This means that any node in the network may quickly deplete its energy resource and may be unwilling to participate in the message forwarding process. Therefore, a tradeoff between energy consumption, transmission and receiving powers, and participation in the route selection process should be made when designing energy-aware routing protocols for OppNets; and the resulting energy expenditures are usually dependent on the considered design choices.

2.1.3 OppNet Architectures, Mobility Models, Tools, and Main Research Challenges

Most representative OppNet architectures include: pocket switched networks (PSNs) [16] - where user mobility and occasional transmission opportunities are exploited to carry the message between nodes up to the destination; autonomous networks [17] - where the aforementioned network regions are interconnected in an automatic way so as to enable the message delivery to the intended destination; socio-aware community networks [18] - where the focus is on human-to-human communication to deliver the message to destination. In this case, intermittent connectivity, node context information, and social mobility models are exploited to determine the community (i.e. group of nodes) that are likely to carrying the message to destination. The routing protocol proposed in this thesis belongs to this later category of schemes.

In the literature, some models and tools for representing the behaviour of OppNets when designing such networks have been investigated. Examples of these include:

- Models based on the expectations of how mobility is performed in specific situations such as campus and vehicular mobility models [19].
- Models that allow to adjust the Random WayPoint parameters with specific distributions to yield more realistic OppNet scenarios [20]

- Mobility measurements performed both indoor and outdoor. For instance, like in the iMotes experiments [21], [22] where mobile users carried Bluetooth enabled devices, that periodically records the presence of other Bluetooth enabled devices such as PDAs, other iMotes, laptops or mobile phones.
- Mobility models such as the Sociological Interaction Mobility for Population Simulation (SIMPS) [23], where the mobility modelling approach is centered on human behavioural rules.
- Synthetic traces collected by using the *second life* concept [24]
- Using the Opportunistic Network Environment (ONE) simulator [5] to simulate mobility and OppNets behaviour [25].

In this thesis, we have used the dataset available in [22], which is made of real mobility traces created using 36 nodes of bluetooth sightings by a group of users carrying small mobile devices for a number of days, by importing it to the ONE simulator [5].

The main research challenges in OppNets include: (1) challenges in OppNet architectures and applications, energy efficiency and fairness, content dissemination, routing [26]; (2) challenges in security and privacy [8], (3) challenges in mobility characterization and discovery [27]; challenges in scheduling, resource allocation and MAC schemes [28] to name a few.

2.1.4 Opportunistic Networks Applications

OppNets can be deployed for several types of applications, including Emergency Preparedness and Response (EPR), protection of critical infrastructures, environment, heath care, manufacturing, wireless and underwater sensor networks, smart homes, day-to-day scenarios, privacy and security, to name a few [29]. Few OppNets case studies and framework architectures have also been investigated. Examples of these include: the Haggle Project [30] - which focuses on solutions for autonomic behaviour for communication in autonomous-based

OppNets.

These solutions are specifically target to smartphones applications; the ZebraNet project [31] - where a prototype OppNet has been deployed on zebras to study their migrations and interactions; the DakNet project [32] and KioskNet [33] - where an OppNet design has been deployed in rural areas in India to provide internet connectivity using few collection points that are able to temporarily store the messages addressed to the Internet; PoDNet [34] - This is a framework that enables opportunistic sharing of content of interest among the nodes based on a publish/subscribe paradigm such as Podcasts; SCAMPI [35] - This is a service platform built on top of the DTN and the above-mentioned Haggle [30] and PodNet [34] frameworks, used to enable flexible routing of messages between heterogeneous nodes based on the concept of DTN bundles [12]. A prototype of a small scale OppNet implementation (called microOppnet) has also been introduced in [36].

2.1.5 Routing in Opportunistic Networks

In traditional MANETs routing schemes [9], it is implicitly assumed that the network is connected and an end-to-end path always exists between the source and destination nodes. In OppNets, due to network partitions, node's mobility and failure, a fixed path between the sender and the receiver may never exist [37]. Routing is performed in a store-carry-and-forward fashion based on contact opportunity between nodes caused by mobility.

Typically, at any given point in time, a sender node has to select a node (so-called encounter) from a group of nodes in its neighbourhood, that is qualified as best forwarder node towards the desired destination. In case the best forwarder node is not found, the sender node must keep the message copy in its buffer, until it opportunistically encounter another suitable node or the destination [38]. Thereby, in order to be able to store all copies of messages while waiting for the best forwarder, each intermediate node involved in this process must maintain an adequate buffer capacity, which may lead to potentially long delays experienced by the messages [39].

2.1.5.1 Classification of Routing Protocols for OppNets

Routing in OppNets can be classified in two categories: infrastructure-based [40] and infrastructureless-based [41]. In the former category, some form of infrastructure constructed using a set of specific nodes (for instance those nodes with high power level, enough transmission range, buffer capacity, etc) is exploited for message forwarding purpose using reliable agents. Examples of such routing schemes are given in [42], [43], [44]. In the latter category, node's mobility and contact opportunity between nodes are the main driven parameters for the message routing and forwarding decision. Examples of such routing schemes are given in [45], [6], [46], [47], [3], [48], [49]. The routing scheme proposed in this thesis belongs to this category. Indeed, it is an energy-efficient version (so-called AEHBPR) of the HBPR protocol introduced in [3]. A pseudo-code of our implementation of the energy-aware version of HBPR is provided in **Appendix A**.

For the sake of comparison, we have also considered the Probabilistic Routing Protocol using History of Encounters and Transitivity (so-called ProPHet) [6] as benchmark protocol, on which the same energy-aware mechanism used for AEHBPR has been implemented (yielding the so-called AEProphet scheme). For this reason, a brief description of the ProPHet routing protocol is provided next.

2.1.5.2 ProPHet Routing Protocol

In the ProPHet protocol [6], it is assumed that the nodes move in a predictable fashion based on the property of transitivity, the history of encounter, and some mobility patterns that are likely to be repetitive. This scheme relies on the idea that if a node A has encountered another node B more often or has visited node's B location many times in its past history, then the probability that node A will meet node B again or will visit node B's location again in a near future is considered high. Following this idea, prior to sending a message, each node (referred to as source node) calculates a probabilistic metric (called delivery predictability) for its known destination. Whenever this source node meets an encounter, it forwards a copy of the message to that encountered node only if that node has the highest delivery predictability value among all the encounters. A pseudo-code of our implementation of the energy-aware version of PRoPHET is provided in **Appendix B**.

2.2 Related Work

To the best of our knowledge, only few energy-aware routing schemes for OppNets have been investigated in the literature [50], [51], [52], [53], [54], [55], [56], [1].

In [50], Lu et al. proposed n-epidemic, an energy-efficient routing protocol for delay tolerant networks, which relies on the idea that a relay node should not forward a packet when it meets an encounter, but rather should wait for more encounters in its neighbourhood so that its packet when transmitted can reach them, thereby consume less energy in packet forwarding while guaranteeing that more node will receive the packet. To guarantee that a relay node cannot send a packet as casually as it was the case in the epidemic routing, a lower bound is imposed on the above-mentioned number of neighbours, leading to the calculation of a suitable node s delivery predictability.

In [51], Gao et al. proposed an energy aware routing scheme for OppNets that improves the routing efficiency of the Spray-and-Wait protocol [46]. In their scheme, an utility metric that relies on the residual energy and current velocity of a node is designed to determine the optimal number of copies of the message that should be exchanged when two nodes encounter each other, helping to avoid the blindness in the Spray phase of the Spray-and-Wait routing scheme [46].

Similarly, in [52], Patel et al. introduced an energy aware routing scheme for OppNets that consists of modifying the spray phase of the Spray-and-Wait protocol [46]. Indeed, in the spray phase, whenever a node meets an encounter, their vibrancy are exchanged. Based on this information and the amount of remaining energy at the sender node, the number of copies of the message to be forwarded to the recipient node is determined, which helps avoiding the blindness in the spray strategy of Spray-and-Wait protocol [46], resulting to an alleviated wait phase.

In [53], Chilipirea et al. proposed an energy-aware version of the BUBBLE Rap protocol [57]. In their scheme, a relay node decides on the next hop to carry the message by using the same utility function that was introduced in Bubble Rap, but which now incorporates some energy related parameters. Based on this utility function, the probability for a next node to carry the message is evaluated to determine whether the next hop has sufficient energy resources or not to support the message being transferred. Therefore, their scheme prevents any node from accepting the messages in transit if its battery level is depleted.

In [54], Dhurandher et al. proposed an energy-efficient routing protocol for Oppnets that uses a genetic algorithm to select the best forwarder of a node based on its personal information table and some information about the neighbour node groups. Basically, a set of random chromosomes is initialized and later updated based on some selection and crossover mechanism, and a fitness function is utilized to find out whether a node can be qualified as a suitable forwarder.

In [55], Yao et al. investigated the energy consumption wastage caused by the presence of a large number of isolated nodes in a sparse OppNet topology at the idle listening stage. An energy-aware routing protocol for sparse OppNets (so-called ERASA) is proposed, which is based on the asynchronous sleep approach. Through simulations, the proposed ERASA scheme is shown to achieve a considerable energy saving by imposing the isolated nodes to enter the low power consumption dormancy state and to become timely awaken when other nodes enter into their communication range.

In [56], Wennerstromy et al. discussed on ways to characterize the heterogeneous link quality of connections that can occur in an OppNet from the perspective of routing decision making. They suggested that the knowledge of multi-contact opportunities can be exploited to limit the energy consumption when performing the routing process. Using this idea, an energy efficient routing protocol for OppNets is proposed, in which the multi-contact opportunities are leveraged with link quality features to improve the network energy savings while yielding a reduced number of message relays.

In [1], Dhurandher et al. proposed an energy-aware HBPR scheme [3] by introducing the following energy-related factors in the utility function used by HBPR for deciding on the next hop node to forward the message: (1) the perpendicular factor - to consume lesser energy when a relay node forwards a message to the next hop, the actual distance between the relay node and the next hop node is checked instead of the perpendicular distance between the neighbouring nodes and the SD line, (2) transmission factor: this new factor is meant to impose the number of message copies with a unique ID that a node can forward, and (3) a new sparse constant factor: this is used to avoid the calculation of the utility metric in case the number of nodes in the sender's neighbourhood is lesser than a sparse constant factor.

In this thesis, the HBPR protocol [3] is redesigned by incorporating a mechanism in it that helps: (1) avoiding the unnecessary transmissions of packets that have already been delivered to the destination, and (2) forwarding the packets only through those encountered relay nodes that have enough amount of remaining battery power, yielding the AEHBPR scheme.

Chapter 3

Methodologies

In this chapter, the HBPR protocol [3] is described, followed by the design of our proposed AEHBPR protocol.

3.1 HBPR Protocol

The HBPR protocol [3] has been designed according to the steps presented in Fig. 3.1. These steps are described in Subsection 3.1.2. In the following, we describe the data structures used by HBPR.

