

A
Dissertation
On

**CONGESTION CONTROL AND AWARENESS FOR
MULTIPATH ADHOC ENVIRONMENTS**

Submitted in Partial Fulfilment of the Requirement
For the Award of the Degree of

Master of Technology
in
Computer Science and Engineering
by
Parmenus Makunura
University Roll No. 2K13/CSE/36

Under the Esteemed Guidance of

Mr. V. Kumar

Associate Professor, Computer Engineering Department, DTU



2013-2015
COMPUTER ENGINEERING DEPARTMENT
DELHI TECHNOLOGICAL UNIVERSITY
DELHI – 110042, INDIA

ABSTRACT

In Wireless Sensor Networks (WSNs) the congestion problem is quite different as compared to traditional wired networks. Mobile Ad hoc Networks (MANETs) suffer from the dangers of congestion due to the problem of few resources. Due to the nature of a wireless channel that is dynamic in design, packet broadcasts suffer from the dangers of interruption and dwindling signal power. Data via a given path in heterogeneous ad hoc networks depends on the least possible data rate of all its paths. Most recent congestion control procedures attempt to lessen the congestion by dropping the rate at which the source nodes inoculate packets into the system. However, the proposed traffic control scheme introduces the dynamic active queue and variation of the link expiration time.

If a node with a high data rate directs traffic flow to another node with a low data rate, then the possibility of congestion is high, this will lead to long queuing postponements. The recommended protocol makes use of a hop-by-hop congestion aware routing approach that monitors the buffer levels at each intermediary node.

The multipath route establishment uses the method that discovers multiple multi hops communication between source and destination. Multi-path routing can stabilise the load more superior than the solitary route routing in ad hoc systems, thus decreasing the congestion by separating the traffic into numerous paths. This research presents a new approach of Multipath Load Balancing with AOMDV routing protocol and Dynamic Queue based congestion control approach for evading congestion in system communication flows. In this scheme the store and forwarding capability of nodes are enhanced by varying the queue length according the number of packets presently being dropped. The multipath routing performance in this technique will increase the link expiration time with vibrant queue length method. The multipath routing is no doubt better than the unipath routing and balance the load by proving the alternative path if the already established path are congested.

Consequently, there has been an greater than before attentiveness in examining Active Queue Management (AQM) in wireless nodes so as to lessen the queue dormancy and meet the difficulties of time thoughtful uses. In this study, we largely concentrate on investigating the effectiveness of multipath routing and a newly propounded AQM tool termed Controlled Delay (CoDel). We examine the usefulness of multipath routing that incorporate CoDel (AOMDVCCA) by implementing simulations in NS-2 and paralleling its working with normal unipath and multipath protocols in wireless network situations. Founded on the simulation

outcomes attained, we deliberate the benefits and deficiencies of AOMDVCCA in relation to packet drop rate and average queue length.

Keywords: Congestion; Active Queue Management; Buffer; Unipath; Multipath; Load Balancing; Dynamic Queue; Controlled Delay.

ACKNOWLEDGEMENT

Most importantly, I am motivated to express my bottomless logic of esteem and gratefulness to my project supervisor Mr. V. Kumar for providing me with the occasion of executing this project and being the superintendent strength behind this effort. I am profoundly thankful to him for the backing, instruction and inspiration he offered to make this project a success.

In addition, I am appreciative to Prof. O. P Verma, HOD in the Computer Engineering Department of DTU for his gigantic backing. I also acknowledge Delhi Technological University library and staff for offering the precise service, academic resources and environment for this work to be duly executed.

Last but not the least I would like to express sincere gratitude to my parents and friends for constantly encouraging me during the completion of work.

Parmenus Makunura

University Roll No: 2K13/CSE/36

M.Tech (Computer Science & Engineering)

Department of Computer Engineering

Delhi Technological University Delhi – 110042



Computer Engineering Department
Delhi Technological University
Delhi-110042
www.dtu.ac.in

CERTIFICATE

This is to certify that the dissertation titled “**Congestion Control and Awareness for Multipath Adhoc Environments**” is a bona fide record of work done by **Parmenus Makunura, Roll No. 2K13/CSE/36** at **Delhi Technological University** for partial fulfilment of the requirements for the degree of Master of Technology in Computer Science & Engineering. This project was carried out under my supervision and has not been submitted elsewhere, either in part or full, for the award of any other degree or diploma to the best of my knowledge and belief.

Date:

(Mr. V. Kumar)
Associate Professor and Project Guide
Department of Computer Engineering
Delhi Technological University

Table of Contents

Abstract	i
Acknowledgment	iii
Certificate	iv
List of Figures	vii
List of Abbreviations	viii

Chapter 1

Introduction

1.1 Background	1
1.2 Related work	2
1.3 Motivation	5
1.4 Research problem	6
1.5 Scope of Work	7

Chapter 2

An Overview of Wireless Sensor Networks, Congestion Control and Awareness

2.1 Wireless Sensor Networks	8
2.2 Congestion Control	11
2.3 Congestion Control Techniques	15
2.4 Congestion Avoidance Schemes	17
2.5 Wireless Sensor Networks Development Tools	19

Chapter 3

Wireless Sensor Network Routing Protocols

3.1 Routing Protocols	22
3.1.1 Path Establishment based Routing Protocols	23
3.1.2 Network Based Routing Protocols	24
3.1.3 Operation Based Routing Protocols	26
3.1.4 Next Hop Routing Protocols	27

3.2 Ad hoc On-Demand Distance Vector Routing	28
3.2.1 Route Discovery	29
3.2.2 Sequence Numbers and Loop Freedom	30
3.2.3 Use of Soft State and Route Maintenance	30
3.3 Ad hoc On-Demand Multipath Distance Vector Routing	31
3.3.1 Loop Freedom	31
3.3.2 Disjoint Paths	34
Chapter 4	
Ad Hoc On-Demand Multipath Distance Vector with Congestion Control and Awareness (AOMDVCCA)	
4.1 Shortcomings of AOMDV	39
4.1.1 Congestion and Contention	39
4.1.2 Limitations of Static Route Switching	41
4.2 Active Queue Management	41
4.3 AOMDVCCA	42
4.3.1 System Model	42
4.3.2 Proposed Congestion Control Scheme	44
Chapter 5	
Simulation Results and Analysis	
5.1 Simulation Setup	55
5.1.1 Ubuntu 14	55
5.1.2 The Network Simulator (NS2)	55
5.2 Performance Evaluation of the Proposed Protocol	57
5.2.1 Performance Comparisons of AOMDVCCA and AODV	59
5.2.2 Performance Comparisons of AOMDVCCA and AOMDV	63
5.2.3 Summary of Performance Metrics	68
Chapter 6	
Conclusion and Future Work	69
References	71

List of Figures

Figure 2.1 Overview of Sensor Networks	9
Figure 2.2 (a) Effect of congestion on throughput	13
Figure 2.2 (b) Effect of congestion on delay	13
Figure 2.3 Network performance as a function of the load	14
Figure 2.4 Congestion control categories	15
Figure 2.5 Cygwin	20
Figure. 3.1 WSN Routing Protocols	23
Figure 3.2 AODV route update rule	30
Figure 3.3 Examples of possible routing loop situations with multiple path	32
Figure 3.4 Paths preserved at different nodes to a sink	35
Figure 3.5 AOMDV Seen Table Structure	36
Figure 3.6 AOMDV RREP structure	37
Figure 4.1 Route Discovery Procedure in AOMDV	40
Figure 4.2 Congestion due to Intersected Routes	40
Figure 4.3 AOMDVCCA model	44
Figure 4.4 CoDel drop scheduler behavior	46
Figure 4.5 Simplified CoDel algorithm flowchart	47
Figure 4.6 AOMDVCCA flow chart	50
Figure 5.1 Fields of a trace file	56
Figure 5.2 Throughput (AODV, AOMDVCCA)	59
Figure 5.3 End-to-End Delay (AODV, AOMDVCCA)	60
Figure 5.4 Packet Delivery ratio (AODV, AOMDVCCA)	60
Figure 5.5(a) Packet loss (AODV, AOMDVCCA)	61
Figure 5.5(b) End-to-End Delay (AODV, AOMDVCCA)	62
Figure 5.5(c) Route Discovery Latency (AODV, AOMDVCCA)	62
Figure 5.6 PDR Analysis (AODMV, AOMDVCCA)	64
Figure 5.7 Throughput analysis (AODMV, AOMDVCCA)	65
Figure 5.9 UDP packets received analysis (AODMV, AOMDVCCA)	66
Figure 5.10 UDP packets lost analysis (AODMV, AOMDVCCA)	67

List of Abbreviations

WSNs	Wireless Sensor Networks
MANETs	Mobile ad-hoc Networks
AIMD	Additive Increase Multiplicative Decrease
AODV	Ad hoc On-Demand Distance Vector
AOMDV	Ad hoc On-Demand Multipath Distance Vector
TCL	Tool Command Language
AQM	Active Queue Management
CoDel	Controlled Delay
RED	Random Early Drop
MTU	Maximum Transmission Unit
RTT	Round Trip Time
UDP	User Datagram Protocol
TCP	Transport Control Protocol
FTP	File Transfer Protocol
RREQ	Route Request
RREP	Route Reply
RERR	Route Error
EAR	Energy Aware Routing
DD	Directed Diffusion
SAR	Sequential Assignment Routing
SPIN	Sensor Protocols for Information via Negotiation
MMSPEED	Multi path and Multi SPEED
DSR	Dynamic Source Routing
ECN	Explicit Congestion Notification

CHAPTER 1

INTRODUCTION

The chapter gives a background to the area of interest in this study. We look at related works that have been published in congestion control and awareness. The motivation of this research has been the lack rigorous congestion control with a flare of awareness of the original TCP protocols. The TCP congestion control variants introduced in the early 2000s have struggled to cope with a rise in network usage and the maiden area of wireless sensor networks. The research problem and the respective questions that this study intend to answer are presented.

1.1 Background

Heterogeneous Ad-hoc Wireless Networks

An ad hoc network is also called an infrastructure-less networks which is a collection of mobile nodes that make up a system that is free from any centrally controlled or administered set up that is common in conventional traditional networks. Such ad hoc systems can easily organise themselves to suit the changes that emanate from these movements. They also are independent from unnecessary hardware and this makes them a good candidate for rescue and disaster scenarios. To build and maintain this system intact, the constituent nodes are responsible – they are self-organising. To reach a desired destination, the nodes will seek help from their neighbouring nodes since they have a limited range of transmission.

Routing in Mobile Ad-hoc Wireless Networks

Routing protocols that are specifically designed are responsible for the forwarding of packets using other intermediate nodes. Although the set-up is dynamic in such cases, it maintains its organisations and this makes these protocols unique. The main classes of such protocols are: Proactive (Table-driven) and Reactive (On-demand). Proactive protocols evaluate the routes in the network beforehand such that when a packet is ready to be dispatched, it follows a path already protracted. As for the reactive counterparts, they appeal for a route in real time as the need arises.

Congestion in Mobile Ad-hoc Wireless Networks

As already highlighted before, the main cause of congestion in MANETs is the lack of adequate resources. Interference and fading are the main culprits that affect ad hoc networks mainly because of their dynamic terrain. Faults that happen will lead to an increase in the load of the

system. Multimedia communications and real time data increase the need for MANETs to introduce novel antics to solve congestion. Packet losses, bandwidth degradation and wasted time and energy on congestion recovery are the unwanted effects of congestion.

1.2 Related Work

Neha Tiwari and Sini Shibu [1] in their work titled, “Congestion Control in Multi-Flow Environment Using Multipath Routing Base in MANET”, present new dimension of Multipath Load Balancing with AOMDV routing protocol and Dynamic Queue based congestion control method for avoiding congestion in networks. Here the store and forwarding capability of nodes are enhanced by changing the length of the queue according to incoming data. The AOMDV protocol performance is also enhanced by incrementing the link expiration time after failure of the current connection. The new connection is established according to the new link expiration time value. Thus, the possibility that a link might fail in AOMDV is also lowered. They showed that the performance of AOMDV is improved after adding their scheme. Their work was purely focused on the controlling of congestion. In the event of a possible overwhelming network traffic flood, it would issue no prior procedures to minimise the possibility of failure.

Shitalkumar A Jain [2] in, “A Study on Congestion Aware Multipath Routing Protocols in MANET”, pointed out that traditional routing approaches forward all their data along only one path. Multipath routing protocols consider a number of possible path for the same source and destination pairings. This property is applied to keep congestion under check and overcome the performance degradation. Multipath routing, unlike single path routing, relies mainly on multiple paths, that is, if one route fails without halting the data transmission multipath routing strategy chooses another alternative path and continues the transmission. In his paper, several fundamental multipath routing protocols are discussed at depth. Every protocol has its feature and it controls congestion in its own way. In his work, Jain was more general in trying to identify how different protocols employ their own congestion control methods.

Hitesh Gupta and Pankaj Pandey [3] in their publication entitled, “Congestion Control Using Varying Queue Base Approach along with Multipath Routing Under MANET”, alluded to the fact that mobile ad-hoc network is dynamic since every node travels spontaneously inside the network. They pointed out that MANET can’t foresee node movement and traffic load of each node, consequently range of nodes area is not ideal and range of node area units vary loads

thereby leading to congestion in the network. In this paper, they prearranged a technique, AOMDV, and fluctuating queue base subject for lowering congestion. Thus they have a tendency to find frequent congestion occurrence in the system then apply multipath routing. Furthermore, a varied queue technique minimizes the congestion and increases the performance of the network. The researchers designed the proposed work and proposed algorithm. Simulation through all network parameters was done for performance evaluation.

Kezhong Liu, Layuan Li et al. [4], in his work titled “Research of QoS-Aware Routing Protocol with Load Balancing of Mobile Ad Hoc Networks”, combines the load balancing structure and multiple-constraint QoS mechanism to hunt for the best route between the source node and terminus node. The main aim of the research is to come up with a load balancing approach that can track any variations to the load rank of the surrounding nodes and be able to choose the routes with lowest loads with the knowledge or awareness of the surrounding load status. Their AQRL protocol forms an AODV advancement and exploits the node's resolvable bandwidth and burden data to assign the network loads. According to the study, the network is prevented from slipping into a state of cramming, and preserve the power of a jammed node.

Jingyuan Wang, Jiangtao Wen et al. [5] in their work titled “An Improved TCP Congestion Control Algorithm and its Performance “, they propound a novel and clever congestion control algorithm, called TCP-FIT, which can succeed satisfactorily together in high BDP and wireless setups. The approach was motivated by parallel TCP, but with the clear divisions that only one TCP construction with a one congestion window is created for each TCP period, and that no changes to other strata of the end-to-end system must be effected. This work not only tackled transport stratum congestion control through TCP enhancement technique but congestion also happens in routing time so as to enhance work through routing base congestion control technique.

S.Santhosh baboo and B. Narasimhan [6] in their work,” A Congestion-Aware hop-by-hop Routing Protocol of Heterogeneous Mobile Ad-hoc Networks”, pointed out that in MANETs of a heterogeneous nature, congestion happens due to limited resources. Because of the common dynamic topology and wireless channel, packet transmissions face stern tests from meddling and fading. They established that throughput via a specified route in heterogeneous ad hoc systems, hinges on the lowest data rate of all the links that belong to it. In a path of links with dissimilar data rates, if a node with high data rate directs further traffic to a node with low data rate, the chances of congestion are high and this can lead to queuing delays. Transmission

capability, reliability, and congestion about a link metric has to be integrated in a MANETs' congestion-aware routing. They proposed a hop-by-hop modus operandi for congestion aware routing which uses a weight value routing metric based on the queuing delay with combined MAC overhead, link quality and data rate. Amongst the routes that have been learnt, the path with least cost value is chosen, which is founded on the total network node weight. Their work was exclusively concentrating on congestion awareness. On the occasion of high traffic network, if the congestion levels are still high, the approach will not achieve the congestion control operations.

Rajkumar, G. and K. Duraiswamy [7] in their title, "A Fault Tolerant Congestion Aware Routing Protocol for Mobile Adhoc Networks", identified the problem that the operation of ad hoc routing protocols will considerably worsen once there are defective nodes in the system. Packet loss and a decline in bandwidth are caused by congestion and therefore, energy and time are wasted throughout salvage of the network. Their fault tolerant congestion aware routing protocol solves these problems by exploring the network redundancy using multipath routing. In their approach, they recommended to come up with a fault lenient congestion aware multipath routing protocol that lessens the route outages and congestion losses. They used the AOMDV protocol as a foundation for the multipath routing. This suggested system permits more nodes to recover a dropped packet. In their conclusion, Rajkumar, G. and K. Duraiswamy found out that the effective congestion control technique they proposed was able to proactively detect node level and link level congestion and performs congestion control using the fault-tolerant multiple paths.

A congestion-aware routing metric which considered MAC overhead, data-rate and buffer queuing delay, with preference given to less-congested high throughput paths with the ultimate aim being to improve channel utilization was designed by Xiaoqin Chen et al. [8]. They have also proposed the Congestion Aware Routing protocol for Mobile ad hoc networks (CARM). CARM uses a link data-rate classification approach to prevent routes with mismatched link data-rates. CARM was deliberated upon and simulated in relation to IEEE 802.11b systems.

Yi et al. [9] came up with the Multipath Optimized Link State Routing (MP-OLSR) protocol. The multipath is obtained using the multipath Dijkstra algorithm. This algorithm amasses extensibility and great flexibility. Their study also implement loop detection with route recovery in MP-OLSR to increase quality of service. Their approach is also compatible with

OLSR. The end-to-end delay and jitter happening during MP-OLSR for quality of service are more accurately necessary for better multimedia services.

1.3 Motivation

The Wireless Sensor Network is a grouping of nodes that are interconnected which offer a best-effort service for data transmission to users using an identified protocol. The current WSNs rely on the end-host congestion avoidance mechanisms of modern TCP to solve traffic congestion and prevent congestion collapse. Present TCP traffic systems look at their individual transmission to point out network packet losses, mark them as in-built congestion flags from intermediate nodes and lower their transmission speed so that congestion is done away with using mechanisms that employ bandwidth adaptation methods. The congestion avoidance mechanism with a bandwidth adaptation tendency of TCP is mainly identified with the Additive Increase Multiplicative Decrease (AIMD) algorithm [11].

Nodes utilise queues that are outbound to take care of traffic eruptions and come up with a high link utilization level in the system. FIFO queues that passively drop incoming packets when queues are full are mostly used by current nodes. However, FIFO drop-tail queues, when they are faced with persistent congestion are often over-provisioned with huge queues to achieve maximum throughput, fill up leading to delays in transmission.

Also, congestion collapse due to a malicious or even unintentional misuse of the network threatens the simple drop-tail queue mechanism [12]. The queues are abundantly static. The result of unstable congestion control adaptation to the pattern of traffic is not yet well researched in networks of today. This adaptation can lead to major developments since it provides another facet that today's TCP does not try to venture into.

The motivation behind congestion control with awareness is to provide high data communication rate with reliability and high efficiency. When the wireless network has less traffic, it is not exposed to congestion, but as the traffic increases, the system is more open to the dangers of congestion and thereby negatively affecting the overall performance of the system [10]. This is the motivation behind the development of a protocol to control and predict congestion. Though there are various sources for congestion, the work concentrate on congestion as a result of multiple transmission. It happens when a high number of sensed data passes through different paths to reach the destination node or sink node, thus leading to bottleneck mainly in intermediate nodes. Applications that require high data-rate can easily introduce congestion problems usually at intermediary nodes. Their proposed method forms

awareness and identifies influx at intermediary nodes and then controls it in order to provide both high reliability and huge data transfer.

Differences of link speeds - Just for the reason that you add fresh high-speed links to a system does not imply that the old low-speed links disappear. Joining high speed and lower speed networks generates congestion difficulties at the interconnect point.

1.4 Research Problem

Mobile ad-hoc networks work under a dynamic topology base and same-time multiple senders share a common path that is prone to congestion. That has motivated the new innovative idea of congestion control methods to solve the problem of congestion with a multipath congestion aware orientation that can prevent a possible congestion scenario. Here, the study uses a novel active queue management system, buffering capacity of node to control the congestion and ensure congestion awareness through the network.

This study will investigate the effects of improved congestion control with awareness on packets in a multipath routing environment. The investigation will be conducted with Network Simulator 2 (NS2). The present study, within the context of congestion control with awareness, will be concerned with the following questions:

1. Does a multipath routing procedure decrease the incidents of congestion in a network as compared to a unipath routing algorithm?
2. Does the procedure of congestion control with awareness have increased throughput, packet delivery ratio and storage and forwarding capabilities difference in comparison to a normal multipath routing protocol?
3. Does the method of congestion control with awareness affect the rate of link failure?

1.5 Scope of Work

This work presents a new approach of Dynamic Queue based congestion control approach for escaping jamming in network messaging flows. In this scheme the store and forwarding capability of nodes are enhanced by varying the queue occupation according to outgoing data. The proposed framework first identifies the level of each queue before listing a packet in the buffer in the network. This is the first test that a packet encounters before enqueue. It is necessary because the buffer is a finite resource that can never be stretched beyond its full capacity. Tests on a queue that can be used for controlling purposes are only performed when

the packet exits the queue by hopping to another node or by being dropped. All computations are local to the respective internetwork nodes.

Beside the active queue manipulations, the implementation of a multipath protocol is in its own respect a novel approach of dealing with the problem of network congestion. Therefore, our proposed protocol is also expected to show an improved performance in relation to the traditional Ad hoc On Demand Distance Vector (AODV) protocol.

Our framework is implemented based on the Ad hoc On Demand Multipath Distance Vector (AOMDV) protocol to form the improved AOMDV with congestion control and awareness (AOMDVCCA).

