Energy Efficient Opportunistic Connectivity for Wireless Sensor Network

Sivakumar Sivaramakrishnan

School of Engineering,

Auckland University of Technology

A thesis submitted to Auckland University of Technology in fulfilment of the requirements for the degree of

Doctor of Philosophy

April, 2013

I, Sivakumar Sivaramakrishnan, hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning.

Abstract

This thesis provides a theoretical analysis of the effects of mobility, node density and a limited transmission range on the connectivity of a varying density of nodes in wireless sensor networks. Connectivity in cellular networks has the advantage of a fixed centralised infrastructure that can provide wide communication coverage. Wireless sensor networks, on the other hand, have a limited range. This limited range, coupled with nodes' mobility, often results in network holes. As the architecture is de-centralised, there is no central node that monitors the nodes' joining or leaving the network. The challenge of identifying these nodes, which is due to their dynamic nature of movement, is presented here.

Opportunistic connectivity addresses the challenge of providing connectivity to isolated mobile nodes. This is through the process of discovery of regions where good density of network nodes are available. The concept involves four key components. These are adaptive sampling, coverage, handoff and directional communication. These act on the minimisation of energy cost incurred with the discovery of related nodes and establishment of connectivity in the network. The window of time for communication is extended in an energy–efficient manner through coverage, handoff and direction for such delay–tolerant networks. The overall contribution of this thesis is a protocol design for opportunistic connectivity, its implementation and analysis, with reference to the conservation of energy and reduction of packet drops, in conjunction with protocol testing on an application scenario.

The thesis is structured into seven chapters. The first two chapters provide the background and the literature analysis. The third chapter deals with systems and tools which are used for the modelling and testing. It gives an insight into the different available tools and their ability to validate the parameter of our concept of an opportunistic connectivity protocol. Subsequently, the thesis discusses the design of the 'adaptive Energy COnscious DElay Tolerant OpportUnistic Routing' (ECO-DETOUR) protocol for such delay–tolerant networks in chapter four, as a four stage process involving adaptive sampling, coverage, direction and handoff. Design of the protocol is followed by implementation in chapter five, which was performed using the OPNET and MATLAB environments. The chapter details the different conditions in which each of the four parameters are triggered and discusses the implementation of each of the four parameters as pseudo-code. Finally in chapter six the protocol is tested on a wildlife application scenario. The effectiveness of the protocol is measured in relation to the energy saved and the reduction in number of packet drops achieved under different mobility conditions. Results show that ECO-DETOUR achieves a 45% - 60% reduction in expended energy to set up communication and exchange data packets. The bulk of the saving in energy by the ECO-DETOUR protocol comes from adaptive sampling which is followed by coverage, handoff and direction.

Acknowledgements

Vakratunda mahakaya surya koti samaprabha,
Nirvighnam kurume deva sarva karyeshu sarvada.
This Sanskrit invocation is done to start everything important.
In-essence it means *'wade away all troubles and help achieve the goal in good stead.'*

First and foremost, I thank my primary supervisor Professor Adnan Al-Anbuky whose constant guidance, motivation and support helped me complete this research work towards my Ph.D. His valuable mentoring and constant motivation helped me sail smoothly through tough and easy times. The weekly progress meetings and brainstorming sessions helped me stay focused and progress in my work. His thorough proof-reading of my drafts, prompt responses and raising the bar, helped me improve and do better over the course of study.
I take this opportunity to thank my secondary supervisor Dr. Barbara B.Breen whose valuable comments helped me improve my documentation during various phases of study. I thank Ms Mary Bateman for proof-reading this thesis.

The 'Guru' or the teacher, mentors and shows the path to enlightenment but without the support and blessing of the family, it is virtually impossible to achieve success. My parents, who are the source of inspiration, have always stood by my side. Their constant encouragement and guidance has enabled me to progress in my studies. My father taught me the four 'P's' Plan, Patience, Persistence and Perseverance which gave me the strength to move on during trying times. I found my beautiful, loving and caring wife *Divya* and a loving extended family (my in-laws) during the course of my study. My wife's un-relenting faith and belief in me, gave me the confidence to keep moving on.

My grandparents and *Abima* have prayed for my success and this thesis is a tribute to them. I take this opportunity to extend a special thanks to: my wife, father *K.Sivakumar*, mother *K.Hemalatha* and in-laws, father-in-law *N.Balasubramanian*, mother-in-law *Prema Balasubramanian*, uncle *K.Ranganathan*, aunt *R.Vathsala*, my brother *S.Shriram* and sister *R.Abhaya*, for providing me with a strong support which helped me undertake this doctoral journey and successfully prepare this thesis.

This Thesis is dedicated to my Grandparents.

Paternal Grandfather *Ramaswamy Krishnamurthy* and
Grandmother *Krishnamurthy Parvati*

and

Maternal Grandfather *Ramaswamy Krishnamurthy* and
Grandmother *Krishnamurthy Sethu*.

Contents

List of Figures

List of Tables

List of Algorithms

List of Symbols and Acronyms

B_r : Baud rate
C : Frequent visit count of the mobile node
C_n : n^{th} Frequent visit count value
C_D : Decrementing counter
Cr : Coverage radius of a node
d : Difference of distance between hot spot and current position
d_c : Current distance from origin
d_{c0} : Past value of d_c
d_{hs} : Distance to hot spot from origin
δd_{hs} : Tolerance to deviations from the trajectory path to the hot spot
δ_θ : Tolerance to heading
D_p : Data packet size
E_{bit} : Energy required to transmit 1 bit
E_{ee} : Energy expense for adaptive sampling
f_{vs} : Frequency of velocity polling
G_t : Transmitter gain
G_r : Receiver gain
H_s : Hot spot, regions with high density of nodes
H_{sn} : n^{th} Hot spot
H_n : Number of nodes needed for Handoff
H_{na} : Number of nodes available for Handoff
M_n : Mobile node
N_{bb} : Number of bits for transmission
N_s : Stationary nodes
θ : Heading of the node

θ_m : Heading of the mobile node
θ_{si} : Heading of the stationary node
θ_t : Heading at time t
θ_{t-1} : Heading at time t-1
P_t : Transmission power
R : Transmission radius given by Friis equation
S_a : Adaptive Sampling for nodes in the network
S_m : Receiver sensitivity
t_{ep} : Time elapsed since last measurement of heading or velocity
t_t : Current time
t_{t-1} : Past instance of time
t_{sa} : Time to sample by transmitting beacons
t_w : Window of time of connectivity
t_{wr} : Required window of time of connectivity
Tr : The mobile node trajectory
Tr_m : m^{th} sub-trajectory segment
Tr_{mn} : Trajectory map for the n^{th} Hot spot
v : Relative velocity of the node
v_t : Velocity at time t
v_{t-1} : Velocity at time t-1
x_t : Current position on the x axis of a Euclidean space
x_{t+1} : Future position on the x axis of a Euclidean space
y_t : Current position on the y axis of a Euclidean space
y_{t+1} : Future position on the y axis of a Euclidean space
λ : Wavelength
η_t : Ratio of overhead transmission to data transmissions

ACQUIRE: *ACtive QUery forwarding In sensoR nEtworks*
ANSI: *American National Standards Institute*
AODV: *Ad − hoc On Demand distance Vector routing*
APTEEN: *Adaptive Threshold sensitive Energy Efficient sensor Network*

ART1: *Adaptive Resonance Theory*
ATEMU: *AVR processor based Sensor Network Emulator /Simulator /Debugger*
CAR: *Context − aware Adaptive Routing*
CADR: *Constrained Anisotropic Diffusion Routing*
CDMA: *Code Division Multiple Access*
DD: *Direction Diffusion*
ECO-DETOUR: *adaptive Energy COnscious DElay Tolerant OpportUnistic Routing*
EBR: *Encounter Based Routing*
EPFL: *Ecole Polytechnique Fédérale de Lausanne*
FSM: *Finite State Machine*
GAF: *Geographic Adaptive Fidelity*
GBR: *Gradient Based Routing*
GEAR: *Geograhic and Energy Aware Routing*
GloMoSim: *Global Mobile Information Systems Simulation Library*
GOAFR: *The Greedy Other Adaptive Face Routing*
GSM: *Global System of Mobiles*
HPAR: *Hierarchical Power − Aware Routing*
J-Sim: *Java based Network Simulator*
LEACH: *Low Energy Adaptive Clustering Hierarchy*
LAN: *Local Area Network*
MCFA: *Minimum Cost Forwarding Routing*
NAM: *Network Animator for NS − 2*
NED: *Network description file for OMNET++*
NFS: *Network File System*
NN: *Neural Network*
NS2: *Network Simulator* 2
NS3: *Network Simulator* 3
ORBIT: *Open access research testbed*
OSI: *Open Systems Interconnections model for defining the layers of communication*

PEGASIS: *The Power − Efficient GAthering in Sensor Information Systems*
PRoPHET: *Probabilistic Routing in Intermittently Connected Networks*
PSN: *Pocket Switched Networks*
RR: *Rumor Routing*
SCADDS: *Scalable Coordination Architectures for Deeply Distributed Systems*
SENS: *A Sensor, Environment and Network Simulator*
SOM: *Self Organising Map*
SOP: *Self Organizing Protocol*
SPIN: *Sensor Protocol for Information via Negotiation*
SWANS: *Scalable Wireless Ad − hoc Network Simulator*
Tcl: *Tool command language*
TEEN: *Threshold sensitive Energy Efficient sensor Network*
TORA: *Temporally Ordered Routing Algorithm*
TOSSIM: *TinyOS Simulator*
TTL: *SensorNodes′time to live*
TWIST: *TKN Wireless Sensor Network Testbed*
VGA: *Virtual Grid Architecture Routing*
WLAN: *Wireless Local Area Network*
WSN: *Wireless Sensor Network*
WISEBED: *Wireless Sensor Network Testbed*
WUSTL: *Washington University in St. Louis testbed*
WINLAB: *Wireless Information Network Laboratory*

Chapter 1

Introduction

1.1 Introduction: Wireless Sensor Networks

Wireless Sensor Network (WSN) is a network of small wireless sensor devices that are used for sensing and monitoring the environment. Applications such as animal tracking, vehicle monitoring, wildfire monitoring, to name a few, require such units that perform sensing, data processing, localisation and communication of such information. These devices are characterised by their ability to work independently and collectively to maximise the information gathered on their operational space. Multiple sensing devices are required to perform such monitoring task. As the objects that are monitored may not necessarily remain stationary, freedom of mobility has to form part of such networks.
Wireless Sensor Networks unlike GSM, CDMA, 3G and now 4G networks, are characterised by low energy consumption, short communication range, and low bandwidth. They deal with small sensor data packets. The infrastructure cost of setting up a network like 3G and 4G runs into millions of dollars and there is an ongoing maintenance cost. Moreover, these networks have a limited presence in rural and under-developed areas.

Wireless Sensor Network nodes are battery powered, small and inexpensive. They are meant to be deployed in large numbers. These do not require any infrastructure to be set up and are deployed in hundreds and thousands when used to cover the entire region of interest.

Large scale infrastructural networks have high mounted antennas and transmit high power to cover a wide region. As wide coverage is provided by such networks, mobile nodes are able to easily move without any disruption to connections. Mobility in a sensor network is a challenge because the coverage of nodes is small. Addressing the sensing requirements for a large area of interest, mobility of nodes leads to creation of network holes (regions of no connectivity). Network holes require the mobile nodes to perform node discovery and establish connection. Beacons that are transmitted for node discovery, consume battery power. Moreover, transmission of beacons does not contribute to any exchange of data. Therefore these transmissions result in a complete waste of battery power. This thesis focuses on the development of ECO-DETOUR 'adaptive Energy COnscious DElay Tolerant OpportUnistic Routing' protocol for a sparse sensor network with mobile node.

1.2 Background

Sensor Network nodes have short range and collect sensing information of the environment that are small data packets. The main focus of research for wireless sensor networks has been the forwarding of data from the source[1] to the sink[2]. Research has looked into the topology of the network, sensor node density and deployment of sensor network to forward data packets. As the sensor network nodes rely on battery power, which is limited, energy efficiency of transmission has been an important research area. This section reviews these areas that was a motivation to develop a new protocol.

1.2.1 Wireless Sensor Network Deployment, Topology and Protocol

Data forwarding in sensor networks has relied on the topology[3] of the network. This section details different topology and the protocols that suit them.

[1]sensor node where data originates
[2]The final destination of the data
[3]Network topology is the organization of the nodes in the network

1.2.1.1 Sensor node deployment

Deployment pattern of sensor network nodes is varied with application. Applications like monitoring crops or building management have static nodes placed at points of interest and their location doesn't change. Other applications like nodes used for monitoring animals (nodes are mounted on animals), vehicular networks, patient monitoring, etc., have mobile nodes. The networks for such applications have varying density of nodes as shown in Figure 1.1 within the region of operational coverage. As an example, for animal monitoring applications, the social pattern of animals will determine the distribution pattern of nodes in the network. The density of nodes for vehicular networks is affected by the traffic on the road. Patient mobility affects the connectivity amongst nodes. The knowl-

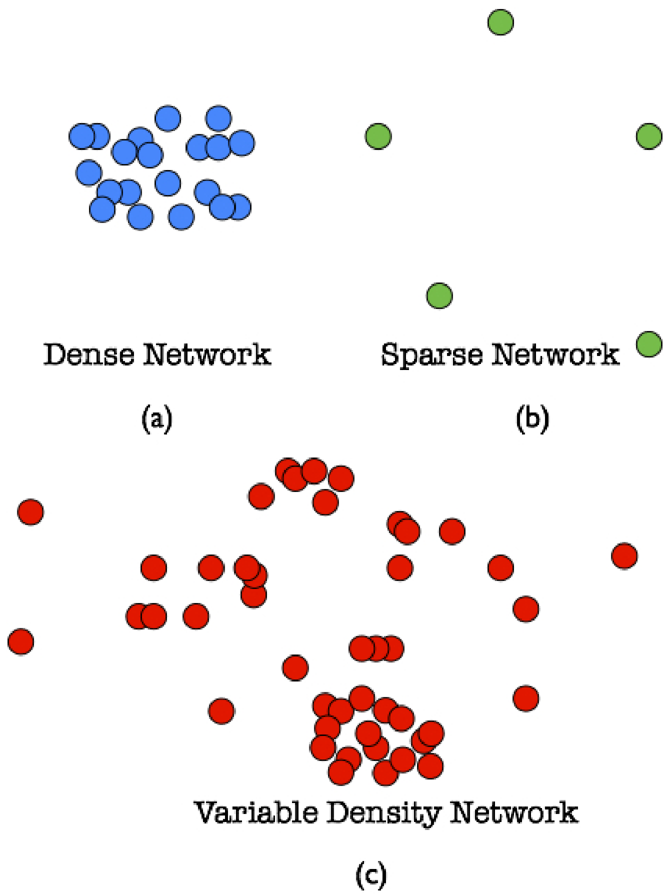


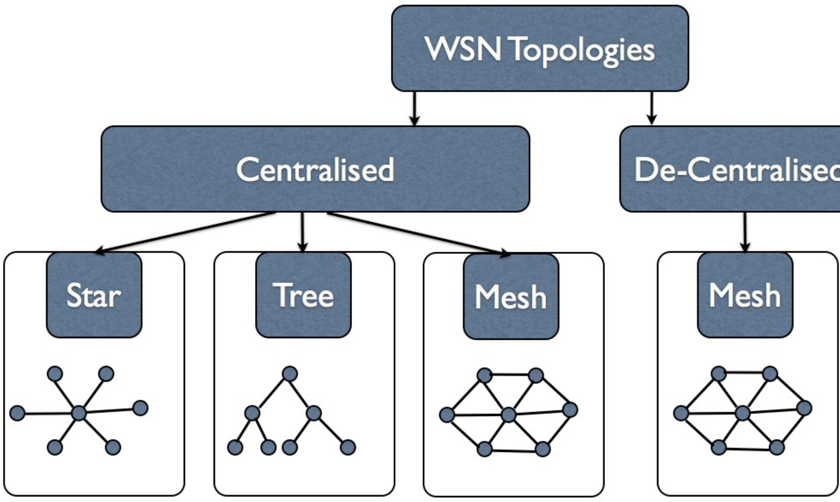
Figure 1.1: Varying Node Density: (a) Dense Network. (b) Sparse Network. (c) Variable Density Network.

edge of node distribution is important as it helps in the process of node discovery and data exchange. In networks with a varying density of nodes, some nodes are in coverage of each other and others, are not. The later results in holes within the network. Mobility of nodes further results in limited duration of connectivity amongst the nodes.

1.2.1.2 Network Topologies

Wireless networks can be broadly classified as either 'Centralised' or 'De-centralised' as shown in Figure 1.2. Centralised networks have a co-ordinator that schedules, the sleep-wake pattern and data exchange amongst nodes. De-centralised networks on the other hand, communicate on a peer-to-peer method, exchanging data with the next hop. The star and tree networks are completely centralised. The mesh network could be co-ordinated using a central co-ordinating node or in a peer-to-peer manner. The former being categorised as a centralised network and the latter a decentralised network.

A further classification of the network is 'Hierarchical' and 'Flat'. Centralised networks are essentially Hierarchical. Hierarchical networks have nodes in a hierarchy specified by the function performed by the nodes in that level. These networks have a cluster head on the top level that collects and processes all information. The next level is, hop nodes or routers, that only pass on the information to the cluster head from the sensing nodes. The end device or sensing layer are a set of nodes that perform the sensing operation and push it to the cluster head for processing. Flat networks on the other hand have nodes performing all the functions, sensing, clustering and routing. Both set of networks have their own advantages and disadvantages but decentralised-flat network provide a degree of freedom with respect to the node deployment and node mobility. This is ideal for wireless sensor networks that are meant to form a network on the fly and not be restricted to specific positioning of nodes.



Dependence on a central node	No dependence on any node
Limited mobility	No restriction on mobility
Node death results in creation of network holes	Multiple paths available between source and destination
Suitable for a planned deployment as node positions are fixed	Suitable for random deployment as network forms on the fly

Figure 1.2: Network Topologies

1.2.1.3 Wireless Sensor Network Routing Protocols

Data exchange between source and sink requires routing if they are not in direct coverage range of each other. Multiple wireless sensor network protocols have been designed. Table 1.1 lists a few popular protocols that are categorised based on the topology: Flat [1–9], Hierarchical [10–17] and location–based routing [18–20]. The location–based routing protocols assume the nodes know their location and the data exchange takes place in the geographical vicinity. Most of the developed protocols assume limited mobility, good connectivity and a definite region of deployment. Based on these assumption, the designed protocols get limited to a specific region of deployment that are well connected and have redundant routes between source and destination. As these routing protocols are developed based on the requirements of the application they are highly optimised for a par-

Table 1.1: Wireless Sensor Network Routing Protocols

Topology	Protocol	Mobility	Data delivery Model
Flat [1]	SPIN (Sensor Protocol for Information via Negotiation)	Mobility is Possible	Event driven
Flat [2]	DD (Directed Diffusion)	Limited Mobility	Demand driven
Flat [3]	RR (Rumor Routing)	Limited Mobility	Demand driven
Flat [4]	MCFA(Minimum Cost Forwarding Algorithm)	No Mobility	Broadcast to next hop neighbour
Flat [5]	GBR(Gradient Based Routing)	Limited Mobility	Hybrid
Flat [6]	CADR(Constrained Anisotropic Diffusion Routing)	No Mobility	Continuous
Flat [7]	ACQUIRE (ACtive QUery forwarding In sensoR nEtworks)	Limited Mobility	Query based
Flat [8]	AODV (Ad-hoc On Demand distance Vector Routing)	Limited Mobility	Demand driven
Flat [9]	CAR (Context Aware Routing)	Mobility is Possible	Event driven
Hierarchical [10,11]	LEACH (Low Energy Adaptive Clustering Hierarchy)	Limited Mobility	Cluster head
Hierarchical [12]	PEGASIS (The Power-Efficient GAthering in Sensor Information Systems)	Limited Mobility	One hop neighbour
Hierarchical [13,14]	TEEN & APTEEN (Adaptive Threshold sensitive Energy Efficient sensor Network)	Limited Mobility	Cluster head
Hierarchical [15]	SOP(Self Organizing Protocol)	No Mobility	Continuous
Hierarchical [16]	VGA (Virtual Grid Architecture Routing)	No Mobility	Cluster head
Hierarchical [17]	HPAR(Hierarchical Power-aware Routing)	No Mobility	Energy Based
Location Based [18]	GAF(Geographic Adaptive Fidelity)	Limited Mobility	Virtual grid based data forwarding
Location Based [19]	GEAR(Geographic and Energy Aware Routing)	Limited Mobility	Demand driven
Location Based [20]	GOAFR(The Greedy Other Adaptive Face Routing)	No Mobility	Nearest One hop neighbour

ticular functionality. This acts as a trade-off for the adaptability of the protocol. Any major variations to the network after deployment due to power, environment, or other reasons would render the network fragmented and incapable of communication between source and sink.

1.2.2 Energy Efficiency

Energy aware routing aims at prolonging the life of the network. Substantial work has been done in the development of energy aware wireless sensor networking protocols, but it is limited to routing protocols for dense network topologies where all nodes are connected.

Node redundancies are typically assumed for addressing the energy related issues. Deep sleep modes, coordinated scheduling of traffic amongst active nodes and data aggregation are techniques that have been proposed. LEACH (Low Energy Adaptive Clustering Hierarchy) [10] [11], LEACH-C, TEEN (Threshold sensitive Energy Efficient sensor Network protocol) [13], APTEEN (Adaptive Threshold sensitive Energy Efficient sensor Network protocol), [14], PEDAP (Power Efficient Data gathering and Aggregation Protocol) [21], SOP [15], VGA [16], HPAR [17], PEGASIS (Power-Efficient Gathering in Sensor Information System) [12] [22] and COUGAR [12], are few protocols that use the clustering technique and data aggregation [23] for collecting the data. Techniques like cluster head selection through probability criterion, data compression and setting thresholds for data transmission to the cluster head, are used to ensure energy balance amongst the nodes in the network.

Ad-hoc networks protocols assume redundancy. Here AODV(Ad-hoc On Demand Distance Vector), GEAR or Geographical and Energy Aware Routing [19], GAF [18] Geography-informed Energy Conservation for Ad Hoc Routing and TORA or Temporally Ordered Routing Algorithm [24] help in maintaining connectivity amongst mobile nodes. The efficiency of most of these protocols reduces to flooding when the density of the nodes in the network reduces. SPIN(Sensor Protocols for Information via Negotiation) [1] does try to address this issue of flooding but its performance degrades for sparse networks. This is because SPIN works on the assumption that it has other neighbouring nodes with which data exchange can take place. CAR [9] is designed for sparse networks. It balances energy by temporally scheduling the sleep wake pattern of the nodes. Scheduling

is done by predicting the occurrence of the events. The efficiency of the protocol is dependent on the accuracy of prediction of the event.
The study of the currently available energy–efficient protocols show that they work with the assumptions that the nodes are static or have limited mobility, there exist node redundancy, nodes are in range of each other, have GPS for localisation, the network is bounded by a geographical region, and the nodes know the location of next hop nodes. The life of the node depends greatly on how communication takes place. Each communication transaction directly impacts node's battery power. High redundancy amongst nodes brings new issues like more collision and packet retransmissions. Hence, it is pertinent to control the number of transactions required for performing the same task. Size of packets and throughput also impact power consumption. This makes the optimum usage of power for increasing the longevity of the wireless sensor networks an important objective. WSN nodes work with limited battery power and their deployment is in places where they cannot be retrieved to replace batteries [25]. These factors bring about constraints on the accessibility, availability, scalability, robustness and performance of such networks.
Networks where the topology and density of nodes is not uniform, the energy utilisation of the node is directly dependent on the traffic that it handles. Nodes with redundancy can share the traffic load, whereas, scattered nodes with limited connectivity would require intelligent opportunistic routing of traffic to ensure network longevity.
Maximum power is expended in the radio of a wireless sensor node. Energy efficiency can be improved by limiting the use of the radio needed for a given routing task.

1.2.3 Opportunistic Connectivity

Collective intelligence [26; 27], swarm intelligence [28; 29], are terms used to associate intelligence with wireless sensor networks. Wireless sensor networks are meant to emulate the collective intelligence exhibited by ants, bees, fishes and many other similar insects and animals. These animals work in groups with coordination amongst each other. These provide inspiration for formulating the sensor network architectures. Considering networks to be fully connected and have redundancy, is practically not possible for large deployments. Maintenance of a synchronous connectivity is possible only if there is certainty that the next hop node would remain connected throughout the communication process. Moreover learning from nature, there is '*Synchronism even in Asynchronism*'. Presumption that a 'window of time' would be provided by a neighbouring node for completion of the communication is not always practical for a random deployment with varying density of nodes.

Decentralised networks with a fixed stationary topology have a predictable connectivity but with networks having mobile nodes, connectivity is established opportunistically whenever two or more nodes come in vicinity of each other. Maintenance of connectivity is extremely challenging as the network topology keeps changing dynamically. The quality of service of such networks greatly depends on the ability to find the next hop node. Density of the nodes in the network have a huge impact on opportunistic connectivity. The more the number of nodes, the higher is the probability of the nodes being in range of each other. This increases the probability of a successful data exchange. This helps in providing good connectivity amongst nodes. Mobility of a WSN node with low radio communication range adds a further dimension to the connectivity issues. Here possibility of the nodes going out of range becomes higher. With density of nodes becoming sparse, the connectivity becomes poor due to formation of holes in the network. The formation of holes in the network occurs if nodes do not fall in the coverage area of each other, or if the nodes fail due to loss of power.

1.3 Research Motivation, Objectives and Contribution

Wireless sensor network protocols' dependence on topology and impact of mobility on energy–efficient connectivity is the motivation for this thesis. This section gives details of the thesis motivation and the research objectives.

1.3.1 Motivation

Research related to energy–efficient routing in wireless sensor network is concentrated on addressing routing for networks with limited mobility capabilities. Research related to mobility parameters (velocity and heading) and their impact on connectivity is addressed in this thesis. Figure 1.3 shows drop in the reception of data packets with increasing velocity at constant transmission power. These packet drops lead to energy wastage and network inefficiency.

Figure 1.3: Impact of Velocity on Packet Reception

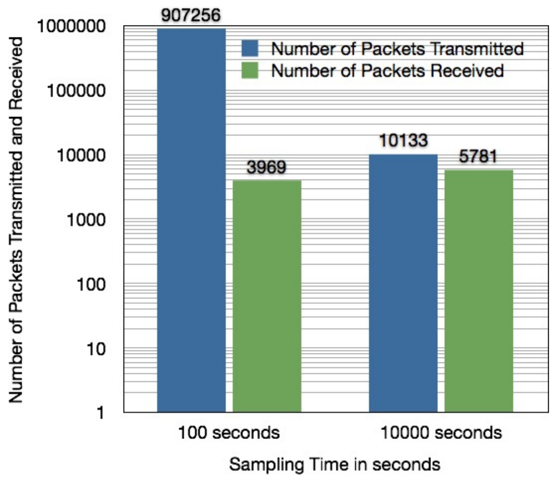


Figure 1.4: Increase in Sampling frequency resulting in Wasteful Beacon Transmissions

Moreover, mobility parameters (velocity and direction of motion) when coupled with varying density of nodes will result in network holes. This also results in further wastage of energy. Frequent sampling of network with beacons is necessary to overcome network holes and discover nodes. Figure 1.4 shows that frequent sampling of the network at regular intervals of time results in large transmission of beacons. Wireless sensor network nodes are limited in battery energy. Connectivity amongst nodes gets impaired due to mobility. Harnessing mobility to opportunistically establish connectivity is the motivation for this research thesis.

1.3.2 Objectives

The primary objective of the research is to device a protocol that is self organising. There should be limited constraint of issues related to deployment, topology, node density and mobility of nodes. The thesis addresses this issue through the design of the ECO-DETOUR protocol. The work proposes to make a node self sufficient in establishing and maintaining a communication by intelligently adapting itself to network changes.

1.3.3 Scope and Contribution

Most of the protocols mentioned in Table 1.1 have a substantial dependence on network topology and node density to perform data routing. The thesis is based on the belief that efficient routing of data exchange does not necessarily require any predetermined structure of a network. Data exchange can happen as and when the opportunity arises. As with any natural process there is an element of delay, similarly with opportunistic routing, nodes would encounter delay due to the process of node discovery. The network enables emulating real life scenarios that occur by events (in this case node discovery and data exchange process). Energy is an important factor within the process of wireless sensor networks. Power consumption is related to radio communication. Node discovery involves the process of sampling the network for the discovery of prospective next hop node. This thesis proposes the use of artificial intelligence to learn the demographic distribution of nodes and predict a sampling time for transmission of beacons. This approach results in substantial reduction in beacon transmission as discussed in our paper [30]. Once successful node discovery happens, maintaining it under conditions of mobility is a challenging task. Factors like relative velocity amongst nodes and direction of motion affect the connectivity and the data exchange process. To maintain the connectivity between two nodes the primary requirement is sufficiency of transmission range. To overcome the effect of relative velocity the thesis proposes modulation of transmission power to maintain coverage required for data exchange as entailed in our paper [31]. Further extension to the node's coverage could be made through handoff for connectivity with neighbouring nodes. The direction of motion of the node (heading) plays an important role in selecting nodes for handoff. In regions with high density of nodes, nodes would need to know a–priori the identification of the incoming node. Direction helps in selecting appropriate nodes. Handoff and direction are discussed in our paper [32]. The protocol involves four key components that act on the minimisation of energy cost in the establishment of the discovery of related nodes and connectivity to the network. These components are adaptive sampling, coverage, handoff and directional communication.

In summary, the contribution of the thesis involves design and implementation of the four key components of the protocol. Testing is performed on OPNET, where different deployment and mobility scenarios are tested and protocol performance is measured.

The thesis is structured in seven chapters. Chapter 2 deals with analysing the literature and providing critical review for the related work. Chapter 3 discusses the modelling and development tools used for this research. Chapter 4 covers the ECO-DETOUR Protocol design. The concept of Energy Efficient Opportunistic Connectivity is discussed here. In Chapter 5 the concept modelling and simulation is discussed. Results of this model are then discussed in chapter 6. Finally the conclusive remarks are given in chapter 7.

Chapter 2

Literature Review

2.1 Introduction

This chapter provides an insight into the work related to the aspects of opportunistic connectivity for delay–tolerant wireless sensor networks. The different protocols designed for opportunistic connectivity are discussed and the requirement for a new opportunistic connectivity protocol is highlighted. Close attention is paid to the area of energy–efficient routing as it applies to mobility and opportunistic connectivity. This chapter identifies the key parameters for the protocol design.

2.2 Review of Opportunistic Connectivity Protocols

Routing of data in wireless network uses multiple neighbouring and intermediate nodes for forwarding the data to the destination through multiple hops. The opportunistic routing techniques can be broadly classified as Flooding or Epidemic routing and Random or Gossip–based routing.

Flooding is a routing technique involving transmitting multiple copies of a data packet to be spread in the network. In its simplest form, it is data broadcasting. The intermediate hop nodes receive packets and forward it to all neighbouring nodes. Epidemic routing [33], which is necessarily flooding, uses the summary vector proposed by A.Vahdat and D.Becker [34]. This vector is a list of all nodes carrying the message. So any node not having the message saves a copy.

Random routing or Gossip based routing [35] is a controlled flooding of data. It arbitrarily chooses intermediate nodes based on a set probability for data forwarding. The typical probability values are between 60-80%, to cover the entire network as noted by Haas et al. [35]. Rumor routing, proposed by Braginsky and D. Estrin [36], forwards data randomly to any neighbours in a unicast. In this protocol the source generates the data which is 'rumoured' (passed) across the network. Sink, which queries for the data, also rumours the query in the network and the query propagates until it reaches any one of the intermediate nodes containing the generated data [37].

A further classification of these protocols is 'Reactive', 'Proactive' and 'Hybrid' protocols. Reactive protocols flood the network when a node requires the topology data; it differs from proactive routing, wherein topology information is periodically exchanged amongst the nodes. The Hybrid protocol is a combination of reactive and proactive routing and employs reactive routing when nodes are sparsely populated and proactive routing for dense regions of the network [38].

Multiple opportunistic connectivity protocols have been developed; a few of the significant works which are entailed in this section are Spray and Wait [39], PRoPHET [40], MaxProp [41], EBR [42], CAR [9], Bubble Rap [43], SocialCast [44] and PeopleRank [45].

Spray and Wait [39], as the name suggests, transmits or sprays data in a network and waits for it to be opportunistically forwarded to the destination. The ideology is to exploit the node mobility for the data exchange process. The protocol assumes that all nodes are aware of the destination and also the path to the destination, as the proposition expects the node to directly deliver the data to the destination in a single hop during the wait phase. This theoretical assumption requires all nodes in the network to know with high probability the precise destination, which mandates knowledge of network topology. This is in contradiction to the protocol's assumption of simple, and little prior knowledge of network topology. The protocol combines epidemic routing and direct transmission. The protocol starts by epidemically spreading the data across the network. When the protocol determines that a sufficient number of copies have been spread, it goes into a wait state where the receiving nodes attempt to forward the data to the destination directly in a single hop. The protocol's performance, with respect to total transmission time improves with increasing node density and compares with epidemic routing.

The Probabilistic Routing in Intermittently Connected Networks(PRoPHET) [40] by Lindgren et al. uses the knowledge of encounters (meeting of nodes) to calculate probability of nodes meeting in future. This coupled with node mobility, enables the protocol to forward messages to the destination. PRoPHET is an enhancement of epidemic routing. It exchanges the encounter probabilities of nodes along with the summary vector that is exchanged by epidemic routing. An aging factor is used to reduce the encounter probability if a node does not encounter over a period of time.
The protocol is compared here with the epidemic routing. The mobility pattern of PRoPHET is based on a 'community model', unlike the 'random-way point model' used in epidemic routing. Lindren et al. believe that mobility amongst nodes have a pattern and are not completely random. Mobile node velocity is assumed to be 20-30 Km/hr. A comparison of PRoPHET with Epidemic shows an improved message delivery with shorter delays.

MaxProp [41] is similar to PRoPHET, as it exchanges encounter probabilities with each other. MaxProp uses 'incremental averaging' for probability computation. The protocol assumes an unlimited buffer size for its own messages and a fixed buffer size for messages from other nodes. The window of time for communication and bandwidth is considered to be limited. It is also assumed that the information pertaining to the network topology is not required. The protocol is implemented on a network of 30 buses and uses multiple access points on the bus route for its implementation.
The protocol computes a path cost by summing up the probabilities of all nodes it encountered. The final path chosen is the one with the smallest path cost amongst all computed path costs. MaxProp assigns priority to data which are newly generated and also to the ones which are destined to go to their neighbours. Acknowledgement of delivered packets is transmitted regardless of the

number of hops required between source and destination. Importance is given to the acknowledgements as it is used to clear the buffer and make room for new data.

Encounter-Based Routing (EBR) [42] aims at maximising delivery ratio and minimizing overheads and delay. EBR proposes the idea of identifying key network characteristics that would allow intelligent message forwarding and replication. The protocol, on analysing network characteristics, assumes node encounters can be roughly predicted using past data. EBR uses a special variables 'current window counter' (CWC), which obtains information about the number of encounters in the current time interval.
The nodes, on encounter, exchange their encounter values 'EV', similar to PRoPHET and MaxProp. EBR, unlike PRoPHET and MaxProp, transmits packets proportional to the ratio of the EVs of the two encountering nodes. This scheme helps in controlling the number of packets in the network.

