Load Dependent Dynamic Path Selection in Multi-Radio Hybrid Wireless Mesh Networks

By
Raja Farrukh Ali
2010-NUST-MS-CCS-10

Supervisor
Dr. Adnan Khalid Kiani
Department of Electrical Engineering

A thesis submitted in partial fulfilment of the requirements for the degree of Masters in Computer and Communication Security (MS CCS)

In
School of Electrical Engineering and Computer Science,
National University of Sciences and Technology (NUST),
Islamabad, Pakistan.

(May, 2014)

Approval

It is certified that the contents and form of the thesis entitled "**Load Dependent Dynamic Path Selection in Multi-Radio Hybrid Wireless Mesh Networks**" submitted by **Raja Farrukh Ali** has been found satisfactory for the requirement of the degree.

Advisor: **Dr. Adnan K. Kiani**

Signature: _____
Date: _____

Committee Member 1: **Dr. Zawar Shah**

Signature: _____
Date: _____

Committee Member 2: **Dr. Hassaan Khaliq**

Signature: _____
Date: _____

Committee Member 3: **Dr. Syed Ali Hassan**

Signature: _____
Date: _____

To my Parents,
whose love and support enabled me to see this day

Certificate of Originality

I hereby declare that the research work titled "**Load Dependent Dynamic Path Selection in Multi-Radio Hybrid Wireless Mesh Networks**" is my own work to the best of my knowledge. It contains no materials previously written or published by any other person, nor material which to a substantial extent has been accepted for the award of any degree or diploma at NUST or any other education institute, except where due acknowledgment, is made in the thesis. Any contribution made to the research by others, with whom I have worked at NUST or elsewhere, is explicitly acknowledged in the thesis.

I also declare that the intellectual content of this thesis is the product of my own work, except to the extent that assistance from others in the thesis' design and conception or in style, presentation and linguistic is acknowledged. I also verified the originality of contents through plagiarism software.

Author Name: **Raja Farrukh Ali**

Signature: _____

Acknowledgment

First and foremost, I am greatly indebted to Almighty Allah for He has been very kind to me during this whole journey. I am thankful to the faculty of SEECS for their guidance, cooperation and counselling throughout my research work. I would like to express my special gratitude to my thesis research supervisor Dr. Adnan Khalid Kiani for his support, vision, mentoring, kindness and guidance. I have benefited from his useful assistance, feedback and critique. I am also thankful to my committee members Dr. Zawar Shah, Dr. Syed Ali Hassan and Dr. Hassaan Khaliq for their support, availability and useful input in my research work.

I extend my gratitude to my classmates and friends at SEECS NUST for being great comrades in this journey. In particular, I wish to thank Faisal Bhutta, Muhammad Ali and Rahat Masood for their invaluable help. Lastly, I am grateful to my family for their love and support.

Abstract

Hybrid Wireless Mesh Networks provide improved connectivity and reliability over traditional wireless networks. However due to rapidly changing traffic patterns, such networks are more likely to experience congestion which leads to data loss. Inability to differentiate between congested links and select best possible paths dynamically can be very expensive in a rapidly changing network topology. This thesis proposes D-WCETT (Dynamic Weighted Cumulative Expected Transmission Time), an enhanced version of the WCETT routing metric, with focus on forming routes based upon network load information. The proposed routing metric dynamically selects paths with least level of congestion by taking into consideration locally available queue information. We present an evaluation of our implementation via extensive simulations and results indicate that the proposed metric significantly outperforms its predecessor by virtue of its ability to distinguish and dynamically select least congested paths in a multi-radio environment.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This chapter is directed to provide the overview of basic concepts and approaches that are primarily important for this research work and also intends to define our targeted research area. It serves as a roadmap for the rest of the thesis and describes the overall organization of the thesis.

1.1 Wireless Mesh Networks

Wireless Mesh Networks provide robust and reliable wireless communication with minimal cost. Their dynamic, self-organizing and self-configuring nature enables nodes to establish and maintain mesh connectivity among themselves. A Wireless Mesh Network (WMN) consists of Mesh Routers and Mesh Clients in which Mesh Routers form the static network backhaul, possess better power resources and are equipped with multiple radios. They also provide gateway functionality by connecting the WMN to other networks. Mesh Clients are mobile nodes having limited battery resources and are mostly equipped with a single radio. They extend the reach of the network owing to their mobility but are dependent on Mesh Routers for their connectivity. WMNs are of three main types [1]; Infrastructure WMN consists of static Mesh Routers with Mesh Clients connected only via a single hop and devoid of any packet forwarding functionality. Client WMN represents a true Mobile Adhoc Network (MANET) as it consists of only Mesh Clients which communicate among each other directly and perform routing and configuration functions themselves. Hybrid WMN merges the connectivity model of Infrastructure and Client networks in which both Mesh Routers and Clients are involved in routing and packet forwarding and Mesh Clients can gain access to the network via multiple client hops.