The AOMDVCCA is expected to fulfil the following objectives after implementation:

1. To increase throughput and packet delivery ratios (Improved Quality of Service – QoS) as a result of reduced packet delay and drop.
2. To enhance storage and forwarding capabilities of nodes due to the variation of queue occupancy with respect to incoming data.
3. To minimise link failure due to the use of a new connection establishment.

CHAPTER 2

AN OVERVIEW OF WIRELESS SENSOR NETWORKS, CONGESTION CONTROL AND AWARENESS

In this chapter we focus generally on the area of wireless sensor networks, congestion control and awareness at length. We first make an insight into what a wireless network and congestion control entail. Next we focus on congestion avoidance and its role in augmenting congestion control. We consider different modeling techniques as applied to congestion control and avoidance. A variety of congestion avoidance models are also explored. We then look at tools that support congestion control process. Finally, we draw a summary of findings from different research papers in the ambit of Congestion Control and Avoidance. In order to simulate the operation of different scenarios, there are a variety of tools that can be used. However, NS-2 [10] and Matlab are the widely used for the simulation of wireless sensor networks projects.

2.1 Wireless Sensor Networks

A wireless ad-hoc network is a grouping of mobile or semi-mobile nodules without an already established set-up, thus coming up with a temporary system. Every node communicate with one another through radio or infrared and possess a wireless interface. Nodules in the ad-hoc network are normally mobile, but may also include stationary nodes, for example, access points to the Internet.

A sensor network application such as that for weather monitoring must have data like temperature, barometric pressure etc., and are known as sensing modalities. There exist different sensors to sense each sensing modality. As depicted in Figure 2.1, various steps involved in WSN applications are generally divided into:

- a. Data Acquisition network
 - Monitor and collect data
 - Assessing and evaluating the information
 - Storing
- b. Data Distribution network
 - Serve useful data to external network (web)

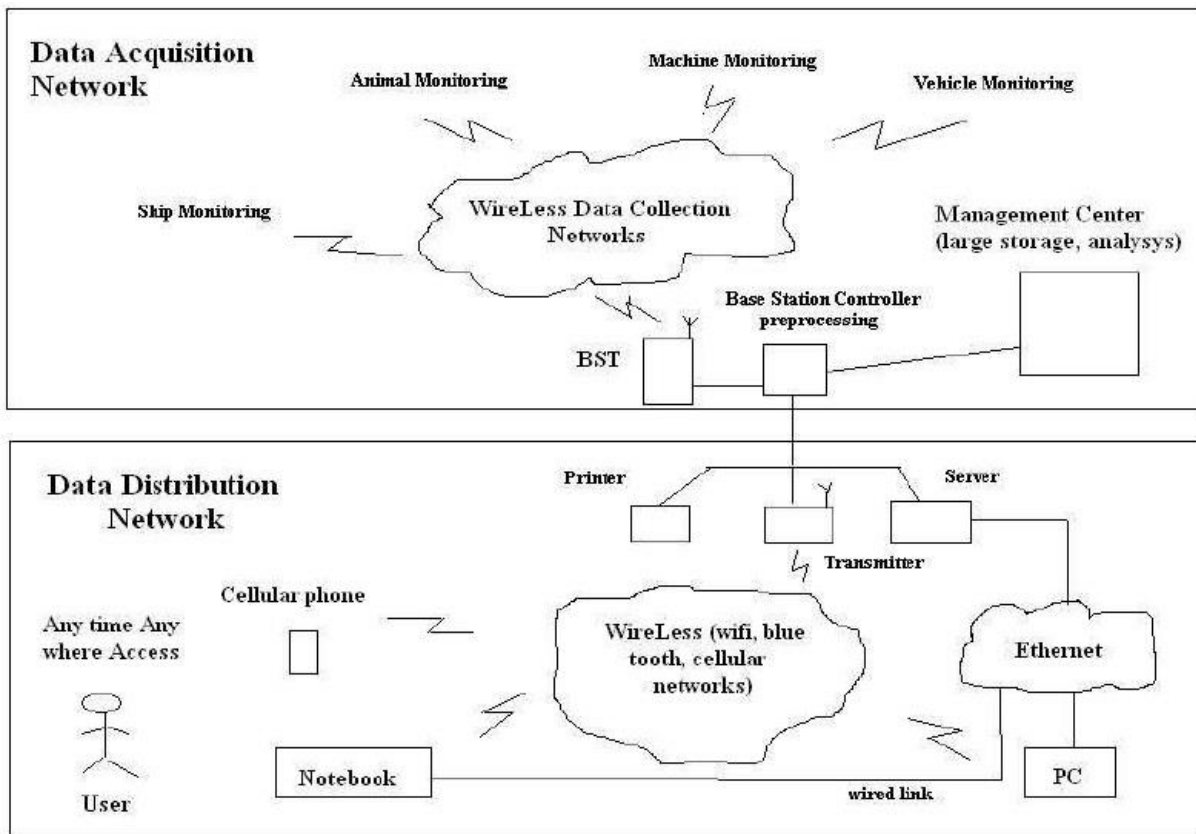


Figure 2.1: Overview of Sensor Networks

In Figure 2.1 above is a generic broad view infrastructure of where wireless sensor networks come into play in the whole network. The networks organise themselves and share data using each other's capability. They connect to other networks outside their own ambit for giving data for analysis and aggregation.

We have already highlighted that an ad-hoc network has no strict administration that is central to the system. This is a favourable feature since in the event that a node dies or collapses, it will not lead to loss of information since other routes can easily be formed. Entrance into and exit out of the network should be easy. Since the transmitter range of each node is limited, there is usually multiple hops to reach intended nodes. All nodes have a multitasking capability since they can act as host and as routers as well. Amongst other duties, a router is an entity that operates a routing protocol. In the conventional sense, a mobile host is an entity that can uniquely be identified with its IP address.

Topological shifts and ill-functions in nodes are well handled in ad-hoc networks. It is fixed using network reconfiguration. For example, if a node causes links to break by going out of the network, new routes to solve the problem by affected nodes are requested. Although this introduces latency, the network remains operational.

2.1.1 Usage

There is no clear cut area where such kinds of networks can be used. Potential uses range from sharing documents at gatherings to enhancements of infrastructure and military operations.

Where there is no infrastructure, for instance, where the internet does not exist, an ad-hoc system could be utilised by a wireless mobile host collection. Cases of such related instances may include disaster recovery people or troops in military situations where there is unavailability of structures because they are just not there or they have been demolished.

Some uses are business confidantes wanting to exchange documents using an identified terminal, or lecture interactions of students during a lecture. The only prelude is for each of the devices to possess a wireless network enabled network interface to enable interactions.

2.1.2 Characteristics

Ad-hoc systems are commonly identified by a constantly changing topology since nodes change their physical locations by intermittently shifting locations. This is an ideal scenario for protocols that discover routes on-demand over the historically known algorithms such as the link state or distance vector [25].

An additional distinguishable feature is a very limited CPU capacity that a host or node has, battery power, storage capacity and bandwidth. This is rightly known also as “thin client”.

The radio environment and the access media also has special characteristics that have to be considered when setting up ad-hoc systems. A case of this can be single direction links. These links come into being for example when two given hosts have strengths that are different on their transmitters, leading a scenario where only one of the node can hear another. Disturbances from the surroundings can also cause this.

A radio scenario multi-hopping may lead to general power gain and transmit capacity gain, as a consequence of the inter-link between the aimed output and the power coverage. Consequently, by using multi-hopping, nodes can transport the packets with a lot lower output power.

2.2 Congestion Control

2.2.1 Congestion

Congestion, in the perspective of networks, denotes to a network form where a node or link transmits considerable data such that it may depreciate network quality, causing queuing delay, data packet or frame loss and the obstruction of new linkages. In a congested network, reply

time slows down with lowered network throughput. Congestion takes place when bandwidth is inadequate and system data traffic surpasses capacity [9].

Data packet forfeiture as a result of congestion is in some measure counteracted by forceful network procedure retransmissions that preserve a network congestion state after decreasing the preliminary data load. This can generate two unchanging states under the identical data traffic load - one dealing with upholding reduced network throughput and the other with initial load.

2.2.2 Causes of Congestion

Congestion can happen because of several reasons. For example, if suddenly a stream of packets arrive on many input streaks and must be out on a single output line, at that moment a long queue can be formed for that output. If there is not enough memory to accommodate these packets, then packets will be dropped. Increasing memory may not help in certain scenarios. Even if a router has an infinite amount of memory, instead of congestion being lowered, it gets worse. This is because by the time packets get at the front of the queue, to be transmitted out to the exit link, they would by now have timed-out recurrently, and replicas may also be in that line. All the packets will then be sent to the next node up to sink. Along the path, only growing the burden to the network. At last, after arriving at the destination, the packet will be thrown away, due to time out. Therefore, instead of being dropped at any intermediate node (in case memory is limited) this type of packet will go all the way up to the destination, unnecessarily increasing the system burden all over and then lastly gets throw down there.

Sluggish processors are also a contributor to congestion. If the node CPU is slow at performing the task asked of them (updating tables, reporting any exceptions, queuing buffers etc.), the queue can size up, although there may be surplus capacity in the line. By the same token, low-bandwidth streaks can similarly lead to congestion. Advancing lines without varying slow processors, or the opposite, often has little impact; these can just move the jam point to another point. The actual problematic thing is the discrepancy amongst different parts of the system. Congestion has a tendency of relying upon itself to get even worse. Nodes respond to overloading by dropping packets. If these packets have TCP segments, the segments will not reach their sink, and they are therefore left unacknowledged, which later leads to timeout and retransmission.

Therefore, the chief source of congestion is regularly the bursty habit of traffic. If the hosts can be coerced into transmitting at the same rate, then the congestion problem will be less common.

Even all other causes will not lead to congestion since other sources just act as a catalyst which boosts the congestion when the traffic is bursty.

This implies that when a device directs a packet and does not obtain an acknowledgment from the recipient, generally it can be presumed that the packets have been thrown down by intermediary devices as a result of congestion. By sensing the rate at which sections are sent but not acknowledged, the source or a middle node can deduce the level of congestion on the network. In the subsequent section we shall focus on the bad outcomes of congestion [13].

2.2.3 Effects of Congestion

Congestion affects two very important variables of network performance, viz. throughput and delay. In simpler terms, the throughput is explained as the fractional exploitation of the system capacity. Figure 2.1(a) demonstrates how throughput responds as available load increases. Initially throughput increases linearly with available load, because utilization of the network rises. However, as the offered load rises yonder a definite limit, say 65% of the network capacity, the throughput dips. If the available load rises further, a point is reached when no sole packet is transported to some destination, which is typically recognised as a deadlock state. There are three arches in Fig. 2.1(a), the finest relates to the condition when all the packets presented are transported to their terminus inside the highest capacity of the system. The next one relates to a situation when there is no congestion control in place. The third one is the circumstance when a particular congestion control technique is applied. This evades the throughput collapse, but offers lesser throughput than the finest condition due to overhead of the congestion control method.

The delay also surges with presented load, as presented in Fig. 2.1(b). Irregardless of what method is applied for congestion control, the delay rises with no limit as the load touches the capacity of the network. It may be detected that firstly there is longer delay when congestion control policy is applied. Nevertheless, the system disadvantaged of any congestion control will flood at a lower presented load [13].

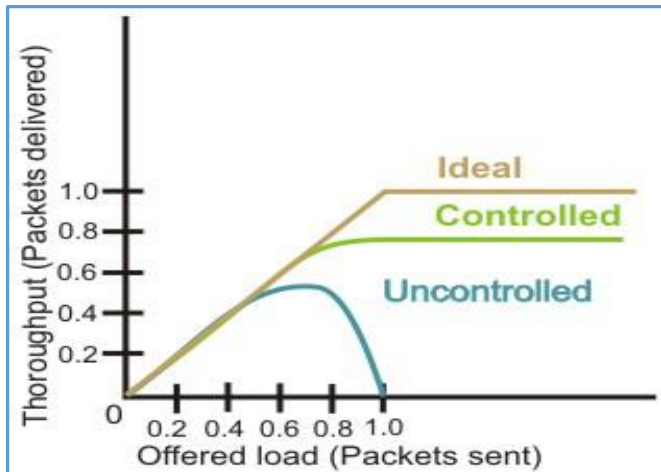


Figure 2.2 (a) Effect of congestion on throughput

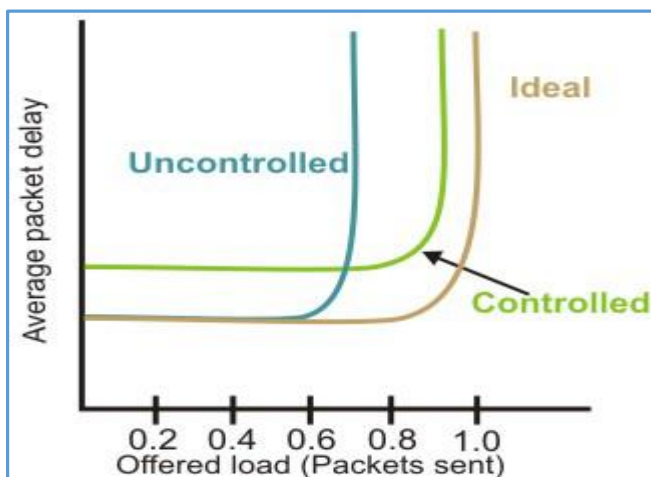


Figure 2.2 (b) Effect of congestion on delay

2.3 Congestion Awareness (Avoidance)

Congestion avoidance methods supervise network traffic loads with a determination to expect and circumvent congestion at usual network bottlenecks. Congestion avoidance is achieved through the dropping of packets.

Amongst the more frequently used congestion avoidance tools is Random Early Detection (RED) that is optimal for high-speed transportation networks. For example, the Cisco IOS QoS comprises of an employment of RED which, when designed, monitors when the node throws down packets. If you don't configure Weighted Random Early Detection (WRED), the node exploits the simpler default packet drop tool named tail drop.

2.3.1 Nature of Congestion Awareness

Conventional congestion control methods assist in the improvement of the performance in the aftermath of congestion. In Figure 2.2 is a simplified graph of response time and throughput of a system as the network load increases. Throughput normally accommodates the load if the

burden is small. Throughput rises as the load increases. As well stated in [14] throughput stops increasing once the load touches the network capacity. In the event that the load is risen further, buffers begin forming, most likely leading to the dropping of packets. When the load rises over this point, throughput may suddenly drop and the network enters into a state of congestion.

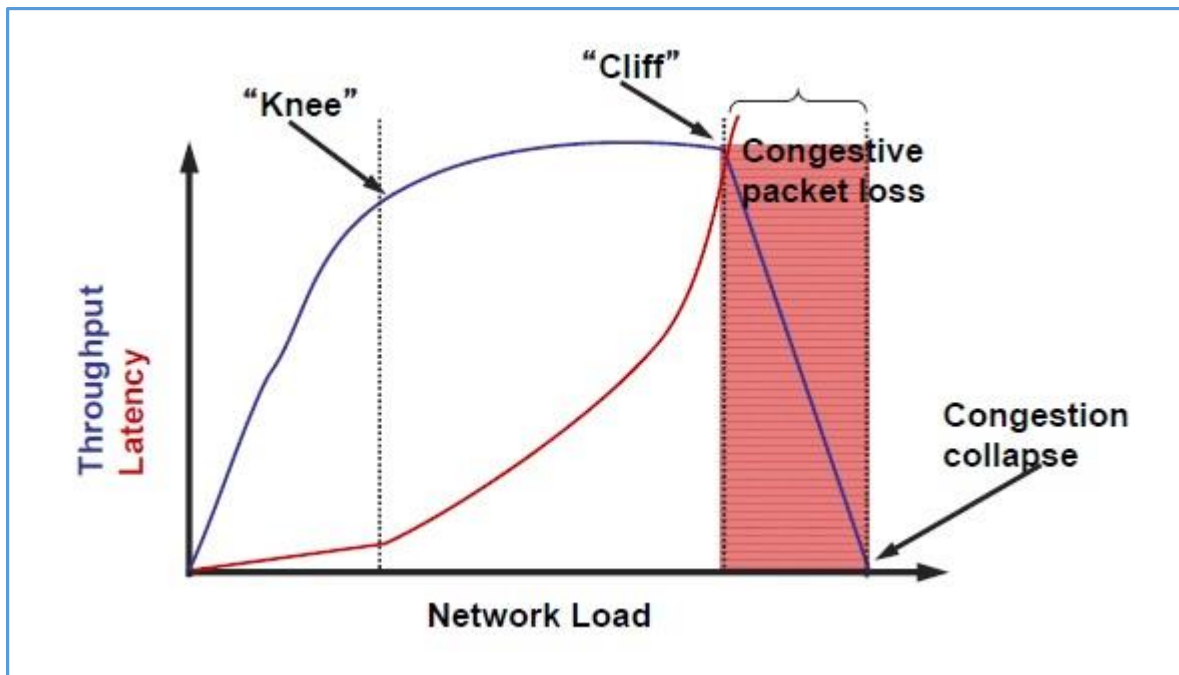


Figure 2.3 Network performance as a function of the load. Dotted curves show performance with deterministic service and inter-arrival times.

The same pattern is followed by the response time curve. Initially the latency rises little as the load increases. When the buffers begin to form, the latency rises in a linear form until, just as the queues begin to overflow, the response time immediately increases.

The moment of congestion collapse is when throughput reaches zero. At this point, latency also reaches infinity. The logic of a congestion control method [14, 15] is to find out that the system has reached the moment of congestion collapse. This will therefore lead to losses of packet and reduces the load so that the system can then return to a state that is not congested.

A cliff is the point of congestion collapse. It is so called because of the fact that the throughput cascades off enormously after that point. If maximizing throughput and consequently reducing the response time is the aim of the system design, then the knee is a healthier point of manoeuvre as shown in the depiction above. After this point, the rise in the throughput is minimal, but the response time notably rises afterwards.

A plot of power [16], as a function of the load is also shown off Figure 2.2. The ratio of throughput to response time is the power. The knee is where the highest point of the power line occurs.

A congestion avoidance system is a scheme that permits the network to work at the knee as distinguished from a congestion control system which endeavours to limit the system on the left side of that cliff. A congestion avoidance method that is well constructed will make sure that the operators are incentivised to increase their load only to an extent where this does not notably affect latency and are made to decrease it if that occurs.

Thus, congestion will not happen as long as the network plays around the knee. Nevertheless, the methods of congestion control are still needed to safeguard the system in the event that it reaches the cliff as a result of ephemeral shifts in the system.

2.4 Congestion Control Techniques

Congestion control methods are machineries and skills used to regulate congestion and uphold the traffic under the capacity of the system. As exposed in Fig. 2.3, the schemes of congestion control can mainly be classified into two wide-reaching classes:

- Close loop: Protocols that permit the network to enter a congested position, discover it, and resolve it.
- Open loop: Protocols to inhibit or evade congestion, guaranteeing that the system (or network in deliberation) will under no circumstances go into a Congested State.

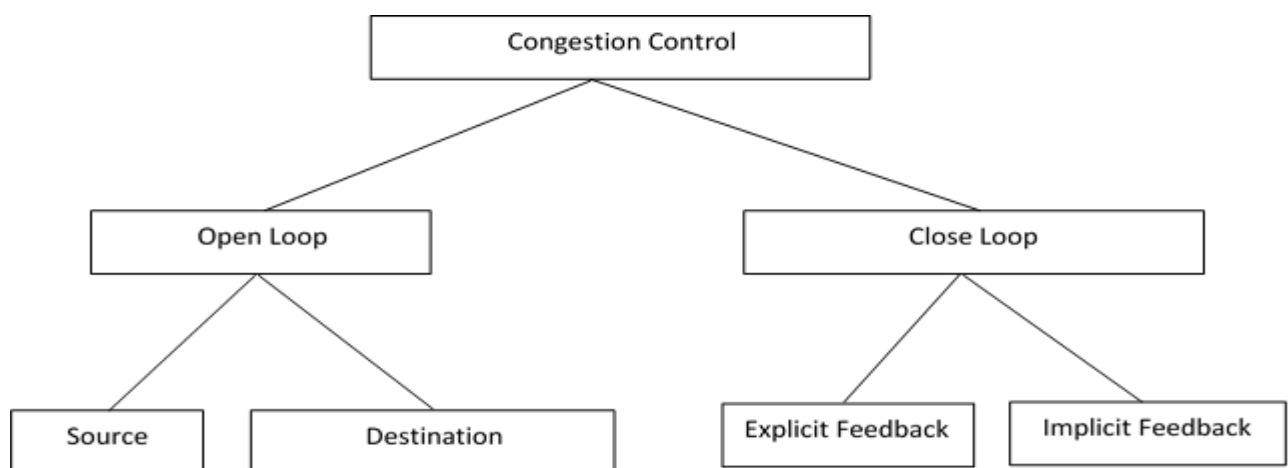


Figure 2.4 Congestion control categories

The first class of solutions tries to solve the problem by using a good design, at first, to ensure that it will not happen at all. When system is up and operating, inter-course corrections are not

applied. These protocols are to some extent naturally static, as the guidelines to control congestion never change considerably with regard to the current state of the system or network configuration. Such approaches are also referred to as Open Loop solutions. These schemes include making a choice about when to receive traffic flow, when to dispose of it, making arrangement decisions among other considerations. The most important thing here is that they make decision regardless of the present state of the system. The open loop procedures are further separated on the basis of whether they act on source or that they act upon the destination.

The second category focuses on the aspect of feedback. Throughout operation, some network constraints are measured and served back to parts of the sub network that can take decisions to moderate the congestion. The approach can be broken down into 3 steps:

- Monitor the network to detect whether it is congested or not and the actual location and devices involved.
- To route this information to the points where actions can be taken.
- Adjust the system's functioning to address the problem.