Context-aware Adaptive Routing, CAR [9] performs unicast communication for delay–tolerant mobile ad-hoc networks. It is based on the ideology of using nodes as message carriers in regions of disconnection. Kalman filter is used to predict and select the best carrier. CAR follows two types of delivery process; synchronous, where data is directly delivered to the sink, if both source and sink are connected; asynchronous communication is used if the nodes are disconnected. In asynchronous communication the node stores the data and forwards it opportunistically. The protocol is termed context-aware because the delivery probability is computed based on the context information. The context information are attributes of the system (node and network) which can be used to deliver messages. An example of the context given by the author Musolesi et al. [9], is that of 'rate of change of connectivity', the number of connections and disconnections in time 't'. This parameter is a measure of relative mobility and is used to find the number of encounters a node has with other nodes. Once context information is ascertained,the delivery probability of potential message carriers is predicted using the Kalman filter. The prediction of the delivery probability is done to optimise the bandwidth[1].
The protocol reduces the exchange of delivery probabilities by predicting them,

[1] Avoid exchange of delivery probabilities, as most of the values are predictable

which reduces the exchange of control packet transmission. The delivery ratio of the packets at different velocities are compared by the authors and it is noticed that the performance of Spray and Wait and PRoPHET is better than CAR for all velocities greater than 10Km/hr. This can be attributed to the restrictions imposed in the exchange of delivery probabilities, as PRoPHET and Spray and Wait exchanges encounter probabilities with all encountering nodes.

Bubble Rap [43] is a forwarding algorithm for 'Pocket Switched Network' (PSN), a type of delay–tolerant network. It is based on the ideology that the topology of the delay–tolerant network changes dynamically due to the mobility of nodes. Hui et al.[43] believe that due to mobility, the exchange of routing tables is not an efficient technique. Therefore, they propose to use the characteristic network features which do not change frequently for the exchange of data. The proposition is that, as PSNs are formed by people, the social relationships between people would change slowly compared with the network topology, and this property can be used for forwarding packets.
The protocol assumes a global community which is subdivided into sub-communities. Every node has a global ranking (level of popularity in the global community) and a local ranking (level of popularity). The concept of 'level of popularity' is based on societies where some people are more popular than others. The message is passed up the global ranking tree until it reaches another node of the same community. Now the message is passed with its local ranking until the message reaches the destination. A comparison of the protocol with PRoPHET shows the two protocols to have comparable delivery ratios.

SocialCast [44] 'Socially-aware Routing for Publish-Subscribe in Delay-tolerant Mobile Ad Hoc Networks' is based on publish–subscribe networks. Similar to Bubble Rap, it exploits the social relations amongst humans, to forward data. Publish-subscribe networks publish the data and interested nodes subscribe to them. SocialCast is a three phase routing requiring interest dissemination, carrier selection and message dissemination. In the interest dissemination phase the node broadcasts the interest to a 1-hop neighbour. The interests are assigned utility values which help in the forwarding decision. In the carrier selection phase, the utility values of the local node are compared with neighbouring nodes to select the node with higher utility value as carrier. The message dissemination phase

consists of re-evaluating the content of the buffer against a new subscription and forwarding it to the interested nodes. Here-after, a copy of the message matching an interest 'i' is sent to all the neighbours.
The protocol does look into reducing the number of replicas and allows for a controlled dissemination of data in the network. But as the data is forwarded to all the 1-hop neighbours for carrier evaluation and message dissemination, the node's radio is being used for communication. This necessarily means packets are being transmitted in the network that may be accepted or rejected, depending on the condition but the energy loss due to packet transmission does occur.

PeopleRank protocol [45] is based on the Google's Pagerank algorithm [46] that measures the importance of a web page. Mtibaa et al.[45], have extended this algorithm to capture the social interaction between people, which is intended to optimise data forwarding as in SocialCast, Bubble Rap, and others. Similar to Bubble Rap, PeopleRank tries to find the popular nodes, as they are considered more likely to meet other network nodes. The algorithm keeps incrementing the rank of a node depending on the frequency with which it meets other nodes.

The work on opportunistic connectivity protocols is concerned with data forwarding and related issues like packets' TTL (time to live), data storage issue and prediction of the delivery probabilities. The mobility of nodes is the cause of setting up opportunistic data exchange. Mobility models are used to improve the predictability of data transfers.

2.2.1 Mobility Models

Opportunistic data exchange depends largely on the encounters that occur between nodes. The predictability of successful data transfer requires knowledge of the mobility pattern of the nodes. This section briefly highlights the different mobility models.

Random mobility models are a class of mobility models which randomly change the velocity and direction of motion. The different sub-classifications of the random model are briefly discussed below. The random waypoint model randomly changes direction and velocity and follows a pattern of Brownian motion. There is a pause time between any change of direction or velocity [47].

The Random walk model randomly changes direction and velocity, but any single movement is limited by time 't' [47]. In the random direction model, the node chooses a direction and walks to the simulation boundary, where it pauses and again selects a new direction and continues. The probabilistic version of the random walk model uses a probability matrix to compute the next position of the mobile node in the network. It is less random than the purely random models.

Rather than using the random mobility models, which does not match the real life mobility patterns, it is prudent to use more specific mobility traces. The human mobility model, community–based mobility model and vehicle mobility model are captured from real data.

Human mobility models are mobility models of humans, collected by monitoring their movement and interaction with others. The data is usually collected by asking people to participate in a study [48–50]. The interaction data is collected using wireless LAN and Bluetooth. Camp et al. [47], Bai et al. [51], and Musolesi and Mascolo [52] provide a detailed explanation of the working of these models. The community–based model developed by Musolesi and Mascolo [52] contains a probability matrix called interaction matrix. The authors extracted the community structure from the matrix using an algorithm proposed by [53]. A '1' is placed in the matrix if the weight of the cell is greater than a threshold probability value. This '1' signifies the community. The authors note, that the weights assigned to the interactions is not based on any clear definition of interactions, by citing [54]. This brings in an uncertainty factor. Vehicle mobility models define the movement of vehicles on the roads. The Manhattan mobility model [55–58] is a grid based model, in which probabilities are assigned to the movement of the vehicles. '0.5' is for moving straight and '0.25' for turnings. Freeway mobility model [55; 58] has lanes in both directions and vehicles are restricted to their lanes, such that there is no random motion.

2.2.2 Gaps in the present state of the art

Most of the protocols discussed are enhancements of epidemic routing through the use of encounter and delivery probabilities. The delivery probabilities are assigned to nodes and the regions frequently visited by the node. Mobility models are considered by the nodes likely to encounter such regions and assigned delivery probabilities. The random mobility model is not well suited for opportunistic connectivity [59] due to their unrealistic mobility patterns. With the human mobility model and community–based mobility models should use a nomenclature, which can be utilised to recognise ('tag') the nodes or regions and update the probabilities. These assumptions are necessary for the building of matrices with the probability of encounter. But these model do not answer the following questions:

- What happens if a node moves out of the region?
- The determination of travel time between source and destination cannot be predicted from these models. This would lead to uncertainty, as to when beaconing should be performed for node discovery.
- On encounter, how does a mobile node maximise data transfer?

All these different opportunistic connectivity protocols have dealt with the issue of forwarding. The main concern has been to choose an appropriate next hop node which would maximise the chances of the data reaching the destination. These protocols have evolved from either flooding or epidemic routing and perform significantly better at addressing the issue of packet delivery to a destination without creating too many redundant data packets. In the process of optimising the data forwarding, significant control sequences to determine the aptness of the node as a carrier are exchanged. Although there is reduction in the presence of data packets in the network, the usage of radio to control the signalling has not been affected.

Transmission of any kind (data or control) adds to consumption of battery power. Therefore, from the perspective of optimisation of transmission, nodes should be aware of the demographic distribution in the network. This would result in each node knowing specific regions as discussed in Lindgren et al. [40]. The authors state that if a transmission takes place in this region, the probability of successful delivery of data to the destination increases.

The protocol PRoPHET assigns the mobile nodes with the probability of being found in each of the important areas like 'home', 'gathering place' and ‘elsewhere’. This is a good first approximation at identifying important regions where communication can be established. This can be further enhanced by building nodes with intelligence, such that they are capable of identifying these regions themselves. Currently there is no method described which allows nodes to recognise the region and tag it as ‘home’, ‘gathering place’ or ‘elsewhere’ themselves. The protocol reduces the probability if the node does not visit these regions for a long time. This may not be very accurate as, according to SocialCast [44] that states if two nodes are friends and have not met for a long time, it does not mean their probability of meeting in future should diminish.

Mobility models look at the assignment of probability values by modelling different types of encounters. Most of the work related to mobility modelling concentrates on collecting mobility statistics and trying to find a correlation between different movement patterns.

Simulation studies do benefit from the mobility models but their use in practical implementation appears limited, as nodes would need some characteristic network features as ‘tags’ to relate it to the probability values.

To reduce communication overheads, our proposition is to learn and predict the possible regions and ‘tag’ them, such that when the node visits the region again it can recognise and opportunistically exchange data.

This thesis uses prediction to discover the next possible region for data exchange by monitoring the current mobility pattern of the node. Prediction of such regions would allow a completely distributed network with nodes directly transmitting data to the sink as assumed in Spray and Wait [39].

2.3 Prediction Algorithms for tagging regions

Tagging of regions is required so that nodes can identify the regions which have a high probability of successfully transferring the data to the destination. Currently nodes need to exchange the probability values with each other on every encounter. Instead, if they are aware of 1-hop destinations where direct transmission could be performed, this would reduce the overhead of multiple beacon transmissions for an energy–constrained node. This would necessitate node discovery for such networks, which have large variations in velocity, and changing topology.

Dyo and Mascolo [60; 61] have proposed an algorithm based on reinforcement learning to perform an energy–efficient node discovery by modelling the scanning rate of the beacon transmission. These authors vary the scanning rate by learning from encounter history. They propose increasing the scanning when a node expects to encounter other nodes, and reduce it, when the encounter rate decreases. The protocol starts by dividing a day into N time–slots. The goal is to count the number of successful encounters in each time slot, which is considered as a reward. Therefore, based on the rewards the node schedules its sleep-wake pattern. Dyo and Mascolo believe that encounter patterns would repeat over a period of 24 hours. This assumption is valid for events which are regular and periodic like going to the office every day. Aperiodic events such as going to the shopping centre or a party, need to have a more involved prediction mechanism which is not merely temporal but is spatio-temporal.

NetDetect, an algorithm developed by Iyer et al. [62], estimates the local neighbourhood node density using a maximum likelihood estimator and assigns a probability of message transmission to the nodes. Authors find that it is difficult to estimate neighbours in a mobile environment. Kohvakka et al. [63] propose an energy–efficient neighbour discovery technique for dense networks. The protocol exchanges the synchronisation information with 2-hop neighbours. The protocol is capable of saving energy up to 80% at mobile node velocities of 1-3 m/s. Madan and Lall [64] developed a neighbour discovery algorithm in which the nodes are assumed to be distributed randomly according to Poissons distribution. The Markov decision process is used for modelling the relationship between energy used and the probability of finding nodes. The protocol performs well with high node density but with low node density the performance degrades. As these techniques work for low velocities or consider periodicity to assign a probability value for a successful encounter and data exchange, there is a need for building intelligence in the node such that they are able to adapt to the varying velocities, node densities and network topologies. This can be achieved through learning and prediction of the node demography in such regions.

Estimation techniques like Least squares estimation [65; 66] and the Kalman filter [67; 68] are traditionally used for predicting future states based on past trends.

These mathematical prediction tools require a system model. Their performance is good if the nodes need to be trained for specific topologies over a definite period of time [61]. These techniques can follow and learn the variation in a system once they achieve a steady state. After achieving a steady state, if there are variations in the system or if this trained node is used in another environment with a different network topology, the node would need to be trained again. Soft computing techniques, like Fuzzy logic, Neural Networks, Bayesian Networks and Evolutionary Algorithms, enable finding solutions for complex problems with incomplete definitions [69–72] and do not require a system definition.

Neural Networks do not need a system model for prediction [73; 74]. This is ideal in dealing with complex mobility models like human mobility, community–based mobility and others, where development of a mathematical model may not be very accurate.

Neural Networks are classified into two categories, based on their training requirements as an unsupervised artificial neural network and a supervised artificial neural network [74; 75]. Supervised artificial neural networks need to be trained before use. Their prediction capability is limited to trends, which are an extrapolation of their training. Unsupervised neural networks have the capability to learn and adapt to trends with which they have not been trained. Considering mobility amongst humans, there may be many friends of a person; if every friend is considered to be a region where data exchange can happen, then the network would learn all the regions through training. If one makes a new friend, the system would not recognise it as it is not trained for this new input. As real life scenarios are dynamic, unsupervised learning techniques would be able to learn and adapt to the change. Self Organising Map (SOM) and Adaptive Resonance Theory (ART) are types of unsupervised artificial neural network algorithms [76]. Markou and Singh [77] provide a survey of different neural network algorithms. They highlight that the classification of ART in comparison to SOM, is better.

ART algorithm was developed by [78] in an attempt to model the brain. ART is a clustering algorithm which clusters the input depending on a variable parameter termed as 'vigilance parameter'. The vigilance parameter ranges between 0 and 1. Higher values of vigilance causes small variations in input to be classified as different clusters, lower vigilance values cluster inputs with minor variations

together.
The encounter amongst nodes, if fed into the ART algorithm, would enable creation of a spatio-temporal clustering of different regions where node encounter takes place. This would allow nodes to schedule their sleep wake pattern without assignment of probabilities. In [60; 61] once a nodes learns a pattern, if network topology or mobility pattern changes, the encounter pattern will also change. This would result in the system requiring a new training. As ART can accept new inputs and create a new cluster for them, there would be no need for new training.
Putting nodes in a low energy state until they encounter a region where they can exchange data would allow nodes to conserve energy. Once nodes encounter other nodes they should then maximise the opportunity to exchange data. The transmission radius or transmission coverage of a node can affect the number of packets the node can successfully exchange with other nodes.

2.4 Extending Window of Time on Node Encounters

The window of time for a mobile node with other, relatively stationary nodes, depends on the amount of power a node transmits, which directly relates to the coverage area. It also relates to the number of nodes which collaborate in providing data handoff to the mobile node. This section discusses literature on both coverage and handoff.

2.4.1 Coverage

The coverage of a node is normally treated in terms of the total sensing area of the network according to Yu et al., Xueqing et al. and Jae-Hwan et al. [79–81]. These works focus on the requirement of covering targets, by determining the minimum number of nodes required to cover a region for static networks noted by Francois et al. [82]. Zuniga and Krishnamachari [83] address the optimal coverage of a node for providing connectivity amongst nodes. However, the requirements of coverage for a node under conditions of mobility is not detailed. The concept of optimal transmission radius is used by Francois et al. in [82] for broadcasting. Transmission radii is optimised to efficiently control the topology of the network based on the target range. This helps in conserving energy. Francois et al., claim the protocol to be efficient for all types of network density (number of nodes). Their protocol is meant to handle only static networks with node redundancy. Extension of this topology management technique for mobile networks through modulation of power based on mobility parameter velocity would enable varying the transmission coverage depending on node density in the region. The issues of optimum transmission radius for query propagation within large scale network is discussed in [83]. This work shows that choosing a transmission radius without previous analysis can lead to an unnecessarily long settling time (the time it takes for the data propagation) in the network, which will decrease the overall network throughput.

Hence it becomes important to analyse the requirement of the coverage radius of a node based on the relative velocity and data size of the mobile node.

2.4.2 Handoff

The connectivity of a wireless sensor network becomes affected under conditions of mobility. The mobile node would move from coverage of one node to another and hence would not be able to perform proper data transfer. To facilitate data transfer, the concept of Handoff is proposed for WSN. When the mobile node comes in the coverage of one node, it transfers part of the data and enters the coverage of another node where it transfers the rest of the data. These data packets are then collated. The other node cannot accept data if it is not aware of the identification of this mobile node. So when the mobile node enters the first node, this node based on the direction of motion performs Handoff (transfer of control information) to the next node. The literature addresses the Handoff issues for GSM and WLAN networks. There is no evidence of Handoff being adopted at the sensor network level. Handoff latency [84–86] is an issue which results in disruption of communication. Its effect becomes more pronounced in

a wireless sensor network because the coverage area of the nodes is small, hence the time spent by the mobile node in the coverage area of a stationary node for data exchange is small. Under such conditions, if the latency is high, by the time the Handoff takes place, the mobile node might have left the coverage area of the node, with effectively no data exchange. Delays like Probe Delay, Authentication Delay and Reassociation Delay have been identified. These delays are caused by extensive exchange of control signals. These effects are more critical to a voice call, which needs to be jitter free, unlike data which is broken down for transferring to each node. The sensor network does not have such stringent requirements. Nevertheless, delay has to be controlled by limiting the use of control signals.
The use of Handoff here is merely to increase the overall coverage area. The mobile node transfers part of the data to each node involved in Handoff. This data is later integrated.

2.5 Range and Direction of motion

Detection of range and direction of motion are important factors for Handoff amongst nodes. Detection of nodes for Handoff is possible by monitoring the signal strength. But under conditions where multiple nodes overlap with the node initiating the Handoff, it becomes necessary to determine the direction of motion to provide Handoff to the correct node. Ali et al.[87] entail the use of an accelerometer to track the movement of a truck with an accuracy of two inches. San-Yuan et al.[88], use an accelerometer coupled with a gyro and magnetometer to track motion. Liu in [89] uses signal strength to sense the link between nodes. This link sensing is proposed for a highly mobile ad-hoc network where the links break frequently and need to be repaired to avoid packet loss in Handoff. The work proposes transmission of a 'hello signal' which would be compared with a threshold signal strength to either establish or reject connectivity with the node. It proposes rejection of received packets to establish the fact that the signal strength is weak but it also has the repercussion of losing a complete packet. Musolesi and Mascolo [90] and Munir et al.[91] use signal strength to establish reliable paths between source and destination for the exchange of heavy traffic. The protocol searches for paths with high signal strength and uses this path for future transmissions. The protocol by Musolesi et al. in [90] works with the assumption of limited mobility, as under conditions of high mobility there is no guarantee of a fixed path between source and destination. The proposal by Munir et al. in [91] for high mobility requires frequent polling for signal strength that would impact the life of the node. The scheme proposed by Liu in [89] is more reliable compared to Munir et al. [91], except for the concept of rejection of the received packet.

2.6 Conclusion

Relating power–efficient communication and connectivity requires addressing the challenge of optimum usage of radio. This is because radio communication consumes a significant amount of energy. The process of establishing connectivity requires scanning the network for a possible next hop node with beacons and maintaining it with an optimum coverage. All these processes require the radio transmission which leads to expending the energy.

Through the review of the literature it is evident that studies have concentrated more on issues such as data storage and control of packet flooding in the network. These studies have not identified the parameters of connectivity in a mobile environment.

This work identifies the parameters that impact connectivity, such as velocity, direction, duration of connectivity and node coverage.

Chapter 3

Modelling and Development Tools

3.1 Introduction

Wireless sensor networks are meant to have a large scale deployment, with thousands of nodes. Testing of concepts and protocols requires scalable tools which can effectively result in monitoring and analysis of the protocol within the deployed network. The different categories of tools used, is shown in Figure 3.1. The list of tools mentioned in Figure 3.1 is only indicative of what is available currently, as researchers and companies are constantly producing upgrades and new tools.

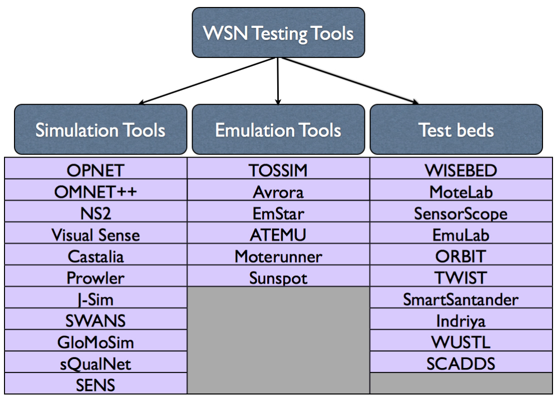


Figure 3.1: Test tool categories

3.1.1 Sensor network concept development environments

The testing of wireless sensor networks can be performed by using either simulators, emulators, test beds or a combination. Each has their own advantages and disadvantages. The following section discusses the simulation tools. A brief discussion of the emulators and test beds is given in Appendix C, as they do not fall in the purview of this thesis.

Simulators do not require actual hardware for testing purposes. There is virtually no limit to the number of nodes that can be simulated. It is constrained, by the processing capabilities of the machine on which simulation is performed. With better high performance computers, complex simulations are possible.

Visual Sense[92] is a graphical Java–based simulator for wireless sensor networks, built on Ptolemy II. The tool is very easy to use and has very good visualisation. It supports continuous time simulation and discrete event simulation. Components and actors are readily available or can be abstracted or created in Java. The tool is suitable for developing first approximation models. The software lacks a good model for the radio, which provides limited confidence in its results.

Prowler[93] is a discrete event simulator which runs under MATLAB. The tool is designed primarily to optimise network parameters. Prowler, unlike Visual sense, has limited mobility modelling ability. Prowler has a graphical user interface for visualisation of simulation. The tool provides accurate channel models but the modelling of the effects of terrain on signal degradation is not possible. Prowler has a Java implementation called JProwler.

J-Sim[94] has an autonomous component architecture–based simulation environment written in Java. The simulator has components as basic entities which are assembled to design nodes and scenarios. The simulator requires a scripting language to define a scenario. It offers support for Perl, Tcl or Python. J-Sim is a real time process driven simulator. Its support for mobility is limited.

JiST/SWANS[95] is a ad-hoc wireless network, discrete event simulator. The simulator has nodes as entities, which are composed of OSI layers. Java programming language is used for modelling. The tool has limited visualisation support and lacks support for wireless sensor networks.

SENS[96] is a wireless sensor network simulator which uses C++. It has a modular and layered structure with Application, Network, Physical and Environment as four customisable components given by Sameer et al. [97]. Each of the components needs to be defined to develop a scenario. The current limitation of the system is that it supports only the Linux operating system.

Visual Sense, Prowler, J-Sim and SENS are not full scale simulators that are scalable. These are useful to test a scenario for the first approximation, but extensive testing for parameters such as throughput, packet loss, dynamic interaction of node for communication with accurate radio modelling, effects of terrain and transmission losses requires more involved tools such as OPNET, OMNET++, NS2 (now NS3), QualNet and GloMoSim.

OPNET[98] is a commercial network simulator hosting the implementation of many protocols and commercial modules (the servers of Cisco, HP and others); protocols like Zigbee are available for WSN implementation. The software allows the design of custom nodes and protocols. Visualisation is very good. Graphical implementation of mobility trajectories helps designers visualise the scenarios they create. Custom protocol designing or modification of existing protocols requires a knowledge of C/C++. The wireless channels are implemented as a 14–stage pipeline. The default models can be easily modified to model channel characteristics.

OMNET++ [99], which is similar to OPNET, is a discrete event simulator framework which uses C++. The simulator is supported by communities and does not natively come with implemented protocols. It has a graphical user interface, which enables visualisation of networks. The advantage of community support is that many protocols and models are continuously being added. The simulator uses NED (network description) files to describe the interconnections of a node. This involves a learning curve for the software.

MiXiM [100] is a project which caters for the requirements of wireless sensor networks. Castalia[101] is built on OMNET++. It is specifically designed for wireless models and has an advanced channel model and a radio model. It supports enhanced modelling of sensor devices. It has a special feature to monitor the node clock drift. The downside of Castalia is the inherent issue with OMNET++. A major drawback with OMNET++ currently is multiple projects are available providing different network features, but they do not have a common

interconnecting interface that would enable combining the features of these different projects.
NS2[102] is a discrete event simulator specifically designed for network research. It uses C++ for protocol designing and Tcl (Tool command language) for scripting the interconnections in a scenario that includes detailed scripting. An external visualisation tool for NS2 is network animator (NAM). NAM does not provide support for wireless network visualisations. NS2 does not have a native graphical editor for scenario deployments, which is a drawback.

A good graphical editor helps in the visualisation of a deployment, and also helps the researchers and designers to maintain focus on the core idea and its performance analysis, rather than the coding and implementation of network deployment scenarios.
GloMoSim[103] is a scalable simulation environment for wireless network systems. It supports multiple routing protocols. It was designed using the capabilities of Parsec, a parallel discrete event simulator. For analysis of results it uses a java–based visualiser and shell scripting. The protocol designing is done in C language. Familiarisation with Unix shell scripting is required. It supports all OSI layers. Mobility model needs to be chosen from a set of predefined models or loaded from a file. This does not give the designers the visualisation of the trajectory while creating the file. The commercial version of GloMoSim is QualNet [104]. sQualNet[105] which is an extension of QualNet for wireless sensor networks. This simulation framework has support for the Sensor channel, Battery model, Processor power consumption, a Sensor traffic model, the Non-IP and Multiple network protocol, a power consumption model for Mica motes and WINS[106].
The different simulators are summarised, with their features, in Table 3.1.

Among the many network simulators, NS2, OMNET++, QualNet and OPNET are preferred simulators. They have a general architecture (not limited by specific technologies like TinyOS, AVR and others), support for multiple protocols, analysis support, scalability and better channel models and documentation. NS2 lacks a native graphical user interface. The mobility pattern needs to be entered in a file as coordinates. Due to lack of a visualisation tool it becomes difficult to place the nodes and generate their corresponding trajectories. Even the new version of NS2, i.e NS3, does not have a default graphical interface, but

Table 3.1: Comparison of Simulators

	Suitablility for WSN modelling	Scalability	Mobility Trajectory	Graphical User Interface	Visualizations	Effectiveness of Wireless Channel Model	Library of Nodes & Protocols	Analysis Tools
Visual Sense	Component based Preliminary Modeling	Supports 10-50 nodes	Limited Mobility Modeling	Yes	Good Visualization of mobility and data transaction	Poor Model	Absent	Generic graphing tool
Prowler	Matlab based Good Network Model	100-500 nodes	Limited Mobility Modeling	Yes	Good Visualization	Tool specifically designed to model radio propagation	Absent	Uses Matlab's analysis capability
J-Sim	Java based Preliminary Modeling	100-500 nodes	Limited Mobility Modeling	Yes	Limited Visualization	Limited Model	Absent	Good graphing tool
JiST/ SWANS	Java based Involved Modeling	upto 1 million nodes	Input Trajectory File	Yes	Limited Visualization	Good Model	Absent	Good graphing tool
SENS	C++ based Good Modeling	upto 1000 nodes	Limited Mobility Modeling	Yes	Limited Visualization	Good Model	Absent	Good graphing tool
OPNET	Objective C based Involved Modeling	Dependent on Memory and Operating System	GUI based Trajectory specification	Yes	Good Visualization of mobility and data transaction	Good Model	Extensive Library	Good graphing tool
MiXiM	Involved Modeling	Dependent on OMNET ++	Input Trajectory File	Yes	Good Visualization using OMVis	Good Model	Use OMNET ++ resources	Uses graphing capabilities of OMNET++
Castalia	Involved Modeling	Dependent on OMNET ++	Input Trajectory File	Yes	Good Visualization using OMVis	Good Model	Use OMNET ++ resources	Uses graphing capabilities of OMNET++
OMNET++	Involved Modeling	Dependent on Memory and Operating System	Input Trajectory File	Yes	Good Visualization using OMVis	Good Model	External Library	Good graphing tool
NS2	Involved Modeling	Dependent on Memory and Operating System	Input Trajectory File	External GUI tool	External GUI tool	Good Model	Extensive Library	Good graphing tool
sQualNet	Involved Modeling	Dependent on Memory 1000-5000	Input Trajectory File	Yes	Good Visualization	Good Model	Library for Sensor Networks	Good graphing tool
GloMoSim	Involved Modeling	10,000-100, 000	Input Trajectory File	External GUI tool	External GUI tool	Good Model	Limited Library of nodes	No native graphing tool

comes with a separate animation tool NetAnim [107]. NetAnim currently does not support wireless animations, which is important for the study of connectivity. QualNet has an extensive library, but due to its prohibitive cost, it is not used for this work. OMNET++ is a free software for non-profit work. The default OMNET++ tool only comes with a discrete event simulator environment. Researchers contribute to different frameworks which are built on top of OMNET++ and use it for their execution. The tool has a graphical user interface. The programming is a combination of C++ and uses a scripting language to define scenarios. It currently has support for WSN [100], which is under development.

OPNET, in comparison with OMNET++, relies more on the use of a graphical user interface for defining scenarios, node structures and the finite state machine FSM to define protocols. OPNET has support for run time debugging by allowing stepping through the activation of different processes and through the use of breakpoints. The graphical user interface is used to draw trajectories in OPNET.

OPNET is chosen over NS2 and OMNET++ because of its native graphical user interface which allows users to visualise while drawing a trajectory, providing greater precision in its definition. It allows ease of duplication of scenarios and their comparison with different protocols for their performance analysis. OPNET allows importing road maps on which mobility trajectories can be set [32]. This feature allows ease of defining node mobility models.

Opportunistic connectivity protocols need to have a mechanism to monitor their encounters with other nodes in the network as detailed in the literature. This requires mathematical modelling of the encounters. Network simulators have limited mathematical processing capabilities, so MATLAB is chosen to simulate mathematical models. It has an extensive mathematical library and toolbox, which are not available in network simulators. The mathematically complex aspects of the protocol are time consuming to code in C, which has an ANSI standard set of limited functionalities. MATLAB's mathematical library has ready–to–use functions and thorough documentation explaining their use.

3.2 Tools for simulation

The tools chosen for this work are OPNET and MATLAB for network and mathematical simulations. The simulation setup to test opportunistic connectivity scenarios and perform protocol comparison is shown in Figure 3.2. Network scenarios are implemented in OPNET and the simulation data is collected as '*.csv*' file. Each node in OPNET creates its own '*.csv*' file, which are processed individually in MATLAB. The features of the different tools used to conduct this work are entailed in this section.

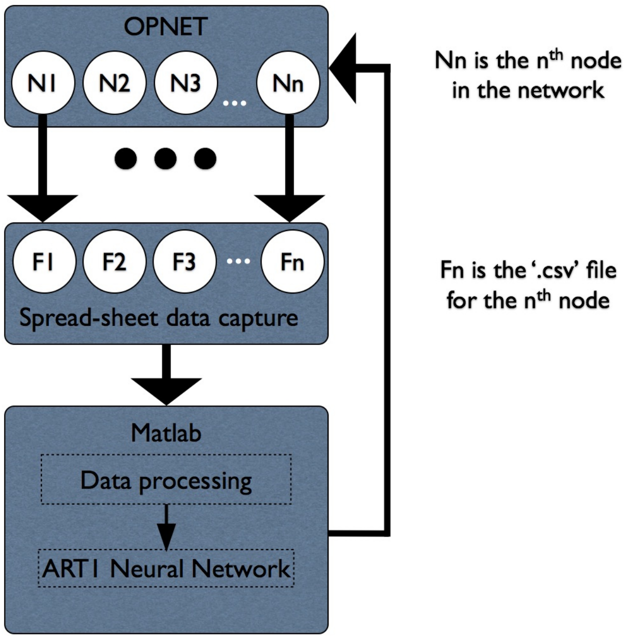


Figure 3.2: Simulation Setup

3.2.1 Features of OPNET

OPNET models the network in different layers. '*Network Model*', '*Node Model*' and '*Process Model*' are used to specify the network and node, define the protocol respectively. The link and the path characteristics of the channel are modelled using the '*Link Model*' and the '*Path Model*'. This compartmentalisation of the different models allows model reuse. The network model, node model and process model are linked hierarchically, as shown in Figure 3.3. Design and development of protocols require the creation of a node model that is simulated as scenarios in the network model. A typical OPNET scenario is shown in Figure 3.4 (a) and (b). Opportunistic connectivity protocols deal with node mobility. OPNET provides definition of trajectories for nodes, to enable simulation of mobility. The white line in Figure 3.4 (b) shows the trajectory for mobile node 0. The window for trajectory definition is shown in Figure 3.4 (c), which accepts a name for the trajectory. On clicking '*Define Path*', shown in Figure 3.4 (c), the window to specify trajectory parameters such as velocity, yaw(heading), etc. comes up, as

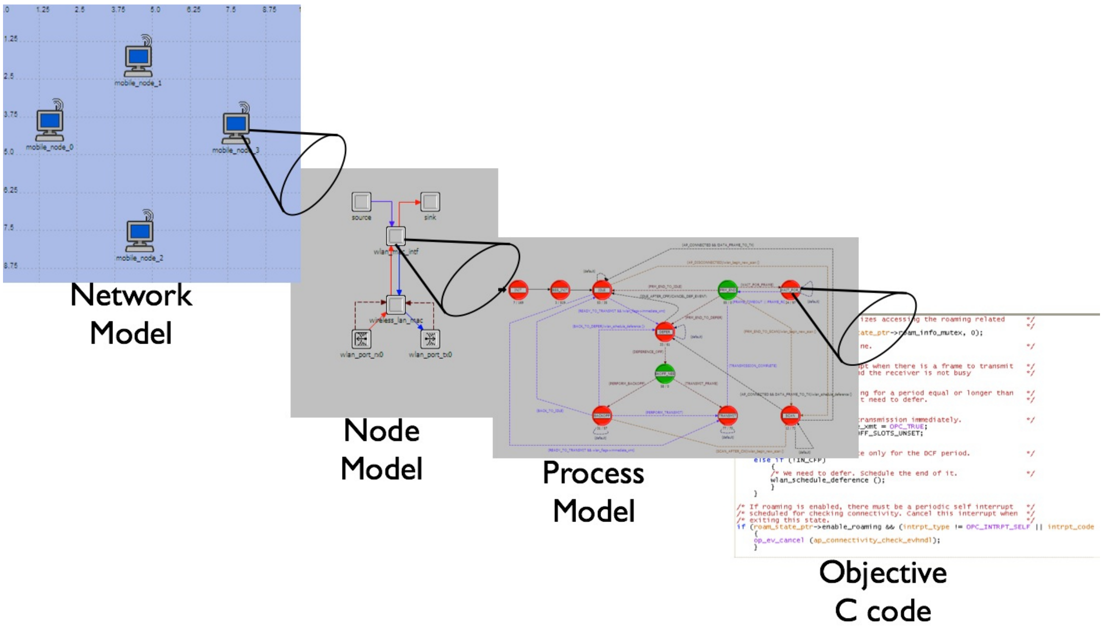


Figure 3.3: OPNET modelling hierarchy

shown in Figure 3.4 (d) and (e). These two windows allow accurate specification of trajectory in the following units (distance in Kilometer, miles or meters and time in hours, minutes and seconds). The trajectories are drawn graphically with the mouse, which makes estimating the length of the trajectory difficult while drawing. OPNET provides the window shown in Figure 3.4 (e), which shows the trajectory length and its traversal duration, which helps in the drawing of the trajectories. OPNET has a large database of nodes for different hardware and protocols, which can be selected from the '*Object Palette Tree*'. The wireless sensor networks usually use the Zigbee protocol. The Zigbee protocol on OPNET is provided as an object file, which prohibits any modification to the code. The WLAN nodes available within OPNET allow complete access to all the models (node model, process model and the Objective-C code), making it ideal for implementation of the ECO-DETOUR protocol. To develop a protocol, only the radio of the node is required for wireless transmission and reception. The choice of WLAN nodes provides the basic mechanism to establish communication amongst nodes.

