A COMMUNICATION FRAMEWORK FOR MULTIHOP WIRELESS ACCESS AND SENSOR NETWORKS: ANYCAST ROUTING & SIMULATION TOOLS

by

Kashif Sharif

A dissertation submitted to the faculty of
The University of North Carolina at Charlotte
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in
Computing and Information Systems

Charlotte

2012

Approved by:

____
Dr. Teresa A. Dahlberg

____
Dr. Yu Wang

____
Dr. Jamie Payton (Souvenir)

____
Dr. Jiang (Linda) Xie

____
Dr. Lyndon P. Abrams

ABSTRACT

KASHIF SHARIF. A communication framework for multihop wireless access and sensor networks: Anycast routing & simulation tools. (Under the direction of DR. TERESA A. DAHLBERG)

The reliance on wireless networks has grown tremendously within a number of varied application domains, prompting an evolution towards the use of heterogeneous multihop network architectures. We propose and analyze two communication frameworks for such networks. A first framework is designed for communications within multihop wireless access networks. The framework supports dynamic algorithms for locating access points using anycast routing with multiple metrics and balancing network load. The evaluation shows significant performance improvement over traditional solutions. A second framework is designed for communication within sensor networks and includes lightweight versions of our algorithms to fit the limitations of sensor networks. Analysis shows that this stripped down version can work almost equally well if tailored to the needs of a sensor network. We have also developed an extensive simulation environment using NS-2 to test realistic situations for the evaluations of our work. Our tools support analysis of realistic scenarios including the spreading of a forest fire within an area, and can easily be ported to other simulation software. Lastly, we us our algorithms and simulation environment to investigate sink movements optimization within sensor networks. Based on these results, we propose strategies, to be addressed in follow-on work, for building topology maps and finding optimal data collection points. Altogether, the communication framework and realistic simulation tools provide a complete communication and evaluation solution for access and sensor networks.

ACKNOWLEDGEMENTS

I would like to express my sincere thanks to my advisor, committee members, research colleagues, family and friends, for their support and encouragement without which this research work would never have been possible.

I am especially thankful to Dr. Teresa Dahlberg, who not only served as my supervisor, but also encouraged and challenged me throughout my doctoral program. Working with her, and under her supervision in this research work and other projects has truly been a remarkable experience for me. The knowledge and experience that I have gained could not have been possible without her help. I also appreciate and thank her for her efforts in securing economical assistantships for my stay at UNC Charlotte.

I am grateful to my dissertation committee: Dr. Yu Wang, Dr. Jamie Payton, Dr. Linda Xie, and Dr. Lyndon Abrams, for their guidance and critical evaluation of my work. Dr. Wang and Dr. Payton have given me crucial guidance for which I am very thankful. I would also like to thank my co-workers and friends, Dr. Lijuan Cao, Dr. Marguerite Doman, and Dr. Sarah Khan for their collaborative work, encouragement, and help. Dr. Cao has been very instrumental in simulation work and refining of ideas for research.

I am also grateful to my family, who has supported me in all manners during my entire academic process. My mother has been the greatest source of encouragement, guidance, support and inspiration. She has helped through thick and thin of life, and I do not have words to express my thankfulness and appreciation for it.

This work would not have been possible without the generous financial support of Graduate School and Graduate & Professional Student Government, at UNC Charlotte.

TABLE OF CONTENTS

LIST OF FIGURES	vii
LIST OF ABBREVIATIONS	x
CHAPTER 1: INTRODUCTION	1
CHAPTER 2: LITERATURE REVIEW	7
2.1 Network Routing	7
2.2 Routing in MANETs	11
2.3 Routing in Sensor Networks	19
2.4 Network Routing Simulation Systems	25
CHAPTER 3: HYBRID ANYCAST ROUTING	30
3.1 Future Network Architecture	30
3.2 Challenges and Motivation	31
3.3 Hybrid Anycast Communication Framework	33
3.4 Hybrid Anycast Routing Protocol	38
3.5 Implementation & Evaluation of Framework using NS-2	45
CHAPTER 4: ANYCAST ROUTING FOR SENSOR NETWORKS	51
4.1 Sensor Networks	51
4.2 Challenges in Sensor Network Protocol Design	54
4.3 Sensor Network Communication Framework	55
CHAPTER 5: LIGHTWEIGHT HYBRIDANYCAST ROUTING PROTOCOL FOR SENSOR NETWORKS	86
5.1 Protocol Design Details	61
5.2 Evaluation and Analysis	63

CHAPTER 6: EVENT BASED AND REALISTIC SCENARIO SIMULATION IN NS-2 94

6.1 Event Based Data Generation 95

6.2 Capabilities of Network Simulation Software 97

6.3 Motivation and Challenges 97

6.4 Basic Proximity Based Models 99

6.5 Forest Fire Spreading Model 107

CHAPTER 7: MOBILITY OPTIMIZATION AND MULTI-TIER COMMUNICATION OF SINK NODES 115

7.1 Location Awareness by Mobile Sinks 116

7.2 Mobility Optimization using Path Cost Metrics 118

7.3 Topology based Connectivity Graph 120

CHAPTER 8: CONCLUSION AND FUTURE WORK 122

REFERENCES 126

LIST OF FIGURES

FIGURE 1.1: Mobile Ad hoc Network 2

FIGURE 1.2: Wireless Sensor Network 3

FIGURE 2.1: MANET routing protocol hierarchy 11

FIGURE 2.2: Sensor Network routing protocol hierarchy 22

FIGURE 3.1: Heterogeneous Network Architecture 30

FIGURE 3.2: Architecture of framework 34

FIGURE 3.3: Overhead vs. Proactive Radius – plot of $f(m)$ 43

FIGURE 3.4: AP Hello Packet 46

FIGURE 3.5: Route Request Packet 47

FIGURE 3.6: Route Reply Packet 47

FIGURE 4.1: Sensor network in a forest 52

FIGURE 4.2: Sensor network in a battle field 53

FIGURE 4.3: Sensor Network Communication Architecture 56

FIGURE 5.1: Hello Packet 62

FIGURE 5.2: Route Request Packet (RREQ) 63

FIGURE 5.3: Route Reply Packet (RREP) 64

FIGURE 5.4: Route Error Packet (RERR) 64

FIGURE 5.5: Delivery percentage for basic protocol operation 70

FIGURE 5.6: Delivery percentage against pause time of sink nodes 70

FIGURE 5.7: Delay of data packets measured in *milliseconds* (*ms)* 71

FIGURE 5.8: Delay of data packets against pause time of sink nodes (*milliseconds*) 71

FIGURE 5.9: Control overhear, measured as control bytes sent per data byte 72

FIGURE 5.10: Control overhead against pause time of sinks. (*Control byte per data byte*) 72

FIGURE 5.11: Delivery percentage for *'Set A'* 76

FIGURE 5.12: Delivery percentage against pause time of sink nodes in *'Set A'* 76

FIGURE 5.13: Delay in milliseconds (*ms*) for *'Set A'* 77

FIGURE 5.14: Delay measured in milliseconds (*ms*) against pause time of sinks in *'Set A'* 77

FIGURE 5.15: Control overhead in control bytes per data byte for *'Set A'* 78

FIGURE 5.16: Control overhead against pause time for *'Set A'* 78

FIGURE 5.17: Delivery percentage for *'Set B'* 79

FIGURE 5.18: Delivery percentage against pause time of sink nodes in *'Set B'* 79

FIGURE 5.19: Delay in milliseconds (*ms*) for *'Set B'* 80

FIGURE 5.20: Delay measured in milliseconds (*ms*) against pause time of sinks in *'Set B'* 80

FIGURE 5.21: Control overhead in control bytes per data byte for *'Set B'* 81

FIGURE 5.22: Control overhead against pause time for *'Set B'* 81

FIGURE 5.23: Control overhead comparison of *'Set A'* and *'Set B'* 82

FIGURE 5.24: Performance of '*Sink Proximity List*' technique 83

FIGURE 5.25: Control overhead comparison with and without *'Sink Proximity List'* technique, in Light-Weight Hybrid Anycast Routing 83

FIGURE 5.26: Maximum Speed (y-axis in meter per sec) and Pause Time combinations used in simulation 86

FIGURE 5.27: Performance results of Chaotic Movement Experiment 88

FIGURE 5.28: Light-Weight Anycast Routing Protocol performance results of mobility experiments 89

FIGURE 5.29: Dead node analysis for Light-Weight Hybrid Anycast Routing protocol 90

FIGURE 5.30: Random dead zones in the simulation environment 92

FIGURE 5.31: Performance of Dead Zone simulation 93

FIGURE 6.1: Nodes in close proximity of dead zone will generate data only 95

FIGURE 6.2: Images of real forest fires 96

FIGURE 6.3: Dead Zone and Proximity Zone 100

FIGURE 6.4: Dead & Proximity Zones in Circular Proximity Model 102

FIGURE 6.5: Extended Circles Proximity Model 104

FIGURE 6.6: Extended Circles Proximity model performance evaluation 106

FIGURE 6.7: Rectangular model for forest fire extension with direction variables 109

FIGURE 6.8: Example of Rectangular model for forest fire extension 110

FIGURE 6.9: Circular spreading directions and example arcs 112

FIGURE 6.10: Circular expansion on *e* arc 112

FIGURE 6.11: Performance of expanding circular model for forest fires 114

FIGURE 7.1: Path Back-Tracking Algorithm experiment 119

LIST OF ABBREVIATIONS

AODV	Ad hoc On-demand Distance Vector
AP	Access Point
BS	Base Station
CDMA	Code Division Multiple Access
CTS	Clear to Send
DCF	Distributed Coordination Function
DSR	Dynamic Source Routing
GPS	Global Positioning System
IEEE	Institute of Electrical and Electronic Engineers
IETF	Internet Engineering Task Force
IP	Internet Protocol
MAC	Medium Access Control
MANET	Mobile Ad hoc Network(s)
MN	Mobile Nodes(s)
NS-2	Network Simulator – 2
OTcl	Object Tool Command Language
QoS	Quality of Service
RFC	Request for Comments
RTS	Request to Send
Tcl	Tool Command Language
TTL	Time to Live
WSN	Wireless Sensor Network(s)

CHAPTER 1: INTRODUCTION

Wireless networks have evolved from supporting communications with simple transmitter-receiver equipment to support of highly sophisticated communication paradigms. Overall, wireless communication has been dominated by cellular communication, mainly due to the huge subscriber base and existing infrastructure. The wired infrastructure of local area networks has also been transformed, with replacement by wireless networks and the movement towards personal networking. Standardization bodies like the Internet Engineering Task Force (IETF) and the Institute for Electrical and Electronics Engineers (IEEE) have played key roles in providing specifications in this regard. To support this wireless paradigm, communication devices have become progressively more sophisticated. From huge bulky desktops, smart laptops have emerged with wireless interfaces as well as palm size computing devices. From brick size cellular phones to compact and stylish devices, wireless terminals have been reduced in size and increased in computing and battery power. The limitations of cellular communications (range, deployment, cost, speed, etc.) have led to other wireless architectures like Mobile Ad hoc networks, Mesh networks, and Sensor networks.

'A MANET is a network that results from the cooperative engagement of a collection of (mobile) nodes without any centralized control.' A Mobile Ad Hoc Network (MANET) is a network architecture that can be rapidly deployed without relying on pre-existing fixed network infrastructure. The nodes in a MANET can join and leave the

network dynamically, frequently, often without warning, and possibly without disruption to other node's communication. Finally, the nodes in the network can be highly mobile, thus rapidly changing the node constellation and the presence or absence of links.

Nodes in the MANET exhibit nomadic behavior by freely migrating within the same area, dynamically creating and tearing down associations with other nodes. Groups of nodes that have a common goal can create formations (clusters) and migrate together, similar to military units on missions or guided tours on excursions. Nodes can communicate with each other at any time and without restrictions, except for connectivity limitations and subject to security provisions. MANETs are intended to provide a data network that is immediately deployable in arbitrary communication environments and is responsive to changes in network topology. Because ad hoc networks are intended to be deployable anywhere, existing infrastructure might not be present. Therefore, the mobile nodes are likely to be the sole elements of the network. Different mobility patterns and radio propagation conditions that vary with time and position can result in intermittent and sporadic connectivity between adjacent nodes. The result is a time-varying network topology, with characteristics of self-organization, multihop communication, constrained energy consumption, constrained bandwidth links, and limited security.

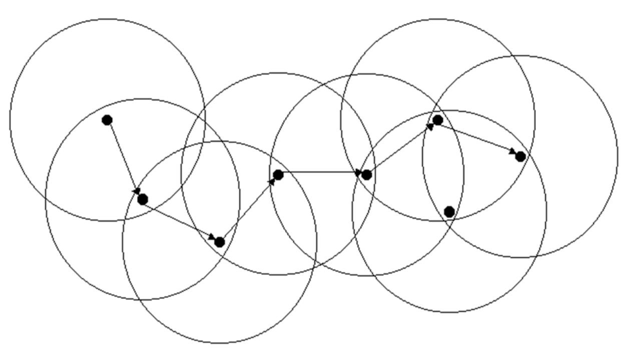


Figure 1.1: Mobile Ad hoc Network

A Sensor network (a specialized application of MANETs) is a collection of tiny devices that have the capability of collecting information from the environment. These tiny devices (in order of hundreds and thousands) can be randomly or strategically deployed in the target environment to monitor, analyze and even react to different stimuli. Thus the network as a whole is capable of providing access to information anywhere in the coverage area of network. The architecture of the sensor node's hardware consists of five components [1]: sensing hardware, processor, memory, power supply and transceiver. These devices are easily deployed because no infrastructure and human control is required. They sense, compute and actuate into the physical environments. They can self-organize and adapt to support several applications. Each sensor node has wireless communication capability and sufficient intelligence for signal processing and for disseminating the data. The limited energy, computational power, and communication resources of a sensor node requires the use of a huge number of sensor nodes in a wider region. This large number also allows the sensor network to report with higher accuracy, exact speed, direction, size, and other characteristics of a moving object than is possible with a single sensor.

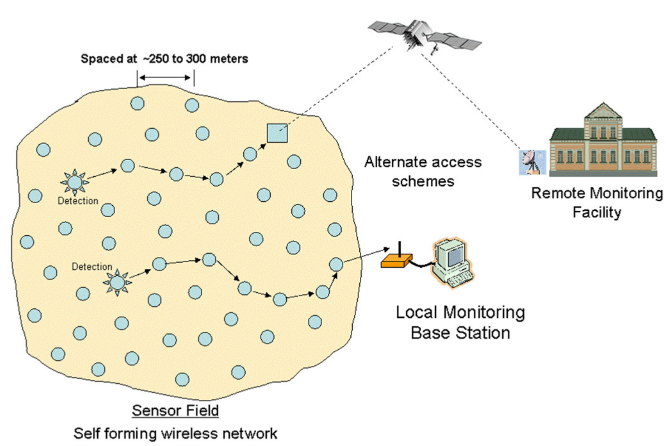


Figure 1.2: Wireless Sensor Network [2]

Technological advances in hardware and its economic availability have enabled them to be used in hundreds of different application scenarios today. In [47], the authors have classified their applications into 3 main categories:

- Monitoring space: Environmental and habitat monitoring, Precision agriculture, Indoor climate control, Surveillance, Treaty verification, Intelligent alarms, etc.
- Monitoring things: Structural monitoring, Eco physiology, Condition-based, Equipment maintenance, Medical diagnostics, Urban terrain mapping, etc.
- Monitoring the interaction of things with each other & the encompassing space: Monitoring complex interactions, Including wildlife habitats, Disaster management, Emergency response, Ubiquitous computing environments, Asset tracking, Healthcare, Manufacturing process flow, etc.

Considering the architectures of cellular network, MANETs, and sensor networks, next generation wireless networks have become a collection of heterogeneous devices, with multiple technologies, different capabilities, and numerous uses. In order to provide a comprehensive solution, which caters to all (or most) of the needs of such systems, new architectures are required. These architectures have to be well defined, and yet flexible enough to allow different techniques to be incorporated in it to solve different problems. In this research work we argue that the new architectures for wireless communication (Ad hoc and sensor) needs to be viewed from a complete applications point of view first, and then broken down to individual layers that are more integrated with each other, rather than solving problems individually and then combined to form a solution.

We propose a Hybrid Anycast Framework for ad hoc communication of wireless networks. In our work the term '*hybrid*' refers to the combined reactive and pro-active nature of routing algorithm. It shall not be confused with the heterogeneity of technologies or multihop access nature of communication. The proposed framework is divided into multiple sub-parts to individually address routing, Quality of Service (QoS), performance, and other tasks. Thus, it's not simply a routing protocol, but rather a complete framework which can use different techniques, to solve a number of problems. We also propose an Anycast Hybrid Routing Protocol that runs at the heart of the framework, and has the flexibility to incorporate multiple QoS techniques. We have modified and proposed a light-weight version of this framework for sensor networks. Generic routing protocols designed for wireless ad hoc networks fail in sensor networks primarily due to the fact that they are designed for more powerful nodes with higher transmission range and power as compared to sensors. In addition to this, the packet structure, routing table sizes, implemented code footprint, and many other states that are maintained, cannot be ported directly on to tiny sensors. We develop the sensor network framework in order to address these issues. A light-weight routing protocol with added functionality of the optimal sink selection using anycast approach is also part of the sensor network framework.

The implementation and performance evaluation of our proposed work has been done using simulation software. Simulation tools generically are not designed to implement complete application solutions, nor can then test the protocols in realistic environments. In the third stage of our work we have developed new models for NS-2 (simulation software) to mimic real time spreading of forest fires. We have tested all of our framework and protocol implementations on NS-2, with different network parameters. Finally we complete our work with an investigation of sink movement optimization. We evaluate the knowledge of topological and topographic information, and propose strategies in building such maps. These maps can then be used to find optimal data collection points for sink nodes.

The research work is described in detail in the following chapters: In Chapter 2, we describe a detailed literature review of the existing techniques and algorithms available, along with their advantages and drawbacks. Chapter 3 details the Hybrid Anycast Framework work, its challenges & motivations, and the Hybrid Anycast Routing Protocol for ad hoc networks. Chapter 4 describes the Hybrid Anycast Architecture for sensor networks. We discuss the motivation for designing a separate architecture for sensor networks and its benefits. The implementation of Light-Weight Hybrid Anycast Routing for Sensor Networks and its simulation results are discussed in chapter 5. In Chapter 6, we detail the challenges faced in simulating realistic scenarios, and the models we have developed for use within NS-2 to overcome this problem. We detail simulations and results with our Light-Weight Hybrid Anycast protocol for different forest fire simulation scenarios. Chapter 7 describes our investigation into path optimization for sink nodes in a sensor network. Chapter 8 concludes our research work, followed by references.

CHAPTER 2: LITERATURE REVIEW

2.1 Network Routing

Routing is a basic and important component for multihop networks, which aims to determine the path for traffic to travel from source to destination. There are several important issues in routing protocol development.

2.1.1 Routing Paradigms

Routing paradigms differ in their delivery semantics, which are mainly divided into four categories: unicast, broadcast, multicast, and anycast. Recently other extensions like manycast and geocast have also been proposed. A unicast address uniquely identifies a single receiver endpoint. A packet sent to a unicast address is received by only one interface which is currently associated to that address. Broadcast and multicast paradigms are similar, which enable a single source node to send a copy of packet to a set of receiver endpoints. The difference is that, for broadcast, each destination address identifies all other nodes in the network except the source, while multicast identifies a subset of other network nodes. Geocast [3] refers to the delivery of information to a group of destinations in a network identified by their geographical locations. It is a specialized form of multicast used by some routing protocols for MANETs, which may consist of geographic co-ordinates, center points of circular/polygon regions, etc. A GPS system or a location awareness service is required for such networks. Anycast delivers each transmission to at least one, and preferably only one member of the group, which is the "closest" to the sender as determined by the network layer. Manycast [4] is similar to multicast, but instead of selecting all the members of a group for communication, one client communicates simultaneously with some threshold number, *k* out of *m* total members of a group.

2.1.2 Single path vs. Multipath Routing

Single path routing assigns a single path to all traffic between a given pair of source and destination nodes. Although single path routing may simplify the routing tables and the packet flow paths, yet it cannot provide fault tolerant service. On the contrary, multipath routing distributes traffic for a given source and destination pair over several paths. Multipath routing has a potential to aggregate bandwidth on various paths, allowing a network to support data transfer rates higher than what is possible with any single path. Furthermore, multipath routing can provide fault tolerant service and balance the traffic load of the network across multiple paths.

2.1.3 Conventional protocols

If a routing protocol is needed, it can be argued why conventional routing protocol like link-state or distance vector can't be used? Especially when they are well tested and most computer communication is done using them. This argument falls apart when we take into account the main problem with link-state and distance vector i.e. they are designed for a static topology, which means that they would have problems to converge to a steady state in an ad-hoc network with a frequently changing topology.

Link state and distance vector would probably work very well in an ad-hoc network with low mobility, i.e. a network where the topology is not changing very often. The problem that still remains is that, link-state and distance vector are highly dependent on periodic control messages. As the number of network nodes can be large, the potential number of destinations is also large. This requires large and frequent exchange of data among network nodes. Since all updates in a wireless interconnected ad hoc network are transmitted over the air, it results in high resource consumption, such as bandwidth, battery power and CPU. Moreover, as both link-state and distance vector try to maintain routes to all reachable destinations, which, also wastes resources for the same reason as above.

Another characteristic of the conventional protocols, is that they assume bi-directional links, e.g. the transmission between two hosts works equally well in both directions. However, this is not always guaranteed in the wireless radio environment.

Because many of the proposed ad-hoc routing protocols have a traditional routing protocol as underlying algorithm, it is necessary to understand the basic operation of conventional protocols like distance vector, link-state and source routing.

Link State: In link-state routing, each node maintains a view if the complete topology with a cost for each link. To keep these consistent; each node periodically broadcasts the link cost of its outgoing links to all other nodes using flooding. As each node receives this information, it updates its view of the network and applies a shortest path algorithm to choose the next-hop for each destination. Some link costs in a node view can be incorrect because of long propagation delays, partitioned networks, etc. Such inconsistent network topology can lead to formation of routing-loops. These loops are however short-lived. Because they disappear in time it takes a message to traverse the diameter of the network.

Distance Vector: In distance vector [5], each node only maintains the cost of its outgoing links, but instead of broadcasting this information to all nodes, it periodically broadcasts to each of its neighbors an estimate of the shortest distance to every other node in the network. The receiving nodes then use this information to recalculate the routing tables, by using a shortest path algorithm. Compared to link-state, distance vector is more computation efficient, easier to implement and requires much less storage space. However, it is well known that distance vector can cause the formation of both short-lived and long-lived routing loops. The primary cause of this is that the nodes choose their next-hops in a completely distributed manner based on information that can be stale.

Source Routing: Source routing means that each packet must carry the complete path that the packet should take through the network. The routing decision is therefore made at the source. The advantage with this approach is that it is very easy to avoid routing loops. The disadvantage is that each packet requires a slight overhead.

Flooding: Many routing protocols use broadcast to distribute control information, that is, send the control information from an originating node to all other nodes. A widely used form of broadcasting is flooding and operates as follows. The originating node sends its information to its neighbors (in wireless case, this means all nodes that are within transmitter range). The neighbors relay it to their neighbors and so on, until the packet has reached all nodes in the network. A node will only relay a packet once, and to ensure this some sort of sequence number can be used. This sequence number is increased for each new packet a node sends.

2.2 Routing in MANETS

In mobile ad-hoc networks, because of the fact that it may be necessary to traverse several hops (multihop) before a packet reaches the destination, a routing protocol is needed. The routing protocol has two main functions, selection of routes for various source-destination pairs and the delivery of message to their correct destination. The second function is conceptually straightforward using a variety of protocols and data structures (using tables).

2.2.1 Taxonomy

MANET routing protocols may be classified according to different criteria, reflecting fundamental design and implementation choices as shown in Figure 2.1.

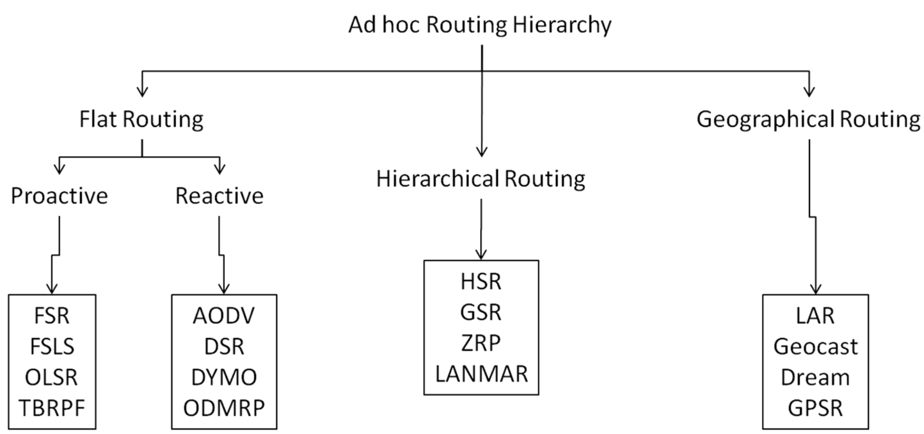


Figure 2.1: MANET routing protocol hierarchy

2.2.1.1 Communication Model

The basic difference lies in the underlying wireless communication model, which separates the protocol designed for multi-channel and single channel communication. Multi-channel protocols are low-level protocols, which combine channel assignment and routing functionality. TDMA and CDMA-based network generally use such protocols. Some MANET routing protocols are based on more specific link-layer properties such as

RTS/CTS control sequence used by the IEEE 802.11 MAC layers to avoid collision due to hidden and exposed terminals.

2.2.1.2 Structure

Routing protocols may be categorized as uniform or non-uniform protocols.

Uniform protocols: In such protocols, none of the nodes take any distinguished role. Each node sends and responds to the control messages in the same manner. Also, no hierarchical structure is imposed on the network. Scalability becomes an issue in such protocols, while the cost of the overall network is reduced.

Non-uniform protocols: Non-uniform protocols attempt to limit routing complexity by reducing the number of nodes participating in a route computation. Such an approach increases the scalability and reduces the communication overhead. In addition, higher-level topology information can facilitate load balancing and QoS support. Non-uniform protocols fall into two categories: protocols in which each node focuses on routing activity on a subset of its neighbors and protocols in which the network is topologically partitioned.

2.2.1.3 State Information

Protocols may be described in terms of state information obtained at each node and/or exchanged among nodes.

Topology-based protocols: Nodes participating in topology-based protocols maintain large-scale topology information. The best known such protocol is “Link-state” protocol. Such protocols are effective for routing in fixed Internet, but the amount of data and the frequency with which it must be distributed throughout the network are a significant disadvantage in the resource-constrained, highly dynamic MANET environment. Large-scale topology information can also be used not only for basic routing functionality, but also improve route selection, load balancing and QoS management.

Destination-based protocols: Such protocols do not maintain large-scale topology information, although some maintain local topology information (e.g. 1 or 2-hop neighbor). The best known such protocol is “distance-vector” protocol. Several proposed protocols adapt the distance vector approach for operation in mobile ad-hoc networks. Techniques include the use of sequence numbers and next-to-last-hop in the distance-vector information to ensure freedom from long-lived routing loops.

Other destination-based protocols entirely avoid the exchange of distance information. Nodes only maintain distance vector routing information for “active” destinations – those to which they are sending or forwarding traffic.

2.2.1.4 Scheduling

Finally, protocols can be considered in terms of when a source obtains route information as it initiates traffic flow to a destination.

Proactive protocols: In pro-active or table-driven routing protocols, each node maintains one or more tables containing routing information to every other node in the network. All nodes update these tables so as to maintain a consistent and up-to-date view of the network. When the network topology changes, the nodes propagate update messages throughout the network in order to maintain consistent and up-to-date routing information about the whole network. These routing protocols differ in the method by which the topology change information is distributed across the network and the number of necessary routing-related tables.

Reactive protocols: These protocols take a lazy approach to routing. In contrast to proactive routing protocols, all up-to-date routes are not maintained at every node, instead, the routes are created as required. When a source wants to send to a destination, it invokes the route discovery mechanisms to find the path to the destination. The route remains valid till the destination is reachable or until the route is no longer needed.

2.2.2 Routing Protocols

In earlier research, classical routing methods were adopted for MANET. Destination-Sequenced Distance-Vector Routing (DSDV) [8] made modifications to RIP [7] and used the distance vector approach; while Optimized Link State Routing (OLSR) [9] adapted the link state algorithm. Both DSDV and OLSR are proactive routing protocols, which maintain routing information between each pair of nodes all the time and perform periodical updates even though they do not have to communicate with each other. However, reactive approaches only initialize route discovery when necessary. Ad Hoc On-Demand Distance Vector Routing (AODV) [10], Dynamic Source Routing (DSR) [6], Lightweight Mobile Routing [11], Temporally Ordered Routing Algorithm (TORA) [12] are all reactive protocols. As a combination, hybrid mechanisms are based on the idea of organizing nodes in groups and then assigning nodes different functionalities inside and outside a group, usually, applying proactive routing inside the group, while reactive approach among groups, for example, Hierarchical State Routing (HSR) [13], Cluster head-Gateway Switch Routing (CGSR) [14], Zone Routing Protocol (ZRP) [15] and Landmark Ad Hoc Routing protocol (LANMAR) [16] are four examples in this category. [17] first reviewed MANET routing protocols and classified them into two groups according to the time when routing activities are initiated; while [18] extended the work of [17] by adding more protocols and provided a classification according to the network structure underlying these routing protocols. The three main categories found in literature are: flat routing, hierarchical routing and geographic position assisted routing.

2.2.2.1 Broadcast Protocols

Broadcasting has wide application in MANETs not only for data dissemination, but also for route discovery and maintenance in many ad hoc unicast routing protocols. Blind flooding which is the simplest broadcast protocol, allows each node to forward the message once to its neighbors. However, due to omnidirectional radio propagation and transmission range overlap, blind flooding is usually very costly and will result in serious redundancy, contention, and collision, which is referred as the broadcast storm problem in [19].

In order to reduce redundant transmission, different broadcast protocols have been developed in recent research work. [19, 20] applied probabilistic approaches. [21–25] used deterministic approaches, which select certain number of forwarding nodes based on topology information to achieve full delivery. With the development of directional antenna, which can control radiation pattern to reduce broadcast redundancy, several protocols [26–28] have been proposed for efficient broadcasting.

To achieve energy conservation, minimum energy broadcast routing protocols were also developed in recent research, which allow nodes to adjust their transmission power in order to minimize total energy consumption but still guarantee the broadcast message to reach all other nodes in the network. Such algorithms can be broadly classified into globalized approaches [29] and localized approaches [30–32].

2.2.2.2 Multicast Protocols

Multicasting is a very efficient and useful communication paradigm supporting group-oriented applications, especially for MANET, which has constrained bandwidth and energy resource. There has been extensive work to develop multicast protocols for MANETs to provide one-to-many service, e.g., [33–42]

As a variant of conventional multicasting, geocasting delivers message to nodes within a geographical region, which can provide new services and applications, e.g., geographic advertising. A number of geocast protocols have been developed, and [43] classified them into two categories based on whether flooding or a variant of flooding is used to forward data from source to the geocast region. Protocols proposed in [44–47] are examples for routing with flooding approach and [48] is an example of routing without flooding approaches.