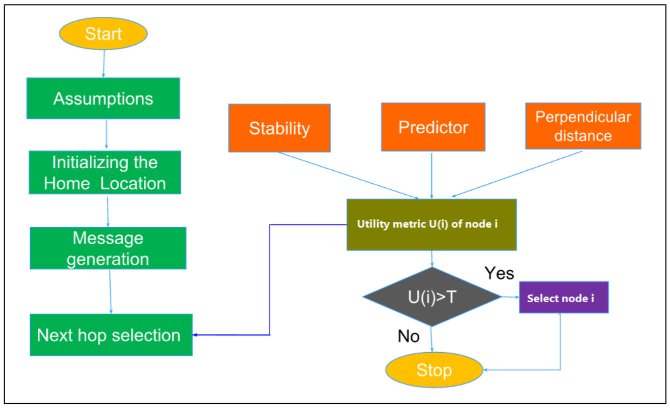

Figure 3.1: Flow Chart of the HBPR protocol.

3.1.1 Data Structures

For the design of the HBPR protocol, the following assumptions were made [3]:

- The nodes in the network are cooperative and have no malicious intent.
- The nodes move according to a customized mobility model(CMM), visiting some locations more frequently that others.
- The simulation area is subdivided into numbered cells, each representing the location (i.e. coordinates) of a node at any given time. A cell that a node visits more frequently than others is considered as its home location.

Based on these assumptions, the history of a node's mobility pattern over a period of time, the direction of the node's movement, the amount of time taken by a node to meet the other nodes, are the main steps used to decide the message forwarding, using the following data structures.

3.1.1.1 History Table

Each node keeps tracks of its own movement in a *History Table* whose records are in the form of *(Time, Location)*, reflecting the change in the location and timer (timestamp) of nodes assuming that any node's timer is set to 0 at the start of the simulation. Generally, it is considered that a node maintains a record of up to 100 movements, and any 101^{th} entry automatically removes the first one based on a First-In-First-Out (FIFO) mechanism in order to preserve the memory space. An example of *History Table* showing the cell numbers {23,45,45,40,45} included in the simulation area is shown in Table 3.1.

3.1.1.2 Home Location Table

A cell that a node visits more frequently is considered as its home location. Each node initially advertises its home location so that all nodes are aware of this information. Whenever a node opportunistically encounters another one, it keeps track of this encounter's home

Time	Location
0.45	23
1.2	45
2.6	45
2.9	40
3.3	45

Table 3.1: History Table [3]

location (i.e. Host ID) in a Home Location Table, whose records are in the form *(Host ID, Home Location)*. This step is referred to as Home location initialization. At every encounter , the nodes shares their Home Location tables, and in case the same entries are not found, the home location tables are updated according to the most recent home location entry of that node from its history table. An example of Home Location Table is depicted in Table 3.2, where {31,43,84} are the cell numbers in the simulation area.

Host ID	Home Location
P1	31
P5	43
C2	43
T2	43
C7	84
T4	84

Table 3.2: Home Location Table

3.1.2 How the HBPR Protocol Works

Using the above mentioned data structures, the functionality of HBPR protocol (Fig. 3.1) can be described in three phases as follows.

1. Initialization of the Home Location: Initially, a head start is given to the network by initializing its nodes with the home locations(cell numbers) they are residing in. Each node floods their home location in the network so that all nodes are aware of

each other's location. Each node stores the other node's home location in its Home Location Table as shown in 3.2. This step is referred to as Home location initialization.

2. Message Generation and Home Location Update: Some specific nodes are selected that generate new messages, each of which contains the destination ID. Basically, these messages are meant to alert some nodes in the network to update their Home Location Tables due to changes that may have occurred in their locations. This step is referred to as Message generation and home location update.

3. Next Hop Selection: For a node k to select the next best forwarder, a utility metric $U(k)$ is calculated which depends on three parameters, which are: *stability of the node's movements*, *prediction of the next location of a node* (i.e. of the future movement of the node), and *perpendicular distance of the neighboring node from the line of sight of source and destination (SD line)*. Any node whose utility value is greater than or equal to a prescribed threshold is assigned a message copy, which may eventually arrives at the destination.

   - Stability of Node's Movements:

     This reflects a change in the average speeds of a node, which can be either significant (meaning that the node's movement is unstable) or nominal (meaning that the node's movement is stable). The timestamp parameter of the History Table 3.1 is used to calculate the average speed of a node over any two different position. A list of these averages speeds is recorded. Based on this list, it is possible to determined whether a change in average speed is nominal or significant. If the change in average speed is significant, the node is considered as unstable, otherwise it is considered as stable. Typically, all nodes are initially assigned a stability value S; and if the change in two consecutive average speeds is less that 10 units per second, the stability value S is unchanged. Otherwise, it is decreased using the following equation [3]:

$$S = S * (1 - S) * S_i \tag{3.1}$$

- Prediction of the Next Location of a Node:

  Based on the location history contained in the History Table of the node, the prediction of the next location of the node is calculated by means of a 2-state Markov predictor model [3]. For a given pattern of visits, a table that records the frequencies of visit for any location is maintained and the predictor uses this table to find the next location of a node.

  As an example, let's considering that the past history is
  AGHBGTYGHIGHYKLOPWNGHWBKJDNGH
  RJBFJGHYKJFNGHYLKJNKSGHWOKSADGH,
  the next location will be Y because Y occurs the most as shown in the sequence below
  A**GH**BGTY**GH**I**GH**YKLOPWN**GH**WBKJDN**GH**
  RJBFJ**GH**YKJFN**GH**YLKJNKS**GH**WOKSAD**GH**

- Perpendicular distance of the neighboring nodes from the SD line:

  This metric is used to select those nodes that are at closer distance to the SD line. The utility metric $U(i)$ of node i is calculated as [3]

$$U_i = \sum_{j=1}^{3} W(j) * V_i(j) \tag{3.2}$$

where Vi(1) is the stability metric, Vi(2) is prediction metric, Vi(3) is the perpendicular distance metric for node i, and W(j) is a weight parameter, $j = 1, 2, 3$.

With this calculation, any node k whose utility value satisfies $U(k) \geq T$, where T is a prescribed threshold, can be considered as a candidate node for message forwarding. For all our simulation results (see Chapter 4), we have considered $T = 0.6$ because HBPR performs well with that value compared to any other threshold value in terms of number of messages delivered while keeping the overhead ratio as low as possible.

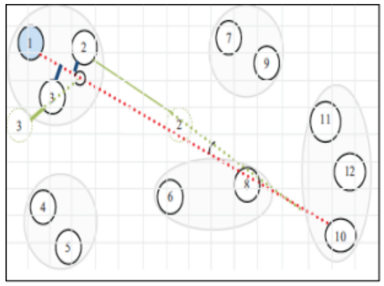

Figure 3.2: An Example scenario of the HBPR Protocol taken from [3].

A network scenario simulating the working of HBPR is illustrated in Fig. 3.2, where the future predicted locations of all nodes, the SD line, and the perpendicular distance from the SD line are shown in dotted green circles, red line, and blue line respectively. In this scenario, the (S,D) pair is (1,10) which is connected by an imaginary SD line. The future predicted locations of nodes 2 and 3 are shown as well as the perpendicular distances of nodes 2 and 3 with respect to the SD line using the dark blue line.

3.2 Proposed AEHBPR Protocol

For the design of the proposed AEHBPR protocol, the same assumptions utilized for HBPR [3] prevail, and new energy-related parameters have been introduced as follows:

A) When a message has been delivered to its destination, there still many copies of it moving around in the network. This unnecessary transmission consumes a lot of energy in the sending and receiving of messages by the nodes. Also, these additional copies of the delivered messages occupy unnecessary spaces in the buffers of nodes and may cause congestion, thus may degrade the network performance due to increased overhead ratio that may occur. To remedy to this situation, a one-hop acknowledgment mechanism is introduced which consists of removing the copies of already delivered messages from the buffers of other nodes carrying the same message, and ensuring that no extra copy of the same message is generated and transmitted in the network.

Typically, when a message reaches its destination, an acknowledgment message (Ack_M) containing (Message ID, Source ID, Destination ID) is sent from the destination node to the last encounter node that had send the message to the destination. Both nodes update their acknowledgment tables (Ack_Table) with the Ack_M information as shown in Table 3.3, then remove the message from their buffers to prevent further relaying of it.

Message ID	Source ID	Destination ID

Table 3.3: Ack_Table containing Ack_M

Afterwards, these two nodes flood the Ack_Table information to all the nodes in the network. Any node carrying the already delivered message that come in contact with the node having the updated Ack_Table updates its table by removing a copy of that message in its buffer. This process continues until all copies of the message have been removed. This process is referred to as one hop acknowledgment because only a one-hop acknowledgment message Ack_M is sent from the destination to the last relay node, and the rest of the Ack_M flooding is done via a Ack_Table information exchange among the other nodes.

B) A minimum energy threshold (in Joules) is set so that whenever the energy level of an encountered relay node (which is not the destination node) is checked and found to be less than that threshold, the message will not be sent to that node. This threshold is set by the network system administrator. In this thesis, its value has been set through simulation experiments.

C) If the next hop is the destination node, the message is forwarded directly to it without computing the utility metric as done in HBPR [3].

3.2.1 AEHBPR Flowchart and Algorithm

Following the above-mentioned requirements (a), (b), (c), the flowchart and the pseudo-code of the proposed AEHBPR protocol are described as follows.

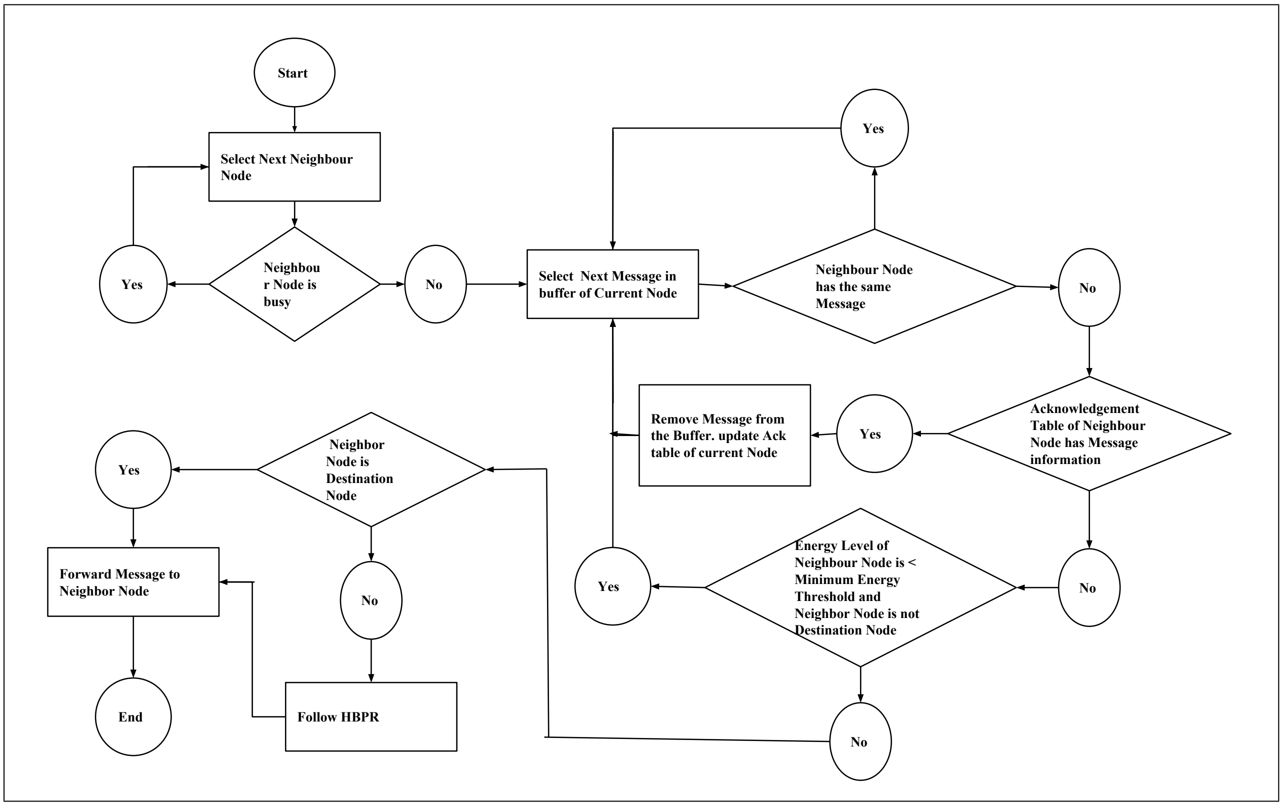

Figure 3.3: Flow Chart of AEHBPR

According to Fig. 3.3, the proposed AEHBPR works as follows: whenever a node (called current node) wishes to send messages to its destination, (i) it selects an available neighbour node among the existing ones; (ii) upon connection with this neighbor node, the acknowledgement tables are exchanged; (ii) using this table, the current node loops on the messages in its buffer one by one to check if the selected neighbour node already has any of these messages; (iii) if the acknowledgement table of the neighbour node has the same message information than that of the current node, it means that the current selected message has already been delivered by that neighbor node, indicating that it is an extra copy. In that case, the current message is removed from the buffer of the current node and the current