The wireless node configuration is shown in Figure 3.5. The configuration

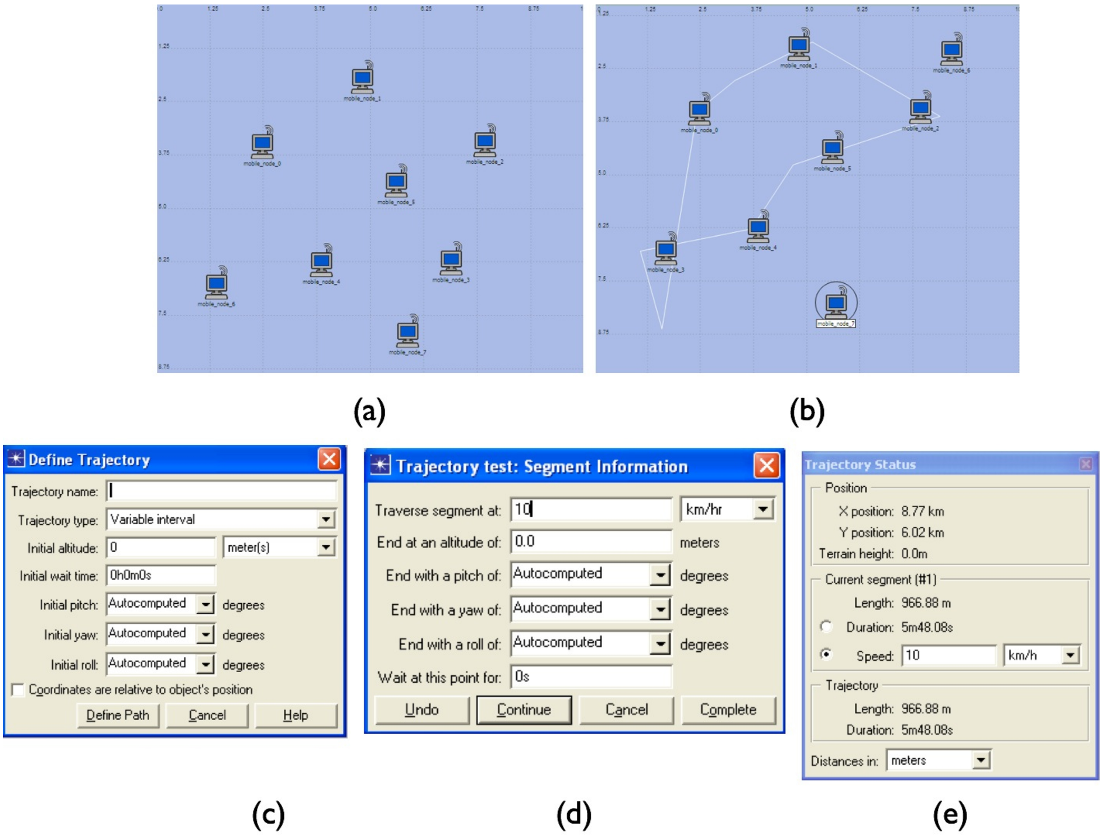


Figure 3.4: OPNET scenario with mobility modelling: (a) OPNET scenario with static nodes. (b) OPNET scenario with mobile nodes. (c) Trajectory definition window. (d) OPNET window to specify trajectory parameters. (e) OPNET window showing trajectory length and its traversal duration

allows for setting the mobility trajectory, transmission power, receiver sensitivity and other parameters. These attributes can be manually set before the simulation begins or can be set through coding in the process model using Kernel APIs. Manipulation of these attributes through code allows changes during the run-time of the simulation. This allows study of the effects of these attributes on the simulation.

Further features of OPNET such as '*Scenario duplication*', '*Node Model*' and '*Process Model*' including file operations is given in Appendix D.

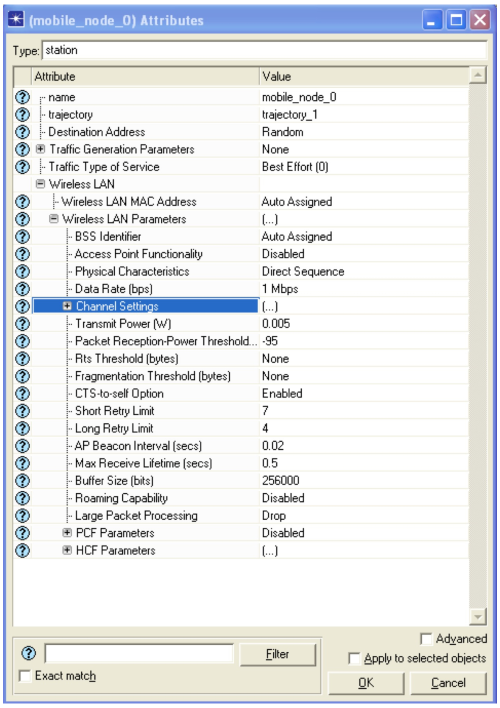


Figure 3.5: WLAN node configuration parameters

3.2.2 Features of MATLAB

The most discerning feature of MATLAB is that all variables are treated as matrices and there is no requirement to declare the dimensions of the variables before they are used. This feature enables handling variables whose dimensions may vary over the course of execution of the simulation.

Performing similar operations in C would require a dynamic memory allocation in which memory would have to be created and destroyed. This operation is automatically handled by MATLAB.

The protocol implementation requires dealing with variables of unknown dimensions. The ART1 neural network algorithm handles matrices of different sizes. The input to the neural network is a 2-D matrix, but the neural network algorithm processes the inputs as a 1-D binary string. The implementation also requires reshaping a two–dimensional matrix into a row vector and later the row vector must be converted back to a two–dimensional matrix. Conventional programming would necessitate multiple loops to perform the operation affecting the

```
>> a

a =

     0     0     1     1     0
     1     1     0     0     1
     0     1     0     1     1
     0     0     0     0     1

>> reshape(a,1,prod(size(a)))

ans =

  Columns 1 through 14

     0     1     0     0     0     1     1     0     1     0     0     0     1     0

  Columns 15 through 20

     1     0     0     1     1     1
```

Figure 3.6: Conversion of 2-D matrix to 1-D vector

flow of the code. MATLAB uses the function 'reshape()' to change the dimensions of a matrix as shown in Figure 3.6, in which a 4x5 matrix is converted into a row vector of 1x20.

3.2.3 The ART neural network

The ART1 Neural Network architecture offers continuous learning and adaptation capability. The ART1 NN architecture is shown in Figure 3.7. ART1 NN uses a

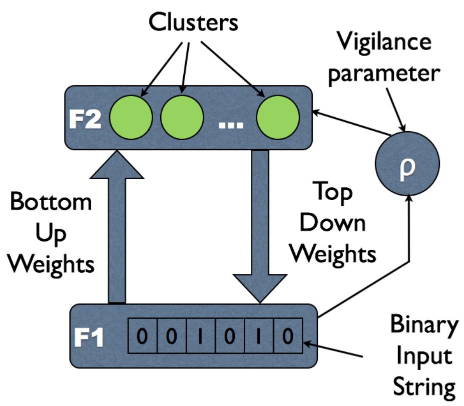


Figure 3.7: ART1 NN architecture

'$match-based\ learning$'. The top down and bottom up weights act as memories to affect the clustering. The match–based learning affects the weights only when a completely new input (which has never been clustered before) appears. The weights will not be affected if the input matches the internal expectations of the network [78] and would be clustered with a previous similar input. Figure 3.8 shows the clustering of alphabets A,B and C presented to the neural network algorithm in a random order. It is evident from the clustering that the neural network algorithm is tolerant to noisy inputs. The ART1 algorithm creates three clusters for each of the three characters. The characters are presented in a random order one after another and the clustering happens in real time, i.e the network does not need supervision to identify and cluster the input characters.

The predictions of node encounters in opportunistic connectivity scenarios cannot be predetermined and therefore the unsupervised learning capability of the ART1 neural network algorithm would help in the prediction of encounters.

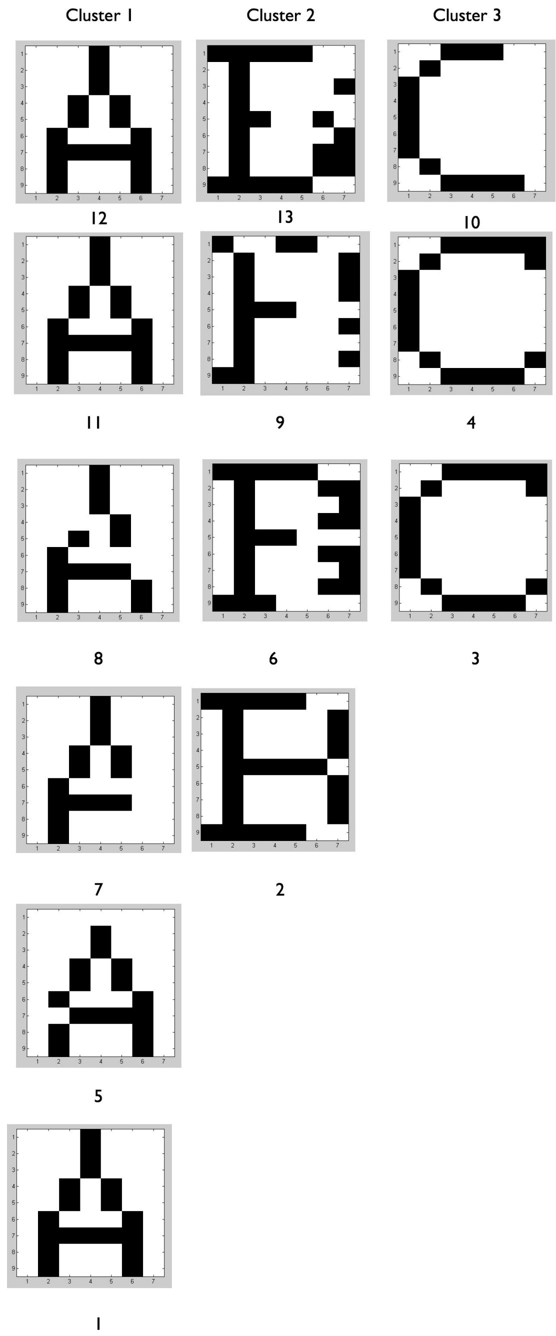


Figure 3.8: Clustering of input characters

3.2.4 OPNET and MATLAB connectivity

OPNET uses Objective C for programming. MATLAB programming is done in MATLAB's native scripting language. Connecting the two softwares, allows the use of MATLAB commands in Objective C code. Appendix A gives the environment configurations required to connect the two softwares.

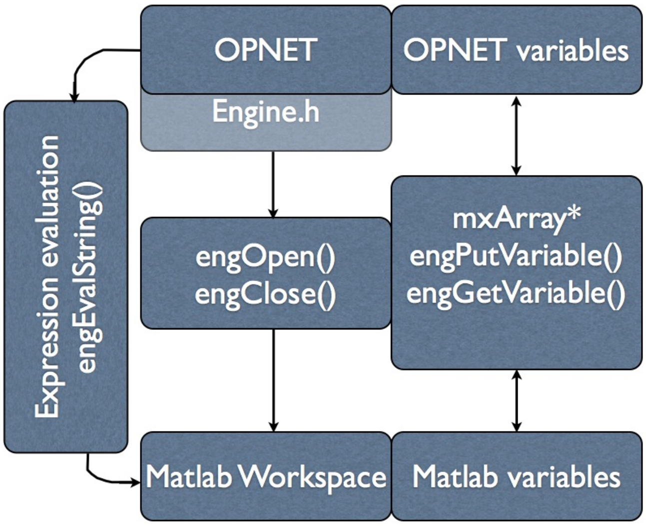


Figure 3.9: Call to MATLAB from OPNET

MATLAB workspace can be called from OPNET by using the '*Engine.h*' header file. The '$engOpen()$' function calls the workspace and the variables are exchanged between the two environments as arrays using the pointer '$mxArray*$' as shown in Figure 3.9. This connectivity of software allows embedding of MATLAB script within the C code as shown in Figure 3.10. The '$mxCreateString(char*)$' creates a string variable in MATLAB. The '$engPutVariable()$' and the '$engGet Variable()$' functions allow putting and getting variables to and from the MATLAB workspace. To execute any MATLAB script the '$engEvalString()$' function is used. Additional detailed description of these functions can be found in MATLAB help.

```
Engine *ep;
mxArray *filename[1];
mxArray *map1 = NULL;

if (!(ep = engOpen("\0")))
{
        fprintf(stderr, "\nCannot open MATLAB\n");
        return 0;
}
/*passing the .csv file name
Variable filname contains the .csv file name of the node*/
filename[0] = mxCreateString(filname);

/*
 * Place the .csv filename of the node into the MATLAB workspace
 */

engPutVariable(ep, "filename", filename[0]);
engEvalString(ep, "filename");
engEvalString(ep,"map=trajectorymap(filename)");
map1=engGetVariable(ep,"map");
engClose(ep);
```

Figure 3.10: MATLAB script in C

3.3 Conclusion

Opportunistic connectivity involves node mobility, which impacts on their behaviour with other network nodes. Therefore this thesis necessitates using a network tool which supports mobility. As communication is the core aspect of the thesis, the network and communication models of the tool should be reliable and accurate. Amongst the many simulation tools available, NS2, OMNET++ and OPNET were found to have continuous support from their designers and be scalable and robust. Due to lack of a native graphical user interface, NS2 was not considered. OMNET++ lacks support when a simulation has to be performed using multiple framework, as different frameworks have different features. OPNET has a complete graphical user interface and a hierarchical encapsulated design methodology, which is helpful in designing nodes and also debugging. The graphical user interface allows the drawing of a trajectory, which helps in visualising the trajectory path and the placement of nodes. The opportunistic connectivity protocol requires implementation of a neural network algorithm. As MATLAB has a large library of mathematical functions, it was chosen for the implementation of the artificial neural network. OPNET and MATLAB, which were chosen for this work, are connected through C–based function calls, which call MATLAB workspace for the exchange of variables.

Chapter 4

ECO-DETOUR Protocol Concept Development

4.1 Introduction

Opportunistic connectivity arises due to node mobility, variation in the network node density and limited transmission range. These three factors result in a degree of uncertainty in the establishment and maintenance of connectivity. This uncertainty leads to the probing of the right opportunity for establishing connectivity. The literature shows that opportunistic connectivity protocols need to exchange probability values on every encounter [9; 33; 40], in order to choose an appropriate carrier of the data. The approach is appropriate for processes which are periodic and repeatable.

If the position of the node changes due to mobility, or the encountering pattern changes, the probability values will also need to change. This means more control signals need to be exchanged to change these probability values. This is an overhead on the data communication process. ECO-DETOUR looks at an energy–efficient approach by optimising the communication overhead by embedding intelligence in the node. It is designed to constantly learn and intelligently recognise the regions where the probability of finding data carriers is high.

Energy expense can be further limited by optimising the transmission power of a node by using coverage and data handoff. These are determined based on the window of time[1] for connectivity amongst nodes. It is important that the window of time be sufficient for a successful transaction of data. Node velocity, size of data and the data rate dictate the window of time. This is used to choose either coverage or handoff.

ECO-DETOUR protocol is designed based on the following assumptions:

1. The node density varies over the region of deployment. Some regions would have high node density and other regions be sparsely populated.

2. The node that is searching for opportunistic connectivity is a mobile node that follows a given trajectory pattern.

3. All nodes in the network that are potential recipients of the mobile node data are capable of being mobile.

4. The nodes considered stationary are either at velocity zero or their relative velocity with other nodes in the network is zero.

This chapter begins by providing an ideology and a high level protocol design. It then details the design of each of the suggested operational components.

4.2 Ideology of ECO-DETOUR Protocol

Opportunistic connectivity requires nodes to keep searching for other nodes that fall in their coverage range, and enable the exchange of data. The searching process is based on the process of sampling the potential for connectivity. Sampling with a beacon helps in the discovery of neighbourhood nodes. The main objective here is to enable sparse nodes to off–load their accumulated data collected about their status or a related process, into a discovered node or group of nodes. In the process of node discovery, excessive sampling results in energy wastage. Appendix B shows the relation between the probability of successful sampling and

[1]Window of time is the time a node spends in the coverage area of another node for data exchange

beacon transmission for different node density. It is evident that even at high sampling frequencies, if the density of nodes in the region is low, the probability of successful sampling is low. This issue makes it pertinent to the optimisation of network sampling to reduce energy wastage. We term sampling, in a smart and energy–efficient manner as adaptive sampling.

Adaptive sampling attempts to minimise the number of beacons during the process of node discovery. The approach for this research is based on the use of artificial neural networks. It attempts to learn the motion trajectory of the node. The regions where sampling is successful is termed a 'hot spot'. On node discovery, it is prudent to maximise the data transfer between two nodes. This ensures maximisation of energy utilisation towards data exchange, which is the goal of a network. Modulation of the coverage area of a node enables controlling the window of time for connectivity. In opportunistic networks, data keeps accumulating within the node until a node discovery is successfully performed, which sometimes results in an increase of data size.

Coverage of a single node may not be sufficient to exchange all the data in a single transaction. Data would need to be transferred over multiple nodes. The issue with transmitting data over multiple next hop nodes is the exchange of control signals of these nodes with the mobile node. As the node is mobile, the amount of time it spends in the coverage of another node is limited. This means if the node expends time exchanging control signals, the window of time to exchange data gets affected. This issue can be addressed by increasing the window of time through handoff of a mobile node amongst stationary nodes. The stationary nodes bear the overhead of exchanging control signals and providing a seamless path for the mobile node.

Handoff is performed in regions where the density of nodes is high. This allows multiple nodes to collaborate and collect data from the mobile node. Handoff is made dependent on direction of motion to select suitable nodes that lie in the trajectory of the mobile node to participate in data exchange. This is done to reduce the unwanted overhearing and packet loss.

Figure 4.1 indicates the four key components that play an important role in defining successful opportunistic connectivity. The node connects opportunistically with other nodes. It samples the network through adaptive sampling.

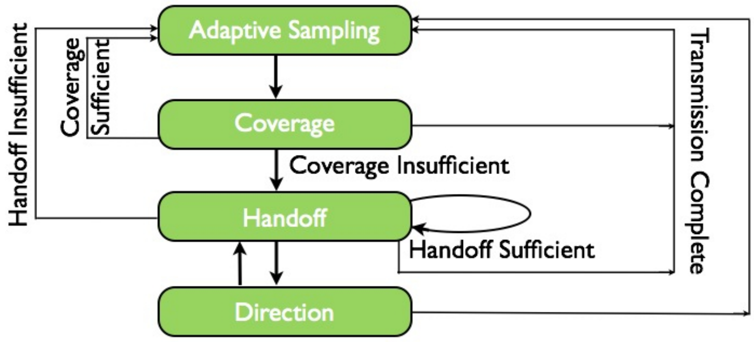


Figure 4.1: ECO-DETOUR protocol illustration

Adaptive sampling monitors the mobile nodes' motion trajectory and identifies hot spots. Once the discovery of a node in the neighbourhood takes place, the nodes' coverage is adapted to maximise the data exchange through the coverage process. If the available coverage capacity is inadequate, data would be split and transferred over multiple nodes through the handoff process. For a successful handoff, nodes that are within the network coverage, and lie within the mobile node heading, need to be identified. The direction process determines the heading of the node to aid the handoff. Once handoff is complete or when it is not possible to have a handoff, new nodes need to be discovered, therefore the process of adaptive sampling is reinitiated.

4.3 ECO-DETOUR Protocol Design Parameters

ECO-DETOUR protocols' acting functional components and variables are depicted in the tree diagram shown in Figure 4.2.

The independent variables required for the functioning of the protocol are 'hot spots', 'frequent visit count' ,'velocity', 'time', 'data size', 'baud rate', 'node heading' and 'number of nodes'. These primitive variables feed the four functional components ('adaptive sampling', 'coverage', 'direction' and 'handoff') which make up the protocol.

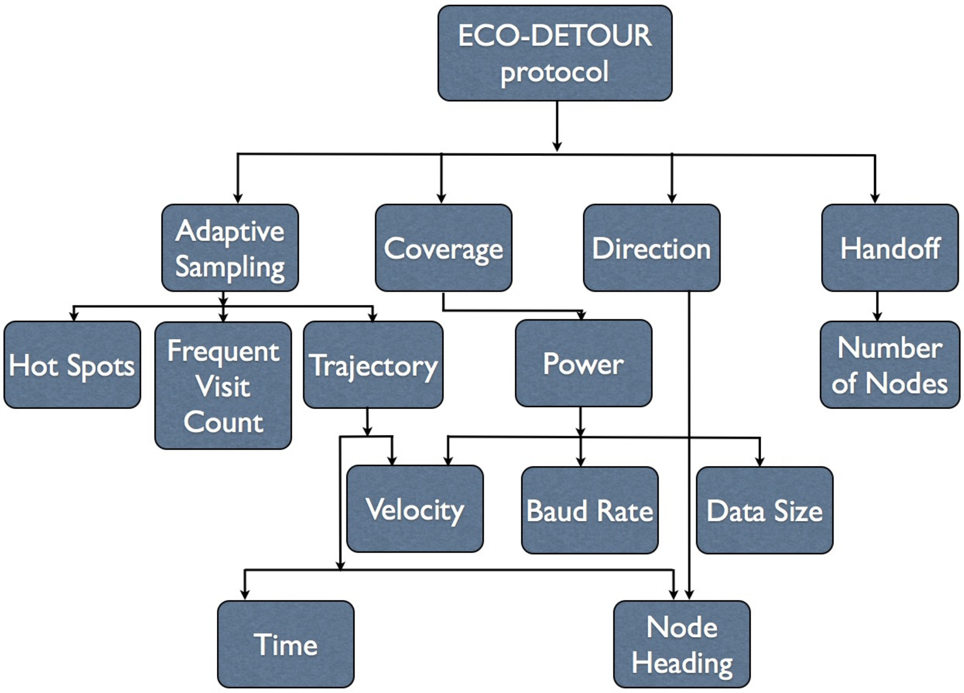


Figure 4.2: Overview of ECO-DETOUR Protocol Parameters

The tracing of trajectory Tr for adaptive sampling of the mobile node requires velocity 'v' and time 't_{ep}', together with the mobile nodes heading 'θ'. It is a function of the tuple as given in equation 4.1. Knowledge of the trajectory of the mobile node helps identify a pattern in the motion of the node. This pattern is learnt to predict the 'hot spots'H_s. A count C is maintained of the frequently visited hot spot to assist in prediction. Adaptive sampling predicts hot spots using a mobile node trajectory and keeps a count of all the successfully predicted hot spots, using a frequent visit count. Therefore, adaptive sampling S_a is expressed as a function, given in equation 4.2.

$$Tr = f(v, \theta, t_{ep}) \tag{4.1}$$

$$S_a = f(Tr, H_s, C) \tag{4.2}$$

The ability to modulate the coverage area of a node, depending on the mobile node velocity, data size and transmission baud rate, helps in transmission of an optimum power sufficient for the successful exchange of data. Therefore, coverage radius Cr can be expressed as a function of mobile node velocity v, data size (D_p) and baud rate (B_r) as given in equation 4.3. This approach to transmission power modulation helps in conserving the limited battery power of the sensor node.

$$Cr = f(v, D_p, B_r) \tag{4.3}$$

Handoff is activated to increase the window of time 't_w' by using multiple stationary nodes. The number of nodes 'H_n' required for successful transmission of data is given in equation 4.4.

$$H_n = t_w v/(2Cr) \tag{4.4}$$

The mobile node queries the stationary node for H_n nodes. If the number of stationary nodes are equal to or greater than H_n, handoff is setup.

The ECO-DETOUR protocols' direction process monitors the mobile node heading θ. The nodes' heading helps in selecting a node which falls in its trajectory, thus avoiding the problem of node overhearing as given in equation 4.5. It can be observed, from equation 4.5, that all nodes participating in handoff should be aligned within tolerance angle δ_θ to the heading of the mobile node. The tolerance δ_θ can be chosen as specific to implementation, e.g in vehicular networks, the heading θ_m of the mobile vehicle may vary between θ_{si} (0°) (heading of the relatively stationary vehicles) *and* δ_θ (±45°), when moving between parallel lanes, and the exit from the lane at 45° *or* 315°. Therefore the vehicle with heading θ_m (0°) can exchange data with all angles less than δ_θ.

$$H_n\theta_m = \{H_1\theta_{s1}, H_2\theta_{s2}, H_3\theta_{s}3, ..., H_i\theta_{si}\} \ \forall \ (\delta_\theta \ \pm \ \theta_{si}) \ \in \ \theta_m \tag{4.5}$$

Figure 4.3 shows the suggested functional flow diagram of the ECO-DETOUR

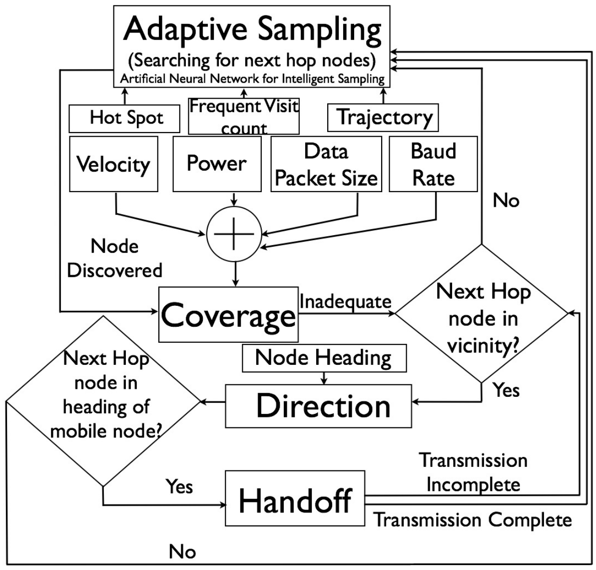


Figure 4.3: Functional flow diagram for ECO-DETOUR Protocol

protocol as deduced from equation 4.1, 4.2, 4.3, 4.4 and 4.5, with the independent variables feeding the functional components. The following sections discuss the design of each functional component in detail.

4.4 Adaptive Sampling

The need for sampling arises as a result of the existence of network holes or the variations in the density of nodes in the network. As shown in Figure 4.4, there are regions where density of nodes is high and other regions where it is low.

Mobile node movement is spatio-temporal as expressed in equation 4.1. Whenever the value of either heading or velocity changes, it is mapped by tabulating with reference to time.

Once the pattern is learnt, beacons are transmitted as the node approaches the hot spot.

Adaptive sampling is performed by a mobile node. It works in two modes; learning and prediction. The process is illustrated by the flow diagram in Figure 4.5. A mobile node that does not have data to exchange or does not have any

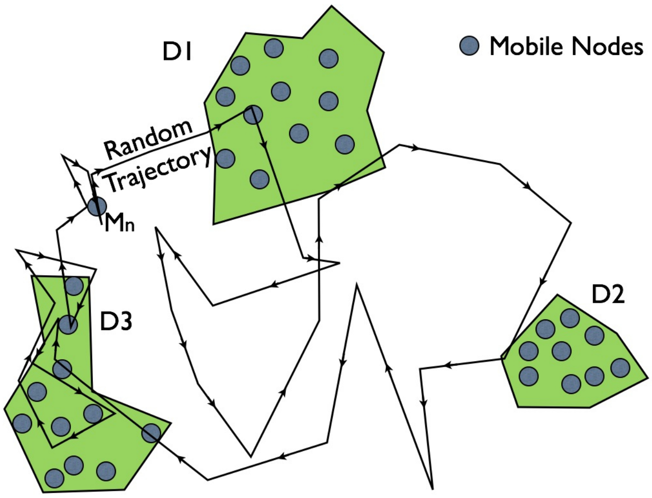


Figure 4.4: Varying Density of Nodes

identified hot spots goes into the learning mode, else it goes into the prediction mode.

Adaptive samplings' learning and prediction require the following:

1. Collection of trajectory data
2. Mapping of the trajectory data
3. Clustering of the map

The collection of the data happens continuously for both learning and prediction. With learning, the complete trajectory data between the source and destination is mapped (set of trajectories) and clustered (stored maps). During learning, a new cluster is created for any newly identified hot spot. Prediction uses incomplete or sub-trajectory data and associates it with the clustered trajectory maps.

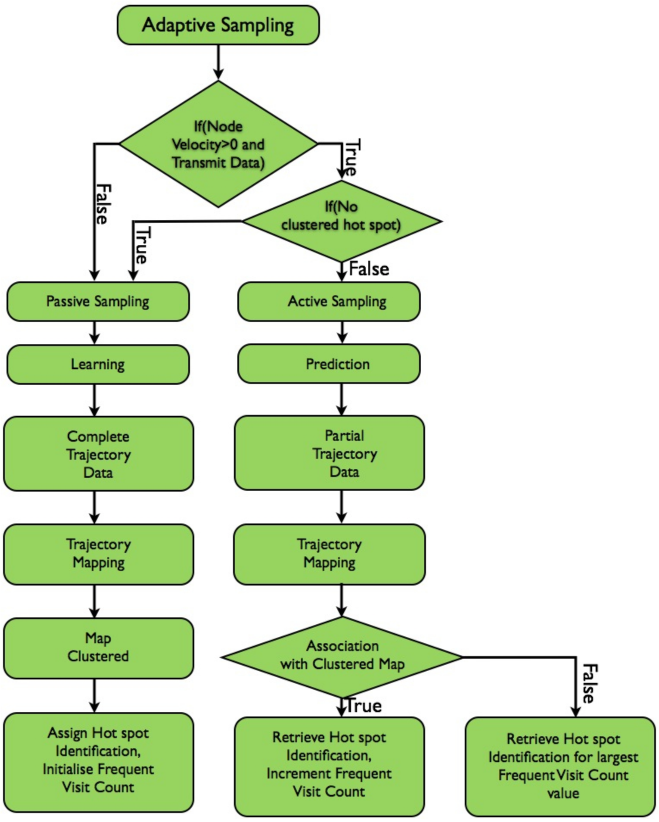


Figure 4.5: Adaptive Sampling process flow diagram

4.4.1 Mapping Spatio-Temporal Characteristics of Node Mobility

A few nodes move in a small area and others move over a larger area. Figure 4.4 shows locations for regions D1, D2 and D3, where the density of nodes is high. M_n is the mobile node that moves across the overall space covering these regions with a random trajectory. The probability of finding a next hop node is quite high in regions D1, D2 and D3. If a mobile node moves out of these regions or if we consider an external mobile node, data exchange is possible only if it comes into any of these regions.

Trajectory mapping is the process of collecting the path traversed by the node between the origin and the hot spot. Using equation 4.1, the map Tr_{mn} can be specified in equation 4.6 as a set of sub-trajectories, Tr_m leading to a particular hot spot H_{sn}. Here it is assumed that the origin of the mobile node always remains the same.

$$Tr_{mn} = \{Tr_1, Tr_2, Tr_3, ...Tr_m\} \in H_{sn} \tag{4.6}$$

Mapping of trajectories is application–dependent and mapping depends on the time it takes for the node to complete a trajectory path.

1. Hot Spot identification :
In sparsely connected areas of a network, the mobile node sporadically has an opportunity to transmit. Node discovery requires transmission of beacons, but at the same time discovery may happen opportunistically by receiving beacons from other nodes. For system initialisation purposes, two modes are used; active and passive sampling. In active sampling, which is performed on expiry of the sampling time, the mobile node transmits a beacon. At other times the node performs passive sampling by turning off the transmitter and allowing the node to listen for any ongoing communication in the vicinity. On overhearing any communication, the mobile node realises the presence of other nodes and prepares itself to establish communication. The process helps to localise regions with a high density of nodes ("Hot Spots") and anchor the hot spot within the trajectory map.

2. Frequent Visit Counts :
The frequent visit count is a counter which is maintained with every hot spot. Every time a particular site is re-visited, the counter is incremented. If, for some reason, a hot spot moves to a different location, and on sampling that particular hot spot the mobile node does not find any response, the counter for that hot spot is decremented.

Frequent visit count establishes a measure of confidence in the input trajectories, and is used to reduce any error in prediction due to slight variations in the current trajectory as compared to the previous trajectory. Frequent visit count 'C' is as shown in Figure 4.6. Amongst the different trajectories 1,2,3 and 4, the mobile node visits hot spot A and B in each of the four trajectories. Therefore the frequent visit count 'C' is incremented successively every time. Hot spot B

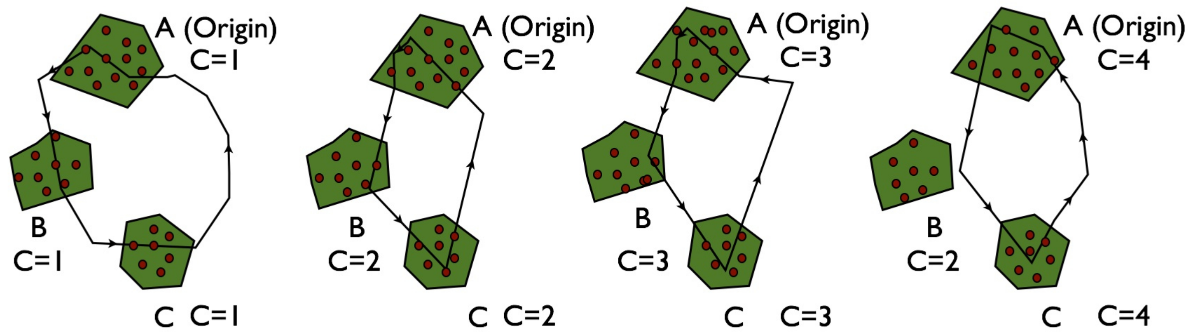


Figure 4.6: Mobile node Trajectories with corresponding Hot Spots and Frequent Visit Counts

is visited three times. On the fourth visit hot spot B moves from the current position and hence, when it is predicted, the prediction is unsuccessful. Therefore the frequent visit count 'C' value is decremented.

4.4.2 Hot Spot Learning and Prediction

The mobile node trajectories are learnt to predict the location of hot spots in the region where data transfer can take place. Artificial Neural Networks have a proven capability to learn and predict from what they have been trained. Figure 4.7 shows the scheme adopted for continuous learning and prediction of the neural network. Continuous learning is a requirement, to constantly create awareness in the network nodes of the ever–changing network demography (migration and movement of network nodes from one location to another). When the node is operational, it only covers those segments of the trajectory which are continuously collected by the continuous learning module. When the distance tends to zero or when triggered by the retraining module, the mapping of the trajectory is fed into the training module.

The artificial neural network algorithm goes through three stages to continuously learn, predict and adapt to the changing requirements of the node and the network. All the three processes are discrete in the design of adaptive sampling for opportunistic connectivity.