These answers are referred to as Closed Loop solutions. Different variables can be used to observe the system for congestion. They include: the number of packets that are timed-out, the average queue span, average packet delay, number of packets thrown away or missing due to absence of buffer space, etc. A common feedback step could be, for instance, a node which detects the congestion directs distinct packets to the source (accountable for the congestion) proclaiming the misadventure. These extra packets raise the load at that point in time, but are essential to humble down the congestion at a future stage. Other methods are also used at times to bring down the congestion. For example, hosts or nodes dispatch out investigation packets at consistent intervals to unambiguously ask about the congestion and the source itself will regulate its transmission rate, if congestion is found in the system. This kind of tactic is proactive, since source makes an effort to get knowledge concerning congestion in the network and act commensurately.

An alternative approach may be wherever in the place of conveying information back to the source. A midway node which detects the congestion sends the information about the congestion to the whole network, piggy backed on the outbound packets. This tactic will definitely place an extra load on the network (since it does not send any type of distinct packet

for response). As soon as the congestion has been identified and this message has been delivered to a point where the action needs to be taken, then there are two basic strategies that can solve the difficulty.

These are:

- Either to rise the resources or to reduce the load. For instance, distinct alternate links can be applied to increase the bandwidth in the middle of two points, where congestion takes place.
- One more situation could be to lower the speed at which a specific sender is conveying packets out into the system.

The closed loop procedures can also be separated into two groups, these are, explicit feedback and implicit feedback algorithms. With regard to the explicit approach, distinct packets are directed back to the sources to lessen or reduce the congestion. While in implicit approach, the source performs pro-actively and endeavours to sense the presence of congestion by creating local interpretations [13].

2.5 Congestion Avoidance Schemes

Congestion avoidance and congestion control are supposed to be dynamic system control matters. Just as are other control methods they comprise of two phases: a control apparatus and a feedback apparatus. The feedback apparatus makes the network to notify the operators of the present status of the system. The control approach makes the users to change accordingly their system load.

In a congestion avoidance setup, the feedback signal notifies the operators whether the network is functioning over or under the knee. In a congestion control approach, the feedback indicator tells the operators if the network is operating under the cliff or over the cliff.

Literature has discussed the difficulty of congestion control extensively. A lot of feedback methods have been brought forward. Extending these approaches to signal manipulations around the knee instead of the cliff, we conjure a congestion avoidance apparatus.

The control scheme will also have to be adjusted without fail, to assist the network to function about the knee instead of the cliff. These are the alternatives for the feedback approaches:

1. End-to-end investigative packets forwarded by sources.

2. Every packet has a congestion feedback field that is added in by routers in packets going in the opposite way.
3. Congestion response through packets sent from nodes to sources.
4. A congestion response field is filled in by nodes in packets that are heading in the forward way.
5. Feedback included in the routing communications exchanged amongst nodes.

The initial method is mainly referred to as choke packet [16] or a source quench communication in ARPAnet [17]. It needs bringing in more tracking in the system throughout congestion, although this may not be the best way. Encouraging sources to increase the load during instances when load is below normal level is a viable alternative to this method. If encouragement messages are not there, it insists that there is an overload. The method does not bring more traffic throughout the course of congestion. However, even if there is no problem, it introduces control expenses on the system.

The second approach of raising the cost of congested paths, has been used before in delay-sensitive routing from ARPAnet. The interruptions were found to shift sharply, thereby introducing a high number of stability challenges and routing communications. Moreover, the overhead was not justifiable [15].

Alternative three, probe packets, also suffers from the disadvantage of increased overhead unless probe packets had a dual role of accommodating other information in them. If that was the case, there would be no point not to use every packet going through the network as a probe packet. This may be achievable by reserving a field in the packet that is used by the system to signal congestion. This will lead us to the last two alternatives.

The fourth option is that which uses reverse feedback, needs nodes to piggyback the signal on the packets going in the direction opposite the congestion. The main additive of this approach is that the feedback reaches the source quickly. However, the forward and reverse traffic may be different. The destinations of the reverse traffic may not be the product in the congestion on the forward path or the consequent. Also, many networks have path splitting such that the path from B to A is not necessarily the same as that from A to B.

The fifth alternative, forward feedback, sends the signal in the packets heading in the forward direction. The sink either asks the source to adjust the load or returns the signal back to the source in the packets going in the reverse way.

2.6 Wireless Sensor Networks Development Tools

Presently, the WSN is a highly rated research area. A lot of network information in WSNs is not final and true. It is very expensive to build a WSNs testing ground. The cost and difficulty for performing real experiments on a test bed are very high. Also, repeating actions is largely negatively affected since a lot of factors have a bearing on the experimental results at the same time. It is not easy to isolate a single aspect of your test.

Moreover, it is always time consuming to run actual experiments. Therefore, WSNs simulation is invaluable for WSNs improvement. Schemes, protocols, even innovative ideas can be tested on a very large scale. Operators get a chance to isolate different factors by changing configurable parameters in WSNs simulators.

As a result, simulation is critical to the study of WSNs, being the stand-alone method to test fresh ideas and systems in the arena. This has resulted in the latest explosion of simulator design.

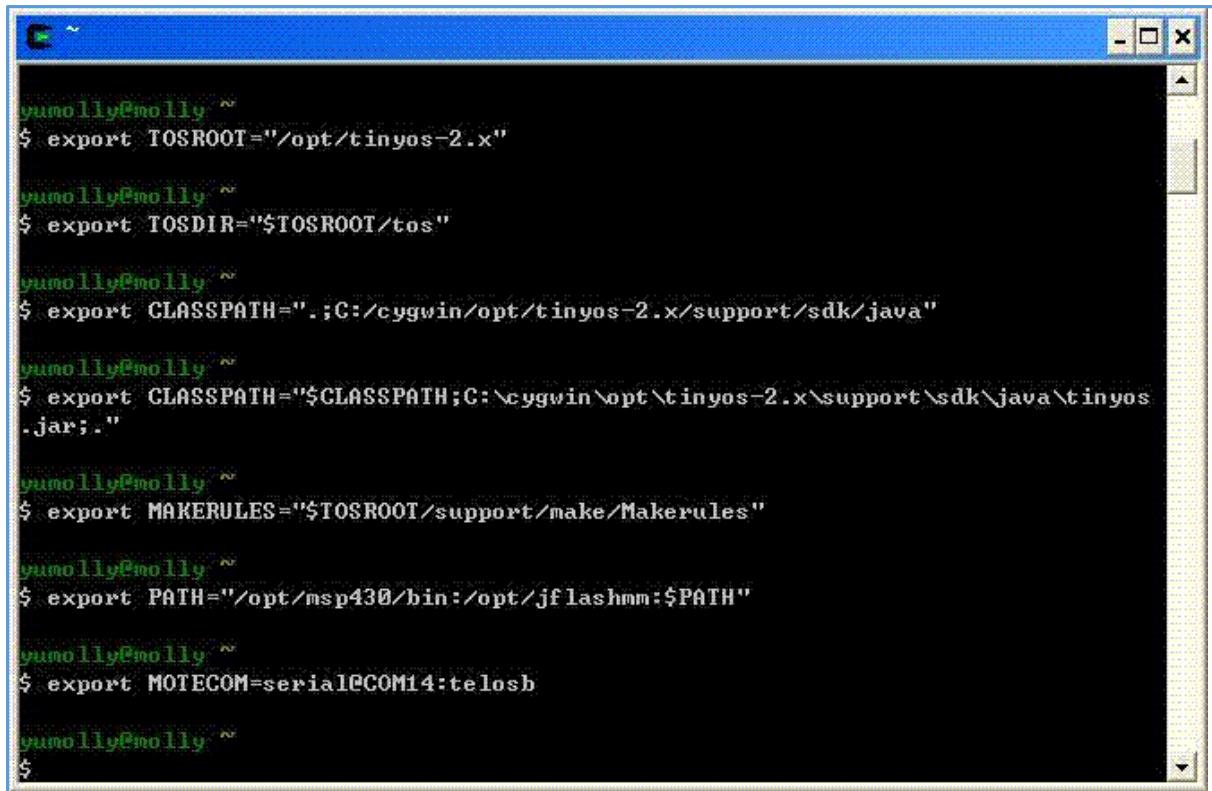
However, it is not a meek task to obtain tangible suppositions from a simulation learning. WSNs simulators have two vital facets namely:

- (1) The appropriateness of a specified tool to use the model and,
- (2) The simulation models' exactness.

To produce results that are trustworthy a "correct" model founded on concrete assumption is mandatory. The all-important trade-off becomes: exactness and requirement of specifics versus execution and scalability. In this section, several main-stream WSNs simulators are described.

2.6.1 NS-2

NS-2 [20, 21] stands for Network Simulator version two, which was principally designed in 1989 named REAL network simulator. At present, NS-2 is supported by the National Science Foundation and Defense Advanced Research Projects Agency (DARPA). NS-2 is constructed in Object-Oriented addition of the Tool Command Language (Tcl) and C++ as a discrete event network simulator. It can be run on Cygwin, presented in Figure 2.4 below or on Linux Operating Systems. This is a Unix-resembling atmosphere and command-line environment operating on Windows. NS-2 can be used in both wired and wireless systems as a prevalent non-precise network simulator. This simulator is an open source and documentation is readily available online.



```
yumolly@molly ~  
$ export TOSROOT="/opt/tinyos-2.x"  
  
yumolly@molly ~  
$ export TOSDIR="$TOSROOT/tos"  
  
yumolly@molly ~  
$ export CLASSPATH=".;C:/cygwin/opt/tinyos-2.x/support/sdk/java"  
  
yumolly@molly ~  
$ export CLASSPATH="$CLASSPATH;C:\cygwin\opt\tinyos-2.x\support\jdk\java\tinyos.jar;."  
  
yumolly@molly ~  
$ export MAKERULES="$TOSROOT/support/make/MakeRules"  
  
yumolly@molly ~  
$ export PATH="/opt/msp430/bin:/opt/jflashmm:$PATH"  
  
yumolly@molly ~  
$ export MOTECOM=serial@COM14:telosb  
  
yumolly@molly ~  
$
```

Figure 2.5: Cygwin

2.6.2 EmStar

EmStar [21] is an emulator explicitly intended for WSN constructed in C, and it was initially designed by University of California in Los Angeles. EmStar is identified as a trace-driven emulator [22] running in real-time.

This emulator, can be operated on Linux operating systems. It supports the development of WSN applications on improved hardware sensors. In addition to tools, services and libraries, an addition of Linux micro-kernel is also included in the EmStar emulator.

2.6.3 OMNeT++

OMNeT++ [23] is a discrete event network simulator set up in C++. OMNeT++ offers both a non-moneymaking license, used at educational organisations or non-profit creation research establishments, and a moneymaking license, operated at commercial environments. This simulator supports a module programming model. OMNeT++ simulator can be run on Windows, Unix-like systems and Linux Operating Systems. OMNeT++ can be used in both wired and wireless networks. Most of the structures and simulation models in OMNeT++ are resources available as open source.

2.6.4 TOSSIM

TOSSIM [19, 20] is an emulator exclusively designed on TinyOS for WSN implementation: this is an open basis operating system for embedded operating systems. TOSSIM was firstly established by UC Berkeley's TinyOS taskforce crew in 2003. It is a non-continuous bit-level event network emulator created in Python, a high-level programming language focusing on code readability, and C++. The simulator can be run on Cygwin on Windows or on Linux Operating Systems. TOSSIM is also has open source software and online documents available for unlimited access.

CHAPTER 3

WIRELESS SENSOR NETWORK ROUTING PROTOCOLS

The area of routing protocols is widespread with respect to wireless, quasi-wireless and wired networks. Here, we look very closely at an outlook of wireless sensor network routing protocols. They are primarily classified into Path establishment Based Routing Protocols, Network Based Routing Protocols, Operation Based Routing Protocols and Next-Hop Selection Based Routing Protocols. The most prominent distance vector protocol is the Ad hoc On Demand Distance Vector [4] which led to the development of its more powerful multipath counterpart – the Ad hoc On Demand Multipath Distance Vector [6]. The operation of both protocols is almost similar save for the fact that the later computes more disjoint paths to the same destination. This gives the multipath approach an edge over the former approach which is basically unipath [3, 4].

3.1 Routing Protocols

Routing is the art of choosing finest paths in a system. Historically, the term routing was also used to imply despatching network traffic amongst networks. Nonetheless this latter purpose is much better defined as merely forwarding. Routing is executed for many types of networks, as well as the telephone system (circuit switching), electronic data systems (for instance the Internet), and transportation links. This study is concerned principally with routing in packet switching tools for electronic data networks [30].

In packet switching setups, routing provides guidance in packet forwarding (the transfer of logically identified system packs from their source in the direction of their eventual destination) via intermediate nodes. Middle nodes are characteristically network hardware tools for example routers, gateways, bridges, switches, or firewalls. Universal-purpose computers can also advance packets and execute routing, although they are not dedicated equipment and may be hurt from inadequate performance [7]. The routing procedure generally guides forwarding on the foundation of routing tables that preserve a record of the paths to numerous network endpoints. Consequently building routing tables that are held in the router's memory is very imperative for well-organized routing. A majority of routing algorithms utilise only one network pathway at a given time. Multipath routing methods permit the use of multiple alternate paths.

WSN Routing Protocols can earnestly be put into five classes according to:

- the method used in forming the routing paths,
- the structure of the network,
- the protocol operation,
- the initiator of communications,
- how a protocol selects a next-hop on the link of the message that has been despatched, as diagrammatised in Figure 3.1 below.

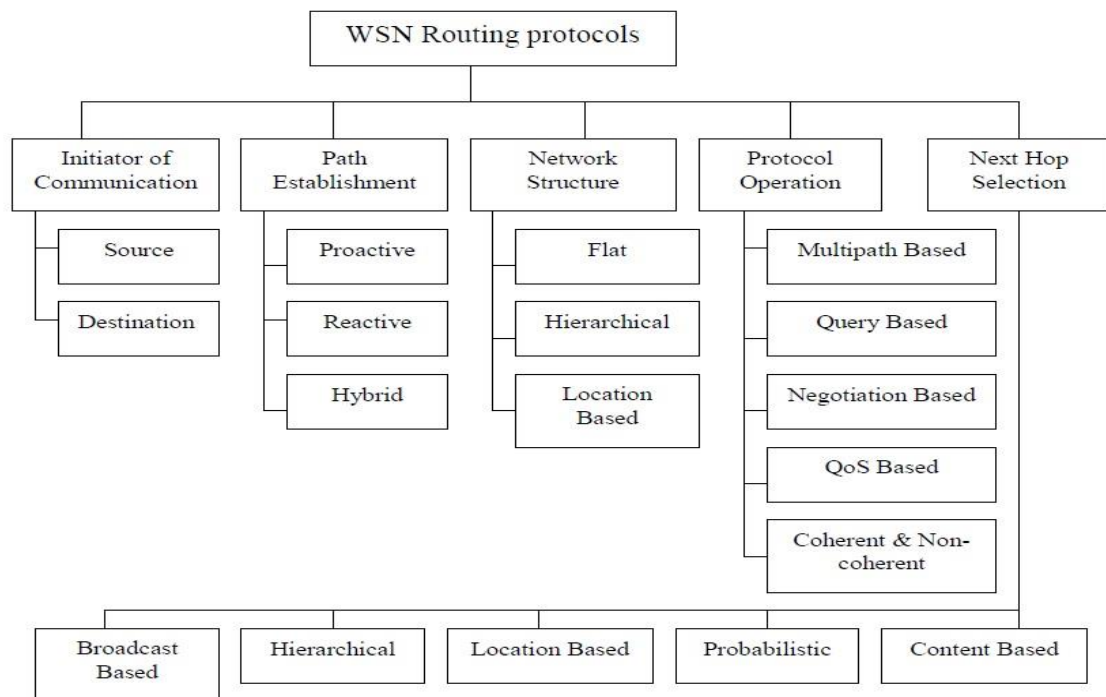


Figure. 3.1: WSN Routing Protocols

3.1.1 Path establishment Based Routing Protocols

Routing paths are brought into existence using the three methods - proactive, reactive or hybrid. Proactive protocols calculate all the paths in advance of them being readily needed and accumulate these paths in a routing table in every node. Reactive protocols come up with routes the instant when they are required. Hybrid protocols utilise a mixture of these two approaches [30].

3.1.1.1 Proactive Protocols

Proactive routing methods are those that maintain unswerving and precise routing tables of all the nodules in the system making use of periodic forwarding of information for routing. In this approach of message forwarding, all paths are defined or set up before they are needed. Most

of these routing protocols may be utilised both in hierarchic structure and flat systems. The capability to calculate ideal route which needs overhead to calculate is not tolerated by most environments. Nevertheless, it is one of the positives of flat proactive routing. In order to meet the routing demands for bigger ad hoc networks the preferred solution is the hierarchal proactive routing [27].

3.1.1.2 Reactive Protocols

Reactive routing methods are known for not keeping safe the comprehensive information of every node in a system. The route formation concerning source and destination is founded on its vibrant exploration according to the requests instead. For it to design a route from source to sink, the reverse path and a route discovery query is applied for the replies to query queries.

Therefore, in routing protocols that are reactive, choosing of the route is on-request by means of route querying beforehand path establishment. These approaches differ in twofold: by creating again and calculating again the path in the event of disjointness occurring and by lessening communication expenses from introducing flooding on systems [27].

3.1.1.3 Hybrid Protocols

This strategy is used in large networks. Hybrid routing strategies include both proactive and reactive routing approaches. It uses a clustering technique that guarantees a stable and scalable network. The network cloud is partitioned into a lot of clusters which are preserved with dynamism if a node is added or has left a particular cluster.

This approach uses a proactive technique when routing is required inside clusters and reactive method when routing is required across the clusters. Hybrid routing exude network overhead needed in maintaining clusters [27].

3.1.2 Network Based Routing Protocols

Protocols classification in line with the setup of a system for the needed operation is very crucial. Depending on their functionalities, the protocols that fall into this class are further divided into three subcategories. A brief highlight of these protocols is given below [26].

3.1.2.1 Flat-Based Routing

Flat based routing is required in the case that a huge amount of sensor nodes are needed and every node executes an identical role. Since there is a high number of sensor nodes, it is unfeasible for each node to be given a particular identification. At last, a data-centric routing

tactic in which a base station directs query to a cluster of certain nodes in a region and awaits for reply is formed. Routing protocols that are Flat-based include [26, 28, and 29]:

- Energy Aware Routing (EAR).
- Directed Diffusion (DD).
- Sequential Assignment Routing (SAR).
- Minimum Cost Forwarding Algorithm (MCFA).
- Sensor Protocols for Information via Negotiation (SPIN).
- Active Query forwarding in sensor network (ACQUIRE)

3.1.2.2 Hierarchical-Based Routing

Hierarchical-based routing is most relevant in instances where network scalability and efficient communication is needed. Cluster based routing is what it is also referred to as. In a hierarchical-based routing there is high energy efficiency in which high energy nodes are randomly chosen for handling and directing data. Little energy nodes are utilised for detecting and directing data to the cluster heads. Minimum energy, lifetime and scalability of network is a notable characteristic of hierarchical-based routing.

The following are examples of routing protocols that are hierarchical-based; [28, 29, 30].

- Hierarchical Power-Active Routing (HPAR).
- Sensor information systems with efficient power gathering.
- Threshold Sensitive Energy Efficient Sensor Network Protocol (TEEN).
- Minimum Energy Communication Network (MECN).

3.1.2.3 Location-Based Routing

Sensor nodes are randomly scattered in an area of interest and most identified by the geographic position in which they are operating in this nature of network arrangement. Utilisation of GPS is used to mainly locate them. The signal strength coming from those nodes and coordinates are computed by sharing information amongst nodes in the neighbourhood to estimate space between nodes.

Examples of location-based routing networks are; [28, 29, 30]

- Sequential Assignment Routing (SAR).
- Ad-Hoc Positioning System (APS).
- Geographic Adaptive Fidelity (GAP).

- Greedy Other Adaptive Face Routing (GOAFR).
- Geographic and Energy Aware Routing (GEAR).
- Geographic Distance Routing (GEDIR).

3.1.3. Operation Based Routing Protocols

Depending on their functionalities, WSNs applications can then be grouped. Therefore, routing protocols are categorised according to their operations to equate to these functionalities. Attainment of ideal performance and to save the few resources of the network is the reasoning behind their categorisation [30].

3.1.3.1 Multipath Routing Protocols

Protocols falling in this class, as its name entails, offer multiple route selection for a message to arrive at a destination, thereby raising performance of the network and lowering its delay. Network reliability is achieved because of raised overhead. There is a greater use of energy since network paths remain alive by sending periodic messages. Multipath routing protocols are [30]:

- Sensor Protocols for Information via Negotiation (SPIN).
- Multi path and Multi SPEED (MMSPEED).

3.1.3.2 Query Based Routing Protocols

Forwarding and receiving queries of data is what this group of protocols focuses on. The sink node forwards a query of interest from a node through the network. The node with this concern equals the query and direct back to the node which started the query. The query usually utilises languages of a high level.

Instances of query based routing protocols include [30]:

- Directed Diffusion (DD).
- COUGAR.
- Sensor Protocols for Information via Negotiation (SPIN).

3.1.3.3 Negotiation Based Routing Protocols

A high level data descriptors to eliminate delayed and unwanted data transmission through negotiation is what is utilised by this group of protocols. Intelligent decisions for

communication are performed by such protocols or other actions relying on facts to ascertain the level of available resources. Routing protocols that are negotiation based include [28]:

- Sensor Protocols for Information via Negotiation (SPIN).
- Sequential Assignment Routing (SAR).
- Directed Diffusion (DD).