1.2 Hybrid Wireless Mesh Networks

A Hybrid WMN presents the most generic and prevalent type of mesh network. Multi-radio nodes can significantly enhance the capacity of the network by utilising the multiple orthogonal channels available to them and overcome limitation of half-duplex single radio nodes. Despite their widespread use, Hybrid WMNs face challenges in their deployment. Mobility poses a complex problem as it results in frequent route disruption and requires significant efforts for route maintenance under dynamic conditions. Moreover contention for wireless medium among nodes results in intra-channel and inter-channel interference. In order to overcome these limitations, routing protocols are constantly evolving to support mobility, reduce interference and establish reliable routes.

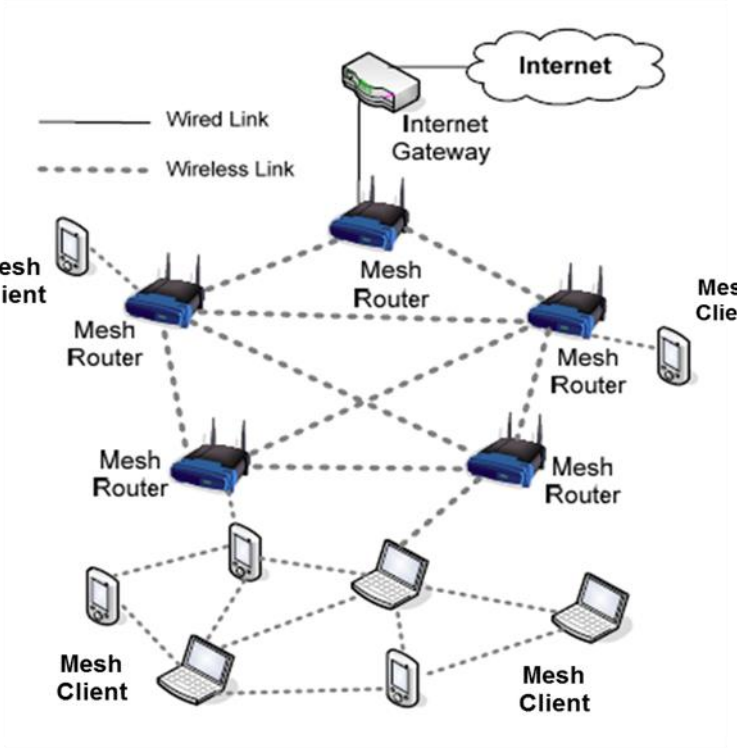


Figure 1 A typical Hybrid Wireless Mesh Network

1.3 Routing Protocols for Wireless Mesh Networks

Routing protocols are designed keeping in view the networks in which they are to be employed. While a static network aims at improving the performance of individual transfers, a mobile network needs a protocol which can effectively maintain connectivity. A routing protocol employs a routing metric to select best possible paths between nodes. Routing metrics can be based on factors such as traffic, energy, topology and mobility [2] with hop count, delay, packet

loss ratio, signal strength being a few examples. Traditionally, routing protocols for wireless adhoc networks have used hop count as the default metric but it has been shown [3], [4] that shortest-path routing is not the most suitable metric as it tends to favour longer but slower paths. Subsequently proposed enhancements have used packet loss and channel interference among other factors that a routing protocol should consider in order to find better links. Such metrics tend to decrease path length, end to-end delay and make effective utilization of available network bandwidth by reducing co-channel interference. There are two basic types of co-channel interference: intra-flow and inter-flow interference. Intra-flow interference is created by simultaneous transmission of data belonging to a single data flow and route. Inter-flow interference occurs due to simultaneous transmission of unrelated data, i.e. belonging to independent flows and routes.

1.4 Problem Statement

Hybrid WMNs however present a case where changing network topology adds another dimension to existing metric considerations. A routing metric should be able to adapt itself according to the dynamic nature of the network and its measurements should be reflective of the current network conditions. As will be shown later, a number of routing metrics designed for multi-hop multi-radio networks fail to meet this requirement. While employing multiple factors to make better overall judgement, they usually associate a tuneable factor to alternate between different link quality indicators. This tuneable factor cannot be changed dynamically and hence always gives a pre-determined weight to its constituent path functions. There is a need to make routing metrics adaptable to current network conditions which should enable them to increase the routing efficiency in a network.

1.5 Thesis Contributions

In this thesis we present Dynamic Weighted Cumulative Expected Transmission Time (D-WCETT) metric. The key contributions of this thesis include a routing metric with the ability to select least congested links along a path by taking into consideration the interface queue length of the link. Moreover this selection is dynamic and is reflective of the prevalent load level of a link. Since this information is locally available at the node, it also does not incur any additional communication overhead. D-WCETT has been implemented in a multi-radio Hybrid WMN and has been evaluated at different mesh client speeds, packet transmission rates and traffic loads. Results indicate significantly better performance of the D-WCETT metric with

dynamic path selection over WCETT. This improved performance is attributed to the integration of the metric with the contemporary network load.