2.2.2.3 Anycast Protocols

All the servers in an anycast group share a single anycast address. Similar to a unicast flow, the client and server are unaware that anycasting is being used. The server that receives a specific routing packet is determined by the unicast routing protocol used in that domain. No anycast routing table is maintained equivalent to multicast routing tables. Servers are configured with the same anycast address, and are located at different locations in the network. The routing protocol automatically delivers packets from the client to the closest destination with the anycast address. Regardless of its limited global deployment, anycasting is an attractive technique, as it reduces the use of router and link resources, simplifies configuration, and provides resource redundancy along with load distribution & reduced latency [33].

Anycasting has been extended from the Network Layer to the Application Layer in recent years, where server selection is based on application metrics, such as capacity, measured response times, number of active connections, processor loads, etc. [34] and to the MAC layer where anycast decisions are made on a per hop basis [35]. The core architecture is based on a query-response system. The query contains the selection criteria for choosing a server from the group. The response thus contains the IP address of the selected server. In some architectures, a Server Push strategy is also used, in which the server tells its state to a single entity known as a "Resolver", which alone handles the client queries. Further extension to Application Layer Anycasting is the use of Network layer feedback for selecting the optimal path for communication.

A number of anycast routing protocols have been deployed in wired networks. In conventional link state, distance vector, and link reversal routing algorithms for wired networks, a concept of virtual node [36] has been introduced, which represents the anycast service availability. These virtual nodes are connected to the real nodes which are members of an anycast group through virtual links. While building the connectivity graphs, these virtual nodes act as destinations for anycast group members.

The main challenge still lies in using anycasting for robustness, efficiency, and resource sharing in ad hoc networks. Some recent work on anycasting in ad hoc/sensor networks can be found in [37-43]. In addition to other new anycast routing protocols for ad hoc/sensor networks, enhancements have been proposed to unicast reactive ad hoc routing protocols, e.g., in [44,45]. The fundamental benefit these protocols provide is their reactive route discovery nature, which gives the flexibility to locate an anycast group member when required. In [41] anycast routing is supported by introducing a 4-bit

Anycast Group ID. The Route Request message is modified to contain this ID along with other flags for discovery of the nearest anycast service provider. This protocol is designed to work purely in ad hoc environments for evenly distributing the load on multiple available anycast server nodes. The results show that there is 10% and 15% improvement in packet delivery fraction with 30 and 40 sources respectively at high mobility. And with almost-zero mobility, the performance is increased by 50% and 40% respectively. At lower number of sources (10-20), performance is almost the same as unicast AODV. Results also show a decrease in average end-to-end delay and normalized routing load (routing packets per data packet). In addition to the Anycast Group ID, a route request (RREQ) packet contains a list of anycast server IPs and their last known sequence numbers. This increases the size of RREQ packets. Information about nodes joining or leaving an anycast group is broadcasted in the network, but a new node in the network will not have this information.

In [42], a modification of Dynamic Source Routing Protocol is proposed, which is a similar idea as in [41] for anycast ID or Anycast address. A RREQ packet does not contain a list of anycast servers, which reduces the packet size to some extent. But the source routing mechanism introduces its own overhead. The results show that performance improves by 30% and 35% with 30 and 40 sources respectively.

In [46] authors compare the basic AODV and DSR routing mechanisms, and the results show that AODV in high mobility, high data rates, and large network scenarios performs better than DSR. Anycast routing protocols based on AODV and DSR routing are expected to follow the same pattern.

2.3 Routing in Sensor Networks

Despite the advancement in technology, a number of design factors need to be considered before deploying a sensor network. Some of these are [47]; fault tolerance, scalability of network, node density, power consumption, transmission media, communication technology, etc. My emphasis is on the communication and routing of information inside the network. Communication protocols for sensor networks are themselves challenged by many influencing factors. A summary of these factors is given below [48]:

Node Deployment: Node deployment can be random or pre-planned. This affects the path establishment challenges as unknown topology warrants dynamic path building and optimal cluster formation for prolonged connectivity.

Energy Consumption: As the sensor nodes have limited energy capacity, computation and transmission of data should be kept to minimum possible levels, without sacrificing the accuracy and timeliness. Failure of nodes due to loss of energy will trigger topological changes that will require more complex route maintenance algorithms.

Data Reporting Model: Data can be collected in a number of ways depending on the application of sensor network. It can be time-drive, event-driven, query-based, or a hybrid of these. The working of routing protocol is highly influenced by the data reporting model used.

Node Heterogeneity: Depending on the application, the nodes may have different capabilities in a sensor network. Some may have more computing power than others, while some may have more energy available, and some may even be mobile. Furthermore the availability of multiple sensors on the same device complicates how nodes are deployed. To fully optimize the operation of such a heterogeneous set of nodes, complex communication algorithms are needed.

Fault Tolerance: Nodes in a network may fail due to any number of reasons: power failure, physical damage, transmission blockages, etc. The routing protocol must be capable enough to overcome these link failures, by incorporating redundant paths, self-healing, varying energy consumption levels, etc.

Scalability: A sensor network deployed to monitor temperature variations in a forest (to detect forest fires) would cover hundreds of square miles. This means, required number of nodes may be in thousands. Also the number of events triggering multiple responses can be extraordinarily large in such cases. The routing protocol should be scalable enough to work with such a huge network.

Node Mobility: Depending on the application, the nodes may be static or mobile. Mobile nodes will result in link breakages, requiring an efficient self-healing mechanism.

Transmission Media: Nodes in sensor networks are generally low powered devices; as a result, the transmission rates may be very low. Also the generic wireless medium problems affect the communication.

Connectivity: Limited transmission ranges and node density highly affect the connectivity of a network. It is possible that, some parts of the network have many redundant links, but others make a sparse network. Maintaining a high level of connectivity requires clever techniques.

Coverage: In sensor networks, each sensor node obtains a certain view of the environment. A given sensor's view of the environment is limited both in range and in accuracy; it can only cover a limited physical area of the environment. Hence, area coverage is also an important design parameter in wireless sensor networks.

Data Aggregation: Data aggregation is the combination of data from different sources. The data collected by sensor nodes may contain duplicate information, which will consume the network resources if transmitted without any aggregation function. Communication protocol must employ efficient aggregation techniques to maximize the data accuracy and minimize data redundancy.

Quality of Service: Based on application environment, some data may need better QoS. For example, certain environments may be more time critical than the other. This brings a great challenge in classifying the traffic generated by the sensor nodes.

2.3.1 Routing Protocols

In recent years, dozens of routing protocols for wireless sensor networks have been proposed. Broadly these protocols can be classified into two categories depending on their Network structure and Protocol Operation. Figure 2.2 gives a summary of this classification. Protocols can be divided into Flat Network Routing, Hierarchical Routing, & Location based Routing, based on the network structure. The same protocols can be divided into Negotiation Routing, Multi-path Routing, Query-based Routing, QoS-based Routing, & Coherent Routing based on Protocol Operation. A brief description of each category is given below.

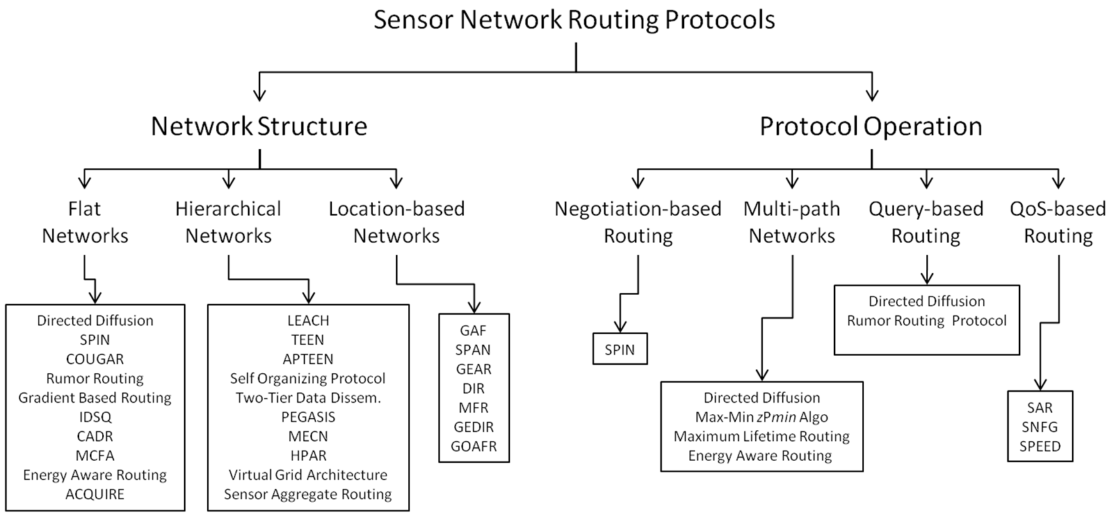


Figure 2.2: Sensor Network routing protocol hierarchy

2.3.1.1 Network Structure Based Protocols

The underlying structure of the network plays an important role how the routing is performed. The nodes can be logically organized into different patterns and groups in order to achieve optimal performance. Protocols under this classification can be further divided into these categories.

Flat Routing: In flat routing all nodes play the same role of sensing and forwarding the data. This is a data centric approach, where the sink queries nodes to gather information. Each node has a global identifier which is usually preconfigured. Flat routing protocols tend to achieve more optimal routes as compared to Hierarchical routing described in next section. Examples of such protocols are; Sensor Protocols for Information via Negotiation (SPIN) [50-51], Directed Diffusion [49], Rumor Routing [52], Minimum Cost Forwarding Algorithm (MCFA) [53], Gradient Based Routing [54], Information-driven Sensor Querying (IDSQ) [55], Constrained Anisotropic Diffusion Routing (CADR) [55], COUGAR [56], ACQUIRE [58], Energy Aware Routing [57], Routing protocols with Random Walks, etc.

Hierarchical Routing: This is a cluster-based routing mechanism, where a set of nodes is elected (or preconfigured) to act as cluster heads. All nodes inside a cluster forward their data to cluster head, which is responsible for sending it to the sink. Usually the cluster heads form a backbone in the network, thus reducing the involvement of other (low powered) nodes in the forwarding mechanism. The cluster heads, based on their capabilities, may incorporate data aggregation techniques to reduce the redundant data from flowing through the network. Examples of such protocols include; Low Energy Adaptive Clustering Hierarchy (LEACH) Protocol [59], Power-Efficient Gathering in Sensor Information Systems (PEGASIS) [63], Threshold-sensitivity Energy Efficient Protocol (Adaptive) [60-61], Small Minimum Energy Communication Network (MECN) [65], Self Organizing Protocol (SOP) [62], Sensor Aggregates Routing [68], Virtual Grid Architecture (VGA) Routing [67], Hierarchical Power Aware Routing (HPAR) [66], Two-Tier Data Dissemination (TTDD) [64], etc.

Location Based Routing Protocols: These protocols rely on Global Positioning System (GPS) or similar mechanisms to accurately determine the location of each node. Thus the nodes are addressed based on their physical locations. Routes are built using location information of next hop towards the destination. Examples of such protocols are: Geographic Adaptive Fidelity (GAF) [69], Geographic and Energy Aware Routing (GEAR) [71], Most Forward within Radius (MFR), DIR, Geographic Distance Routing (GEDIR) [72], Greedy Other Adaptive Face Routing (GOAFR) [73], SPAN [70], etc.

2.3.1.2 Protocol Operation Based Routing

This classification is based on the routing technique used for establishing paths. Protocols under this classification can be further divided into the following categories.

Multipath Routing Protocols: These protocols maintain more than one path between the node and the sink. The primary purpose of this is to provide fault tolerance to link breakages and topological changes. Protocols that employ such a scheme include; Directed Diffusion, Max-Min zPmin Algorithm [74], , Maximum Lifetime Routing [75], Energy Aware Routing for Low Energy Ad Hoc Sensor Networks [76], etc.

Query Based Routing: This type of protocol technique is strictly based on queries by the sink node. Sensor nodes do not sense or do not report data on their own to the sink. When information is required a path is established via query message and information is retrieved. Examples include; Rumor Routing Protocol [77], Directed Diffusion [49], etc.

Negotiation Based Routing Protocols: The basic approach of these protocols is to eliminate transmission of redundant information. Protocols similar to SPIN [50-51] are included in this category.

QoS based Routing: These routing algorithms satisfy certain metrics before they deliver the data to the destination nodes. These metrics could be delay, energy consumption, bandwidth, transmission power, error rate, etc. Sequential Assignment Routing (SAR) [78], Stateless Geographic Non-Deterministic Forwarding (SNFG)/SPEED [79], are some examples of QoS routing protocols for sensor networks.

2.3.2 Data Collection using Mobile Sink

Protocol in [99] describes construction of multicast trees. Sensor nodes and sinks are joined through branches and proxy nodes. This scheme trades off overhead in tree maintenance against possible longer paths due to extension on multicast tree by unicast branches. Due to the complexity of the application scenario, longer paths may not be desirable. Thus using a technique with such handicap is not the optimal solution. [100]

has also discussed mobile sinks and proposes a permanent resolver node in the network. Extreme conditions under different applications may not work well with a dedicated resolver. If the resolver is destroyed or runs out of battery power, the whole network will be dead till a replacement resolver is made available. [101-102] considers usage of MULEs as data gathers, and argues that they are considerably more energy efficient than multihop communication and can increase lifetime of the network without impeding data collection quality. The interesting question pertaining the MULEs is the interdependence between the movement patterns of the MULEs, time between visits to sensor nodes, data collection rates between sensors, buffer space at sensors and MULEs, communication speed between MULE and sensor, and the resulting delay and data delivery rate at the actual data sink.

2.4 Network Routing Simulation Systems

Routing protocol performance measurements is a very complex and costly process. This can be achieved by either deploying a complete test bed system, or by using simulation software that can mimic the communication networks.

Deploying a real network or even a test bed for performance evaluation is a costly and complex process, and requires resources which are not usually available to all. More over the cost of building such a system may not be justified by the research problem and its evaluation. Thus most of the research is initially tested using simulation software, before it can be optimized and tested on real networks. There are numerous network simulation software available with varying capabilities. Network Simulator (NS-2, NS-3), GloMoSim, QualNet, Scalable Simulation Network (SSF), JavaDim (J-Sim), OMNET++, OpNet are some commonly used examples. Majority of these simulation systems have been developed for research work, but the commercial software like OpNet and QualNet are also used to design larger communication systems.

The biggest challenge while using simulation software for evaluation is to answer the question: How close the simulation is to real world? This question is very important to answer before the use of any evaluation process, as it determines if the evaluation process holds any merit in the real world. Given the magnitude and vastness of operations in communications, it is possible to evaluate different modules independent of each other. Most of the simulation systems take benefit of this, and are designed to simulate only parts of the overall network. This works very well and has been used extensively in research as well as in industry.

2.4.1 Network Simulator (NS-2)

NS-2 (Network Simulator, version 2) is an open source, discrete event simulation system. NS-2 has been widely used in the research community to test communication protocols for both wired and wireless communication. Mainly it allows implementation of protocols at physical, data link, MAC, Network and Transport layers of OSI model, although implementations of application layer FTP, HTTP, and other data generation protocols are available. It uses multiple languages to simulate networks. C/C++ is used to provide the core implementation of node structures, simulation control, and protocol implementation. Tcl and OTcl (Object Tool Command Language) are used to design the network and set up the simulation scenarios. An interpreter is used as in interface between these two sides of the simulator.

Initially NS-2 was designed to simulate wired networks, but mobility extensions were provided by CMU's Monarch project, which have now been extensively improved by other research groups. It includes 802.11 MAC implementation using Distributed Coordination Function (DCF), MAC functionality modules (transmission, reception, transmission coordination, reception coordination, back-off manager, and channel state monitor), Cumulative SINR computation, MAC frame capture capabilities, Multiple modulation scheme support, Packet drop tracing at the PHY layer, Nakagami fading model, Multi rate modules (802.11 b/g standard), etc. Ad hoc routing protocol implementations have been provided by many researchers. Most notable is the AODV implementation provided by Uppsala University, which complies with the ADOV RFC. We have used this implementation as a benchmark to evaluate the performance of our protocols also.

Once the protocol and simulation scenarios are implemented in NS-2, the simulation system generates a trace file, which contains all events that occurred during the simulation process. Each transmission of data/control packet by each layer of each node is recorded along with its contents and is time stamped. Visualization tools like Network Animator (NAM) can be used to see a graphical output of network traffic. Usually to obtain useful data, specific parsing programs can be written to extract and calculate the results of the simulation. In our simulations we have used special parsers written in C++ to extract useful data from the trace files.

2.4.2 Limitations of Simulation Systems

As discussed earlier, simulation systems suffer the biggest limitation by not being able to provide a complete solution to all the network simulation scenarios. In this section we discuss two major limitations that impact basic research and protocol evaluation.

Consistency across Simulators: Consistency of simulation results across different simulation software is a major concern in conducting research on networking protocols. This is not limited to any particular layer in the OSI model but rather affects all types of protocols implemented. Camp [88] has discussed this issue in great detail. The main cause of this problem is the way simulators are implemented, their device and node architectures, protocol implementation architectures, and actual simulation/emulation procedures. It has been shown in different studies that implementation of same protocol on different simulators, and evaluations done using similar network parameters, produce results that are barely comparable. This produces a significant challenge to researchers while benchmarking the performance of protocols. This problem has also affected our research. To simplify this problem we have used AODV implementations done according to the RFC, and used that as a benchmark for performance evaluation of our proposed solutions.

Simulation of Realistic Scenarios: This is another major issue that arises when the simulation and evaluation of protocols crosses the boundaries of network layers. In our research this problem has led us to the development of new models to simulate node errors. We have used NS-2 as the simulation tool. Inherently NS-2 is not designed to simulate networks with realistic events. The basic objective of NS-2 is to simulate the basic node behavior, evaluate protocol operation, and provide other related services. The major benefit of using NS-2 is that, it is open source and can accommodate new models. In order to simulate a sensor network in a forest, to detect forest fires, NS-2 provides the physical model of nodes, communication structures, and flexibility to implement the routing protocol. But it does not have the capability to model the actual spread of fire.

Similarly in a battle field scenario, it is not possible with NS-2 to simulate the movement of soldiers, creation of dead zones, and other realistic situations which will test the performance of routing protocols in a more realistic environment.

CHAPTER 3: HYBRID ANYCAST ROUTING

3.1 Future Network Architecture

The vision of Future Generation Networks is evolving towards one that includes interoperable heterogeneous wireless access technologies to provide seamless access to core networks. A heterogeneous wireless access network is composed of various devices with single or multiple interfaces as well as the capability to relay traffic between interfaces. Such a heterogeneous network architecture as shown in Figure 3.1. The network architecture tightly integrates heterogeneous wireless multihop networks, wireless access networks, and the core networks. In wireless multihop region, mobile nodes (MNs) are connected with each other via various interfaces, e.g., a laptop can communicate with a cell phone via 802.11 interface or 3G interface. MNs can forward traffic to an AP or to other MNs, serving as either a source or a relay. In wireless access network region, a number of APs which may use different access technologies (e.g.,

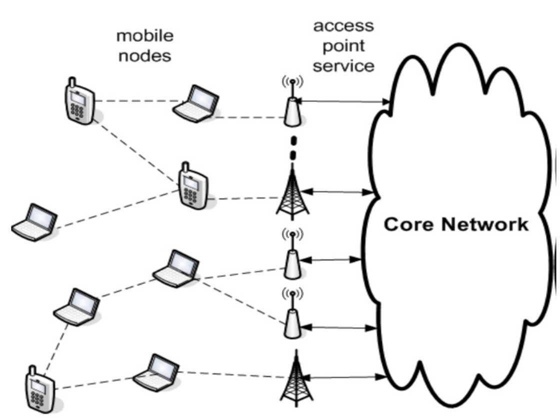


Figure 3.1: Heterogeneous Access Network

Cellular, WiMax, Wifi), these APs connect MNs to the core network and terminate the wireless portion of the network. This research work is focused on the wireless side of this architecture and assumes existing protocols or system designs are available to integrate heterogeneous access technologies into the core network.

3.2 Challenges and Motivation

As the goal is to provide a “globally” optimum solution that meets application’s requirements and optimize network utility, the proposed network architecture introduces a significantly new set of design challenges for diversified resource management, protocol development, etc.

3.2.1 Path Cost Calculation

The criteria used to represent path cost guides path selection and resource consumption in the network. Instead of only considering the hop count in the route selection as classical routing protocol, multiple cost metrics might be required from application’s point of view, e.g. delay, bandwidth, etc. However due to device diversity, mechanism assuming that all network devices are identical, does not fit well in such networks. The calculation of path cost is a critical component of route discovery and also a challenging task for the protocol development, which has to satisfy application’s requirement and can adaptively adjust to device diversity.

3.2.2 Path Acquisition and Maintenance

The path acquisition is different from multihop routing, such as unicast routing protocols for ad hoc networks, due to the following reasons:

Multiple Choices of Destinations: Rather than having one specified destination, the mobiles attempt to locate the best access point (AP) for connection to the Internet, where the notion of best can be described as 'most willing' or 'optimum for communication' based on some selection criteria. While finding a route from a mobile to "one or more of a group" of APs is better modeled by an anycast or manycast communications paradigm, rather than unicast or multicast. However, anycasting solutions have not been explored sufficiently in the context of heterogeneous wireless networks, though several anycast routing protocols [81–86] have been proposed in both web services and ad hoc and sensor networks. Most of the anycasting protocols for ad hoc network are simply modified from existing ad hoc unicast routing methods or are based on tailored multicast trees. For heterogeneous wireless access networks, new anycast paradigm needs to be explored.

Fixed Destinations: The unique feature of heterogeneous wireless network routing, as compared to ad hoc routing, is that APs serve as "destinations" for the wireless part. These destinations are fixed, and are limited. This calls for a fresh look at the tradeoffs in using proactive or reactive routing policies. In a proactive approach, mobile users and APs participate in periodic updating of link status. Hence, a user can make a route selection as needed, based on available information. In contrast, a reactive routing protocol would perform route discoveries to the set of selected APs to obtain the information required for route selection. As in ad hoc networks, the tradeoff between the two would be between latency and routing overhead. Although a hybrid solution which combines proactive and reactive can improve the performance, it is a challenging task to find the optimal solution for the division of the proactive and reactive region.

3.2.3 Diversified Network Resource Management

Radio resource management has been well addressed for Cellular network or WLANs by applying centralized algorithms running at a BS or AP to manage a single pool of wireless capacity. However, for heterogeneous wireless access network architecture, the cost of providing services to users vary from one AP to another, as determined by a complex combination of issues including available bandwidth, channel capacity, service availability, etc. Moreover, the multihop relay service provided by other users, will consume certain amount of resource from those users. Therefore, resource management should include radio resource as well as constrained resource at MNs. Various constrained resource are included in heterogeneous wireless access network, for example, low bandwidth of links connecting two MNs, constrained battery capacity on MNs, limited buffer space, CPU processing capability, memory size, etc. As consumption of such resource can greatly impact system performance, it is important to design and apply proper resource management mechanisms.

3.3 Hybrid Anycast Communication Framework

In order to address challenges described in the above section, we proposed a framework as shown in Figure 3.2. The framework includes three main modules, which can interact with each other.

3.3.1 Application Requirements

This is the input for the framework, which includes resource and cost metrics for guidance of route discovery. Bandwidth, battery capacity, and buffer space are examples of resource metrics, while delay, transmission power, link quality or stability, can be used as cost metrics. Applications may clarify requirements in different ways:

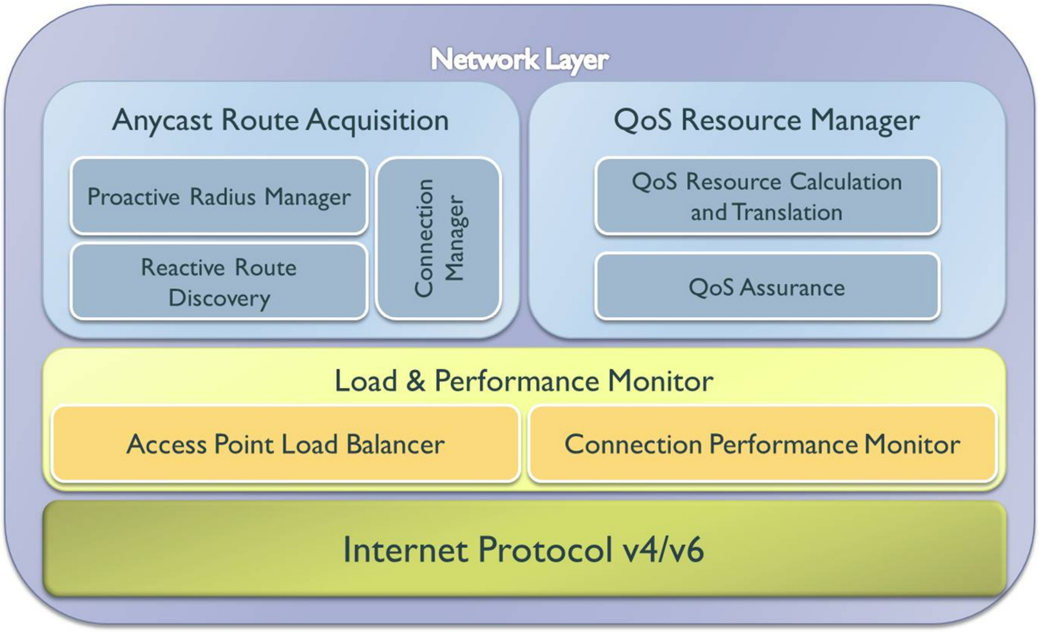


Figure 3.2: Architecture of framework

- Minimize certain total cost metrics along the path; e.g., discover the path with minimum end to end delay.
- Control route discovery by constraining certain resource or cost metrics; e.g., discover the path with maximum tolerable delay *d*.

Therefore, user requirements can specify type(s) of metrics, and how metrics will be used to guide the route discovery. Furthermore, user requirements can also define the degree of importance for each metric when there are multiple metrics. The objective is to locate the best AP among a group as well as discover the minimum cost path to that AP based on the requirements of application. Therefore, we classify the application requirements into two categories:

- Requirements for AP Selection: Different applications might have different requirements for certain types of resource at APs. Due to device or technology diversity, different APs might have different levels of capability. During the route discovery process, AP selection must satisfy application requirements including insuring that the selected AP has enough resources for the application.

- Requirements for Path Selection: As the multihop relay service is provided by other MNs in the networks, resources are usually constrained. As availability of such resource can greatly affect the performance of the connection, applications require the route discovery to guarantee the sufficiency of certain resources. Besides the availability of resources to guarantee the communications, applications might also require to minimize certain cost metrics in order to maximize the performance. There might be multiple APs that can satisfy the requirements for AP selection, and even with only one AP, there might be multiple paths available between source and the AP(s). Based on the multiple-metric path cost specified by the application requirement, paths with the minimum cost value will be selected as the best route.

3.3.2 QoS Resource Management Module

Each node has a QoS Resource Management Module running on it, which can be divided into two sub-modules: QoS Resource Calculation and Translation, and QoS Assurance.

- QoS Resource Calculation and Translation: This sub-module will maintain accurate information for available network resources, both locally and globally. As the format of application requirements is expressed in a general way, such as "delay sensitive", the QoS resource calculation and translation module is responsible for translating application requirements into a format that can be directly used by the routing module. Meanwhile, this sub-module calculates and estimates the amount of other related resource parameters based on the traffic information, such as required throughput and bandwidth for delay sensitive applications.

- QoS Assurance: Once the communication is setup, the user enters into a “contract” with the AP and with the MNs along the route, expressing its desirable requirements and price. However, due to variations in the multihop environment and other factors, the available resources can also change. This module monitors certain cost metrics, such as link stability or quality, delay, or BER, for functions to meet the user requirement for the “contract”: when certain resource or cost constraints specified by user requirements cannot be guaranteed, the routing protocol should be notified.

3.3.3 Anycast Route Acquisition Module

Three sub-modules are contained in this module, and they are: Proactive Radius Manager, Reactive Route Discovery and Route Connection Manager.

- *Proactive Radius Control*: This is an interface required for network administrators to set the borderline between the above proactive and reactive regions. Either static or dynamic approaches can be applied according to the requirements of the network administrator.
- *Reactive Route Discovery*: This sub-module aims to select an AP and discover a path for the source to connect to the selected AP, both of which should satisfy requirements defined by the QoS Resource Management Module. Thus, interactions between Routing and the QoS Resource Management module are required.

- *Connection Manager*: This sub-module keeps track of the state of the current route. Once a broken link is detected, it will notify the source node. The source node can reinitiate the route discovery procedure and set up a new route for the connection.

3.3.4 Load and Performance Monitor Module

Load and Performance Monitoring is done using AP Load Balancing, and Overall Connection Performance monitoring sub modules. These sub modules are described in the following sections.

- *AP Load Balancer*: In order to balance the traffic load among the APs in the network, this sub-module is designed to use general load metrics for AP selection, which can be the number of connections, number of packets, or other complicated load metrics.
- *Connection Performance Monitor*: This sub-module provides an interface for the network administrator to keep track of the operation of the whole network by calculating certain performance metrics. For example, it can provide feedback on the average remaining battery capacity or the variance of the remaining battery capacity to the administrator. Therefore, this sub-module is designed to facilitate the performance monitoring method of the network and individual nodes, and it does not interfere with the functionality of the protocol.

3.4 Hybrid Anycast Routing Protocol

There has been extensive research on protocol design for MANETs to discover multihop paths between source and destination pairs. However, these protocols do not appropriately fit heterogeneous wireless access network due to its unique features. First, rather than having one specified destination, MNs attempt to locate the best AP among multiple available ones based on certain selection criteria. Second, instead of having communication between MNs and MNs, APs serve as the "destinations" for the wireless part, which are fixed and limited. To meet the above special features of heterogeneous wireless access networks, this chapter illustrates our design of a hybrid anycast protocol which can support any type of cost metric for path selection. Our hybrid mechanism divides the multihop portion of the access network into two regions. The proactive region enables APs to advertise their services by maintaining state information at mobiles or relays within close proximity of the APs. The reactive region enables mobiles to discover APs, as needed, by interrogating nodes in the proactive region. A combination of proactive and reactive routing reduces communication overhead and delay, while increasing throughput.

3.4.1 Protocol Functionality

This hybrid anycast routing protocol is primarily based on AODV architecture; with major modifications to support anycasting and distributed regions. The routing protocol combines both proactive and reactive mechanisms. The network is divided into two regions:

- *Proactive Region*: APs and MNs within an *m* hop radius of an AP are in the proactive region. All MNs maintain active information about AP in this region through periodic Hello packets sent by AP. Hereafter, we call *m* as proactive radius.

- *Reactive Region*: All MNs more than *m* hops away from an AP are part of the reactive region, and use a reactive anycast routing protocol to discover routes to an AP.

The objective of our hybrid anycast protocol is for a mobile node to establish communication with an AP in an anycast group so that the selected AP can forward packets to the destination in the core network. The route to the destination for all data packets is selected through any of the APs based on a decision metric. APs are entry points for MNs to access the Internet and are part of one or more anycast group(s).

Protocol functionality of the proposed anycast routing protocol can be divided into the following different phases.

1. Hello Message Transmission: All APs periodically transmit Hello packets (denoted by HELLO), which only traverse *m* hops (i.e., inside the proactive region), as defined by using the TTL value in the IP header. Upon receiving a Hello packet, the node first determines whether it is within *m-hop* distance from the AP. If so, the route to the AP is created or updated. Only nodes $m-1$ hops away from the AP decrease the TTL value and rebroadcast the packet. The Hello packets include an anycast group identifier number and a generic load metric which represents the load/availability of the AP. This metric value may be updated before broadcasting.
2. Route Discovery *(Proactive Region)*: A node determines that it is in the proactive region if it has received a Hello packet from any AP that belongs to the destination anycast group in the previous Route Expiration time interval. Then, it

can start sending data using the information in the routing table without performing route discovery phase.