Algorithm 1 for AEHBPR

Step1: Select the next neighbour node(NN)
Step2: If NN is busy then goto Step1
Step3: Repeat All message(M) of current node(CN)
    4a: if NN has M then goto Step3
    4b: check Ack_Table(AT_NN) of NN for M
        If Ack_Table of NN has M then
            Remove M from buffer of CN
            Update Ack_Table(AT_CN) of CN
            goto Step3 for next M
        end if
    4c: If Energy Level of NN < Minimum Energy
        Threshold(MET) and NN is not
        Destination Node(DN) then
        goto Step3
    4d: If NN is DN then
            forward M to NN
        else Follow HBPR to send message to NN

Algorithm 2 for the destination Node receiving a Message

Step1: Receive message(M) from the Last sender Node(LSN)
Step2: If Destination Node(DN) of M is current node(CN) then
        Send Ack_M to LSN
        update Ack_Table of CN with Ack_M
        remove M from the buffer of CN
    end if

Algorithm 3 for the last sender receiving the acknowledgment

Step1: Receive Message(M) from the Destination Node(DN)
Step2: If M contains Ack_M then
        update Ack_Table of CN with Ack_M
        remove M from the buffer of CN
    end if

node's acknowledgment table is updated, and the next message is selected from its buffer to continue the process; (iv) if in step (iii), it is found that a current message is not already delivered, it is checked if the energy level of the neighbour node is less than the minimum energy threshold and if the neighbour node is not the destination node. If that is true, the neighbour node is not selected as message forwarder and the process repeats from step (i). In step (iv), if the neighbour node is the destination node, the message is forwarded to it directly. If not, the HBPR protocol is followed to decide on the selection of the next neighbour node to forward the message.

3.2.2 AEProphet Protocol Design

AEProphet is the energy-aware version of the ProPHet protocol [6] on which the above-mentioned energy-aware mechanism has been implemented. The detail of the code for AE-Prophet is provided in *Appendix A*. The flowchart of the AEProphet algorithm is given in Fig. 3.4. It can be described in a similar way than that of AEHBPR (see above steps (i) to (iv)), with the exception that in the above step (iv), if the neighbour node is the destination node, the message is forwarded to it directly. If not, the ProPHet protocol is followed to decide on the selection of the next neighbour node to forward the message.

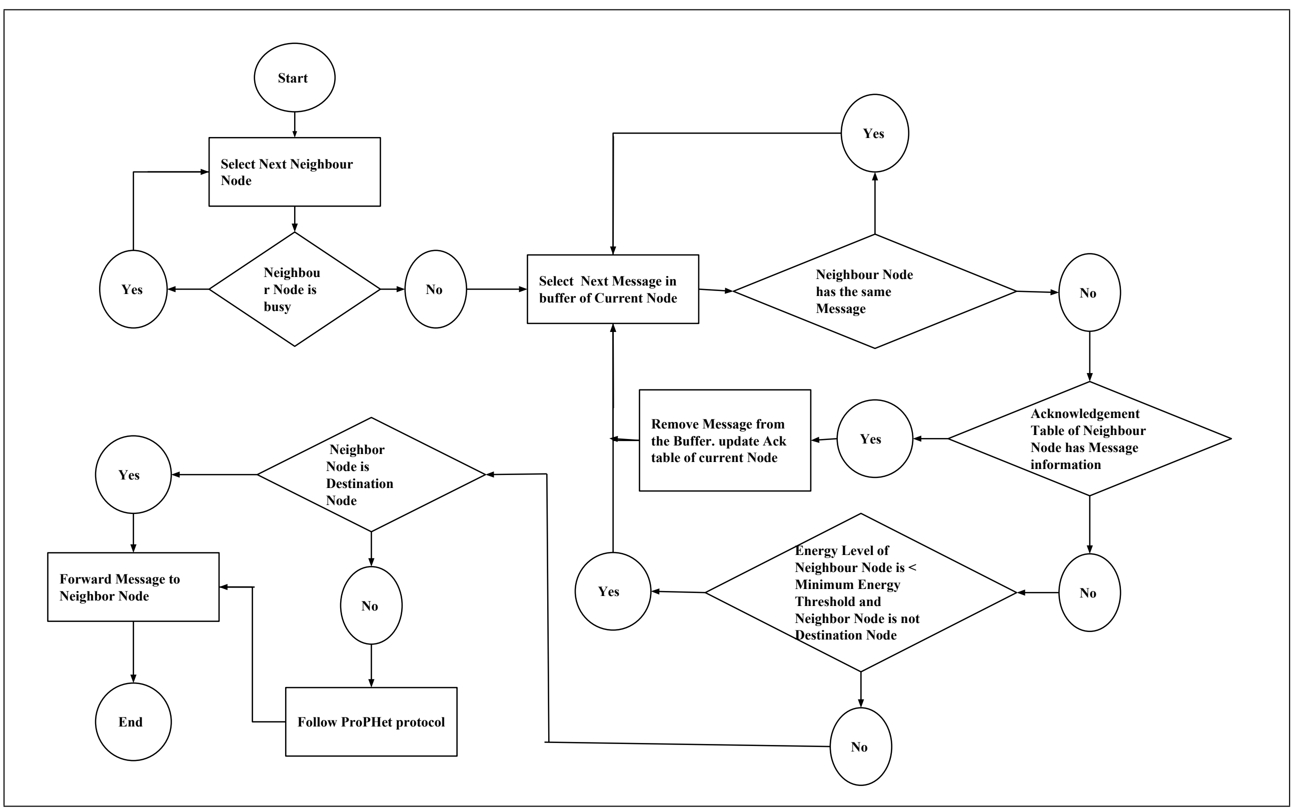

Figure 3.4: Flow Chart of AEProphet

Algorithm 4 for AEProphet

Step1: Select the next neighbour node(NN)
Step2: If NN is busy then goto Step1
Step3: Repeat All message(M) of current node(CN)
4a: if NN has M then goto Step3
4b: check Ack_Table(AT_NN) of NN for M
If Ack_Table of NN has M then
Remove M from buffer of CN
Update Ack_Table(AT_CN) of CN
goto Step3 for next M
end if
4c: If Energy Level of NN < Minimum Energy Threshold(MET) and NN is not Destination Node(DN) then
goto Step3
4d: If NN is DN then
forward M to NN
else Follow ProPHet to send message to NN

Algorithm 5 for the destination Node receiving a Message

Step1: Receive message(M) from the Last sender Node(LSN)
Step2: If Destination Node(DN) of M is current node(CN) then
Send Ack_M to LSN
update Ack_Table of CN with Ack_M
remove M from the buffer of CN
end if

Algorithm 6 for the last sender receiving the acknowledgment

Step1: Receive Message(M) from the Destination Node(DN)
Step2: If M contains Ack_M then
update Ack_Table of CN with Ack_M
remove M from the buffer of CN
end if

Chapter 4

Performance Evaluation

In this Chapter, we study the performance of our proposed AEHBPR protocol using the ONE simulator [5] version 1.5.1 RC2, and compare it against the performance of the HBPR and AEProphet protocols, using the Custom Mobility Model (CMM), and a real mobility traces dataset taken from [22], under varying number of nodes, message size, message generation interval.

4.1 The ONE Simulator

The ONE simulator is a Java-based platform on which various delay tolerant network (DTN) protocols functionalities and simulation features are implemented. These include importing/-exporting mobility traces (real and synthetic), analysis and visualization interfaces, event generation, energy consumption of nodes, node movement modelling, message handling, routing and forwarding, inter-node contacts modelling, to name a few.

In the ONE simulator, the Node is the main component; it is considered as mobile, and it is equipped with the required hardware. Examples of nodes include car, train, pedestrian, tram, to name a few. Nodes are managed by groups, each of which is configured with its own set of parameters. The basic parameters of a node include energy consumption, persistent storage, movement, routing, interface, to name a few. These parameters can be used to implement energy-aware specific algorithms such as the one proposed in this thesis.

In the ONE simulator, the way that nodes move is determined by the mobility model used. Such models are characterized by factors such as node speed and coordinates, pause-time, to name a few. Several mobility models have been implemented in the ONE simulator [5]. These include: the Shortest Path Map Based Movement model, the Map-constraint, the RWP movement model, the Car movement model, the Working Day Movement model, the Human behavior based movement model [3]. However, more customized models such as the Custom Mobility Model (CMM) introduced in [3] and used in this thesis can also be embedded in the ONE simulator.

In the ONE simulator, there are routing modules that describe how the routing of messages can be performed in a DTN environment using the store-carry-and-forward fashion. Many benchmark DTN routing techniques so far proposed in the literature have been implemented [5]. Examples of such routing protocols are Spray and Wait, PRoPHET, Epidemic, MaxProp. Each routing module inherits the basic functionality of the MessageRouter and ActiveRouter modules [5], namely: buffer management, message transfers, message aborts, call backs for various messages events, to name a few.

in the ONE simulator [5], the message generation can be performed in two different ways: (1) via message generators - where messages are generated randomly using a fixed source, destination, interval or size, or (2) at specific times by specifying the message ID and a fixed source-destination node pair.

A graphical user interface of the ONE simulator [5] is illustrated in Fig. 4.1. In this figure, the main window shows the node location, current paths, and transmission range of nodes. The right side of this window shows the number of nodes in the simulation area. The simulation is started by using the pause/play button on the top left corner of the window. In the event log section of the window, the connections that have been created and the messages that have been transmitted are shown. A statistic report module is available which helps for collecting the statistics on the performance of the run protocol. Examples of statistics include the number of messages delivered, the delivery ratio, the overhead ratio, just to name a few.