1.6 Thesis Outline

The thesis is structured as follows:

Related Work (Chapter 2): This chapter presents related previous work, which has been carried out in the field of routing metrics for adhoc routing protocols. These works include various routing metrics that have evolved with the passage of time for adhoc wireless networks as well has hybrid wireless mesh networks.

Research Methodology (Chapter 3): This chapter explains the limitations of WCETT routing metric and presents the proposed extension, Dynamic WCETT. The additional parameters required for computation of D-WCETT are explained in detail.

Implementation (Chapter 4): The implementation of D-WCETT is discussed in this chapter. Establishing the simulation test bed in NS-2 and its configuration are presented.

Simulation Results and Analysis (Chapter 5): The performance of the proposed D-WCETT routing metric is evaluated in this chapter. The variety of tests performed and their results are analysed.

Conclusion (Chapter 6): This chapter offers some concluding remarks and examines the possibility of future extensions.

Chapter 2

Related Work

Related work focusing on routing metrics for wireless adhoc networks is presented in the following sections.

2.1 Hop Count

Hop count is the most common metric used in adhoc routing protocols. However minimum hop count favours longer links and fails to consider important parameters such as transmission rates and link quality, consequently resulting in non-optimal performance [2]–[4].

2.2 Expected Transmission Count

Expected Transmission Count (ETX) [3] establishes a path depending upon the expected number of MAC layer transmissions which will result in delivering a packet successfully.

$$ETX = \frac{1}{d_f \cdot d_r}$$

ETX uses probe packets for its calculation such that probe packets successfully received from a neighbouring node indicate the reverse delivery ratio (dr) whereas forward delivery ratio (df) is measured through packets successfully received by a neighbour. These ratios are then used to calculate the probability of a successful transmission. Although ETX performs better than hop count, its limitations include inability to differentiate between links of different bandwidths and susceptibility to co-channel interference in multi-radio networks.

2.3 Weighted Cumulative Expected Transmission Time

The Weighted Cumulative Expected Transmission Time (WCETT) [4] was developed specifically in the context of multi-radio networks. The main component of this metric is ETT which builds upon the ETX metric by incorporating packet size (S) and bandwidth of the link (B) and denotes the expected time required to successfully send a packet.

$$ETT = ETX \cdot \frac{S}{B}$$

The purpose of using multi-radio nodes is to maximise the available bandwidth by using multiple orthogonal channels available in the spectrum. It is desired that a packet uses multiple channels along a path such that all channels are equally utilized. A path which is most channel diverse is least likely to experience co-channel interference. WCETT of a path p is defined as:

$$WCETT_p = (1 - \beta) \cdot \sum_{l \in p} ETT_l + \beta \cdot \max_{1 \le j \le k} X_j$$

where $max\ Xj$ is the sum of ETTs on the most consumed channel and k is the total number of channels available. β is a tuneable parameter used to alternate between the first term i.e. sum of ETTs which indicates path length and the second term i.e. $max\ Xj$ which signifies channel diversity. The path with least WCETT value is chosen such that when β is towards 0, the WCETT value will reflect the path with least aggregate ETTs and when β is towards 1, the WCETT will favour a path in which all channels have been equally used resulting in a channel diverse path.

WCETT performs better than ETX in multi-radio networks owing to its ability to exploit channel diverse paths and address co-channel interference. However WCETT has two important limitations. First limitation is WCETT's inability to adapt to changing network load. β has been identified as critical to the performance of WCETT. However its value cannot be changed dynamically according to the network behaviour and hence it has to persist with a certain predefined balance between path length and channel diversity, resultantly unable to take advantage of its flexibility. Second limitation is WCETT's channel diversity component fails to address inter-flow interference completely and only partially addresses intra-flow interference because it is unable to account for interference caused by the use of same channel in successive links along a path. Few metrics such as MIC [5], iAWARE [6] have been

proposed to address the limitations of WCETT with regards to inter-flow interference. Both employ ETT in addition to other parameters and show improved performance over WCETT. However these metrics are more focused on interference-aware routing and do not address the aforementioned first shortcoming. This is especially the case for iAWARE as it employs a tuneable parameter which remains constant and hence fails to find an effective way to alternate between metric components signifying path length and channel interference.