3.1.3.4 QoS Based Routing Protocols

The network needs to possess a balanced approach for the QoS concerning application in this class of routing. The application can therefore defer delicate data so as to accomplish this QoS numeric. Another metric in the form of energy consumption has to be looked for the network when communicating to the base station. To achieve QoS the cost function for the desired QoS therefore, also has to be taken into account. Common examples of such routing are: [28]

- SPEED.
- Multi path and Multi SPEED (MMSPEED).
- Sequential Assignment Routing (SAR).

3.1.4 Next-Hop Selection Based Routing Protocols

3.1.4.1 Content-based routing protocols

These protocols compute the next-hop on the route solely based on the contents of the query. The architecture of sensor networks commonly uses this class of routing protocols since specific nodes are not queried by a base station. It rather requests only for data regardless of where it is emanating from [27, 29].

- Directed Diffusion (DD).
- Energy Aware Routing (EAR).
- GBR.

3.1.4.2 Probabilistic routing protocols

The assumption of such protocols that all sensor nodes deployed at random are the same. Sensor nodes randomly choose the neighbor of next-hop for each communication to be dispatched when using this routing protocol. The possibility of selecting a certain neighbor is inversely related to its cost [27].

- Energy Aware Routing Protocol.

3.1.4.3 Location-based routing protocols

These protocols choose the next-hop towards the sink based on the known location of the neighbors and the sink. The position of the destination may signify the centre of a region or the actual position of an identified node. Location-based routing protocols avoid the communication expense emanating from flooding, but the computation of the locations of neighbors may lead to extra overhead. The local minimum problem is present in all devolved location-founded routing protocols: it may be a circumstance where all neighbors of an intermediary node are far away from the terminus than that node. So as to avoid this difficulty, each protocol practises diverse routing procedures [27, 30].

- GEAR (Geographical and Energy Aware Routing).

3.1.4.4 Hierarchical-based routing protocols

With respect to hierarchical protocols, all nodes send a message for a node (also named aggregator) that dwells in an upper hierarchy class in comparison to the forwarder. Each node combines the incoming data, thereby reducing the communication burden and keeps more energy.

Thus, these protocols increase the system lifetime and are also more scalable. The group of nodes that send to an identical aggregator is referred to as cluster, and the aggregator is also called the cluster head. Cluster heads are highly resourced nodes, where resource generally refers to their residual energy level which is generally higher than the average. This is so because they are traversed by a high tracking system and they execute more computation (aggregation) as compared to other nodes in the cluster [30].

One layer selecting cluster heads and the other layer performing routing in a generally a double-layered routing is a common feature of hierarchical routing [27, 29].

- LEACH (Low Energy Adaptive Clustering Hierarchy) protocol.

3.2 Ad hoc On-demand Distance Vector Routing

Developing an on-demand multipath protocol enhanced with congestion control and awareness that provides the advantages of multiple paths without suffering from many additional overhead is the goal that this study is to fulfil. We present an on-demand multipath distance vector protocol in the framework of AODV [32, 33]. Compared to DSR in recent performance studies [34, 35], particularly in extensive networks, AODV has demonstrated competitive or

superiority in performance. A multipath protocol termed Ad hoc On-demand Multipath Distance Vector (AOMDV) [31] was developed from AODV. We refer to the resulting protocol in our study as Ad hoc On Demand Multipath Distance Vector with Congestion Control and Awareness (AOMDVCCA) – discussed in detail in Chapter 4. Delivering effectual fault tolerance with wisdom of faster and efficient regaining from route catastrophes with an improved congestion control and awareness mechanism is the primary objective behind the design of AOMDVCCA.

To articulate a single path, loop-free, distance vector, on-demand protocol, AODV syndicates the usage of DSDV [36] destination sequence numbers just like the on-demand route discovery technique of DSR [37]. In contrast to DSR, AODV uses hop-by-hop routing instead of source routing. We assess some of the fundamental structures of AODV to offer appropriate background for AOMDV described in the next section further down.

3.2.1 Route discovery

A traffic source starts a route discovery process if it requires a route to a destination. A network-wide flood of a route request (*RREQ*) for the endpoint and ahead of a route reply (*RREP*) is what route discovery basically entails. At all intermediate nodes, duplicate copies of *RREQ*s are disregarded. In each *RREQ* that it generates, the source includes a strictly increasing broadcast id. The source id and the broadcast id of the *RREQ* are used to identify duplicates. After receiving a non-duplicate *RREQ*, an intermediate node first forms a reverse path to the source utilising the *RREQ*'s earlier hop of as the subsequent hop on the opposite path. The intermediate node generates a *RREP* once a valid route is present, otherwise the *RREQ* is rebroadcast. The destination instigates a *RREP* upon the destination receiving a non-duplicate *RREQ* for itself. Through the reverse path, the *RREP* is passed back to the source node. A forward path to the destination is formed during the procession of the *RREP* towards the source.

```

if (seqnumdi < seqnumdj) or
    ((seqnumdi = seqnumdj) and
     (hopcountdi > hopcountdj)) then

    seqnumdi := seqnumdj;
    hopcountdi := hopcountdj + 1
    nexthopdi := j

endif

```

Figure 3.2: AODV route update instruction. This is utilised whenever a node i receives a route update to a destination d from a certain neighbor j . The $seqnum^{d_i}$, $hopcount^{d_i}$ and $nexthop^{d_i}$ variables stand for the sequence number, the destination hopcount, and the nexthop respectively, for a destination at node i which is d .

Observe that the route discovery process that has just been explained requires that between the source and the sink a bidirectional path be present. A technique to form at least one such bidirectional path given a pool of unidirectional links is explained by the current AODV specification [33].

3.2.2 Sequence Numbers and Loop Freedom

Loop freedom is guaranteed by the use of sequence numbers in AODV. A sequence number is maintained by every node for itself that monotonically increases. It keeps track of the maximum sequence numbers known for every terminus in the routing table (known as “destination sequence numbers”). Destination sequence numbers enable a mechanism to find the relative freshness of two items of routing information originating from two different nodes for the same sink by accompanying all routing messages. Destination sequence numbers monotonically rising along a valid route is an invariant that the AODV protocol follows, thereby avoiding routing loops. The following section further explains this as it will define a crucial role in the operation of the multipath protocol [31].

A node can obtain a routing appraisal through a *RREQ* or *RREP* packet in AODV, thereby establishing a forward or reverse path. The update rule is instigated whenever such a routing packet is received as shown in Figure 3.2. Loops cannot be formed as a concrete certainty. Consider the tuple $(-seqnum^{d_i}, hopcount^{d_i})$ where $seqnum$ characterises the sequence number on node i aimed at the destination d . In a similar fashion, $hopcount^{d_i}$ is the hopcount to the sink

d commencing from node i . The route update law in Figure 3.2 imposes that j being the downstream node, for whichever two succeeding nodes i and j on a legal pathway to the terminus;

$$(-seqnum^d_i, hopcount^d_i) > (-seqnum^d_j, hopcount^d_j),$$

Where lexicographic sense is used in the comparison. Thus, the tuples $(-seqnum^d_i, hopcount^d_i)$ which in turn points to a loop freedom since in any valid path are in a lexicographic total order.

3.2.3 Use of Soft State and Route Maintenance

AODV utilises a timer-based approach to discard stale routes immediately. Each routing entry is linked with a soft state timer referred to as a route expiration timeout. This timer is refreshed every time a route is used. At staggering intervals, most recently expired routes are invalidated. Maintenance of the route is executed using route error (RERR) packs. When a linkage botch is identified, routes to sinks that become unreachable are discarded.

A *RERR* is a broadcast that has the list of unreachable destinations with their sequence numbers. The *RERR* forwarding mechanism ensures that all sources using the downed link receive the *RERR*. When a node is not able to send a data packet because of lack of a route, *RERR* is also formed. Upon receiving a *RERR* from a downstream neighbor for some destination, a node invalidates the corresponding route and updates the sequence number from the *RERR*. If at least one destination becomes unreachable, *RERR* is re-broadcast [31].

3.3 Ad Hoc On-Demand Multipath Distance Vector Routing

AOMDV and AODV have a lot of common features. AOMDV relies on the idea of the distance vector and uses hop-by-hop routing theme. AOMDV uses a route discovery approach to establish routes on demand. The number of routes found per route discovery is the main difference. *RREQ* passing from the source towards the destination in AOMDV forms many reverse paths at intermediary nodes in addition to the destination.

Numerous *RREPs* travel through these reverse paths in the opposite direction to form multiple forward links to the sink at the intermediate and source nodes. Observe that AOMDV also offers alternate paths to intermediate nodes since they are important in reducing frequency of route discovery [41].

Making sure that multiple routes found are free of any loops and that they are indeed disjoint, and in efficiently computing such paths by use of a flood-centred route discovery is the backbone of the AOMDV protocol. Locally at every node, route update rules of AOMDV are used to offer a key role in keeping disjointness and loop-freedom features. The underlying AODV protocol is what AOMDV relies mainly on for the routing information already present, thereby reducing the expense incurred in coming up with multiple paths. In particular, it does not particularly make use of any distinguishable control packets. The only added overhead in AOMDV compared to AODV is from additional *RREPs* and *RERRs* precisely meant for multipath detection and upkeep along in addition to limited additional fields in control packets for routing (i.e., *RREPs*, *RREQs*, and *RERRs*).

3.3.1 Loop Freedom

A node cannot have more than one path for each destination because AODV route update rules (*Figure 3.2*) inhibit that. Alterations are therefore necessary to these route update rules so as to achieve an excess of one path for each end point at a node. However, these alterations must be executed in a way that ensures the preservation of loop freedom. When calculating multiple loop-free paths at a given node for a destination, two matters arise. Firstly, of the multiple paths, which one shall a node advertise or suggest to the rest? A random choice can result in loops since each of these routes may have different hop counts. Secondly, of the advertised paths, which one a node should accept? Accommodating all paths without serious considerations may introduce loops, once more [41].

Figure 3.3, below, depicts these shortcomings using modest illustrations. In *Figure 3.3(a)*, *D* is the sink node and node *I* has two paths to *D*—a five hop link via node *M* (*I – M – N – O – P – D*), and a direct link with a single hop (*I – D*). Suppose that *I* advertises the path *I – M – N – O – P – D* up to node *J* and then the link *I – D* to node *K*. At that point both *J* and *K* possess a link to *D* via *I*, however, each of them has a different hop count. Later, if *I* gets a four hop path to *D* from *L* (*L – K – I – D*), *I* cannot determine or establish whether *L* is downstream or upstream to itself, since in the route advertisements, it is just the hop count data which is used.

So *I* computes a link through *L* leading to a loop. Such a scenario happens due to the fact that a node (*I* in this case here) advertises a smaller path (*I – D*) when there is also (*I – M – N – O – P – D*), which in this case is an optional longer path.

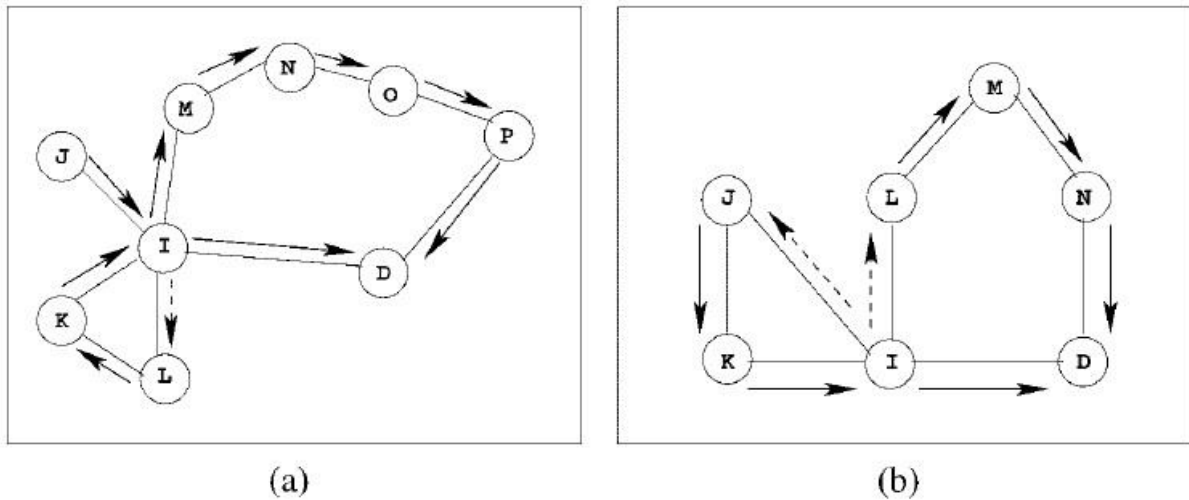


Figure 3.3: Possible routing loop situations examples with multiple path calculation.

Figure 3.3(b) illustrates another probable loop scenario. Node *J* possess a three hop path to *D* through *K* (*J* – *K* – *I* – *D*) where node *D* is the sink. Node *L* likewise has a three hop route to *D* through *M* (*L* – *M* – *N* – *D*). For instance, given that *I* acquires a quadruple hop link to *D* from *L*. In such a case, *I* cannot conclude whether *L* is an upstream node or not since *J* can also offer a four hop path to *D*. A loop in the routing may be caused by accepting a longer path after having advertised a shorter path to neighbors given this scenario.

Below is a formulation of a set of sufficient conditions that will guarantee loop-freedom, based on this argument above. Multiple paths are maintained at a node meant for a destination under these conditions [41].

Sufficient Conditions

1. *Rule of the sequence number*: exclusively for the highest identified destination sequence number it keeps paths for them. There is a restriction for every destination, that multiple paths kept by a node have an identical destination sequence number. A loop freedom invariant similar to AODV can be maintained with this limitation. At the moment when a route announcement comprising a greater destination sequence number is received, every route corresponding to the older sequence number is let fall. Nevertheless, as in AODV, different nodes (on a link) may possess different sequence numbers for the identical terminus.
2. For the same destination sequence number,

(a) Route acceptance rule: Under no circumstances should it consent to a route longer than one already advertised.

(b) Route advertisement rule: Under no circumstances should it advertise a route smaller than one previously advertised.

For the identical sequence number, in order to keep multiple paths, the idea of an ‘advertised hop count’ is utilised by AOMDV. A variable named ‘*advertised_hop_count*’ for every destination is kept by every node. The length of the ‘longest’ path available for the destination at the moment of first advertisement for a certain destination sequence value is equated to this variable. Until the sequence number varies, the announced hop count does not change. An additional number of alternate paths to be maintained by advertising the longest path length.

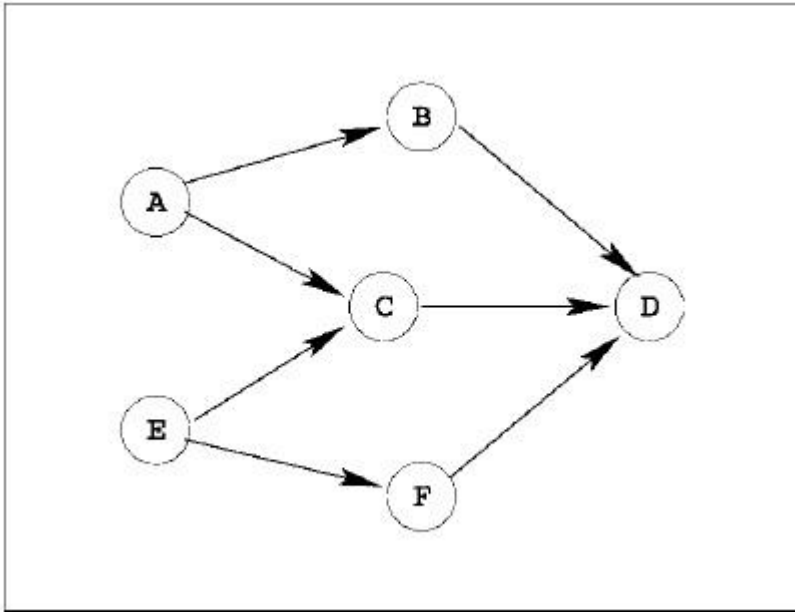
3.3.2 Disjoint paths

AOMDV aims to discover disjoint alternate paths, besides maintaining multiple paths that are loop-free. Disjoint paths are an important choice for selecting a subset of different paths from a potentially large group for the purpose of increasing fault tolerance using several paths, because the possibility of their correlated and simultaneous failure is lower as compared to alternate paths that are overlapping [41].

We look at two kinds of disjoint paths: link disjoint and node disjoint. Whereas node-disjointness additionally excludes common intermediate nodes, link disjoint group of paths between a pair of nodes do not have common links.

The idea of disjointness is restricted to one pair of nodes and doesn’t take into account disjointness across different node pairs as differentiated from the general disjoint paths delinquent found in algorithms works and graph theory. Precisely, all links that can be mapped out from P to D are disjoint at any node P , for a sink D . This does not fundamentally imply that all paths that occur in the network leading to D are necessarily disjoint. We demonstrate this difference using an example in Figure 3.4, for simplicity.

Observe that from the fault tolerance perspective, this limited idea of disjointness is satisfactory and is generally used in a lot of disjoint multipath routing protocols [43, 44, and 45].



*Figure 3.4: Paths may not be communally disjoint simply because they are preserved at different nodes to a sink. Here **D** is the sink. Node **A** devises two disjoint paths to **D**: **A – B – D** and **A – C – D**. Correspondingly, node **E** devises two disjoint paths to **D**: **E – C – D** and **E – F – D**. But the paths **A – C – D** and **E – C – D** are not disjoint since they share a communal link **C – D**.*

We do not overtly adjust either distance of other paths or cardinality in discovering disjoint paths. The crescendos of the route discovery procedure are the ones that mainly compute the quality of disjoint paths and number learnt by AOMDV. However, by setting a length preserved at each node and maximum cap on number of alternative links, it is conceivable to take over control of these attributes. Given the questionably higher overheads associated with distributed calculation of ideal and class of different disjoint links short-lived nature of links in mobile ad hoc networks this tactic is vindicated.

Here we explain the main idea behind node disjoint path computation [40]. AOMDV can discover either link or node disjoint links.

3.3.2.1 Route Discovery Phase

A source node looks up the routing table for the next-hop en route for the destination of the packet when it wants to send a packet. In the event that there is a lively entry for the sink in the routing table, the data packet passes on to the next hop. Or else the route discovery stage starts. Paths are found by means of the two types of control messages during the route discovery stage,

(a) Route reply messages (*RREPs*) and

(b) Route request messages (*RREQs*).

RREQ messages are inundated into the system by the source node. Every transitional node that obtain a *RREQ*, probes a record in the Seen Table to verify whether it has a duplicate node or a new one by one. The table accumulates the two entries of the source IP address and *RREQ* flooding ID, which recognises a *RREQ* message in the system distinctively. An entry is deliberated as a duplicate *RREQ* message and thrown away without further dissemination if it is existing in the Seen Table for the received *RREQ* message. The node also brings its routing table up-to-date for forward path prior to disseminating the *RREQ* message and forms a record in the Seen Table [40].

Source IP Address	Flooding ID	Seen Table
195.1.1.1	22.231.113.64

Figure 3.5: AOMDV Seen Table Structure

By using this concept if some route gets abortive owing to hefty congestion or traffic then a substitute path will be pursued, since it has been well-defined the AOMDV Table. Under these cases we will calculate the level of cramming using a threshold-value of congestion.

In AOMDV, singularly the destination node can forward *RREPs* after receiving a *RREQ* message. Even if they possess an active route to destination, the transitional nodes are prohibited to send *RREPs*. This is a good practised done in order to get the node-disjoint paths. The terminal node has to send a *RREP* communication for each *RREQ* acknowledged in AODV, even if the *RREQ* is a matching one. As revealed in Figures 3.5 and 3.6, we alter the *RREP* message and data structure for Seen Table. We improve I the Seen Table with an extra field that operates as a flag recognised as seen-flag. At start, this flag is set to FALSE i.e. after a node acquires its first *RREQ* message at the moment of inserting an entry in the Seen Table. In AODMV the *RREP* messages instigated by destination node encompass one extra field identified as broadcast ID (*b id*).

A terminal node generates the conforming *RREP* message when it obtains a *RREQ* message, it. Into the *b id* field of forwarded *RREP* message, the terminal node counterfeits the *f id* from the received *RREQ* message. The *RREP* is unicast en route to the instigator of the *RREQ* by means of the reverse path to construct the forward path. The sink does the above mentioned process for all *RREQs* received. The middle nodes in the inverse path look up the seenflag

value in their Seen Table when they obtain the *RREP* message. A seenflag fixed to FLASE this points out that it is the maiden *RREP* message on the way to the source node on the reverse link. Therefore, the intermediary nodes retune the value of seenflag after transmitting the *RREP* in the direction of the source. The node modestly rubbishes the *RREP* message on the foundation of seenflag value once the intermediary node obtains a *RREP* dispatch for the same *RREQ* message it got previously. The intermediate nodule, owing to this, can only participate on any one route from amongst the current numerous routes [40].

Type	R	A	Reserved	Prefix Size	Hop Count
Destination IP					
Destination Sequence Number					
Source IP Address					
Source Sequence Number					
Broadcasting ID					

Figure 3.6: AOMDV RREP structure

3.3.2.2 Data Packet Transmission and Route Selection Process

When there is no path offered in the routing table for the foundation node to direct its data packs to, the route discovery procedure is instigated by the node. Immediately after getting the first route for destination node recognised as primary route, the source node effects data communication. Deposits in the routing table as secondary routes are the rest node-disjoint routes that are learnt. A definite quantity of secondary routes in the table, every other route is not put in storage once the primary route is stored. Every other routes that are learnt afterwards of storing the primary path, if they possess lower hop count for destination as related to existing ones can substitute the present secondary paths. The route choosing function works in such a way that all the time it selects the primary route provided it is presented every time a route is needed for data conduction. The route selection function opt for the route with least hop count of the presented secondary routes in the event that the primary route is inactive [40].