- **Training :** For the purpose of training the neural network, the complete

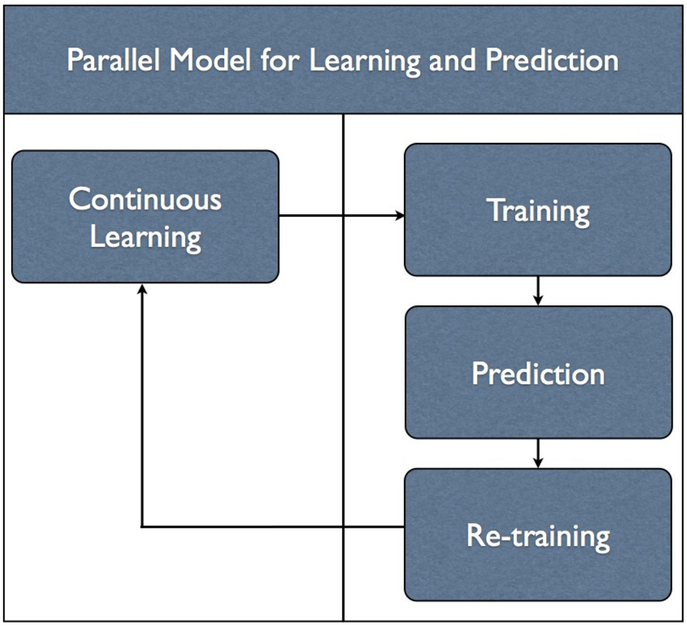


Figure 4.7: Parallel Model for Learning and Prediction

trajectory information is created as a map and fed for training. The neural network is trained once it discovers a hot spot.

- **Prediction :** During prediction, the input data comprises of a subset of the trajectory. The sub-trajectory is only an approximation of the trajectories for which the network is trained. The accuracy of prediction depends on the length of the input trajectory segment.

- **Retraining :** The retraining is a trigger which gets fired if a new hot spot is discovered through passive sampling and if the current input trajectory sample does not match a cluster. The retraining module is connected to the continuous learning module, which transfers collected data to the training module for adding a new cluster to the neural network.

The neural network algorithm clusters the input. Every cluster denotes a hot spot. Therefore, when a new hot spot is processed for clustering, a new cluster is created. The neural network continuously learns and clusters whenever it encounters a new hotspot. Whenever a new data set is presented which is not available in the cluster, a new cluster is formed. If the new data set is similar to a pre-existing cluster then the neural network clusters them together.

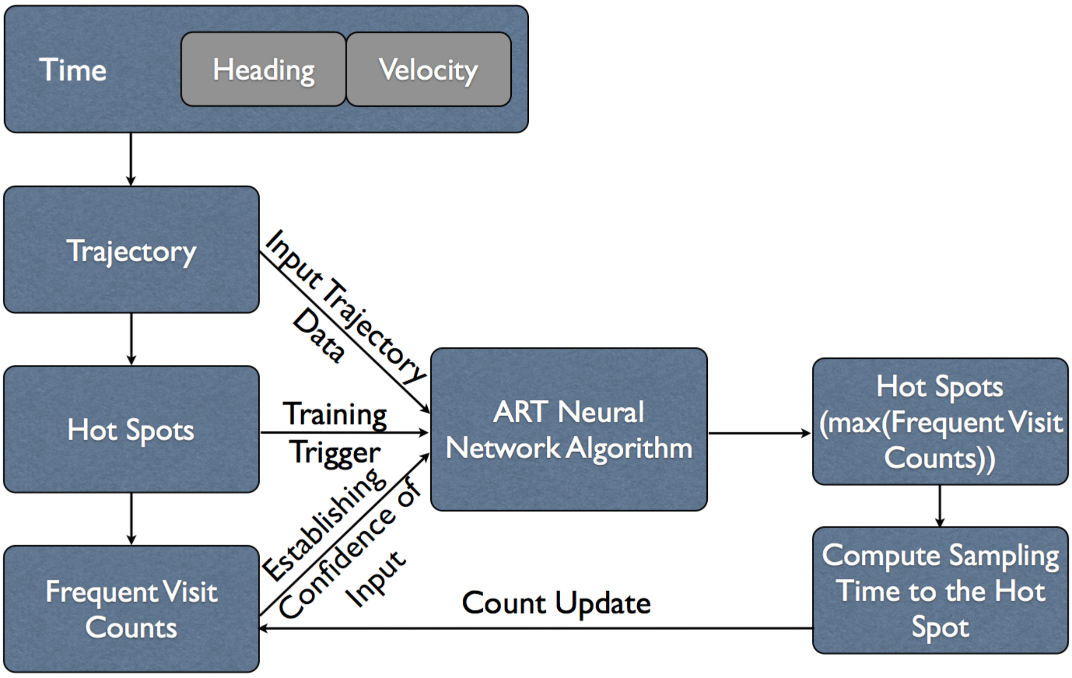


Figure 4.8: Adaptive Sampling system organisation

The different input and output variables and their relation is shown in Figure 4.8. Mobile node trajectory, Hot spot and frequent visit count, all form input for the training of the ART1 neural network algorithm. The output of the artificial neural network is the predicted hot spot. The frequent visit count 'C' is incremented every time the predicted hot spot is successfully sampled. Otherwise each time the hot spot is missed, the count is decremented. The use of neural network for training and prediction requires data to be organised so it can be clustered. The trajectory tuple (Heading 'θ', Velocity 'v', Time 't'), is tabulated along with the Hot spots 'H_{sn}' and its corresponding frequent visit counts 'C_n' as shown in Figure 4.9 (a). As the ART1 neural network requires data to be represented as binary strings for clustering, the trajectory tuple (θ, v, t) is modelled as pixels on a 2-D map and tabulated along with hot spots H_{sn} and frequent visit count C_n as shown in Figure 4.9 (b). The neural network is trained with the complete trajectory data. During prediction, one segment of the trajectory is presented to the neural network. Therefore, for the purpose of prediction, the input trajectory

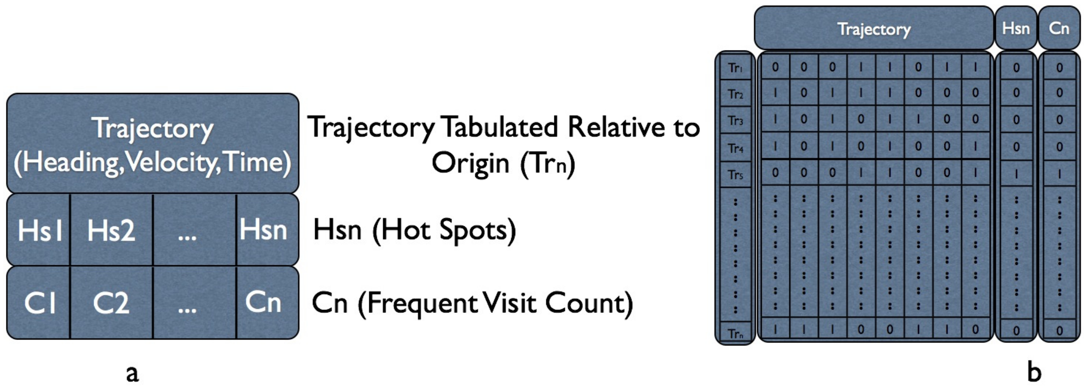


Figure 4.9: Structuring of Trajectory data for clustering

segment has to be matched with the clustered trajectory segment.

Computation of Sampling Time :

Sampling time is the time, on expiry of which, a beacon is transmitted to search for a next hop node. Computation of sampling time is based on the prediction of the hot spot. The distance of the predicted hot spot d_{hs} from the origin is tabulated in a table being maintained for monitoring the trajectory. The current distance d_c of the node from the origin is continuously monitored. The difference between the two distances d, given in equation 4.7, when divided by velocity gives the time to sample t_{sa}, given in equation 4.8:

$$d = (d_{hs} + \delta d_{hs}) - d_c \tag{4.7}$$

$$t_{sa} = d/v \tag{4.8}$$

There may be small deviations δd_{hs} in the current trajectory of the mobile node in comparison with the previous trajectory. These deviations would affect the previously computed distance to a hot spot d_{hs}. Therefore, these deviations are added to d_{hs} as tolerance. As the mobile node moves towards the registered hot spot, the value of d diminishes. Beacons should be transmitted when the mobile node is close to the hot spot. Hence sampling is performed when $t_{sa} \to 0$.

Velocity is read from the sensors. Velocity being a dynamic variable, there would be error in the computation of d_c, which would lead to the erroneous estimation of t_{sa} as $t_{sa} \rightarrow 0$. As velocity increases the error increases. If the frequency of polling of the sensor to read velocity f_{vs} is increased, d_c would be computed more often, making the estimation accurate. Therefore, the computation of d_c can be expressed in terms of the frequency of polling the velocity, as given in equation 4.9. The term d_{c0} is the past value of d_c.

$$d_c = d_{c0} + \sum_{d_c=origin}^{d_c=d_{hs}} \frac{v}{f_{vs}} \tag{4.9}$$

Computational Complexity :
The computational complexity of Adaptive sampling depends on the number of clusters and the dimensions of the 2D trajectory map. The clustering and prediction of the trajectories require three nested loops. The complexity can be denoted by $O(n^3)$. 'n' is the largest index of the trajectory map. Computational complexity of other probabilistic opportunistic connectivity protocols is $O(1)$ for computing the probability of another node as data carrier.

According to Pottie and Kaiser [108], Liang and Peng [109] approximately 3000 instructions can be processed for the amount of energy spent in transmitting a bit over a distance of 100 m. As the probabilistic opportunistic connectivity protocol needs to establish communication to exchange the probability, therefore the total cost for transmitting x bits and computing the probability would be a product of $(3000x + 1/3000)$ and Number of encounters. With ECO-DETOUR the total cost only relates to computation which is $\sim n^3/3000$ for the highest order polynomial. At $n = 208$ the cost of computation of ECO-DETOUR equals the cost of exchanging the probability value to determine a data carrier for a single encounter.

4.4.3 Adaptive sampling's Energy analysis

A sampling estimation error leads to the transmission of beacons as the mobile node approaches the hot spot as shown in Figure 4.10.

This leads to a situation where the mobile node might overshoot the hot spot

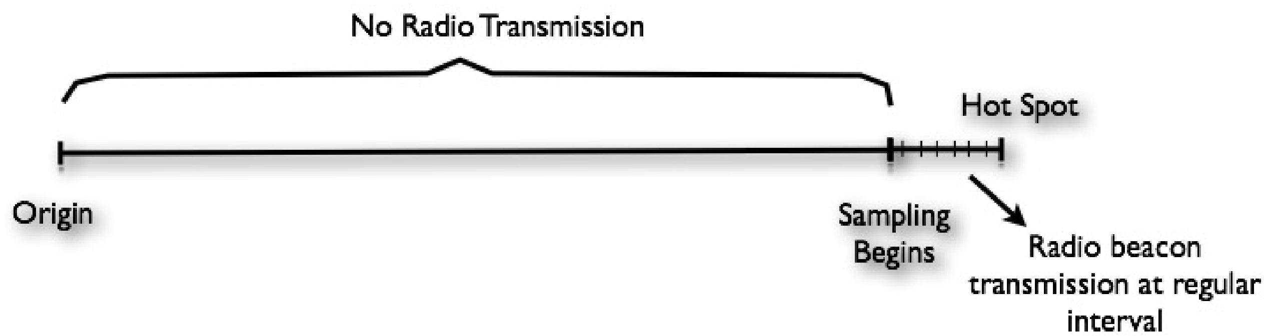


Figure 4.10: Radio Beaconing for Hot spot discovery

region without sampling. To avoid this, the strategy allows for the beacons to be transmitted at regular intervals as the node approaches the hot spot.

$$E_{ee} = E_{bit} N_{bb} \tag{4.10}$$

The excessive energy expense E_{ee} can be expressed in terms of transmission energy required per bit E_{bit}, and number of beacon packets N_{bb}, as expressed by equation 4.10.

4.5 Coverage Radius Model

Once the hot spot is successfully discovered through adaptive sampling, it is important to maximise the data transfer. As given in equation 4.3, coverage is a function of velocity, data size and baud rate. The tuple (v, D_p, B_r), affects the window of time, which is used to modulate the transmission power for modulating the coverage radius, as shown in Figure 4.11.

When a node is mobile, the window of time t_w spent in the vicinity of another node reduces with increasing velocity. Considering the stationary node as a point object, the coverage radius of the mobile node Cr is considered as the distance moved by the mobile node. This available window of time is given in equation 4.11.

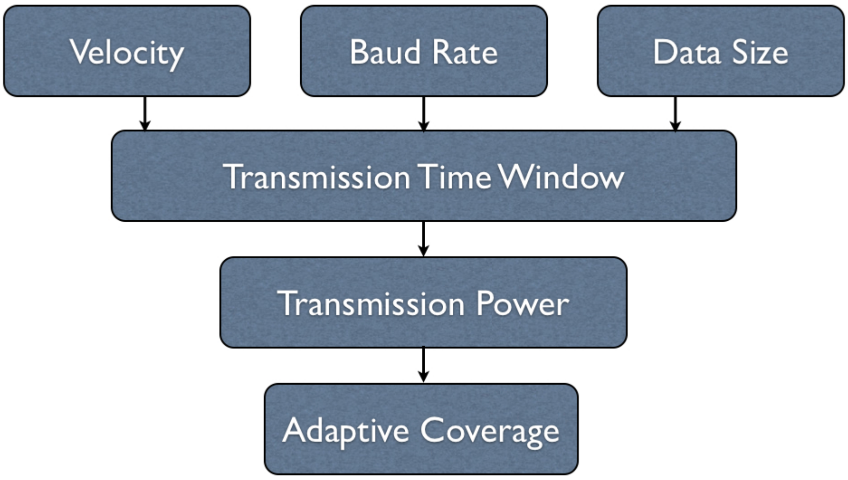


Figure 4.11: Parameters of Adaptive Coverage

$$t_w = 2Cr/v \tag{4.11}$$

The transmission baud rate B_r affects the number of packets D_p that can be transmitted in a given window of time. Equation 4.12 gives the required window of time t_{wr} to successfully transmit the complete data.

$$t_{wr} = D_p/B_r \tag{4.12}$$

Equating equation 4.12 with equation 4.11, we get the coverage radius Cr of a mobile node with respect to velocity as equation 4.13

$$Cr = \frac{D_p v}{2B_r} \tag{4.13}$$

4.5.1 Coverage Power Modulation

Friis Transmission equation 4.14 gives the coverage radius R at a given transmission power P_t:

$$R = \sqrt{\frac{P_t G_t G_r \lambda^2}{(4\pi)^2 \mathrm{S}_m}} \tag{4.14}$$

$P_t = Transmission\ power,\ G_t = Transmitter\ gain,$
$G_r = Receiver\ gain,\ \lambda = Wavelength,$
$S_m = Receiver\ sensitivity$
As the radiation pattern considered is omnidirectional, hence the gain $G_t = G_r = 1$.

Making $Cr = R$, one can express transmission power P_t of equation 4.14 as equation:

$$P_t = \frac{\left(Cr^2 \left(4\pi\right)^2 \mathrm{S}_m\right)}{\lambda^2} \tag{4.15}$$

Substituting for Cr from equation 4.13, transmission power P_t is expressed as in equation 4.16.

$$P_t = \frac{\left(\left(\frac{D_p v}{2B_r}\right)^2 \left(4\pi\right)^2 \mathrm{S}_m\right)}{\lambda^2} \tag{4.16}$$

Factors such as the velocity of the mobile node, data size and baud rate affect the transmission power through equation 4.16, thereby modulating coverage according to changing requirements. This work considers transmission to be line of sight and the transmission losses considered are free space losses.

4.6 Handoff

The inadequacy of coverage due to large data packets and/or high velocity of mobile nodes leads to the requirement of data handoff over multiple nodes, so that complete data is captured. Handoff is required when the window of time for coverage of a single node is not sufficient for transmission of all the required data. To extend the window of time, the data has to be transferred over multiple stationary nodes N_s. As the stationary nodes collaborate to establish handoff, they need to identify the nodes present in the vicinity through neighbour discovery.

This neighbour discovery table contains the Id's of immediate next hop neighbours, as well as further hop neighbours, as shown in Figure 4.13. The process is

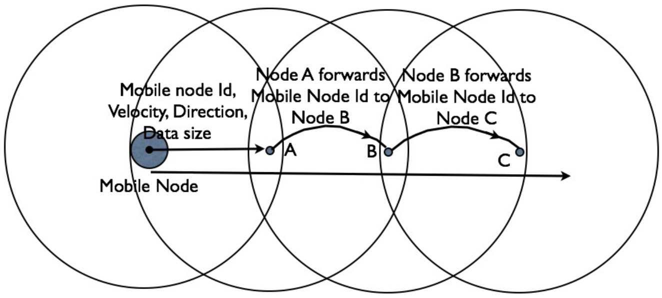


Figure 4.12: Handoff Scenario

part of the data exchange. This ensures that the table is always updated with the latest nodes in the vicinity, which get shared during data exchange. The created

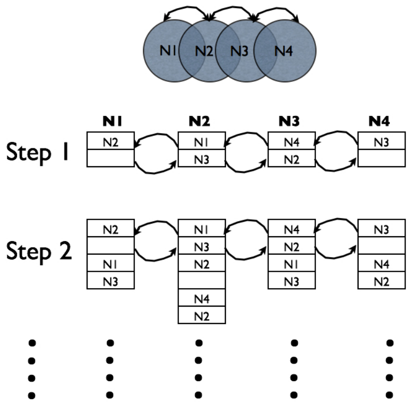


Figure 4.13: Creation of Neighbour Id's Table

tables are now used by the mobile node to determine the number of connectivities. As shown in the Figure 4.13, during the creation of the tables there are many duplicate id's and the id of 'self' (i.e the nodes' own id), which are removed. A 'count' is determined of the number of nodes from the table available for handoff. During handoff initiation, the mobile node transmits a request for the number of nodes it needs for handoff. With the knowledge of the 'count' value, the first node providing handoff to the mobile node knows exactly the number of nodes that are available for handoff that would later participate in providing handoff.

The handoff requires the mobile node M_n to transmit the tuple (*node id*, *velocity*, *direction of motion*, *data packet size*).

Using this information, the stationary node informs their next hop nodes

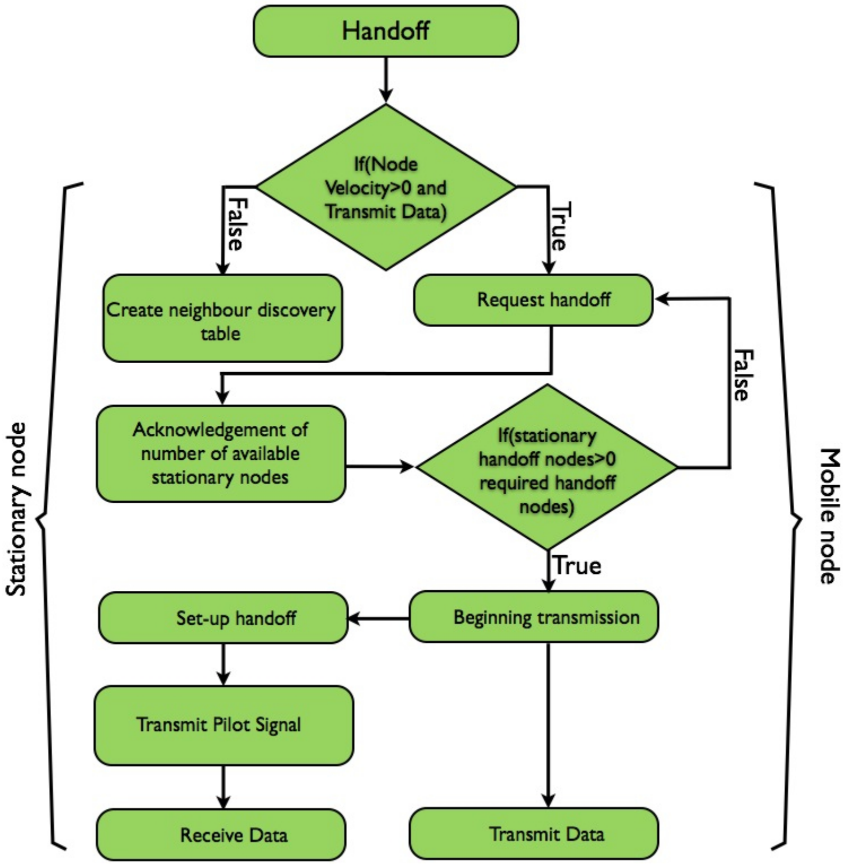


Figure 4.14: Handoff: flow diagram

about the incoming node, as shown in Figure 4.12. Figure 4.14 shows the flow of different handoff signals between stationary and mobile nodes for the establishment of handoff. The mobile node computes the number of nodes it needs, H_n, using equation 4.4, which is based on the window of time that determines the overall distance required at current velocity. As it is assumed that the stationary nodes know their neighbours through neighbour discovery, the first node receiving the control signal computes the number of nodes and returns an acknowledgement to the mobile node. The acknowledgement consists of the number of nodes available for handoff, H_{na} . The mobile node checks for the condition as given in equation 4.17. If the condition is true, the mobile node transmits a '$Beginning\ Transmission$' signal to inform the stationary node to propagate the pilot signal. After transmission of '$Beginning\ Transmission$', the mobile node starts data transmission.

$$H_n \leq H_{na} \tag{4.17}$$

Figure 4.15 shows signalling sequence (with respect to time) amongst the mobile and stationary nodes for the handoff process.

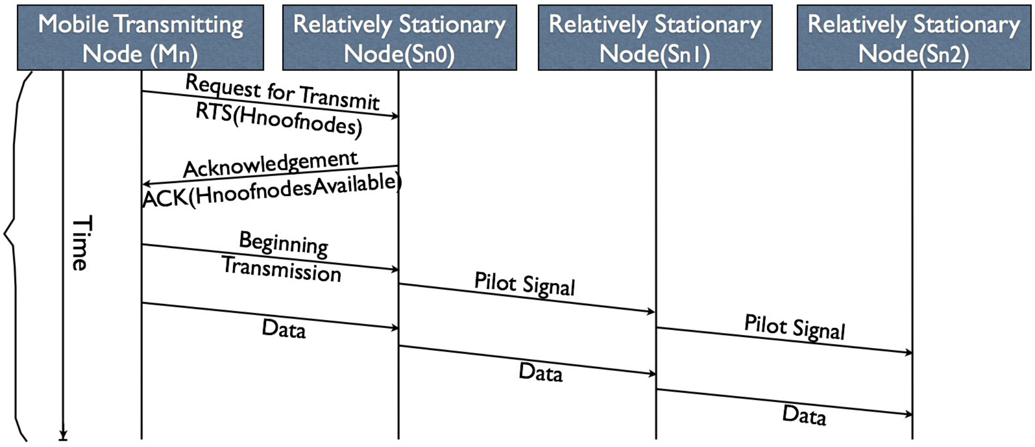


Figure 4.15: Handoff Control Signal Exchange

The first stationary node encountered by the mobile node exchanges most of the control signals for establishing handoff. The other stationary nodes only receive the pilot signal and become active to accept data.

Pilot Signal :

The Pilot signal is used to alert other nodes of an incoming mobile node, when the mobile node transmits the '*Beginning Transmission*' signal. The receiver node issues a handoff set up signal to the other stationary nodes. This signal consists of the tuple (Mobile node Identity M_nId, Mobile node Velocity v, Direction of Motion θ and a Decrementing counter C_D). The decrementing counter (C_D) stores the value H_{na}. The value of C_D is decremented and forwarded until it becomes zero as shown in Figure 4.16.

Once 'count' becomes zero the pilot signal stops propagating in this way, all the nodes '$N1 - N4$' are alerted to a mobile node 'M_n' and become ready to receive its data.

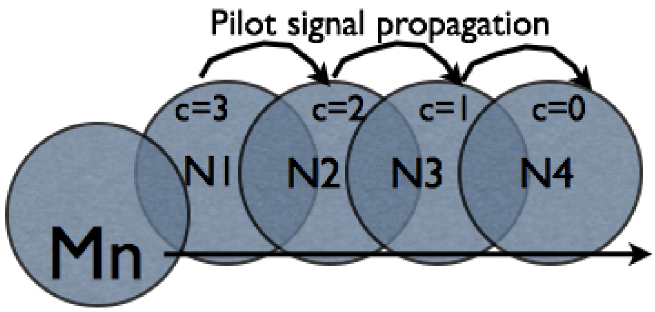


Figure 4.16: Propagation of Decrementing Counter ‘c’ in the Pilot signal

4.7 Direction of Motion

The direction of motion of a mobile node is important in selecting the appropriate node in dense regions of a network. In sparse networks, direction is used in mapping trajectories. Node overhearing (which has a direct impact on the battery life of the node) is avoided by using the direction of motion of the mobile node. Figure 4.17 gives the flow diagram for the direction process. The adaptive sampling module constantly polls for direction to create trajectory maps and the handoff module monitors direction to establish node orientation with respect to neighbouring nodes.

Pilot signal sequentially propagates amongst the subsequent stationary nodes. If a stationary node is headed in a direction different from the mobile node and other stationary nodes, it is possible that it may not be a suitable data carrier, as it may have temporarily come into the hot spot, as do mobile nodes, and would leave the hot spot in a short interval of time. If data is transmitted to this node it would be able to receive it, as it would be in the radio range, but the data may not reach its destination, resulting in loss of network energy.

To avoid this unwanted energy expenditure stationary nodes turn themselves off by comparing the direction received through the pilot signal with their own heading, as given in equation 4.5.

As shown in Figure 4.18 (a), each mobile node knows its own direction to head, hence the final heading of a node with respect to another node would give a bearing between the two nodes, as shown in Figure 4.18 (b). Assuming node A

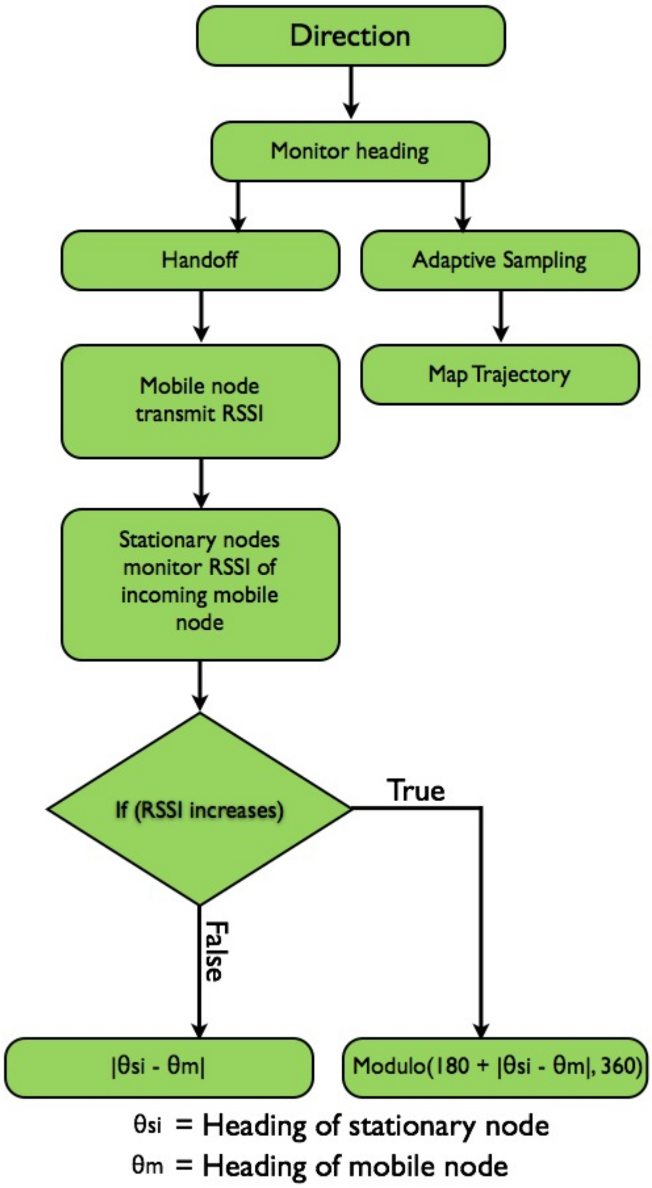


Figure 4.17: Handoff of Mobile Node with Random Trajectory

is the node monitoring nodes B and C, the relative bearings are found by taking the difference of the approach heading of the mobile node (B, C) from node A. As with an omnidirectional radiation pattern, signal strength is ideally the same in all directions. Hence, when two or more nodes are in a vicinity, they cannot determine their angular positioning.

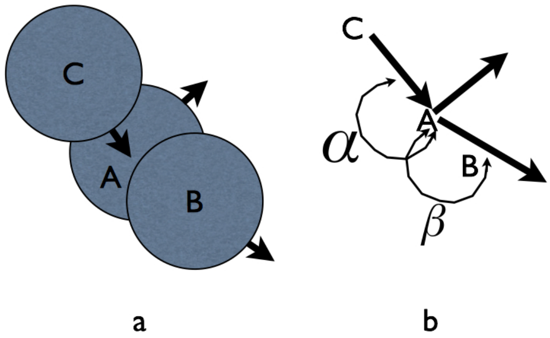


Figure 4.18: Node Headings: (a) Nodes' know own heading. (b) Bearing with respect to other nodes

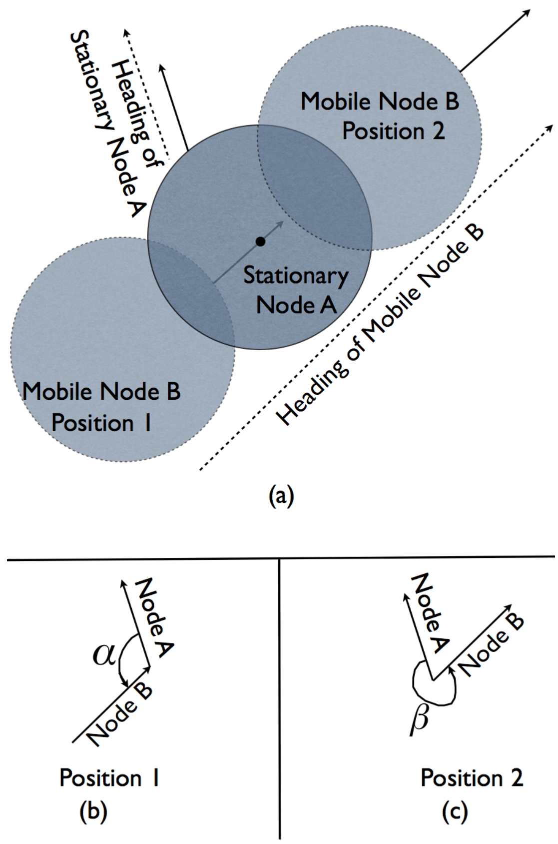


Figure 4.19: Direction Estimation at Different Positions: (a)Different node positions. (b) Angle at position 1. (b) Angle at position 2

The heading of mobile node B is constant with respect to node A, but its bearing is dependent on its position (mobile node B) in the coverage of node A (shown in Figure 4.19 (a)). Figure 4.19 (b and c) shows the different headings for position 1 (angle is α) and for position 2 (angle is β), respectively.

By constantly monitoring the received signal strength, the stationary node is able to determine whether the mobile node is coming towards it or moving away from it. Based on whether the mobile node is approaching the stationary node or moving away from it, $\angle\alpha$ or $\angle\beta$ may be selected as the bearing between mobile node and stationary node.

4.8 Conclusion

The chapter discusses the ECO-DETOUR protocol design. The four functional components (adaptive sampling, coverage, handoff and direction) for the protocol are determined and their relationship with the independent variables are established.

Adaptive sampling discovers hot spots and later predicts them. This is done by collecting the velocity and heading data, with respect to time, to create trajectory maps. Trajectory maps are fed to the neural network algorithm for clustering and prediction.

On successful discovery of a hot spot, the coverage area is modulated by modulating transmission power to increase the window of time for data transfer. The transmission power is modelled with velocity, data size and transmission baud rate as inputs. To handle situations where the coverage of a single node cannot provide the required window of time, handoff is activated. Handoff determines the number of next hop nodes required to meet the window of time. The stationary nodes providing handoff keep a count of their neighbours. Handoff with random motion necessitates constant monitoring of the heading. This helps in waking up appropriate nodes and avoids node overhearing.

The direction module monitors the nodes' heading, which is used in trajectory mapping and handoff. It helps avoid node overhearing in handoff by monitoring the relative heading and received signal strength.

Chapter 5

Modelling and Simulation of ECO-DETOUR protocol

5.1 Introduction

This chapter discusses the implementation of the ECO-DETOUR protocol. The OPNET simulator and MATLAB have been used as the main tools for developing the environment. OPNET is a good discrete event–based network simulator, but lacks a strong mathematical simulation framework. MATLAB offers a better mathematical environment. This chapter discusses the components of the ECO-DETOUR protocol and then details the modelling of the key components. The model implementation and the testing of the individual components is presented. This chapter also discusses the savings in energy when establishing communication and the improvement rate for a successful packet delivery within a sparse network.

5.2 Generic Node Implementation

The four functional activities (adaptive sampling, coverage, handoff and direction), of the protocol are modelled as 'Processors' within the OPNET node model. A processor in OPNET is a self–contained entity performing a designated operation. Different processors and the wireless radio are connected via 'Streams' that carry information amongst them. The wireless radio enables communication with other nodes in the network. The implementation of the process activation of a node is shown in the Figure 5.1, and the pseudo-code detailing the activation of the different processors is given by algorithm 5.1.

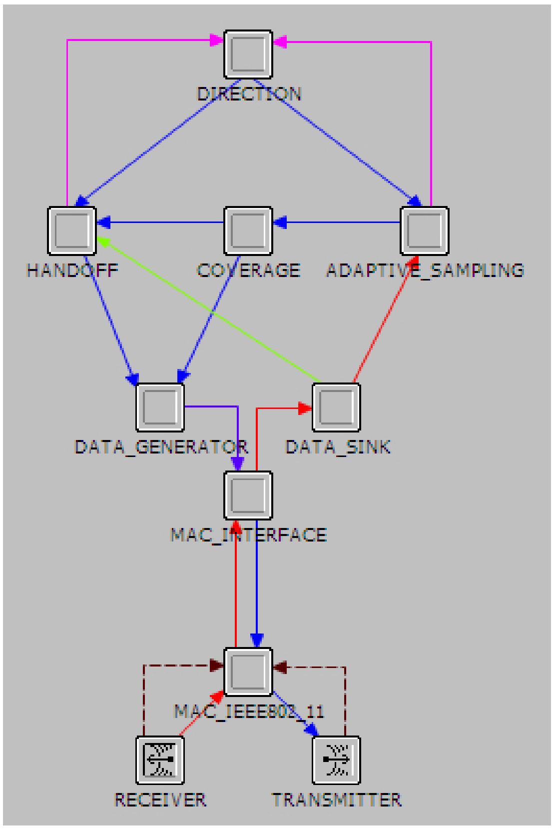


Figure 5.1: Implementation of ECO-DETOUR Protocol using OPNET

```
if v > 0 Km/hr then
    if Packet_receive == TRUE then
        Collect the packets at the Data_Sink Processor;
        if Packet == Neighbour_Discovery then
            Trigger (Adaptive_Sampling Processor);
        Trigger (Direction Processor) goto adaptive_sampling;
        else if Packet == Handoff_acknowledgement ∧ H_na > H_n then
            Transmit (Beginning Transmission);
        Trigger (Data_Generator Processor);
    else if Packet_receive == FALSE then
        adaptive_sampling:
        Trigger (Adaptive_Sampling Processor);
        Trigger (Coverage Processor);
        if Coverage Power < Coverage_threshold_power then
            Trigger(Data_Generator Processor);
        else
            Trigger (Handoff Processor);
            Trigger (Direction Processor);
            Transmit (Handoff Request);
        end
    end
else
    if Packet_receive == TRUE then
        if Packet == Handoff_Request then  Transmit (Handoff
        Acknowledgement);
        else if Packet == Pilot_Signal ∨ Packet == Beginning
        Transmission then
            Trigger (Handoff Processor);
            Trigger (Direction Processor) ;
            if Decrementing_Count > 0 then
                Transmit (Pilot_Signal);
                Decrement (Decrementing_Count);
            end
        else if Packet == Neighbout_Discovery then
            Trigger (Data_Generator);
        else
            Transmit (Neighbour_Discovery);
            Trigger (Data_Generator);
        end
    end
end
```

Algorithm 5.1: Pseudo-code for implementation of ECO-DETOUR Protocol in OPNET

The protocol considers a node to be either mobile or stationary (relative velocity zero). If the nodes are stationary they get the opportunity to exchange neighbour discovery signals and data more often as they are in each other's vicinity. This results in the presence of a large amount of electromagnetic radiation in the region. Therefore, mobile nodes looking for hot spots can sense this radiation and begin communication. To take advantage of this scheme, the nodes' data generator is activated through the adaptive sampling processor, which is activated by the receiver's sink processor, as shown in the Figure 5.1. The pseudo-code shown in algorithm 5.1 manage the different processors by monitoring the mobility of the node. If the node is mobile and receives a neighbour discovery packet, it means there is transmission happening in the vicinity. As the node has discovered this region through passive sampling, it triggers the adaptive sampling processor to cluster this hot spot. Once a hot spot is detected through adaptive sampling, the adaptive sampling processor triggers the coverage processor, which computes the required transmission power for successful communication. The coverage processor triggers the handoff processor if the computed transmission power is greater than the preset threshold power, or else it triggers the data generator processor. If the handoff processor is triggered, then it triggers the data generator to transmit a handoff request.