2.4 WCETT-LB

WCETT with Load Balancing (WCETT-LB) [7] modifies WCETT by introducing load balancing at mesh routers with the help of average queue length.

$$WCETT - LB_p = WCETT_p + L_p$$

where load Lp of a path p is given by:

$$L_p = \sum_{nodei \in p} \frac{QL_i}{b_i} + \min(ETT)N_i$$

QLi is the average queue length at node i, bi is the transmission rate and min (ETT) reflects traffic concentration at node Ni. WCETT-LB's performance has only been evaluated against hop count which fails to show whether it achieves any improvement over WCETT or not. Moreover it carries with it all the limitations of WCETT, including failure to switch between path length and channel diversity dynamically. Also WCETT-LB has only been analysed in a single-radio environment.

2.5 LARM

Load-Aware Routing Metric (LARM) [8] also utilizes ETT and queue information to form load balanced paths in a multi-radio wireless mesh network.

$$LARM = (1 - \beta) \cdot \sum_{j=1}^{k} CL(j) + \beta \cdot \max_{1 \le j \le k} CL(j)$$

where Channel Load CL of channel j is defined as:

$$CL(j) = \sum_{l \in channelj} ETT_l \cdot Q_l$$

Ql is the average number of packets buffered on a link l between two neighbouring nodes and k is the total number of channels available. First term in LARM shows accumulated load of occupied channels on the routing path whereas the second term reflects channel diversity. LARM struggles to show any significant performance improvement over WCETT while continuing to use a static β value. It also fails to address the issue of inter-channel interference.

2.6 Other Proposed Metrics

Recently proposed metrics such as Load-Aware Airtime Link Cost Metric [9] and Neighbourhood Load Routing (NLR) metric [10] use queue length in their metric considerations. Load-Aware Airtime has been evaluated against WCETT-LB and LARM but fails to show any substantial improvement while also using a static tuneable parameter. NLR has been developed for single radio mesh networks and no evaluation is offered for multi-radio nodes.

Chapter 3

Research Methodology

This chapter presents the research problem and our research methodology that we have used throughout the research work. The proposed solution is explained in detail afterwards.

3.1 Problem of Static β

The Dynamic WCETT (D-WCETT) consists of a number of modifications to WCETT metric and hence it is important to provide a brief overview of WCETT's original implementation before discussing our proposed extensions. WCETT was implemented on a source driven, proactive routing protocol called Multi-Radio Link Quality Source Routing (MR-LQSR).The protocol identifies all network nodes and computes the ETT values of all paths. Depending upon the value of β it either favours shortest path or channel diverse path. The value of β is static and cannot be changed once assigned to a network. Hence a wireless network using WCETT is either configured to favour shortest path (β=0.1) or channel diverse paths (β=0.9) or a compromise between the two (β=0.5). This presents a drawback as WCETT cannot adapt to a changing network topology such as the one presented by a Hybrid WMN. Moreover it is also shown in [4] that when load levels are increased and the throughput is evaluated for different values of β, the maximum throughput is achieved using the lowest value of β=0.1. Therefore WCETT's performance is dependent upon the value of β and results conclude that at high load levels, network throughput can be maximized by using lower values of β. Linking β to a parameter that can accurately reflect the network load may result in better performance.