3.3.2.3 Route Maintenance Process

When an active route is ruined during accomplishment of a data communication, the route maintenance method is summoned. Rubia Singla et al, in the event of route discontinuities, analyze and implement the performance of three route maintenance methods. In the first method transmission of data is sustained using the secondary routes when the primary route is ruined. The lifetime of each active secondary route is risen after a fixed amount of time to preserve the secondary routes in active mode while using the primary route. The source begins a new route finding process once all the secondary routes have also been broken. The routing expense resulting from discovering and preserving multiple routes can be minimized in this way [40].

CHAPTER 4

AD HOC ON-DEMAND MULTIPATH DISTANCE VECTOR WITH CONGESTION CONTROL AND AWARENESS (AOMDVCCA)

The preceding chapter rigorously discusses the AOMDV and the AODV routing protocols highlighting how they operate. Nevertheless, the protocols come with their shortcomings especially in high traffic situations. The Ad hoc On Demand Multipath Routing Protocol with Congestion Control and Awareness is a novel approach that aims to implement a newly designed active queue management (AQM) technique. Controlled Delay (CoDel) monitors the buffers to predict and prevent congestion. We present the general model of the proposed system and lay down the algorithms that will set the basis for development.

4.1 Shortcomings of AOMDV

4.1.1 Congestion and Contention

In capacity, the MANET comprises of numerous nodes. The nodes of extraordinary performance are simpler to be incorporated as route members since the route discovery process chooses the route has with the smallest amount of delay as the main route. For instance, in Figure 4.1, we undertake that as compared to others in the network the node G possesses high capacity. Since, accordingly *RREQ* (B) passes the route comprising the node G, before the path of the node G is designated as an affiliate of the primary route amid S and D, *RREQ* (B) arrives to the node D at first.

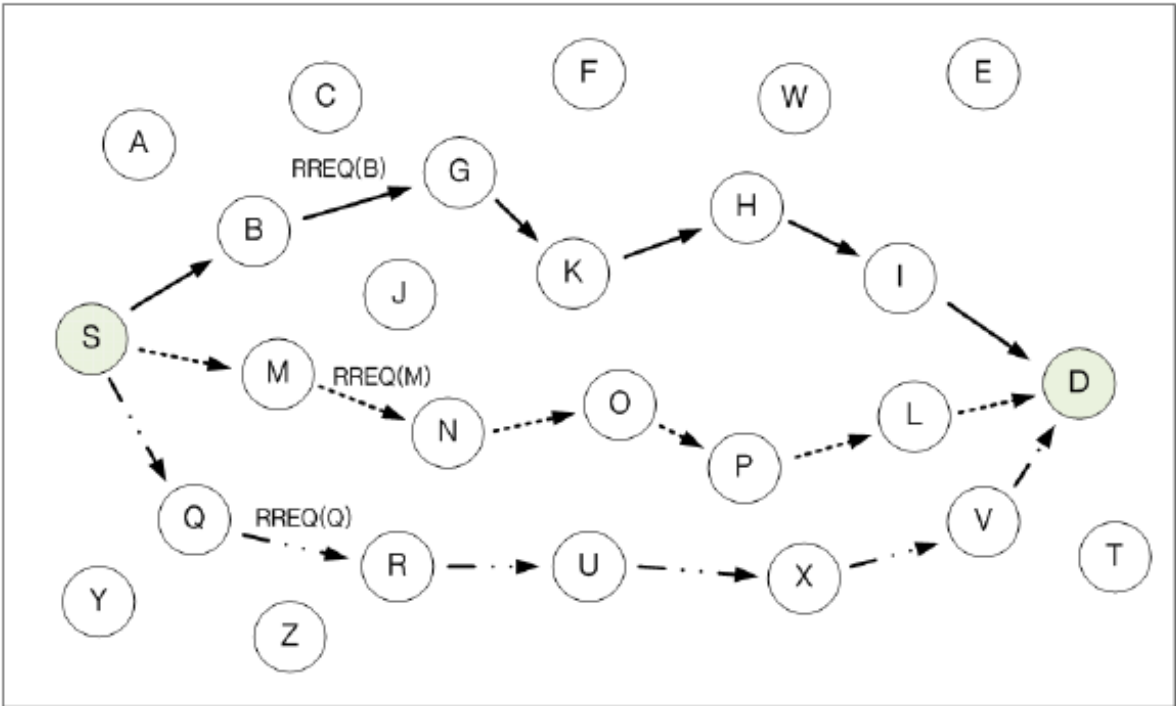


Figure 4.1: Route Discovery Procedure in AOMDV.

As the number of primary routes that contains the node G rises, however, the node G can jam of the paths because movement loads are concentrated on it. The situation that the routes (C, U), (S, D) and (F, Z) are interconnected at the node G is shown Figure 4.2.

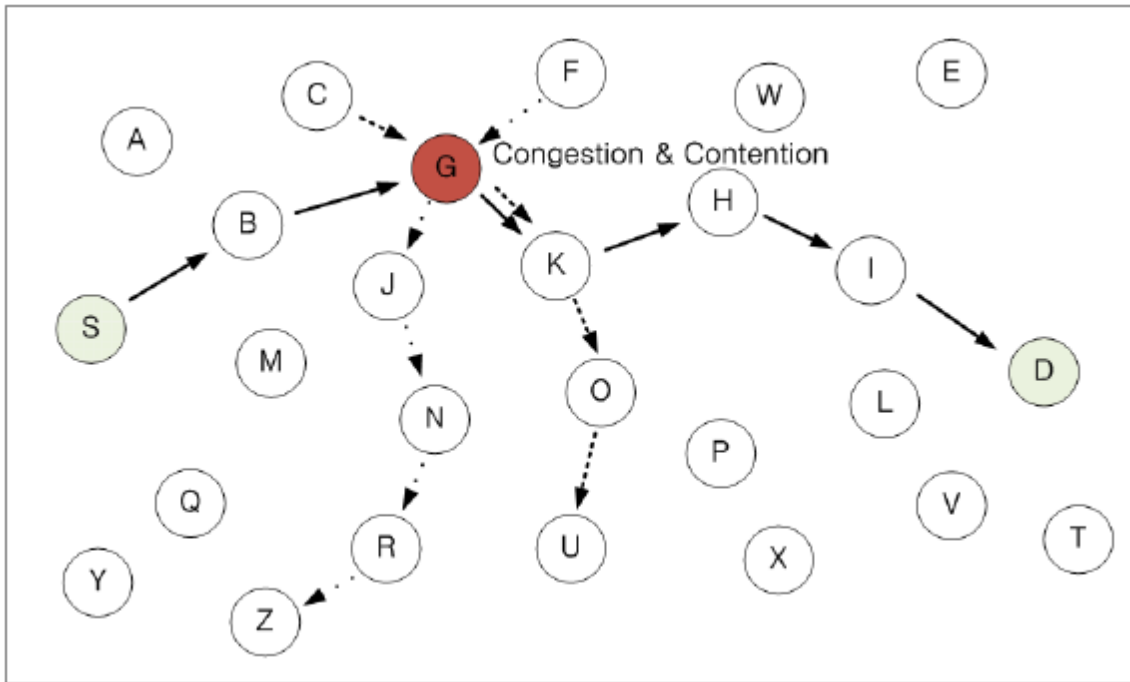


Figure 4.2: Intersected Routes leading to Congestion.

Owing to the features of wireless communications, the more extreme contention is triggered as further lively nodes are within the communication scope. This damages the operation of the bottleneck node.

4.1.2 Limitation of Static Route Switching

In terms of reply time and bandwidth, numerous paths have several different performances. The rest are utilised as alternate routes while the finest of them is nominated as the main route. When the principal route is destroyed in AOMDV, in order to avert an additional route discovery procedure the source node picks one of the substitute paths.

It presents the following glitches nonetheless. First, it cannot acclimatize to the vigorous change of the MANET as only in the event of a route error, is when the route switching in AOMDV happens. Routes that possess improved performance than the principal route can be obtained at any time since the system state of the MANET alters regularly. The static route exchanging still fails to obtain the benefit of the change. The performance of the alternate path cannot be assured since the route switching is accomplished minus information on up-to-date status of that alternative. Second, the alternative routes cannot schematically be placed in order of importance. Since a field in the routing table appropriate for managing information on the routes is not present in AOMDV, there is a lack evaluation of performance when the choice of the alternative routes is performed.

4.2 Active Queue Management

AQM denotes to algorithms and methods to regulate the amount of data kept in network node buffers. The elementary standard of AQM is that of pre-emptively dropping packets on condition of some metric of confined congestion or queue exploitation. They can also serve as the basis of unambiguous congestion notification, by for example, marking packets rather than dropping them.

Why are AQM systems critical? Buffers cannot be effective for their proposed purpose of controlling bursts of packets if they fill up, and TCP cannot operate appropriately in the case of congestion except it is gestured to slow down by packet drop or Explicit Congestion Notification (ECN) in a well-timed manner. TCP's reaction to sharing a link is quadratic: 10 times the delay means it will react 100 times more slowly to contending traffic. Some researchers point towards local active queue management (AQM) systems (i.e., affecting the

scheduling and dropping of packets in the buffer contrarily from a traditional FIFO persuasion) as the decisive solution of dropping queuing delay.

Active queue management algorithms are also desirable to attack the latency concern. Buffers are needed to cope with traffic bursts, so congestion cannot be solved simply by plummeting buffer sizes to very small values. AQM's are needed wherever there is a chance of a queue developing. Since current arrayed algorithms are characteristically random early detection - RED (Floyd et al, 1993) alternatives, they need alteration.

CoDel (Controlling Queue Delay) by Kathie Nichols and Van Jacobson (Nichols et al, 2012) has some operation plusses over other AQMs and can resolve the above deliberated hitches of adaptive AQMs. It ensures that nearly all of its work is done at dequeue stage (when packets are transferred). CoDel does involve tallying a timestamp to each separate packet as it is received, but even if this can't be done by the network hardware, timing information is openly accessible from a CPU register in modern CPUs.

4.3 Ad Hoc On-Demand Multipath Distance Vector with Congestion Control and Awareness (AOMDVCCA)

4.3.1 System Model

N mobile resource constrained sensor nodes (SN) and mobile resource-constrained sinks is what makes up the WSN scenario. Normally, hundreds to thousands of cheap SNs are used in constructing WSNs. There are $n!$ likely vectors of size n which can describe the environment as reported by sensors for a WSN consisting of n SNs, at any moment. The proposed system is evidently elucidated by the working of a suggested load balancing scheme. The first sender establishes the path to receiver on the basis of typical AOMDV protocol. It implies that between sender and receiver, more than one path is formed at a given time the data is delivered through one of the shortest selected path. The working of the AOMDV protocol has been extensively discussed in section 3.3.

In this study, we suggest a congestion free network with minimal packet drop design for mobile ad-hoc networks. Three approaches are fundamentally used to abate the congestion. It entails:

- i. Multipath routing using AOMDV.
- ii. Varying queuing procedure.
- iii. Scrutiny of packet drop using static and varying queue (drop counts).

In our first part Ad-hoc On-demand Multipath Distance Vector Routing (AOMDV) is used as a multipath routing protocol. It computes multiple loop-free and disjoint paths as an addition to the AODV protocol. Along with the conforming hop counts, the routing records for each terminus contain a list of the next-hops. A similar sequence figure is applied for all subsequent hops. In order to possess way of a route, this is of paramount importance.

A node preserves the publicised hop count for all destinations, which is used for distributing path advertisements of the target and this is well distinguished as the maximum hop count for entire paths. Route announcements that are identical and acknowledged by a node express a substitute path to the intended destination. For a node, if it has a fewer hop count than the advertised hop count for that target, loop freedom is guaranteed by acceptable substitute routes to the sink. The advertised hop count consequently does not alter for the identical order value since the supreme hop count is used. The advertised hop count and the next-hop list are re-initialized when a route advertisement is acknowledged for a target with a superior sequence number. AOMDV, accordingly provides several substitute paths to forward the data packets.

In the second part, the protocol's queuing model is monitored by a controlled delay (CoDel) scheme. CoDel is a recent Active Queue Management (AQM) introduced in 2012 by Kathy Nichols and Van Jacobson. In this study it is implemented in order to ensure that a queue does not become bloated with traffic, thereby leading to congestion.

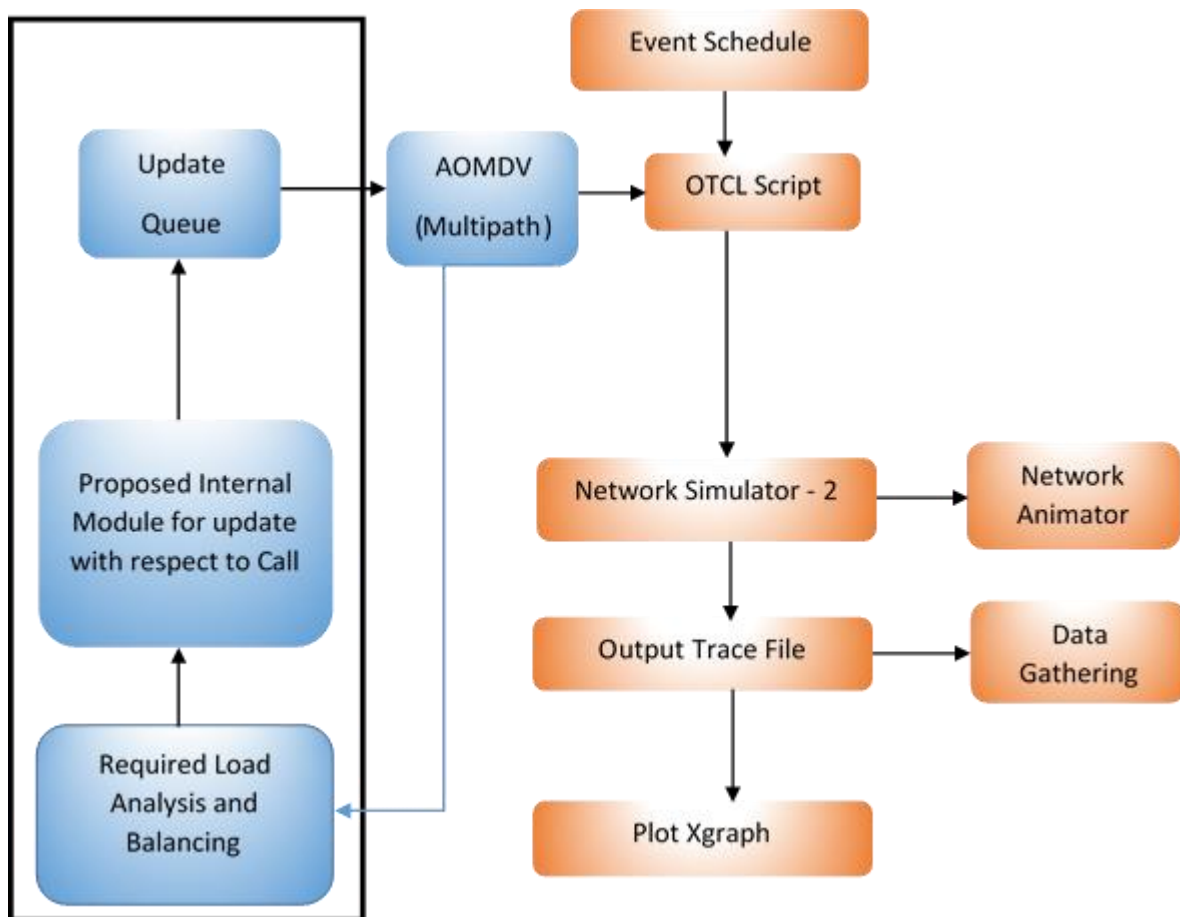


Figure 4.3: AOMDVCCA model

4.3.2 Proposed Congestion Control Scheme

This proposed scheme's main point is to observe the load in each node. Here the multipath routing means to execute routing procedures to select the multiple paths in between sender to destination and the number of nodes in them. CoDel is used to evaluate the buffer activity of each node. The rate controlling is required so that buffer capacity is not overly stretched if the sender and the receiver play no active role in controlling their data rates.

CoDel is intended to overwhelm congestion in network links by imposing restrictions on the delay network packets experience due to passing through the buffer managed by CoDel. CoDel intends at improving the overall performance of the RED algorithm by concentrating on some important delusions in the algorithm (as observed by Jacobson) and by being easier to manage (meanwhile, unlike RED, CoDel does not involve manual alignment) [47].

4.3.2.1 Queues in CoDel

Good queue: Well-defined as a queue that displays no congestion i.e. it rapidly drains as packets are conveyed. These queues are needed to smooth over the characteristic bursts of incoming packets at jams in the system. Such bottlenecks can be between a fast transmitting node and a slower processing receiving node.

Bad queue: Defined as a queue that is choked up at a similar or higher rate as packets are transmitted, so the queue will certainly not be emptied. In its steady state, the TCP protocol will discharge more packets as the reception of previous packets is admitted. At this point, a queue at the bottleneck between the sender and the receiver will become a rising or standing queue.

For it to be effective against congestion, a resolution in the form of an active queue management (AQM) algorithm that must be able to foresee or identify an occurrence of congestion in a queue and respond with arraying operational countermeasures [48].

4.3.2.2 CoDel: Working

CoDel works by totting a timestamp to every packet as it is received and enqueued. Once the packet touches the head of the queue, the time consumed in the queue is deliberated; it is an artless calculation of a solitary value, with no locking necessary, thus it will be quick. If the time used up by a packet inside the queue is greater than a definite threshold, the algorithm arrays a timer to drop a packet at dequeue. This dropping is only performed when the queuing delay within a time frame is beyond the threshold charge.

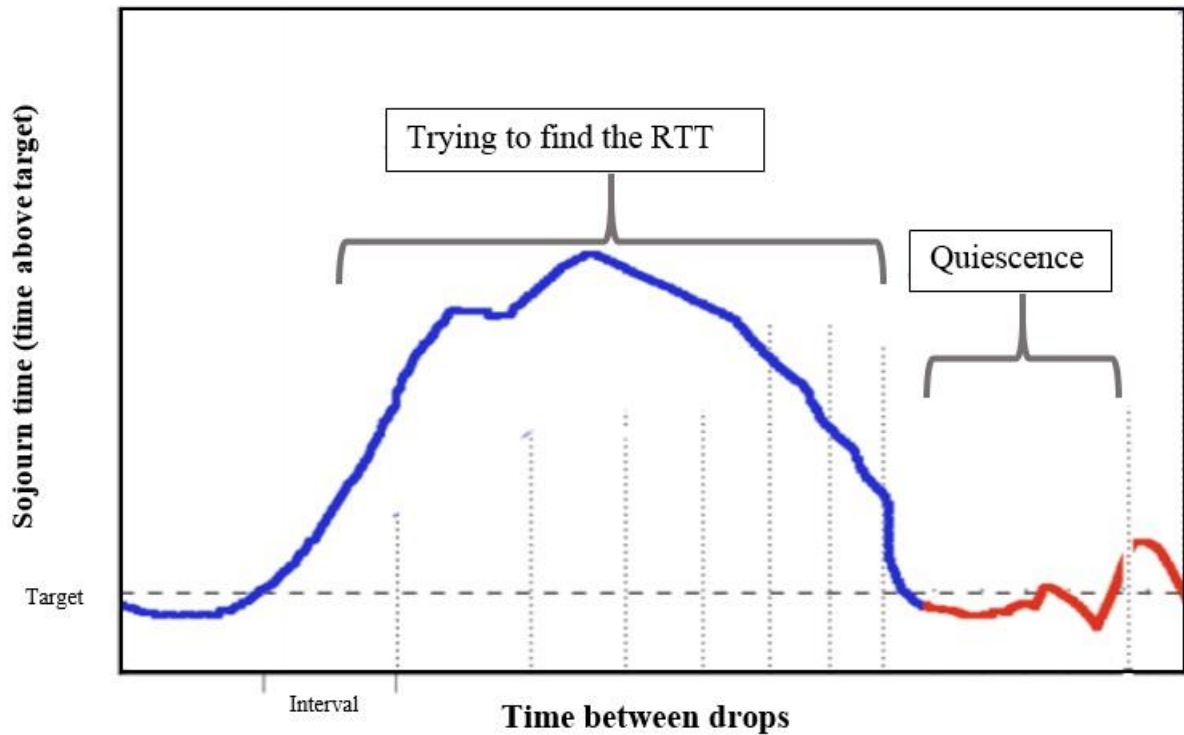


Figure 4.4: CoDel drop scheduler behavior.

From the sketch:

- CoDel assumes that a permanent queue of target is suitable (as affirmed by the local minimum) and that it is undesirable to drop when there are less than an MTU worth of bytes in the buffer.
- To guarantee that the minimum value is not stale, it has to have been experienced in the latest interval (sliding window).
- Once the sojourn time has surpassed target for no less than an interval, a packet is plunged and a control law used for setting the next drop time.
- This subsequent drop time is dwindled in inverse proportion of the square root of the total drops subsequently after the dropping state was arrived (to get an undeviating change in throughput).
- After the sojourn time veers lower than target, the controller stops dropping.
- Prevented from returning to dropping state hastily after escaping it and the dropping state is recommenced at a latest control level if one is present.