3. Route Discovery *(Reactive Region)*: RREQ and RREP packets are similar to AODV specifications, but have additional fields to include an anycast group ID and a load metric. If a node does not have any valid route available to any member of the anycast group in its routing table, it broadcasts a RREQ. Most of the RREQ processing is the same as that described in [93]. RREP can only be generated by AP members of the anycast group or MNs in the proactive area that have an active path to any member of the anycast group.
4. Route Selection: Route selection is related to the cost metric used in the protocol. That is, AODV selects the path with the first RREP. While using the load metric included in the RREQ, our anycast protocol selects the route with the best load value out of the available destinations in the anycast group. If two or more APs have the same load value, then the route with maximum life time is selected to forward the packets.
5. Route Maintenance: Route maintenance is the same as for classical AODV.

3.4.2 Optimal Proactive Radius Analysis

The hybrid proactive/reactive approach in our hybrid anycast protocol can reduce overhead of AP discovery. However, the radius *m* of proactive region is an important parameter, which can greatly influence the network performance. Therefore, in order to drive the optimal radius of proactive region, theoretical analysis on communication overhead is conducted.

Table 3.1 presents all the parameters we use in our analysis. The total overhead of AP discovery can be divided into two parts: Hello messages from APs inside the proactive region and RREQ messages from MNs inside the reactive region. Here, we ignore the RREP messages, since they are sent along unicast routes which lead to a much lower number as compared to the number of HELLO and RREQ messages which are sent by flooding.

Table 3.1: Parameters for analysis

Symbol	Description	Value
R	Network radius	2000m
r	Transmission range of an AP or MN	250m
m	Radius of proactive region of an AP	Various
s	Number of traffic sources in network	Various
l	Number of APs in anycast group	Various
d	MNs distribution density	4 per m^2
t	Total time of operation	500 sec
η	HELLO broadcast interval	20 sec

Each HELLO message floods the proactive region and it can reach m hops, with $m - 1$ re-broadcasts. Therefore, the total number of HELLO messages broadcasted per AP is calculated as

$$\pi((m-1)r)^2 d \tag{3.1}$$

where d is the node density and $\pi((m-1)r)^2$ is the area of the proactive region of this AP. Then, the total number of HELLO messages from all l APs during the whole operation is

$$l[\pi\big((m-1)r\big)^2 d]\frac{t}{\eta} \tag{3.2}$$

Also, we calculate the total number of RREQ messages. If the source node is located in the proactive region, there should be no RREQ overhead; otherwise, the number of RREQ messages per flooding for one route discovery is

$$\pi(R^2 - lm^2r^2)d \tag{3.3}$$

Here $\pi(R^2 - lm^2r^2)$ is the area of the reactive region. Notice that when a RREQ reaches the proactive region of any AP in the anycast group, it will not be rebroadcasted anymore. Since we assume *s* sources are uniformly distributed in the network, the number of traffic sources located in the reactive area is

$$s\,\frac{R^2 - lm^2r^2}{R^2} \tag{3.4}$$

Thus, the total number of RREQ messages of all the *s* sources can be calculated by multiplying Equations (3) and (4):

$$s\,\frac{R^2 - lm^2r^2}{R^2} \times \pi(R^2 - lm^2r^2)d \;=\; \pi sd\,\frac{(R^2 - lm^2r^2)^2}{R^2} \tag{3.5}$$

Therefore, the total overhead can be expressed as a function of *m*, (if HELLO and RREQ packets have the same size)

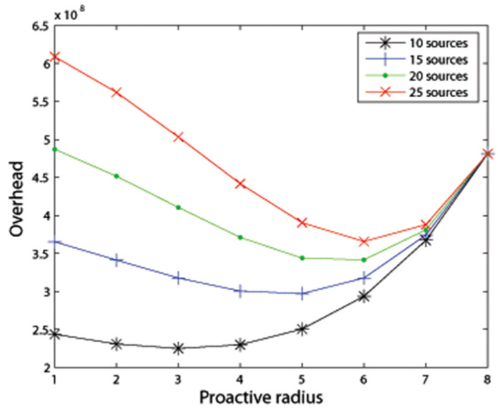
$$f(m) = l\left[\pi\big((m-1)r\big)^2 d\right]\frac{t}{\eta} + \pi sd\,\frac{(R^2 - lm^2r^2)^2}{R^2} \tag{3.6}$$

If HELLO and RREQ are not of the same size, we can modify the above equation to:

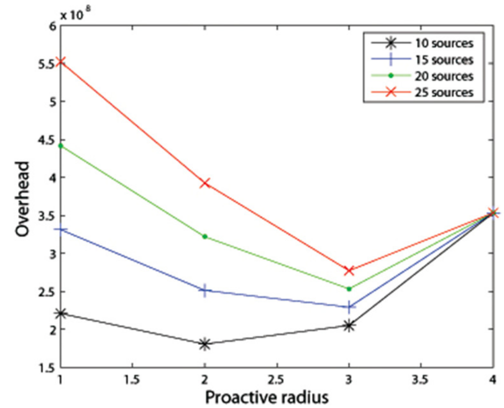
$$f(m) = \alpha l\left[\pi\big((m-1)r\big)^2 d\right]\frac{t}{\eta} + \beta\pi sd\,\frac{(R^2 - lm^2r^2)^2}{R^2} \tag{3.7}$$

where α and β are the HELLO and RREQ packet size, respectively.

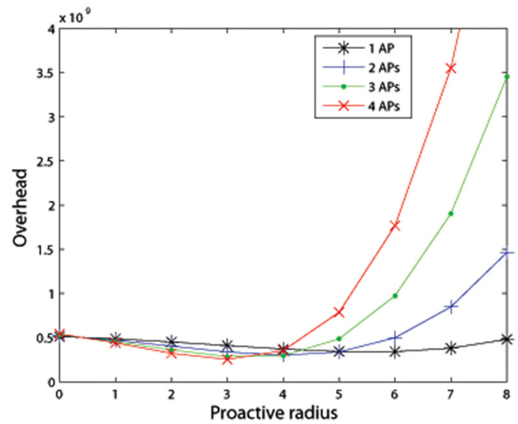
Using a common setting of the parameters, as shown in Table 3.1, we plot the overhead function of *f(m)* with different combinations of *l* and *s*. Figure 3.3(a) shows the plot of *f(m)* with various numbers of sources when only a single AP is inside the anycast group. The following observations are summarized: (1) total overhead increases with the number of traffic source increases, since the number of RREQ messages increases; (2) all



(a) 1 Access Point



(b) 4 Access Points



(c) 20 Sources

Figure 3.3: Overhead vs. Proactive Radius – plot of *f(m)*

curves follow the same trend, first decreasing to the lowest point, then increasing, and merging together when $r = 8$, where it is completely proactive; (3) the optimal value of m (where *f(m)* is minimized) increases as the source number increases, i.e., 3 for 10 sources, 5 for 15 sources, 6 for 20 and 25 sources; this has been confirmed later in our simulation. Figure 3.3(b) shows a similar scenario with 4 APs. The trend of all curves is the same as that with single AP. Notice that the merge point of all curves shifts to 4, since $m = 4$ can guarantee the proactive region covers most of the network. To observe how the quantity of AP affects the overhead function, we fix $s = 20$ and plot the set of curves with different l values in Figure 3.3(c). The curves follow the similar trend (first decreasing then increasing) as shown in Figures 4.3(a)(b). It is interesting that the optimal value of m decreases as the number of AP increases. In other words, as more APs belong to the same anycast group, each AP can reduce its proactive radius.

3.4.3 Load Balancing Scheme

Since the traffic sources and relaying MNs (or even APs) are not evenly distributed in the real access networks, simply using the nearest AP for access may lead to unbalanced load among APs (i.e., some APs may be overloaded while some APs are always idle). Due to wide diversity of access techniques in heterogeneous access networks, different APs may also have various capacities to serve MNs. Thus, load balancing among multiple heterogeneous APs is an important and challenging task. Anycast mechanism has the potential to achieve better load balancing, because multiple APs in the same anycast group can all provide same access services. In this subsection, we use a simple load metric plus several load balancing policies at APs to demonstrate the power of combining load balancing with anycast route selection.

Each AP keeps track of its load information and broadcasts it to MNs within the proactive area via Hello messages. Here, we use the number of packets received per second as a simple load metric, which can be easily extended to other complicated load related metrics. This load metric is included in both HELLO and RREP messages and used during the route selection. Beside the load metric, APs can actively take actions to balance load by changing its attitude towards RREQ and HELLO messages. Our load balancing approach classified the status of APs into three zones based on their traffic load information:

Green Zone: When traffic load is below the threshold Th_{green}, AP is in normal state. Therefore, AP keeps broadcasting HELLO messages and also corresponding to normal RREQ it receives.

Red Zone: When traffic load is between thresholds Th_{green} and Th_{red}, AP is in overload-avoidance state. APs stop broadcasting HELLO message when they enter this state, moreover, they only correspond RREQs from MNs inside its proactive region, which limits connection requests to avoid overload situation.

Black Zone: When traffic load is above Th_{red}, AP is in overload state. APs stop corresponding to any RREQs and decline any new connections. If the traffic load keeps increasing even though there are no new connections coming in, APs can explicitly send a message to the sending source to announce the overload status. Then, the source node can switch to another available AP or start a new round of RREQ if no other entries are available in its routing table.

These threshold values can be configured by network administrators based on equipment properties, network deployment, traffic load or other factors.

3.5 Implementation and Evaluation of Framework using NS-2

In this section we describe the implementation and evaluation through simulation of the complete Integrated Resource Management and Routing Framework for the Hybrid Anycast Routing. We have done extensive modifications to unicast AODV routing implementation provided by Uppsala University. In the following sub-sections, we first describe the communication packet formats, and later describe the simulation experimentation for evaluation. Details of this work can be found in [106].

3.5.1 Routing Packet Formats

Similar to AODV, we have used the basic three types of packets (Route Request, Route Reply, Route Error), and an additional HELLO packet that is different is nature that the optional Hello defined by AODV RFC.

Hello Packet (HELLO): This packet is a special type of packet generated only by the APs, which is broadcast periodically inside the proactive region. As shown in Figure 3.4, the packet includes the following fields:

0	8	16	24	32
Type	Lifetime	Originator Anycast Group ID		
Originator IP Address				
Originator Sequence Number				
Path Cost Metric				
Current Cpacity				

Figure 3.4: AP Hello Packet

Route Request Packet (RREQ): For MNs that do not have any valid route available to any member of the anycast group in its routing table, a RREQ packet is generated to initialize the route discovery. Figure 3.5 shows the format of the packet, which is similar to that of the AODV protocol. The major differences are: instead of using a unicast address as a destination address, the packet uses the anycast group ID as the destination address; two more fields are added for adapting application requirements and utilizing multiple metrics as path cost. Accumulative path cost is the generic path cost to the current node using Equation (1). Generic application requirements include both requirements for AP selection (e.g., capacity) and requirements for path selection (e.g., the weighted factors of each routing metrics).

0	8	16	24	32
Type	Hopcount	Requested Anycast Group ID		
Originator IP Address				
Originator Sequence Number				
Request ID				
Accumulative Path Cost				
Generic Application Requirements				

Figure 3.5: Route Request Packet

Route Reply Packet (RREP): This packet is generated by APs or MNs in a proactive region for corresponding RREQ packets. The format of the packet, as shown in Figure 3.6, adds two more fields as compared to that of AODV. While the destination anycast group ID represents the anycast group that the destination node belongs to, the accumulative path cost is the accumulative cost along the path from the destination node to the source node.

0	8	16	24	32
Type	Hopcount	Destination Anycast Group ID		
Destination IP Address				
Destination Sequence Number				
Originator IP Address				
Lifetime				
Accumulative Path Cost				

Figure 3.6: Route Reply Packet

Route Error Packet (RERR): The route error packet (RERR) is the same as that of the AODV protocol.

3.5.2 Simulated Evaluation in NS-2

For the evaluation of protocol, we assume that different APs may have different capacities, i.e., they can only serve up to a certain amount of traffic. We use the total possible data rate at each AP as the measurement of its capacity. In our simulation, for a simple demonstration, our protocol adopts three path cost metrics: hop count, energy cost, and traffic load. Also, the path cost metric field in a HELLO message contains the three cost metrics; while the accumulative path cost field in a RREQ contains the value of combined path cost from the following equation.

$$cost = cost' + \alpha_1 \times 1 + \alpha_2 \times load + \alpha_3 \times energy\_cost \quad (3.8)$$

In RREQ, the generic application requirement field includes two portions: one is the required capacity (data rate), and the other is three weighted factors for path cost calculation.

We conduct several sets of simulations with ns-2 to evaluate our proposed multiple-metric hybrid protocol. In order to demonstrate how different requirements and path cost metrics guide route discovery and resource consumption, we conduct simulations for three different studies. Summaries of these are presented in the following section.

- *Hybrid Anycasting: Study on Proactive Radius*: In the first set of simulations, we test the performance of hybrid anycasting with different proactive radii. Nodes are randomly distributed in rectangular area, and move suing Random Way Point model [8]. The mobility is variated, and we observe the average delivery ratio and normalized overhead. Results have shown that an optimal radius can be found for networks with different mobility and density values. This optimal point will increase the delivery ratio at a reduced control overhead. This finding gives rise to the conclusion that dynamic algorithms can be used to find optimal proactive radii over time, to improve the performance of the network.
- *Load Balancing using Simple Load Metric*: To give a simple illustration of our load balancing scheme, we conduct the second set of simulations first with a fixed network deployment [106]. We run the simulation both with and without load

balancing schemes. We observe from results that, with load balancing, the traffic load is distributed more evenly among access points as compared to without-load balancing technique. We also conduct load balancing simulations in a mobile environment, and close analysis of resulting network data reveals that load distribution among APs is better when our load balancing technique is used. The upper bound of load per AP is significantly reduced and new connections are forced to find other closer APs.

- *AP and Path Selection using Different Cost Metrics*: In this set, we evaluated three different scenarios as shown in [106]. We first use a simple network with two APs to demonstrate the selection of APs based on AP capacity. We evaluate the throughput with and without our proposed work, and find that our protocol with the capability of load balancing improves the overall throughput of the network. We have also evaluated the network delay, control overhead and energy consumption over different paths. Distribution of network load increases the performance across the board. In the second set of simulation we combine multiple metrics together using Equation 4.8. The results show that this is a more realistic approach to evaluating path cost, and using even simpler aggregation formulas results in significant performance improvement in deliver percentage, delay and control overhead. In the third set of simulations, more complex network architecture was used to evaluate a combination of path costs and application requirements for Aps. Numerous combinations were used to evaluate the delivery rate, network rate, control overhead and energy consumption. Detailed simulation setup and results can be seen in [106]. In a summary, the availability of transporting AP selection criteria and corresponding evaluation method, and the addition of path cost calculation per hop, improves the overall performance of the network. It especially helps in reducing the network delay and increases the average network energy levels. More complex algorithms can used to evaluate path cost, but the key contribution of our work is the ability using the modular approach of the framework to incorporate such algorithms.

CHAPTER 4: ANYCAST ROUTING FOR SENSOR NETWORK

4.1 Sensor Networks

Technological advances in sensor network hardware and its economic availability have enabled them to be used in hundreds of different application scenarios today. Sensors and other hybrid devices can be randomly or strategically deployed in the target environment to monitor, analyze and even react to different stimuli. [87] Civil applications include but are not limited to, animal habitat monitoring, ecological networks, animal migration sensing, marine life observation networks, bridge & structural monitoring systems, smart energy systems, vehicle identification, health care monitoring, and many more. A major usage of sensor networks has been observed in military applications for battle field surveillance, nuclear/chemical/biological hazard detection, target tracking, etc. The concept of ad hoc networking has further facilitated the usage of these networks.

The increase in applications for such networks has also increased the complexity of these systems. Regardless of the fact that hundreds of research contributions are published every year in this domain [88], many of the research issues are still open ended. Each network layer including security, heterogeneity, network management, and architectures have numerous sub-research topics. [89] described 6 major challenges in sensor network research. The growth of application domain has complicated these issues, but 3 broad categories of challenges can be identified: 1) Physical (hardware, power, transmission range, memory, processing capabilities, mobility, link capacity, etc.), 2) Communication (routing, path/sink discovery & maintenance, QoS, overhead, scalability, etc.), and 3) Application (data query and collection, etc.).

Future sensors and ad hoc networks will not comprise of an isolated collection of sensing nodes. These will be a combination of different types of devices and sub networks, integrated to form a complex network. Furthermore, the mobility and other hardware characteristics will vary from a static tiny sensor, to mobile handheld data collector with visual display, to high mobility low flying objects, and so on. This hybrid nature of the sensor network is dictated by the application scenario. For example, a sensor network deployed for warehouse inventory control is totally different than that deployed in a battle field to monitor enemy movement, and so are the protocols that are required to run over it. Generic routing protocols designed for wireless ad hoc networks fail in sensor networks primarily due to the fact that they are designed for more powerful nodes with higher transmission range and power capacity as compared to sensors. In addition to this, the packet structure, routing table sizes, implemented code size, and many other states that are maintained, cannot be ported directly on to tiny sensors.

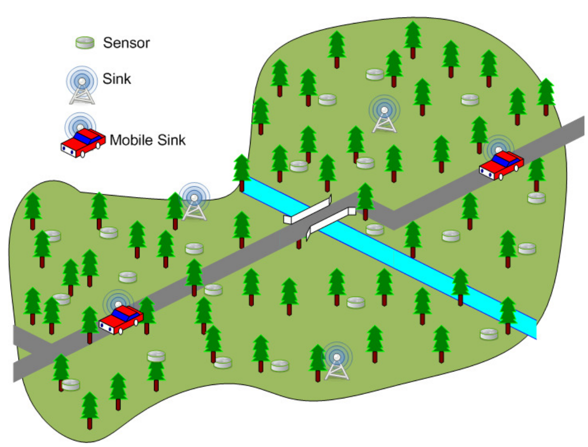


Figure 4.1: Sensor network in a forest

In this chapter, we address the sink discovery and routing in application specific hybrid sensor networks, where tiny sensors are deployed over a very large geographic area to monitor different environment variables. These variables may include temperature, motion, EM fields, etc. Due to the nature of these nodes, they have little or no mobility at all. Mobile robotic platforms (sinks) are used to collect data. These platforms have higher communication ranges (possible a second communication layer), energy and processing capabilities. This situation can be mapped to real world scenarios of forest fire monitoring (Figure 4.1) and battle field operations (Figure 4.2). In a battle field scenario, hundreds to thousands of tiny sensors can be dropped to monitor enemy movement (rather than the use of landmines) [90], measure effectiveness of attack, and detect biological or chemical warfare, etc. Mobile robots or vehicles (equipped with better communication & processing power, higher energy levels, and more sophisticated hardware) can be used as sinks to collect data from such a large terrain. Thus the whole network is physically 2 tiered. A network of tiny sensors, which gathers basic information such as motion, which is collected by 1 out of *n* available sink nodes. Secondly, a network of mobile sinks, that can communicate to strategically place themselves to gather maximum data.

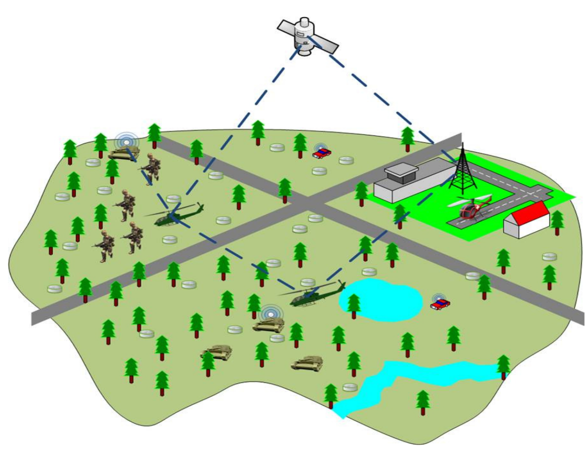


Figure 4.2: Sensor network in a battle field

4.2 Challenges in Sensor Network Protocol Design

The communication architecture/protocols of a sensor network are dictated by the application it is being deployed for. There are several routing protocols available for sensor networks [48]. Performance of these protocols varies from application scenario to scenario. It is almost impossible to conceive a routing protocol that can perform equally well for all application platforms.

Sensor network routing protocols can be classified in many ways. The basic classification that is inherited from ad hoc networks taxonomy is of reactive & proactive routing. In reality the routing technique should be bound to the architecture of WSN. Some WSNs are required to periodically sense and generate data, for which routes to the sink need to be maintained all the time. On the contrary, some networks are designed only to provide data when queried. In addition to this, the initiator of route discovery also varies. Some sensor networks may require the sensors to report data when certain events happen (e.g. rise in temperature, change in health status of soldier, etc.), which will require the sensor to originate a route discovery message. On the other hand, if the sink or data collection point needs data from a particular sensor or group of sensors, then that side originates the route discovery messages.

The particular battle field operation or forest fire scenario, it is assumed (and practically it is) to be a query and/or event based WSN, where data is not generated periodically; rather it is generated when queried, or when a certain event happens (e.g. movement detected by the sensors). This requires a reactive routing strategy.

AODV [44] is an on-demand reactive routing protocol designed for ad hoc networks. Scaled down versions of AODV to fit WSN needs are discussed in [91]. These protocols are designed for node-to-sink, and some for node-to-node communication.

As argued in [89], the majority of protocols are designed with simplified assumptions making them non-optimal for particular scenarios. E.g., in battlefield operations, or forest fire monitoring scenarios, the sensor nodes are relatively static, and there is no need for node-to-node communication. A temperature sensing node may relay the information towards the sink, but will not be required to process it. Also, a redundant number of sinks in the form of robots, unmanned vehicles, and other devices may be present in the area. In certain situations, there could be specialized vehicles for different purposes. Also different sinks might need different types of data with different requirements. For example, the commander of a group of soldiers engaging the enemy will need enemy movement data with the least possible delay, disregarding the energy costs & other parameters for that localized area. On the other hand the command HQ may try to increase the sensor network life time, by balancing the data routes over the network. Thus there is a need to send data to the optimal sink. This gives rise to the challenge of sink and path selection across the entire network. These challenges are often overlooked in the design of the sensor network routing protocols. The motivation of our architecture stems from these challenges.

4.3 Sensor Network Communication Framework

To overcome the challenges discussed in the previous section, we have developed a new architecture for communication between the sensor nodes and the sinks. This architecture is primarily based on the Anycast Hybrid Routing Framework discussed in the previous chapter. Many of the capabilities have been borrowed from it, but major modifications have been done to the protocol that runs inside the modules of the architecture. Figure 4.3 shows the main modules of this framework.

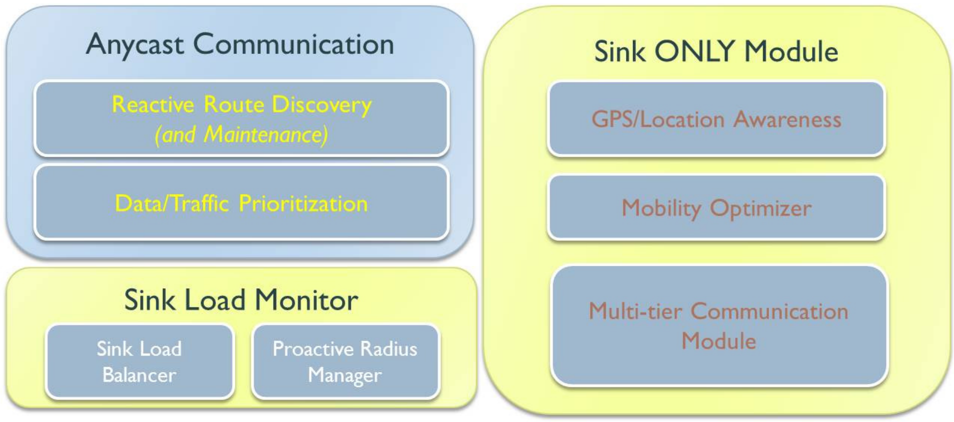


Figure 4.3: Sensor Network Communication Architecture

As seen in the figure, framework is divided into 3 major parts. All these parts are implemented at the network layer of the communication stack. In the following sections we discuss each module and its sub-modules in detail.

4.3.1 Anycast Communication

This is the most important module of the framework, as it is responsible of discovering the routes, maintain them, and provide sensor data packets with prioritization. These tasks are achieved by two sub-modules as shown in the diagram. Every device in the network implements this module (as compared to the others), as it is the core component of the system. We discuss the working of the sub modules as following:

Reactive Route Discovery: The routing discovery is reactive in nature, and follows closely the rules of Anycast Hybrid variation of AODV discussed in the previous chapter. In contrast to the previous rules, the implementation strips off all unnecessary information from RREQ and RREP messages. This includes the sequence numbers and

Originator IP address. Information about sequencing is removed as the data carried is real time information. As argued before this information is small bits of environment data, and it is always beneficial to have the latest and discard the older one. This differentiation can easily be made from time stamps in the packet header. As for the IP address, usually the sensor nodes use sensor identifiers instead of IP addresses. Thus the IP address filed is replaced with the 2 byte identifier. In addition, the Request ID filed has been reduced to 2 bytes from 4 bytes, and Accumulative Path Cost and Generic Application Requirements take 2 bytes each. Thus, effectively we reduce the size of RREQ packet from 24 bytes to 12 bytes. Similarly the RREP packet is reduced to 12 bytes. Details of packet structure can be seen in the implementation section.

Data/Traffic Prioritization: This is a new sub module introduced in this architecture. The purpose is to add a small header with each sensor data (after connection & path establishment), to notify the intermediate nodes about the priority of this data. Higher priority data is forwarded first. This helps in providing better service to time sensitive data that have to reach the sink on priority. The overhead added with this head is very minor and negligible as shown in the simulation results.

4.3.2 Sink Load Monitor

Monitoring of the data flow at the sink is one of the most critical, as it determines the load on a particular sink, its capabilities, and life time. Optimizing the life time and the type of services it can provide, the sink must have a mechanism determine what information it can receive and how many data routes it can support. Out architecture includes a dedicated sink load monitoring module, which provides the sink with protocol definitions of configurable traffic management. The purpose of this module is not to 'manage' the load, but to provide architecture that can 'implement solutions' to manage the load, type of service, visibility to sensors, and optimization of life time. It is also important to note that this module is present at all the sensor nodes, because under certain circumstances a sensor node may be required to perform as a sink (unlike the third module of architecture, which is present only at dedicated sink nodes). This module further has two sub modules, described below.

Proactive Radius Manager: The purpose of this sub-module is to control the radius of the HELLO packets broadcasted by the sink to advertise its presence to the sensors. In our original Hybrid Anycast Routing Protocol, a similar module was present. This module is very similar in nature, with pre-configurable or dynamic approaches to setting the proactive radius.

Sink Load Balancer: This sub-module is responsible of multiple functionality support that is required to support the load balancing. It is responsible of 1) determining if the sink node is capable of provided a certain requirement (delay, fault tolerance, etc.) to the incoming route request (requirements of AP selection); 2) determining the number of active connections to the sink & accepting/rejecting more route requests; 3) determining if the path taken by RREQ satisfies the requirements (requirements of path selection); 4) reserving the resources at the sink for the accepted route requests; 5) assuring that the accepted RREQs are provided the QoS throughout their route life time; 6) overall monitoring of the connections for optimization.

4.3.3 Sink ONLY Module

As the name suggests this module is only functional on the sinks that have the capability of implementing it. Due to tremendous technological advancements, it is not uncommon to find the required hardware in sink nodes. In a realistic deployment of sensor network (again referring to the forest fire, or battle field examples), the sink can be a node mounted on mobile robots, low flying air crafts, vehicles, human personal. Thus, hardware like GPS trackers, long range & relatively powerful communication equipment, extended battery life, and more processing capability are not uncommon. Three sub modules are proposed in this module, but more components can be added as required by the sensor network.

GPS/Location Awareness: This sub module is functional if the sink is capable of locating its positions using GPS or other technologies. Once location awareness is achieved it can be used to optimize the mobility of sink, increase the precision of the sensor information, etc.

Mobility Optimizer: The purpose of this module is to provide mobile sink with information about possible directional movement to better cover the sensor network. IT may include movement to optimize the information generated by a particular region of the network, or to maximize the foot print of the sink. In either case, the sink has to be aware of its location, either by means of a dedicated positioning hardware or logically built map based on traffic flows.

Multi-Tier Communication Module: This module is a concept introduced to integrate multiple communication technologies available at the sink. In real world it is very much possible that the sensor nodes use low level communication systems with limited capabilities. This gives the extended battery life, size and cost reductions, etc. But as the sinks may be larger devices with more resources available, they would be capable of having multiple communication technologies, multiple antennas, etc. Thus they can affectively build another layer of communication among themselves or other networks. This capability of having multiple layers of communication can be utilized to communicate locally collected information, optimize positioning, movement, and other benefits. We discuss the details of mobility optimization in later chapters.

CHAPTER 5: LIGHTWEIGHT HYBRID ANYCAST ROUTING PROTOCOL FOR SENSOR NETWORKS

We have implemented our proposed architecture for Sensor Networks Hybrid Anycast Communication using a light-weight version of previously proposed Hybrid Anycast Routing Protocol. The implementation includes the definition of packet formats, path discovery and maintenance processes and related capabilities. In the following sections we discuss these in greater detail.

5.1 Protocol Design Details

Protocol design includes precise definition of the control packet structures, the exchange mechanisms, algorithms for their working, and the implementation details. The implementation of our protocols is done in a simulation environment, which will be discussed in the next section.

5.1.1 Packet Structures

In an ad hoc communication protocol that relies on a hybrid nature (reactive & proactive), four control packets are necessary. Similar packets were discussed in the previous chapter for hybrid anycast routing protocol implementation. Also, the working is primarily based on AODV architecture, but with major modification.

Hello Packet (HELLO): Hello is a special packet introduced in the light-weight protocol that is responsible for advertising the presence of sink. Every sink broadcasts this packet in a radius of n hops. The number of hops (proactive radius), can be adjusted dynamically or preconfigured depending on the requirements of the network application.

0	8	16	24	32
Type	Life Time	Anycast Group ID		
Originator Identifier		Path Cost Metric		
Sink Capability/Capacity				

Figure 5.1: Hello Packet

The packet structure is considerably smaller than that of Hybrid Anycast Routing Protocol. The total size is 12 bytes as compared to 20 bytes of previously proposed Hello packet in hybrid anycasting protocol.

- *Type*: 1 byte field to identify the type of packet
- *Lifetime*: 1 byte filed that specifies the time for which this information is valid
- *Originator Anycast Group ID*: Identifies the group of anycast sinks that this sink belongs to.
- *Originator Identifier*: Uniquely identifies the sink that originated this packet
- *Path Cost Metric*: 2 byte value which carries the cost of path along which this packet travelled.
- *Sink Capability*: 4 byte field to specify the capabilities of this sink.

It is possible that the transmission range of the sink is larger than that of the sensor nodes. Thus this variable must be carefully used as a node receiving a HELLO packet may not be in range in the opposite direction of communication. It is recommended that the sink nodes should reduce their transmission range to match the range of sensor nodes. This will automatically remove the problem of two way communication. In cases where this is not desirable, then the protocol recommends to disable the proactive cashing of routes to sink by sensor nodes. In such a scenario, the sensor nodes can maintain '*sink information table*' for saving information about possible destinations in the vicinity. This information can then be used to control the RREQ broadcasting to certain number of hops for discovering the sink. In contrast to expanding ring search used in AODV or Hybrid Anycast Routing Protocol, this technique will result in reduced control overhead in larger networks with fewer sinks.