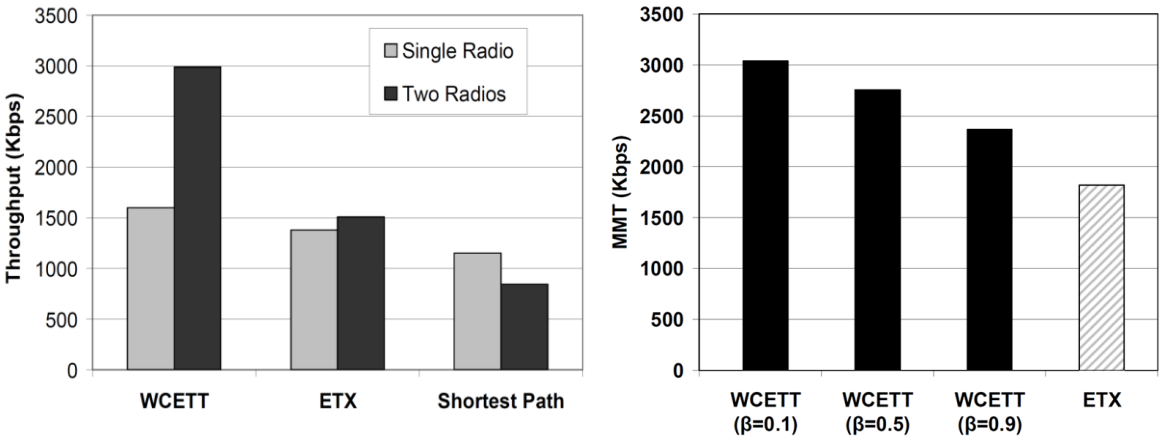


Figure 2 Comparison of Throughput after Increasing Load

3.2 Interface Queue length (IFQ)

To calculate network load we have used interface queue length (IFQ) which is a parameter indicating the total number of packets in the queue of a channel belonging to a node [2]. The IFQ is a drop-tail buffer present at the MAC layer of IEEE 802.11 radios and contains outbound frames for the physical layer. A build-up of frames indicates congestion due to high network traffic or low link quality. WMNs have a shared wireless medium which invites contention among nodes. This contention leads to congestion in the outbound IFQ of nodes. Queue lengths are properties of network nodes. Every network node possesses ingress and an egress queue in which incoming and outgoing packets are stored if the interface is unable to forward them immediately. The queue length gives an indication of the current state of the device: An empty queue is reflective of the fact that the device can process more traffic, whereas a full queue depicts a contented interface which cannot handle more packets. Usually, the queue length is small and as it becomes full, the node starts dropping packets. D-WCETT uses the IFQ as an indicator of link congestion. IFQ is indicative of a number of parameters including link quality, link load and external interference. The most significant advantage of using IFQ is that it is locally available at the node and hence can be used by nodes to make a decision about route selection without incurring additional communication overhead. Also IFQ can be calculated instantaneously making D-WCETT agile and able to adapt to dynamic networks.

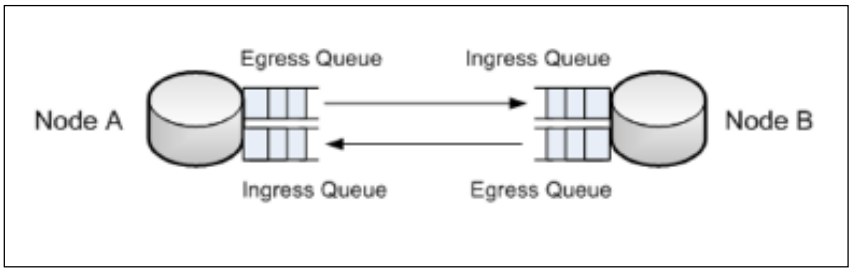


Figure 3 Ingress and Egress Queues

3.3 IFQ Window Size (IFQ WIN SZ)

For parameters that are subject to high variation, it is usually desired that short-term variations do not influence the value of a metric, as this could cause disproportionate adaptations of the metric and would increase the risk of self-interference. Therefore metric measurements should be filtered over time by making use of a moving average. In D-WCETT the IFQ value is calculated as a moving average, filtered over a fixed interval of time called IFQ Window Size.

3.4 Queue Discharge Interval (QDI)

To cater for interfaces with dissimilar data rates, we divide the IFQ length by the bandwidth of the channel (BW). This term is referred to as the Queue Discharge Interval (QDI).

$$QDI = \frac{IFQ}{BW}$$

The QDI value represents the time required by a packet to remain in the IFQ before transmission. By normalizing the value it is ensured that the IFQ of different nodes with varying channel bandwidths are comparable

3.5 Integrating β with Network Load

Since QDI gives a real-time and accurate picture of the network load, we calculate β such that when a link has a high value of QDI, the β values for calculating WCETT metric are lower. Hence D-WCETT does not have a constant pre-defined value of β but a dynamic value, changing according to the level of congestion in the link. D-WCETT makes use of a simple relation to express the inverse relationship between β and link load at a given instant.

$$\beta = 1 - QDI$$

Whenever a route request is received at a node, the current value of link congestion on all

available channels is obtained through the IFQ value. Links operating on different bandwidths are compensated to get QDI which is then normalized between 0 and 1, where 0 indicates an empty queue and 1 indicates a full queue. Depending upon the value of QDI, a value of β is selected according to (9). Using this β value, D-WCETT metric is calculated for all channels and route request is forwarded on the channel with the least D-WCETT value. In essence D-WCETT gives higher weight to path length (sum of ETTs) as compared to channel diversity when links are having higher load levels and vice versa.

Chapter 4

Implementation

This chapter elaborates on the implementation scheme of the proposed solution. The setting up of the simulation test bed used during the thesis is explained.

4.1 Simulation Test bed

4.1.1 Overview

The simulation software used in this thesis is the network simulator-2 (NS-2) version 2.1b9a which was installed on Red Hat Enterprise Linux 3 Operating System (OS). For the ease of work, a virtual machine software known as VMware Server was used which allowed working simultaneously on both Windows and Linux Red Hat 3. It helped in running simulations on Linux OS while at the same time analysing the already completed tests on Windows XP. Some of the existing ad-hoc routing protocols are pre-built in NS-2, namely, AODV, DSR, DSDV and TORA. Virtual machine software, NS-2 and the details of the simulator are presented in the following paragraphs. Furthermore, the values for specific simulation parameters and the methodology for generating it are explained. Finally, the simulation procedure is described.