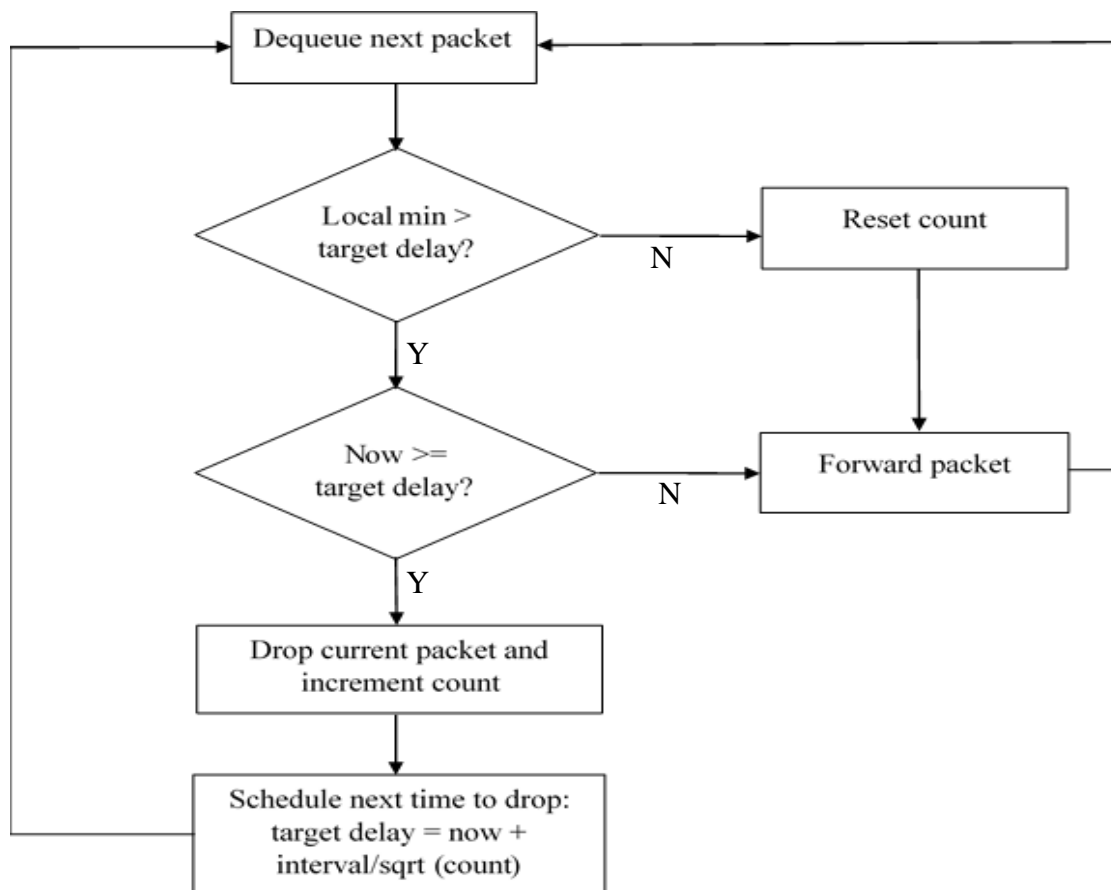


Figure 4.5: Simplified CoDel algorithm flowchart

CoDel relies on the packet sojourn time i.e. the actual queue delay taken by a packet as a metric to foresee congestion in the network. If the packet sojourn time is beyond the target value for an indicated interval of time, CoDel starts proactively dropping/marking the packets to control the queue length. Nevertheless, CoDel circumvents the underutilization of outgoing link by not dropping/marking the packets beforehand in a scenario when current queue size is beneath one MTU.

Below is the algorithm for CoDel AQM.

For every packet arrival:

if current_size_of_queue < queue_limit **then**

Add a timestamp to the packet header

Enqueue the packet

else

Drop the packet

end

For every packet departure:

Calculate packet sojourn time as;

`pkt_sojourn_time = dequeue_time - enqueue_time`

If CoDel is inside the dropping state

if dropping_state = 1 **then**

if sojourn_time < target

or current_size_of_queue < 1MTU **then**

 Forward the packet

 Come out of dropping state:

 dropping_state = 0

else

while dequeue_time \geq next_drop_time **do**

 Drop the packet

 Increment the total no of packet dropped since
 entered into dropping_state

 count = count + 1

 Update the next_drop_time as:

 next_drop_time = next_drop_time
 + interval / $\sqrt{\text{count}}$

end

end

else

If CoDel is outside dropping state and sojourn_time of
packet is more than target:

```

if dropping_state = 0 and sojourn_time > target
then

    Drop the packet

    Enter dropping state:

    dropping_state = 1

else

    Forward the packet

end

end

```

The Algorithm above pronounces the operation of CoDel algorithm. When a packet reaches a node, it is modestly enqueued if there is any existing room. If the queue limit is already touched, the received packet is discarded. While enqueueing the incoming packet, a timestamp is also appended to the packet header, which turn out to be the enqueue time. When the packet touches the finish of the queue, it is dequeued and packet sojourn time is computed by deducting the enqueue time from dequeue time.

There are principally two conditions in CoDel viz. dropping state and non-dropping state. If the CoDel is presently in non-dropping state and packet sojourn time rests beyond the target for a definite interval of time, CoDel goes into the dropping state. Once after moving into dropping state, packets are dropped/marked till the packet sojourn time lethargies beneath the target value. When sojourn time falls below the target value CoDel transforms out to non-dropping state. Contrasting RED and its other alternatives which drops the packet at enqueue time, CoDel discards the packet at dequeue time.

CoDel utilises two constraints viz. *target* and *interval*. *target* (constant 5ms) is the lowest tolerable queue delay, while *interval* (constant between 10ms to 1 sec but 100 ms in our case) is the worst case RTT through the bottleneck [49]. After dropping a packet the next drop time is calculated as:

$$\text{next_drop_time} = \text{next_drop_time} + \text{interval} / \sqrt{\text{count}}$$

where, *count* is the total number of packets dropped since we entered into dropping state.

CoDel has shown significant improvements as compared to RED and ARED on wired networks for various scenarios of variable bottleneck bandwidth, variable bottleneck RTT and variable FTP flows [50].

4.3.2.3 AOMDVCCA Algorithm (Proposed Protocol)

The proposed protocol will merge the multipath routing techniques of AOMDV and the active queue management approach of CoDel. CoDel monitors every buffer at each node to take preventive steps in order to avoid congestion (bloating). This is the awareness part. On the other hand, the AQM will manage to recover from a scenario of congestion and transport the packets normally.

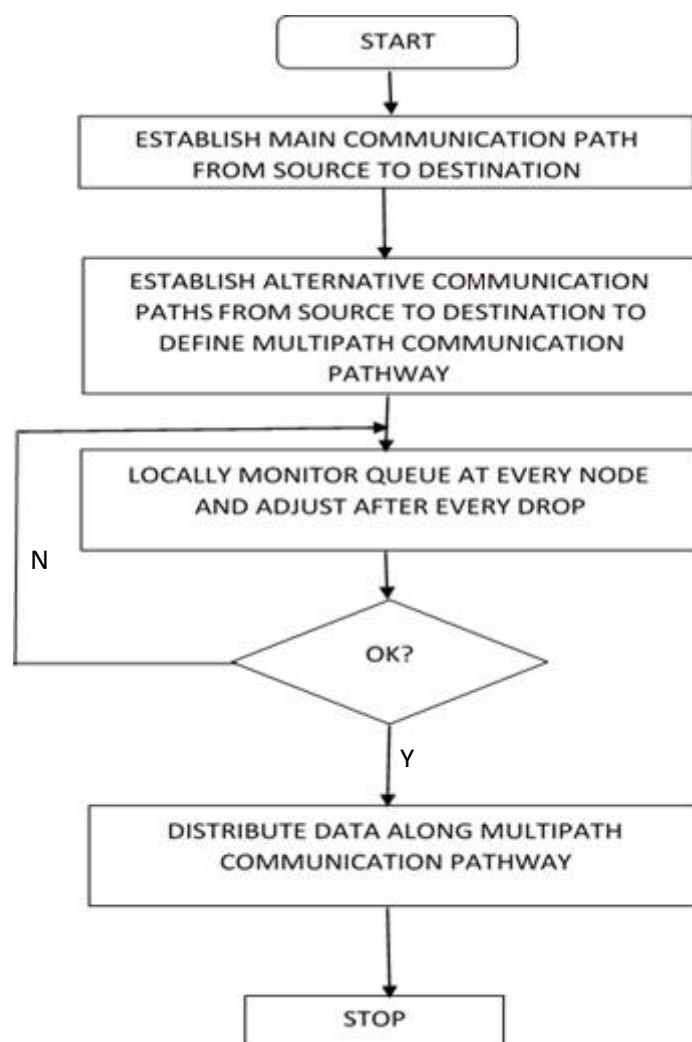


Figure 4.6: AOMDVCCA flow chart.

The flexible queue length scheme has elevated the storing and forwarding capability or processing quickness of nodes in the system. The recipe of these approaches advances the AOMDV routing performance and appropriate load balancing in network.

```

(1) Set M = Mobile Node's
(2) Set S = sender
(3) R = Receiver Node
(4) Routing Protocol = AOMDV
(5) Broadcast Route Request (RREQ)
(6) Set TARGET = 5 //time to live 5 ms
(7) QUEUE_LIMIT // the maximum buffer size
(8) PKT_SOJOURN_TIME // time taken by the packet in the buffer
(9) DROPPING_STATE = 0 // true of false
{
if (Route in between S to R found)
{
    Check number of route;

    if (route ==> 1) //means alternative route exist in network
    {
        if (next-hop != R && Route Disjoint)
        {
            Continue tile all radio range nodes route packet receives
        }
    }

    else
    {
        Receiver found;

        Data_send ();
    }
}

```

```

    }
else
{
    Route error (RERR)
}

Data send (S, R, data) // sending case drop minimization
{
    Sender send's data through computed path;
    Check (Q size of Intermediate (I) node's)
    //dynamic variation Q scheme at intermediate node
    For every packet arrival:
if current_size_of_queue < QUEUE_LIMIT then
    Add a timestamp to the packet header
    Enqueue the packet
else
    Drop the packet
end

    For every packet departure:
    Calculate packet sojourn time as:
    PKT_SOJOURN_TIME = dequeue_time - enqueue_time
    If CoDel is inside the dropping state
if DROPPING_STATE = 1 then
        if sojourn_time < TARGET
        or current_size_of_queue < 1MTU then
            Forward the packet

```

Come out of dropping state:

```
DROPPING_STATE = 0
```

else

```
while dequeue_time ≥ next_drop_time do
```

```
    Drop the packet
```

```
    Increment the total no of packet dropped since  
    entered into dropping_state
```

```
    count = count + 1
```

```
    Update the next_drop_time as:
```

```
    next_drop_time = next_drop_time  
                    + interval / √count
```

```
end
```

```
end
```

else

If CoDel is outside dropping state and sojourn_time of packet is more than target

```
if DROPPING_STATE = 0 and PKT_SOJOURN_TIME > TARGET
```

```
then
```

```
    Drop the packet
```

```
    Enter dropping state:
```

```
    DROPPING_STATE = 1
```

```
else
```

```
    Forward the packet
```

```
end
```

```
end
```

}

}

This is therefore the full algorithm showing the new protocol designed in our study – Ad Hoc On Demand Multipath Distance Vector with Congestion Control and Awareness. The CoDel AQM is augmented as the appropriator in the aggregation of packets to be sent at every intermediary node. The main objective of this algorithm is to increase network throughput by forecasting bottleneck links and managing them to avoid high packet drops from congestion.

4.3.2.4 Properties

The CoDel procedure for proficient communication in the ad hoc multipath distance vector protocols addresses a solution to our problem whose features can be discussed as:

- It is parameterless—it has no knobs for users, implementers or operators to regulate.
- It acts on good queue and bad queue in a different way—i.e., it preserves the delays small while permitting bursts of traffic.
- It manages delay, even though unmoved by link rates, traffic loads and round-trip delays.
- It acclimatizes to enthusiastically fluctuating link rates with no undesirable bearing on utilization.

CHAPTER 5

SIMULATION RESULTS AND ANALYSIS

Performance of AOMDVCCA is tested on a Linux operating system running NS-2. The Tool Command Language (TCL), C++ and C are the prominent development software used to implement this proposed protocol. Output files from NS-2 such as the trace file that indicate network activity provide the data for analysis that can be presented using Xgraph. Common metrics such as throughput, end-to-end delay and packet delivery ratio are some of the variables used to test the performance of AOMDVCCA against AODV and then against AOMDV. The tests were performed by either varying the number of nodes in the network or the speed of mobile nodes. All protocols are exposed to the same constraints in order to provide the study with conclusive results of performance.

5.1 Simulation Setup

5.1.1 Ubuntu 14

Ubuntu 14.04.1 is an open operating system centred on Linux. Red Hat being designed by the open source society and the Red Hat engineers promote the expansion of Ubuntu. Certain main features of Ubuntu are widespread performance advances, backing a fresh Graphical User Interface (GUI) virtualization manager and Intel-based Macs.

5.1.2 The Network Simulator (NS2)

Simulation can be well-defined as “Reproducing or approximating how proceedings might occur in a real scenario”. It can include composite mathematical modelling, role playing deprived of the aid of technology, or mixtures. The worth lies in the stepping of you under genuine conditions that alternate as a result of conduct of others involved, so you cannot expect the order of proceedings or the final result.

5.1.2.1 NS2 Overview

NS [51] is an event driven network simulator designed at University of California at Berkeley, USA, as a REAL system simulator projects in 1989 and was designed with support of numerous organizations. Currently, it is a VINT scheme sustained by DARPA. NS is not a complete tool that can accomplish all types of network model. It is still a continuing work of research and

development. The operators are accountable in verifying that their network model simulation does not comprise of any errors and the public should share their findings with everyone. There is a guidebook called NS manual for user direction.

5.1.2.2 The Network Animation (NAM)

The network animator was inaugurated in 1990 as a modest tool for mimicking packet trace data. This trace data is characteristically a derivative output from a network simulator such as NS or from actual network dimensions, e.g., using *tcpdump*. Steven McCanne inscribed the first version as an associate of the Lawrence Berkeley National Laboratory Network Research Group, and has sporadically upgraded the design, as he's desired it in his investigation. Marylou Orayani enhanced it more and used it for her Master's thesis in 1995 and 1996. The NAM design effort was a continuing partnership with the VINT scheme. Presently, it is being technologically advanced at ISI by the Conser and SAMAN projects.

5.1.2.3 The Trace File

The trace file is an ASCII programme file and the trace is prearranged in 12 fields as depicted in figure below.

1	2	3	4	5	6	7	8	9	10	11	12
<i>Event</i>	<i>Time</i>	<i>From node</i>	<i>To node</i>	<i>Pkt Type</i>	<i>Pkt Size</i>	<i>Flags</i>	<i>Fid</i>	<i>Src addr</i>	<i>Dst addr</i>	<i>Seq num</i>	<i>Pkt id</i>

Figure 5.1: Fields of a trace file.

The leading field is the event type and specified by one of four existing signs *r*, *+*, *-* and *d* which match respectively to received, enqueued, dequeued and dropped. The next field is showing the time which the event happens. The third and fourth are the input and output node of the connection at which point the events happen. The fifth signifies the packet type for instance transmission control protocol (*tcp*) or continuous bit rate (*cbr*). The sixth is the magnitude of the packet and the subsequent field is some sort of flags. The eighth place is the flow identity of *IPv6* that can insist on stream color of the NAM presentation and can be used for additional analysis tenacities. The ninth and tenth sections are the source and terminus address in the

arrangement of “*node.port*”. The eleventh is the network layer protocol’s packet order value. NS possesses, for investigation purposes, track of UDP packet sequence number. The final part is the unique identity of the packet. Outcomes of simulation are deposited into trace file (**.tr*). X-Graph was used in the analysis of the trace file.

5.1.2.4 The X-Graph

The X-graph platform plots a graph on an X display supplied data read from either one of data files or from standard input if no files are indicated. It can exhibit up to sixty four autonomous data sets using dissimilar colours and/or line flairs for every set. It marks the graph with a title, axis markers, grid outlines or tick symbols, grid tags, and a legend. There are choices to control the look of most constituents of the graph.

A data set comprises of an ordered list of points of the custom “directive X - Y”. For directive “draw”, a streak will be drawn sandwiched between the preceding point and the present point. Stipulating a “move” directive commands X-graph not to mark a line between the points. “Draw” is the default instruction. The name of a records set can be identified by encompassing the name in twin quotes.

The platform used to stipulate the size and location of this window hinge on the window manager presently in use. After the window has been unlocked, all of the data sets will be shown graphically with a legend in the higher right angle of the display.

X-graph furthermore offers three regulator buttons in the top left corner of each window: Hardcopy, Close and About. X-graph admits a huge number of choices most of which can be itemised either on the command line, in the operator’s *.Xdefaults* or *.Xresources* file, or in the same data files.

5.2 Performance Evaluation - AOMDVCCA

The proposed scheme has been implemented in NS2. The simulation environment consists of different number of nodes in a rectangular region of varying size.

Routing Protocol	AODV
Algorithms	CoDel
Propagation	TwoRayGround
Radio range of a node	150 m
Channel capacity	2 Mb/sec to 11 Mb/sec
Medium Access Control (MAC) protocol	IEEE802.11 Distributed Coordination Function (DCF)
Traffic pattern	100 CBR/FTP (we assume bidirectional communication here)
Data packet size	512 bytes
Rate of data	1 packet/sec
Number of nodes	10 to 200
Average speed	5m/s, 8m/s, 10m/s
Simulation time	100 seconds
Dimension of simulated area	800×800
Transport Layer Protocol	TCP, UDP
Maximum Speed (m/s)	10
Value of <i>interval</i>	100ms (constant)
Value of <i>target delay</i>	5ms (constant)

Table 5.1: The Simulation Parameters

The following metrics are used to analyze the performance of the proposed scheme.

A. Network Throughput

Throughput is the average data rate of a source sending packets and received by the receiver. This metric represents the total number of bits forwarded to higher layers per second. It is measured in *bps*. It can also be defined as the total amount of data a receiver actually receives from sender divided by the time taken by the receiver to obtain the last packet.

B. Average end-to-end delay of Data Packets

This is the average delay between the sending of the data packet by the constant bit rate source and its receipt at the corresponding constant bit rate receiver.

C. Packet Delivery Ratio

Packet delivery ratio is the ratio of number of packets received at the destination nodes to the number of packets sent from the source nodes. When packet delivery ratio is high, the performance is better (Gupta et al, 2010)

D. Routing overhead

This variable defines how many routing packets for route discovery and route maintenance must be sent so as to disseminate the data packets.

5.2.1 Performance Evaluation of AOMDVCCA vs AODV

Multipath routing techniques employed in the new protocol designed in this study are also a way to manage the problem of congestion. Tests were carried out using the metrics such as network throughput, packet delivery ratio and average end-to-end delay. The number of nodes chosen for these tests was 50.