On the receiver side, once a node receives the handoff request, it responds with a handoff acknowledgement. The mobile node, on receiving the handoff acknowledgement, compares the number of nodes it needs for handoff with the number of nodes available, which is sent via the handoff acknowledgement. If the number of nodes available are greater than the required number of nodes, a 'Beginning transmission' signal is transmitted. On receiving the 'Beginning transmission' signal, stationary nodes transmit a Pilot signal. When a node receives the Pilot signal, the handoff processor is triggered directly and bypasses adaptive sampling and coverage. The Pilot signal propagates until the decrementing counter is greater than zero. The processor for direction is connected to the adaptive sampling processor and handoff processor. The dual interconnections provide 'heading' to adaptive sampling for trajectory mapping and to handoff for selecting the next hop handoff node.

5.3 Adaptive Sampling Component Modelling

The 'Data Sink' processor triggers the Adaptive sampling process once the received signal is of a 'neighbour discovery' packet type. The Adaptive sampling process has two concurrent activities, initiated by two triggers. These are motion trajectory data acquisition and hot spot prediction. Triggers T1 and T2 are

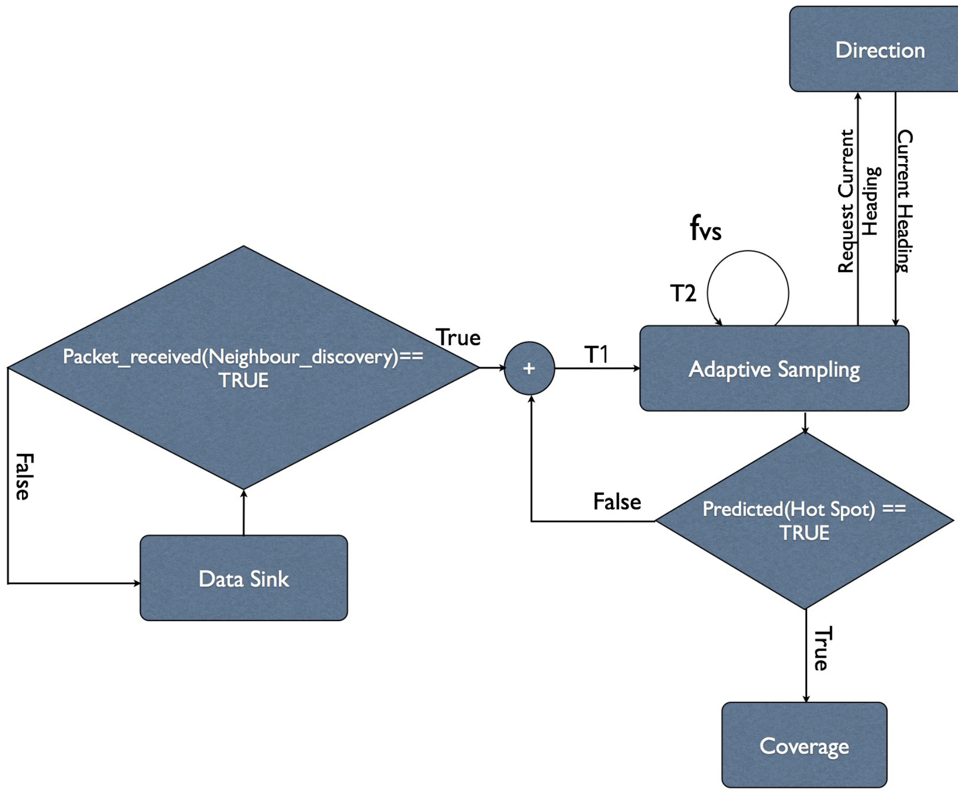


Figure 5.2: Process of initiating Adaptive Sampling component

shown in Figure 5.2. Under normal operation, the T2 mode is activated. This constantly monitors the velocity and heading with respect to time.

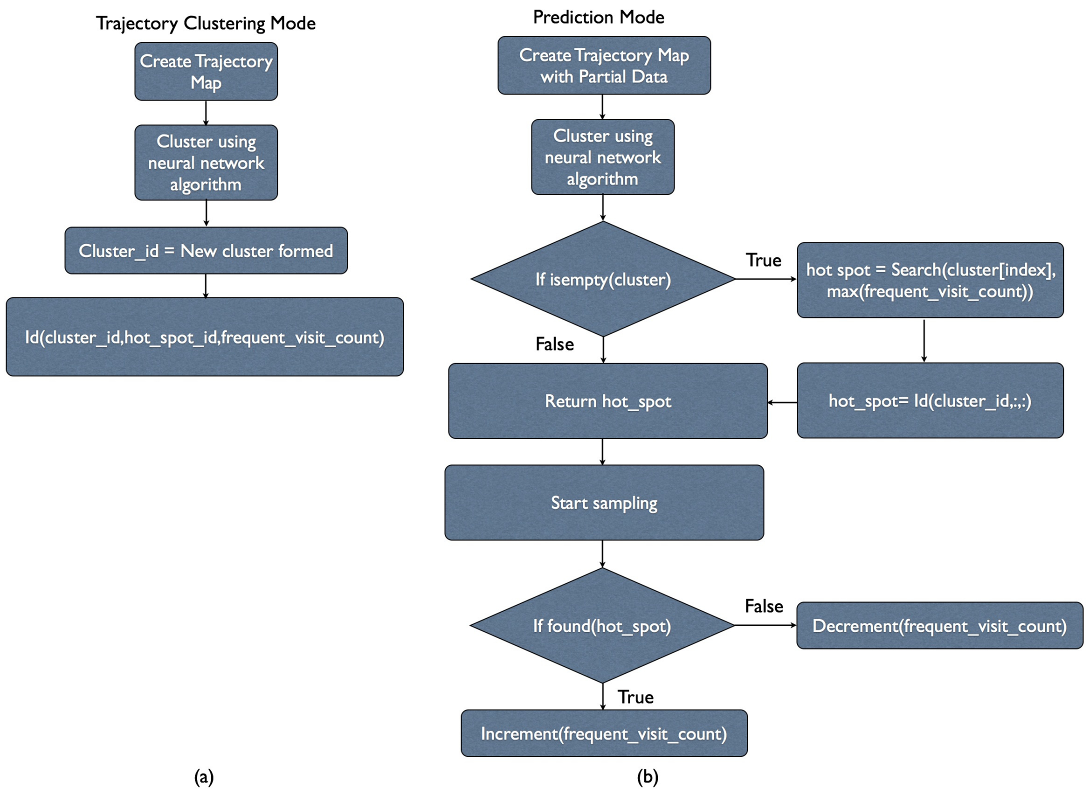


Figure 5.3: Flow diagrams for Clustering and Prediction: (a) Trajectory clustering mode. (b) Trajectory prediction mode.

```
Data: v_t ← 0, v_{t−1} ← 0, θ_t ← 0, θ_{t−1} ← 0
Data: t_t ← 0, t_{t−1} ← 0, t_ep ← 0
// t_t, t_{t−1}, v_t, v_{t−1}, θ_t, θ_{t−1} are current and past, time, velocity
   and heading respectively
top:
if Packet_receive == FALSE then
    v_t ← sensor_read(v);
    θ_t ← sensor_read(θ);
    // sensor_read operates at frequency f_vs
    if (v_t − v_{t−1}) > 0 ∨ abs((θ_t − θ_{t−1})) > 0 then
        read_time(t_t);
        t_ep ← t_t − t_{t−1};
        v_{t−1} ← v_t;
        θ_{t−1} ← θ_t;
        t_{t−1} ← t_t;
    end
else if Packet_receive == TRUE then
    Collect the packets at the Sink;
    if Packet == Neighbour_Discovery then
        Trigger(Adaptive_Sampling Processor) goto
        adaptive_sample_clustering;
    end
else if Data_Transmission == TRUE then
    Trigger(Adaptive_Sampling Processor) goto
    adaptive_sample_prediction;
else
    goto top;
end
adaptive_sample_clustering: goto map cluster;
Frequent_visit_count ← 1;
adaptive_sample_prediction:
goto map cluster;
hot_spot_id ← Retrieve(Id(Cluster_id));
map cluster:
Create Trajectory_Map(Traj_Map(v_t, θ_t, t_ep));
Cluster_id ← ART1(Traj_Map);
if hot_spot_id != NULL then
    Increment(frequent_visit_count);
else
    hot_spot_id ← Retrieve(Id(max(frequent_visit_count)));
    Decrement(frequent_visit_count);
    goto adaptive_sample_prediction;
end
```

Algorithm 5.2: Pseudo-code for Clustering and Prediction

T1 trigger signifies that the mobile node has received a 'neighbour discovery' packet through passive sampling of the network. On receiving the T1 trigger, the collected data is clustered by the neural network algorithm. Figure 5.3 gives the flow diagram for the clustering. The prediction performed by adaptive sampling is given by algorithm 5.2. The condition $Packet\_receive =$'$FALSE$', signifies no packet is received at the 'Data_Sink' processor and therefore the mobile node constantly samples its velocity and heading at frequency f_{vs}, as given by equation 4.9. The node monitors the changes in velocity or heading through the local sensors. These are tabulated with respect to the time elapsed since the last tabulation of data (velocity and heading). When the node starts receiving data packets, the Data_Sink processor analyses the type of packet. If the packet received through passive sampling is a neighbour_discovery packet, it means that the mobile node has come into a hot spot region.

On discovering a hot spot, the mobile node goes into the clustering mode to cluster the trajectory as given in the flow diagram Figure 5.3(a).This region is then updated as a 'hot spot' and initialised with a 'frequent visit count' value of '1'. The prediction can be performed once the node has gathered some trajectory data and clustered it through the artificial neural network algorithm. Each cluster is a trajectory to a hot spot. If the current partial trajectory map matches with a cluster, then the index value of that cluster is used to identify the hot spot. Prediction mode is activated if the mobile node needs to transmit data and is away from the hot spot region. The flow diagram is given in Figure 5.3(b). Both modes (i.e. clustering and prediction modes) require trajectory mapping. In the clustering mode, the neural network algorithm clusters the trajectory map and assigns a hot spot identification to it. In the prediction mode, the neural network algorithm matches the partial trajectory data with the clusters. If a match is found, the hot spot identification is retrieved, and the node follows equation 4.8 and equation 4.9 for sampling the hot spot. Alternatively, if a match is not found, then the hot spot associated with the largest frequent visit count value is taken.

The trajectory data (velocity and heading at different times) is received from the sensors and used, together with time elapsed, to work out the node's location. The recorded information, over time, formulates the parts of the trajectory map. The ART1 neural network is used for the clustering of the trajectories. As the algorithm accepts a binary string as inputs, the distance and heading needs to be pre-processed into a binary string.

To create a binary string, a 2D matrix filled with zeros '0' is taken. The size of the matrix depends on the size of the trajectory. Each cell of the matrix is considered as a displacement of $1m$. A sample matrix with a trajectory is shown in Figure 5.4, which depicts 50x50 cells covering an area of 50 meter x 50 meter.

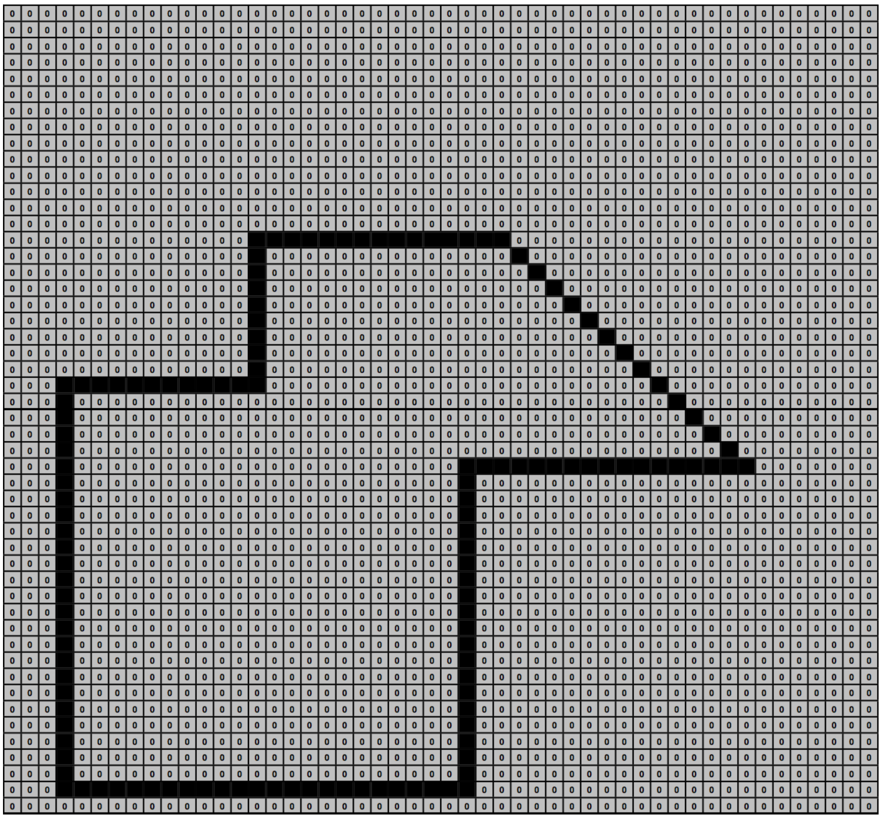


Figure 5.4: Matrix with tracing of trajectory

As the node moves, the squares which fall on the trajectory are filled with '1'. The velocity of the node is in meter per seconds, 'mps'. The pseudo-code for creating the trajectory map is given in algorithm 5.3. The current implementation consists of 8 angles for heading ($0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°$ *and* $360°$). More angles can be accommodated by decreasing the least count of the cells from the current $1m$ to $10cm$ or even less. This implementation uses trajectory data passed to MATLAB by the OPNET model. The data is passed through a .csv

```
Initialise Traj_map[m,n] ← 0;
Data: θ ← sensor_read(heading)
Data: v ← sensor_read(velocity)
// x_t, y_t and x_{t+1}, y_{t+1} are current and future position on x
   and y axis in the Euclidean space
d ← v * t_ep;
for i ∈ d do
    switch θ do
        case 0°
            x_{t+1} ← x_t + 1, y_{t+1} ← y_t
        case 45°
            x_{t+1} ← x_t + 1, y_{t+1} ← y_t + 1
        case 90°
            x_{t+1} ← x_t, y_{t+1} ← y_t + 1
        case 135°
            x_{t+1} ← x_t − 1, y_{t+1} ← y_t + 1
        case 180°
            x_{t+1} ← x_t − 1, y_{t+1} ← y_t
        case 225°
            x_{t+1} ← x_t − 1, y_{t+1} ← y_t − 1
        case 270°
            x_{t+1} ← x_t, y_{t+1} ← y_t − 1
        case 315°
            x_{t+1} ← x_t + 1, y_{t+1} ← y_t − 1
        case 360°
            x_{t+1} ← x_t + 1, y_{t+1} ← y_t
        endsw
    endsw
end
for x ∈ x_{t+1} do
    for y ∈ y_{t+1} do
        Traj_map[x,y] ← 1
    end
end
```

Algorithm 5.3: Trajectory Map Matrix

file. Each node in the network maintains its own .csv file for creation of its own trajectory.

The created trajectory map is a 2D binary matrix. The neural network processes a stream of binary input values. Therefore the 2D matrix has to be converted into a one–dimensional row vector representing the definition of the trajectory on the defined space.

The trajectory size would be different for each mapped trajectory. This leads to the problem of different sized inputs. The difference in vector size affects the processing ability of the neural network algorithm. To address this, the vectors are buffered with zeros at the end, to match the cluster size.

5.3.1 Adaptive Sampling Component Testing

The scenario shown in Figure 5.5 is used to test adaptive sampling. The scenario has two paths for a mobile node to follow. The first path has a heading of 0° and the second path has a heading of 270°.

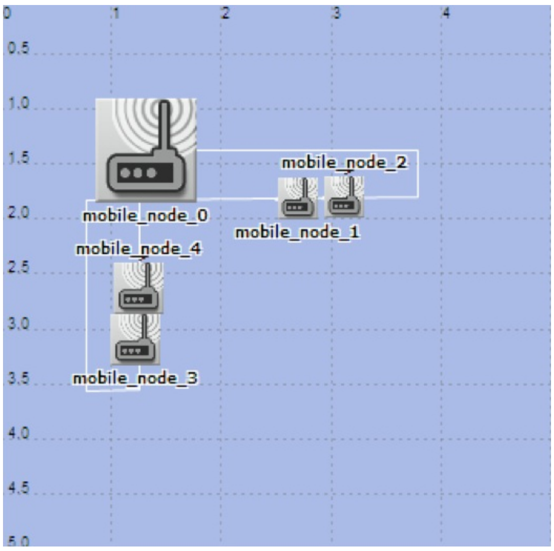


Figure 5.5: OPNET scenario for neural network based hot spot prediction

Figure 5.6 shows an accurate prediction of the hot spots in the OPNET scenario 5.5. On the first run, the mobile node collects the trajectory data from the origin to the hot spots. This is used to create clusters, using the neural network algorithm as shown in Figure 5.6 (a).

The prediction of the hot spot was done by providing the neural network algorithm with sub-trajectory (enroute) data. The algorithm was able to match the incomplete trajectory with the corresponding hot spot as shown in Figure 5.6(b)

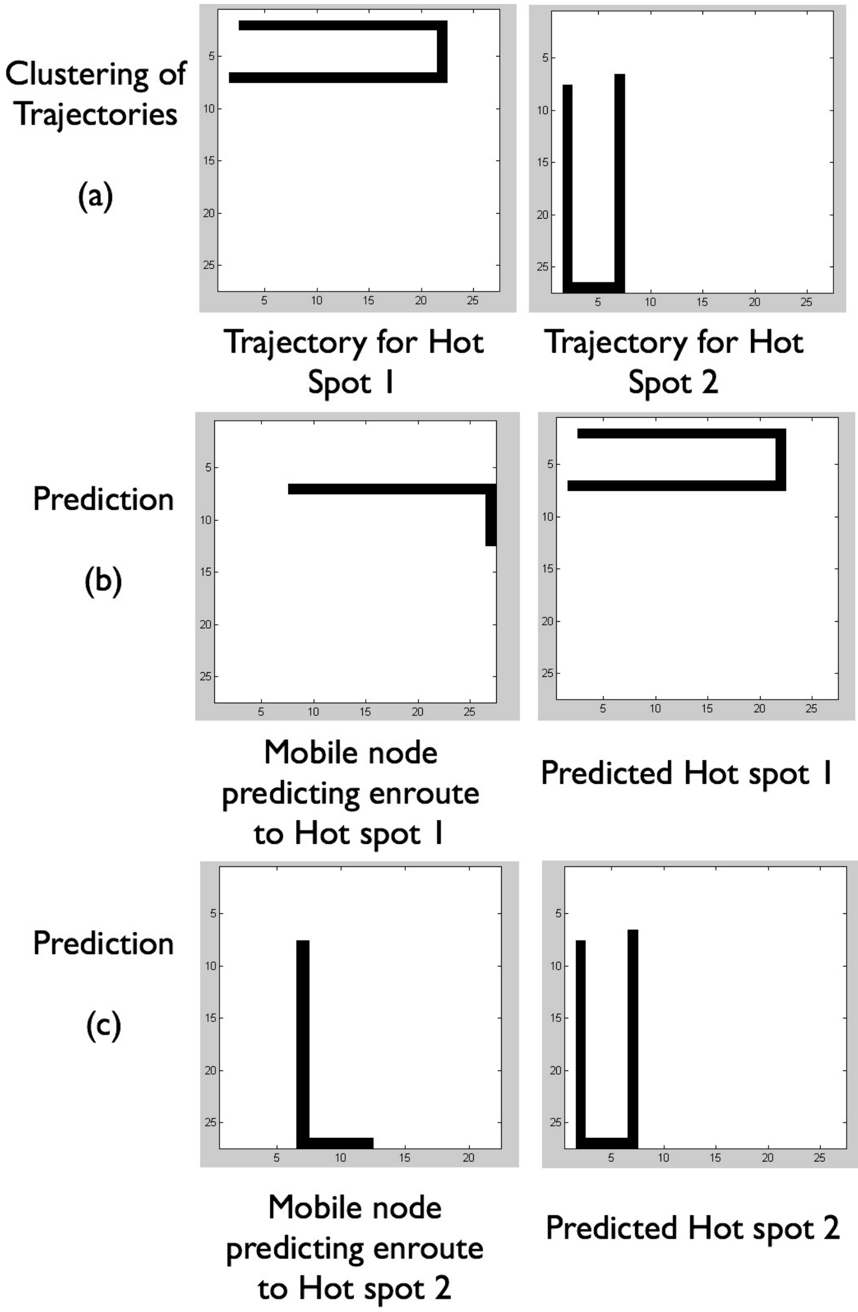


Figure 5.6: Hot spot clustering prediction

and (c). Once the hot spot is predicted, the sampling needs to be performed as the mobile node approaches the hot spot. The number of samples that are transmitted depends on how accurately the mobile node is able to estimate its

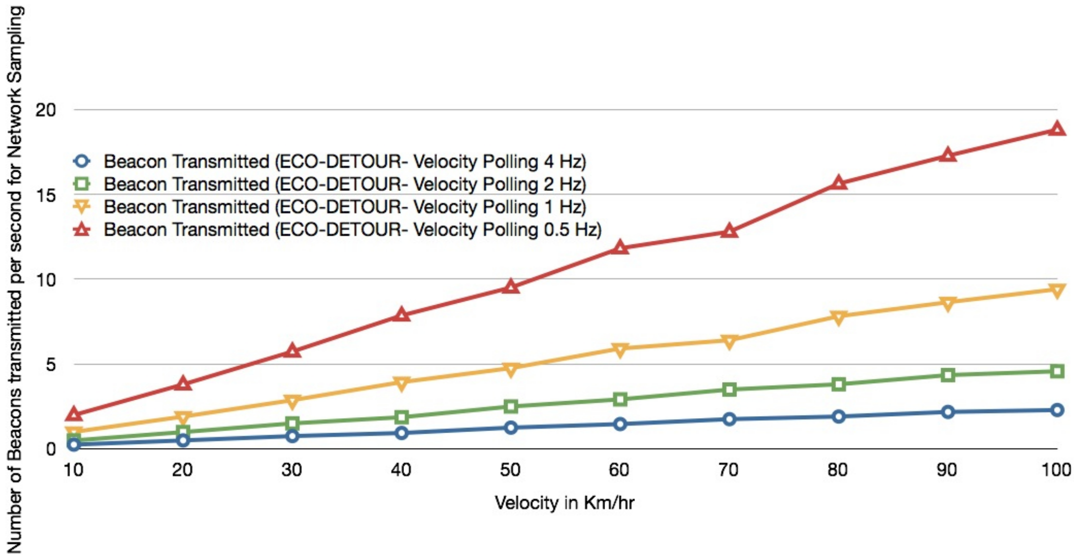


Figure 5.7: Relationship between Beacon transmission and Velocity Polling

distance to the hot spot. This is dependent on the velocity of the mobile node. At higher velocities, estimation becomes poorer, according to equation 4.9, which leads to more beacon transmissions. It is evident from Figure 5.7 that more beacons are required at higher velocities, but if the node polls for the velocity faster, the estimation is improved and the number of beacon transmissions reduces. In Figure 5.7, as the polling f_{vs} of the mobile node velocity is increased, the number of beacons transmitted for sampling reduces.

5.4 Coverage Radius Modelling

The adaptive sampling processor triggers the coverage processor into successful node discovery. It collects the 'Data size' from the 'Data_Generator' processor. In OPNET the op_ima_attr_get(processor_id, attribute_name) is used to read values of the node attributes ('D_p', 'v' and 'B_r') and op_ima_attr_set (processor_id, attribute_name) is used to write values to attributes. The 'Coverage' processor computes the transmit power as given by the algorithm 5.4. The computed power is set as the transmission power for the radio through the lower MAC layer shown in Figure 5.1.

Data: $v \leftarrow$ sensor_read(velocity)
Data: $D_p \leftarrow$ sizeof(data)
Data: $B_r \leftarrow$ set(Baud_rate)
$t_w \leftarrow D_p/B_r$;
$Cr \leftarrow (v * t_w)/2$;
$P_t \leftarrow Cr^2 * (4\pi)^2 * (S_m/\lambda^2)$;

Algorithm 5.4: Pseudo-code for Coverage radius

5.4.1 Coverage Radius Component Testing

The transmission power requirement increases with increasing velocity, to maintain a constant window of time, given in Figure 5.8.

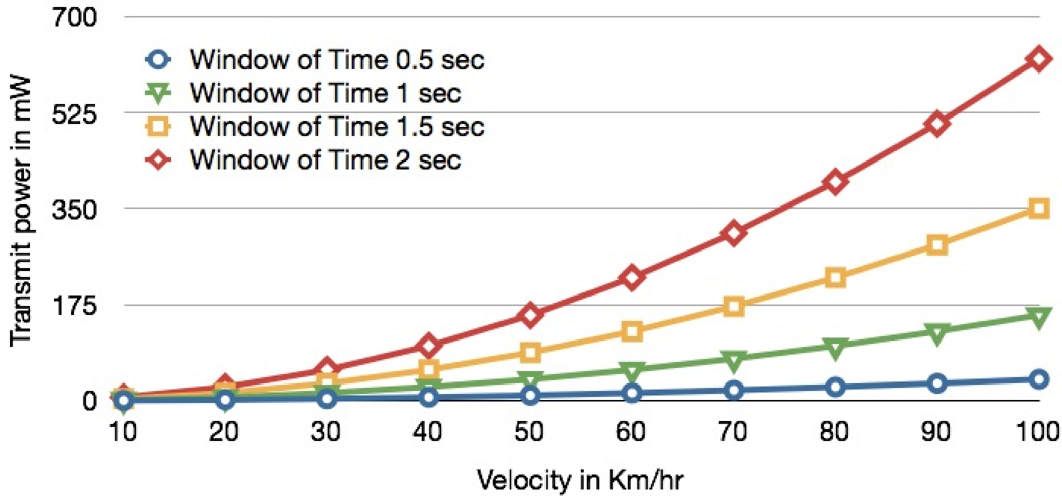


Figure 5.8: Power Modulation to maintain a constant window of time for connectivity

The simulation setup is shown in Figure 5.9. The setup consists of two nodes, one mobile node and the other stationary. The mobile nodes' trajectory is shown as a white arrow, in the figure. The velocity of the mobile node is varied from 10Km/hr to 100Km/hr in increments of 10Km/hr for successive executions of the simulation. The testing is conducted with a receiver sensitivity of -85dBm and a free space loss of -67.33dBm (for the 25m range). The distance of 25m can be increased or compensation for path losses can be incorporated by biasing equation 4.16 with a constant factor for power.

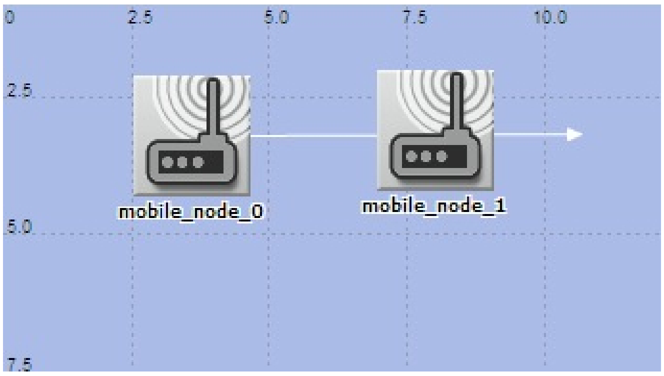


Figure 5.9: Simulation setup in OPNET to test Coverage

A comparison of the percentage of transmitted packets to received packets at different velocities is given in Figure 5.10. The power is modulated to transmit 2048 bits. It is evident that the window of time shrinks with increasing velocity. The ECO-DETOUR protocol varies the power to successfully transmit the complete data. The computed power for 2048 bits transmitted at different velocities varies from 0.067mW to 6.76mW. The coverage's modulated transmission power is compared with the minimum and the maximum transmission power in Figure 5.10. The number of data bits received ranges between 5% to 10% for 0.067mW at velocities greater than 20Km/hr. At the maximum transmission power of 6.76mW, 100% of data is received. With adaptive transmission power 100% of packets are received but the transmission power using adaptive transmission power is lower in comparison with the maximum transmit power of 6.76mW.

The lower transmission power level of adaptive transmission helps in conserving the battery power of the node. Figure 5.11 shows the percentage of battery power conserved through coverage at different velocities. The power saving decreases with increasing velocity and ranges from 6% at 90Km/hr to 97% at 10 Km/hr.

The requirement for power increases with increasing data size and velocity, as shown in Figure 5.12.

Maximum transmission power is bounded by a given threshold, as specified by the transmitter capacity or by the maximum size of the data to be transferred. A threshold power of 6.76 mW is considered here to be the maximum power needed

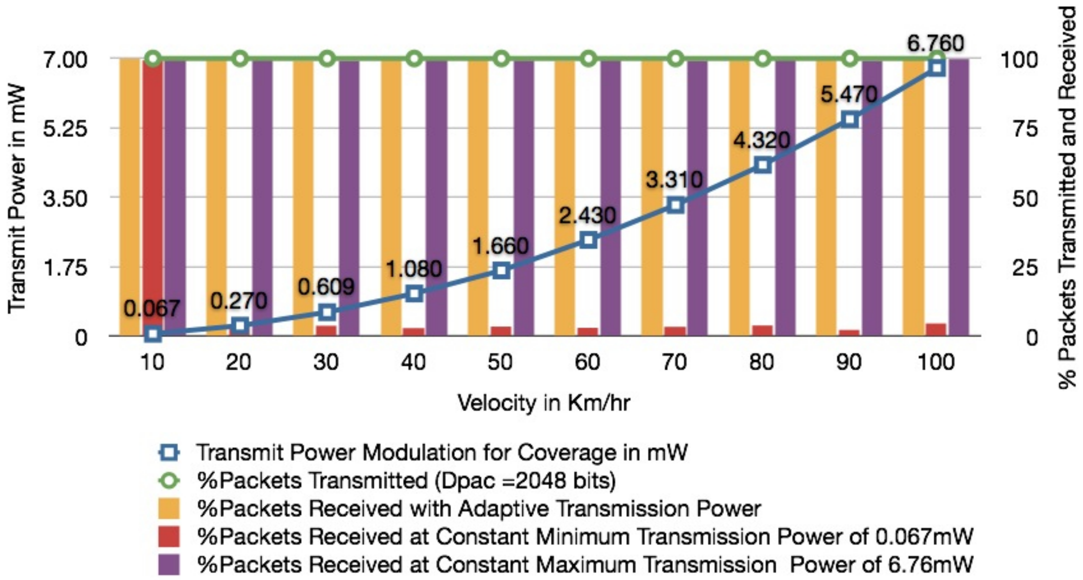


Figure 5.10: % Packet Received at different velocities with Adaptive Coverage

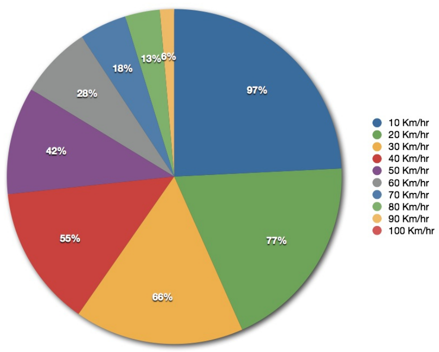


Figure 5.11: % Power saved through Coverage power modulation

to successfully transmit 2048 bits at 100Km/hr.

On doubling the data size from 2048 bits to 4096 bits, the power requirement increases, as given in Figure 5.12. Figure 5.13 shows that with adaptive transmission power, complete data is successfully transmitted but it is achieved an increased power level. To ensure an energy–efficient transmission, a maximum threshold transmission power is set at 6.76 mW. With the threshold power of 6.76 mW data packets are successfully received at velocities lower than 50 Km/hr, but packets start dropping at greater velocities as shown in Figure 5.13.

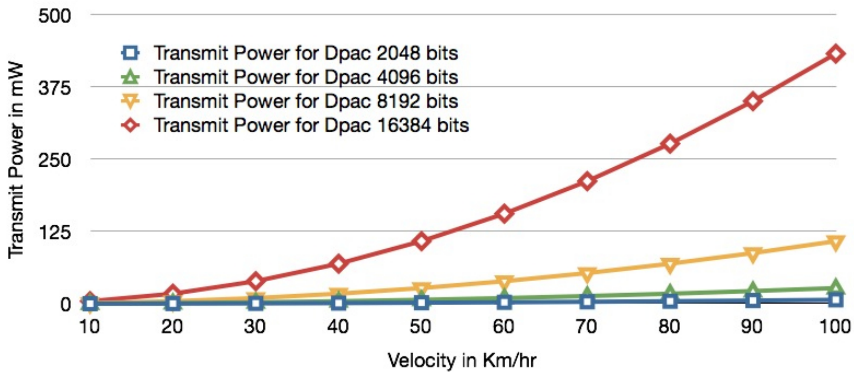


Figure 5.12: % Data packet size versus transmit power

For velocities lower than 50 Km/hr packets are successfully received because the transmission power of 6.76 mW is greater than the required transmission power for transmitting 4096 bits.