0 8 16 24 32

Type	Hop Count	Requested Anycast Group ID
Originator Identifier		RREQ ID
Path Cost Metric		Requested App. Requirements

Figure 5.2: Route Request Packet (RREQ)

Route Request Packet (RREQ): These packets are generated by any node that requires route to a destination. This can be initiated by a sensor node to a particular sink, or from a sink to a sensor. In either case the packet size and contents remain the same, as show in Figure 5.2. This packet is also considerable smaller (12 bytes), than its counter parts of AODV and Hybrid Anycast Routing Protocol (24 bytes), although, the information content is similar. We have removed the sequence numbering of originator altogether, as the information to be sent is very small, and in majority of the cases will fit in one packet. Thus the packets arriving latest at the receiver can be identified by the time stamp in lower layer packets. Path cost metric and Application requirements have been reduced to fit into 4 bytes.

Route Reply Packet (RREP): Route reply packets are generated in response to the RREQ packets, if and only if the sink satisfies the requirements demanded by the originator in the RREQ packet. Similar to size as of RREQ packet, RREP packet contains the originator and destination identifiers, cumulative path cost metrics that were obtained from the RREQ packet, and the life time filed which contains the validity time period of this information. As the RREP packets are only generated if the sink can provide the quality of service required, there is no need to re advertise the capabilities of the sink. The structure of the packet is shown in the figure below.

0	8	16 24 32
Type	Hop Count	Destination Anycast Group ID
Destination Identifier		Originator Identifier
Cumulative Path Cost Metric		Life Time

Figure 5.3: Route Reply Packet (RREP)

Route Error Packet (RERR): Route errors are occasionally encountered as the sinks may be moving in and out of range of other nodes, and intermediate nodes may not be accessible to any number of reasons. Thus there is a need to notify the sender and receiver of information that the path has broken. This plays a critical part in the re-establishment of routes.

0	8	16 24 32
Type	Hop Count to Destination	Unreachable Destination Anycast Group ID
Unreachable Destination Identifier		Reserved

Figure 5.4: Route Error Packet (RERR)

The packet contains information about hops to the unreachable destination, it's anycast group ID, and it's identifier number. 2 bytes are reserved in the end of this packet. In the current implementation we have not used these bytes, but in future extensions this space can be used to specify the reasons for route error. It is also important to note that we have removed the functionality of *n* number of destination listings as available to

AODV and Hybrid Anycast Routing Protocol. Although, it is easy to extend that functionality in our proposed architecture, but based on our simulation results we have removed it from the basic design, as it was never utilized.

5.1.2 Protocol Operation

The protocol operation is conceptually described in the functionality of the architecture. Implementation follows the same guidelines.

Sink Advertisements: All sink nodes transmit HELLO messages to advertise their presence to the sensor nodes. As discussed earlier, the transmission range is critical, as sink nodes may be more powerful as compared to sensor nodes. If the transmission range of the sink is larger, then the reception of the HELLO at sensor node cannot guarantee that 2-way communication is possible. In our previous Hybrid Anycast Routing Protocol, it was assumed that all nodes have same transmission range. Although it is an acceptable assumption for ad hoc networks, but with sensor nodes it become unrealistic. To avoid this 2-way communication issue, we recommend in our protocol that either the sink nodes reduce their transmission range to match that of sensor nodes, or sensor nodes use '*sink information table*'. In either case the strategy has to be decided first, and configured into the nodes. If the first strategy is used (same Tx range), the sensor nodes cache the route and same routing procedures are used as described for Hybrid Anycast Routing Protocol. In this case, the proactive radius can be increased to *n* number of hops.

Sink Proximity List: This is a unique approach that we propose to avoid the 2-way communication problem when the transmission range of Hello packet is greater than that of sensor nodes. Sink Proximity List is a specialized table of sinks and their capabilities maintained by the sensor nodes as they receive the HELLO packets. The purpose is to cache the knowledge of presence of a sink in the vicinity. Although the sink is not in immediate neighborhood, and there is no way of knowing how many hops, but the information can be utilized to reduce the expanding ring search overhead. Traditionally in AODV and Hybrid Anycast Routing Protocol, the nodes initiating the RREQ adopt the expanding ring search to avoid global flooding of network. With the Sink Proximity List, the sensor nodes can start RREQ with a relatively larger initial radius than 1 hop. Simulation experiments have shown that over longer periods of time with large networks, and smaller number of sink nodes, this technique helps in reducing the control overhead.

Route Discovery: Route between sensor and sink can be required by either of the nodes. As our architecture design argues that sensor information can be generated either in response to a stimulus on senor side, or demanded by the sink explicitly. The node that requires the route initiates an RREQ (if a route is not cached), with appropriate information in the RREQ packet. As the RREQ packet travels the path cost is calculated and replaced in the packet (in case the sensor is requesting). If the sink receiving can satisfy the requested QoS requirements, and the sensor accepts the returned path cost tin RREP, the route is established and data transmission proceeds. As the sink does not need a guarantee of service from sensor, requirements of the application are skipped.

Route Maintenance: Route maintenance is similar to Hybrid Anycast Routing Protocol. Routes are repaired if broken, by local repair, or RERR messages are sent to the source for a new route discovery.

5.2 Evaluation and Analysis

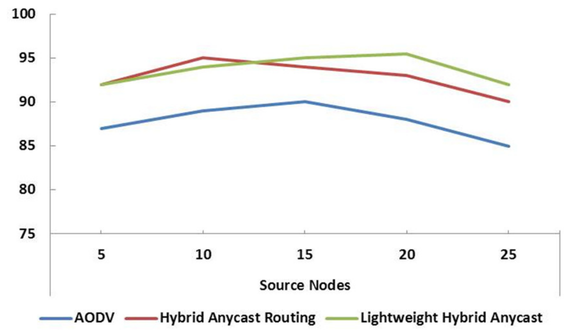
We have evaluated the implemented protocol using NS-2 simulation. To simulate the sensor nodes we have used the built-in node models with minor modification to reflect the limited transmission range. All the nodes use 802.11 wireless protocols for communication.

We have divided the evaluation in different scenarios to assess different capabilities and performance of the proposed protocol. Each of these scenarios has different setup parameters and evaluation objective, but they all evaluate the fundamental concepts of network layer routing capabilities, by changing the network parameters All of the simulations are analyzed for delivery rate, delay, and control overhead. We discuss more elaborative and realistic application scenarios in future chapters.

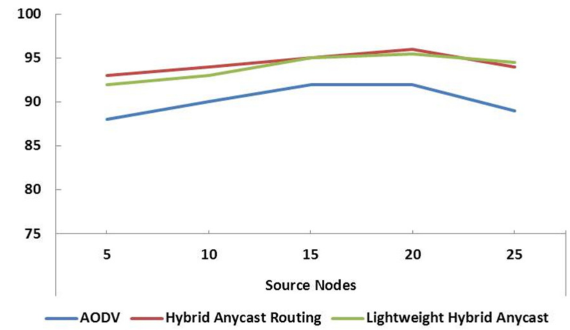
5.2.1 Basic Protocol Operation Evaluation

The objective of this part is to evaluate the basic routing capability of the protocol. This routing capability is tested by changing the number or source nodes, and movement patterns of the sink nodes. All sensor nodes are static, as argued in the previous chapters. The total area is 2000 x 2000 meters, in which the 100 sensor nodes are randomly, but uniformly distributed. The sensor nodes generate data at a rate of 4 packets per second, where the source nodes are selected at random, and vary between 5 to 25 with increments of 5. There are 6 sink nodes and sink mobility is based on Random Way Point model. The pause times used are 0 second, 5 second, 10 second, 30 seconds, and 100 seconds, and 150 seconds. The total simulation time is 150 seconds, and confidence interval is 50 simulations. In this set of simulations we assume that all the sinks have same capabilities and there are no specialized requirements for path and sink selection. Furthermore the sink transmission range is reduced to match the range of sensor nodes, and *Sink Proximity List* is not used.

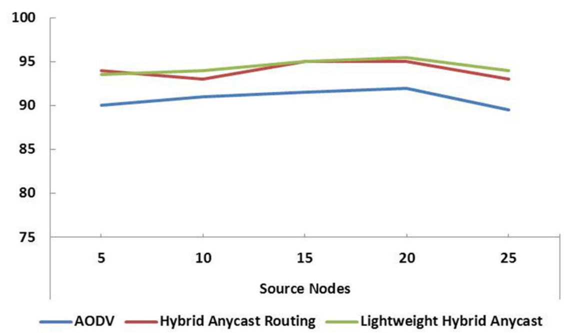
Simulation Analysis: The results show in Figures 6.5 & 6.6, display the percentage of successfully delivered data packets. We observe from the results that the overall performance of the Light-Weight Ad hoc Routing Protocol and that of our previously proposed Hybrid Anycast Routing protocol are almost identical. As we have a high confidence interval (50 simulations), the averaged results provide a smother curve in the performance lines. Clearly in all of the simulated cases AODV is lagging behind in performance. Although its overall delivery rate is in between 85 to 90%, but the real benefit is provided by the hybrid proactive radius establishment of the sink nodes. Also it was interesting to note that the network seems to perform better when the pause times are not too high and not too low, which leads to a suggestion, that moderately mobile sinks are able to gather mode data from sensors. In Figure 5.6, we average the same results to plot them against pause times, and observe that AODV is clearly behind in performance, and both Hybrid Anycast and the Light-Weight version perform head to head in all of the situations. The delay performance of AODV is also not at par with either of the proposed protocols (Figure 5.7). It is important to remember that, in this set of simulations we compare both of our proposed protocols against each other also. As the simulation have shown the delay performance is almost similar, and with the increase in confidence interval, it will become more and more similar to each other. The basic reason for that is the fact that, they are both based on similar routing approaches. Figure 5.8, depicts a similar result: i.e. moderately mobile sinks are able to achieve better delay performance, and Light-Weight Hybrid Anycast Protocol & Hybrid Anycast Routing Protocols perform similar to each other with no statistically significant difference. The biggest advantage of Light-Weight version is in the simulation results of control overhead. We have measured the control overhead in terms of control bytes transferred per data byte sent. In Figures 6.9 and 6.10 we see clear statistically significant difference in the control overhead of Light-Weight version as compared to the full version proposed earlier. AODV again due to its nature is in third position. In conclusion, the simulations have proved that our newer light-weight version for sensor networks can perform as good as a full Ad hoc routing algorithm would. This acknowledges our basic hypothesis that, if the ad hoc routing algorithm is tailored to the needs to specific sensor network, then their performance results are comparable, and ultimately leading to better performance if specialized to cater the needs of specific application scenarios of sensor systems.



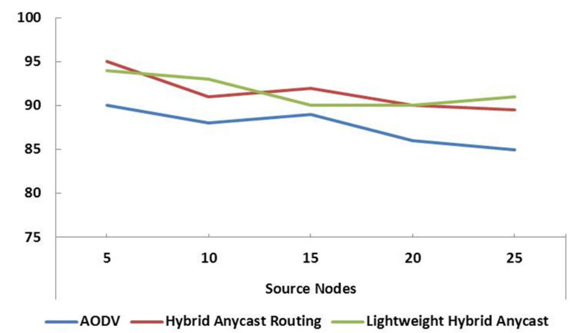
(a) Pause time 150sec



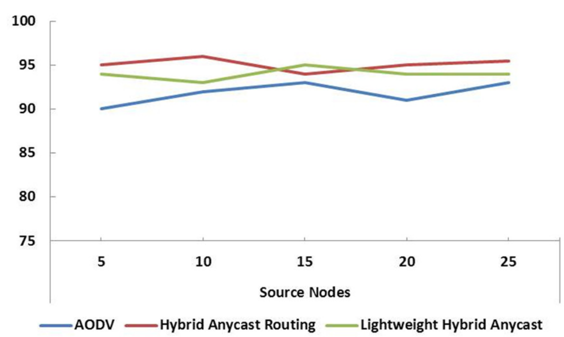
(b) Pause time 100 sec



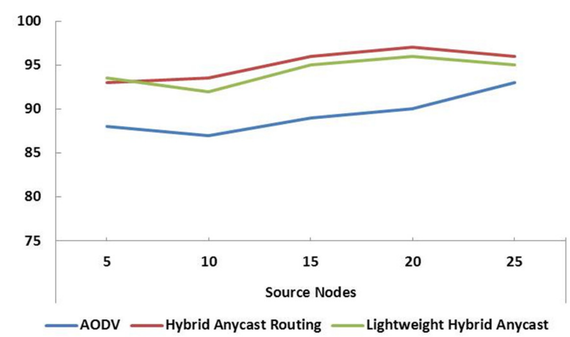
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.5: Delivery percentage for basic protocol operation

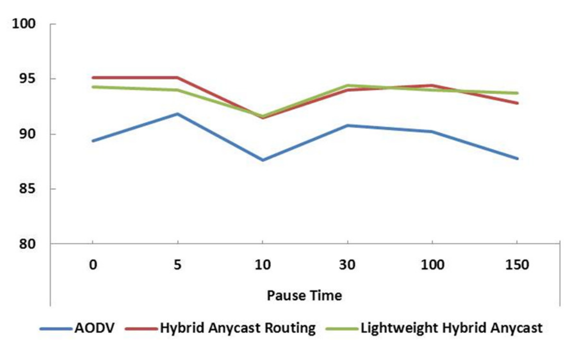
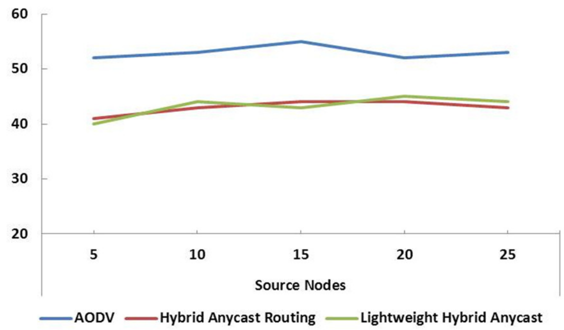
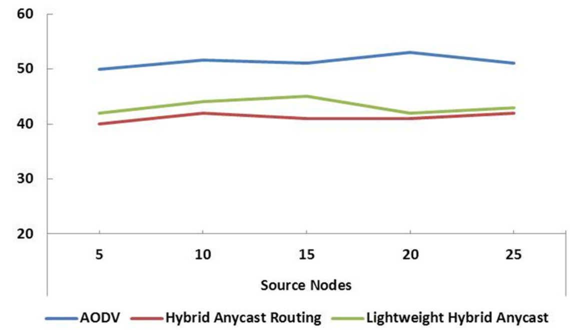


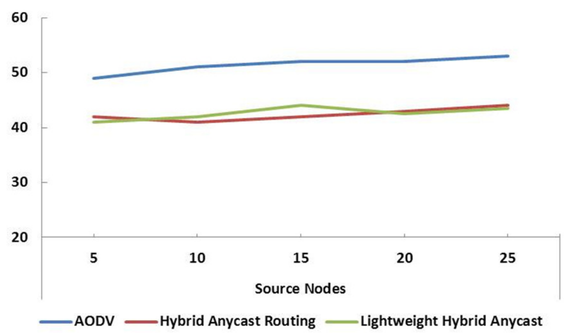
Figure 5.6: Delivery percentage against pause time of sink nodes



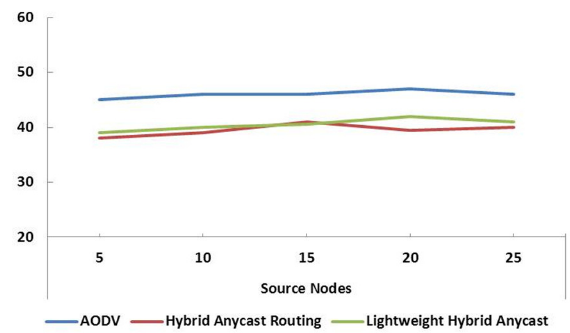
(a) Pause time 150sec



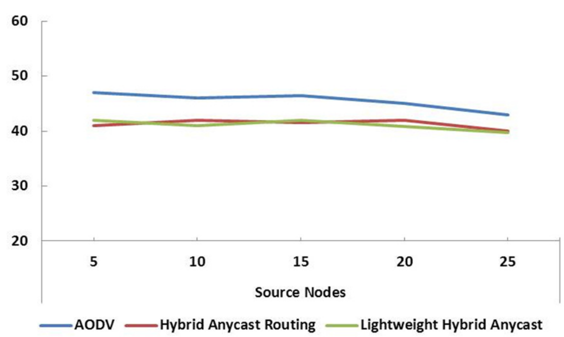
(b) Pause time 100 sec



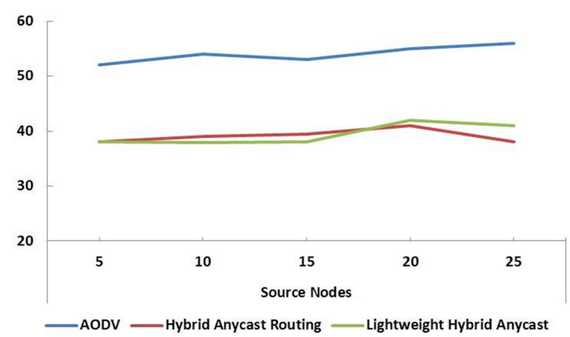
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.7: Delay of data packets measured in *milliseconds* (*ms)*

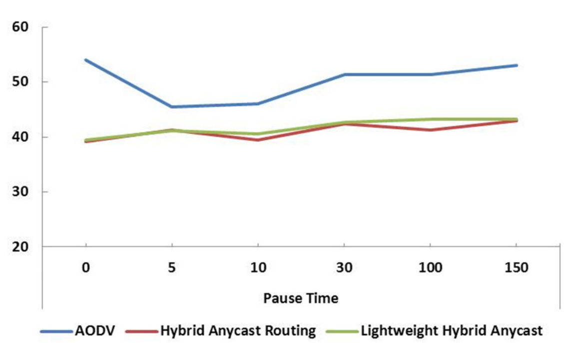
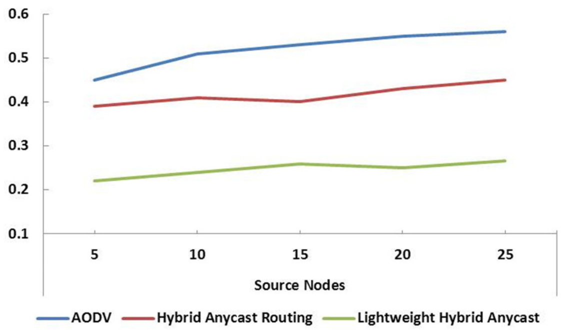
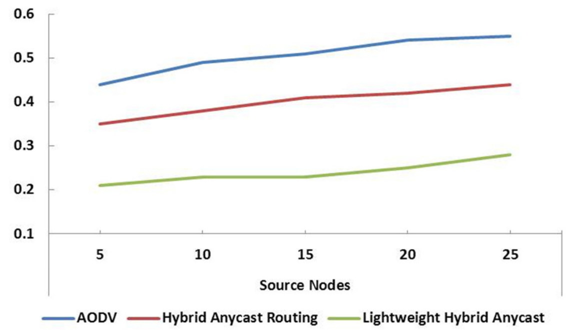


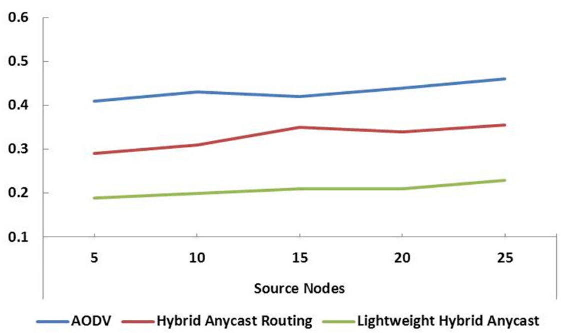
Figure 5.8: Delay of data packets against pause time of sink nodes (*milliseconds*)



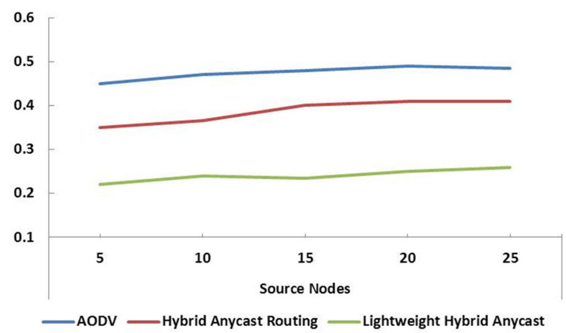
(a) Pause time 150sec



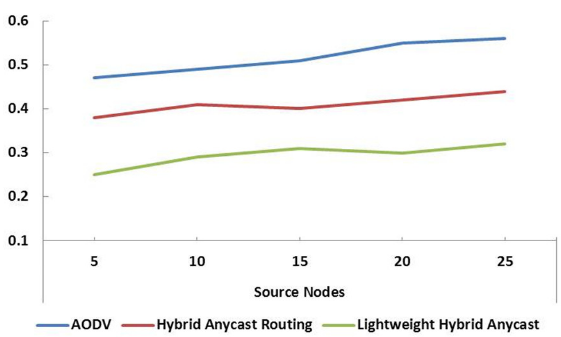
(b) Pause time 100 sec



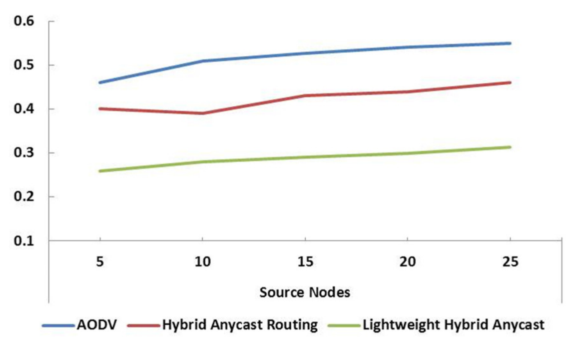
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.9: Control overhear, measured as control bytes sent per data byte

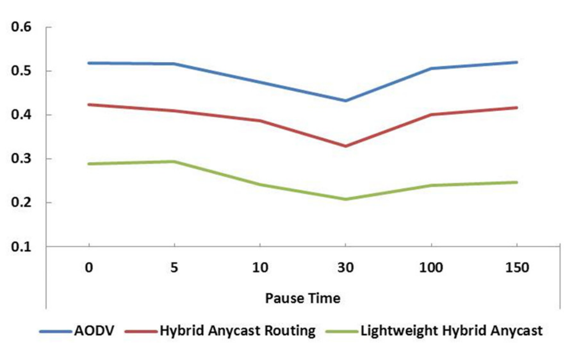


Figure 5.10: Control overhead against pause time of sinks. (*Control byte per data byte*)

5.2.2 Hybrid Anycast Capability Evaluation

In this evaluation experiment, we evaluate the anycast and load metric capabilities of the protocol. This is achieved by dividing the sinks into multiple anycast groups. The sensor nodes then explicitly request the route for a particular anycast group members.

The total area is 2000 x 2000 meters, in which the 100 sensor nodes are randomly, but uniformly distributed. The sensor nodes generate data at a rate of 4 packets per second, where the source nodes are selected at random, and vary between 5 to 25 with increments of 5. There are 6 sink nodes and sink mobility is based on Random Way Point model. The pause times used are 0 second, 5 second, 10 second, 30 seconds, and 100 seconds, and 150 seconds. The total simulation time is 150 seconds, and confidence interval is 50 simulations.

We have divided this experiment into two sets. In '*Set A*' the nodes are divided into 2 anycast groups. In '*Set B*' the sink nodes are divided into 3 anycast groups. The parameters described above remain the same for both sets.

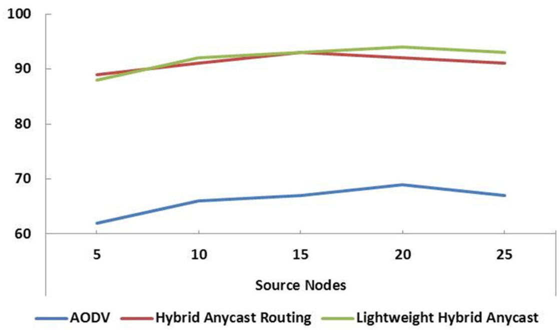
Simulation Analysis of 'Set A': In this set of simulations the results (Figure 5.11 – 5.16) are very different than the previously done simulations, as the destinations are selective. Set A has two groups of Anycast sinks, and the source nodes randomly choose which one they want to send the data to. As a result the routes change more often. As AODV does not have anycasting capability, we modified to simulation system to group IP addresses of sinks. The actual functionality of AODV remains intact, but the simulator authorizes the sinks to send a reply or not to send a reply. Based on this we have observed that the Data delivery percentage of Hybrid Anycast Routing Protocol, and the Light-Weight Hybrid Anycast Protocol perform very close to each other. More than 90%

of data packets are delivered by both of them in almost all of the simulation scenarios. AODV on the other hand, is not able to perform anywhere close to them as it is not designed to cater different sinks, or contain path cost metrics. As see in Figure 5.11 and 5.12, regardless of number of sources sending data, or mobility patterns, our proposed protocols have performed equally well. They have been found to locate the required sink and deliver data successfully 90% of the time. The delay performance of our proposed protocols is superior to AODV as seen in previous simulations and also in *Set A* (Figure 5.13 & 5.14). The interesting thing to note here is the fact that the delay performance in the presence of Anycast sink nodes is still the same (the change is statistically NOT significant), as compared to that of without the sinks. This is a great advantage, as no meaningful additional delay is introduced when sinks are selectively accepting packets, or when paths are being selectively chosen by the source nodes. The overlapping lines in Figure 5.14 show how close both the protocols are and have relatively little affect by the mobility of the sinks. We will observe more significant changes against mobility in the next chapters, when we explore sink movement in greater detail. The overhead performance, as expected, is better of light-weight protocol. The reduced size of control packets has reduced the overhead to almost half as compared to Hybrid Anycast Routing Protocol. In light of similar data rate and delay performance, the decreased overhead makes the light-weight protocol an ideal candidate for sensor networks. As observed in Figure 5.16, mobility grouped with selective anycasting and path metrics have increased the overhead, but we found it not to be statistically significant. The change is in the order of 3rd decimal place, which has little effect on the performance of the protocol.

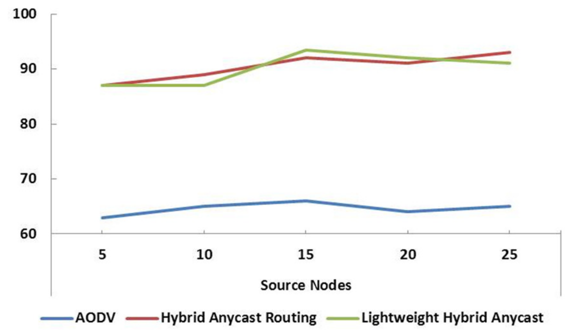
Simulation Analysis of 'Set B': In Set B, the number of anycast groups was increased from 2 to 3. The total number of sink nodes remains at 6. The objective in this simulation is to see the effect of increased number of anycast groups.

As seen from the Figures 5.17 to 5.22, the difference is not much. As a matter of fact after averaging over the confidence interval, the change in value is undetectable. The only noticeable difference is in the performance of AODV, which falls as the number of sink nodes increase. Thus this set of simulation supports the fact that the change in number of anycast groups does not affect the performance of either the Hybrid Anycast Routing Protocol, or the light-weight version of it. Both can perform equally well. A valid question here is that, the change in number of anycast groups is very small, i.e. 2 to 3. This may explain the insignificant change in results. As a matter of fact, the change in number of groups from 2 to 3, reduces the number of sinks in every group by 33%. Given the total size of the network and number of nodes, this is a significant change. Also, it is important to mention that this research focus on the practicality and in realistic deployment of protocols. We have fond that the average number of different anycast groups ranges between 1 to 3 depending on applications.

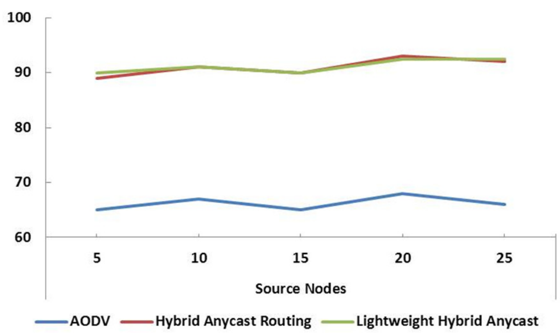
Figure 5.23 shows the same result, in comparing Set A and Set B control overhead performance. The comparative bars show the superior performance of Light-weight version over the other protocols. At the same time we see no difference in the overhead incurred in Set A or Set B.