5.2.1.1 Throughput, End-to-End Delay and Packet Delivery Ratio Variation

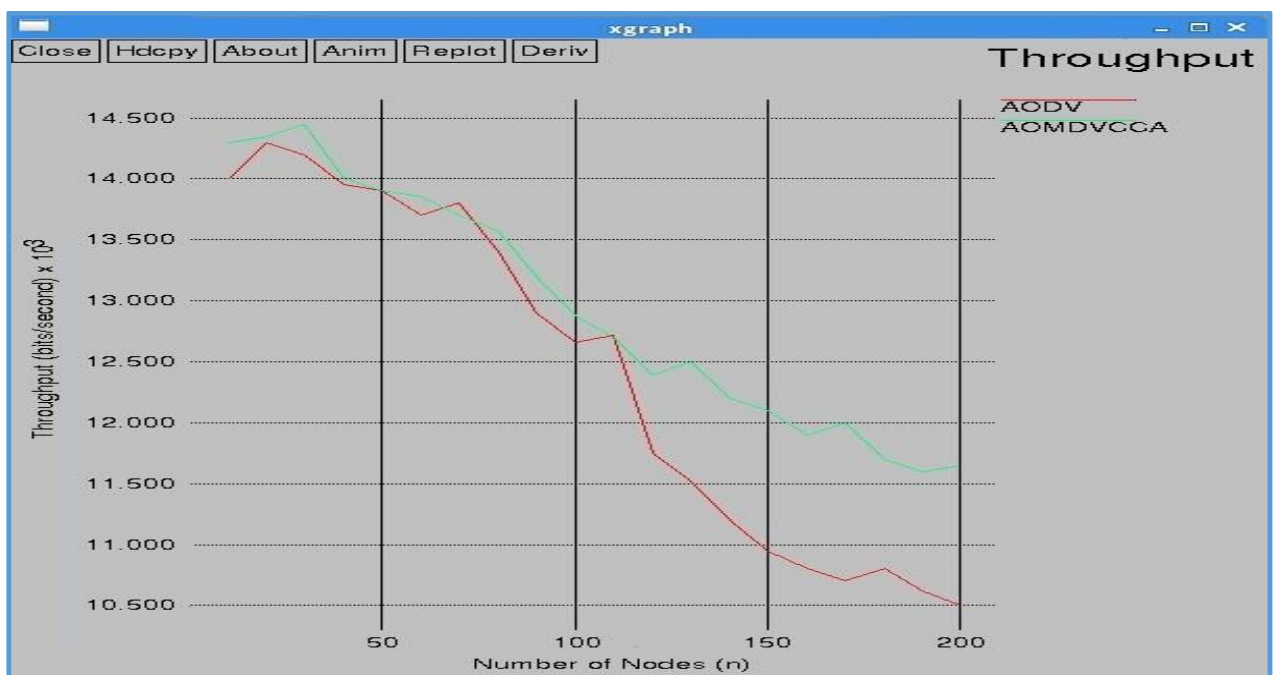


Figure 5.2: Throughput

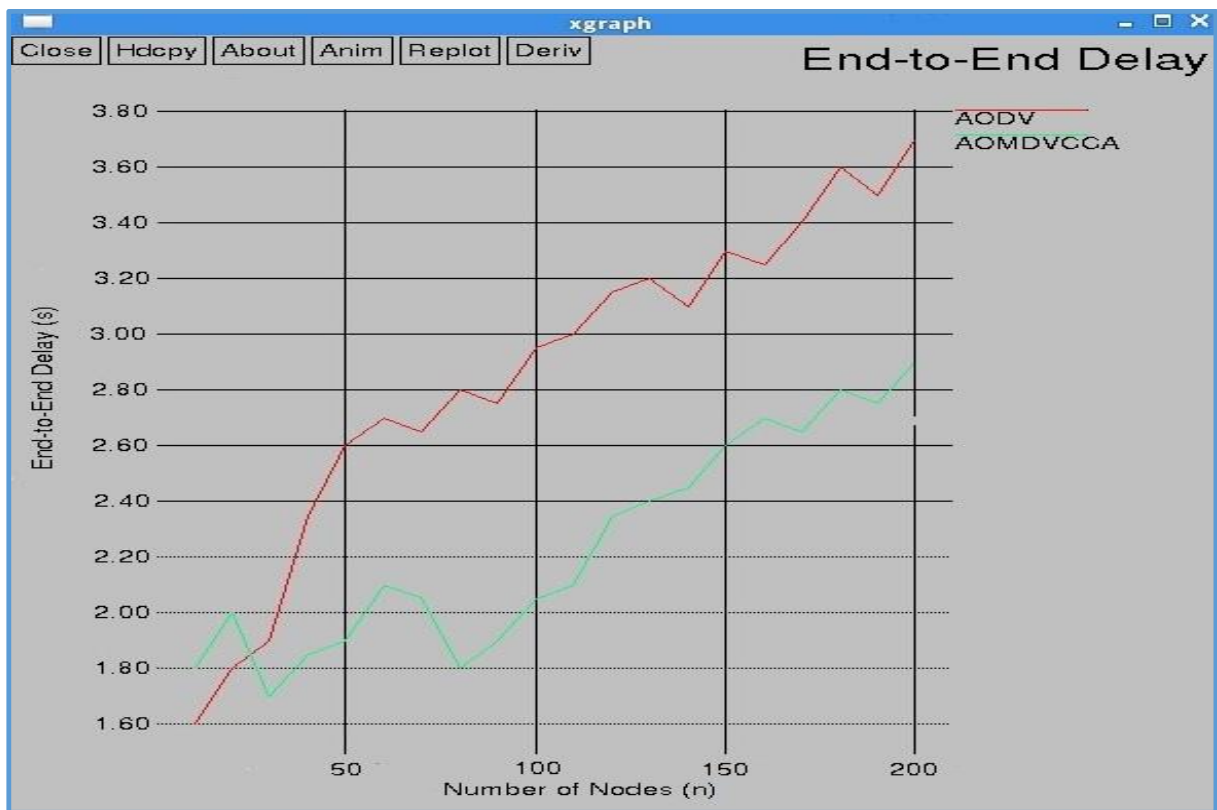


Figure 5.3: End-to-End Delay

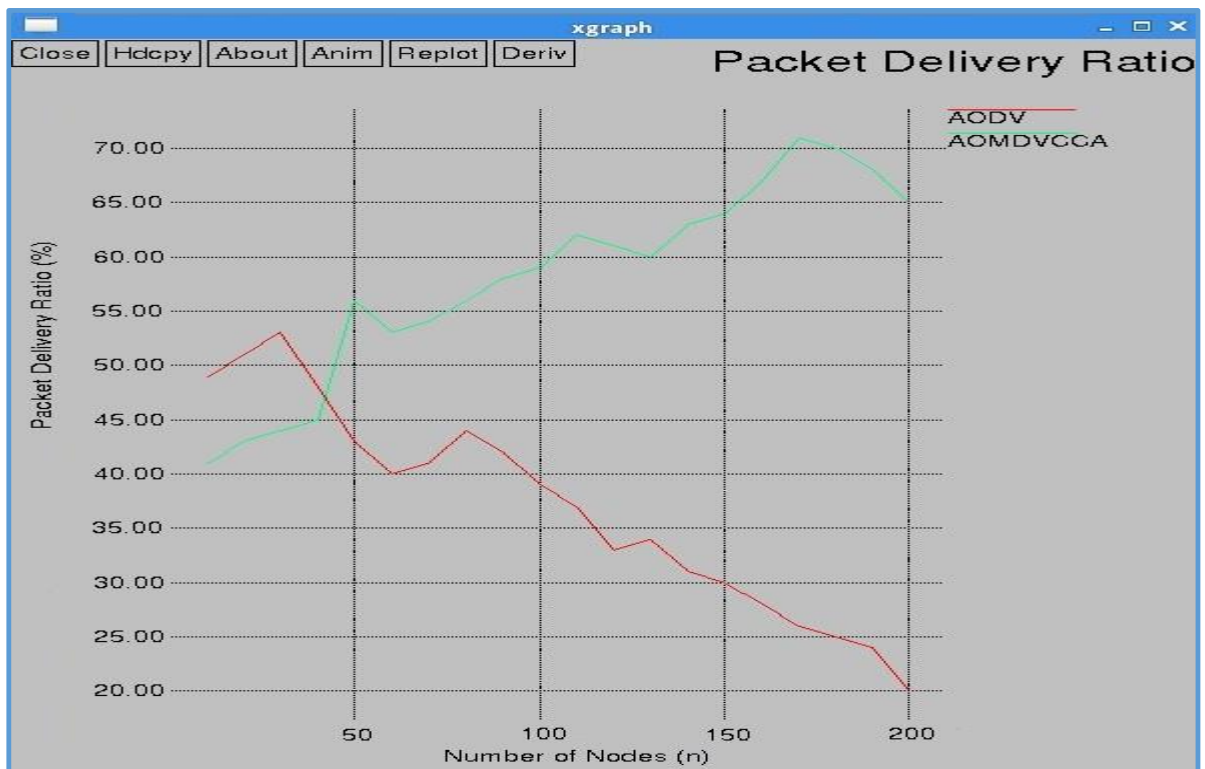


Figure 5.4: Packet Delivery Ratio

Figure.5.2, 5.3 and 5.4 show the number of nodes in the network variation with throughput, packet delivery ratio and end-to-end delay with respectively. Throughput can be seen to have improved and a reduced delay in the proposed scheme from increasing network load. The number of false route breaks increases due to congestion created by more number of active sessions at higher loads.

5.2.1.2 Packet Loss

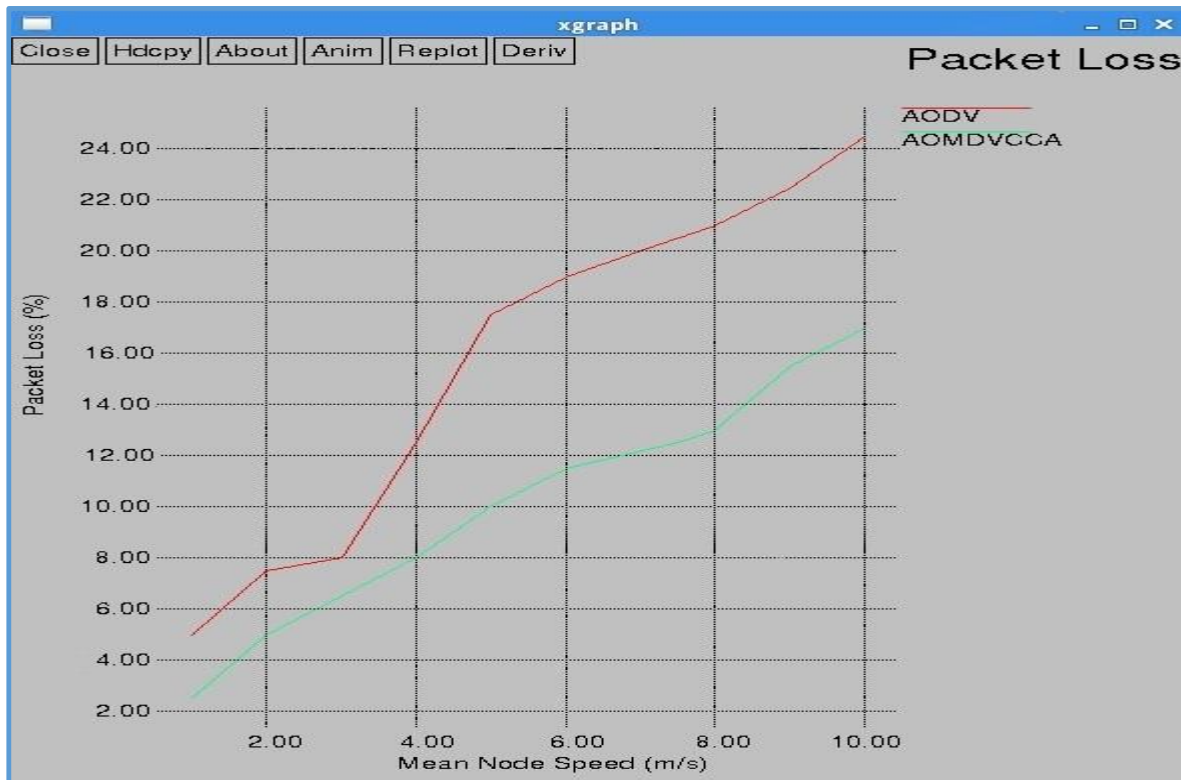


Figure 5.5(a): Packet loss

The packet loss performance of AOMDVCCA and AODV is compared in Figure 5.5(a). Take an observation that mobility is the main cause of packet losses here. When it does not have a route to forward the packet, an intermediate node drops it. When the buffer overflows or when it fails to get a route after several futile route discovery attempts, the source will also drop packets. Re-establishments of new routes will also take time in AODV since more routes start to break at a higher rate as the speed of nodes in the system continues to rise. Consequently, more and more packets drop as compared to AOMDVCCA

5.2.1.3 Average end-to-end delay and route discovery latency at varying node speeds

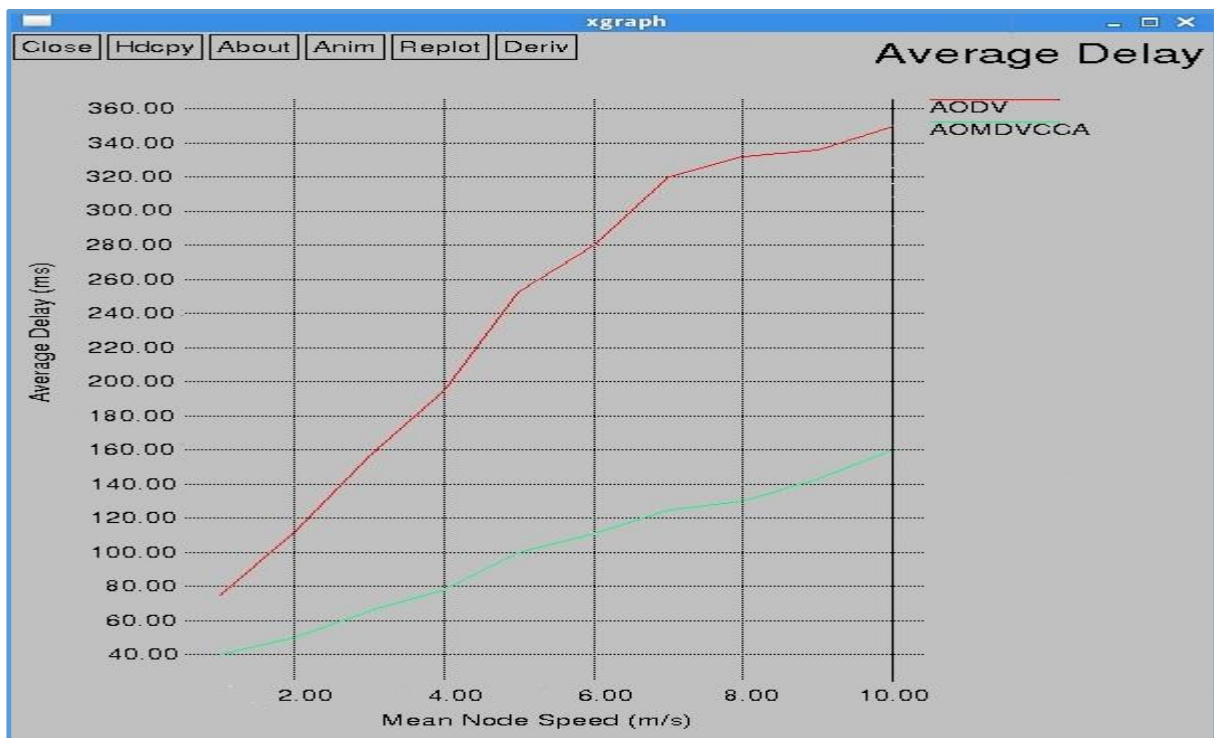


Figure 5.5(b): End-to-End Delay

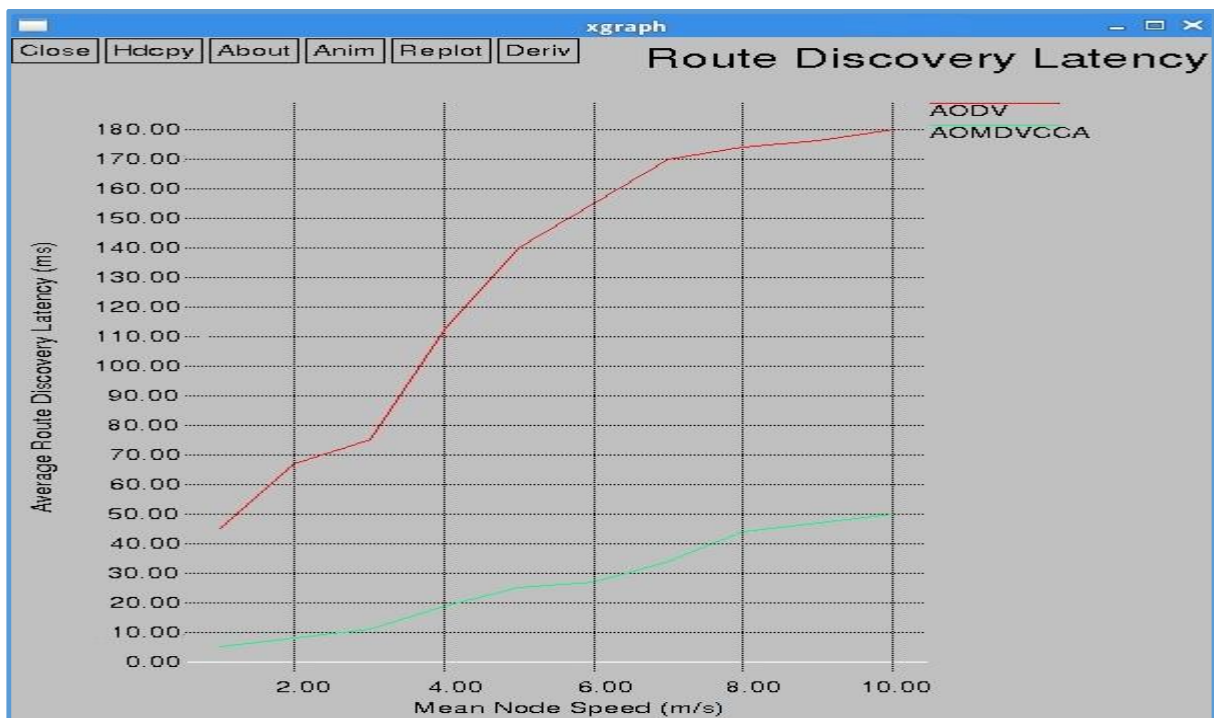


Figure 5.5(c): Route Discovery Latency

Respectively figures 5.5(b) and (c) show the average end-to-end delay and route discovery latency for both protocols. The amount of time a data packet spends in the buffer at the source

is the average route discovery latency represents, averaged over all packets that experience a non-zero latency at the source while a route is being found. Both these, as expected, because of the increase in the number of route failures and consequent route discovery operations, increase with mean node speed. AOMDVCCA significantly improves the delay almost always by more than a factor of two. Improvements in route discovery latency are even more remarkable. A distinguishable factor to note is that the route discovery latency accounts for about half of the overall delay in AODV. However, it is but only about one fourth of the delay in AOMDVCCA. This is due to the fact that as packets sent over failed links are salvaged from network interface buffer and then are re-routed in AODV the 'in-network' delay of packets is higher.

5.2.2 Performance Evaluation of AOMDVCCA vs AOMDV

Under this unit, the results are compared in case of normal AOMDV routing and in case of proposed AOMDVCCA routing. The proposed queue based scheme is showing the better performance and improves the routing capability of AOMDV protocol.

5.2.2.1 Packet Delivery Ratio Analysis

The percentage of packets are effectively acknowledged at destination are determined by packet delivery ratio. This graph signifies the performance of standard AOMDV routing and proposed link expiration time increment with AOMDVCCA which employs an active queue length method. The normal AOMDV definitely provides the superior results as compared to unipath routing and capable of handling heavy load through its multipath technique. In this graph the performance of AOMDV protocol and AOMDVCCA are almost equal at simulation time about 0 to 10 seconds but subsequently the PDF in case of proposed scheme is about 97 % up to end of simulation but then the standard AOMDV only provides the 83 % PDF at the completion of simulation. At time about 40 to 60 seconds the PDF is around 72% and after that recovers and touches 83%. The reason of low PDR values is to change next alternative path if the likelihood of congestion occurs in prevailing path.

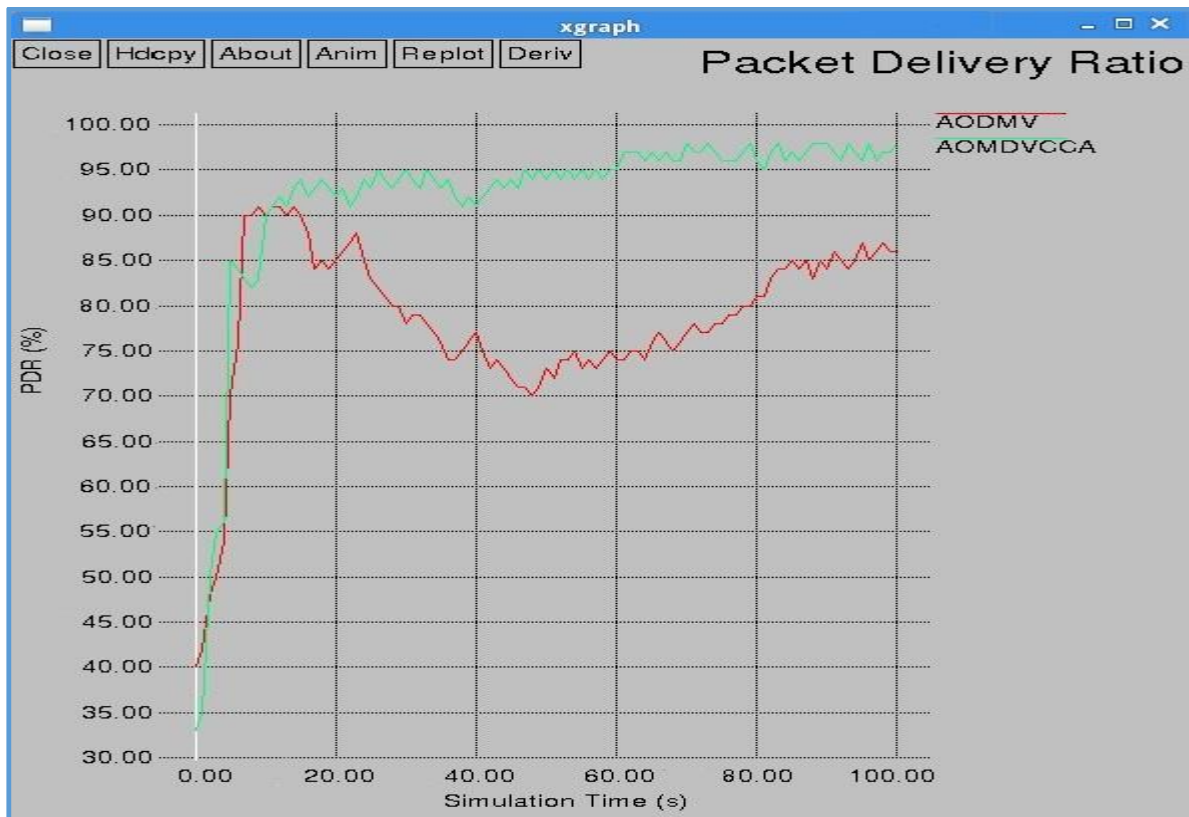


Figure 5.6: PDR Analysis

5.2.2.2 Throughput Analysis

The quantity of packets received at terminus in per unit of time are measured by the throughput parameter. The throughput is greater in network, if the transmission and receiving are running persistently deprived of any interference such as congestion. This chart characterises the throughput performance in case of standard AOMDV routing and the proposed AOMDVCCA routing. In case of suggested scheme the throughput is around 900 packets / seconds but in case of standard AOMDV routing the throughput is merely around 600 packets/ seconds. The performance of standard AOMDV is enhanced on the basis of active dynamic queuing. The proposed system also eradicates the likelihood of congestion and not only offers the alternative but also improves the routing process.



Figure 5.7: Throughput analysis

5.2.2.3 UDP Received Analysis

User Datagram Protocol (UDP) is a Transport layer protocol that operates on connection-less arrangement. Figure 5.9 indicates the UDP packets received investigation of normal AOMDV protocol and proposed AOMDVCCA routing. At this point, in case of proposed technique about 2340 packets are delivered in the network whereas in case of normal AOMDV routing, about 1690 packets are delivered in the network. This performance demonstrates the improvement on storing capability of nodes thereby improving the packets receiving.