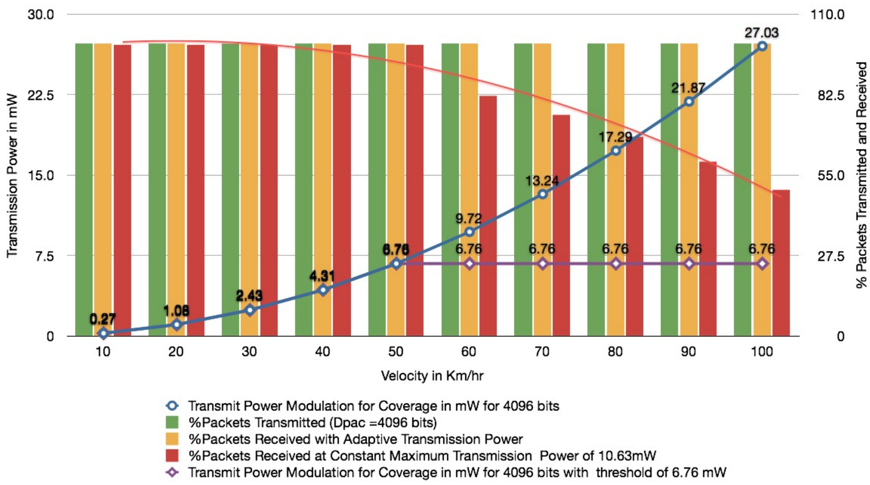


Figure 5.13: % Impact of doubling data packet size

As the power cannot be increased beyond a certain threshold, and there are packet drops due to transmission at lower power, data handoff is necessitated.

5.5 Handoff Modelling

Handoff operates in two modes for mobile nodes and stationary nodes, as given in algorithm 5.5:

```
if v > 0 then
    Transmit (Handoff Request);
    if Packet == Handoff_acknowledgement ∧ H_na > H_n then
        Transmit (Beginning Transmission);
        Trigger (Data_Generator);
    end
else
    if Packet_receive==TRUE then
        if Packet==Handoff_Request then
            Transmit (Handoff_acknowledgement);
        else if Packet==Pilot_Signal ∨ Packet==Beginning Transmission
        then
            Trigger (Handoff Processor);
            Trigger (Direction Processor) goto handoff1;
            handoff1:
            if Decrementing_Count>0 then
                Transmit (Pilot_Signal);
                Decrement (Decrementing_Count);
            end

        else if Packet==Neighbour_discovery then
            Trigger (Data_Generator);
        end

    else
        Transmit Neighbour_discovery;
        Trigger (Data_Generator Processor);
    end
end
```

Algorithm 5.5: Pseudo-code for Handoff

The mobile nodes request for handoff and the stationary nodes exchange neighbour information and propagate the pilot signal. When handoff is requested by a mobile node, the stationary node transmits a 'Pilot signal' ahead of the data to make the nodes aware of the incoming mobile node.

5.5.1 Handoff Component Testing

The handoff is tested with the simulation scenario shown in Figure 5.14. The test setup consists of five nodes; one mobile node and four stationary nodes. The trajectory for the mobile node is shown as a white arrow in the figure. The velocity of the mobile node varies from 10Km/hr to 100Km/hr, in increments of 10Km/hr, for successive executions of the simulation. The test consisted of four scenarios. The threshold power is set as 6.76mW. The four scenarios consist of transmission of 4096 bits and 8192 bits through coverage and handoff respectively. Through coverage, packet drop starts at velocities greater than 50 Km/hr for 4096 bits. For 8192 bits, the packet drop begins at a velocity as low as 20 Km/hr. As coverage of a single node is not able to support the communication, handoff is activated. Transmission of 4096 bits requires 2 nodes in order to successfully receive data at velocities greater than 50Km/hr and 8192 bits require 4 nodes to successfully receive data at velocities greater than 20 Km/hr.

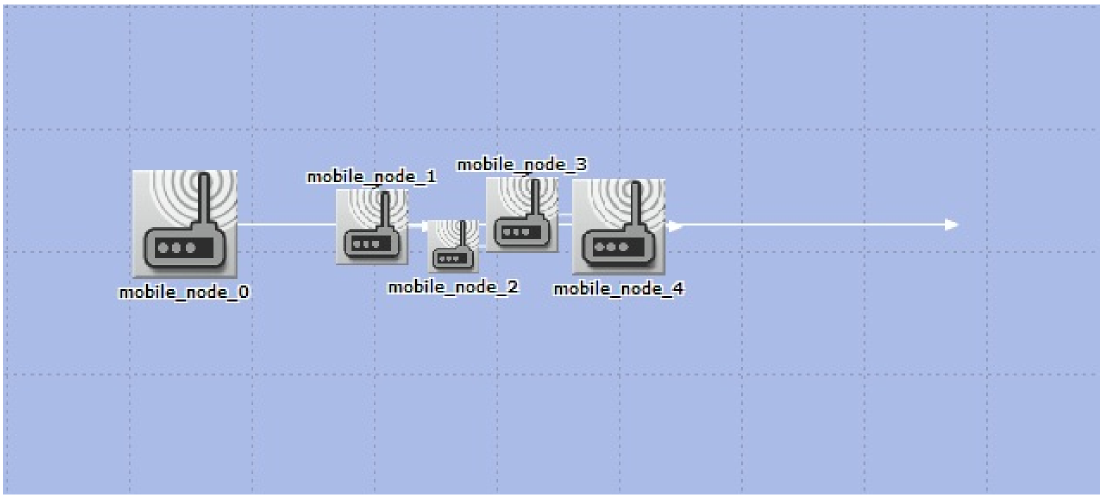


Figure 5.14: Simulation setup in OPNET to test Handoff

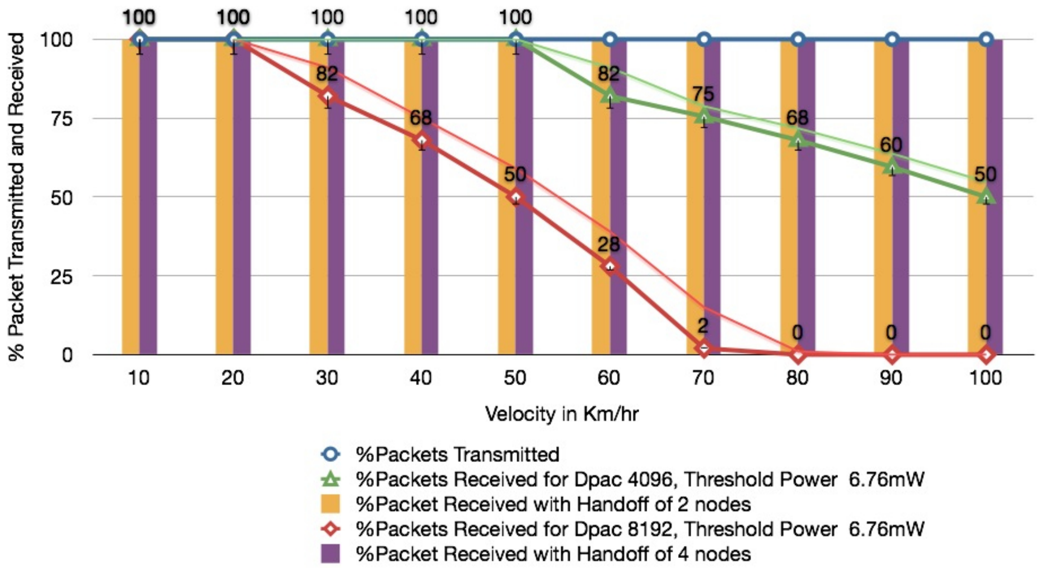


Figure 5.15: % Packets received through handoff

5.6 Direction

The direction of the mobile node serves two purposes: (a) provide heading information to the adaptive sampling processor to map trajectories and (b) help stationary nodes know their relative position.

In OPNET, the command op_ima_attr_get (processor_id, yaw) is used to get the heading of a mobile node. The yaw of a mobile node can be manually specified or computed automatically by the simulator. While specifying the trajectory of the node the default value of the yaw (heading), is 0.0. This is changed to '*autocomputed*' for OPNET to set the value of yaw based on changes in the trajectory. The adaptive sampling processor and handoff processor are connected bi-directionally to the direction processor and poll it to collect the heading data as given in the following pseudo-code:

```
if(Trigger from Adaptive Sampling Processor OR Handoff Processor)
return(yaw)
```

5.6.1 Direction Component Testing

The protocol is designed to incorporate the effect of direction on Handoff. Nodes which do not fall in the trajectory of the mobile node turn off on receiving the Pilot signal, which contains the direction information of the incoming mobile node. Figure 5.16 shows the direction scenario. Nodes 1,2 and 3 are stationary

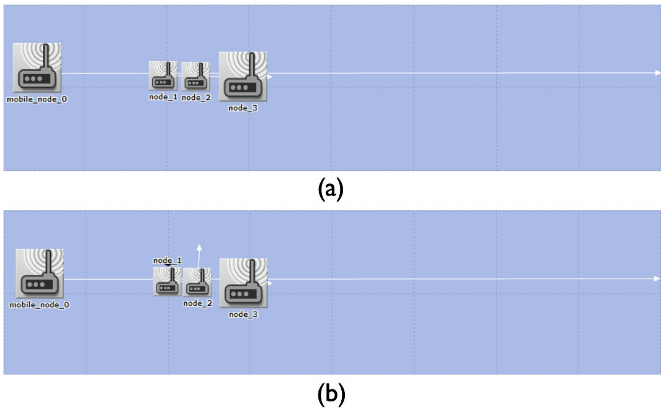


Figure 5.16: Scenario in OPNET for Handoff with Direction: (a) All stationary nodes' heading same as mobile node heading. (b) Stationary node 2 headed perpendicular to mobile node.

nodes. In Figure 5.16 (a) all the stationary nodes point in the same direction as the direction of motion of the mobile node. In Figure 5.16 (b), nodes 1 and 3 are in the direction of motion of the mobile node, but node 2 is oriented perpendicular to the trajectory of the mobile node.

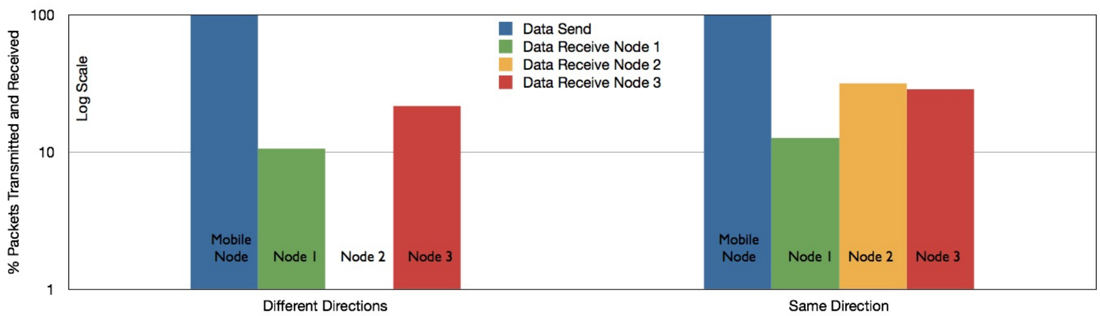


Figure 5.17: Node 2 for Different Direction does not participate in Handoff

As node 2 is oriented at an angle different from the heading of the mobile node, there is a possibility that it may not be a suitable carrier of the data and so the protocol turns it off. This is clarified by the results in Figure 5.17. Without the implementation of direction, node 2 would receive $\sim$30% of the packets transmitted by the mobile node. As node 2 would move in a direction different from node 1 and node 3 and also the mobile node, whatever data it collects cannot be collated by node 1 and 3 to reconstruct the complete data transferred by the mobile node, resulting in a wasted transmission.

5.7 Conclusion

This chapter describes the implementation of the ECO-DETOUR protocol. The four functional parameters are modelled as processors within OPNET. The conditions of triggering of the different processors and their functioning is entailed. The triggers for adaptive sampling component are generated at OPNET environment. Involved processing is done by MATLAB environment. The four components of the ECO-DETOUR protocol are tested individually for their functioning. The test scenario for adaptive sampling consists of two hot spots. Once the hot spots are clustered the adaptive sampling component predicts the hot spots correctly when presented with partial trajectory. ECO-DETOUR shows that energy efficient coverage radius can be achieved through modulating the transmission power. Packet drop occurs if the size of the data packet to be transmitted is increased without modulating the transmission power. This issue is overcome by activating the handoff as soon as the required transmission power exceeds the threshold transmission power. The direction component contributes to saving by turning the nodes off and avoiding node overhearing.

Chapter 6

ECO-DETOUR Performance Evaluation and Application

6.1 Introduction

ECO-DETOUR is compared with other protocols to understand the overall gain achieved through the implementation of this new protocol.

This thesis compares the ECO-DETOUR protocol with the Epidemic, PRoPHET and CAR routing protocols. These protocols have been chosen for comparison because they identify regions or patterns of encounter and assign probability values to determine the next carrier of data to the destination. ECO-DETOUR is similar to these protocols as it identifies and tags regions where it can exchange data. ECO-DETOUR differs from the others in the approach to identification of hot spots. It tags the hot spot regions and predicts them later, whereas the other protocols assign probability values to the encounters. The other protocols are required to transmit control to establish encounter with other nodes, whereas ECO-DETOUR starts the communication once it reaches the hot spot region. This way it does not need to search for nodes in regions where node density is sparse. The cost metrics which we analysed involve the number of beacon signals transmitted for node discovery, packet drops and overall energy expended in transferring data from source to destination. Most of the work reviewed in the literature considers 'delay' in the data reaching destination from source and 'storage memory' as cost metrics. We do not consider delay as we consider hot spots as destinations and once the prediction algorithm clusters a hot spot it is able to accurately predict them (by monitoring the inputs velocity, heading and time). This enables the node to know the delay to the destination. Other protocols need to consider delay as they need to select suitable carriers for forwarding data to the destination. Due to the advancements in high capacity flash memory, storage memory is considered unlimited for the size of data needed to be transferred by a mobile node.

6.2 Protocol comparison

The scenario considered for comparison involves a mobile node moving in a sparsely populated environment towards a hot spot having three relatively stationary nodes, as shown in the Figure 6.1. Amongst the three stationary nodes

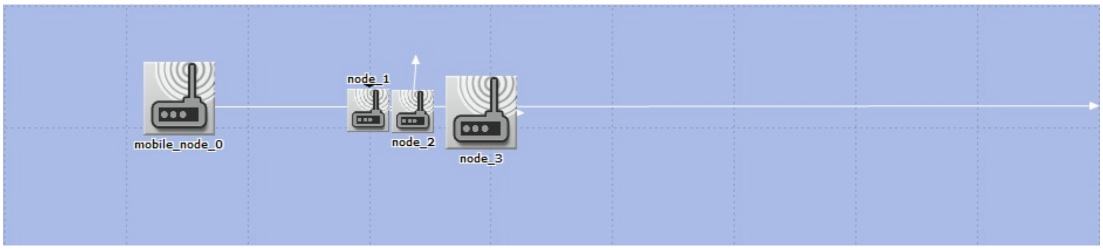


Figure 6.1: Scenario in OPNET for protocol comparison

the heading of 'node 2' is oriented perpendicular to the direction of motion of the mobile node. This scenario allows comparison of ECO-DETOUR with other protocols from the perspective of sampling (to search for a hot spot), coverage (for a velocity lower than 50Km/hr), handoff (a velocity greater than 50Km/hr) and direction (putting a node to sleep to avoid overhearing and energy loss). The velocity of the nodes varies from 0 to 110 Km/hr and its effect on node discovery, packet drops and overall energy expenditure is considered.

6.2.1 Transmission Overheads for Node Discovery

All transmissions except data are considered as overheads on the efficiency of communication. The impact of velocity on η_t, which is the ratio of wasteful transmissions (including beacons and control signals, as they do not carry any information) to data transmission, is given in Figure 6.2. A smaller value of ratio η_t shows greater efficiency of the protocol with PRoPHET and CAR, all nodes communicate with each other to exchange the appropriate probability factor and ascertain the nodes' ability to carry the data forward. This means that exchange of beacons and control signals (probability/ utility factors) happens irrespective of the nodes' ability to carry forward data. Epidemic performs better than PRoPHET, as Epidemic makes no discrimination in the probability of a node to carrying forward data. In Epidemic as velocity increases, the number of beacon transmissions reduce. This is because the distance covered per second is larger and so, there is a decrease in the ratio of wasteful transmission to data transmission with increasing velocity as wasteful transmission decreases in comparison to data transmission. With PRoPHET there is a decrease in the number of transmitted beacons, but the decrease in the transmission of data is greater, due to the probability associated with each node's ability to carry forward the data. Therefore the ratio of wasteful transmission to data transmission is greater than in Epidemic routing. Wasteful transmission in CAR samples is less, compared with Epidemic and PRoPHET. This is based on the author's assumption that the mobile node would meet the recipient, based on the past collocation. With increasing velocity the distances covered also increase, which further reduces the sampling. This results in an improvement in the ratio of wasteful transmission to data transmission.

The ratio η_t increases for ECO-DETOUR with increasing velocity due to the increase in beaconing introduced by adaptive sampling. At velocities below 10 Km/hr Handoff amongst nodes gets activated which contributes to overhead control information due to which there is an increase in η_t at low velocities. Adaptive sampling module and the Handoff module of the ECO-DETOUR protocol contribute to wasteful transmission. The adaptive sampling module transmits beacons for node discovery and the Handoff module propagates Pilot signals to

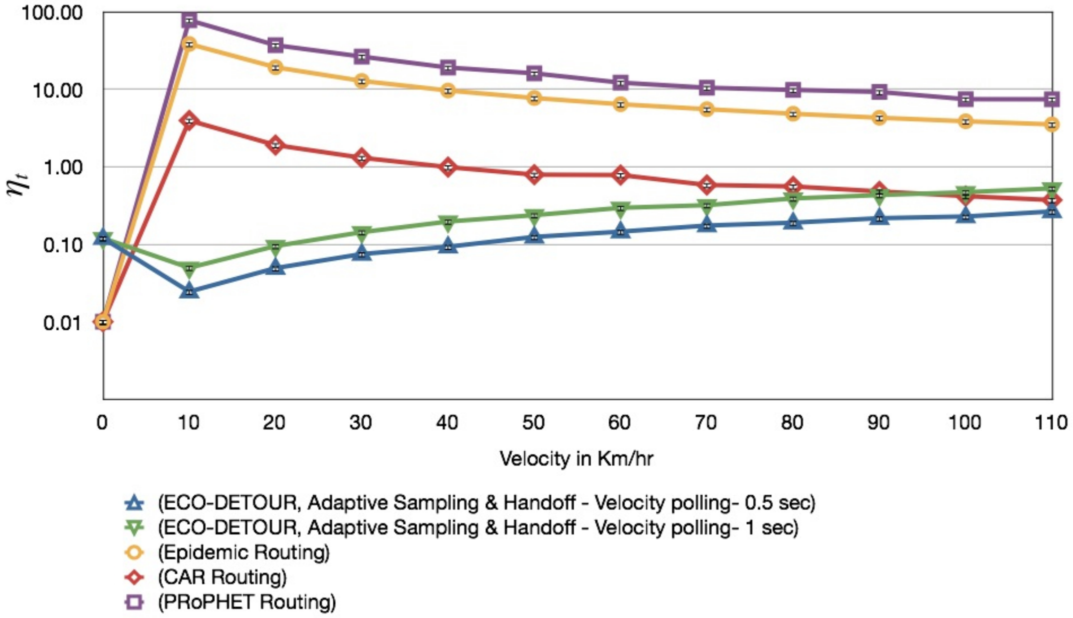


Figure 6.2: Variation in the Ratio of Overhead Transmission to Data Transmission with respect to velocity

set up handoff. The adaptive sampling module predicts the location of the hot spot and starts sampling only when it is close to the hot spot. Sampling should ideally start when the node reaches the hot spot, but as velocity increases, error in determining time to a hot spot also increases. This is related to the rate at which the mobile node polls its velocity. Error is reduced by increasing the polling intervals of velocity.

ECO-DETOUR protocol performs far fewer samples in comparison with Epidemic, PRoPHET and CAR, because it starts sampling after predicting a hot spot. This enables the algorithm to know the distance that it has to cover, and start sampling when close to the hot spot. Epidemic, PRoPHET and CAR work on the principle of providing an opportunity to the encountering node as a possible carrier. This results in an exchange of routing information (probability) on all encounters, so that a number of beacons and control signals that do not meet the probability threshold assigned for the carriers, are transmitted. Handoff for mobile nodes requires the stationary nodes to perform neighbour discovery and transmit pilot signals. The process of neighbour discovery exchanges the decrementing counter value along with the data transmission. Pilot signals are transmitted on request from mobile nodes for handoff. Epidemic, PRoPHET and CAR, at velocity zero, have better performance because they are in the vicinity of each other and require limited sampling of the network to establish communication. ECO-DETOUR, on the other hand, needs to perform neighbour discovery and transmission of pilot signals, and this results in the transmission of overhead signals.

6.2.2 Packet Loss performance

The comparison of packet loss among the Epidemic, PRoPHET, CAR and ECO-DETOUR protocols is based on the scenario shown in Figure 6.1. As the stationary nodes are in communication range of each other and have a relative velocity of zero, packet loss due to node mobility cannot be observed. So we consider the packet losses for a mobile node that uses the hot spot with a relatively stationary node for forwarding data. The velocity of the mobile node varies from 10-110

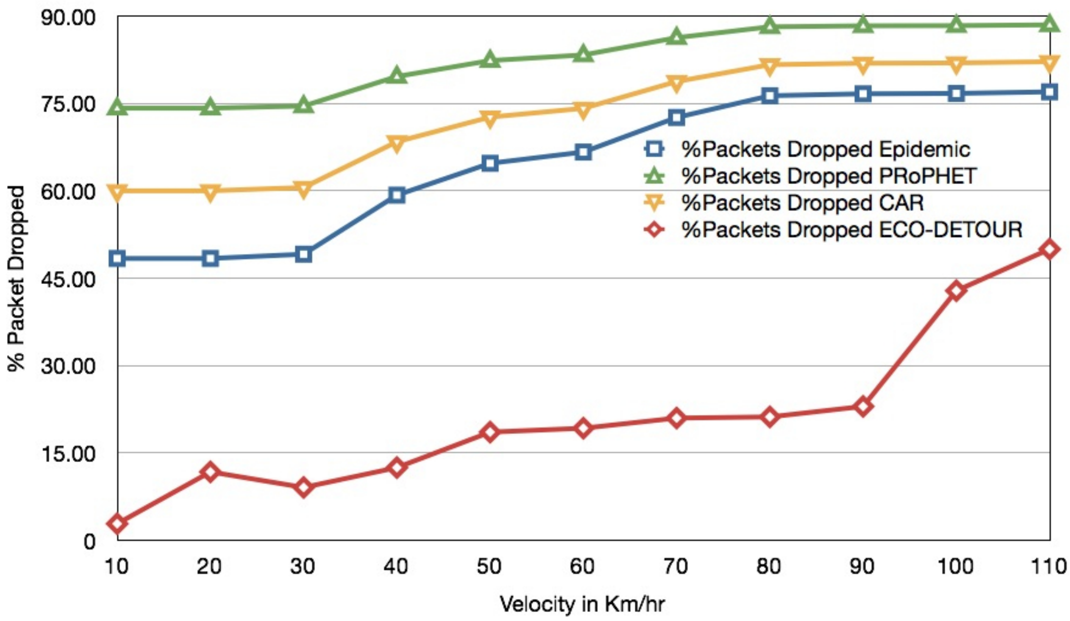


Figure 6.3: % Packets Dropped with increasing velocity

Km/hr and the percentage packet dropped is expressed by ((Packet Transmitted-

Packet Received) / Packet Transmitted)*100. Figure 6.3 gives a comparison of the percentage packet loss amongst these protocols. Epidemic routing performs better compared with PRoPHET and CAR because it forwards packets to every encountering node, thereby increasing the probability of reaching the destination. As it is considered that 'node 2' moves in a direction perpendicular to the direction of the mobile node, any data exchange with it would lead to packet loss.

ECO-DETOUR intelligently turns off node 2, so that there is no data exchange with it and hence there is a reduction in packet loss. CAR and PRoPHET perform poorly due to the uncertainty associated with choosing a carrier.

6.2.3 Energy efficiency performance

Energy efficiency is defined as the ratio of the energy dedicated to transporting data against the energy required to establish communication. Energy for establishing communication involves the transmitted beacons, control signals and the total transmitted data. The scenario for the comparison is shown in Figure 6.1. The energy efficiency of ECO-DETOUR is around 90% at velocities below

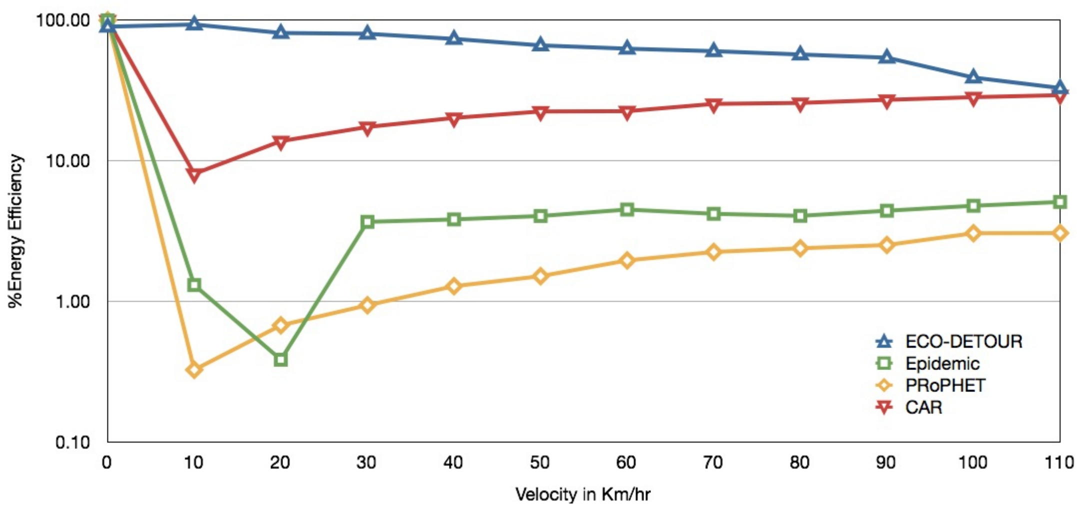


Figure 6.4: % Energy efficiency with increasing velocity

10 Km/hr and it gradually degrades to around 60% at 110 Km/hr. Epidemic, PRoPHET and CAR show very high efficiency at velocities below 10 Km/hr.

This is because the nodes are able to easily select carriers towards a destination when they are fully connected and not mobile. As velocity increases, the efficiency initially degrades and then improves. The initial degradation of efficiency of Epidemic, PRoPHET and CAR occurs because nodes move out of the coverage range of other nodes and require overhead signalling for re-establishing connectivity. The improvement in efficiency is attributed to the increase in node–encounter probability. Epidemic and PRoPHET show a maximum efficiency of 5% -6%. CAR performs well and has a maximum efficiency of 50%. The maximum efficiency of ECO-DETOUR is 90%. The 10% loss occurs, because, at low velocities the stationary nodes exchange neighbour information and pilot signals, resulting in a waste of energy. ECO-DETOUR degrades in performance due to errors in estimating the distance to a hot spot at higher velocities. At lower velocities the error is less, hence the number of beacons transmitted is less. At higher velocities the error increases, resulting in an increase in the number of transmitted beacons.

6.3 Case Study on Wildlife Monitoring

Animal monitoring is one of the principal applications of wireless sensor networks. Lion monitoring in South Africa has gained importance in recent years, to help re-introduce the animal to the wild. A map of the area with the trajectory of movement is given in Figure 6.5 (a) [110]. The trajectory of the animal was plotted by Lehmann et al. [111]. The trajectory length is a total of 24.4Km. The animal prefers to be near water and hence has a trajectory alongside water. The normal velocity of the animal is 0.45±0.07 Km/hr and during a chase it increases to 80Km/hr. The adaptive sampling component learns the trajectory and predicts for future sampling of the regions. The simulation scenario in OPNET is shown in Figure 6.5 (b). The plotting of trajectory in [111] is missing in certain places. Therefore, for simulation in OPNET, the path is completed from hot spot H(a) to the starting point (origin). The next trajectory is started from the origin to hot spot H(b) and back to origin.

Figure 6.5 (c) shows the clustering of the two hot spots H(a) and H(b), and

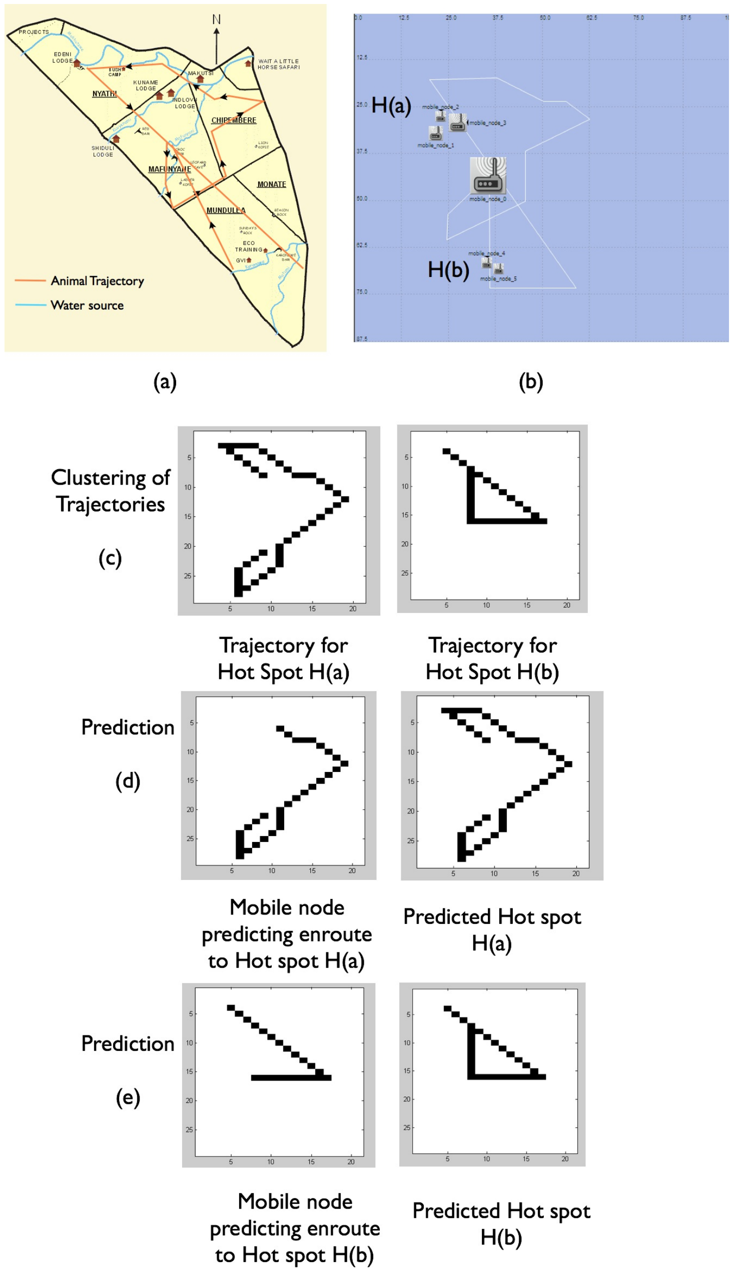


Figure 6.5: Map of Karongwe game reserve: (a) Map of Karongwe reserve. (b) Karongwe trajectory map on OPNET. (c) NN clustering. (d) NN prediction of trajectory for hot spot H(a). (e) NN prediction of trajectory for hot spot H(b).

hot spot H(a) is predicted using a sub–trajectory in Figure 6.5 (d) and hot spot H(b) is predicted using a sub–trajectory in Figure 6.5 (e). Considering hot spot

H(a) is predicted, the distance of H(a) from origin is 24.4Km. Figure 6.6 shows the number of beacons transmitted at different velocities. It is observed that at normal velocity of 0.45Km/hr (0.125m/s), the number of beacons transmitted is lower than when the animal is chasing a prey at velocity 80Km/hr (22.2m/s).

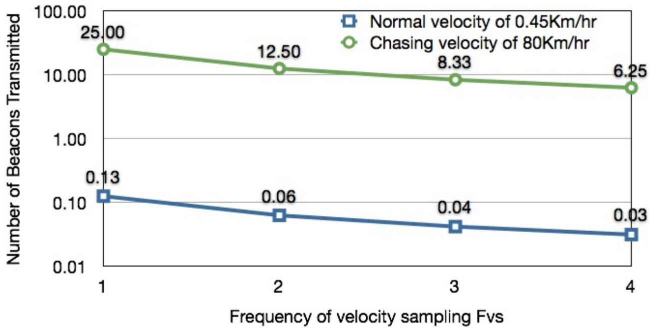


Figure 6.6: Number of Beacon transmissions

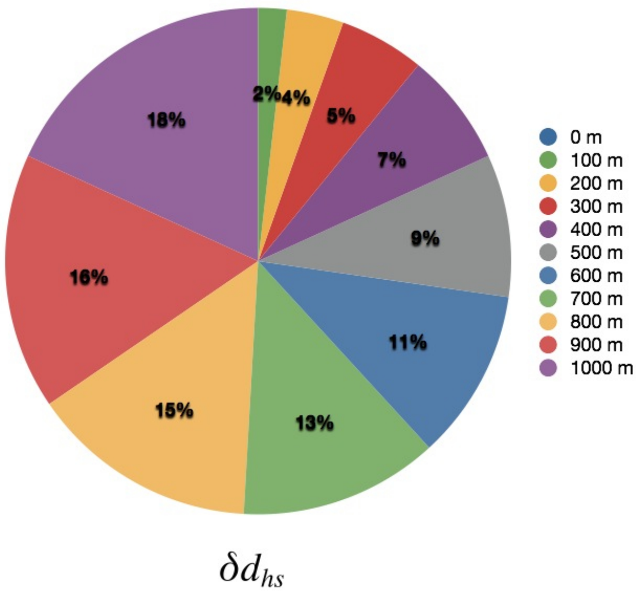


Figure 6.7: % Increase in beaconing due to deviations from trained trajectory

The prediction of hot spots is tolerant to small variations in the trajectory of the node, as shown in Figure 6.8. The node is able to predict the hot spot correctly but variation in the path 'δd_{hs}', affects the beaconing. The tolerance

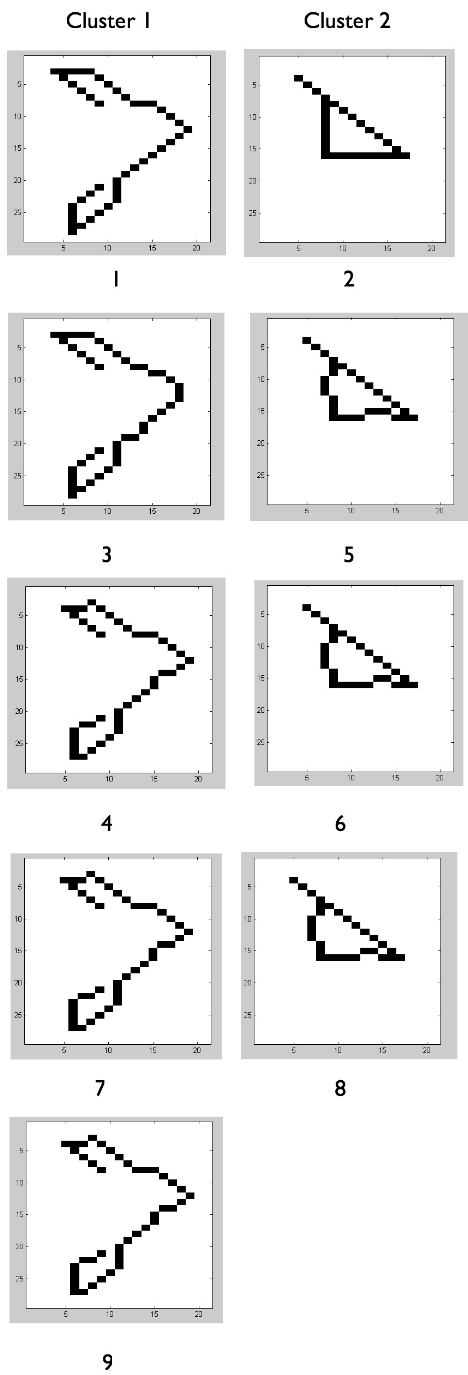


Figure 6.8: Prediction tolerant to variations in input trajectory

δd_{hs} is varied from 0 - 1 Km for this application, and this impacts the beaconing. Figure 6.7 shows the percentage increase in beaconing due to the deviations in the trajectory. The beaconing varies from 2% at 0.1Km deviation to 18% at 1Km deviation from the trajectory.