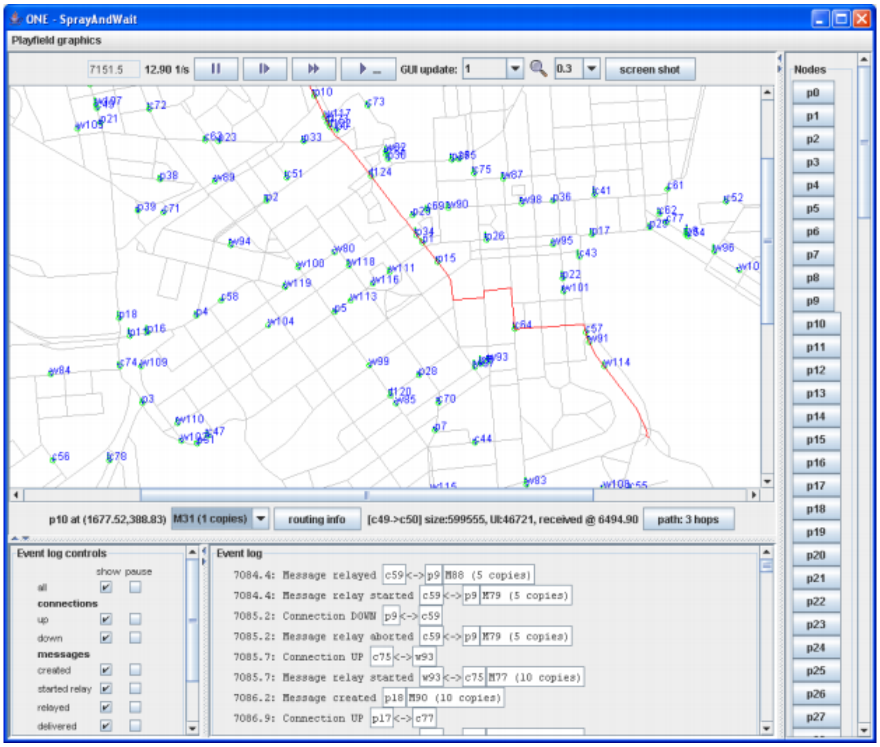

Figure 4.1: Screenshot of the ONE simulator

4.2 Simulation Settings

In our simulations, the nodes are mobile and are divided into four groups (called communities), each of which has 25 nodes. The whole world size is divided into cells of 100 m $\times$100 m. The first group of nodes are pedestrians, with a speed varying between 0.5-1.5 m/sec. The second group is a group of cyclists with a speed varying between 1.5-5 m/sec. The third group is a set of cars, each with a speed varying between 5 - 10 m/sec, and the fourth is the group of Trams, each with a speed varying between 7-13 m/sec. The mobile nodes have a transmission range of 10 meters and a transmission speed of 2 Mbps. The weights that are assigned to the parameters of the utility function for HBPR are inherited from [3], i.e.

$W(1) = 0.4$ for the stability metric, $W(2) = 0.4$ for the prediction metric and $W(3) = 0.2$ for the perpendicular metric. When the simulation starts, every node is present at its Home Location. Afterwards, a node travels to its Home Location with a probability p and to all other locations with a probability $1 - p$. Other simulation parameters are provided in Table 4.1. Most of these parameters are inherited from the HBPR [3] design.

Parameters	**Values**
Simulation area	4500 m * 3400 m
Number of nodes	40, 80, 120, 160
Communication Interface	Bluetooth
Buffer capacity of Group1 and Group2 nodes	5 Mb
Buffer capacity of Group3 and Group4 nodes	50 Mb
Initial energy of all nodes	5000 Joules
Message size	0K–500K, 500K–1MB, 1MB–1.5MB, 1.5-2MB
Message generation interval	5–15sec,15–25sec,25-35sec,35–45sec
Simulation time	43200 s
Message time-to-live	300 min
Real traces datase	from Haggle project [22]
Minimum Energy Threshold	600 Joules
Threshold	0.6

Table 4.1: Simulation Parameters.

4.3 Performance Metrics

The performance metrics used to evaluate the studied protocols are:

- *Average remaining energy:* This is the total average remaining energy of the nodes at the end of the simulations.
- *Dead nodes:* This is the number of nodes whose remaining energy are less than the minimum energy threshold at the end of the simulations.
- *Message delivered:* This is the number of messages successfully delivered to the destination at the end of the simulations.
- *Overhead ratio:* This is a measure of the bandwidth efficiency, calculated as: *(Number of relayed messages - Number of delivered messages) / Number of delivered messages.*

4.4 Mobility Models

Mobility models are intended to represent the way that the nodes move in the simulation. In this thesis , three types of movement models are used, namely:

- Custom Mobility Model [3]: In this model, the nodes move according to the human mobility pattern and are allowed to flood the network with their most visited locations when the network becomes operational.
- Real Mobility Traces: We have used real mobility traces obtained from [22]. This dataset was created with 36 nodes of bluetooth sightings by a group of users carrying small devices (iMotes) for a number of days.

4.5 Simulation Results Using the Custom Mobility Model

This section describes the results obtained when evaluating the studied routing schemes for OppNets using the Custom Mobility Model (CMM) [3] under various scenarios.

4.5.1 Varying Thresholds

In this scenario, the number of nodes is fixed to 240. The aforementioned threshold T on the utility function used in HBPR [3] is varied from 0.2 to 0.7 with an increment of 0.1 each time, and the impact of this variation on the number of delivered messages and overhead ratio is studied when using the proposed AEHBPR protocol. The goal is to determine the best value of T for which the maximum possible number of messages delivered is achieved while maintaining the overhead ratio as low as possible. The results are captured in Fig 4.2 and Fig 4.3.

In Fig. 4.2, it is observed that initially, the number of messages delivered decreases when the threshold is increased, but then, when the threshold value is greater than 0.3, it starts to increase progressively till the point 0.6, then recommence to decrease. On the

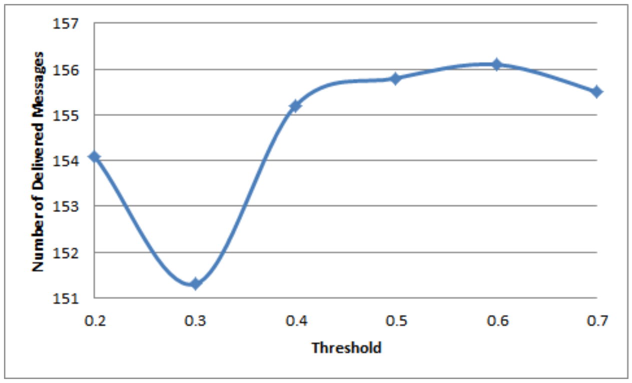

Figure 4.2: Number of delivered messages under varying threshold values using AEHBPR under the CMM Model.

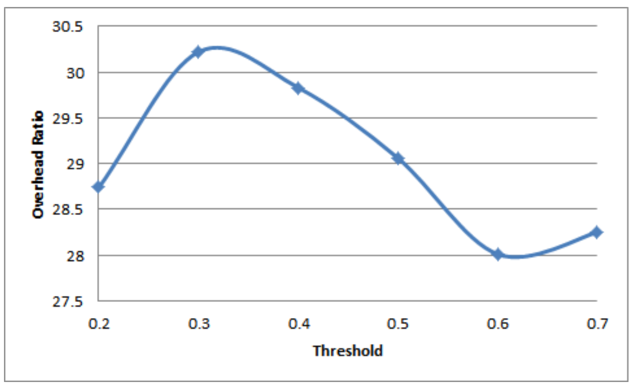

Figure 4.3: Overhead ratio under varying threshold values using AEHBPR under the CMM Model.

other hand, in Fig. 4.3, it is observed that the overhead ratio increases initially up to point 0.3, then decreases progressively till the point 0.6, and then recommence to increase. From these observations, it can be concluded that a threshold of T = 0.6 yields a higher possible number of delivered messages while maintaining the overhead ratio as low as possible. For this reason, we will use T = 0.6 for all our simulation experiments.

4.5.2 Varying Number of Nodes

In this scenario, the number of nodes is varied and the impact of this variation on the number of dead nodes, overhead ratio, and number of delivered messages, of the studied protocols is investigated. The results are captured in Fig 4.4, Fig 4.5, and Fig 4.6 respectively.

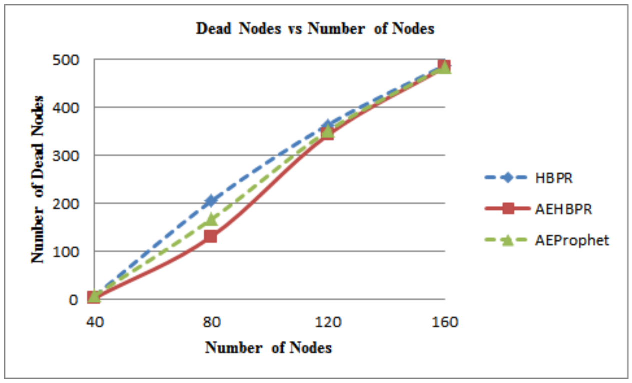

Figure 4.4: Number of dead nodes under varying number of nodes using the CMM model.

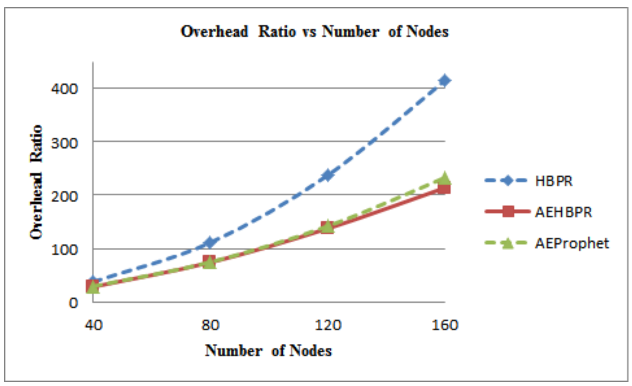

Figure 4.5: Overhead ratio under varying number of nodes using the CMM model.

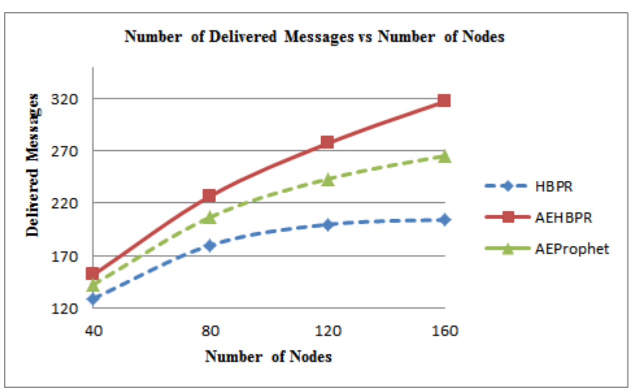

Figure 4.6: Number of delivered messages under varying number of nodes using the CMM model.

In Fig. 4.4, it is observed that the number of dead nodes increases when the number of nodes is increased. In addition, AEHBPR generates about 14.36% (resp. 16.89%) less dead nodes compared to AEProphet (resp. HBPR) when the number of nodes is less than 120. However, when the network size is more than 120, the performance of AEHBPR, AEProphet, and HBPR, are almost similar. In Fig. 4.5, it is observed that the overhead ratio produced by AEHBPR is comparable to that produced by AEProphet. In fact, AEHBPR generates only 5% less overhead ratio than AEProphet, but yields about 37.44% less overhead ratio than that generated by HBPR. Finally, in Fig. 4.6, it is observed that AEHBPR generates about 14.19% (resp. 37.67%) more number of delivered messages in comparison to AEProphet (resp. HBPR).