4.1.2 Network Simulator-2 (NS-2)

The simulator used in the thesis is an open-source network simulator called NS-2. NS-2 is a discrete event simulator targeted for network research and provides support for simulating protocols on conventional networks and wireless networks. NS-2 is an object oriented simulator, written in C++, with an OTcl interpreter as a frontend. 10. An NS-2 simulation requires an OTcl script for network configuration, a mobility pattern describing node

movement, a traffic pattern describing data traffic, and files describing coordinates for obstacles and pathways. In addition, it is necessary to trace the mobility model used, as well as the type of traffic at which level, i.e., routing, media access control (MAC) or application, into some output files for post-simulation analysis. Figure 18 shows the flow of events for an OTcl file run in NS-2. Depending on the user's purpose for an OTcl simulation script, simulation results are stored as trace files, which can be loaded for analysis by an external application.

(a) A NAM trace file (file.nam) for use with any NS-2 animator tool.

(b) A Trace file (file.tr) which has to be parsed to extract useful information.

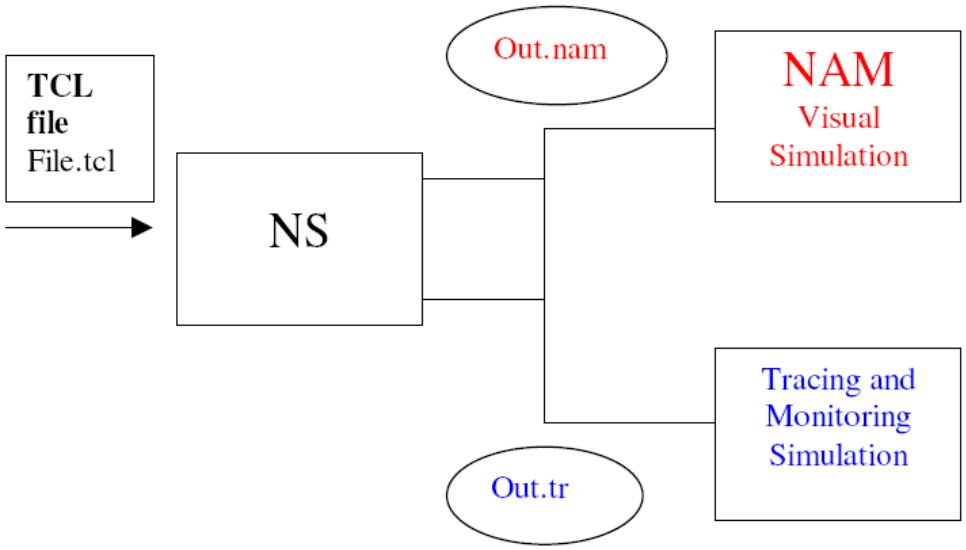


Figure 4 Flow of events for a Tcl file run in NS-2

4.1.3 Network Animator (NAM)

NAM is a Tcl/TK based animation tool for viewing network simulation traces and real world packet trace data. NAM was designed to read simple animation event commands from a large trace file. The first step to use NAM is to produce the nam trace file (file.nam). The trace file contains topology information, e.g., nodes, links, as well as packet traces. During an NS-2 simulation, user can produce topology configurations, layout information, and packet traces using tracing events in ns. It is up to the user whether to generate a trace file for a particular simulation or not. The following lines can be commented in the tcl code if the user does not want nam trace file to be generated for a particular simulation.

```
set tracefd [open $opt(tr) w]
#set namtrace    [open trace.nam w]
#$ns_ use-newtrace
$ns_ trace-all $tracefd
#$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)
```