On reaching the hot spot, if the animal is not chasing it can be considered stationary due to the low mobility velocity. Therefore the scenario of chasing a prey is considered as shown in Figure 6.9. The inter-distance between animals is

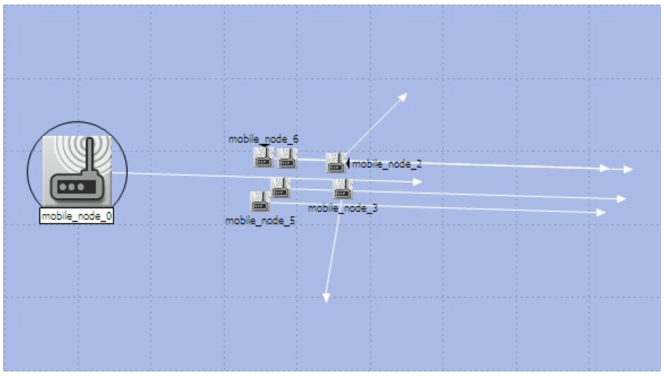


Figure 6.9: Scenario for wildlife

considered to be between 2-5m (0.002-0.005 Km) in the region of habitat. The velocity of lion during chase is considered as 80Km/hr. It is assumed that the lion while travelling between origin and the hot spot accumulates data. The data packet size is assumed to vary between 1-10 Kilobits. Table 6.1 shows the window of time available at a velocity of 80Km/hr for the coverage range of 2-5m.

Table 6.1: Window of time for Coverage Range of 2m and 5m

Coverage Range	Free space losses	Receiver Sensitivity	Window of Time at 80Kmph
2m	46.1dBm	-85dBm	0.09 secs
5m	54 dBm	-85dBm	0.225 secs

The window of time needed to transmit data packets of size 1-10 Kilobits ranges between 0.104 *sec* for 1 Kilobyte to 1.01 *sec* for 10 Kilobyte at a baud rate of 9.6Kbps. Table 6.2 gives the required window of time. It is evident from the table that for packet size around 2048 bits, the coverage of a single node would be insufficient, because even at a maximum coverage radius of 5m, the required window of time is greater than the available window of time 0.225 *sec*. Therefore

Table 6.2: Required window of time

Data Packet size in bits	Required Window of time in sec	Computed Coverage Range	Computed Transmission Power in dBm
1024	0.10416667	2.31481481	0.3449426
2048	0.20589193	4.62962963	6.36554251
3072	0.30761719	6.94444444	9.88736769
4056	0.40934245	9.25925926	12.3861424
5120	0.51106771	11.5740741	14.3243427
6204	0.61279297	13.8888889	15.9079676
7168	0.71451823	16.2037037	17.2469034
8192	0.81624349	18.5185185	18.4067423
9126	0.91796875	20.8333333	19.4297928
10240	1.01969401	23.1481481	20.3449426

coverage is active for data packet sizes up to 2048 bits and for larger packet size handoff is active. The modulation of power for coverage with respect to the data packet size is shown in Figure 6.10.

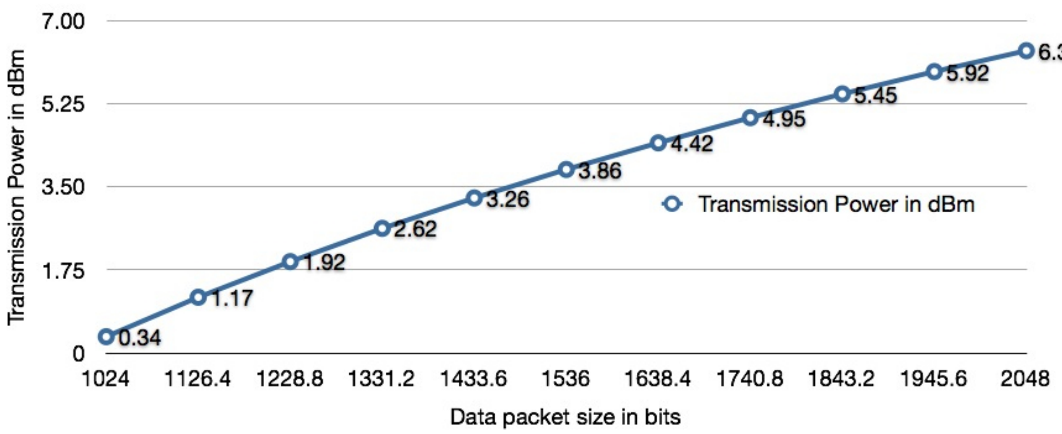


Figure 6.10: Coverage power adapted to increasing data packet size

Figure 6.11 shows packet drops at a constant transmission power of 0.32dBm, whereas with power modulation as shown in Figure 6.10 the transmitted packets are successfully received.

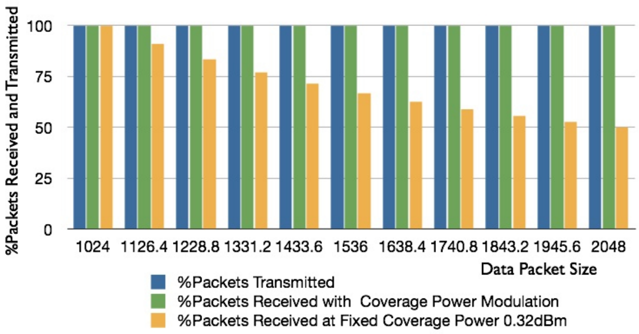


Figure 6.11: Coverage power adapted to increasing data packet size

The data packets greater than 2048 bits require handoff because as shown in Table 6.2 the required coverage range for successful transmission increases beyond the maximum coverage range of the node which is 5m. 2048 bits of data can be transmitted in a coverage range of 5m, therefore, to transmit every chunk of 2048 bits, an extra node would be required. To transmit 8192 bits, 4 stationary nodes would be required for handoff. As shown in Figure 6.9, 4 nodes move in the same direction as that of the mobile node 0. These nodes provide the handoff. As nodes 2 and 3 scatter in different directions they turn off and do not participate in handoff.

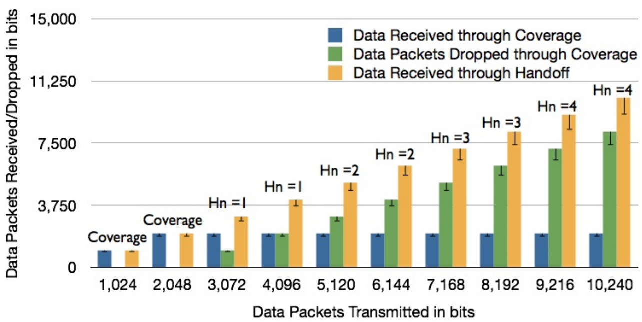


Figure 6.12: Coverage power adapted to increasing data packet size

A comparison of handoff and coverage is given in Figure 6.12. It is evident that coverage of a single node is unable to handle packets greater than 2048 bits and data packets that falls outside the available window of communication time start dropping. This is resolved using handoff among multiple nodes.

6.4 Conclusion

ECO-DETOUR tries to achieve reduction in energy through reduction in network sampling and optimization of transmission power with respect to the velocity and data packet size. The protocol turns off nodes which have a different heading to that of the mobile node to avoid energy consumption due to overhearing. ECO-DETOUR achieves a 45% - 60% reduction in energy consumption for setting up communication and exchanging data packets. It achieves efficiency in the range of 60% to 90% for the velocity range of 110 to 10 Km/hr respectively. Most of the protocols dealt with here perform well in the 0-10 Km/hr velocity range but their efficiency degrades to 5-6% at higher velocities. CAR comes closest to our work achieving an energy efficiency of ∼10% to ∼50% at velocities of 10 Km/hr to 110 Km/hr respectively.

Chapter 7

Conclusion and Future work

7.1 Introduction

ECO-DETOUR Protocol is designed to optimise the transmission of beacons for node discovery, reduce packet drops and minimise energy loss due to node overhearing. The beacon transmission, packet drops and node overhearing are overheads to communication. The energy efficient wireless sensor network protocols present in the literature assume node redundancy to increase the node and network life time. Mobile ad-hoc protocols concentrate on minimising the disruption due to mobility of nodes. These protocols beacon at regular intervals for node discovery. This results in wastage of energy. Opportunistic connectivity protocols concentrate on the study of mobility patterns and issues related to data storage and forwarding.

This thesis is an effort to bring together energy efficiency, node mobility and delay tolerant opportunistic connectivity for data exchange. The communication depends on the demographic distribution of the nodes. The core ideology is to learn and predict the node distribution and increase the window of time for communication.

ECO-DETOUR protocol monitors node mobility parameters (velocity and heading with respect to elapsed time), to discover nodes and achieve the required window of time for connectivity. The protocol design involves, adaptive sampling, coverage, handoff and direction as the four key functional components.

Adaptive sampling supports minimising the efforts needed by the mobile node for the discovery of hot spots. These are regions with high density of nodes. Artificial neural network is used to perform learning and prediction of the hot spots. It monitors the mobile nodes' trajectory and creates the trajectory maps. This scheme of prediction mitigates the requirement of transmitting the beacons at regular intervals, thus reducing the energy consumption of the node. The performance of adaptive sampling depends on the rate of monitoring the velocity of the mobile node. Higher rate of monitoring of velocity leads to lesser number of beacon transmission. Coverage and Handoff address the issue of maximising the window of time for connectivity. Coverage uses one stationary node to collect the data from the mobile node. If the coverage of a single node is insufficient, then handoff, which uses multiple stationary nodes to increase the window of time for communication, is activated. Coverage and Handoff reduce packet drop through power modulation and node coordination and this reduces the wastage in energy. Energy loss of the node occurs through overhearing of communication. The Direction component of the protocol conserves energy by reducing node overhearing. This is achieved by putting nodes to sleep if they do not lie in the path of the mobile node by monitoring the direction of motion of the mobile node.

7.2 Conclusion

The overall contribution of the thesis is the design and implementation of an opportunistic connectivity protocol which reduces packet drops and transmission overheads resulting in improved energy efficiency.

Opportunistic connectivity involve node mobility which impacts their behaviour with other network nodes. Therefore the thesis necessitates a network tool which support mobility. Amongst the many network simulation tools available OPNET has been chosen as it provides a complete graphical user interface and a hierarchical encapsulated design methodology which is helpful in designing and debugging nodes. The graphical user interface allows drawing of trajectory which helps in visualising the trajectory path and the placement of nodes. Matlab offers an extensive library of functions which allow efficient manipulations of matrices used in the implementation of the artificial neural network. ART1 neural network algorithm has an unsupervised learning capability which is useful for the protocol implementation, as opportunistic connectivity environment undergoes changes in real time and the ART1 algorithm can adapt to such changes. The two simulation environments are integrated which allows use of Matlab functions to be used in OPNET.

The protocol design combines the four functional components to extend the window of time for communication in an energy efficient manner. The ECO-DETOUR protocol achieves energy efficiency by reducing the beaconing through adaptive sampling. The adaptive sampling component commences beaconing only when the node approaches the hot spot. This is unlike regular sampling where nodes require to transmit beacons at regular intervals for node discovery. The neural network predicts the hot spots and the node computes the time it takes to reach the hot spot. As velocity is a variable and there may be deviations in the current trajectory from the trained trajectory, the computation of time to hot spot would vary on every execution of the algorithm. This is overcome by increasing the sampling frequency of the velocity and compensating the distance to hot spot by adding the deviations in the trajectory.

A further reduction in expended energy is achieved through optimisation of transmission power and the use of multiple stationary nodes. The coverage component modulates the transmission power based on the node velocity and the transmission data packet size. The power requirement increases with increasing velocity and data packet size. As the power cannot be increased beyond a threshold value, handoff, which uses multiple nodes to receive data is used. Handoff allows a group of stationary nodes to increases the window of time and collect all the data and later collate. The mobile node sends a handoff request if it determines that the coverage of a single node is not sufficient for successfully transmitting the data. An acknowledgement to the handoff request containing the number of available stationary nodes is sent to the mobile node. If the mobile node ascertains that the number of available nodes are sufficient for a successful communication data transmission starts. Handoff, through this collaboration of multiple nodes, allows the mobile nodes, to transmit at a low transmission power, which saves energy.

The protocol controls the energy loss due to node overhearing by using the direction component. The direction component allows stationary nodes receiving the pilot signal to ascertain the heading of the mobile node. They compare their heading with that of the mobile node and if it is greater than tolerance value, the stationary nodes turn off and this avoids overhearing.

The performance of adaptive sampling gets affected by the velocity of the mobile node and the frequency of polling of velocity. Higher velocity increases the error in computation of the time to hot spot, resulting in increased beacon transmission. The modulation of transmission power conserves energy in comparison to a constant transmission power. This is because, with constant transmission power, excess transmission power is set to cover the whole range of velocity which is wasteful.

Implementation of ECO-DETOUR on OPNET is done by modelling the functional components discretely as processors. The processors trigger each other to activate functionality of the four components. Simulation scenarios for protocol testing are set in OPNET. A test scenario containing a mobile node moving towards a hot spot is set to analyse the impact of each of the four components on communication overhead. The testing of adaptive sampling component shows that the neural network algorithm correctly predicts the hot spot when presented with a sub-trajectory to the hot spot. With the computation of the sampling time, there is error due to the variation in velocity and deviations in the trajectory path. A reduction in beacon transmission is achieved by increasing the frequency of polling the velocity.

Coverage modulates the transmission power based on the data size and velocity. For lower velocity and smaller data packet sizes the required transmission power is low. In comparison to constant threshold transmission power an energy saving of 97% is achieved at 10 Km/hr. The energy saved decreases to 0% at 100 Km/hr as the transmission power reaches the threshold power. But for all velocities lower than 100 Km/hr there is a significant saving in energy through coverage.

The implementation of handoff uses multiple nodes to successfully establish communication for the mobile node at a lower transmission power. Large data packet sizes and high velocity trigger handoff which does not require an increase in transmission power, thus resulting in an energy efficient data exchange.

The direction component receives the heading of the mobile node through the pilot signal set up during handoff. The scenario to test direction consisted of three nodes with one node oriented perpendicular to the direction of motion of the mobile node. Without the direction component all the three nodes would receive the data. As the node which is oriented perpendicular is headed in a different direction and may not participate in collating the received data, the data may get lost requiring a retransmission. As the direction component turns the node off there is a saving of 33% by avoiding node overhearing.

ECO-DETOUR is compared with Epidemic, PRoPHET and CAR. It is observed that it achieves 45% - 60% reduction in expended energy to set up communication and exchange data packets. Other protocols perform well in the 0-10 Km/hr velocity range but their efficiency degrades to 5-6% at higher velocities, with the exception of CAR whose efficiency is 50% at higher velocities. ECO-DETOUR achieves an efficiency in the range of 90% to 60% at velocities 10 Km/hr - 110 Km/hr respectively. The bulk of the saving in energy of ECO-DETOUR protocol comes from adaptive sampling which is followed by coverage, handoff and direction.

7.3 Future Work and Directions

The concept at this stage reflects good potential for success. Early stage implementation testing required number of assumptions. These assumptions are made to cater for bypassing areas that are still not well defined and need further attention. In this section some of these areas will be discussed as a potential future work.

1. *Deviations from Origin:* It is currently assumed for adaptive sampling that the node always starts from its origin and returns back to the origin. A more generalised scenario is to start at any location and end at any other location. This requires the neural network model to keep track of the deviations from the origin, such that when predicting hot spots the error between origin and

current position could be incorporated and hot spots could be predicted accurately.

2. *Trajectory Definition:* The preprocessing of the trajectory requires mapping the trajectory on a binary matrix. The current work is limited to the following angles '0°, 45°, 90°, 135°, 180°, 225°, 270°, 315°, 360°'. A finer granularity in the angles helps in better prediction of the trajectories. In the current implementation by moving one cell on the trajectory matrix, results in moving in any one of the 8 directions. Finer granularity can be achieved by using more number of cells to specify a single direction. This would require a larger matrix which comes with the overhead of computational complexity and higher memory requirement. Optimisation of this would be our focus in the future.

3. *Radiation Pattern for Coverage:* The current work assumes line of sight transmission with free space path loss in the calculation of transmission power. We would extend this study to other path loss models like ground reflection- two path model and log-normal shadowing.

4. *Handoff for multiple nodes:* The current implementation of handoff is capable of handling one mobile node at a time. The handoff process requires intimation to other nodes through pilot signalling. This blocks the resources for the mobile node. The resources are not available for another mobile node. To handle this situation, as part of future work we propose building and exchanging, a table of mobile nodes to be serviced with the hot spot, amongst the stationary nodes.

5. *Direction based Handoff:* In the current work it is assumed that if the direction of motion of a relatively stationary node is different from the mobile node, then the relatively stationary node should be turned off to avoid overhearing. This assumption can be made application specific with exceptions. The exceptions would allow communication for certain specific directions and put the node to sleep for all the other directions. As an example, in vehicular networks, vehicle moving in the opposite direction should be allowed

to communicate. So for deployments in vehicular networks the exceptions could be set as 0° and 180°.

7.3.1 Applications for the Protocol

The ECO-DETOUR Protocol is designed to take advantage of the node mobility to conserve energy. The protocol is suited for dynamic networks with frequent node mobility. The protocol's ability to adapt its data exchange process by monitoring its mobility makes it suitable for many applications. Applications like '*Vehicular Networks*', '*Patient Monitoring*', '*Wildlife Monitoring*', '*Object Tracking*' may benefit from it. A brief of the protocol features useful for such applications is entailed as follows.

Vehicular networks carry road traffic information, information of road hazards, mishap and so on. Vehicles make use of each other to forward the data. Energy efficient opportunistic connectivity protocol works on reducing the network traffic by reduction in the packet drop and node overhearing [S.Sivakumar and A.A Anbuky][32]. The protocol uses its ability to adapt coverage depending on the road traffic. With high traffic the vehicles slow down and the protocol reduces its transmit power. This reduces packet loss due to interference and node overhearing. The monitoring of the direction of motion of the vehicle is used to determine the vehicle's suitability to carry data. This further reduces unwanted transmissions.

Object tracking can benefits from the protocol. Monitoring check points can be treated as hot spots which would pick up signals from the object being tracked and can create logs.

Un-intrusive remote patient monitoring is an application of WSN which allows patients to continue with their normal lives. These systems remotely monitor the physiological conditions of the body and does not put the constraints on the patient to be in the hospital for monitoring. They transmit the data periodically. By treating the house, work place, library and other public places as hot spots ECO-DETOUR protocol can learn the places the patient visits frequently, helping in locating the patient in case of emergency.

Wildlife Monitoring applications require energy efficient routing protocols because it is tough to replace batteries on the nodes after deployment. ECO-DETOUR protocol provides the nodes with the ability to stay mobile, record animal movement through trajectory mapping and remain energy efficient through adaptive sampling, coverage and handoff.

References

[1] J. Gomez et al., "Conserving Transmission Power in Wireless Ad Hoc Networks," in *Ninth International Conference on Network Protocols*, pp. 24–34, 11-14 Nov. 2001.

[2] C. Intanagonwiwat et al., "Directed diffusion: a scalable and robust communication paradigm for sensor networks," in *Proceedings of the 6th annual international conference on Mobile computing and networking*, MobiCom '00, (New York, NY, USA), pp. 56–67, ACM, 2000.

[3] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, (New York, NY, USA), pp. 22–31, ACM, 2002.

[4] F. Ye et al., "A scalable solution to minimum cost forwarding in large sensor networks," in *Proceedings of the Tenth International Conference on Computer Communications and Networks*, pp. 304–309, 15-17 Oct. 2001.

[5] J. Faruque and A. Helmy, "Gradient-Based Routing in Sensor Networks," *Mobile Computing and Communications Review*, vol. 2, no. 4, Oct 2003.

[6] M. Chu et al., "Scalable information-driven sensor querying and routing for ad hoc heterogeneous sensor networks," *International Journal of High Performance Computing Applications*, vol. 16, no. 3, 2002.

[7] N. Sadagopan et al., "Active query forwarding in sensor networks," *Journal of Ad Hoc Networks*, vol. 3, no. 1, pp. 91–113, 2005.

[8] C. Perkins et al., "Ad hoc On-Demand Distance Vector (AODV) Routing," IETF RFC 3561, RFC Editor, 2003.

[9] M. Musolesi and C. Mascolo, "CAR: Context-Aware Adaptive Routing for Delay-Tolerant Mobile Networks," *IEEE Transactions on Mobile Computing*, vol. 8, no. 2, pp. 246–260, Feb. 2009.

[10] W. R. Heinzelman et al., "Energy-efficient communication protocol for wireless microsensor networks," in *Proceedings of the 33rd Annual Hawaii International Conference on System Sciences*, vol. 2, p. 10, 4-7 Jan. 2000.

[11] Y. Wei and H. John, "Medium access control in wireless sensor networks," in *Wireless sensor networks*, pp. 73–91, Kluwer Academic Publishers, 2004.

[12] K. Kalpakis et al., "Efficient algorithms for maximum lifetime data gathering and aggregation in wireless sensor networks," *Comput. Netw.*, vol. 42, no. 6, pp. 697–716, 2003.

[13] A. Manjeshwar and D. P. Agrawal, "TEEN: a routing protocol for enhanced efficiency in wireless sensor networks," in *Proceedings 15th International Parallel and Distributed Processing Symposium*, pp. 2009–2015, 2001.

[14] A. Manjeshwar and D. P. Agrawal, "APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks," *International Parallel and Distributed Processing Symposium*, vol. 2, 2002.

[15] K. Shorabi et al., "Protocols for self-organization of a wireless sensor network," *IEEE Personal Communications*, vol. 7, no. 5, pp. 16–27, 2000.

[16] J.N. Al-Karaki et al., "Data aggregation in wireless sensor networks - exact and approximate algorithms," in *Workshop on High Performance Switching and Routing, HPSR*, pp. 241 – 245, April 2004.

[17] Q. Li et al., "Hierarchical Power-aware Routing in Sensor Networks," in *Proceedings of the DIMACS Workshop on Pervasive Networking*, 2001.

[18] W.R. Heinzelman et al., "Adaptive protocols for information dissemination in wireless sensor networks," in *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '99, (New York, NY, USA), pp. 174–185, ACM, 1999.

[19] Y. Wang et al., "Transport layer approaches for improving idle energy in challenged sensor networks," in *Proceedings of the 2006 SIGCOMM workshop on Challenged networks*, CHANTS '06, (New York, NY, USA), pp. 253–260, ACM, 2006.

[20] F. Kuhn et al., "Worst-case optimal and average-case efficient geometric ad-hoc routing," in *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking & computing*, MobiHoc '03, (New York, NY, USA), pp. 267–278, ACM, 2003.

[21] S. Lindsey and C. S. Raghavendra, "PEGASIS: Power-efficient gathering in sensor information systems," in *IEEE Aerospace Conference Proceedings*, vol. 3, pp. 1125–1130, 9–16 Mar. 2002.

[22] Y. Yu et al., "Geographical and Energy Aware Routing: a recursive data dissemination protocol for wireless sensor networks," *Energy*, vol. 463, 2001.

[23] Y. Xu et al., "Geography-informed energy conservation for ad hoc routing," in *Proceedings of the 7th annual international conference on Mobile computing and networking*, MobiCom '01, (New York, NY, USA), pp. 70–84, ACM, 2001.

[24] O. Chipara et al., "Real-time power-aware routing in sensor networks," in *14th IEEE International Workshop on Quality of Service, IWQoS*, pp. 83–92, 2006.

[25] I.F. Akyildiz et al., "A survey on sensor networks," *IEEE Communications Magazine*, vol. 40, no. 8, pp. 102–114, Aug 2002.

[26] P. Beyens et al., "Maccoin: Medium acces control for wireless sensor networks based on collective intelligence," in *Proceedings of GTDT'04, held at*

the 3th International Conference on Autonomous Agents and Multi-Agent Systems, (New York, USA), pp. 1–12, 2004.

[27] K. Shah and M. Kumar, "Resource management in wireless sensor networks using collective intelligence," in *International Conference on Intelligent Sensors, Sensor Networks and Information Processing, ISSNIP 2008.*, pp. 423 –428, 15-18 Dec. 2008.

[28] S. Gajurel and M. Heiferling, "Swarm intelligent routing solution for wireless sensor networks," in *2010 IEEE 35th Conference on Local Computer Networks (LCN)*, pp. 707 –714, 10-14 Oct. 2010.

[29] R. Wang, "Swarm intelligence for the self-organization of wireless sensor network," in *IEEE Congress on Evolutionary Computation, CEC*, pp. 838 –842, 2006.

[30] S. Sivakumar et al., "Adaptive sampling for node discovery: Wildlife monitoring & sensor network," in *16th Asia-Pacific Conference on Communications (APCC), 2010*, pp. 447 –452, 31 Oct - 3 Nov 2010.

[31] S. Sivakumar and A. Al-Anbuky, "Analysis of network connectivity: Wildlife and Sensor Network," in *Australasian Telecommunication Networks and Applications Conference (ATNAC), 2009*, pp. 1–6, 10-12 Nov 2009.

[32] S. Sivakumar and A. Al-Anbuky, "Vehicular Networks Smart Connectivity," in *Eight International Conference on Wireless and Mobile Communication ICWMC, 2012*, pp. 304 –310, 24-29 June 2012.

[33] A. Demers et al., "Epidemic algorithms for replicated database maintenance," in *Proceedings of the sixth annual ACM Symposium on Principles of distributed computing*, PODC '87, (New York, NY, USA), pp. 1–12, ACM, 1987.

[34] A. Vahdat and D. Becker, "Epidemic routing for Partially-Connected ad hoc networks," tech. rep., Duke University, 2000.

[35] H.J. Zygmunt et al., "Gossip-based ad hoc routing," *IEEE/ACM Trans. Netw.*, vol. 14, no. 3, pp. 479–491, 2006.

[36] D. Braginsky and D. Estrin, "Rumor routing algorthim for sensor networks," in *Proceedings of the 1st ACM international workshop on Wireless sensor networks and applications*, WSNA '02, (New York, NY, USA), pp. 22–31, ACM, 2002.

[37] H. Karl and A. Willig, *Protocols and architectures for wireless sensor networks*. John Wiley & Sons, Oct. 2007.

[38] D. Raffo, *Security Schemes for the OLSR Protocol for Ad Hoc Networks*. PhD thesis, Universit Paris 6 and INRIA Rocquencourt, 2005.

[39] T. Spyropoulos et al., "Spray and wait: an efficient routing scheme for intermittently connected mobile networks," in *Proceedings of the 2005 ACM SIGCOMM workshop on Delay-tolerant networking*, WDTN '05, (New York, NY, USA), pp. 252–259, ACM, 2005.

[40] A. Lindgren et al., "Probabilistic Routing in Intermittently Connected Networks," *Lecture Notes in Computer Science*, vol. 3126, pp. 239–254, Jan. 2004.

[41] J. Burgess et al., "Maxprop: Routing for vehicle-based disruption-tolerant networks," in *Proceedings, 25th IEEE International Conference on Computer Communications, INFOCOM 2006*, pp. 1–11, 2006.

[42] S.C. Nelson et al., "Encounter-Based Routing in DTNs," in *The 28th Conference on Computer Communications, IEEE INFOCOM 2009*, pp. 846–854, Apr. 2009.

[43] P. Hui, J. Crowcroft, and E. Yoneki, "Bubble rap: social-based forwarding in delay tolerant networks," in *Proceedings of the 9th ACM international symposium on Mobile ad hoc networking and computing*, MobiHoc '08, (New York, NY, USA), pp. 241–250, ACM, 2008.

[44] P. Costa et al., "Socially-aware routing for publish-subscribe in delay-tolerant mobile ad hoc networks," *IEEE Journal on Selected Areas in Communications*, vol. 26, no. 5, pp. 748 –760, June 2008.

[45] A. Mtibaa et al., "Peoplerank: Social opportunistic forwarding," in *IEEE INFOCOM Proceedings*, pp. 1 –5, March 2010.

[46] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," *Comput. Netw. ISDN Syst.*, vol. 30, pp. 107–117, Apr. 1998.

[47] T. Camp et al., "A Survey of Mobility Models for Ad Hoc Network Research," *Wireless Communications & Mobile Computing (WCMC): Special issue on Mobile Ad Hoc Networking: Research, Trends and Applications*, vol. 2, no. 5, pp. 483–502, 2002.

[48] S. Yang et al., "Using social network theory for modeling human mobility," *Netwrk. Mag. of Global Internetwkg.*, vol. 24, no. 5, pp. 6–13, Sept. 2010.

[49] M. Musolesi and C. Mascolo, "Mobility Models for Systems Evaluation. A Survey," in *Middleware for Network Eccentric and Mobile Applications* (B. Garbinato, H. Miranda, and L. Rodrigues, eds.), pp. 43–62, Springer, February 2009.

[50] N. Eagle and A. S. Pentland, "Reality mining: sensing complex social systems," *Personal Ubiquitous Computing*, vol. 10, no. 4, pp. 255–268, Mar. 2006.

[51] F. Bai and A. Helmy, *A SURVEY OF MOBILITY MODELS in Wireless Adhoc Networks.* Kluwer Academic Publishers.

[52] M. Musolesi and C. Mascolo, "A community based mobility model for ad hoc network research," in *Proceedings of the 2nd international workshop on Multi-hop ad hoc networks: from theory to reality*, REALMAN '06, (New York, NY, USA), pp. 31–38, ACM, 2006.

[53] M. E. J. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Feb. 2004.

[54] S. Wasserman and K. Faust, *Social Network Analysis: methods and applications*. Cambridge University Press, 1994.

[55] A. Muhammad et al., "Integrated Mobility Model (IMM) for VANETs simulation and its impact," in *International Conference on Emerging Technologies*, pp. 452–456, IEEE, Oct. 2009.

[56] B. Fan et al., "The important framework for analyzing the impact of mobility on performance of routing protocols for adhoc networks," *Ad-Hoc Networks Journal*, vol. 1, no. 4, pp. 383–403, 2003.

[57] M. Abdelmoumen et al., "How to improve the performance in delay tolerant networks under manhattan mobility model," in *Proceedings of the IEEE 21st International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC 2010, 26-29 September 2010, Istanbul, Turkey*, pp. 2008–2013, IEEE, 2010.

[58] J. Härri et al., "Mobility models for vehicular ad hoc networks: A survey and taxonomy," *Communications Surveys Tutorials, IEEE*, vol. 11, no. 4, pp. 19–41, 2006.

[59] C. Boldrini et al., "Bio-inspired computing and communication," ch. User-Centric Mobility Models for Opportunistic Networking, pp. 255–267, Berlin, Heidelberg: Springer-Verlag, 2008.

[60] V. Dyo and C. Mascolo, "Efficient Node Discovery in Mobile Wireless Sensor Networks," in *DCOSS '08: Proceedings of the 4th IEEE international conference on Distributed Computing in Sensor Systems*, (Berlin, Heidelberg), pp. 478–485, Springer-Verlag, 2008.

[61] V. Dyo and C. Mascolo, "A node discovery service for partially mobile sensor networks," in *Proceedings of the 2nd international workshop on Middleware for sensor networks*, MidSens '07, (New York, NY, USA), pp. 13–18, ACM, 2007.

[62] V. Iyer et al., "Netdetect: Neighborhood discovery in wireless networks using adaptive beacons," in *Proceedings of the 2011 IEEE Fifth Interna-*

tional Conference on Self-Adaptive and Self-Organizing Systems, SASO '11, (Washington, DC, USA), pp. 31–40, IEEE Computer Society, 2011.

[63] M. Kohvakka et al., "Energy-efficient neighbor discovery protocol for mobile wireless sensor networks," *Ad Hoc Networks*, vol. 7, no. 1, pp. 24 – 41, 2009.

[64] R. Madan and S. Lall, "An energy-optimal algorithm for neighbor discovery in wireless sensor networks," *Mob. Netw. Appl.*, vol. 11, no. 3, pp. 317–326, June 2006.

[65] F. Reichenback et al., "A distributed linear least squares method for precise localization with low complexity in wireless sensor networks," in *Wireless Sensor Networks, IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS06)*, pp. 514–528, Springer Verlag, Juni, 2006.

[66] A. Bertrand and M. Moonen, "Consensus-based distributed total least squares estimation in ad hoc wireless sensor networks," *IEEE Transactions on Signal Processing*, vol. 59, no. 5, pp. 2320 –2330, May 2011.

[67] Kalman et al., "A New Approach to Linear Filtering and Prediction Problems," *Transactions of the ASME–Journal of Basic Engineering*, vol. 82, Series D, pp. 35–45, 1960.

[68] Z. Ye and H. Mohamadian, "WSN topology control design via integration of kalman filtering and adaptive estimation," in *6th International Conference on Electrical Engineering, Computing Science and Automatic Control, CCE, 2009*, pp. 1 –5, Jan. 2009.

[69] A. Shareef et al., "Localization using neural networks in wireless sensor networks," in *Proceedings of the 1st international conference on MOBILe Wireless MiddleWARE, Operating Systems, and Applications*, MOBILWARE '08, 2007.

[70] C. Jian et al., "Modeling and extending lifetime of wireless sensor networks using genetic algorithm," in *Proceedings of the first ACM/SIGEVO Sum-*

mit on Genetic and Evolutionary Computation, GEC '09, (New York, NY, USA), pp. 47–54, ACM, 2009.

[71] F. Michaud et al., “Learning from history for behavior-based mobile robots in non-stationary conditions,” *Auton. Robots*, vol. 5, no. 3-4, pp. 335–354, 1998.

[72] A. Wong and M. H. Ang Jr., “Integration of soft computing towards autonomous legged robots,” in *Autonomous robotic systems: soft computing and hard computing methodologies and applications*, pp. 323–352, Physica-Verlag GmbH, 2003.

[73] A. S. Bashi and E. J. Kaminsky, “Comparison of neural network and extended kalman filter determination of kinematics from impact acceleration tests,” in *IEEE International Conference on Systems, Man, and Cybernetics, 'Computational Cybernetics and Simulation'*, pp. 3501–3506 vol.4, 1997.

[74] R. Rojas, *Neural Networks: A Systematic Introduction.* Springer, 1 ed., July 1996.

[75] E. Oja, “Unsupervised learning in neural computation,” *Theor. Comput. Sci.*, vol. 287, no. 1, pp. 187–207, Sept. 2002.