4.5.3 Varying Message Size

In this scenario, the message size is varied and the impact of this variation on the average remaining energy, number of dead nodes, number of delivered messages, and overhead ratio of the studied protocols is investigated. The results are captured in Fig 4.7, Fig 4.8, Fig 4.9, and Fig 4.10 respectively.

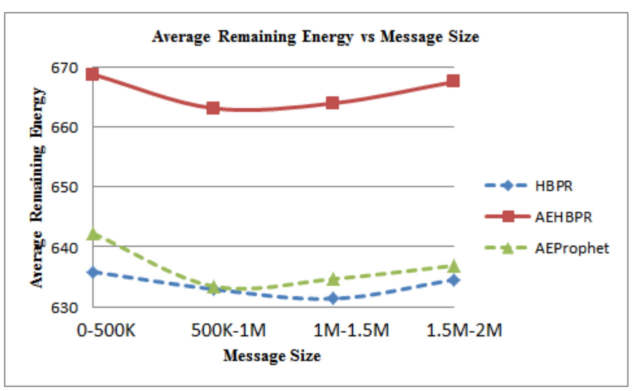

Figure 4.7: Average remaining energy under varying message size using the CMM model.

In Fig. 4.7, it is observed that under varying message size, AEHBPR produces about 4.49% (resp. 5%) more remaining energy than AEProphet (resp. HBPR). Also, Fig. 4.8 shows that AEHBPR generates about 52.17% (resp. 52.82%) less number of dead nodes

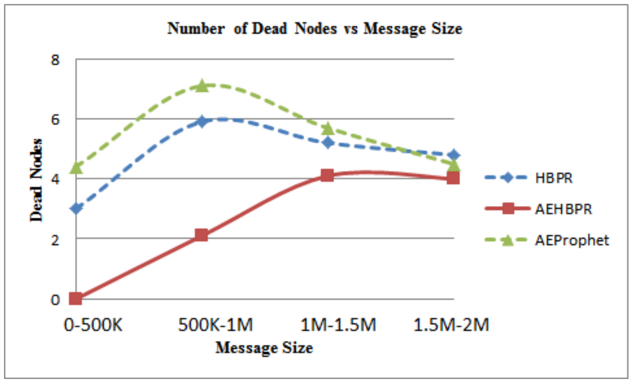

Figure 4.8: Number of dead nodes under varying message size using the CMM model.

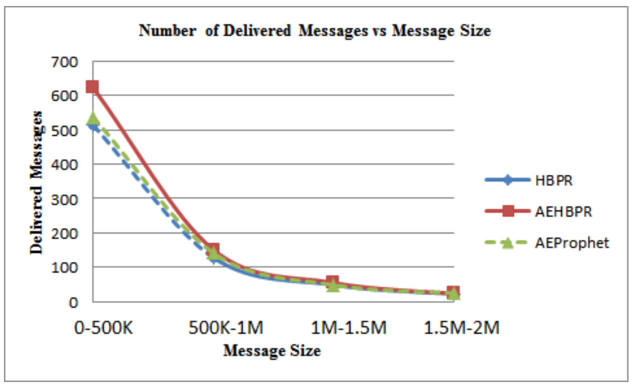

Figure 4.9: Number of delivered messages under varying message size using the CMM model.

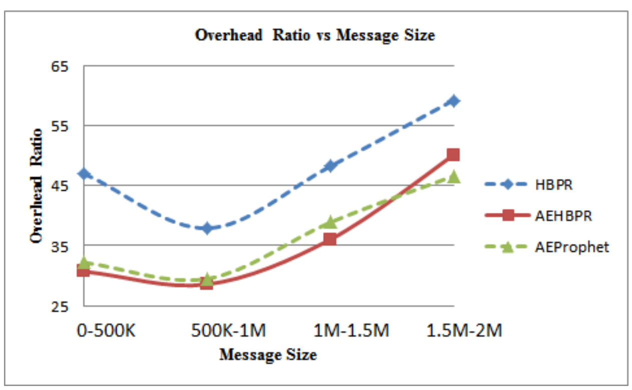

Figure 4.10: Overhead ratio under varying message size using the CMM model.

than that generated by AEProphet (resp. HBPR). However, when the message size is in the range 1.5 M-2 M, AEHBPR and AEProphet yield almost the same performance. In Fig. 4.9, it is observed that when the message size is less than 500K, AEHBPR yields about 14.19%

(resp. 37.67%) more number of delivered messages than AEProphet (resp. HBPR). In all other cases, the performance of the three protocols are almost similar. Finally, Fig. 4.10 shows that AEHBPR yields about 26% less overhead ratio than that generated by HBPR; and there is no significant difference in the performances of AEHBPR and AEProphet.

4.5.4 Varying Message Generation Interval

In this scenario, the message generation interval is varied and the impact of this variation on the overhead ratio, and number of delivered messages of the studied protocols is investigated. The results are captured in Fig 4.11, and Fig 4.12 respectively.

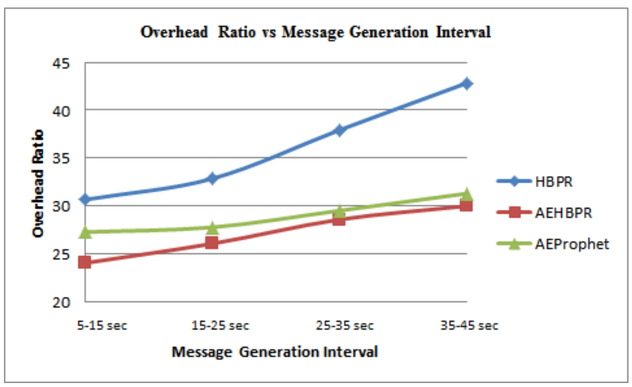

Figure 4.11: Overhead ratio under varying message generation interval using the CMM model.

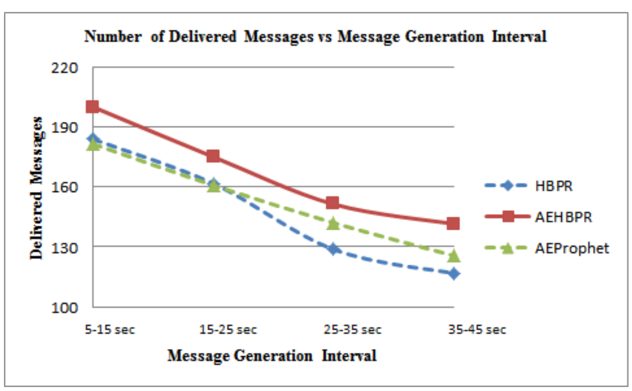

Figure 4.12: Number of delivered messages under varying message generation interval using the CMM model.

In Fig. 4.11, it is observed that AEHBPR yields about 5.2% (resp. 24.29%) less overhead ratio than that generated by AEProphet (resp. HBPR). For larger message generation interval, the performances of AEHBPR and AEProphet are almost similar. Similarly, in Fig. 4.12, it is observed that AEHBPR yields about 9.27% (resp. 13%) more number of delivered messages than AEProphet (resp. HBPR).

4.6 Simulation Results Using Real Mobility Traces

This section describes the results obtained when evaluating the studied routing schemes for OppNets using the real traces dataset obtained from [22].

4.6.1 Varying Message Size Under Real Mobility Traces

In this scenario, the message size is varied and the impact of this variation on the number of dead nodes, overhead ratio of the studied protocols is investigated. The results are captured in Fig 4.13, and Fig 4.14 respectively.

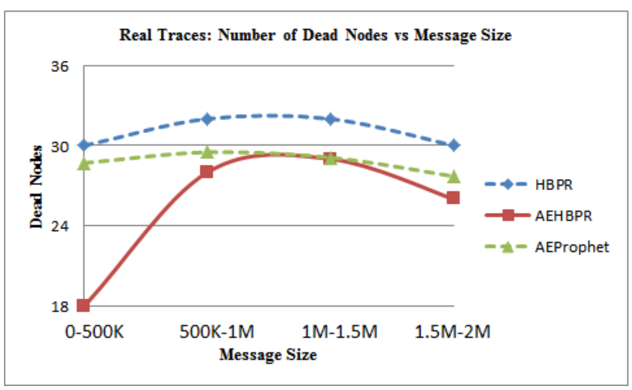

Figure 4.13: Number of dead nodes under varying message size using real mobility traces.

In Fig. 4.13, the performance of AEHBPR in terms of number of generated dead nodes is found to be comparable to that of AEProphet when the message size approaches the range 1M-1.5M. In all other cases, AEHBPR generates about 12.44% (resp. 22.09%) less number of dead nodes compared to AEProphet (resp. HBPR). In Fig. 4.14, it is observed

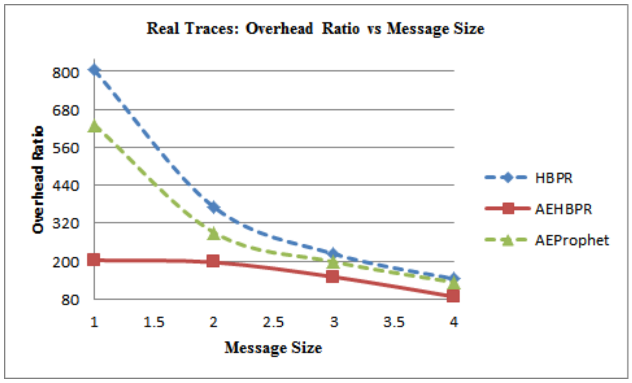

Figure 4.14: Overhead ratio under varying message size using real mobility traces.

that when the message size is increased, the overhead ratio decreases for all protocols. In fact, AEHBPR yields 36.5% (resp. 47.81%) less overhead ratio in comparison to AEProphet (resp. HBPR).

4.6.2 Varying Message Generation Interval Under Real Mobility Traces

In this scenario, the message generation interval is varied and the impact of this variation on the average remaining energy , overhead ratio, and number of delivered messages, of the studied protocols is investigated. The results are captured in Fig 4.15, Fig 4.16, and Fig 4.17 respectively.

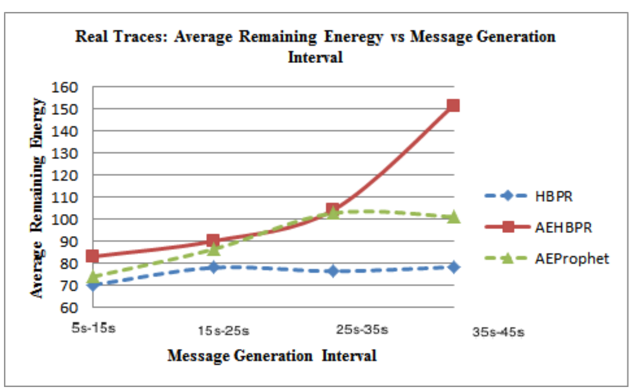

Figure 4.15: Average remaining energy under varying message generation interval using real mobility traces.