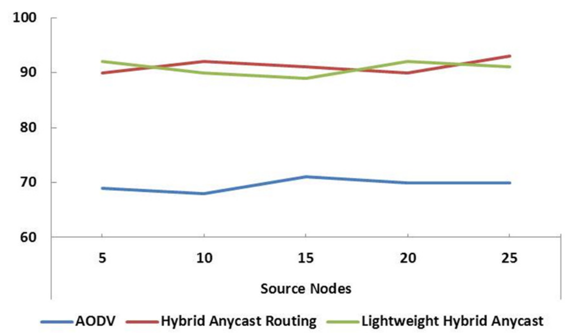
(a) Pause time 150sec



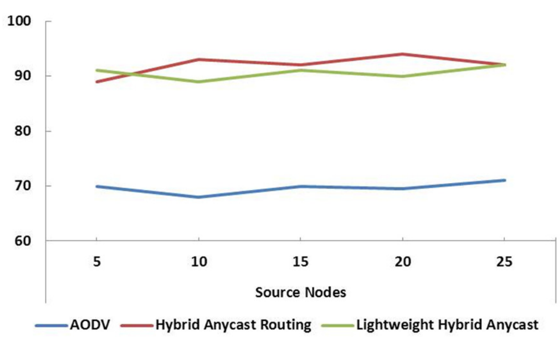
(b) Pause time 100 sec



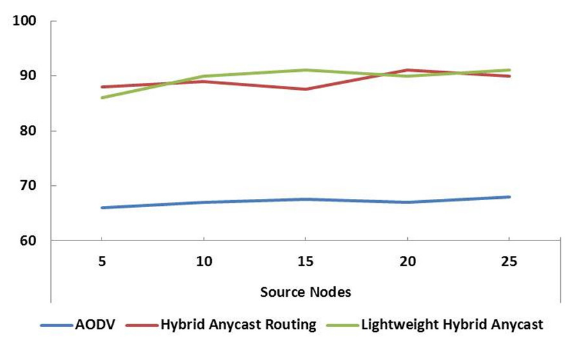
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.11: Delivery percentage for *'Set A'*

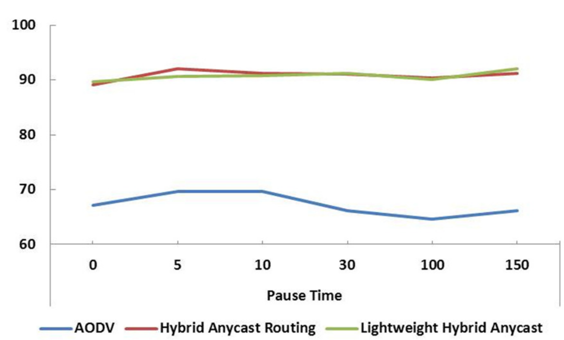
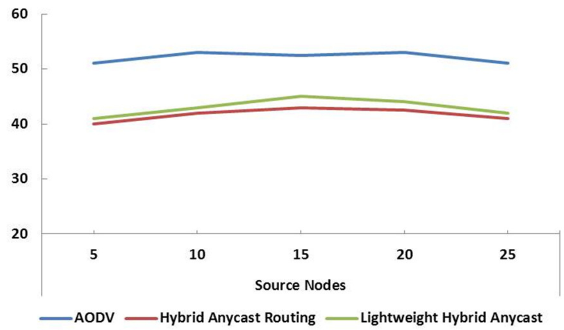
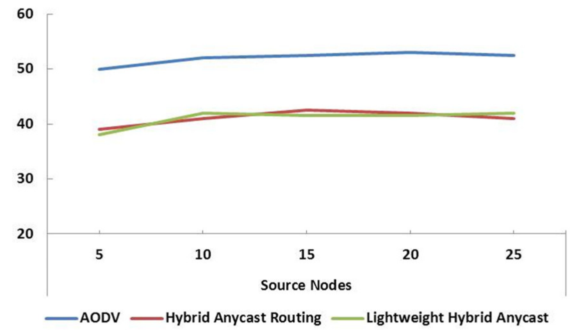


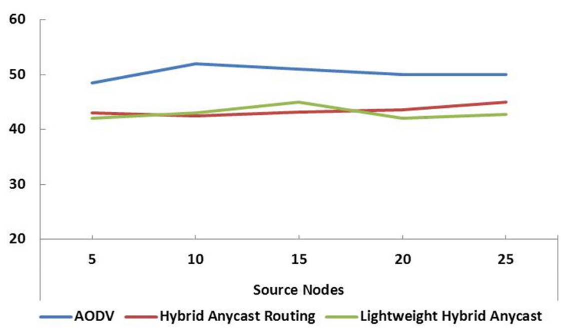
Figure 5.12: Delivery percentage against pause time of sink nodes in *'Set A'*



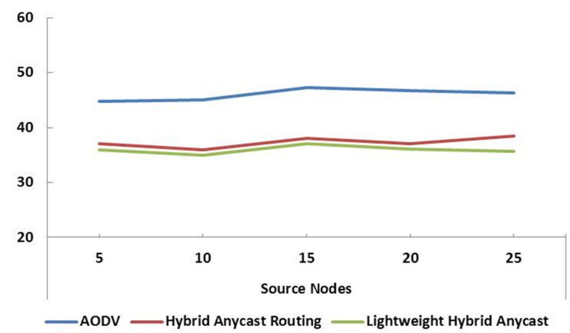
(a) Pause time 150sec



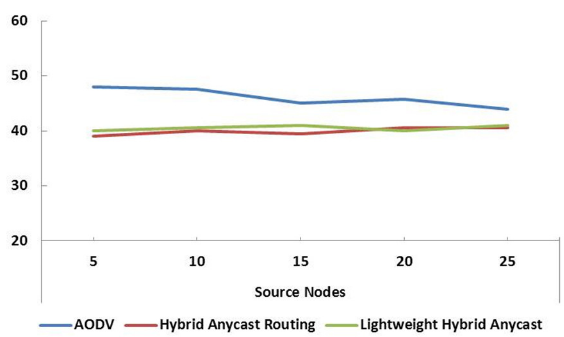
(b) Pause time 100 sec



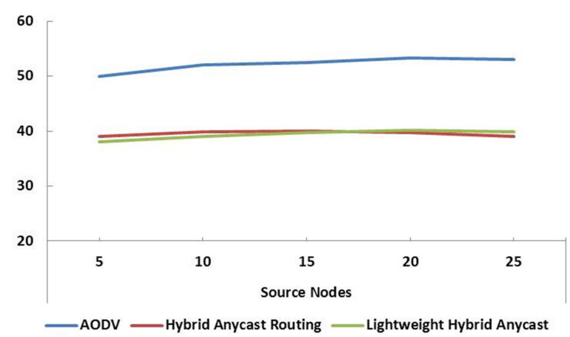
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.13: Delay in milliseconds (*ms*) for *'Set A'*

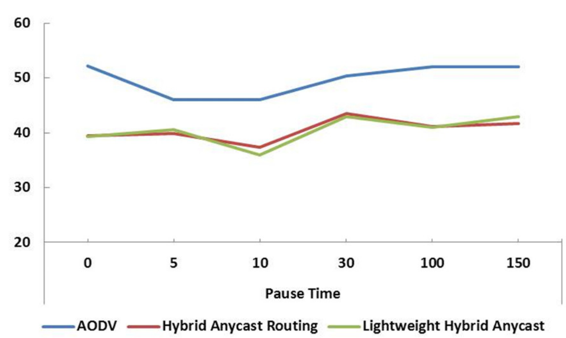
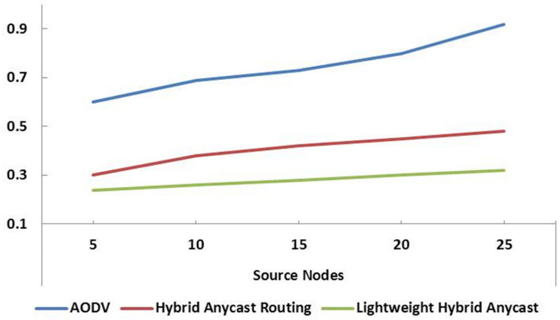
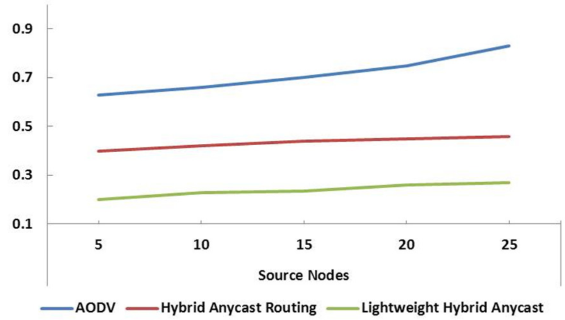


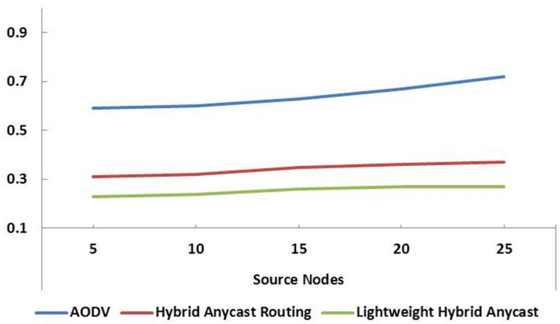
Figure 5.14: Delay measured in milliseconds (*ms*) against pause time of sinks in *'Set A'*



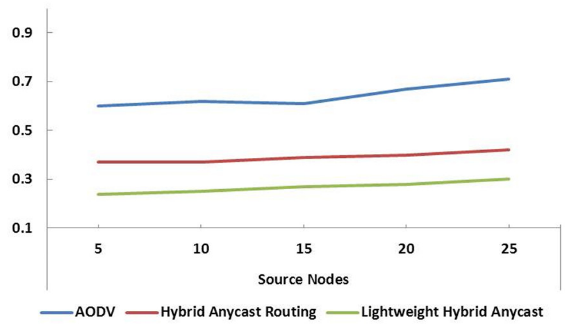
(a) Pause time 150sec



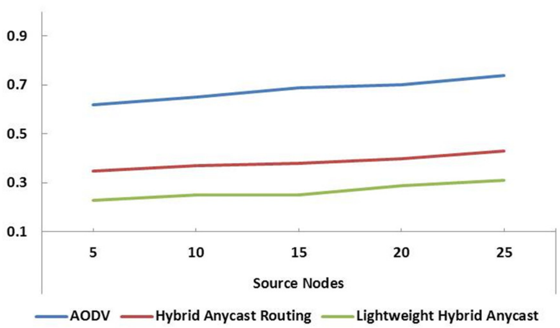
(b) Pause time 100 sec



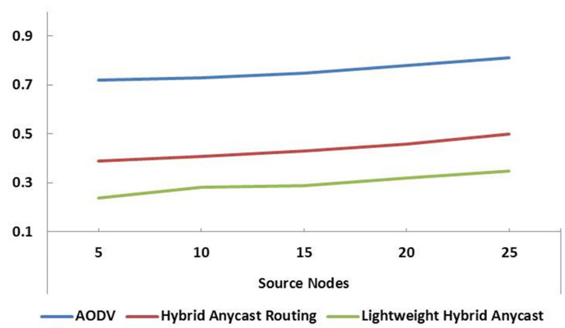
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.15: Control overhead in control bytes per data byte for *'Set A'*

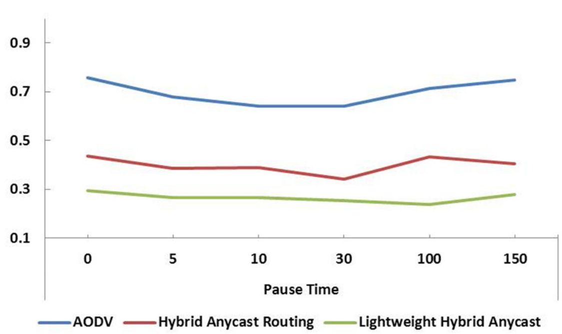
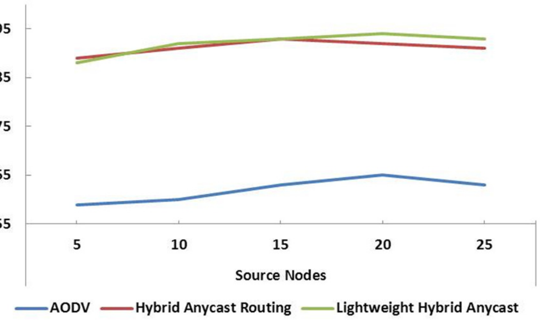
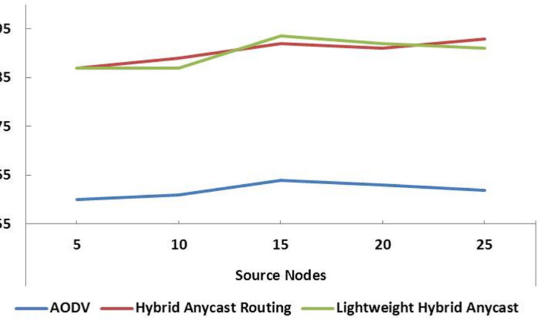


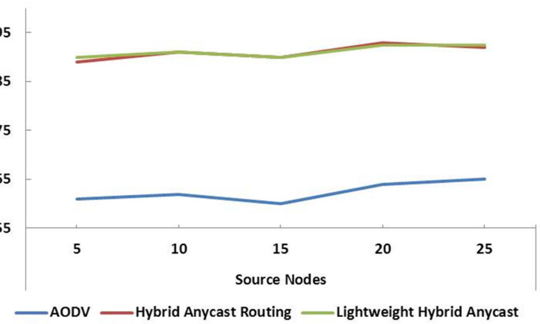
Figure 5.16: Control overhead against pause time for *'Set A'*



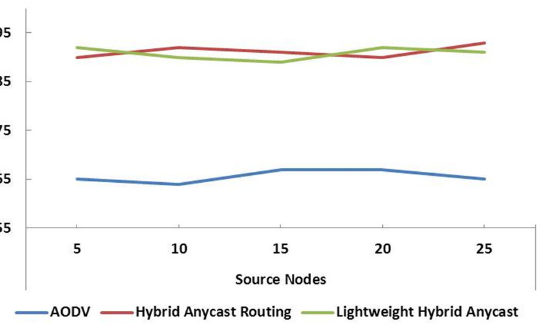
(a) Pause time 150sec



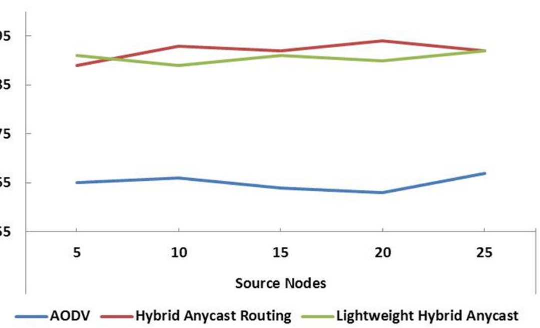
(b) Pause time 100 sec



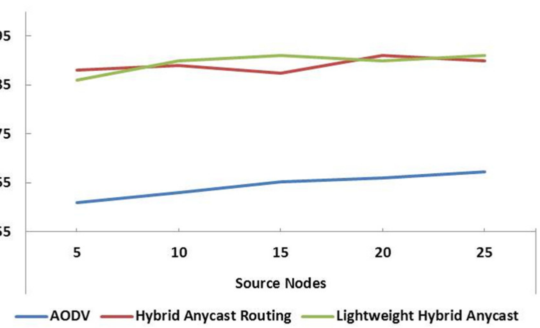
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.17: Delivery percentage for *'Set B'*

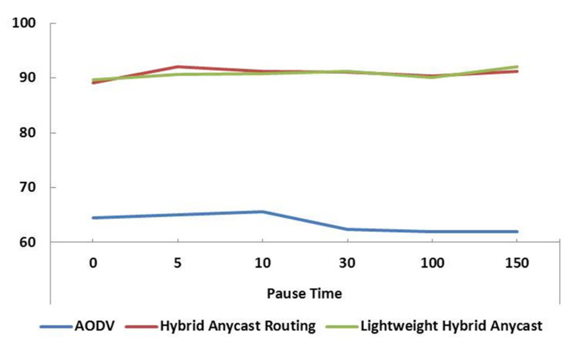
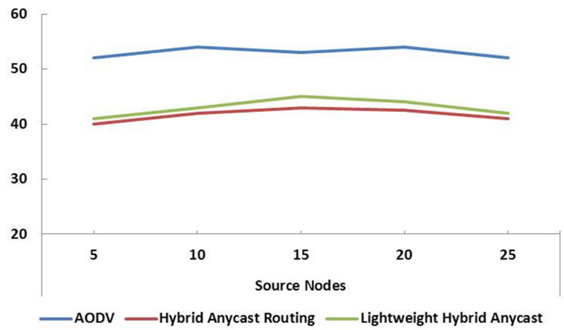
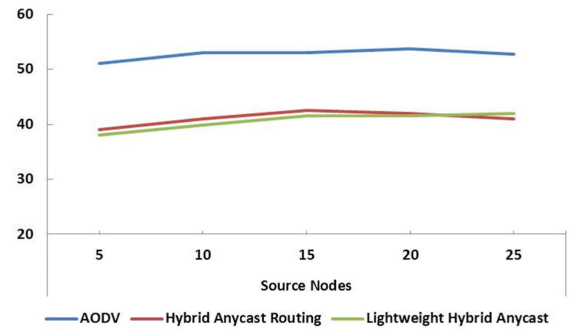


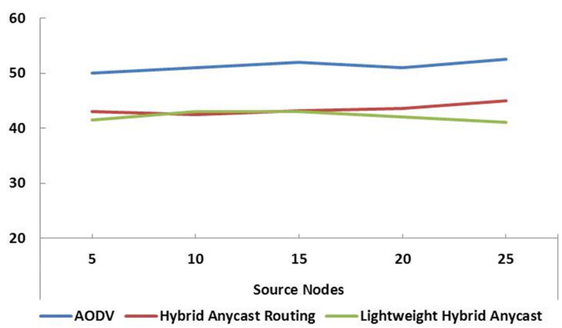
Figure 5.18: Delivery percentage against pause time of sink nodes in *'Set B'*



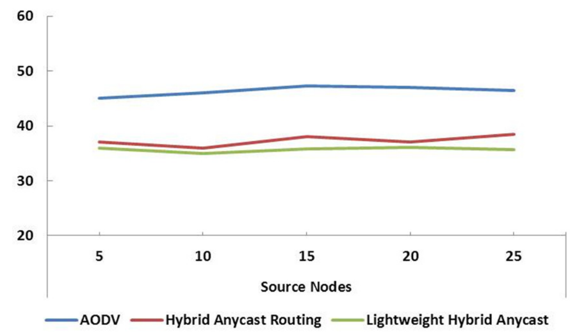
(a) Pause time 150sec



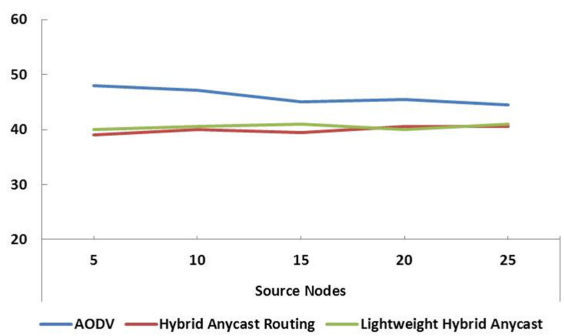
(b) Pause time 100 sec



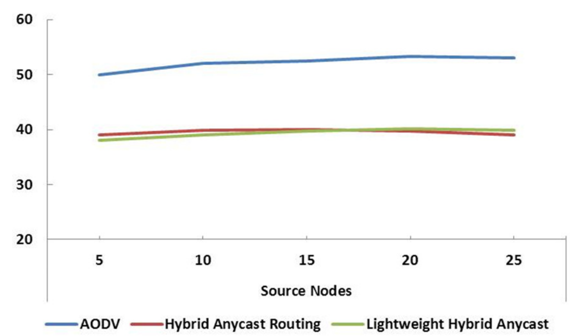
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.19: Delay in milliseconds (*ms*) for *'Set B'*

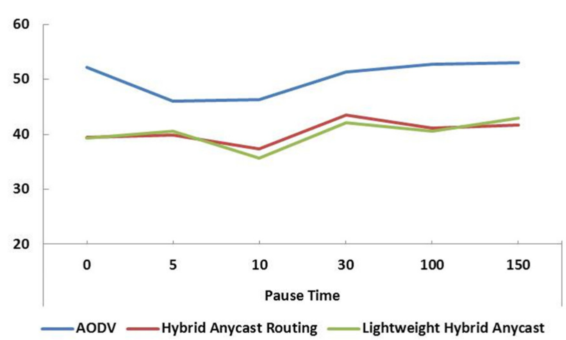
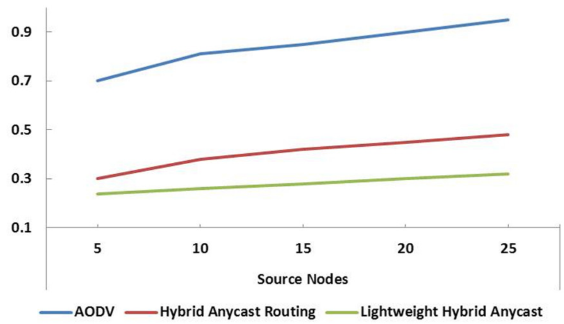
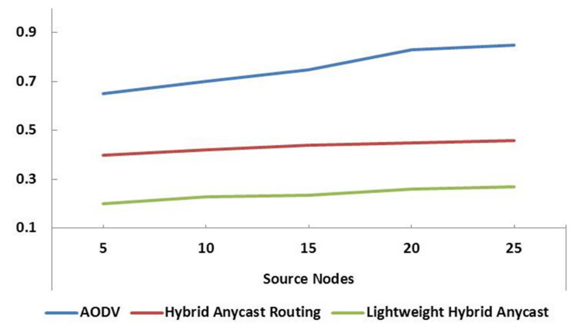


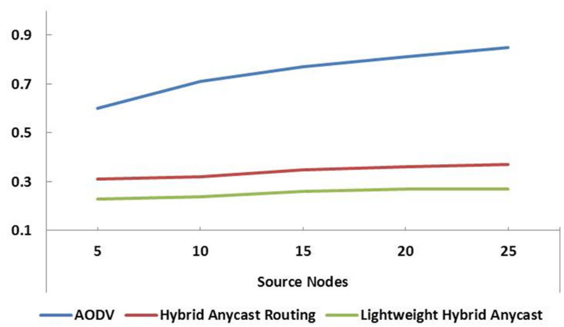
Figure 5.20: Delay measured in milliseconds (*ms*) against pause time of sinks in *'Set B'*



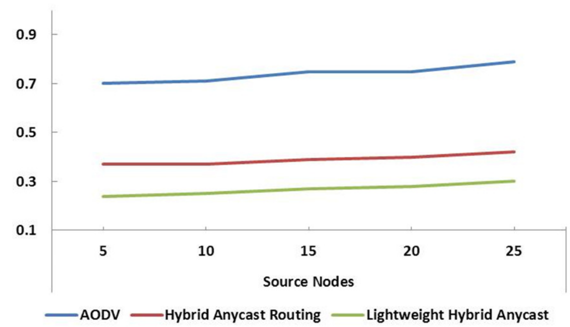
(a) Pause time 150sec



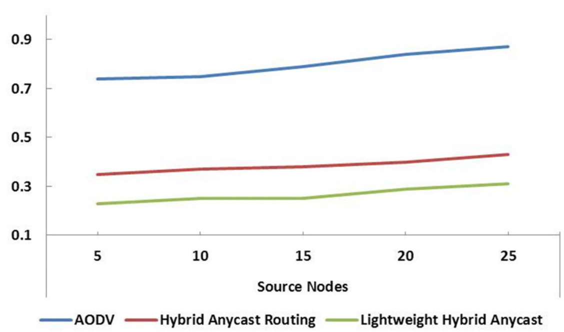
(b) Pause time 100 sec



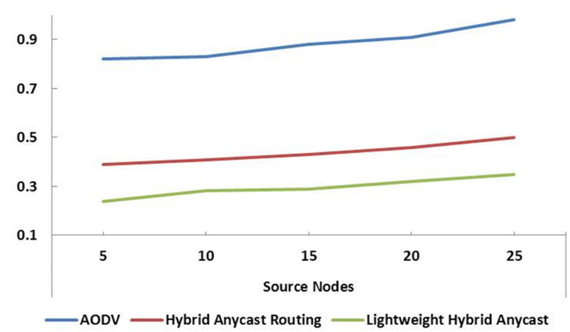
(c) Pause time 30 sec



(d) Pause time 10 sec



(e) Pause time 50 sec



(f) Pause time 0 sec

Figure 5.21: Control overhead in control bytes per data byte for *'Set B'*

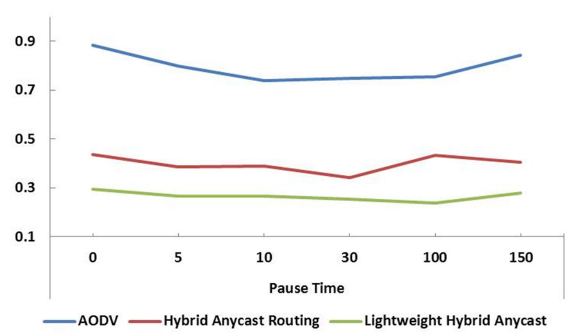


Figure 5.22: Control overhead against pause time for *'Set B'*

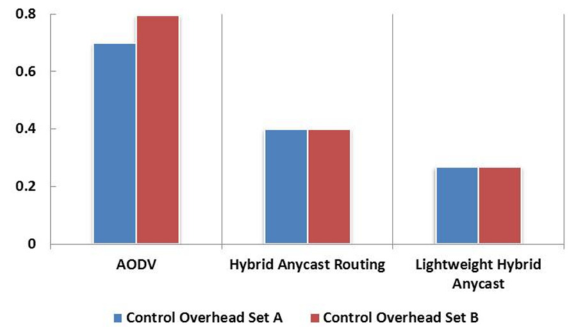


Figure 5.23: Control overhead comparison of *'Set A'* and *'Set B'*

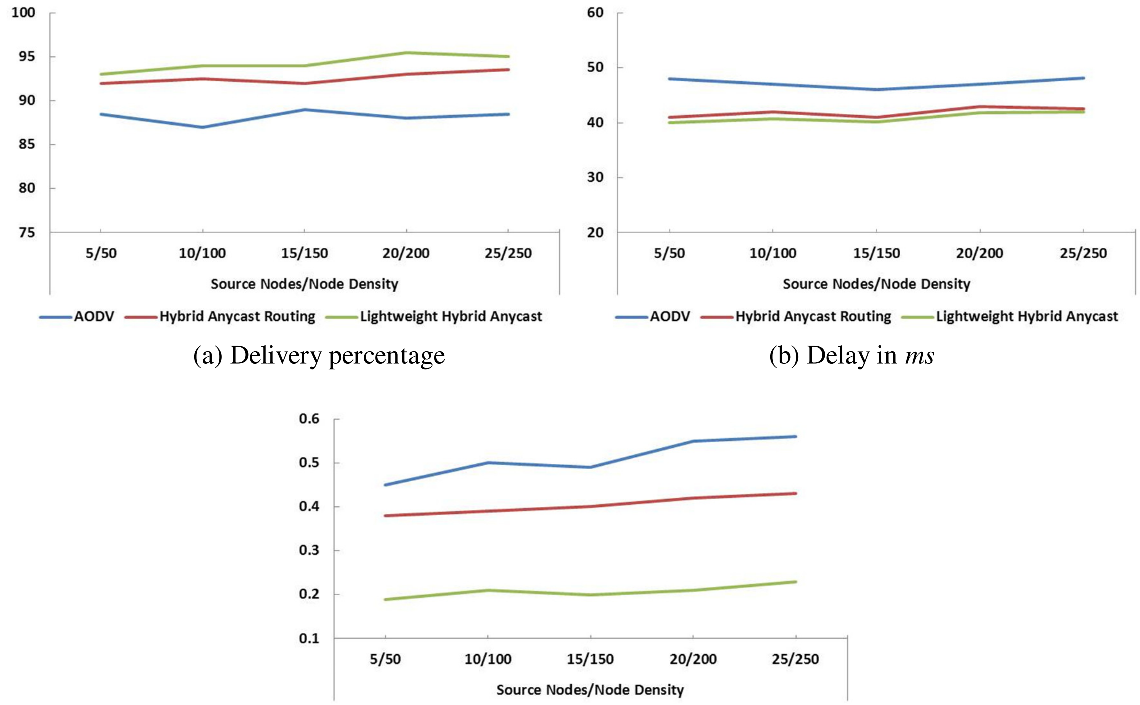
5.2.3 'Sink Proximity List' Evaluation

The purpose of this set of simulations is to evaluate the benefits of Sink Proximity List. As discussed in earlier sections, sensor nodes main Sink Proximity List, if the transmission range of sink node is larger than that of sensor nodes.

In the first simulation experiment we measure data delivery percentage, delay and control overhead, to establish that, this technique does not has any adverse effect on the routing protocol performance. The area use in this simulation is 2500 x 2000 meters. The node density in the area is changed from 50 to 250 nodes, with increments of 50. At the same time, the number of source nodes is also increased proportionally, i.e. 5 sources for 50 nodes, 10 sources for 100 nodes, 15 sources for 150 nodes, 20 sources for 200 nodes, and 25 sources for 250 nodes. All of the sensor nodes generate data at 4 packets per second, and their data generation time is randomly chosen by the simulator. There are 6 sink nodes present in each situation, and is moving using Random Way Point model with 10 second pause time. Total simulation time is set to 150 seconds, with confidence interval at 50 simulations.

Simulation Analysis: As seen in Figure 5.24 (a, b, c), the performance in terms of data delivery percentage, delay, and control overhead suffers from no negative effects.

Rather as a surprising result, the performance of delay and delivery percentage has increased. Although the increase is very minor and not statistically significant, it does lead to the conclusion that avoiding unnecessary expanding ring control overhead, may give way to data packets that may have been otherwise waited in the queue, which route was being discovered.



(a) Delivery percentage

(b) Delay in *ms*

(c) Control Overhead (control bytes per data byte)

Figure 5.24: Performance of '*Sink Proximity List*' technique

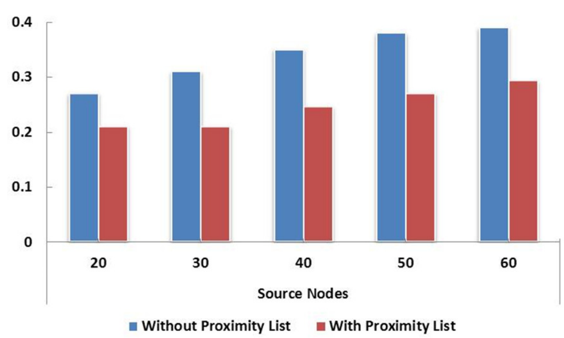


Figure 5.25: Control overhead comparison with and without *'Sink Proximity List'* technique, in Light-Weight Hybrid Anycast Routing

In the second simulation, we use similar parameters as the first one, but fix the total node density to 25, and change the number of sources from 20 to 60 in increments of 10 sources. Also, we compare the Light-Weight Hybrid Anycast routing protocol without 'Sink Proximity List' and with it. Thus this comparison only compares the control overhead generated with or without it.

Simulation Analysis: After running the simulation, we have shown the results in Figure 5.25. The blue bars show the control overhead, measured in control bytes sent per data byte delivered, without the use of 'Sink Proximity List'. The red lines show the performance with the use of proximity list. It is important to remember here, that with the use of proximity list, the source nodes do change the expanding ring search radius from 1 hop to 3 hops. As a result the nodes are quicker in finding the sink nodes, and do not need to send RREQ packets again while the ring expands. The figure clearly shows significant reduction in control overhead per data byte.

5.2.4 Sink Mobility Evaluation

In the beginning of this chapter we have discusses the assumptions of sensor networks. .In our work, we have taken examples of forest fire scenario, and argue that such a network for fire detection will be relatively static. The sensor nodes will not be mobile. Also the data collection points are sinks which can be static (strategically placed) or mobile covering the entire network area. Thus the route breakage will either be due to the movement of sink nodes, or destruction of sensor nodes due to fire. We individually simulate both these situations in this section and the next.

For this series of simulations, we observe the effect of sink mobility on the network performance. We measure the data delivery percentage and control overhead by changing the speed of the mobile sinks. In NS-2 the mobility is modeled using the Random Way Point mode. RWP uses two variables (maximum speed and pause time) to control how the nodes move. Every node that is allowed to move picks a random direction (destination point) and then randomly picks a *speed* (between 0.1 to max speed allowed in meter per second), and moves to the destination. At the destination point the node stops for a pre-defined *pause time* (measured in seconds). After the pause time it repeats the process of destination and speed selection. As a rule of thumb, if the pause time is zero, then the nodes move continuously with random speed (0.1 to max allowed), or, if the pause time is equal to the length of simulation then the mobile nodes are completely static.

5.2.4.1 Chaotic Movement Experiment

In this experiment we used two variables (Max speed and Pause Time) and change them to generate arbitrary movement patterns. These arbitrary movement patterns may or may not resemble realistic sink movement patterns, but will produce extreme conditions for our protocol to be tested on.

Hypothesis: *Establish the fact that under realistic or unrealistic movement patterns of sink nodes, our proposed protocols will perform better than other protocols.*

Essentially what we do is, to have high speed continuous movement to low speed with high pause time, and at the same time we have low speed continuous movement to high speed with high pause time. The graph in Figure 5.26 and Table 5.1 show the values that we use in this set of simulation. We evaluate the delivery percentage, and control overhead under these chaotic movement conditions.

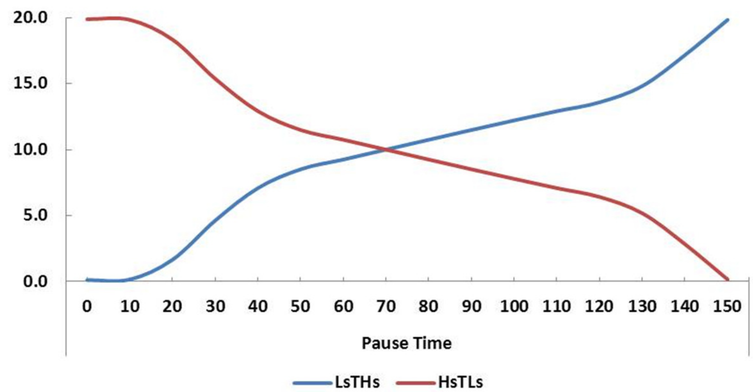


Figure 5.26: Maximum Speed (y-axis in meter per sec) and Pause Time combinations used in simulation

Table 5.1: Maximum Speed (*m/s*) and Pause Time (*seconds*) combinations used in simulation. Simulation Identifier is a marker used to identify pairs in the results.