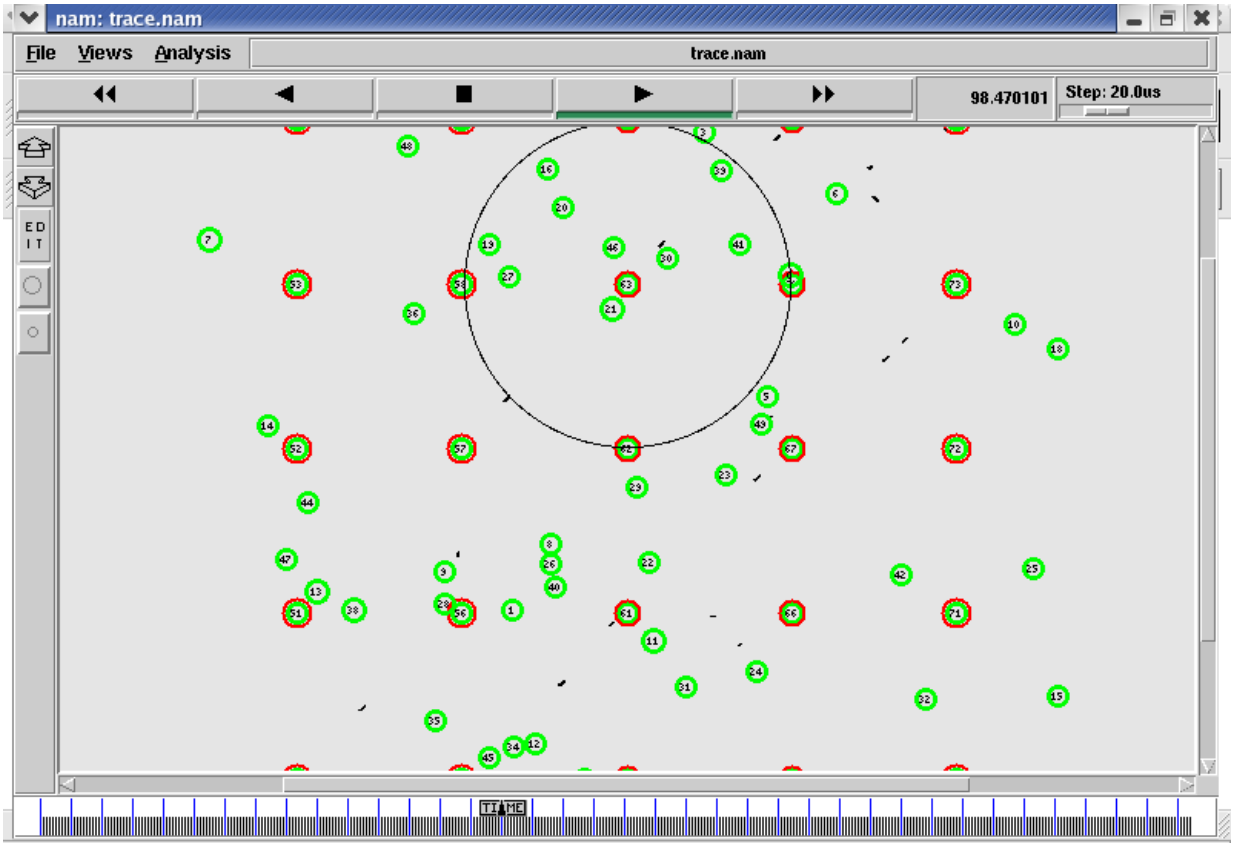


Figure 5 NS-2's Network Animator (NAM)

4.1.4 VMware Workstation

VMware software provides a completely virtualized set of hardware to the guest operating system. Moreover, a number of virtual machines with different operating systems running on them may be setup is such a way that they are simulated as connected through a LAN whose settings can be adjusted by the user as required. The machines setup on VMware has an advantage of mobility that the backup of these created machines can be copied and moved to any other PC using VMware and they need not to be installed repeatedly on different workstations. VMware's desktop software runs on Microsoft Windows, Linux, and Mac OS X.

NS-2 can be installed either in a UNIX (or Linux) or Windows (2000 and XP) environment. For the Windows environment, it is necessary to install a UNIX emulator such as Cygwin prior

to the installation of the NS-2 software. One disadvantage of performing simulations in the Windows environment is the issue of software stability. Moreover, most 3rd party software extensions which are available as contributed codes to NS-2 are neither available nor well supported for the Windows platform. The simulations conducted for this thesis were done using the Linux Red Hat 3 operating system (OS).

A hard-disk size of at least 8 GB should be allocated for storage of the simulation files. This estimate is based on the fact that a typical simulation using 75 nodes and 900 seconds simulation time using AODV can generate up to 2 GB of trace and NAM files separately. For AODV, the actual duration of computer run time is generally between 5 to 50 minutes. Depending on the packet transmission rate it can exceed 3 hours. Although the actual run time may be reduced substantially by reducing the simulation time, it is not advisable because there is a potential risk that the results may be inaccurate due to insufficient time provision for convergence in network routing. Moreover it can also be reduced by not generating the trace file and by using the generated results on the screen to give the required parameters through a C++ code which would be explained later.

4.2 D-WCETT Implementation in AODV

To implement this change, the C++ code of AODV-WCETT was modified to extract the number of packets that are in a node's queue. This information is available at the MAC layer. The IFQ value is computed as a moving average i.e. the values are not calculated instantaneously but are filtered over a particular interval of time. For this purpose the interface queue window size (IFQ WIN SZ) of 0.1 seconds was selected in the simulations. Research has shown that the value of 0.1 seconds yield optimum results for a 75 node network, which is similar to the one simulated in this thesis. When the value of IFQ WIN SZ is increased beyond 0.1 seconds, computation of routing metrics does not converge fast enough. This is due to the relatively large time window during which link statistics are gathered and averaged. Increasing IFQ WIN SZ above 0.1 seconds results in the metric lagging behind the actual network topology, which in turn results in the selection of stale routes and a relatively large number of route breaks.