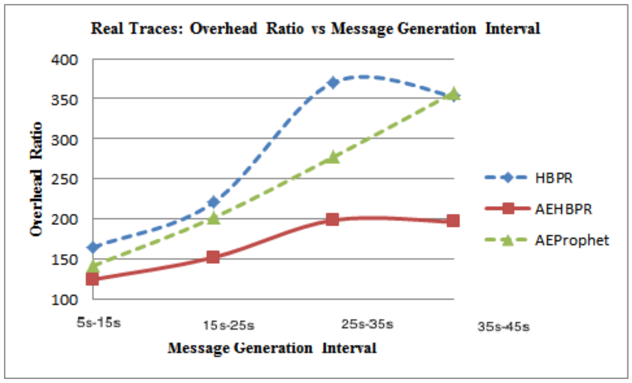

Figure 4.16: Overhead ratio under varying message generation interval using real mobility traces.

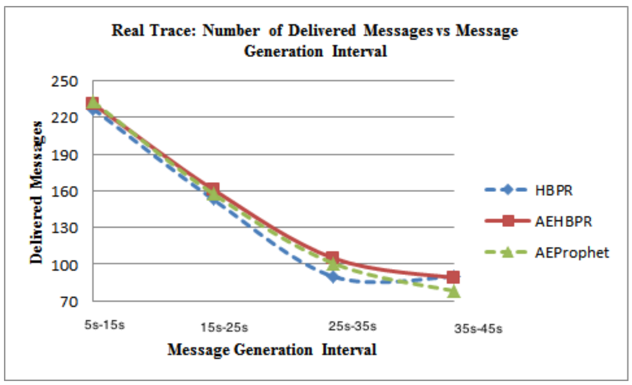

Figure 4.17: Delivered messages under varying message generation interval using real mobility traces.

In Fig. 4.15, the performance of AEHBPR in terms of average remaining energy is found to be comparable to that of AEProphet when the message generation interval is in the range 25 s-35 s. In all other cases, AEHBPR yields about 18.10% (resp. 41.68%) more remaining energy than AEProphet (resp. HBPR). In Fig. 4.16, it is observed that the overhead ratio generated by AEHBPR is much lower than that obtained when using AEProphet and HBPR. In fact, AEHBPR yields 25.5% (resp. 35.94%) less overhead ratio in comparison to AEProphet (resp. HBPR). Finally, in Fig. 4.17, it is observed that the number of delivered messages in all three protocols are almost similar, with AEHBPR generating about 5.42% (resp. 4.6%) more delivered messages than that generated by AEProphet (resp. HBPR) although the differences in the number of delivered messages are quite slow.

The above results disclosing the benefits of our proposed AEHBPR protocol over the AEProphet and HBPR protocols in terms of the studied performance metrics are mainly attributed to the intrinsic one-hop acknowledgment mechanism implemented in AEHBPR, which systematically eliminates the unnecessary redundant messages in the nodes? buffers. Also, the minimum energy threshold factor used in AEHBPR, which helps avoiding the selection of low energy nodes as next hop message forwarders, also contributes to justify some of the above simulation results.

Chapter 5

Conclusion

In this thesis, an energy-efficient version of the HBPR protocol [3] (called AEHBPR) has been proposed, and compared against HBPR and AEProphet, where AEProphet is the ProPHet routing protocol for OppNets on which the same energy-aware constraints have been implemented, which satisfy the following requirements: (1) unnecessary copies of the already delivered messages from the buffers of other nodes carrying these messages are removed, (2) No extra copy of these delivered messages is generated and transmitted in the network, and (3) only the relay nodes that have sufficient energy level are qualified as next hop forwarders. A one-hop acknowledgment mechanism has been introduced to overcome the unnecessary transmission of the already delivered messages. In this mechanism, the destination node is requested to send an acknowledgment to the last intermediate node from which it has received the message (so-called last sender node). This information is flooded into the network so that all other nodes are aware that the message has already been delivered to destination, avoiding its unnecessary retransmission. Our simulations reveal the following:

- Using the CMM model, AEHBPR performs better compared to other protocols in terms of number of dead nodes, average remaining energy, overhead ratio, and number of delivered messages, under varying number of nodes.
- Using the CMM model, AEHBPR performs better compared to other protocols in

terms of average remaining energy, number of dead nodes, number of delivered messages, and overhead ratio, under varying message size

- Using the CMM model, AEHBPR performs better compared to other protocols in terms of average remaining energy, overhead ratio, and number of delivered messages, under varying message generation interval
- Using real mobility traces, AEHBPR performs better compared to other protocols in terms of average remaining energy, number of dead nodes, overhead ratio, and number of delivered messages, under varying message size
- Using real mobility traces, AEHBPR performs better compared to other protocols in terms of average remaining energy, number of dead nodes, overhead ratio, and number of delivered messages, under varying message generation interval

As future work, we plan to compare AEHBPR against other benchmark energy-aware routing protocols for OppNets such as energy-aware Spray-and-Wait, genetic algorithm-based energy-efficient routing, and n-epidemic protocol, to name a few. The AEHBPR protocol can also be experimented using other realistic mobility traces. It is also desirable to make AEHBPR secure, for instance, by implementing a cryptography-based technique that protects the messages before their transmissions and checks the authenticity of the sender of the message.

Appendix A

Source Code of AEHBPR

```
// ==========Algorithm 1 for AEHBPR========================
for (Connection con : getConnections()) {            // Step 1: Select the next neighbour Node (NN)
        DTNHost me = getHost();
        DTNHost other = con.getOtherNode(getHost());
        EEHBPR othRouter = (EEHBPR) other.getRouter();
        if (othRouter.isTransferring()) {                     //Step2: If NN is busy then goto Step1
                continue;
        }
        double nn_energy = (double) othRouter.getHost().getComBus().getProperty(EnergyModel.ENERGY_VALUE_ID);

        for (Message m : msgCollection) {           //Step3: Repeat All message(M) of current node(CN)
                if (othRouter.hasMessage(m.getId())) {  //      3a: if NN has M then goto Step3
                        continue;
                }
                DTNHost dest = m.getTo();
                String key = m.getId()+"<->"+m.getFrom().toString()+"<->"+dest.toString();
                                                /* 3b: check Ack_Table of NN for M
                                                        If Ack_Table of NN has M
                                                                Remove M from buffer of CN
                                                                Update Ack_Table of CN goto step 3 for next M
                                                        end if*/
                if(othRouter.delivered.containsKey(key)) {
                        int cnt = (int)othRouter.delivered.get(key);
                        this.delivered.put(key, ++cnt);
                        msg_to_be_deleted.add(m);
                        continue;
                }
                                                /*3c: If Energy Level of NN < Minimum Energy
                                                                Threshold(MET) and NN is not
                                                                Destination Node(DN) then
                                                        goto step 3
                                                */
                if(nn_energy < this.battery_level_threshold && !dest.equals(other)) {
                        continue;
                }
                                                /*3d: If NN is DN then
```

```
                                                                        //forward Message
                                                        */
                if(dest.equals(other)) {
                        messages.add(new Tuple<Message, Connection>(m, con));
                }
                                                        // Else Follow HBPR to send Message to NN
                else if (calculateMetric(dest, other) > threshold) {
                        messages.add(new Tuple<Message, Connection>(m, con));
                }

        }

}
// ========Algorithm 2 , and 3==================================================================
@Override
public int receiveMessage(Message m, DTNHost from) {
        //======Algorithm 3 starts here ============//
        if(m.getSize() == -1) {   // If message size is -1 then it means its acknowledged message
                String ack_m = m.getId();
                this.delivered.put(ack_m,1);      //update ack_t
                String[] parts = ack_m.split("<->");
                String m_Id = parts[0];
                this.deleteMessage(m_Id, false);  // delete message from the Buffer
                return 0;
        }
        //=======End Algorithm 3 ================//
        int i = super.receiveMessage(m, from);

        //======Algorithm 2 starts here ============//
        if(m.getTo().equals(this.getHost()) && i == RCV_OK ) {
                String ack_m = m.getId()+"<->"+m.getFrom().toString()+"<->"+m.getTo().toString();  //ACK Message
                //Create new message with the size -1 which Indicates that its acknowledged Message
                Message ack_mes = new Message(this.getHost(),from,ack_m,-1);
                from.receiveMessage(ack_mes, this.getHost()); //send ack to the last sender
                        this.delivered.put(ack_m,1);
                }
        //=======End Algorithm 2 ===============//
                return i;
        }
        // ==========================================================================
```

Appendix B

Source Code of AEProphet

```
// ============Algorithm 1 for AEProphet=========================
for (Connection  con : getConnections()) {       // Step 1: Select the next neighbour Node (NN)
        DTNHost me = getHost();
        DTNHost other = con.getOtherNode(getHost());
        ProphetRouter othRouter = (ProphetRouter)other.getRouter();

        if (othRouter.isTransferring()) {        //Step2: If NN is busy then goto Step1
                continue;
        }

        double nn_energy = (double) othRouter.getHost().getComBus().getProperty(EnergyModel.ENERGY_VALUE_ID);
        for (Message m : msgCollection) {        //Step3: Repeat All message(M) of current node(CN)
                if (othRouter.hasMessage(m.getId())) {  //       3a: if NN has M then goto Step3
                        continue;
                }
                DTNHost dest = m.getTo();
                String key = m.getId()+"<->"+m.getFrom().toString()+"<->"+dest.toString();
                                                /* 3b: check Ack_Table of NN for M
                                                                If Ack_Table of NN has M
                                                                        Remove M from buffer of CN
                                                                        Update Ack_Table of CN goto step 3 for next M
                                                                end if*/
                if(othRouter.delivered.containsKey(key)) {
                        int cnt = (int)othRouter.delivered.get(key);
                        this.delivered.put(key, ++cnt);
                        msg_to_be_deleted.add(m);
                        continue;
                }
                                                                /*3c: If Energy Level of NN < Minimum Energy
                                                                                Threshold(MET) and NN is not
                                                                                Destination Node(DN) then
                                                                                goto step 3
                                                                */
                if(nn_energy < this.battery_level_threshold && !dest.equals(other)) {
                        continue;
                }
```

```
                                                        /*3d: If NN is DN then
                                                                //forward Message
                                                        */

                if(dest.equals(other)) {
                                messages.add(new Tuple<Message, Connection>(m, con));
                }
                                                        // Else Follow ProPHet to send Message to NN

                else if (othRouter.getPredFor(m.getTo()) > getPredFor(m.getTo())) {
                                // the other node has higher probability of delivery
                        messages.add(new Tuple<Message, Connection>(m,con));
                }
        }
}

        // ===========================================================================
        // ==========Algorithm 2, and 3=========================
        // ===========================================================================

@Override
public int receiveMessage(Message m, DTNHost from) {

                //======Algorithm 3 starts here ============//
        if(m.getSize() == -1) {   // If message size is -1 then it means its acknowledged message
                String ack_m = m.getId();
                this.delivered.put(ack_m,1);      //update ack_t
                String[] parts = ack_m.split("<->");
                String m_Id = parts[0];
                this.deleteMessage(m_Id, false);  // delete message from the Buffer
                return 0;
        }
                //=======End Algorithm 3 =================//

        int i = super.receiveMessage(m, from);

        //======Algorithm 2 starts here ============//

        if(m.getTo().equals(this.getHost()) && i == RCV_OK ) {
                String ack_m = m.getId()+"<->"+m.getFrom().toString()+"<->"+m.getTo().toString();  //ACK Message
                        //Create new message with the size -1 which Indicates that its acknowledged Message
                Message ack_mes = new Message(this.getHost(),from,ack_m,-1);
                from.receiveMessage(ack_mes, this.getHost()); //send ack to the last sender
                this.delivered.put(ack_m,1);
        }
        //=======End Algorithm 2 ===============//
                return i;
        }
        // ===========================================================================
```