Pause Time	$L_S T H_S$ Maximum Speed	$H_S T L_S$ Maximum Speed	Simulation Identifier
0	0.12	19.88	s1
10	0.16	19.84	s2
20	1.66	18.34	s3
30	4.63	15.37	s4
40	7.09	12.91	s5
50	8.51	11.49	s6
60	9.25	10.75	s7
70	10.00	10.00	s8
80	10.75	9.25	s9
90	11.49	8.51	s10
100	12.21	7.79	s11
110	12.91	7.09	s12
120	13.58	6.42	s13
130	14.82	5.18	s14
140	17.16	2.84	s15
150	19.84	0.16	s16

There are two lines showing two different value pairs for each pause times. The pause times start from 0 sec and go till 150 sec (total simulation time), with increments of 10 seconds. The Low speed to High speed ($L_S T H_S$) values start from 0.1 m/s and go till the 20 m/s, which is the maximum speed allowed in the whole simulation. The High speed to Low speed ($H_S T L_S$) values are the reverse of previous set. Using this scheme we are able to evaluate the performance at different pairs. These value pairs (Maximum Speed & Pause Time) are generated using a modified Sigmoid Function equation, as shown below.

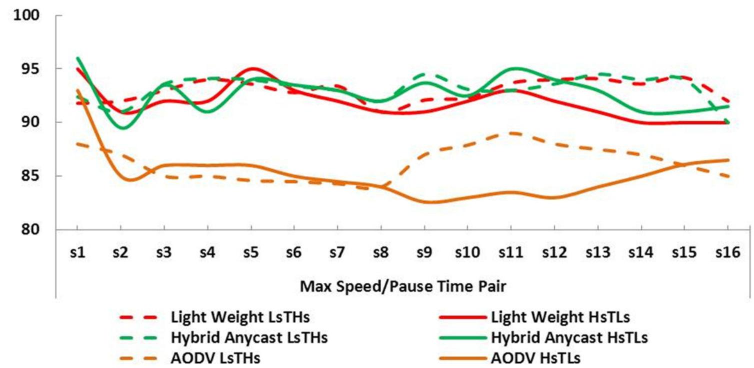
$$f(x) = \alpha \left[\frac{1}{1 + e^{\beta x}}\right] \quad (5.1)$$

where α is the maximum speed allowed in the simulation, β is the smoothing factor, and *x* belongs to the Control set, γ; where $\gamma \in \mathbb{R}$. Control set is defined as a collection of values corresponding to the pause times. This set defines how the function will grow (exponentially, linear, cubic, etc.). In our model the Control set is as follows:

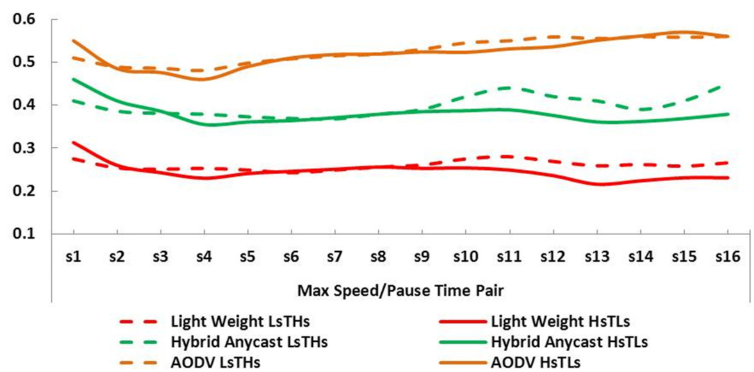
$$\gamma = \{17, 16, 8, 4, 2, 1, 0.5, 0, -0.5, -1, -1.5, -2, -2.5, -3.5, -6, -16\} \quad (5.2)$$

The rest of simulation set up similar to the previously conducted simulations in this chapter. The area used is 2500 x 2000 meters. The node density is 250, with uniform random distribution. There are 25 source nodes randomly placed in the network, with data generation rate at 4 packets per second, and their data generation time is randomly chosen by the simulator. There are 6 sink nodes present in each situation, and is moving using Random Way Point model with the Max speed and pause time pares as shown in Table 5.1. Total simulation time is set to 150 seconds, with confidence interval at 50 simulations.

Simulation Analysis: The results of the simulations are presented in the Figure 5.27 (a, b). The simulation identifier is used to represent the data pairs as show in Table 5.1. We can observe from the data delivery percentage that depending on the movement pairs, the resulting delivery rate does not produce any meaningful pattern. It seems as it randomly goes up and down, but it proves our hypothesis that under any of the circumstances, the Hybrid Anycast Routing protocol and the Light-Weight version for sensor networks, perform equally well. Their performance is superior to that of AODV in all the combinations. Similarly in Figure 5.27b the control overhead of Lightweight Hybrid Anycast protocol is lesser than the competing protocols.



(a) Delivery Percentage

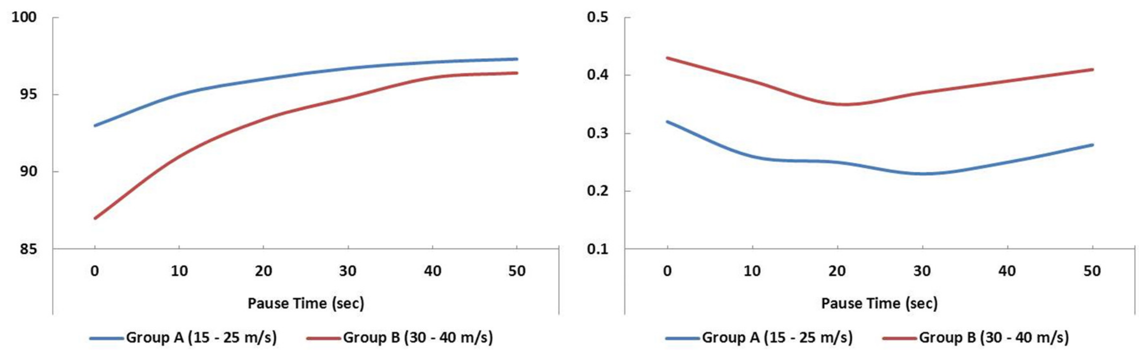


(b) Control Overhead (control bytes per data byte)

Figure 5.27: Performance results of Chaotic Movement Experiment

5.2.4.2 Realistic Mobility Variation Experiment

This is a more realistic experiment that takes proof from the previous chaotic experimentation. As proved earlier that even under unrealistic models our protocol outperforms the competition, thus we only evaluate the performance or Lightweight Hybrid Anycast Protocol for Sensor Networks in this experiment. Here, we do not use the full range of pause times, but select a sub set between 0 and 30 sec, which is more meaningful in comparison to real world. Also we limit the node speed between a maximum and minimum number. We have two distinct speed groups: Group A picks random speed with uniform distribution between 30 to 40 meter per second, Group B picks a random speed with uniform distribution between 15 to 25 meter per second. Each group is paired with 0, 10, 20, and 30 second pause time intervals.



(a) Delivery Percentage (b) Control Overhead

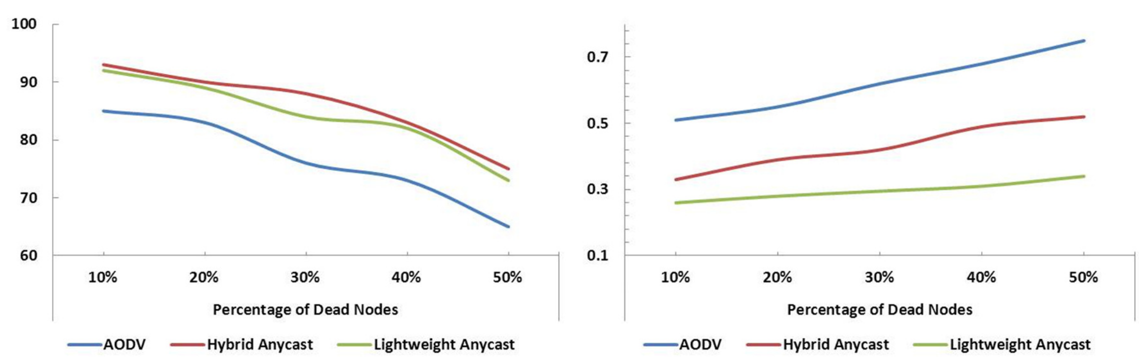
Figure 5.28: Light-Weight Anycast Routing Protocol performance results of mobility experiments. Group A is between 15 to 25 m/s & Group B is between 30 to 40 m/s.

Simulation Analysis: Figure 5.28 (a, b) show some interesting observations for Lightweight protocol's delivery percentage and control overhead. We observed from initial experimentation of basic protocol operation, that the average delivery percentage for different pause times at 15 m/s maximum speed, gave a variating result. Here with more refined parameters for mobility, the pattern that emerges suggests that at high mobility the data rates drop, but as the nodes slow down the quickly recover to higher levels. Corresponding to that the control overhead is more at higher mobility, but gets lower with relatively lower with slower movement patterns. An interesting experiment would be to use more complex mobility algorithms, e.g. Boundless Area Mobility, Gauss-Markov Mobility, or Proba Walk Mobility models, etc.

5.2.5 Evaluation using Dead Nodes & Dead Zones in Environment

A major reason for link breakages in sensor networks is due to sensor destruction, or terrain features which cause communication breakdown. There are two ways to approach such models in simulation.

Dead Node Analysis: A percentage of sensor nodes are programmed to randomly go down and come back online. This will cause the routes to break and new routes will be required. In our simulation of such a model, a percentage of nodes are programmed to randomly turn their power off and then back on. This percentage varies from 10% to 50%. The nodes that are generating the data packets are not allowed to be part of these dead nodes. The area used is 2500 x 2000 meters, with node density of 250, with uniform random distribution. There are 25 source nodes randomly placed in the network, with data generation rate at 4 packets per second, and their data generation time is randomly chosen by the simulator. There are 6 sink nodes present in each situation, and is moving using Random Way Point model with the maximum speed of 20 m/s and pause time of 15 seconds. Total simulation time is set to 150 seconds, with confidence interval at 50 simulations.



(a) Delivery Percentage (b) Control Overhead

Figure 5.29: Dead node analysis for Light-Weight Hybrid Anycast Routing protocol

Figure 5.29 shows the delivery percentage when the percentage of dead nodes increases. We can clearly observe a decline in delivery rate. But the important thing to note is how much control overhead is required to maintain the given rate. Figure 5.29 (b) shows the increase in control bytes required per successful data byte delivered, and we can see that the overhead of our proposed schemes does not rise as sharply as AODV does. Although the rise is expected, but its statistical significance is important. Comparing these results to the chaotic experiment we can clearly see a more meaningful pattern in the results, which suggest that the increase in dead nodes will decrease the delivery rate significantly in all the protocols, but our protocol maintains the control overhead that incurs in finding new routes, thus conserving energy and bandwidth for data usage.

Dead Zone Analysis: A randomly sized geographic area in the whole network that goes dark (i.e. all nodes in that area stop communicating). This area is randomly located, with a pre-configured maximum size. Moreover these areas can be square or circular in shape. In our simulations that follow, we use both shapes randomly. This simulates the effect of a sudden forest fire, or other terrain anomalies that prohibit communication. Figure 5.30 shows a sample of dead zone creation in the network.

In our simulation experiment we have limited the maximum height or width of a rectangular dead zone to 150 meters, and the radius to 75 meters for circular. Moreover, we make sure that there are not more than 3 dead zones at any given point in the network. It is possible in such a scenario that a source node falls inside a dead zone, and thus stops generation of data. In order to maintain the number of source nodes constant we have programed the simulator to randomly select another node that is not part of any dead zone, to act as a source node. There are 25 source nodes randomly selected in the whole network, at any given time, transmitting at 4 packets per second of data. Unlike sensor nodes, if a sink node happens to be in a dead zone it temporarily switches itself off, till it has moved out of the dead zone. Using these parameters, we have evaluated the performance of our proposed protocol, as show in Figure 5.31. Before analyzing the results, theoretically, we expect the performance to be much worse than that of the dead nodes experiment, because, whole blocks of network will go down randomly, thus cutting off communication at larger scales.

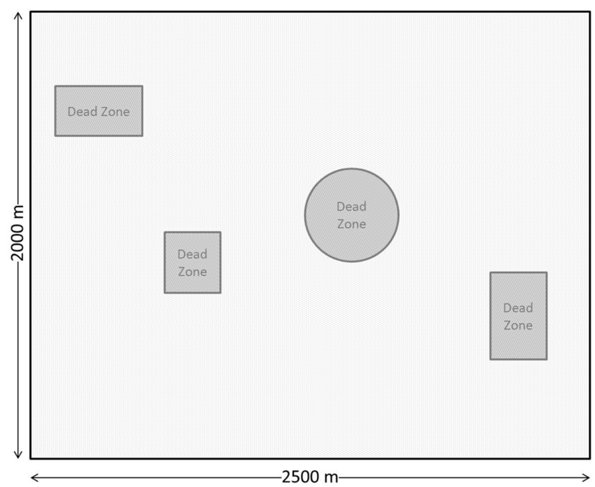
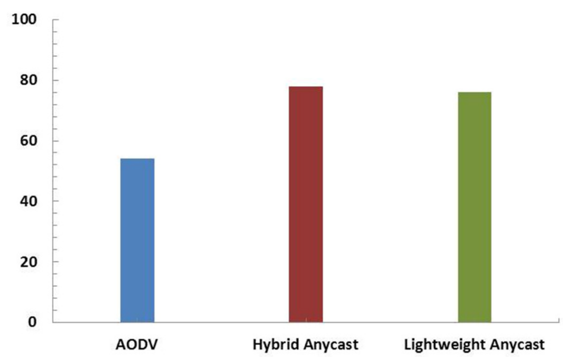


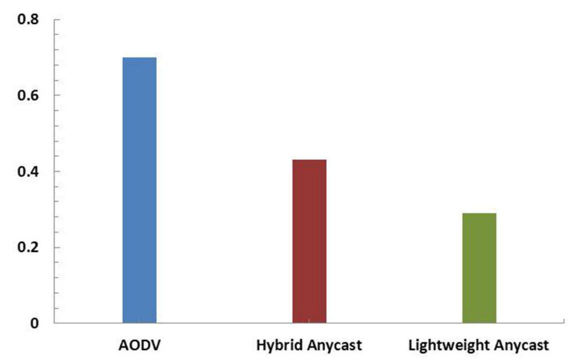
Figure 5.30: Random dead zones in the simulation environment

The results in Figure 5.31 (a, b) show the percentage delivery and control overhead. The pattern is not surprising, as we expected our proposed protocols to outperform AODV. The interesting result is that Lightweight protocol is still able to compete with its more complex and capable version of Hybrid Anycast Routing protocol, proposed earlier. The control overhead shows the bigger advantage of Lightweight version, as it keeps the control overhead below all the other protocols. In Figure 5.21 (c & d), we have transposed the same results on top of the results obtained in the *Dead*

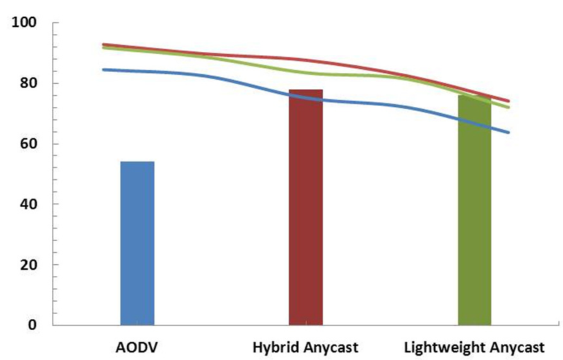
Nodes experiment. This gives a clearer picture how the protocols behaved in different evaluation techniques. We observe that AODV's performance degraded to a greater extent when more realistic *Dead Zone* technique was used. Hybrid Anycast and Lightweight Hybrid Anycast also observed lower delivery performance (which was found to be statistically significant) but the delta change between AODV's performance was greater than that of Hybrid Anycast suite. Control overhead performance also experiences a similar pattern, with Lightweight Hybrid Anycast Routing Protocol out performs significantly.

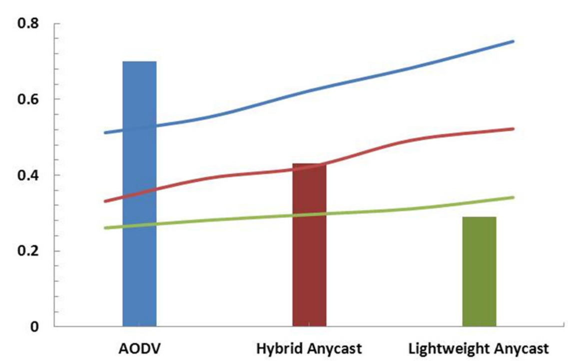


(a) Delivery percentage



(b) Control Overhead





(c) Delivery Percentage & Control overhead, super imposed by results of Dead Node analysis

Figure 5.31: Performance of Dead Zone simulation

CHAPTER 6: EVENT BASED AND REALISTIC SCENARIO SIMULATION IN NS-2

In sensor networks the reason for data origination can be driven by two processes, i.e. Query driven, or Event driven. Considering the example of a forest based sensor network, the purpose of a sensor network is to report changes in temperature (and related environment variable), so that a fire can be detected. The sensor will only generate and report such data if there is a sudden and significant change in observed temperature, or if the control system explicitly request for the temperature readings. In literature there have been numerous algorithms proposed for approximation of data at the control center, in order to minimize the unnecessary reporting of data. In our research we do not evaluate the approximation algorithms. Our communication model is capable of incorporating any approximation technique as a separate module. In this chapter we address a more fundamental question of how to actually simulate and evaluate the data generation of nodes.

Simulation of query based systems is easy and close to reality, and the sink nodes can be programmed request data from specific sensors (or group of sensors) at randomly selected intervals. This will cause a RREQ to be initiated from the sink to the sensor and paths can be maintained until required. On the other hand, an event based system implemented by programming nodes to randomly generate data is not as realistic as it should be. Although this is affective in evaluating the working and performance of algorithm itself, but it does not reflect the performance of the protocol in real world. We discuss the case of event based data generation in the next section, and describe models that can be used to more effectively simulate realistic approaches.

6.1 Event Based Data Generation

Consider a sensor network deployed in a forest to detect sudden changes in temperature, the sensor nodes will generate data when they are close to such an event location. We can assume that sudden eruption of fire will create dead zones in the network, but the nodes that report that data have to be in close proximity of the dead zone. Although, simulation of dead zones in the previous chapter demonstrated the capability of routing protocol to rebuild its data paths, it did not accurately mimic the practical situation. In a real situation, the dead zone will exist very close to the source node, thus creating a communication black out in close vicinity. Such situations are more complex and challenging for communication, and the performance of protocols evaluated using simplistic approaches cannot be mapped to them.

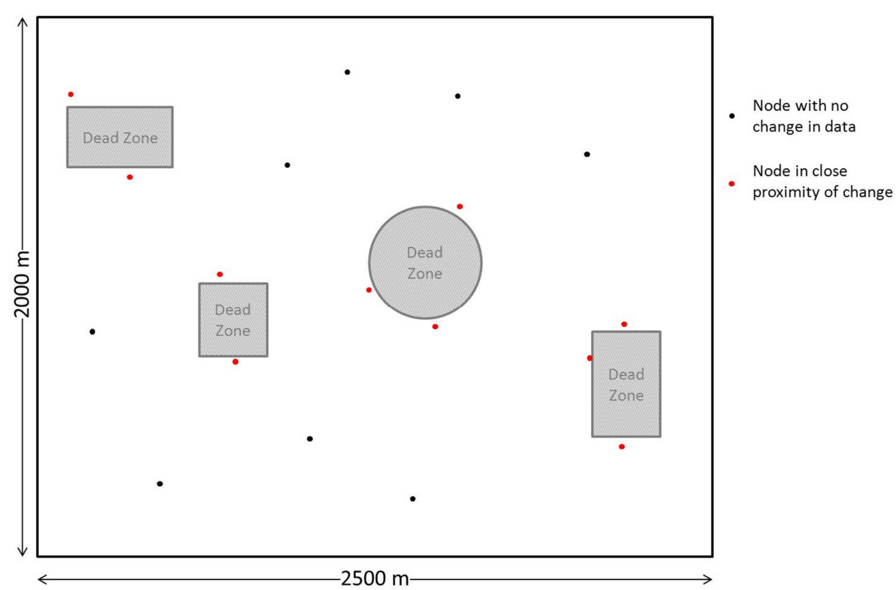


Figure 6.1: Nodes in close proximity of dead zone will generate data (red color). Far away nodes (black color) do not have significant change to report.

As shown in the Figure 6.1, the red nodes are closer to the dead zone (or a fire source), and will be the ones that will detect a change in temperature, which needs to be reports. Nodes that are far away from dead zones will have nothing to report as they will not detect any change. They may choose to follow an approximation algorithm or stay silent till an event occurs near them. Thus, using a random algorithm to select nodes to transmit data is not as realistic as it can be, although it is a good way of testing the protocol functionality. Furthermore as we can see that the dead zones create a wall of communication black out in certain directions for the red nodes. This will have a tremendous impact on the working and efficiency of route acquisition and maintenance.

Real World Forest Fire Spreading: In Figure 6.2 we show thermal images of real forest fires. It can be seen that the fires do not create dead zone in rectangular or circular shapes. There are dozens of variable that affect the spread of fire. Starting from a single point the fires usually spreads in an expanding form, based on the density, dryness, wind, temperature, topography, terrain, etc. The behavior of spreading is very complex and cannot be simulated using simple rectangles and circles. Modeling real forest fires is a challenging task. Just like predicting weather, there are hundreds of variables that affect the movement patters. More over the gathering these variables in a highly dynamic environment is a challenge itself.

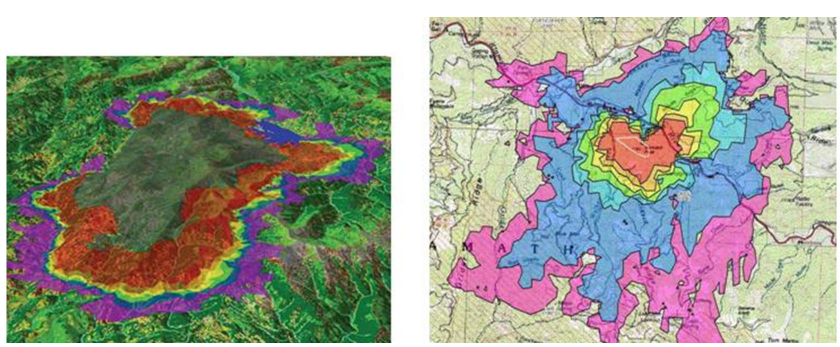


Figure 6.2: Images of real forest fires

6.2 Capabilities of Network Simulation Software

Network simulators are usually designed to simulate or emulate the working of network protocols. Most of these protocols work at physical, data link, network, and transport layer. The technique used is to model the behavior of different network elements using mathematical functions, or by capturing data from real networks and then using it to model the working of other protocols. In either case, the fundamental purpose is to create interactions between the communicating nodes. As the network systems have evolved into more mobile architectures, thus the simulation software have added the mobility component in their mathematical functions. The current breed of simulation software can (to a good extent) model the electromagnetic communication at the physical layer. This has opened the simulation software to incorporate other routing and transport layer protocols, not only for evaluation, but for planning and analysis.

Most of the simulators incorporate randomness in their communication models (either it be mobility or events), to mimic the unpredictability of the real world. This technique helps in a great extent in evaluating protocols with uncertain input conditions.

6.3 Challenges and Motivation

Simulating reality is an almost impossible task. Although this has improved the simulators, and expanded their usability, they cannot mimic the reality. Even at the physical layer of communication models, there are numerous variables (e.g. topography, wavelength bending, urban landscape, and other physical elements), that cannot be simulated realistically. Firstly the availability of such data is critical, and then how to implement those details in a network simulator is a more important question.

The biggest challenge with mobile communication is network simulators, was to simulate the movement behaviors or nodes. This has been solved to a great extent by using the Random Way Point or Random Walk models. By picking the direction, speed and stop times randomly, and iterating the simulation to increase the confidence interval, the results can be closely matched to real movement of people carrying mobile devices. This technique has made the generic mobility problem simple, but when it comes to more complex and networks designed for specific applications, then using random behavior gets strictly limited. In our example of forest fire, the mobility of sink nodes can be modeled using a Random Way Point (with increased confidence interval), but simulating the spread of fire (resulting the death of sensor nodes) cannot be modeled randomly. As shown in the simulation results of the Figure 5.29 and Figure 5.31, there is a huge difference as we move closer to the reality.

Network simulation systems and NS-2 in particular do not have the capability to simulate such realistic applications of sensor or wireless models. As proved before randomness alone cannot solve this problem either. Thus it is very important to have models that can help in simulating realistic applications. At the same time, as the primary objective of network simulators is to model protocols, the realistic application models have to be simple and un-complex (if possible) to keep the overall complexity of the simulators to the minimum.

In our research, heavy emphasis is laid on the forest fire and battle field sensor networks. We have made an effort to propose simple yet affective models for simulating the forest fire in NS-2. The major challenge is to move from simple dead nodes and rectangular dead zones towards are more realistic proximity model, where the nodes closer to the fire generate data. Then expand these models to randomly shaped dead zones, and then models that can simulate a growing dead zone mimicking forest fire.

6.4 Basic Proximity Based Models

We have developed models for realistic simulation based on fundamental geometric concepts. The objective is to keep them as simple and as close to reality as possible, so they can be replicated in majority of the simulation software with little effort. The basic proximity models use simple structures like rectangles and circles to initiate a dead zone, and then define a proximity to that area where, if any nodes present, will generate the data (or in other words, report the sudden change in temperature). We have also shown the basic equations that can be used to determine the location of nodes inside the dead zone and the proximity zones of both circles and rectangles. In the later part of this section we discuss a new concept of extended circular models that can be used to model irregular shaped dead zones and proximity area around them.

6.4.1 Basic Rectangular Proximity Model

The easiest way to create a dead zone in simulators is a rectangular shaped are, because most of the topography in the simulator is based on a 2D XY co-ordinate system. This measuring the sides and positions of nodes becomes very easy. We exploit the same concept of rectangles as we used in the previous chapter to create rectangular dead zones. In addition to the creation of a dead zone, another zone is created around the rectangle, called ‘*proximity zone*’. Figure 6.3 shows the dead zone and the proximity zones.

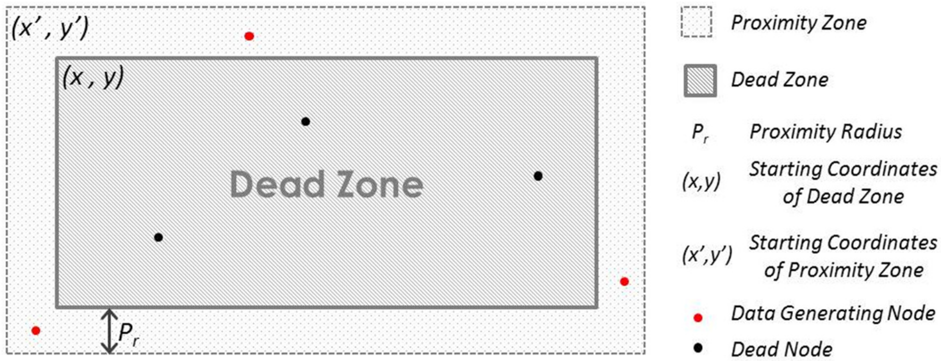


Figure 6.3: Dead & Proximity Zone in Rectangular Proximity Model

Nodes inside the dead zone will immediately disable their communication, and rather that random nodes generate data, the nodes inside the *proximity zone* immediately start generating data. Thus we remove the random generation of data, and replace with more realistic even based data generation algorithm. To keep the implementation and calculations simple, we use variables as defined below to control the model.

Table 6.1: Rectangular Proximity Model Variables

Symbol	Description
x , y	Top left coordinates of Dead zone
x' , y'	Top left coordinates of Proximity Zone
P_r	Proximity radius
l_{max}	Maximum *length* a Dead zone can have
h_{max}	Maximum *height* a Dead zone can have
l	Random length of Dead zone
h	Random length of Dead zone

The algorithm to create dead and proximity zones around them is fairly straight forward with the use of these variables. The steps are described below:

- *Dead Zone*: To create a dead zone, randomly picks the x, y coordinates of the starting point of the zone. These coordinates, must of course, lie inside the total simulation area.

- The length l and height h of the dead zone is again randomly selected where; $l \leq l_{max}$ and $h \leq h_{max}$.
- The rectangular dead zone can then be created with the four vertices coordinates as: (x , y) $(x+l , y)$ $(x , y+h)$ $(x+l , y+h)$.
- *Dead Nodes*: The simulator then can mark all the nodes that lie inside the dead zone as dead, by either turning them off or disabling their communication. Finding nodes in the dead zone can be determined by their coordinates (x_n , y_n), if they satisfy the following conditions:

$$x_n \geq x \quad and \quad x_n \leq x+l$$

$$y_n \geq y \quad and \quad y_n \leq y+h$$

- *Proximity Zone*: Proximity zone is established outside the dead zone, with a certain radius P_r. The radius is a global variable, and is determined by the node sensing capabilities. To determine the origination point of proximity rectangle, (x' , y') can be found as:

$$x' = x - P_r \quad \& \quad y' = y - P_r$$

- Using (x' , y') the proximity area can be drawn as a rectangle with the vertices as: (x' , y') $(x'+2P_r+l , y')$ $(x' , y'+2P\text{r}+h)$ $(x'+2P_r+l , y+2P\text{r}+h)$.
- *Proximity Nodes*: All nodes that fall outside the dead zone and inside the proximity zones can be easily identified by, a) creating four rectangles outside the dead zone and determining their location in them, or b) determining all the nodes inside the proximity rectangle and marking the nodes as proximity nodes which are not already marked dead.

Using this model, simple rectangular regions can be generated with nodes inside certain proximity to generate data. The model is simple enough to be implemented in almost all simulators that use 2D coordinate based topographic map for sensor nodes.

6.4.2 Circular Proximity Model & Extended Circles

As most of the real world dead zones would be less symmetric in nature and more randomly shaped, thus having rectangles may not be the optimal solution. Although it may be simple to implement rectangles, and some simulators designed for certain application may benefit from that model. The circular proximity model is very similar to rectangular model with modifications to how the proximity area is determined. The variables used in this model are given in Table 6.2.