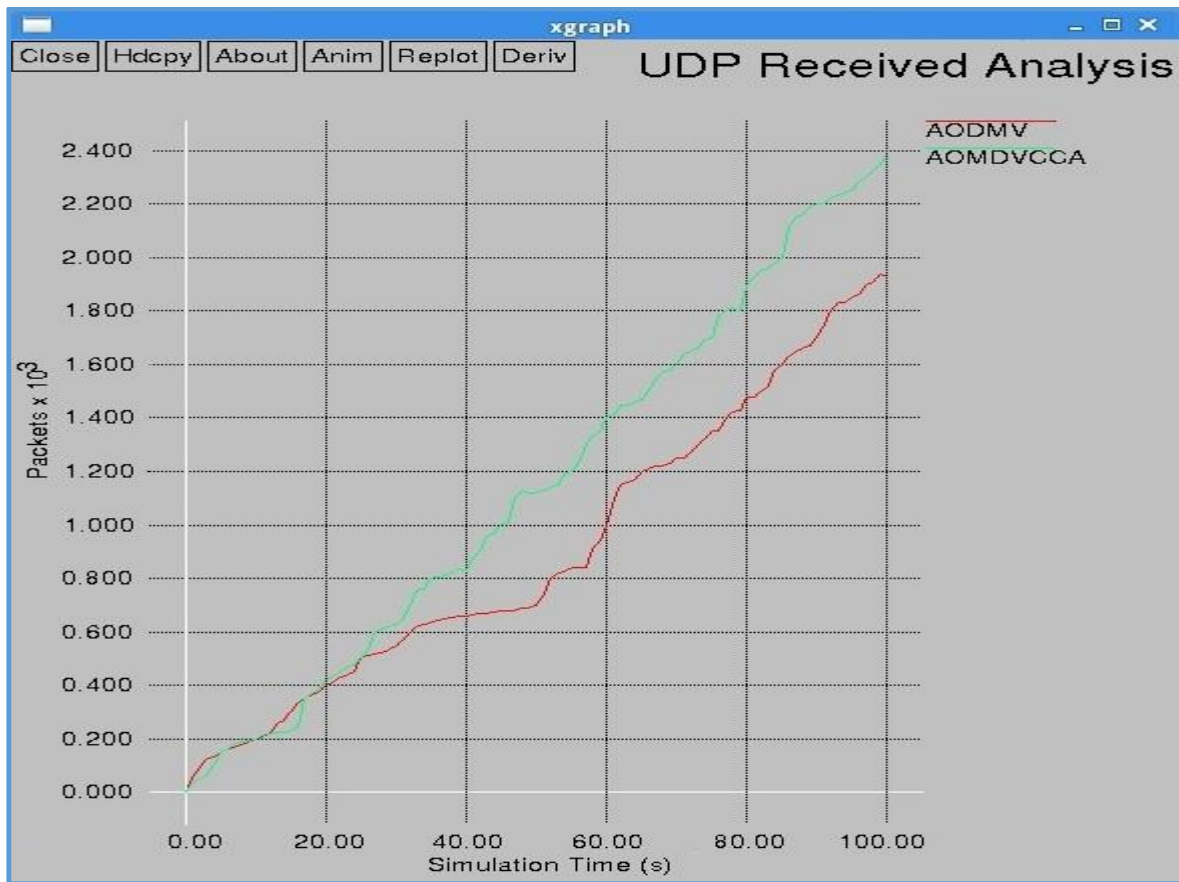


Figure 5.9: UDP packets received analysis

5.2.2.4 UDP Lost Analysis

The chief motive of packet loss in the UDP protocol is that it is not dependable for data transmission owing to incessant delivery of data packets lacking any confirmation with sender of fruitful data delivery. At this point the packet forfeiture in case of AODMV is about 460 but the packet loss in proposed system of AOMDVCCA routing is roughly only 90 packets. This very high disparity of loss of packets means the proposed scheme offers the improved routing technique that lessens the packet loss and appropriately steadies the load in network.

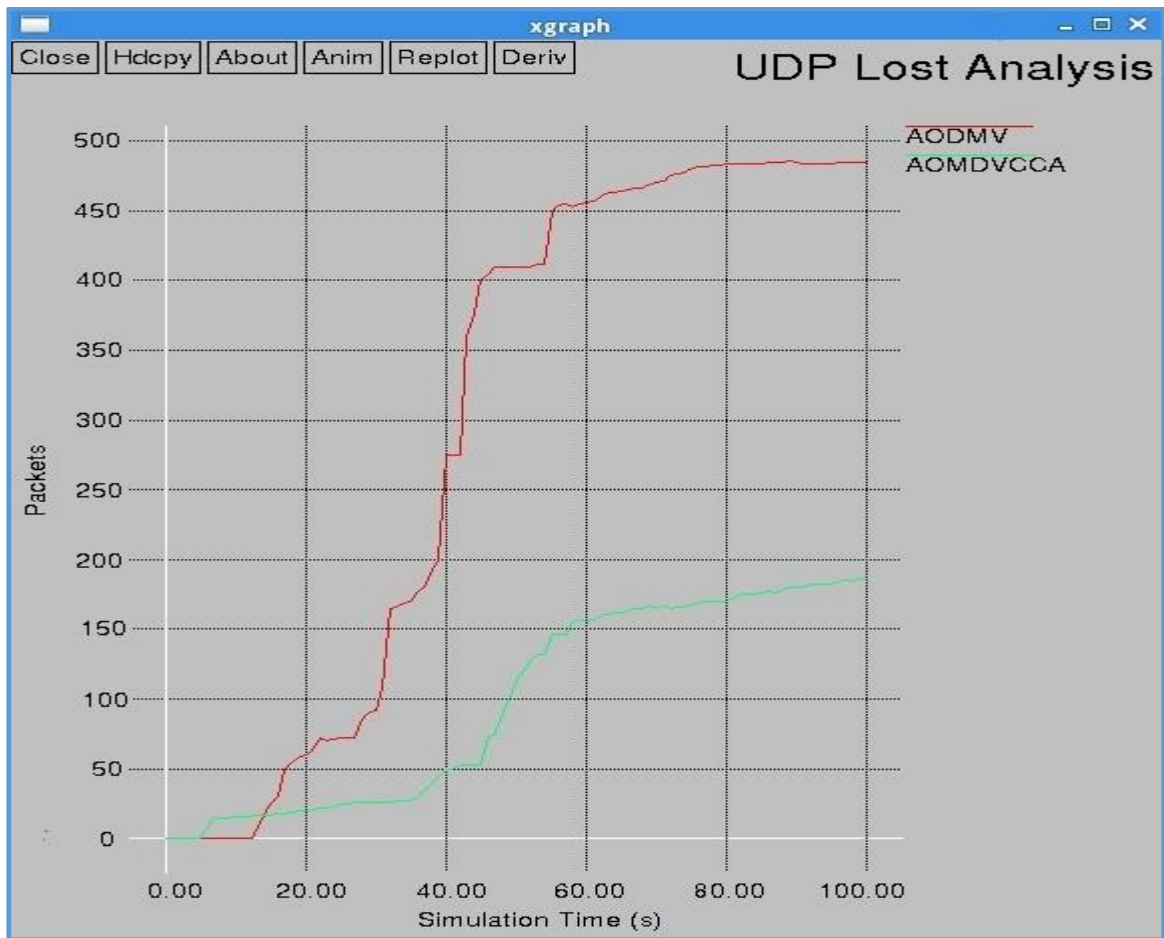


Figure 5.10: UDP packets lost analysis

5.2.3 Overall Summary of Routing Schemes

The general performance of normal AOMDV routing and proposed AOMDVCCA load balancing scheme are mentioned in table 5.1. Here the precise numeric figures are showing the better AOMDVCCA routing performance.

Parameter	AOMDV	Percentage	AOMDVCCA	Percentage
Data Send	5182		6311	
Data Received	4293		6130	
Routing Packets	5385		4925	
Packet Delivery Fraction (PDF)	82.84		97.13	
Normal Routing Load	1.25		0.8	
Number of dropped data	889		181	
Queue Full	852	5.10%	0	0.00%
Total Drop Via Congestion	1000	5.98%	235	1.34%
Total Drop	1852	11.08%	312	2.24%

Table 5.2: Overall Performances

Thus, we observe the increased performance that is brought by the proposed protocol. This is mainly attributed to the active queue management approach that is being utilised by the variation to the AOMDV protocol. The normal AOMDV performs worse when the network experiences bottlenecks. In such scenarios, the drop rates are higher.

AOMDVCCA has a lower network routing load. This is due to the low overhead traffic that the protocol generates since there are slim incidents of route failures or link breakages. The other protocol has a high overhead since the continuous route failures will lead to the network having to generate more traffic in route switching.

Of all the queues that are in the network, over 5% of AOMDV become full during the system active period. However, the new and novel proposed protocol manages its buffers such that there is virtually no full queue.

CHAPTER 6

CONCLUSION AND FUTURE WORK

Routing protocols that are on-demand which possess multipath ability, in contrast to their single path equivalents, can excellently deal with mobility-induced path failures in mobile ad hoc networks. We have suggested in this study, an on-demand multipath protocol called AOMDVCCA that extends the single path AODV protocol to calculate multiple paths. The set of multiple paths are loop-free and the alternative paths at each node are disjoint is an important guarantee of AOMDVCCA. AOMDVCCA possesses other innovative properties that include: coordination overheads that are low amongst intermediate nodes, without using source routing, it is able to discover disjoint paths, minimal additional overhead to obtain alternate paths over AODV.

The burdens for quality centred multipath routing have caused substantial thoughtfulness in the area of load balancing in MANET by researchers. In traditional mobile ad hoc routing conventions, there is an inclination to use intermediary nodes for huge number of routes. This route selection is founded on the routing protocol connection establishment method. The novel routing approach we suggested with AOMDV routing protocol amongst a pair of source and destination nodules using intermediate nodes that have plenty of resources such as processing power, bandwidth etc. This system is founded on an active queue management based on CoDel. This structure advances the routing proficiency of AOMDV multipath routing protocol by demonstrating the ability to increment the buffer. The proposed structure safeguards the presented bandwidth in the network and ensures that it is consumed proficiently ensuring better congestion control and load balancing by allocating traffic evenly. The simulation results confirm that in the proposed system the nodes efficiently handle heavier loads as compared to normal AOMDV. The proposed AOMDVCCA routing scheme provides the superior results as compared to normal AOMDV routing based on the matrices also like routing load, throughput etc.

Additional investigation needed in numerous further concerns connected to the design and evaluation of the AOMDVCCA protocol. First, to efficiently deal with the route cut-off difficulty, and calculate additional disjoint routes when source-destination pairs are far away from each other, the protocol can be enhanced. Second, the collaboration between timeout settings and AOMDVCCA performance need to be cautiously studied. Third, AOMDVCCA needs to be applied for other purposes such as soft computing load balancing based on

intelligent prediction of jamming is another issue for future work. Lastly, AOMDV relative to AOMDVCCA was only assessed using 50 node random way point movement model and CBR/UDP traffic. To see how enhancements vary with other mobility models, other traffic types such as TCP and in contrast to other multipath protocols with different node numbers is of paramount benefit.

REFERENCES

1. Neha Tiwari and Sini Shibu," Congestion Control in Multi-Flow Environment Using Multipath Routing Base in MANET", International Journal of Computer Science and Information Technologies, Vol. 5 (2), 1943-1948, 2014.
2. Shitalkumar A Jain," A Study on Congestion Aware Multipath Routing Protocols in MANET", Multidisciplinary Journal of Research in Engineering and Technology, Volume 1, Issue 1 Pg.119-128, April 2014.
3. Hitesh Gupta and Pankaj Pandey," Congestion Control Using Varying Queue Base Approach as Well as Multipath Routing Under MANET", International Journal of Engineering Research & Technology (IJERT), ISSN: 2278-0181 Vol. 2 Issue 12, December 2013.
4. Chengyong Liu, Kezhong Liu, Layuan Li, "Research of QoS – aware Routing Protocol with Load Balancing for Mobile Ad-hoc Networks", WiCom' 08, 4th International Conference on Wireless communication, 2008.
5. Jingyuan Wang, Jiangtao Wen et al., "An Improved TCP Congestion Control Algorithm and its Performance", IEEE, 2011.
6. S.Santhosh baboo and B.Narasimhan,"A Hop-by-Hop Congestion-Aware Routing Protocol for Heterogeneous Mobile Ad-hoc Networks", (IJCSIS) International Journal of Computer Science and Information Security, Vol. 3, No. 1, 2009.
7. Rajkumar, G. and K. Duraiswamy," A Fault Tolerant Congestion Aware Routing Protocol for Mobile Adhoc Networks", Journal of Computer Science 8 (5): 673-680, ISSN 1549-3636© 2012 Science Publications, 2012.
8. Xiaojiang (James) Du, Dapeng Wu, Wei Liu and Yuguang Fang, "Multiclass Routing and Medium Access Control for Heterogeneous Mobile Ad Hoc Networks", IEEE Transactions on Vehicular Technology, vol. 55, no. 1, January 2006.

9. Yi et al., "Multipath Optimized Link State Routing (MP-OLSR) protocol",
10. F. B. Hussain, Y. Cebi, and G. A. Shah, "A multievent congestion control protocol for wireless sensor networks," EURASIP Journal on Wireless Communications and Networking, Jan. 2008.
11. Jacobson, V., "Congestion Avoidance and Control," in Proceedings of ACM SIGCOMM, Stanford, CA, USA 2, 14, 51, 52, 53, 56, 190, August 1988..
12. Floyd, S. and Jacobson, V., "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, vol. 1, no. 4, pp. 397-413, 3, 7, 26, 59, 60, 1993.
13. Version 2 CSE IIT Kharagpur, 2013.
14. JAIN, R., RAMAKRISHNAN, K., AND CHIU, D.-M., "Congestion avoidance in computer networks with a connectionless network layer", Tech. Rep. DEC-TR-506, Digital Equipment Corporation, Aug. 2008.
15. W. Bux and D. Grillo, "Flow Control in Local-Area Networks of Interconnected Token Rings," IEEE Transactions on Communications, Vol. COM-33, No. 10, pp. 1058-66, October 1985.
16. A. Giessler et al., "An Investigation by Simulation," Computer Networks, Vol. 1, No. 3, pp. 191-204 July 2006.
17. John Nagle, "Congestion Control in TCP/IP Internetworks," Computer Communication Review, Vol. 14, pp. 11-17 No. 4, October 1984.
18. E. Egea-López et al., "Simulation Tools for Wireless Sensor Networks", Summer Simulation Multi-conference – SPECTS, 2005.
19. Fei Yu, "A Survey of Wireless Sensor Networks Simulation Tools", Wiley Computer Publishing, John Wiley & Sons, Inc., 2009.

20. Jaydip Sen,” Wireless Sensor Networks - Platforms, Tools and Simulators”, Innovation Labs, Tata Consultancy Services, Bangalore, India, 2010.
21. Lei Shu, Chun Wu, Yan Zhang, Jiming Chen, Lei Wang, Manfred Hauswirth, “NetTopo: Beyond Simulator and Visualizer for Wireless Sensor Networks”, Future Generation Communication and Networking - FGCN, Volume1, pp. 17-20, ISBN: 978-0-7695-3431-2, 2008.
22. Lewis Girod, Jeremy Elson, Alberto Cerpa, Thanos Stathopoulos, Nithya Ramanathan, Deborah Estrin, “EmStar: a Software Environment for Developing and Deploying Wireless Sensor Networks” , USENIX Technical Conference, 2004.
23. E. Egea-Lopez, J. Vales-Alonso, A. S. Martinez-Sala, P. Pavon-Marino, J. Garcia-Haro, “Simulation Tools for Wireless Sensor Networks”, Summer Simulation Multi-conference, SPECTS, pp.2-9 , 2005.
24. Sourendra Sinha, Zenon Chaczko, Ryszard Klempous, “SNIPER: A Wireless Sensor Network Simulator”, Computer Aided Systems Theory- EUROCAST, Volume 5717/2009, pp. 913-920, 2009.
25. Larry L. Peterson and Bruce S. Davie, “Computer Networks – A Systems Approach”. San Francisco, Morgan Kauffmann Publishers Inc., ISBN 1-55860-368-9, 2008.
26. Sharma G., “Routing in Wireless Sensor Networks”, Master Thesis, Computer Science and Engineering Dept., Thapar Univ., Patiala, 2009.
27. Akyildiz I. F.; Su W.; Sankarasubramaniam Y. and Cayirci C.,”A Survey on Sensor Networks”, IEEE communication magazine, 2002.
28. Acs G. and Butty’an L., "A Taxonomy of Routing Protocols for Wireless Sensor Networks", Budapest University of Technology and Economics, Hungary, 2007.

29. Luis J. et al, "Routing Protocols in Wireless Sensor Networks", Spain, www.mdpi.com/journal/sensors, 2009.
30. Hussein Mohammed Salman," Survey of Routing Protocols in Wireless Sensor Networks", International Journal of Sensors and Sensor Networks. Vol. 2, No. 1, pp. 1-6. doi: 10.11648/j.ijssn.20140201.11, 2014.
31. Mahesh K. Marina and Samir R. Das," On-demand Multipath Distance Vector Routing in Ad Hoc Networks", University of Cincinnati, Cincinnati, OH 45221, 2011.
32. C. E. Perkins and E. M. Royer," Ad hoc on-demand distance vector routing," In Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, pages 90–100, Feb 1999.
33. C. E. Perkins, E. M. Royer, and S. R. Das," Ad Hoc On-Demand Distance Vector (AODV) Routing", <http://www.ietf.org/internet-drafts/draft-ietf-manet-aodv-07.txt>, IETF Internet Draft (work in progress), Nov 2000.
34. P. Johansson, T. Larsson, N. Hedman, and B. Mielczarek, "Routing Protocols for Mobile Ad-hoc Networks – A Comparative Performance Analysis", In Proceedings of the IEEE/ACM MOBICOM, pages 195–206, 1999.
35. C. E. Perkins, E. M. Royer, S. R. Das, and M. K. Marina," Performance Comparison of Two On-demand Routing Protocols for Ad Hoc Networks", IEEE Personal Communications, 8(1):16–28, Feb 2008.
36. C. E. Perkins and P. Bhagwat," Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers", In Proceedings of the ACM SIGCOMM, pages 234–244, August 1994.
37. R. Castaneda and S. R. Das," Query Localization Techniques for On-demand Protocols in Ad Hoc Networks", In Proceedings of the IEEE/ACM MOBICOM, pages 186–194, 1999.

38. S.-Y. Ni, Y.-C. Tseng, Y.-S. Chen, and J.-P. Sheu, "The Broadcast Storm Problem in a Mobile Ad Hoc Network", In Proceedings of the IEEE/ACM MOBICOM), pages 151–162, 1999.
39. D. Johnson and D. Maltz, "Dynamic Source Routing in Ad Hoc Wireless Networks. In T. Imielinski and H. Korth", editors: Mobile computing, chapter 5. Kluwer Academic, 1996.
40. Rubia Singla et al, "Node-Disjoint Multipath Routing Based on AOMDV Protocol for MANETS", (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 5 (4), 5491-5496 , 2014.
41. Mahesh K. Marina^{1*},[†] and Samir R. Das², "Ad hoc on-demand multipath distance vector routing WIRELESS COMMUNICATIONS AND MOBILE COMPUTING Wireless Communication and Mobile Computing"; 6:969–988 Published online in Wiley Inter Science (www.interscience.wiley.com). DOI: 10.1002/wcm.432, 2006.
42. Jubin J, Tornow JD, "The DARPA packet radio network protocols", Proceedings of IEEE 75(1): 21–32, 2003.
43. Ogier R, Shacham N, "A distributed algorithm for finding shortest pairs of disjoint paths", In Proceedings of IEEE Infocom, 2010.
44. Sidhu D, Nair R, Abdallah S, "Finding disjoint paths in networks", In Proceedings of ACM Sigcomm, 2011.
45. Murthy S, Garcia-Luna-Aceves JJ, "An efficient routing protocol for wireless networks," ACM/Baltzer Mobile Networks and Applications; 1(2): 183–197, 2006.
46. Ducksoo Shin, Jonghyup Lee, Jaesung Kim, And Jooseok, "A²OMDV : An Adaptive Ad Hoc On-Demand Multipath Distance Vector Routing Protocol Using Dynamic Route Switching", Journal of Engineering Science and Technology Vol. 4, No. 2 171 – 183 © School of Engineering, Taylor's University College, 2011.

47. Tanvi Sharma,” Controlling Queue Delay (CoDel) to counter the Bufferbloat Problem in Internet”, International Journal of Current Engineering and Technology, E-ISSN 2277 – 4106, P-ISSN 2347 – 5161, ©2014 INPRESSCO, 2014.
48. Tarun Jain, Annappa B., Mohit P. Tahiliani, ”Performance Evaluation of CoDel for Active Queue Management in Wired-cum-Wireless Networks, Fourth International Conference on Advanced Computing & Communication Technologies, Surathkal, Karnataka, India, 2014.
49. K. Nichols and V. Jacobson, “Controlling Queue Delay,” ACM Queue Magazine: Networks, vol. 10, no. 5, pp. 68–81, [Online] Available: <http://queue.acm.org/detail.cfm?id=2209336>, May 2012.
50. Dipesh M. Raghuvanshi, Annappa B., Mohit P. Tahiliani, “On the Effectiveness of CoDel for Active Queue Management,” 3rd International Conference on Advanced Computing & Communication Technologies, pp. 107–114, 2013.
51. K. Fall and K. Varadhan, The NS Manual, November 18, 2005, http://www.isi.edu/nsnam/ns/doc/ns_doc.pdf. 25 July 2009.
52. Aliff Umair Salleh, Zulkifli Ishak , Norashidah Md. Din, Md Zaini Jamaludin,” Trace Analyser for NS-2”, 4th Student Conference on Research and Development (SCORED 2006), Shah Alam, Selangor, Malaysia, 27-28 June, 2008.
53. Naman Jain and Tejaswi Agarwal,” Simulate Your Network with NS2”, Proceedings of the IEEE, vol.86, no. 5, pp. 974-997, May 2012.