Appendix C

Configuration File

```
#settings for the simulation

## Scenario settings
Scenario.name = AEHbpr_40_testing
Scenario.simulateConnections = true
Scenario.updateInterval = 0.1
# 43200s == 12h
Scenario.endTime = 43200

#my specific settings
EEHBPR.threshold = 0.6
#Battery level threshold in u nits
EEHBPR.battery_level_threshold = 600
EEHBPR.transmissionFactor = 6

# energy settings
Group.initialEnergy = 5000
Group.scanEnergy = 0.1
Group.transmitEnergy = 0.2
Group.scanResponseEnergy = 0.1
Group.baseEnergy = 0.01

#setting to change regular basis
Group.router = EEHBPR
Group.nrofHosts = 40
# range of message source/destination addresses
Events1.hosts = 0,126
Report.granularity =43200

## Interface-specific settings:
# type : which interface class the interface belongs to
# For different types, the sub-parameters are interface-specific
# For SimpleBroadcastInterface, the parameters are:
# transmitSpeed : transmit speed of the interface (bytes per second)
# transmitRange : range of the interface (meters)
```

```
# "Bluetooth" interface for all nodes
btInterface.type = SimpleBroadcastInterface
# Transmit speed of 2 Mbps = 250kBps
btInterface.transmitSpeed = 250k
btInterface.transmitRange = 10

# High speed, long range, interface for group 4
highspeedInterface.type = SimpleBroadcastInterface
highspeedInterface.transmitSpeed = 10M
highspeedInterface.transmitRange = 1000

# Define 6 different node groups
Scenario.nrofHostGroups = 6

## Group-specific settings:
# groupID : Group's identifier. Used as the prefix of host names
# nrofHosts: number of hosts in the group
# movementModel: movement model of the hosts (valid class name from movement package)
# waitTime: minimum and maximum wait times (seconds) after reaching destination
# speed: minimum and maximum speeds (m/s) when moving on a path
# bufferSize: size of the message buffer (bytes)
# router: router used to route messages (valid class name from routing package)
# activeTimes: Time intervals when the nodes in the group are active (start1, end1, start2, end2, ...)
# msgTtl : TTL (minutes) of the messages created by this host group, default=infinite

## Group and movement model specific settings
# pois: Points Of Interest indexes and probabilities (poiIndex1, poiProb1, poiIndex2, poiProb2, ... )
#       for ShortestPathMapBasedMovement
# okMaps : which map nodes are OK for the group (map file indexes), default=all
#          for all MapBasedMovent models
# routeFile: route's file path - for MapRouteMovement
# routeType: route's type - for MapRouteMovement

# Common settings for all groups
Group.movementModel = HumanWalk1
Group.bufferSize = 5M
Group.waitTime = 0, 120
# All nodes have the bluetooth interface
Group.nrofInterfaces = 1
Group.interface1 = btInterface
# Walking speeds
Group.speed = 0.5, 1.5
# Message TTL of 300 minutes (5 hours)
Group.msgTtl = 300

# group1 (pedestrians) specific settings
Group1.groupID = p

# group2 specific settings
Group2.groupID = c
# cars can drive only on roads
Group2.okMaps = 1
# 10-50 km/h
Group2.speed = 2.7, 13.9

# another group of pedestrians
```

```
Group3.groupID = w

# The Tram groups
Group4.groupID = t
Group4.bufferSize = 50M
Group4.movementModel = MapRouteMovement
Group4.routeFile = data/tram3.wkt
Group4.routeType = 1
Group4.waitTime = 10, 30
Group4.speed = 7, 10
Group4.nrofHosts = 2
Group4.nrofInterfaces = 2
Group4.interface1 = btInterface
Group4.interface2 = highspeedInterface

Group5.groupID = t
Group5.bufferSize = 50M
Group5.movementModel = MapRouteMovement
Group5.routeFile = data/tram4.wkt
Group5.routeType = 2
Group5.waitTime = 10, 30
Group5.speed = 7, 10
Group5.nrofHosts = 2

Group6.groupID = t
Group6.bufferSize = 50M
Group6.movementModel = MapRouteMovement
Group6.routeFile = data/tram10.wkt
Group6.routeType = 2
Group6.waitTime = 10, 30
Group6.speed = 7, 10
Group6.nrofHosts = 2

## Message creation parameters
# How many event generators
Events.nrof = 1
# Class of the first event generator
Events1.class = MessageEventGenerator
# (following settings are specific for the MessageEventGenerator class)
# Creation interval in seconds (one new message every 25 to 35 seconds)
Events1.interval = 25,35
# Message sizes (500kB - 1MB)
Events1.size = 500k,1M
# Message ID prefix
Events1.prefix = M

## Movement model settings
# seed for movement models' pseudo random number generator (default = 0)
MovementModel.rngSeed = 1
# World's size for Movement Models without implicit size (width, height; meters)
MovementModel.worldSize = 4500, 3400
# How long time to move hosts in the world before real simulation
MovementModel.warmup = 1000

## Map based movement -movement model specific settings
MapBasedMovement.nrofMapFiles = 4
```

```
MapBasedMovement.mapFile1 = data/roads.wkt
MapBasedMovement.mapFile2 = data/main_roads.wkt
MapBasedMovement.mapFile3 = data/pedestrian_paths.wkt
MapBasedMovement.mapFile4 = data/shops.wkt

## Reports - all report names have to be valid report classes

# how many reports to load
Report.nrofReports = 2
# length of the warm up period (simulated seconds)
Report.warmup = 0
# default directory of reports (can be overridden per Report with output setting)
Report.reportDir = reports/
# Report classes to load
Report.report1 = MessageStatsReport
Report.report2 = EnergyLevelReport

## Default settings for some routers settings
ProphetRouter.secondsInTimeUnit = 30
SprayAndWaitRouter.nrofCopies = 6
SprayAndWaitRouter.binaryMode = true

## Optimization settings -- these affect the speed of the simulation
## see World class for details.
Optimization.cellSizeMult = 5
Optimization.randomizeUpdateOrder = true


## GUI settings

# GUI underlay image settings
GUI.UnderlayImage.fileName = data/helsinki_underlay.png
# Image offset in pixels (x, y)
GUI.UnderlayImage.offset = 64, 20
# Scaling factor for the image
GUI.UnderlayImage.scale = 4.75
# Image rotation (radians)
GUI.UnderlayImage.rotate = -0.015

# how many events to show in the log panel (default = 30)
GUI.EventLogPanel.nrofEvents = 100
# Regular Expression log filter (see Pattern-class from the Java API for RE-matching details)
#GUI.EventLogPanel.REfilter = .*p[1-9]<->p[1-9]$
```

Bibliography

[1] S. K. Dhurandher, S. Misra, M. S. Obaidat, V. Bansal, P. Raj Singh,V Punia, “EEAODR: An energy-efficient ad hoc on-demand routing protocol for mobile ad hoc networks”, International Journal of Communication Systems, Vol. 22, Issue 7, pp. 789–817, July 2009.

[2] V. Stavroulaki, K. Tsagkaris, M. Logothetis, A. Georgakopoulos, P. Demestichas, J. Gebert, and M. Filo, “Opportunistic networks”, IEEE Vehicular Technology Magazine, Vol. 3, No. 6, 2011, pp. 52-59.

[3] S.K. Dhurandher, D.K. Sharma, I. Woungang, S. Bhati, “HBPR: History Based Prediction for Routing in Infrastructure-less Opportunistic Networks”, AINA 2013: 931-936.

[4] Z. Yuanyuan, W. Jie, X. Naixue, L. Deshi, “Energy-efficient routing and rate allocation for delay tolerant networks”, Proceedings of the 32nd IEEE International Conference on Distributed Computing Systems Workshops (ICDCSW), Macau, China, June 18-21, 2012, pp. 260-266

[5] Keränen A., Ott J., Kärkkäinen T., “The ONE simulator for DTN protocol evaluation”, Proceedings of the 2nd International Conference on Simulation Tools and Techniques (SIMUTools' 09), Rome, Italy, Mar. 2-6, 2009, pp. –9.

[6] A. Lindgren, A. Doria, and O. Schelen, “Probabilistic routing in intermittently connected networks”, ACM SIGMOBILE, Mobile Computing and Communications, 2003 Review, Vol. 7, Issue 3, pp. 19-20.

[7] L. Lilien, “A Taxonomy of Specialized Ad Hoc Networks and Systems for Emergency Applications”, Proceedings of the 1st International Workshop on Mobile and Ubiquitous Context Aware Systems and Applications (MUBICA 2007), Philadelphia, Pennsylvania, Aug. 2007, 8 pages (Invited paper).

[8] L. Lilien, Z.H. Kamal, V. Bhuse, A. Gupta, “Opportunistic Networks: The Concept and Research Challenges in Privacy and Security”, Proceedings of NSF International Workshop on Research Challenges in Security and Privacy for Mobile and Wireless Networks (WSPWN 2006), Miami, March 2006, pp. 134–147.

[9] P. Shrivastava, S. Kumar, M. Shrivastava, “Study of mobile ad hoc networks”, International Journal of Computer Applications, Vol. 86, No. 3, 2014 (Online).

[10] K. Scott, S. Burleigh, “Bundle protocol specification”, Internet-Draft, Network Working Group RFC 5050, Nov. 2007, https://tools.ietf.org/html/rfc5050 (Last visited March 19, 2016)

[11] J. Segui, E. Jennings, “Delay tolerant networking bundle protocol simulation”, Proceedings of the 2nd IEEE International Conference on Space Mission Challenges for Information Technology (SMC-IT), Pasadena, CA, USA, DOI: 10.1109/SMC–IT.2006.29, July 17-20, 2006.

[12] K. Fall, “A Delay Tolerant Networking Architecture for Challenged Internets”, Proceedings of SIGCOMM 2003, Aug. 2003, pp. 27-34

[13] D. Pan, Z. Ruan, N. Zhou, X. Liu, Z. Song, “A comprehensive integrated buffer management strategy for opportunistic networks”, EURASIP Journal on Wireless Communications and Networking, 2013: 103 (2013), http://jwcn.eurasipjournals.com/content/2013/1/103 (Last accessed March 19, 2016)

[14] I. Woungang, M.K. Denko, “Credit-based Cooperation Enforcement Schemes Tailored to Opportunistic Networks”, Chapter 3 in: M. Denko (Eds.), Mobile Opportunistic

Networks: Architectures, Protocols and Applications, CRC Press, Taylor & Francis Group, Boca Raton, FL, USA, ISBN: 978-142-008-812-0, May 2011, pp. 51-82.

[15] P. Pirozmand, G. Wu, B. Jedari, F. Xia, "Human mobility in opportunistic networks: Characteristics, models and prediction methods", Journal of Network and Computer Applications, Vol. 42, June 2014, pp. 45-58.

[16] P. Hui, A. Chaintreau, J. Scott, R. Gass, J. Crowcroft, C. Diot, "Pocket Switched Networks and Human Mobility in Conference Environments", Proceedings of the 2005 ACM SIGCOMM Workshop on Delay-tolerant Networking, New York, NY, USA, 2005, pp. 244-251.

[17] Autonomic Network Architecture Project, http://www.ana-project.org (Last visited March 19, 2016).