Table 6.2: Circular Proximity Model Variables

Symbol	Description
x , *y*	Center point of Dead zone and Proximity Zone
R	Radius if Dead Zone
R'	Radius of Proximity Zone $(R + P_d)$
P_d	Proximity distance
R_{max}	Maximum *radius* a Dead zone can have

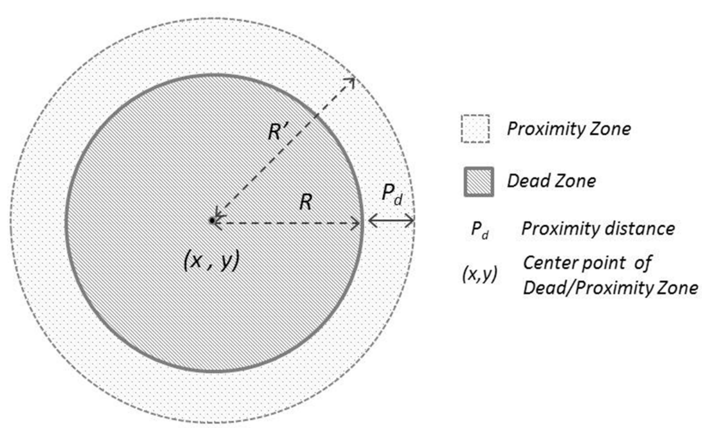


Figure 6.4: Dead & Proximity Zones in Circular Proximity Model

The algorithm to generate this model in a simulator is very simple and is based on the equations for a circle. The objective is to randomly generate a circle centered at (x , y) inside the simulation area. The radius R of the circle is randomly picked and is always less than R_{max}. The nodes inside the circle are then pronounced dead, and can be found using the following condition:

$$(x - x_n)^2 + (y - y_n)^2 \leq R^2$$

where (x_n , y_n) are the coordinates of the subject node. We propose that every node in the system should be checked against this condition. In cases where the number of nodes in the network is too large and time consumption in matching is considered an issue, other mechanisms can be deployed to limit the number of nodes that are checked, buy dividing the network are into a smaller square and checking the nodes that are in that area only. Once the nodes are marked dead, the algorithm can proceed to build the proximity zone, by establishing a circular region centered at (x , y). The radius of this circle is R' where:

$$R' = R + P_d \tag{6.1}$$

The difference P_d between dead zone and the new circle boundaries is declared as the proximity zone, as show in Figure 6.4. Nodes inside the proximity zone can be determined by the same formula of distance:

$$(x - x_n)^2 + (y - y_n)^2 \leq (R')^2 \tag{6.2}$$

Nodes that satisfy this condition, but are not already marked dead, are considered in proximity zone, and will thus generate data.

Extended Circles Proximity Model: This is a specialized version of the Circular Proximity model, and enables the simulators to model arbitrary shaped dead zones and corresponding proximity areas. The technique is very simple, and uses the same working of circular model. A random shaped (with no vertices) area can be achieved by overlapping circles, as show in Figure 6.5.

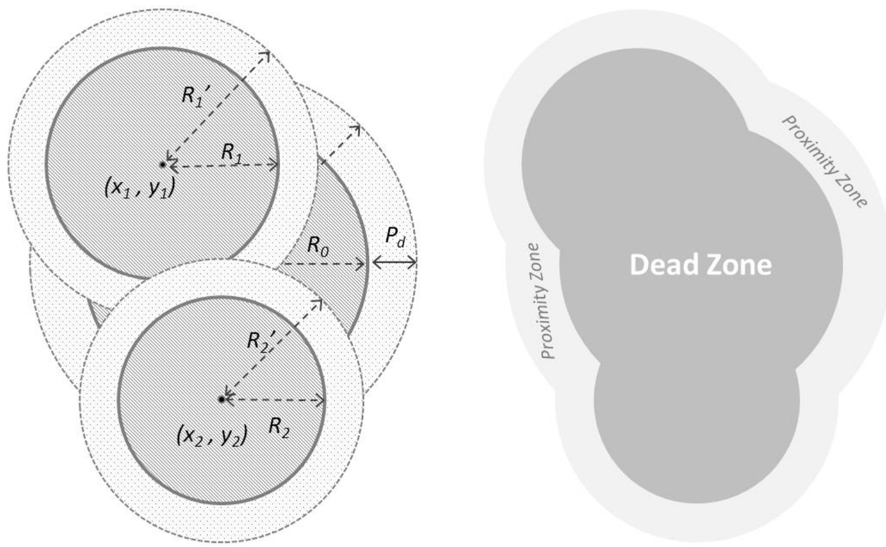


Figure 6.5: Extended Circles Proximity Model

The overlapping circles once viewed as just dead and proximity zones (Figure 6.5 right), gives a clear view of the random area that can be generated. The algorithm follows the following steps (as described previously also) for every circle it generates:

1. Create a Dead zone circle with Radius *R*.
2. Mark the nodes inside the circle as dead.
3. Create a Proximity circle dictated by the P_d.
4. Park the nodes inside the proximity zone as senders of data.

An important point to note here the subsequent circles should be centered. There can be two cases, and neither of them can be better or worse. 1) All the subsequent circles are centered in the original circle. 2) The subsequent circles are centered in the last created circle. There can also be other combinations of these, but the resulting shape will still be a random area. To find the random center point (x_n , y_n) of the new circle in base circle we use the following equation:

$$x_n = x_0 + r * Cos(t) \tag{6.3}$$

$$y_n = y_0 + r * Sin(t) \tag{6.4}$$

where,

$$r = \sqrt{rand(\,)}$$

$$t = 2 \times \pi \times rand(\,)$$

This will result in a uniform random distribution of points inside the base circle, and subsequent circle can be originated from these points.

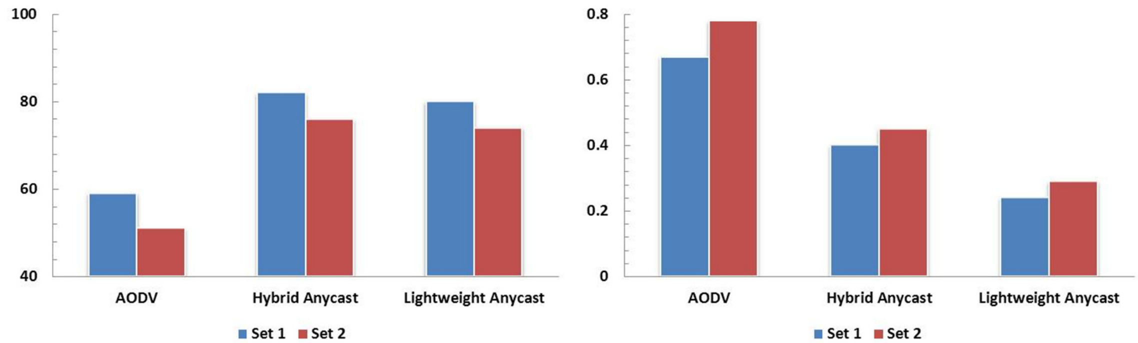
6.4.3 Simulation Analysis using Extended Circles

To evaluate out proposed models we have used NS-2, so that we can compare the change in dead zone modeling to the previous experiments we have conducted. In this set of simulations, we only use Extended Circles Proximity Model. The total simulation area is set to 2500 x 2000 meters, with node density of 250. The nodes are spread using a random uniform distribution. Total simulation time is 150 sec. Number of source nodes is random, and is based on the nodes falling in the proximity zones. Once a node falls into the proximity zone, it starts to generate data at 4 packets per second. We run two different sets of simulation.

Set 1: A single set of Extended Circles Proximity model is used, which is limited to 3 circles, originating from the base circle.

Set 2: There are two sets of Extended Circles Proximity model in the environment with randomly selected initial center points. One set is allowed to extend to 5 circles, and the other one is restricted to 3 circles only.

R_{max} is set to 75 meters, and P_d is 25 meters for both. We measure the performance of our proposed protocols and AODV for Data delivery percentage and Control overhead.



(a) Delivery percentage (b) Control Overhead

Figure 6.6: Extended Circles Proximity model performance evaluation

The simulation results for both the sets of simulation are shown in Figure 6.6. We observe that there is significant change once the number extended circles is increased, and this is obvious. The interesting result is that fact that these values are significantly different than the results we observed with simple dead zone simulation, even though the simulation parameters are approximately similar. Thus establishing the fact that simply having dead zones may not give accurate results, and better techniques can be used for simulating realistic environments. We have designed very basic models, in order to keep the implementation simple and implementable on majority of the network simulators. It is very much possible to use more complex geometric forms to shape the dead zones. The challenge in such algorithms would be to determine, if nodes are inside the area or not. It is not impossible, but will require more complex formulas.

6.5 Forest Fire Spreading Model

We have discussed earlier in this chapter that simulating the movement of fire spreading in the forest is a very complex process. Incorporating it in network simulators will not only make them complex, but it may even be impossible do to the internal architecture of the simulation systems. In this section we describe simpler models and algorithms that can be used to mimic the spread, but at the same time are not complex than the models defined in the previous section.

The shape of forest fires mainly dictated by the wind and fuel available to it. Although there are many other variables that affect it but it can be simplified as, starting from a single point it spreads with a certain speed and direction. During the course of its total time, it may change its speed or direction. Our algorithm uses a chain of linked dead zones (rectangular or circular) by using the following three properties:

Direction: This property (preprogrammed or dynamic) determines the direction of dead zone extension. We divide the direction into 8 types: N, NE, E, SE, S, SW, W, and NW. As we use a 2D topography, determining the direction is easy based on the initial dead zone.

Speed: The speed of spreading of fire is a simple time variable (like pause time), which dictates when the dead zones appear.

Spreading: This property actually determines where the dead zone will appear, how big it will be, and the overall shape of the fire.

The fundamental concept of proximity zones actually determines which nodes will transmit data. Every time a new dead zone is created, the algorithm defined in previous sections is followed to create the proximity zones. Based upon the three properties we present the square and circular techniques in the following sections and then perform simulation analysis for the circular model only.

6.5.1 Rectangular Spreading Model

The basic shape of the dead zone in this model is a rectangle with a maximum allowed height and width. To avoid repetition of variables, we assume that the reader is aware of Table 6.1, and the algorithm associated with it. Figure 6.7 shows the new rectangular dead zone and associated variables with it. The objective is to start from a dead zone, and then link more dead zones in a particular direction with a certain spread.

The direction of fire spread will be determined by the variable set *d*. where

$$d = \{n , ne , e , se , s , sw , w , nw\}$$

The variables used are similar to the nomenclature used for general direction is a 2D topology. These can be predetermined before the simulation starts or selected during the course of the simulation. As the direction of wind is relatively consistent in forest fires, and does not drastically change (e.g. north to south), thus it is recommended that changes to direction should be carefully changed during the simulation, and should not be chosen randomly. The next iteration of dead zones will originate in that *general* direction. The term general direction is important, as the algorithm does not pick only one point as the direction, rather it picks three. E.g. if the direction is picked to be *ne* the next iteration of dead zone originates from the subset of $\{n , ne , e\}$. Table 6.3 shows the directions and subsets.

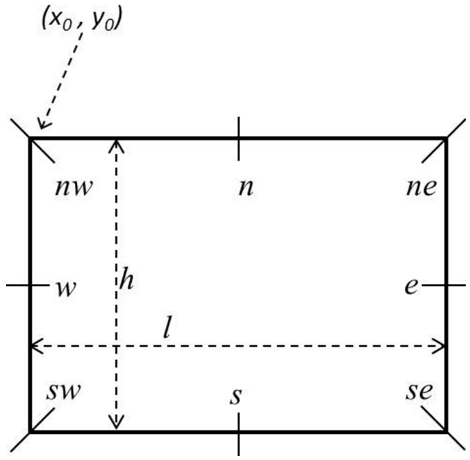


Figure 6.7: Rectangular model for forest fire extension with direction variables

Table 6.3: Dead zone origination point based on direction subset

Direction	Dead zone iteration picked from set
n	$\{nw , n , ne\}$
ne	$\{n , ne , e\}$
e	$\{ne , e , se\}$
se	$\{e , se , s\}$
s	$\{se , s , sw\}$
sw	$\{s , sw, w\}$
w	$\{sw , w , nw\}$
nw	$\{w , nw , n\}$

Using the sets given in Table 6.3 the next iterations of dead zones pick a subset of points to originate. The number of expansion points can be determined from the following equation. Expansion point is defined as the point from where the next iteration of dead zone will originate i.e. the set in Table 6.3.

$$E_p = \left\lceil \frac{S_p' \times \alpha}{\beta} \right\rceil \quad (6.5)$$

where, α is the total number of elements in direction set (3 in Table 6.3), and β is the growth control factor. If β → α, then E_p becomes exponential to α. If β → 1, then E_p becomes linear. This equation is designed to work with more complex geometric structures also.

We demonstrate the working of above algorithm using an example. Consider a forest fire starting at coordinates (x_0 , y_0). This is the location of the zero-th iteration of dead zone (the initial dead zone). Let's assume that the direction of fire spread is selected to be North East represented by the variable *ne*. From Table 6.3 we determine that the set for direction variables will be {n_0 , ne_0 , e_0}. We can use the equation to determine the number of Expansion points from the base dead zone. Using the time variable (in seconds) we will go to the next iteration, where we will generate E_p new dead zones. In most of the cases E_p will be less than α, so the initial points can be picked randomly from the set {n_0 , ne_0 , e_0}. Assuming that we set β = 2, after this step there will be two new dead zones, with their respective direction sets {n_1 , ne_1 , e_1}and {n_2 , ne_2 , e_3}. At this point we will again compute E_p to determine the number of expansion points for 2nd iteration. As there are no 6 possible points of expansion and E_p value will result in 5 we can randomly pick 5 points out of set {n_1 , ne_1 , e_1 , n_2 , ne_2 , e_2} for 3rd iteration of dead zones after the specified time interval. Figure 6.8 shows the graphical representation of this example.

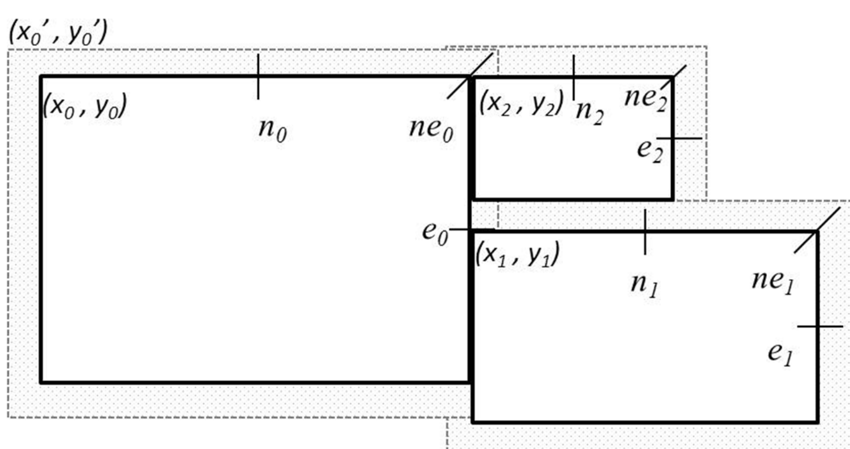


Figure 6.8: Example of Rectangular model for forest fire extension

6.5.2 Circular Spreading Model

Circular spreading uses the same conceptual model as that of rectangular, but achieves the expansion in a more smother and coherent manner. More over in natural spreading of fire, circles are more appropriate to building arbitrary shapes, as they do not have any vertices. We combine the model of Extended Circles Proximity with directional expansion from previous section. Thus, the circle is divided into 8 directions, but each direction is represented with a 90 degree arc on the boundary of the circle. In Figure 6.9 we show the basic directions, and example directional arcs for North West (*nw*), East (*e*), and South (*s*).

We modify the algorithm as described for rectangular spreading, by picking the expansion points randomly on the arc. The expansion point becomes the center of the next circular dead zone. For example, if the direction picked is *e*, then the expansion points are picked in the *e arc*, a show in in Figure 6.10. The equation used to calculate the number of expansion points is same as in previous section. To pick the random point on the arc, we use the following equation:

$$(x_i\ ,\ y_i) = \{\ x_0 + R_0\ Cos(\theta)\ ,\ y_0 + R_0\ Sin(\theta)\ \} \tag{6.6}$$

where, θ is a random angle between arc endpoints for that direction. Selecting a random angle is very critical in this calculation, as it determines where the expansion point will be. We recommend usage of uniform random distribution formulas for this purpose.

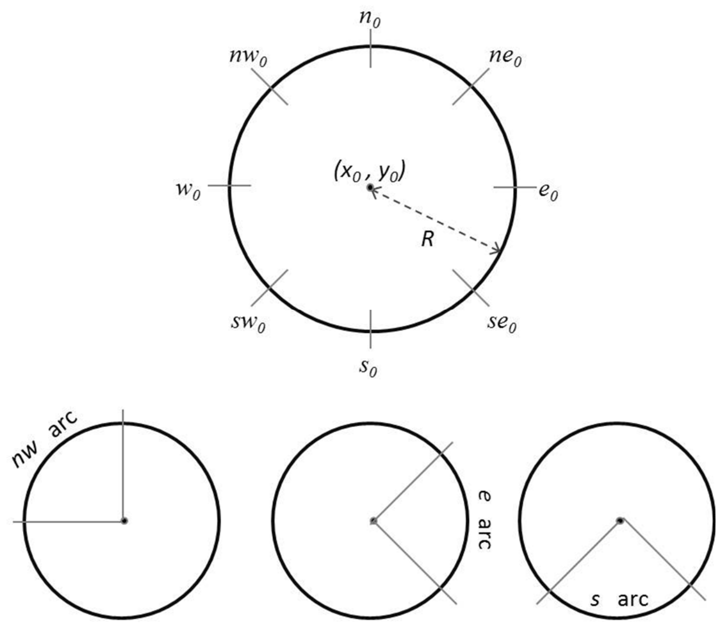


Figure 6.9: Circular spreading directions and example arcs..

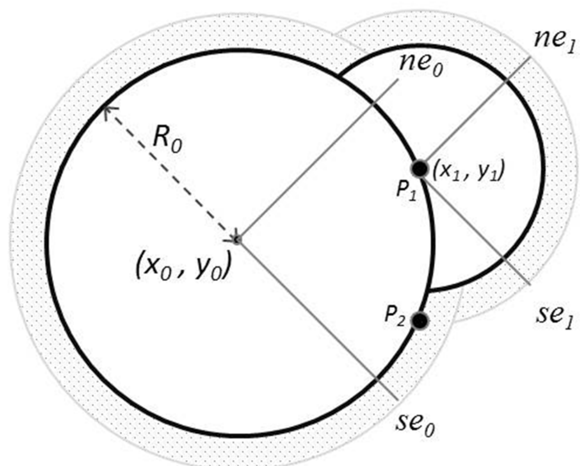


Figure 6.10: Circular expansion on *e* arc

Figure 6.10 show in the randomly selected points on *e* arc as P_1 and P_2, where new circles with random radius can be created to form dead zones along with their proximity areas. In the next iteration, equation will be again used to find the number of expansion points on *e* arcs dead zone centered at P_1 and P_2.

6.5.3 Simulation Analysis of Circular Spreading of Fire

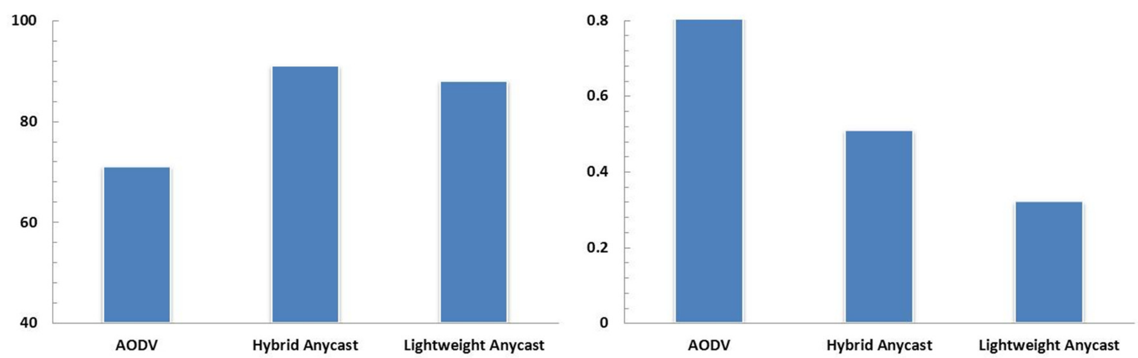
We have only evaluated the circular spreading model we proposed in the previous section, as it is closer to realistic situations. We have used a 2D topography spread over 2500 x 2000 meters, with a uniform random node density of 250. Total simulation time is 150 sec with 6 sink nodes moving with RWP model in the area. Coordinates and time of origin of fire is randomly chosen at the beginning of the simulation. α is set to 3, and β is set to 2 for the whole simulation process. Direction of wind is randomly chosen at the beginning of simulation and remains same for its duration. The circular expansion pause time is randomly chosen between 10 sec and 20 sec at the beginning of simulation, and then remains constant. Confidence interval is set at 50 simulations.

We see from the performance results a different story than we got from the results of the Extended circular model, and simple Dead Zone models. The performance has actually improved in these results, mainly due to the fact that there are more nodes in the proximity zones. Regardless of that fact, the new fact is that there is a change in observed performance, which is closer to realistic situations.

This simulation analysis is very simple in nature, but more complex scenarios can be modeled using our proposed techniques. Based in the direction, spread pause time, expansion points, dead zone radius, and proximity distance, very realistic scenarios can be designed for analysis of protocols. Some of these situations can be:

- Wind direction can change during the course of simulation and can be manipulated by the direction control variable. This will change the direction of dead zone expansion.

- The spread pause time can be change dynamically to model different types of fuel available to the fire in a forest. Dead brush, wet areas, etc. can be modeled using the pause time to simulate the time it takes for them to catch fire.
- By changing the values of α and β the aggression of fire can be controlled to quickly spread or to intensify in certain areas. These can also be dynamically adjusted to get more realistic.



(a) Delivery percentage (b) Control Overhead

Figure 6.11: Performance of expanding circular model for forest fires

CHAPTER 7: MOBILITY OPTIOMIZATION AND MULTI-TIER COMMUNICATION OF SINK NODES

Application scenario described earlier in our research, bring another challenge which is of data collection. Sinks can be randomly or strategically deployed to collect information from sensing nodes. Sensor routing protocols available in literature give little or no importance to the physical positioning of the sink nodes. Usually sinks are considered to be either other sensor nodes or a fixed location outside the network. In reality, depending on application of network, the sinks can be very different than the sensor.

In a battle field or a forest fire situation, sinks can be strategically located, mounted on mobile robots, present on vehicles in the area, or even carried by soldiers. These sinks can perform multiple roles as compared to sinks that are just sensor nodes. Sink nodes mounted on mobile platforms will have the advantage of maneuvering themselves into positions so that they can provide redundancy and cover the whole network. Also these sinks can form a higher level of communication network (using long range communication technologies or high power transmission) to facilitate other application requirements. Furthermore these devices can be equipped with other technologies like location aided services, visual surveillance equipment, robotic arms, etc.

There is a need explore the question, if it is possible for the sink nodes to maneuver themselves into position where they can collect data optimally. There are a number of factors that impact this optimization, e.g. knowledge of topography, knowledge of sensor locations, sensor requirements, to name a few. There is some work available in literature which addresses the issue of collecting information using mobile nodes, and building topological and topographic maps of sensor networks using heuristic and other mathematical approaches. Using mathematical models and simulations it has been shown that these techniques work to a great extent, and can be used in real world. Given such algorithms and techniques, it is also possible that all sensor nodes are already aware of their physical locations (pre-planned or GPS) and thus can convey this information to other nodes. The biggest benefit of this information is awareness of *physical location of the event* (very beneficial for forest fires). But it also addresses the question we explore, that: is it possible to use such information to maneuver the sink nodes into optimal data collection positions?

In the following sections we address different strategies and mechanisms that can be employed by our architecture, in order to become location aware, communicate at multiple levels, and optimizing the path.

7.1 Location Awareness by Mobile Sinks

The fundamental argument in our research is the mobility of sink nodes, and we have shown with different examples that how it is a practical and realistic assumption as compared to static sinks, for certain applications. In our research we assume that the sensor nodes are not aware of their physical locations, and use the sink nodes to build a data base of location of nodes using its own location awareness hardware (e.g. GPS).

Sensor Node Marking: Considering a forest with hundreds of sensors deployed for monitoring the environment variables, there are times when there is no active communication in progress. As we have argued before that sensors will generate data only when a significant change in temperature is to be reported or explicitly queried, resulting in durations where sinks will not be serving to any sensor nodes. These times can be used to collect location information. As the sinks are mobile, they can keep moving around (given ample energy is available) and with low level physical signal sensing, they can identify and mark the locations of different senor nodes.

This algorithm is very simple but yet affective. As the sink moves, it will find many senor nodes along its path. Also the sink nodes are by default programmed to transmit Hello packets. In this algorithm we propose that the sensor node responds with a physical/data link layer transmission with its node identifier only. This will keep the communication overhead to the minimum, and will not waste energy on sensor node's part. The sink can then log the node identifier and its own physical location. The sink does not need to collect a very precise location. If both the sink and sensor can communicate directly, then the location information is good enough. Over time, this collection will generate a complete topographic and topological map of the sensor nodes.

Location Database: Once the location of sensor nodes has been captured, it is important to save it where it can be easily accessed. Although we assume that sinks are powerful nodes, but they may not have enough capabilities to store location information for hundreds of nodes. As this completely depends on the capability of the sink, we propose multiple solutions for this.

- If the sink node has enough memory available, the ideal situation would be to store the location data base with itself. This will result in quick resolution of queries and faster updates.

- In cases where the sink cannot store such information, the location information can be stored on a remote location dedicated for this purpose. Sink can relay the gathered information using other sink nodes to the remote location, or temporarily store it till it comes in contact with the remote location.
- Sink nodes can also exchange such data bases when in contact with each other, or periodically update each other.

Path Reduction Strategy: Using this location awareness scheme the sink nodes can be programmed to move towards the source nodes when data is originating from a particular one. This will physically reduce the distance between the source node and the sink. Although this does not guarantee in optimal path of communication (until the sink and source are in direct contact), but may be helpful in networks with high node density.

7.2 Mobility Optimization using Path Cost Metrics

In this scheme, we propose the usage of Path Cost Metrics that are used by our proposed Anycast protocols. Both the Hybrid Anycast Routing protocol and the Lightweight Hybrid Anycast protocol carry path cost fields which are recalculated at every hop for the incurred cost. The Sink node then returns this cost in the RREP, which is used by the source node to determine if it wants to establish connection to this sink or not.

Path Back-Track Algorithm: In this algorithm we use the knowledge of node locations along the route of data. As we use a destination based routing algorithm (not a source based), thus the sink node has knowledge of two node, i.e. source node, and last hop node. The sink node can refer to its location data base and move to the position of last hop. Consider that a path comprises of *n* hops, once the sink is at the n^{th} hop, the path will be reduced to *n-1* hops. This will give the sink node identity of *n-1* node and sink can locate its physical position and move to its place. Following the reverse path one node at a time, the sink can reach a point where it cannot move any further physically, or another path it severs results in an increased path cost. Following two conditions must be followed while optimizing the path.

- If the sink is serving more than one sources, it always starts optimizing the path that has higher cost. If they are same, randomly pick one.
- Path optimization should immediately stop if another path cost increases as a result of the last movement, and last movement must be reversed.

We demonstrate the workability of this algorithm in the following experiment.

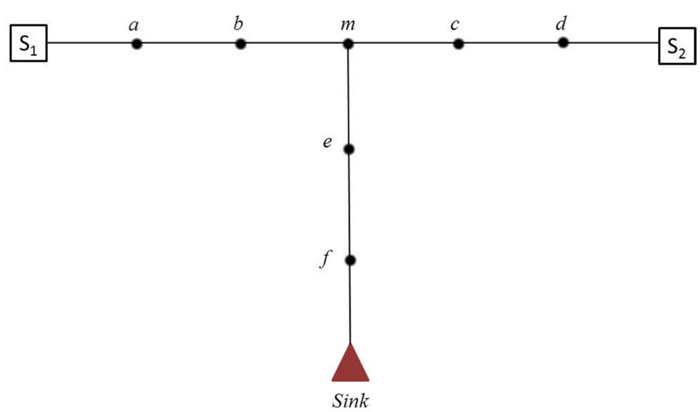


Figure 7.1: Path Back-Tracking Algorithm experiment

Path Back-Tracking Experiment: In the experiment we set up a network as show in Figure 7.1. There are two source nodes (S_1 and S_2), with equal number of intermediate hops to a single sink node. The solid line shows the wireless connectivity among nodes. We assume that the sink knows exact XY coordinates of each node. The path cost metric used is solely based on hop counts, although it can be more complex, but this experiment is designed to demonstrate the basic concept. Both sources generate data at the same time, thus finding the routes to the sink with negligible time difference. As we can see from the figure the optimal place for the sink to be located is at node *m*, so that both the source can get a path cost of 3 hops. Due to the initial location of sink the path cost is 6 hops. The objective is to observe if the sink moves to node *m*.

Table 7.1: Path Back-Tracking movement results for sink node

Time	Movement	Sink Node location	Hops to S_1	Hops to S_2
10.07	1	*f*	5	5
15.4	2	*e*	4	4
21.1	3	*m*	3	3
25.1	4	*c*	4	2
25.1	5	*m*	3	3
	Stopped	*m*	3	3

The results of the movement are shown in Table 7.1. We can observe that the sink node moves from its original location to *f*, *e*, and the *m*. At node m, as the path cost is same for both of the sources, it randomly picks S2, and tries to optimize the path. An immediate return is triggered as path cost for the other source increased. As per the algorithm, the sink stops its movement at this point. This basic algorithm can be evolved into more complex and adaptive technique, as there are many challenging situations in back tracking that can create unwanted oscillations and other movements.

7.3 Topology based Connectivity Graphs

Based on the observation of the Path Back-Track Algorithm, we can also utilize topology based connectivity graph as discussed in literature [104, 105]. These graphs can be generated based upon physical location awareness, or through node communication over time. In case such graphs are available for a network, optimal data gathering points can be pre computed for sink nodes. Sink nodes may not have this computational power, but such a program can be deployed at a remote location, from where data can be provided to sink nodes.

It is important to note that the dynamic nature of forest fire environment may change how the nodes will behave, thus it would be more beneficial to compute the optimal points as the network structure changes due to creation of dead zones. These optimal points will also require special algorithms that can determine the topographic information physical sink path planning to avoid dead zone, other hazards, and areas where sink nodes cannot physically reach. This creates a new challenge in solving the mobility optimization problem. In our research we have not implemented these algorithms, or tested any of the connectivity graphs. This can be a promising future work for our research.

CHAPTER 8: CONCLUSION AND FUTURE WORK

Wireless networks have seen exponential growth in terms of cellular and Wi-Fi communication. The devices have evolved from simple telephones to complex computing devices with multiple technologies built inside. Moreover with the availability of high speed connectivity to Internet on mobile devices, the distinction between voice, data and video communication is quickly vanishing. Thus, if a device is connected to the Internet, all types of communication are possible. This drives the need for new architectures that can harvest the availability of multiple types of technologies available on communication devices. This also raises new challenges of seamless integration, usage of multihop architectures, and issues related to applications supported by the network. In essence, the application dictates how the network should be designed, rather being limited by the network capabilities,

In this research work we introduce two new communication architectures, which are designed to fulfill application requirements first. We have developed a first modular communication framework that can provide heterogeneous devices with access to each other by using hybrid protocols for routing and path maintenance. The framework supports QoS using Path Cost Metrics for both route and access point selection. The framework is implemented using a new Hybrid Anycast Routing protocol, that out-performs traditional ad hoc routing protocols. Our extensive set of simulations has shown that our proposed protocol can select 1 of *n* available access points based on application specific criteria, in various network situations.

A second framework supports similar issue for sensor networks. Wireless sensor networks possess different inherent properties as compared to wireless multihop networks. The most important are the device capabilities and application of networks. We have conducted a detailed study on the differences and shown that protocols and architectures designed for ad hoc or access networks cannot be directly used for sensor networks. There is a need for a detailed framework that can mold itself to the needs of the application of the sensor network. In our research work, we have modified the framework we propose for heterogeneous networks, and scaled it down for the use in sensor networks. This new framework has become very different than its original version, and has shown some promising results through simulation. We have developed a light-weight version of Hybrid Anycast Routing protocol for use in sensor networks. This protocol borrows the anycast and path cost capabilities of its parent, but reduces the unwanted overhead to make it light-weight. In simulation, we have observed that, the light-weight protocol can compete very well with the full-featured protocol. Although the light-weight version cannot achieve the same level of data delivery percentage, it does not lag far behind and it compares well against AODV-based protocols. The biggest benefit it gives is the decreased control overhead, which is crucial in sensor networks due to limited battery and bandwidth. We have also conducted extensive mobility testing for our work and results show promising behavior.