[76] M. Amini et al., “RT-UNNID: A practical solution to real-time network-based intrusion detection using unsupervised neural networks,” *Computers & Security*, vol. 25, no. 6, pp. 459 – 468, 2006.

[77] M. Markou and S. Singh, “Novelty detection: A review - part 2: Neural network based approaches,” *Signal Processing*, vol. 83, pp. 2499–2521, 2003.

[78] G. A. Carpenter and S. Grossberg, “The ART of adaptive pattern recognition by a self-organizing neural network,” *IEEE Computer*, no. 3, pp. 77–88, 1988.

[79] Y. Gu et al., “Target coverage problem in wireless sensor networks: A column generation based approach,” in *6th International Conference on Mobile Adhoc and Sensor Systems, MASS '09. IEEE*, pp. 486–495, 2009.

[80] W. Xueqing and Z. Shuqin, "Research on Efficient Coverage Problem of Node in Wireless Sensor Networks," in *Second International Symposium on Electronic Commerce and Security, ISECS '09.*, vol. 2, pp. 532–536, 2009.

[81] C. Jae-Hwan and T. Leandros, "Maximum lifetime routing in wireless sensor networks," *IEEE/ACM Trans. Netw.*, vol. 12, no. 4, pp. 609–619, 2004.

[82] F. Ingelrest et al., "Optimal Transmission Radius for Energy Efficient Broadcasting Protocols in Ad Hoc and Sensor Networks," *IEEE Trans. Parallel Distrib. Syst.*, vol. 17, no. 6, pp. 536–547, 2006.

[83] M. Z. Zuniga and B. Krishnamachari, "Optimal transmission radius for flooding in large scale sensor networks," in *Proceedings of 23rd International Conference on Distributed Computing Systems Workshops*, pp. 697–702, 2003.

[84] A. Mishra et al., "An empirical analysis of the IEEE 802.11 MAC layer handoff process," *SIGCOMM Comput. Commun. Rev.*, vol. 33, no. 2, pp. 93–102, 2003.

[85] S. Waharte et al., "Selective active scanning for fast handoff in wlan using sensor networks," in *Mobile and Wireless Communication Networks*, pp. 59–70, 2005.

[86] E. Shih et al., "Wake on wireless: an event driven energy saving strategy for battery operated devices," in *MobiCom '02: Proceedings of the 8th annual international conference on Mobile computing and networking*, (New York, NY, USA), pp. 160–171, ACM, 2002.

[87] A. Husnain et al., "Signal strength based link sensing for mobile ad-hoc networks," *Telecommunication Systems*, vol. 42, no. 3, pp. 201–212, 2009.

[88] S.Y. Wang et al., "Signal strength-based routing protocol for mobile ad hoc networks," in *19th International Conference on Advanced Information Networking and Applications, AINA 2005*, pp. 17–20 vol.2, March, 2005.

[89] J. Liu, "Signal Strength based Service Discovery (S3D) in Mobile Ad Hoc Networks," *Proceedings of the 16th Annual IEEE International Symposium on Personal Indoor and Mobile Radio Communications (PIMRC)*, 2005.

[90] M. Musolesi and C. Mascolo, "A framework for multi-region delay tolerant networking," in *Proceedings of the 2008 ACM workshop on Wireless networks and systems for developing regions*, WiNS-DR '08, (New York, NY, USA), pp. 37–42, ACM, 2008.

[91] S.A. Munir et al., "Fuzzy Logic Based Congestion Estimation for QoS in Wireless Sensor Network," in *IEEE Wireless Communications and Networking Conference, WCNC 2007*, pp. 4336–4341, 2007.

[92] "Ptolemy project." `http://ptolemy.eecs.berkeley.edu/ptolemyII/`, August 2012.

[93] "Prowler: Probabilistic wireless network simulator." `http://www.isis.vanderbilt.edu/projects/nest/prowler`, August 2012.

[94] "J-sim." `http://j-sim.cs.uiuc.edu`, August 2012.

[95] "Java in simulation time / scalable wireless ad hoc network simulator." `http://jist.ece.cornell.edu/sw.html`, August 2012.

[96] "SENS: A sensor, environment and network simulator." `http://osl.cs.uiuc.edu/sens/`, August 2012.

[97] S. Sundresh, W. Kim, and G. Agha, "SENS: A sensor, environment and network simulator," 2004.

[98] "OPNET Modeler." `http://www.opnet.com`, August 2012.

[99] "OMNeT++ Network Simulation Framework." `http://www.omnetpp.org`, August 2012.

[100] "MiXiM." `http://mixim.sourceforge.net`, August 2012.

[101] "Castalia: A wireless sensor network simulator." `http://www.omnetpp.org/component/content/article/8-news/3478`, August 2012.

[102] "The network simulator -NS-2." http://nsnam.isi.edu/nsnam/index.php/User_Information, August 2012.

[103] "Global Mobile Information Systems Simulation Library (GloMoSim)." http://pcl.cs.ucla.edu/projects/glomosim/, August 2012.

[104] "QualNet." http://www.scalable-networks.com/content/products/qualnet, August 2012.

[105] "sQualnet: A Scalable Simulation Framework for Sensor Networks." http://nesl.ee.ucla.edu/projects/squalnet/, August 2012.

[106] M. Varshney and R. Bagrodia, "sQualNet version 1.0 beta: Tutorial and programmers manual." http://nesl.ee.ucla.edu/projects/squalnet/internal/software/sQualnet-tutorial.pdf, August 2012.

[107] "Netanim." http://www.nsnam.org/docs/release/3.11/models/html/animation.html, August 2012.

[108] G. J. Pottie and W. J. Kaiser, "Wireless integrated network sensors," *Commun. ACM*, vol. 43, no. 5, pp. 51–58, May 2000.

[109] Y. Liang and W.Peng, "Minimizing energy consumptions in wireless sensor networks via two-modal transmission," *SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 1, pp. 12–18, Jan. 2010.

[110] "Karongwe private game reserve." http://www.karongwe.co.za/Map.htm, Oct 2012.

[111] M.B. Lehmann et al., "Home range utilisation and territorial behaviour of lions (panthera leo) on karongwe game reserve, south africa," *PLoS ONE*, vol. 3, no. 12, p. 3998, 12 2008.

[112] V. Dham, "Link establishment in ad hoc networks using smart antennas," Master's thesis, Faculty of the Virginia Polytechnic Institute and State University, 2003.

[113] P. Levis, "TOSSIM." http://www.cs.berkeley.edu/~pal/research/tossim.html, August 2012.

[114] A. Muhammad et al., "Modelling execution time and energy consumption in sensor node simulation," *PIK Journal, Special Issue on Energy Aware Systems*, vol. 32, no. 2, pp. 127–132, 2009.

[115] "Avrora: The avr simulation and analysis framework." http://compilers.cs.ucla.edu/avrora/, August 2012.

[116] "Atemu." http://www.hynet.umd.edu/research/atemu/, August 2012.

[117] "EmStar." http://static.usenix.org/event/usenix04/tech/general/full_papers/girod/girod_html/eu.html, August 2012.

[118] J. Elson et al., "Emstar: An environment for developing wireless embedded systems software," 2003.

[119] "Moterunner." http://www.zurich.ibm.com/moterunner/, August 2012.

[120] "SunSpot." http://www.sunspotworld.com, August 2012.

[121] "SCADDS." http://www.isi.edu/scadds/pc104testbed/pc104testbed.html, August 2012.

[122] "MoteLab: Harvard sensor network testbed." http://motelab.eecs.harvard.edu, August 2012.

[123] "ORBIT." http://www.orbit-lab.org, August 2012.

[124] "Twist:tkn wireless indoor sensor network testbed." http://www.twist.tu-berlin.de/wiki, August 2012.

[125] "INDRIYA." http://indriya.comp.nus.edu.sg/motelab/html/index.php, August 2012.

[126] "The WUSTL wireless sensor network testbed." http://wsn.cse.wustl.edu/index.php/The_WUSTL_Wireless_Sensor_Network_Testbed, August 2012.

[127] "SensorScope." http://www.sensorscope.ch/index.php?n=En.Home, August 2012.

[128] G. Barrenetxea et al., "Sensorscope: Out-of-the-box environmental monitoring," *International Conference on Information Processing in Sensor Networks*, pp. 332–343, 2008.

[129] "EmuLab." http://www.emulab.net, August 2012.

[130] "Wisebed." http://wisebed.eu/site/conduct-experiments/testbeds/, August 2012.

[131] "Tarwis." http://wisebed.eu/site/conduct-experiments/web-clients/tarwis/s/, August 2012.

[132] "Wiselib." http://wisebed.eu/site/application-development/wiselib/, August 2012.

[133] "Shawn." http://wisebed.eu/site/application-development/shawn/, August 2012.

[134] "Smartsantander." http://www.smartsantander.eu, August 2012.

Appendix A

OPNET and Matlab connectivity configuration

OPNET and Matlab are two distinct softwares which need to be interconnected for passing data between them. This involves setup of compiler and linker options in OPNET [112], setting of environment variable in the Windows operating system and register the Matlab server.

- **Configurations for OPNET:** In OPNET Network Editor under the DES tab Configure/Descrete Event Simulation has to be selected '$DES- > Configure/DescreteEventSimulation$' The window which opens is shown in Figure A.1. On the left hand side of the window select the following '$Execution- > Advanced- > Compilation- > Common\,Compiler\ flags$' The value for common compiler flags is set as follows:

```
/W3 /D_CRT_SECURE_NO_DEPRECATE /IC:\PROGRA~1\OPNET\145~1.A\models\std
\include /IC:\PROGRA~1\MATLAB\R2009b\extern\include
```

  Next the linker has to be connected by setting the linking options at '$Execution- > Advanced- > Linking- > Common\ Linking\ flags$' as shown in Figure A.2:

  Under linking, the Development Kernel Libraries and Optimized Kernel Libraries are set with the same values as follows: Development Kernel Libraries *libeng.lib libmat.lib libmx.lib libmex.lib libut.lib* Optimized Kernel Libraries *libeng.lib libmat.lib libmx.lib libmex.lib libut.lib*

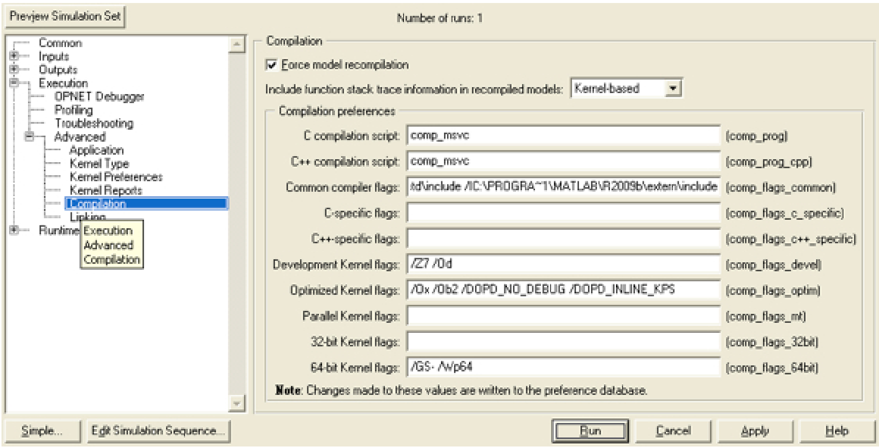


Figure A.1: OPNET compilation settings

/LIBPATH:C:\PROGRA~1\MATLAB\R2009b\extern\lib\win32\ /LIBPATH:C:\PROGRA~1\MATLAB\R2009b\extern\lib\win32\microsoft\ /LIBPATH:C:\PROGRA~1\MATLAB\R2009b\bin\win32\

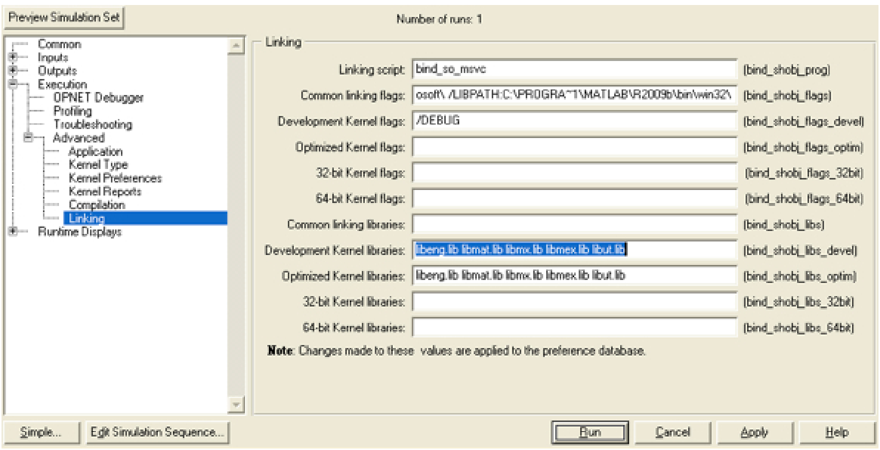


Figure A.2: OPNET linker settings

- **Configurations for Matlab:** In Matlab '*MCRInstaller*' has to be installed. The installer is present in the following path:

  After this on the Matlab command line the following command should be issued: *regmatlabserver* to test whether the installation is successful. On issuance of the command a Matlab command line window comes up.

- **Windows Environment Variables:** The connection of OPNET and Matlab requires setting up Path variable of Windows Environment Variables as follows:

```
C:\Program Files\MATLAB\MATLAB Compiler Runtime\v711\runtime\win32;C:\Program
Files\MATLAB\R2009b\bin\win32;C:\Program Files\MATLAB\R2009b\extern\lib
\win32\microsoft
```

The given paths for compiler and linker are for a 32 bit Windows XP machine with OPNET 14.5. The paths may change for newer version of the software (Matlab and OPNET).
The functions $'engOpen()'$ is used to call the Matlab engine and $'engClose()'$ is used to close the engine. $'engPutArray()'$ and $'engGetArray()'$ are used to put and get variable to and from Matlab workspace.

Appendix B

Impact of Sampling frequency on Successful Node Discovery

The density of nodes (number of nodes per unit area) play an important role in providing good connectivity. With a varying density network there are pockets where density of nodes is high and other regions where its sparse.

The density of node $\rho =$ number of nodes n/a, a is the area. Application with variable density of nodes requires the data is forwarded towards the destination opportunistically. If the average displacement between the nodes is d_{dist} and if the mobile node is moving at a velocity V, then the time required for successfully finding a next hop node would be $Tsecs$ given as $T = \frac{d_{dist}}{V}$. The average displacement in terms of the density of nodes considering a square area is given as $d_{dist} = \sqrt{\frac{n}{\rho}}$.
$\therefore$ to successfully find a next hop node the time required is given as

$$T = \frac{\sqrt{\frac{n}{\rho}}}{V} \tag{1}$$

If we sample for node discovery at a frequency f, then the probability of successfully finding a next hop node is given by

$$X_{prob} = \frac{\sqrt{\frac{n}{\rho}}}{V.T} f \tag{2}$$

The Figure B.1 shows that with the increase in number of nodes and frequency of sampling, the probability of a successful node discovery increases. The trend

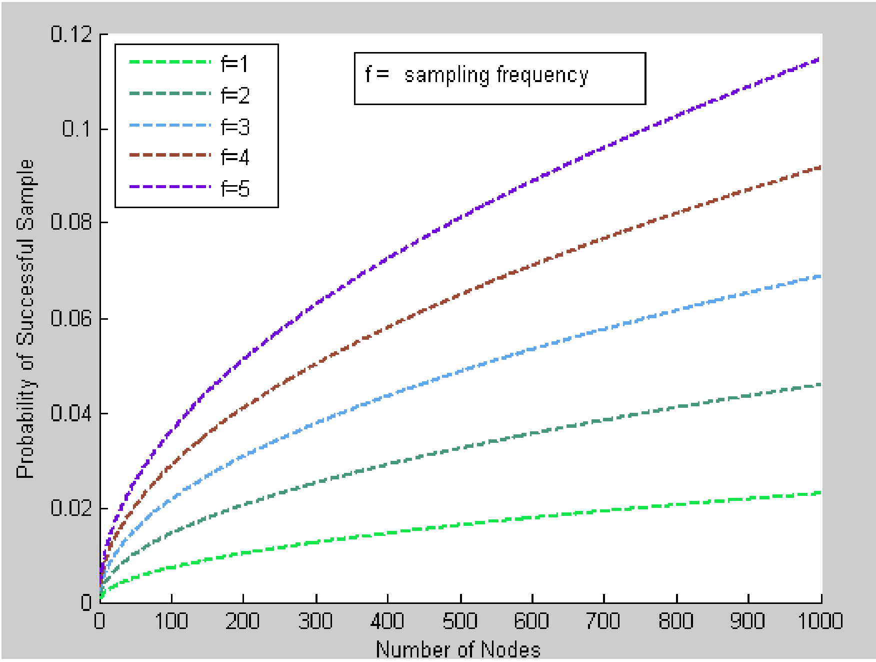


Figure B.1: Probability of a Successful Node Discovery

shown in the graphs appear promising for larger node densities. To address the issue of sparsity of nodes a suitable value of sampling frequency f needs to be determined. Due to the dynamically changing dynamics of the network, fixing the sampling frequency or sampling time $T = 1/f$ to a particular value might not always be optimal for all network scenarios. Therefore adaptive sampling is used to discover nodes in an energy–efficient manner.

Appendix C

Emulators:

Emulators are combination of actual hardware and computer simulations. Advantage of emulators is that they obtain actual data from the hardware. Unlike simulators, emulators capture effect of terrain by monitoring signal strengths. Energy consumption model would be accurate as the energy consumption of the actual hardware is monitored. The major issue with emulators is the scalability in terms of the number of actual nodes that can be deployed.
TOSSIM[113] is a discrete event based simulator for TinyOS, which is scalable for thousands of sensor nodes. Even though TOSSIM is termed as a simulator, it is designed specifically for TinyOS and hence emulates TinyOS code. The code written for TOSSIM can be directly ported on the hardware supporting TinyOS. The tool hides the low level layer with a simulation wrapper [114].
Avrora[115] is a simulation and analysis tool designed specifically for AVR microcontrollers and mica2 sensor nodes. It can simulate TinyOS code and give a level of confidence before the actual implementation on the microcontroller. Avrora is not a general purpose simulator to do designing of nodes and performance analysis of different protocols. Atemu[116] is a emulator for AVR based systems and partial support for mica2 based systems. The emulator has the capability to simulate large number of nodes and also simulate the working of TinyOS code. The tool provides a graphical debugging environment that allows source level debugging support to the developers. The tool is restricted to Linux distributions and Solaris.
Emstar[117] is a Linux based wireless sensor network software. It has a modu-

lar programming model. Services may be flexibly interconnected using standard interfaces. The connections can transmit a flow of packets, stream data, state updates, or configuration commands. EmStar provides a number of development modes (simulation and emulation) that facilitate debugging and evaluation [118]. Moterunner[119] software is a two part software consisting of a runtime environment of hardware nodes and a development environment. The tool allows programming in Java and uses an Eclipse based integrated development environment. It offers a web–based deployment and monitoring framework that emulates actual hardware as well as virtual hardware. SunSpots[120] are nodes developed by Oracle. It comes with Spot Manager that is used to upload code onto the nodes. Solarium is a tool that helps visualise nodes and obtain graphs of sensor data. Code is written in Java and can be uploaded to the nodes over the air. Solarium allows emulation of code on actual or virtual nodes.

Testbeds:

Testbeds are deployment of wireless sensor nodes in a controlled space. Advantage of test beds is they provides actual data. Studying the effects of terrain, mobility, energy characteristics of the node and effectiveness of receiver signal strength–based distance estimation are more accurate when compared to simulation. The cost of establishment of test bed with 100 or more nodes is prohibitive, hence test bed implementations are few. Moreover the test beds are not very generic allowing testing of wide range of applications. The other major drawback is an interface to work with the test beds. The interfaces allow limited capabilities like monitoring sensor data, uploading applications or protocols and few limited analytical capabilities.
SCADDS (Scalable Coordination Architecture for Deeply Distributed Systems)[121] is a project in University of Southern California. The test bed is a distributed system that focuses on gathering the data from the system as a whole rather than concentrate on individual sensors that may have generated it. The test bed is being used to research on protocols like directed diffusion, time synchronisation, localisation and self configuration. MoteLab[122], a deployment of wireless sensor network in Maxwell Dworkin Laboratory, the Electrical Engineering and Computer Science building at Harvard University, hosts 190 TMote Sky sensor

nodes running TinyOS. The deployment is static and permanent. It is on public domain for registered users. Users can create and schedule work on the MoteLab. The execution data is logged that can later analysed. MoteLab currently hosts nodes that include light, humidity and temperature sensors and have an indoor transmission range of 100m. All nodes are connected to the ethernet, which enable easy access for loading and debugging programs. ORBIT[123] is a emulator and test bed that offers the ability to reproduce experimentations. It is operated and developed by WINLAB (Wireless Information Network Laborator), Rutgers University. The test bed consists of 400 IEEE 802.11 nodes. The emulator allows for developing applications and protocols, which are later deployed on the testbed for real life testing. The test bed is static with permanent deployment of nodes. TWIST[124] is a heterogeneous test bed that provides support for flat as well as hierarchical architectures. It has an active power supply control to switch between battery power and USB power. The test bed is an indoor deployment in TU Berlin campus. The current setup has 102 TmoteSky nodes and 102 eyesIFX nodes. TWIST follows a 3-tier architecture with sensing nodes, Super nodes [124] connected to USB hubs and server. Indriya[125] is a large scale test bed in National University of Singapore and uses TelosB nodes. Its architecture is similar to TWIST. It is a three floor static deployment and is powered through active–USB. The active–USB infrastructure is used to program nodes and provide power. The design of Indriya is cluster based with each cluster supporting 127 sensor devices. There are a total of 6 clusters currently. The user interaction with test bed is through a web interface. The test bed is currently being used for research on wireless channel performance and correlation. WUSTL[126] is a test bed that is currently under deployment in Washington University in St. Louis. It is expanding and currently hosts 79 nodes. The primary purpose of the test bed is to study communication characteristics of the nodes. The structure of test bed is hierarchical similar to TWIST(TKN Wireless Indoor Sensor network Testbed) [124]. It consists of NFS (Network File System) server, micro server and USB hubs for sensor nodes. The micro server and NFS server are connected via ethernet. The NFS server forms an interface for users to use the test bed. Python is used to program nodes over a serial interface.

These test beds are established indoor and are static. SensorScope[127], is a static outdoor testbed established by EPSL (Ecole Polytechnique Fédérale de Lausanne). It is established in the rock glaciers of Switzerland. The testbed measures environmental parameters like wind, temperature, humidity and luminosity. It makes data available publicly on Microsofts SensorMap website and Google Maps based web interface [128].

Emulab (Mobile Emulab: A Robotic Wireless and Sensor Network Testbed)[129] is a test bed deployed by University of Utah. Emulab offers a remote accessibility to mobile wireless and sensor testbed that is composed of robots carrying nodes and a single board computer in an indoor environment. The nodes run user software that is written as a script. The test bed allows users to interact in real time. It contains a vision based tracking system that is accurate to 1cm. The test bed is unique as it allows testing mobility based algorithms. The test bed resources are for public access.

WISEBED[130] and its federation of test beds offer a heterogenous deployment of nodes on a large scale. The test beds are deployed in, University of Lübek (UZL), Freie Universität Berlin (FUB), Braunschweig Institute of Technology (TUBS), Research Academic Computer Technology Institute (CTI), Universitat Politecnica de Catalunya (UPC), Universität Bern (UBERN), University of Geneva (UNIGE), Delft University of Technology (TUD), Lancaster University (ULANC). These test beds have different hardwares and softwares. WISEBED provides web clients, desktop clients and API's ((Application Programming Interface)) for one to design their own client, to connect to testbeds.

The programming and debugging issue in this heterogeneous testbed is resolved through TARWIS [131], which allows controlling the testbed through a website. It has multiple different components, TARWIS Server, TARWIS GUI, Reservation System, Identity Provider (IDP) and a web interface (IDP Tools), Service Provider (SP), and Sensor Network Authorization System (SNA) with a web interface (SNA GUI). Desktop clients like SpyGlass and WeyesBed provide visualization tool to visualise sensor networks and their communication links.

WISEBED comes with a library of algorithms as Wiselib [132]. The code is written in C++ and can be called in the Shawn simulator [133]. It provides an operating system Lorien OS for the sensor nodes. The OS is modular and modules can be added or removed from the nodes during operation. WISEBED lacks simulation capability of a general simulator like OPNET, NS2, OMNET++ and others. Though it provides the ability to reproduce results but is limited in its ability to simulate physical layer and packet level simulations. SmartSantander[134], a collaborator of WISEBED, envisions a deployment of 20,000 sensors in Belgrade, Guildford, Lübek and 12,000 sensor in Santander. The test bed is deployed as a 3-iter architecture with sensing nodes, repeaters and gateways. The test bed provisions for researchers and provide services. Therefore to separate the two operations the nodes, repeaters and gateways are equipped with two IEEE 802.15.4 modules. The test bed senses temperature, light, noise, presence of vehicles and carbon monoxide levels.

Emulators are highly specific to hardware architectures and suffer from low speed, limited scalability and platform dependence [114]. Testbeds are meant for actual implementation and testing. It is difficult to achieve repeatability on test beds due to the influence of multiple parameters. Simulators are generic, allowing repeatability of scenarios. Testing of protocol is easy as influence of different parameters can be monitored and controlled. Simulators provide a quick and scalable environment for designing and testing of protocols.

Appendix D

Details of Useful features in OPNET

1. **Scenario Duplication :**

Duplication of scenarios allow for changing and testing one parameter at a time which is useful for comparison.

OPNET features duplication of scenarios, which allows for comparison of their performance as shown in Figure D.1. In a typical opportunistic connectivity scenario considered in Figure D.1 (a) and (b), Figure D.1 (a) has all nodes static and they do not fall in each other's coverage area. The scenario is duplicated as Figure D.1 (b) and mobile node 0 is made mobile so that it encounters nodes 1, 2, 5, 4 and 3 in its path. The mobile node, on encounter with the stationary nodes 1, 2, 5, 4 and 3, transfers its data for the scenario shown in Figure D.1 (b) as shown in Figure D.1 (d). As the nodes in the scenario shown in Figure D.1 (a) are not in coverage range, the data transmitted by the mobile node 0 is not received by any stationary node as shown in Figure D.1 (c).

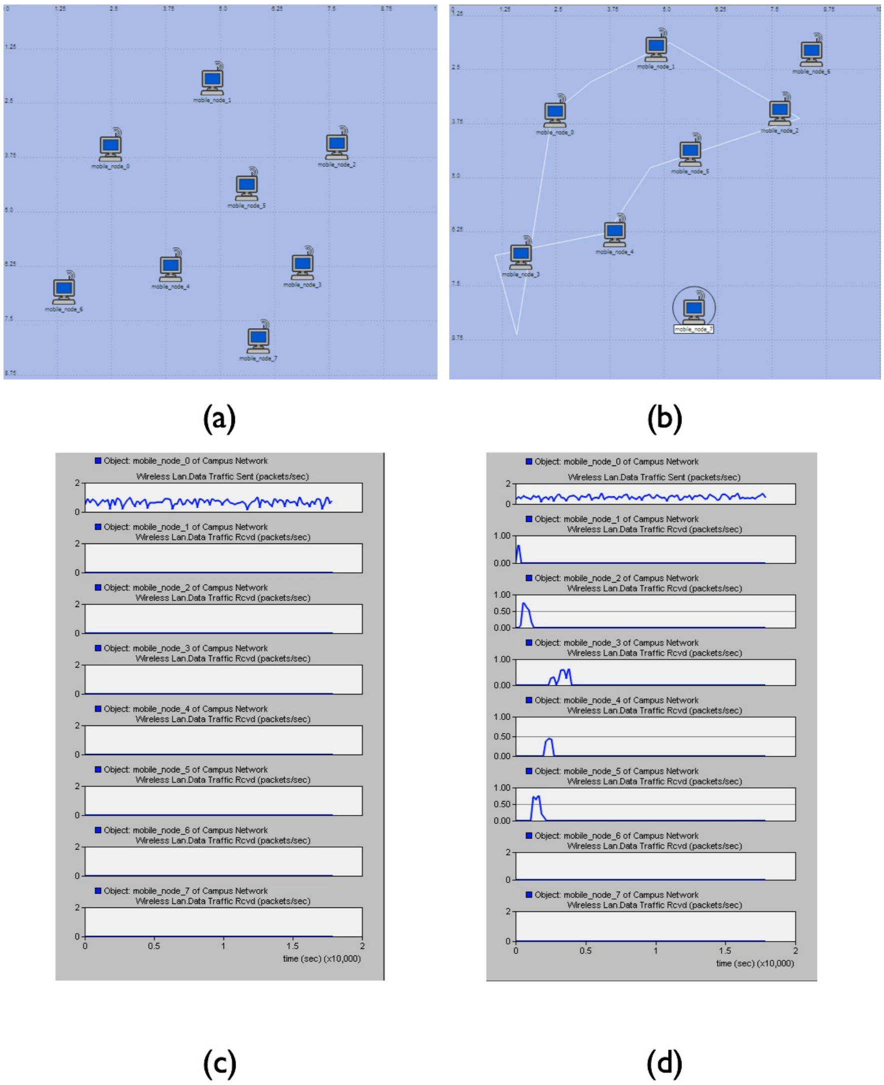


Figure D.1: Scenario duplication: (a) OPNET scenario with static nodes. (b) OPNET scenario with Node 0 mobile. (c) No data received by static nodes. (d) Data received by static nodes.

2. Node Modelling :

OPNET allows new nodes to be modelled or the use of existing nodes from the OPNET library to extend their capabilities. Nodes are made up of processors as shown in Figure D.2 (a), which are entities performing a specific task. The processors in Figure D.2 (a) encapsulate the process model as shown in Figure D.2 (c). To add the process model to the processor, the processor is right clicked using

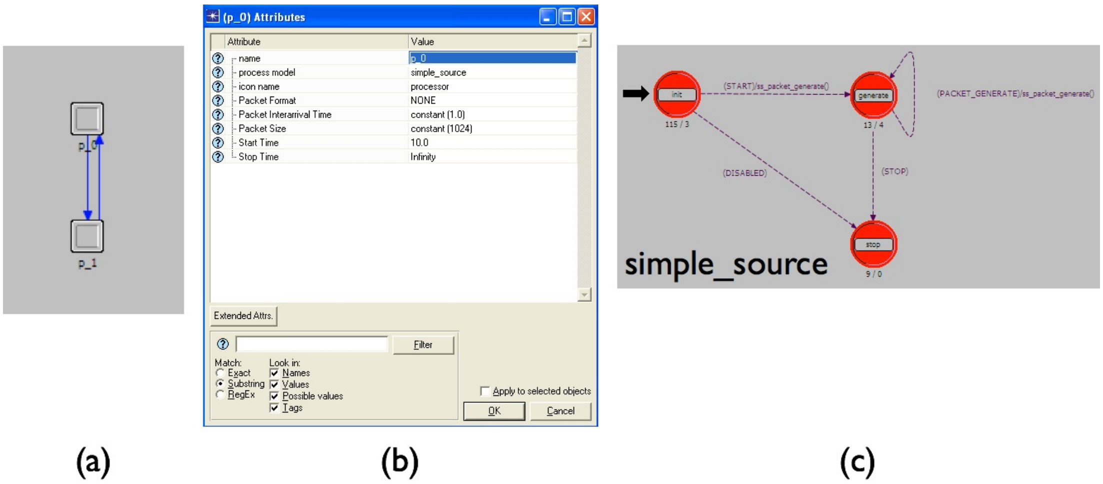


Figure D.2: Node modelling: (a) Node model showing processors. (b) Window to add process model to a processor. (c) Process model.

the mouse and '*process model*' option as shown in Figure D.2 (b) is selected to assign the desired process to the processor. The different processors are connected using streams which carry data amongst processors.

3. Process Modelling :

The process model allows the definition of finite state machines as shown in Figure D.3 (a). The green and the red states contain C code. Each state has an '*Enter*' and '*Exit*' executive. For green states, the control after executing the code in Enter executive goes to the Exit executive to execute its code, whereas for the red states, the control waits for an event interrupt to go from the Enter executive to Exit executive. The Process modelling environment stores variables, header files and function definitions in different blocks as shown in Figure D.3 (b). The variables can be state variables or temporary variables and are declared in their respective blocks. The state variables have a global scope within the process and can be referenced from other processors of the node. Temporary variables have a limited scope within the process. The header files, pre-processor directives, macros and function declarations go into the header block and the function definitions go into the function block.

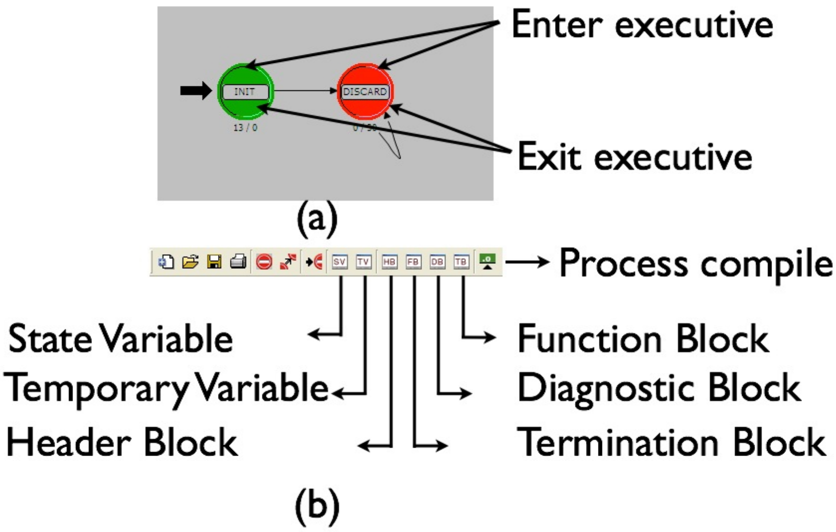


Figure D.3: Process modelling: (a) Different process model states. (b) Code and variable definition blocks

4. File operations :

OPNET can compile the ANSI-C code, which allows the use of FILE structure ('$FILE *fp$') to declare and use file pointers for file operations. This helps in the collection of node data in '*.csv*' file. The code to create '*.csv*' file is given in Figure D.4. OPNET assigns '*Object id*' which is an integer number to all entities of a

```
char node_name[128];
char flname[50];
char node_suffix[5]=".csv";
Objid node_objid;
FILE *fp;
strcpy(flname,"C:\\Documents and Settings");

node_objid = op_topo_parent(op_id_self());
op_ima_obj_attr_get (node_objid, "name", node_name);
strcat(node_name,node_suffix);
strcat(flname,node_name);

fp=fopen(flname,"a");

fprintf(fp,"%f,%f,%f\n",(double)velocity,(double)(yaw),(double)time_elapsed);

fclose(fp);
```

Figure D.4: Objective C code for data collection in '*.csv*' file

scenario (network, subnets, nodes and processors). The data type '*Objid*' holds this integer value. The OPNET kernel APIs '*op_topo_parent*()' and '*op_id_self*()' extract the nodes' object id, which is stored in the variable '*node_objid*'. The name assigned to the node in the network scenario is extracted using the API '*op_ima_obj_attr_get*()'.