[18] M. Xiao, J. Wu, L. Huang, "Community-Aware Opportunistic Routing in Mobile Social Networks", IEEE Transactions on Computers, Vol. 63, No. 7, July 2014, pp. 1682-1695.

[19] A. K. Saha, D. Johnson, "Modelling mobility for vehicular ad hoc networks", Proceedings of the ACM Workshop on Vehicular Ad Hoc Networks (VANET), Philadelphia, PA, USA, July 2004, pp.91–92

[20] C. Bettstetter, "Smooth is better than sharp: a random mobility model for simulation of wireless networks", Proceedings of ACM MSWiM, Rome, Italy, July 2001, pp. 19–27

[21] A. Chaintreau, P. Hui, J. Crowcroft, C. Diot, R. Gass, J. Scott, "Impact of human mobility on opportunistic forwarding algorithms", IEEE Transactions on Mobile Computing, Vol. 6, Issue 6, April 2007, pp. 606-620

[22] J. Scott, R. Gass, J. Crowcroft, P. Hui, C. Diot, A. Chaintreau, "The Cambridge/Haggle dataset (v.2009-05-29)", http://crawdad.org/cambridge/haggle/20090529/imote (Last visited March 19, 2016)

[23] V. Borrel, F. Legendre, M. D. de Amorim, S. Fdida, “SIMPS: Sociological Interaction Mobility for Population Simulation”, IEEE/ACM Transactions on Networking, Vol. 17, No. 3, June 2009, pp. 1-12.

[24] C-A. La, P. Michiardi, “Characterizing User Mobility in Second Life”, Technical Report. RR08212, Institut EURECOM, Sophia Antipolis, France, 2008.

[25] A. Keranen, T. Karkkainen, J. Ott, “Simulating Mobility and DTNs with the ONE”, Journal of Communications, Vol. 5, No. 2, February 2010

[26] S. Trifunovic, “Preparing Opportunistic Networks for Take-Off: From Fair Energy Usage to Secure Content Dissemination”, Master of Science, DISS. ETH No. 22771, DISS. TIK No. 154, ETH Zurich, Switzerland, 2015.

[27] V.F.S. Motaa, F. D. Cunhaa, D. F. Macedoa, J. M.S. Nogueiraa, A.A.F. Loureiroa, “Protocols, mobility models and tools in opportunistic networks: A survey”, Computer Communications, Vol. 48, July 15, 2014, pp. 5–19.

[28] S. Palazzo, R. Scopigno, M. Spirito, I. Korpeoglu, “DR.11.1. State of the art of research on opportunistic networks, and definition of a common framework for reference models and performance metrics”, Report EU Deliverable, 216715 NEWCOM++, DiVA Portal: diva2:592083, 21 pages.

[29] S. Basagni, M. Conti, S. Giordano, I. Stojmenovic, “Applications in Delay-Tolerant and Opportunistic Networks”, Wiley-IEEE Press, Doi: 10.1002/9781118511305.ch9, Mar. 4, 2013, pp. 315-359

[30] J. Scott, P. Hui, J. Crowcroft, C. Diot, “Haggle: A networking architecture designed around mobile users”, Proceedings of the 3rd Annual IFIP Conference on Wireless On-Demand Network Systems and Services (WONS 2006), Jan. 18-20, 2006, Les Mnuires, France, pp. 1-9

[31] M. Martonosi, “ZebraNet and Beyond: Applications and Systems Support for Mobile, Dynamic Networks”, Keynote Talk, CASES ’08, Oct. 19–24, 2008, Atlanta, Georgia, USA, ACM 978–1–60558–469–0/08/10.

[32] A. A. Hasson, R. Fletcher, A. Peatland, “DakNet: A Road to Universal Broadband Connectivity”, Wireless Internet UN ICT Conference Case Study, pp. 1–9, 2003. http://courses.media.mit.edu/2003fall/de/DakNet-Case.pdf (Last visited March 19, 2016)

[33] S. Guo, M. Derakhshani, M. H. Falaki, U. Ismail, R. Luk, E.A. Oliver, S. Ur Rahman, A. Seth, M. A. Zaharia, S. Keshav, “Design and Implementation of the KioskNet System”, Computer Networks, Vol. 55, Issue 1, Jan. 7, 2011, pp. 264-281.

[34] V. Lenders, G. Karlsson, and M. May, “Wireless Ad Hoc Podcasting”, Proceedings of the 4th Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks, Vol. 12, No. 1, pp. 65-67, 2007

[35] M. Pitkanen, M. Conti, A. Passarella, S. Giordano, S. Trifunovic, K. Hummel, and M. May, “SCAMPI : Service Platform for Social Aware Mobile and Pervasive Computing Categories and Subject Descriptors”, ACM SIGCOMM Computer Communication Review, 2012, pp. 7-12.

[36] Z.H. Kamal, A. Gupta, L. Lilien, Z. Yang, “The MicroOppnet Tool for Collaboration Experiments with Class 2 Opportunistic Networks”, Proceedings of the 3rd International Conference on Collaborative Computing: Networking, Applications and Work sharing (CollaborateCom 2007), White Plains, New York, Nov. 2007, 10 pages.

[37] L-J. Chen, C. Hung Yu, C. Tseng, H. Chu, C. Chou, “A Content-Centric Framework for Effective Data Dissemination in Opportunistic Networks”, IEEE Journal on Selected Areas in Communications, Vol. 26, Issue 5, June 2008, pp. 761-772.

[38] Z. Zhang, “Routing in Intermittently Connected Mobile Ad Hoc Networks and Delay Tolerant Networks: Overview and Challenges”, IEEE Communications Surveys and Tutorials, Vol. 8, Issue 1, 2006, pp. 24-37.

[39] S. Jain, K. Fall, and R. Patra, “Routing in a Delay Tolerant Network”, Proceedings of the ACM SIGCOMM 2004, Aug.30-Sept.3, 2004, pp. 145-158.

[40] S.K. Dhurandher, D.K. Sharma, I. Woungang, S. Gupta, S. Goyal, “Routing Protocols in Infrastructure-based Opportunistic Networks”, Chapter 5, Routing in Opportunistic Networks, I. Woungang, S. K. Dhurandher, A. Anpalagan, T. Vasilakos (Eds), Springer Verlag, USA, ISBN 978-1-4614-3514–3, June 2013, pp. 125-144.

[41] S.K. Dhurandher, D.K. Sharma, I. Woungang, S. Bhati, “Routing Protocols in Infrastructure-less Opportunistic Networks”, Chapter 13, Routing in Opportunistic Networks, I. Woungang, S. K. Dhurandher, A. Anpalagan, T. Vasilakos (Eds), Springer Verlag, USA, ISBN 978-1-4614-3514-3, June 2013, pp. 353-382.

[42] D.J. Goodman, J. Borras, N.B. Mandayam, R. D. Yates, “Infestations: A New System Model for Data and Messaging Services”, Proceedings of the IEEE Vehicular Technology Conference (VTC), Vol. 2, May 1997, pp. 969-973.

[43] W. Zhao, M. Ammar, E. Zegura, “A Message Ferrying Approach for Data Delivery in Sparse Mobile Ad Hoc Networks”, Proceedings of the 5th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc’04), ACM Press, Tokyo, Japan, May 24-26, 2004, pp. 187–198.

[44] S. Jain, R. C. Shah, W. Bbrunette, G. Borriello, S. Roy, “Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks”, ACM/Kluwer Mobile Networks and Applications (MONET), Vol. 11, No. 3, June 2006, pp. 327–339.

[45] A. Vahdat, and D. Becker, "Epidemic routing for partially connected ad hoc networks", Technical Report CS-2000-06, Dept. of Computer Science, Duke University, Durham, NC, 2000.

[46] T. Spyropoulos, K. Psounis and C. S. Raghavendra, "Spray and Wait: An Efficient Routing Scheme for Intermittently Connected Mobile Networks", Proceedings of the SIGCOMM Workshop on Delay-Tolerant Networking, PA, USA, Aug. 22-26, 2005, pp. 252–259.

[47] C. Boldrini, M. Conti, I. Iacopini and A. Passarella, "HiBOp: a History Based Routing Protocol for Opportunistic Networks", Proceedings of the IEEE International Symposium on World of Wireless, Mobile and Multimedia Networks, 2007 (WoWMoM), Espoo, Finland, June 18–21, 2007, pp. 1-12.

[48] S.K. Dhurandher, I. Woungang, A. Rajendra, P. Ghaie, P. Chatzimisios, "A Centrality-based ACK Forwarding Mechanism for Efficient Routing in Opportunistic Networks", (Accepted Sept. 17, 2015), Intl. Conference on CYber physiCaL systems, IoT and sensors Networks (Cyclone 2015), Rome, Italy, Oct. 26, 2015. In Press.

[49] S.K. Dhurandher, D.K. Sharma, I. Woungang, A. Saini, "Efficient Routing based on Past Information to Predict the Future Location for Message Passing in Infrastructure-less Opportunistic Networks", Journal of Supercomputing, 71(5): 1694-1711, 2015.

[50] L. Xiaofeng, H. Pan, "An energy-efficient n-epidemic routing protocol for delay tolerant networks", Proceedings of the IEEE 5th International Conference on Networking, Architecture and Storage (NAS), Macau, China, July 15–17, 2010, pp. 341-347

[51] S. Gao, L. Zhang, H. Zhang, "Energy-aware spray and wait routing in mobile opportunistic sensor networks", Proceedings of the 3rd IEEE International Conference on Broadband Network and Multimedia Technology (IC-BNMT), Beijing, China, Oct. 26–28, pp. 1058-1063, 2010.

[52] V. G. Patel, T. K. Oza, D. M. Gohil, “Vibrant energy aware spray and wait routing in delay tolerant network”, Journal of Telematics and Informatics, Vol. 1, No. 1, pp. 43-47, 2013.

[53] C. Chilipirea, A. Petre, C. Dobre, “Energy-aware social-based routing in opportunistic networks”, Proc. of 27th IEEE WAINA 2013, Barcelona, Spain, March 25–28, 2013, pp. 791-796.

[54] S. K. Dhurandher , D. K. Sharma, I. Woungang, R. Gupta, S. Gupta, “GAER: Genetic Algorithm based Energy-efficient Routing Protocol for Infrastructure-less Opportunistic Networks”, Journal of Supercomputing, Springer, Vol. 69, Issue 3, Sept. 2014, pp. 1183-1214.

[55] Y. K. Yao, W. H. Liu, W. X. Zheng, Z. Ren, “An energy-saving routing algorithm for opportunistic networks based on asynchronous sleep approach”, Applied Mechanics and Materials, Vol. 441, 2014, pp. 1001-1004.

[56] H. Wennerstromy, C. Rohnery, D.B. Smith, “Considering multi-contact encounters in opportunistic networks”, In Proceedings of the 10th ACM MobiCom Workshop on Challenged Networks (CHANTS' 15), Paris, France, Sept. 7-11, 2015, pp. 13-18.

[57] P. Hui, J. Crowcroft, E. Yoneki,“Bubble rap: socialbased forwarding in delay tolerant networks”, In Proceedings of the 9th ACM International Symposium on Mobile Ad Hoc Networks and Comp. MobiHoc’08), pp 241-250, New York, USA, 2008.

[58] Mao, Shiwen , “Fundamentals of Communication Networks”, Cognitive Radio Communications and Networks, pp. 201-234, doi:10.1016/B978-0-12-374715-0.00008-3, ISBN 9780123747150, 2010.