In the third part of our research we develop new models for realistically testing sensor network protocols. Most of the simulation systems are built to test the basic functionality (routing) of the protocols. But in order to better design and evaluate the protocols for all its capabilities, there is a need for simulation systems that can model realistic sensor network applications. In our work, we have taken the example of a forest fire detecting sensor network, and have developed models for NS-2 that can mimic the behavior of a fire spread scenario, and the sensor network behavior in response to that scenario. The models we propose are very simple to implement in almost all types of simulation systems. We have also tested our light-weight protocol to evaluate its performance in realistic situations. Our results show that simulation results differ when analyzing protocol performance using generic simulation models as compared to using realistic scenario models. Our results also show that our protocols improve performance over prior approaches, when using generic or realistic scenario models.

In the last part of our work, we have proposed strategies for optimization of mobile sinks in sensor networks. Depending on the sensor network application, the sink nodes may or may not be mobile. In a forest fire the sink nodes can be mounted on mobile robots that collect environmental data. The mobility patterns of these nodes can be optimized to increase the overall network performance. We have proposed simple node tracking algorithms for mobility optimization. More complex schemes can also be used, and other strategies can be adopted for this purpose.

Future work of our research can be in multiple directions. Starting from the Hybrid Anycast routing, there are numerous research points that need to be address, such as proactive radius optimization, complex path cost metric evaluation, and anycast to manycast algorithms. Similar issues need to be researched for the light-weight routing algorithm. Our proposed work has been mainly tested for forest fire application; thus it leaves open other application scenarios, and how to adapt the framework for them. In our realistic modeling for simulation systems, there are many open questions on how to design models that can simulate different sensor network applications. There is a need for algorithms that do not increase the complexity of the simulator itself, but still enable it to mimic the behavior of naturally occurring events that direct the performance and behavior of sensor networks. Lastly, the sink mobility using connectivity graphs and dynamically optimizing the data collection locations in sensor network are an open challenge.

REFERENCES

[1] M. Tubaishat and S. Madria, "Sensor Networks: An Overview", *in IEEE Potentials*, vol. 22(2), April-May 2003, pp. 20-23.

[2] Alico Systems Inc. http://www.alicosystems.com/wireless%20sensor.htm

[3] J. C. Navas and T. Imielinski, "Geocast-geographic addressing and routing", *in Proceedings of the 3rd Annual ACM/IEEE International Conference on Mobile Computing and Networking*, Budapest, Hungary, pp. 66–76, 1997.

[4] C. Carter, S. Yi, P. Ratanchandani, and R. Kravets, "Manycast: Exploring the space between anycast and multicast in ad hoc networks", *in Proceedings of the 9th Annual International conference on Mobile Computing and Networking*, San Diego, CA, pp. 273–285, 2003.

[5] Lu, Y., Wang, W., Zhong, Y. and Bhargava, G. "Study of Distance Vector Routing Protocols for Mobile Ad Hoc Networks." *in Proceedings of 1st IEEE International Conference on Pervasive Computing and Communications*, Texas, pp. 187-194, 2003.

[6] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks," *in Mobile Computing*, vol. 353, pp. 153-18,1 Kluwer Academic Publishers, 1996.

[7] C. Hedrick, "Routing information protocol," RFC 1058, 1988.

[8] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers", *in Proceedings of the Conference on Communications Architectures, Protocols and Applications*, New York, NY, pp. 234–244, 1994.

[9] T. Clausen, P. Jacquet, A. Laouiti, P. Muhlethaler, A. Qayyum, and L. Viennot, "Optimized link state routing protocol", *in IEEE INMIC* Pakistan.

[10] C. E. Perkins and E. M. Royer, "Ad-hoc on-demand distance vector routing," *in Proceedings of the Second IEEE Workshop on Mobile Computer Systems and Applications*, Washington, DC, IEEE Computer Society, p. 90, 1999.

[11] M. S. Corson and A. Ephremides, "A distributed routing algorithm for mobile wireless networks," *Wireless Networks*, vol. 1, no. 1, pp. 61–81, 1995.

[12] Vincent D. Park and M. Scott Corson, "A highly adaptive distributed routing algorithm for mobile wireless networks," *in Proceedings of IEEE Conference on Computer Communications (INFOCOM'97)*, April, Kobe, Japan, vol. 3, pp. 1405–1413, 1997.

[13] G. Pei, M. Gerla, X. Hong, and C.-C. Chiang, "A Wireless Hierarchical Routing Protocol with Group Mobility", *in Proceedings of IEEE Wireless Communications and Networking (WCNC 1999)*, New Orleans, LA, vol. 3, pp. 1538–1542, 1999.

[14] C-C. Chiang and M. Gerla, "Routing and Multicast in Multihop, Mobile Wireless Networks," *in Proceedings of IEEE 6th International Conference on Universal Personal Communications Record*, San Diego, CA, vol. 2, pp. 546–551, 1997.

[15] Z. J. Haas and M. R. Pearlman, "ZRP: A Hybrid Framework for Routing in Ad hoc Networks," *in Ad hoc Networking*, pp. 221–253. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2001.

[16] K. X. Xu, X. Y. Hong, and M. Gerla, "Landmark Routing in Ad hoc Networks with Mobile Backbones", *in Journal of Parallel and Distributed Computing*, vol. 63, no. 2, pp. 110–122, Feb 2003.

[17] E. Royer and C. Toh, "A Review of Current Routing Protocols for Ad hoc Mobile Wireless Networks", *in IEEE Personal Communications*, vol. 6, pp. 46–55, Apr 1999.

[18] X. Y. Hong, K. X. Xu, and M. Gerla, "Scalable Routing Protocols For Mobile Ad Hoc Networks", *in IEEE Network*, vol. 16, pp. 11–21, Jul-Aug 2002.

[19] Y-C. Tseng, S-Y. Ni, Y-S. Chen, and J-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad hoc Network", *in Wireless Networks*, vol. 8, no. 2/3, pp. 153–167, March-May 2002.

[20] J. Cartigny and D. Simplot, "Border Node Retransmission Based Probabilistic Broadcast Protocols in Ad hoc Networks", *in 36th Annual Hawaii International Conference on System Sciences*, Hawaii, HI, Jan 2003.

[21] W. Lou and J. Wu, "On Reducing Broadcast Redundancy in Ad hoc Wireless Networks", *in IEEE Transactions on Mobile Computing*, vol. 1, no. 2, pp. 111–123, Apr 2002.

[22] W. Peng and X-C. Lu, "On The Reduction of Broadcast Redundancy in Mobile Ad hoc Networks", *in Proceedings of the 1st ACM International Symposium on Mobile Ad hoc Networking & Computing*, Boston, Massachusetts, pp. 129–130, 2000.

[23] A. Qayyum, L. Viennot, and A. Laouiti, "Multipoint Relaying for Flooding Broadcast Messages in Mobile Wireless Networks", *in Proceedings of the 35th Annual Hawaii International Conference on System Sciences*, pp. 3866–3875, 2002.

[24] J. Sucec and I. Marsic, "An Efficient Distributed Network-Wide Broadcast Algorithm for Mobile Ad hoc Networks", Tech. Rep., Rutgers University, Sep 2000.

[25] J. Wu and F. Dai, "Broadcasting in Ad hoc Networks Based on Self-pruning", *in Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 2240–2250, 2003.

[26] R. R. Choudhury and N. H. Vaidya, "Ad hoc Routing using Directional Antennas", Tech. Rep., University of Illinois at Urbana Champaign, May 2002.

[27] C. Hu, Y. Hong, and J. Hou, "On Mitigating the Broadcast Storm Problem with Directional Antennas", *in Proceedings of IEEE International Conference on Communications*, vol. 1, pp. 104–110, 2003.

[28] C-C. Shen, Z. Huang, and C. Jaikaeo, "Directional Broadcast for Mobile Ad hoc Networks with Percolation Theory", *in IEEE Transactions on Mobile Computing*, vol. 5, no. 4, pp. 317–332, Apr 2006.

[29] J. E. Wieselthier, G. D. Nguyen, and A. Ephremides, "On the Construction of Energy-Efficient Broadcast and Multicast Trees in Wireless Networks", *in Proceedings of 19th Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, pp. 585–594, 2000.

[30] J. Cartigny, D. Simplot, and I. Stojmenovic, "Localized Minimum-energy Broadcasting in Ad hoc Networks", *in Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 2210–2217, 2003.

[31] N. Li, J. Hou, and L. Sha, "Design and Analysis of an mst-based Topology Control Algorithm", *in Proceedings of 22nd Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 3, pp. 1702–1712, 2003.

[32] F. Ingelrest and D. Simplot-Ryl, "Localized Broadcast Incremental Power Protocol for Wireless Ad hoc Networks", *in 10th IEEE Symposium on Computers and Communications*, pp. 28–33, 2005.

[33] C. Metz, "IP Anycast Point-to-(any) Point Communications", *in IEEE Internet Computing*, vol. 6, issu. 2, pp. 94-98, 2002.

[34] S. Bhattacharjee, M. Ammar, E. Zegura, V. Shah, and Z. Fei, "Application-Layer Anycasting", *in Proceedings 16th Annual Joint Conference*, California, pp. 1388-1396, 1997.

[35] Romit Roy Choudhury, and Nitin H. Vaidya, "MAC-Layer Anycasting in Ad hoc Networks", *in ACM SIGCOMM Computer Communications Review*, vol. 3, iss. 1, pp. 75-80, 2004.

[36] V Park, and J. Macker, "Anycast Routing for Mobile Services", *in Proceedings of 33rd Annual Conference on Information Sciences and Systems*, Baltimore, MD, March 1999.

[37] N. Thepvilojanapong, Y. Tobe, and K. Sezaki, "HAR: Hierarchy-Based Anycast Routing Protocol for Wireless Sensor Networks", *in IEEE SAINT Symposium*, Italy, pp. 204-212, Feb. 2005.

[38] Y. Thomas Hou, Y. Shi, and H. Sherali, "On Base Station Selection for Anycast Flow Routing in Energy-Constrained Wireless Sensor Networks", *in Proceedings of 2nd IEEE International Conference on QoS in Heterogeneous Wired/Wireless Networks*, Florida, pp. 8, 2005.

[39] V. Lenders, M. May, and B. Plattner, "Density-based vs. Proximity-based Anycast Routing for Mobile Networks", *in Proceedings of 25th IEEE Infocom*, Barcelona, Spain, April 2006.

[40] C. Intanagonwiwat, and D. Lucia, "The Sink-based Anycast Routing Protocol, for Ad hoc Wireless Sensor Networks", Technical Report, 99-698, Dept. of Comp. Sci., USC.

[41] J. Wang, Y. Xheng, and W. Jia, "An AODV-Based Anycast Protocol for Mobile Ad hoc Network", *in Proceedings of 14th International IEEE Symposium on Personal, Indoor and Mobile Radio Communications*, Beijing, China, pp. 221-225, 2003.

[42] G. Peng, Yinaghu Yang, and Chuanshan Gao, "ARDSR: An Anycast Routing Protocol for Mobile Ad Hoc Network", *in Proceedings of 6th IEEE Symposium on Emerging Technologies: Mobile and Wireless Communication*, Shanghai, China, pp. 505-508, 2004.

[43] H. Luo, R. Ramjee, P. Sinha, Li (Reean) Li, and S. Lu, "UCAN: A Unified Cellular and Ad hoc Network Architecture", *in Proceedings of 9th Annual International Conference Mobile Computing and Networking*, California, pp. 353-367, 2003.

[44] C. Perkins, E. Belding-Royer, S. Das, "Ad hoc On-demand Distance Vector Routing," IETF RFC 3561, 2003.

[45] David B. Johnson, David A. Maltz, and Yi-Chun Hu, "The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR)", IETF Internet Draft, draft-ietf-manet-dsr10.txt, Work in Progress, July 2004.

[46] Samir R. Das, Charles E. Perkins, and Elizabeth M. Royer, "Performance Comparison of Two On-Demand Routing Protocols for Ad Hoc Networks", *in Proceedings of IEEE Conference on Computer Communications (INFOCOM)*, Tel Aviv, Israel, pp. 3-12, 2000.

[47] Akyildiz, I.F., Su, W., Sankarasubramaniam, Y., and Cayirci, E., "A Survey on Sensor Networks", *in IEEE Communications Magazine*, vol. 40(8), pp. 102-116, August 2002.

[48] J. N. Al-karaki and A. E. Kamal, "Routing Techniques in Wireless Sensor Networks: A survey", *in IEEE Wireless Communications*, vol. 1, pp. 6-28, 2004.

[49] C. Intanagonwiwat, R. Govindan, and D. Estrin, "Directed diffusion: A Scalable and Robust Communication Paradigm For Sensor Networks", *in Proceedings of ACM MobiCom*, Boston, MA, pp. 56-67, 2000.

[50] W. Heinzelman, J. Kulik, and H. Balakrishnan, "Adaptive Protocols for Information Dissemination in Wireless Sensor Networks", *in Proceedings of 5th ACM/IEEE Mobicom Conference (MobiCom '99),* Seattle, WA, August, pp. 174-85, 1999.

[51] J. Kulik, W. R. Heinzelman, and H. Balakrishnan, "Negotiation-based Protocols for Disseminating Information in Wireless Sensor Networks", *in Wireless Networks*, vol. 8, pp. 169-185, 2002.

[52] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks", *in the Proceedings of the 1st Workshop on Sensor Networks and Applications (WSNA),* Atlanta, GA, October 2002.

[53] F. Ye, A. Chen, S. Liu, L. Zhang, "A Scalable Solution to Minimum Cost Forwarding in Large Sensor Networks", *in Proceedings of the 10th International Conference on Computer Communications and Networks (ICCCN),* pp. 304-309, 2001.

[54] C. Schurgers and M.B. Srivastava, "Energy Efficient Routing in Wireless Sensor Networks", *in the MILCOM Proceedings on Communications for Network-Centric Operations: Creating the Information Force*, McLean, VA, 2001.

[55] M. Chu, H. Haussecker, and F. Zhao, "Scalable Information-Driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks", *in The International Journal of High Performance Computing Applications,* vol. 16(3), August 2002.

[56] Y. Yao and J. Gehrke, "The Cougar Approach to In-network Query Processing in Sensor Networks", *in SIGMOD Record*, September 2002.

[57] R. C. Shah and J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", *in IEEE Wireless Communications and Networking Conference (WCNC),* Orlando, FL, 2002.

[58] N. Sadagopan et al., "The ACQUIRE Mechanism for Efficient Querying in Sensor Networks", *in the Proceedings of the 1st International Workshop on Sensor Network Protocol and Applications*, Anchorage, Alaska, May 2003.

[59] W. Heinzelman, A. Chandrakasan and H. Balakrishnan, "Energy-Efficient Communication Protocol for Wireless Micro sensor Networks", *in Proceedings of the 33rd Hawaii International Conference on System Sciences (HICSS '00)*, January 2000.

[60] A. Manjeshwar and D. P. Agarwal, "TEEN: A Routing Protocol for Enhanced Efficiency in Wireless Sensor Networks", *in 1st International Workshop on Parallel and Distributed Computing Issues in Wireless Networks and Mobile Computing*, April 2001.

[61] A. Manjeshwar and D. P. Agarwal, "APTEEN: A Hybrid Protocol for Efficient Routing and Comprehensive Information Retrieval in Wireless Sensor Networks", *in Proceedings International of Parallel and Distributed Processing Symposium*, pp. 195-202, 2002.

[62] L. Subramanian and R. H. Katz, "An Architecture for Building Self Configurable Systems", *in Proceedings of IEEE/ACM Workshop on Mobile Ad Hoc Networking and Computing*, Boston, MA, August 2000.

[63] S. Lindsey, C. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems", *in Proceedings of IEEE Aerospace Conference*, vol. 3, 9-16 pp. 1125-1130, 2002.

[64] F. Ye, H. Luo, J. Cheng, S. Lu, L. Zhang, "A Two-tier Data Dissemination Model for Large-Scale Wireless Sensor Networks", *in Proceedings of 8th ACM Annual International Conference on Mobile Computing and Networking (MobiCom)*, pp. 148-159, 2002.

[65] V. Rodoplu and T. H. Meng, "Minimum Energy Mobile Wireless Networks", *in IEEE Journal Selected Areas in Communications*, vol. 17, no. 8, pp. 1333-1344, 1999.

[66] Q. Li and J. Aslam and D. Rus, "Hierarchical Power-aware Routing in Sensor Networks", *in Proceedings of the DIMACS Workshop on Pervasive Networking*, May, 2001.

[67] Jamal N. Al-Karaki, Raza Ul-Mustafa, Ahmed E. Kamal, "Data Aggregation in Wireless Sensor Networks: Exact and Approximate Algorithms", *in Proceedings of IEEE Workshop on High Performance Switching and Routing (HPSR)*, Phoenix, Arizona, 2004.

[68] Q. Fang, F. Zhao, and L. Guibas, "Lightweight Sensing and Communication Protocols for Target Enumeration and Aggregation", *in Proceedings of the 4th ACM international symposium on Mobile ad hoc Networking and Computing (MOBIHOC)*, pp. 165-176, 2003.

[69] Y. Xu, J. Heidemann, D. Estrin, "Geography-informed Energy Conservation for Ad-hoc Routing", *in Proceedings of the 7th Annual ACM/IEEE International Conference on Mobile Computing and Networking*, pp. 70-84, 2001.

[70] B. Chen, K. Jamieson, H. Balakrishnan, R. Morris, "SPAN: An Energy-Efficient Coordination Algorithm for Topology Maintenance in Ad Hoc Wireless Networks", *in Wireless Networks*, vol. 8, no. 5, pp. 481-494, September 2002.

[71] Y. Yu, D. Estrin, and R. Govindan, "Geographical and Energy-Aware Routing: A Recursive Data Dissemination Protocol for Wireless Sensor Networks", UCLA Computer Science Department Technical Report, UCLA-CSD TR-01-0023, May 2001.

[72] I. Stojmenovic and X. Lin, "GEDIR: Loop-Free Location Based Routing in Wireless Networks", *in Proceedings of International Conference on Parallel and Distributed Computing and Systems*, Boston, MA, Nov. 1999.

[73] F. Kuhn, R. Wattenhofer, A. Zollinger, "Worst-Case Optimal and Average-case Efficient Geometric ad hoc Routing", *in Proceedings of the 4th ACM International Conference on Mobile Computing and Networking*, pp. 267-278, 2003.

[74] Q. Li and J. Aslam and D. Rus, "Hierarchical Power-aware Routing in Sensor Networks", *in Proceedings of the DIMACS Workshop on Pervasive Networking*, May, 2001.

[75] J.-H. Chang and L. Tassiulas, "Maximum Lifetime Routing in Wireless Sensor Networks", *in Proceedings of Advanced Telecommunications and Information Distribution Research Program (ATIRP 2000)*, College Park, MD, Mar. 2000.

[76] C. Rahul, J. Rabaey, "Energy Aware Routing for Low Energy Ad Hoc Sensor Networks", *in IEEE Wireless Communications and Networking Conference (WCNC)*, vol. 1, Orlando, FL, pp. 350-355, 2002.

[77] D. Braginsky and D. Estrin, "Rumor Routing Algorithm for Sensor Networks", *in International Conference on Distributed Computing Systems (ICDCS'01)*, November 2001.

[78] K. Sohrabi, J. Pottie, "Protocols for Self-organization of a Wireless Sensor Network", *in IEEE Personal Communications*, vol. 7, iss. 5, pp. 16-27, 2000.

[79] T. He et al., "SPEED: A Stateless Protocol for Real-Time Communication in Sensor Networks", *in Proceedings of International Conference on Distributed Computing Systems*, Providence, RI, May 2003.

[80] W. Wolf, "Building the Software Radio," *in IEEE Computer*, vol. 38, iss. 3, pp. 87-89, 2005.

[81] N. Thepvilojanapong, Y. Tobe, and K. Sezaki, "Har: Hierarchy-based Anycast Routing Protocol for Wireless Sensor Networks", *in Proceedings of Symposium on Applications and the Internet*, pp. 204–212, 2005.

[82] Y. Hou, Y. Shi, and H. D. Sherali, "On Base Station Selection for Anycast Flow Routing in Energy-Constrained Wireless Sensor Network", *in 2nd International Conference on Quality of Service in Heterogeneous Wired/Wireless Networks,* Aug 2005.

[83] V. Lenders and B. Plattner, "Density-based vs. Proximity-Based Anycast Routing for Mobile Networks", *in 25th IEEE International Conference on Computer Communications*, Barcelona, Spain, pp. 1–13, 2006.

[84] C. Intanagonwiwat and D. D. Lucia, "The Sink-Based Anycast Routing Protocol, for Ad Hoc Wireless Sensor Networks", Tech. Rep. 99-698, USC Computer Science, CA, USA, 1999.

[85] J. Wang, Y. Zheng, and W. Jia, "An aodv-based anycast protocol for mobile ad hoc network", *in 14th IEEE Proceedings on Personal, Indoor and Mobile Radio Communications*, vol. 1, pp. 221–225, 2003.

[86] G. Peng, J. Yang, and C. Gao, "Ardsr: An anycast routing protocol for mobile ad hoc network", *in IEEE 6th Circuits and Systems Symposium on Frontiers of Mobile and Wireless Communication*,, vol. 2, pp. 505–508, 2004.

[87] C. Cordeiro and D. Agarwal, "Ad hoc & Sensor Networks: Theory and Applications", Singapore, World Scientific Publishing, 2006, pp. 429-451.

[88] S. Kurkowski, T. Camp and M. Colagrosso, "MANET Simulation Studies: The Incredible", *in ACM Mobile Computing and Communications Review*, vol. 9, no. 4, pp. 50-61, 2005.

[89] J. A. Stankovic, "Research Challenges for Wireless Sensor Networks", *in ACM SIGBED Review*, vol. 1, iss. 2, pp. 9-12, 2004.

[90] Ian Sample, "It'll blow you away", in New Scientist, iss. 2285, April 2001, www.newscientist.com.

[91] C. Comez, P Salvatella, O. Alonso and J. Paradells, "Adaptive AODV for IEEE 802.15.4 Mesh Sensor Networks: Theoretical Discussion and Performance Evaluation in a Real Environment", *in Proceedings of International Symposium on World of Wireless, Mobile and Multimedia Networks (WoWMoM'06),* Buffalo NY, pp. 159-170, 2006.

[92] L. Cao, K. Sharif, Y. Wang and T. A. Dahlberg, "Multiple-Metric Hybrid Routing Protocol for Heterogeneous Wireless Access Networks", *in Proceedings of 6th IEEE Consumer Communication and Networking Conference (CCNC'09)*, Las Vegas NV, January 2009.

[93] K. Sharif, L. Cao, Y. Wang and T. A. Dahlberg, "A Hybrid Routing Protocol for Load Balancing in Heterogeneous Access Networks", *in Proceedings of 17th International Conference on Computer Communication and Networks (ICCCN'08)*, St. Thomas, August 2008.

[94] L. Cao, K. Sharif, Y. Wang, T. Dahlberg, "Multiple Metric Hybrid Anycast Protocol for Heterogeneous Access Networks", *in International Journal of Ad Hoc and Ubiquitous Computing, Special Issue on Wireless Network Algorithm and Theory*, 2011.

[95] L. Cao, K. Sharif, Y. Wang, T. Dahlberg, "Adaptive Multiple Metrics Routing Protocols for Heterogeneous Multi Hop Wireless Networks", *in Proceedings of 5th IEEE Consumer Communications and Networking Conference (CCNC '08),* Las Vegas Nevada, pp. 13-17, January 2008.

[96] L. Cao and T. Dahlberg, "Path Cost Metrics for Multihop Network Routing", *in 25th IEEE International Performance, Computing and Communications Conference*, Phoenix, pp. 15–21, 2006.

[97] L.Cao, T. Dahlberg, and Y. Wang, "Performance evaluation of energy efficient ad hoc routing protocols", *in IEEE International Performance, Computing and Communications Conference*, New Orleans, pp. 306–313, 2007.

[98] H. Karl and A. Willing, "Protocols and Architecture for Wireless Sensor Networks", John wiley & Sons Ltd, July 2007.

[99] H.S. Kin, T. F. Abdelzaher and W. H. Kwon, "Minimum-Energy Asynchronous Dissemination to Mobile Sinks in Wireless Sensor Networks", *in Proceedings of SenSys*, Los Angeles, CA, pp. 193- 204, Nov. 2003.

[100] C. Shen, C. Srisathapornphat, and C. Jaikaeo, "Sensor Information Networking Architecture and Applications", *in IEEE Personal Communications*, pp. 52-59, 2001.

[101] R. C. Shah, S. Roy, S. Jain, and W. Brunette, "Data MULEs: Modeling a Three-tier Architecture for Sparse Sensor Networks", Intel Research, Tech. report IRS-TR-03-001, January 2003.

[102] S. Jain , R. C. Shah, W. Brunette, G. Borriello, and S. Roy, "Exploiting Mobility for Energy Efficient Data Collection in Wireless Sensor Networks", *in Mobile Networks and Applications*, vol. 11(3), Springer Science, pp. 327-339, 2006.

[103] K. Sharif, L. Cao, T. Dahlberg, "Anycast Based Lightweight Routing Protocol for Mobile Sink Discovery in Sensor Networks", *in Proceedings of 7th IEEE Consumer Communication and Networking Conference (CCNC 2010)*, Las Vegas Nevada, January 2010.

[104] S. Aldosari and J. Moura, “Distributed Detection, in Sensor Networks: Connectivity Graph and Small World Networks”, *in Proceedings of 39th IEEE Asilomar Conference on Signals, Systems, and Computers*, Pacific Grove, CA, pp. 230-234, Nov 2005.

[105] A. Balister , B. Bollobas , A. Sarkar and M. Walters, "Connectivity of Random k-nearest Neighbor Graphs", *in Advanced Application Probability*, vol. 37, 2005.

[106] L. Cao, “Integrated Resource Management and Routing for Heterogeneous Access Networks”, University of North Carolina at Charlotte, 2008.

[107] F. Zhao and L. Guibas, “Wireless Sensor Networks: An Information Processing Approach”, Elsevier Inc., 2004.

[108] G. Yang, editor. “Body Sensor Networks”, Springer-Verlag, London, 2006.

[109] K. Lorincz and M. Welsh, “MoteTrack: A Robust, Decentralized Approach to RF-Based Location Tracking”, *in Springer Personal and Ubiquitous Computing, Special Issue on Location and Context-Awareness*, October 2006.

[110] Federal Communication Commission (FCC), http://www.fcc.gov/

[111] IEEE 802.15.1, “IEEE Standard for Information Technology: Telecommunications and Information Exchange between Systems, Local and Metropolitan Area Networks”, http://standard .ieee.org

[112] ZigBee 2007. ZigBee Alliance 2008.

[113] M. Chu, H. Haussecker, and F. Zhao, “Scalable Information-driven Sensor Querying and Routing for ad hoc Heterogeneous Sensor Networks”, *in International Journal of High Performance Computing Applications*, vol. 16(3), pp. 90-110, 2002.

[114] L. Doherty, K. Pister, and L. Ghaoui, “Convex Position Estimation in Wireless Sensor Networks”, *in Proceedings of 20th Annual Joint Conference of the IEEE Computers and Communications Societies (InfoCom 2001)*, vol. 3, pp. 1655-1663, 2001.

[115] L. Doherty, B. Warneke, B. Boser, and K. Pister, “Energy and Performance Consideration for Smart Dust”, *in International Journal of Parallel Distributed Systems and Networks*, vol.4(3), pp. 121-133, 2001.

[116] D. Estrin, R. Govindan, J. Heidmann, and S. Kumar, “Next Century Challenges: Scalable Coordination in Sensor Networks”, *in Proceedings of 5th Annual International Conference on Mobile Computing and Networking (MobiCom 1999)*, Seattle, WA, pp. 263-270, 1999.

[117] Q. Fang, J. Liu, L. Guibas, and F. Zhao, "Roam HBA: Maintaining Group Connectivity in Sensor Networks", *in Proceedings of 3rd International Symposium on Information Processing in Sensor Networks*, Berkeley, CA, pp. 151-160, April 2004.

[118] Frost and Sullivan, "Wireless Sensors, and Integrated Wireless Sensor Networks", Frost & Sullivan report, 2003.

[119] J. Gao, L. Guibas, J. Hershberger, L. Zhang, and A. Zhu, "Geometric Spanners for Routing in Mobile Networks", *in Proceedings of 2nd ACM International Cymposium on Mobile Ad hoc Networking and Computing (MobiHoc 2001)*, pp. 45-55, 2001.

[120] P. Gupta and P. Kumar, "The Capacity of Wireless Networks", *in IEEE Transaction on Information Theory*, vol.46(2), pp. 388-404, 2000.

[121] B. Krishnamachari, D. Estrin, and S. Wicker, "Modeling Data-centric Routing in Wireless Sensor Networks", Technical Report CENG 02-14, University of Southern California, 2002.

[122] F. Kuhn, R. Wattenhofer, and A. Zollinger, "Worst-case Optimal and Average-case efficient Geomatric Ad hoc Routing", *in Proceedings of 4th ACM International Symposium on Mobile Ad hoc Networking and Computing MobiHoc 2003)*, Annapolis, MD, pp. 267-278, 2003.

[123] P. Wan, X. Xu, and Z. Wang, "Wireless Coverage with Disparate Ranges", *in Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2011)*, New York, NY, 2011.

[124] T. Hossmann, T. Spyropoulos, and F. Legendre, "Putting contacts into context: Mobility modeling beyond inter-contact times" *in Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2011),* New York, NY, 2011.

[125] S. Hariharan and N. Shroff, "On Optimal Energy Efficient Convergecasting in Unreliable Sensor Networks with Applications to Target Tracking" *in Proceedings of the 12th ACM International Symposium on Mobile Ad Hoc Networking and Computing (MobiHoc 2011)*, New York, NY, 2011.

[126] S. Heimlicher and K. Salamatian, Kav, "Globs in the primordial soup: The Emergence of Connected Crowds in Mobile Wireless Networks", *in Proceedings of the 11th ACM International Symposium on Mobile Ad hoc networking and computing (MobiHoc 2010)*, Chicago, IL, pp. 91-100, 2010.

[127] J. Lee, B. Kusy, T. Azim, B. Shihada and P. Levis, "Whirlpool Routing for Mobility", *in Proceedings of the 11th ACM International Symposium on Mobile Ad hoc networking and computing (MobiHoc 2010)*, Chicago, IL, pp. 131-140, 2010

[128] A. Kermarrec and G. Tan, “Greedy Geographic Routing in Large-scale Sensor networks: A minimum network decomposition approach”, *in Proceedings of the 11th ACM International Symposium on Mobile Ad hoc networking and computing (MobiHoc 2010)*, Chicago, IL, pp. 161-170, 2010.

[129] B. Firner, C. Xu, R. Howard, and Y. Zhang, “Multiple Receiver Strategies for Minimizing Packet Loss in Dense Sensor Networks”, *in Proceedings of the 11th ACM International Symposium on Mobile Ad hoc networking and computing (MobiHoc 2010),* Chicago, IL, pp. 211-220, 2010.