The IFQ length was specified to be 50 packets, which means that a node interface can hold up to a maximum of 50 packets in its queue after which it will start dropping packets based on FIFO (First In First Out) scheduling. The value of IFQ was normalized between 0 and 1 by

multiplying with a factor of 0.02 (1/50=0.02). Hence an IFQ value of 1 corresponds to a queue of 50 packets whereas an IFQ value of 0 corresponds to an empty queue. From this value of IFQ, the value of QDI was calculated. The bandwidth of all network interfaces was pre-configured to a certain value as discussed earlier. After obtaining QDI value, β is calculated using equation 7. This value of β is then used in calculating WCETT metric from equation 3 which then forms the basis of which RREQ needs to be forwarded and which is to be discarded. Some excerpts of modifications in C++ code are as follows.

```
double temp_etx, Xj, WCETT;
int temp_iface;
double IFQ=0; //Current Channel average IFQ value
.
.
IFQ=ifqueuelist[Cur_Chan-1]-
>mov_avg_ifq_len(IFQ_WIN_SZ)*0.02; /*normalize between 0
and 1 for a queue length of 50, 1/50=0.02 or (50 * 0.02 =
1)*/

double BETA=1-IFQ; //Calculate β

//Find WCETT
Xj=0;
for (int i=0;i<numifs;i++) {
if (rq->rq_ch_etx[i] > Xj) Xj=rq->rq_ch_etx[i];}

rq->rq_etx= rq->rq_etx + temp_etx;

WCETT=(1 - BETA) * rq->rq_etx + BETA * Xj;

if ( (rq->rq_src_seqno > rt0->rt_seqno ) || ((rq-
>rq_src_seqno == rt0->rt_seqno) && (WCETT < rt0->rt_etx))
) {

rt0->rt_etx= WCETT;
rt0->rt_interface=temp_iface;
rt0->rt_channel=temp_iface + 1;
```

4.3 Setting up NS-2 for D-WCETT Simulations

4.3.1 Generating the Scenario File

The scenario file or the mobility file defines the movement of nodes throughout the run of simulation. It defines at what location in xy coordinates a node will start and at what time will it travel to which coordinate. The pause time it will take before moving on to another location. All of this is generated randomly depending upon the seed that is given as input. The mobility file generator script is available under ~ns/indep-utils/cmu-scen-gen/setdest and is called setdest.

The following is an example script to generate a mobility file.

```
$ setdest –n75 –p10 –s20 –t900 –x1000 –y1000 > scen-75
```

The following is a description of the parameters:

a) -n is the number of nodes
b) -p is pause time a node will take after reaching a destination
c) -s is the max speed of any node in the simulation
d) -t is the time of the simulation run
e) -x is the x-dimensions of the topography
f) -y is the y-dimensions of the topography

The scenario file generated is stored under filename “scen-75” and will be given as input to an OTCL script.

4.3.2 Generating the Traffic Pattern File

Random traffic connections of the transmission control protocol (TCP) and constant bit rate (CBR) can be setup between mobile nodes using a traffic-scenario generator script. The traffic generator script is available under ~ns/indep-utils/cmu-scen-gen and is called cbrgen.tcl. It can be used to create CBR and TCP traffics connections between mobile nodes. An example script can be executed as follows:

```
$ ns cbrgen.tcl –type cbr –nn 75 –seed 1.0 –mc 30 –rate
32.0 > cbr-75-c30r32
```

The following is a description of the parameters:

a) -type either CBR or TCP traffic

b) -nn is the number of node(s) to be simulated

c) -seed is the seed to the random number generator

d) -mc is the maximum number of connections (pair-wise)

e) -rate is the rate at which one source generates traffic in packets/second

The traffic file generated is stored under filename “cbr-75-c30r32”.

4.3.3 Configuring the TCL file

4.3.3.1 Defining the Constants

The first step in the simulation is to define the wireless physical medium parameters and initialize the simulation. After setting up the initial parameters, the simulator objects are created.

```
# ======================================================================
# Define options
# ======================================================================
set val(chan)           Channel/WirelessChannel ; #Channel in use
set val(prop)           Propagation/TwoRayGround; #Propagation Model
set val(netif)          Phy/WirelessPhy/Wireless_802_11_Phy; #Network Interface
set val(mac)            Mac/802_11
set val(ifq)            Queue/DropTail/PriQueue; #Queue/DropTail/PriQueue
set val(ll)             LL
set val(ant)            Antenna/OmniAntenna ; # Omnidirectional Antenna
set val(em)             EnergyModel
set val(ie)             1000    ; # initial energy of the nodes
set opt(tr)             "/dev/stdout" ; #trace file

set val(x)              1000    ;# X dimension of the topography
set val(y)              1000    ;# Y dimension of the topography
set val(ifqlen)         50      ;# max packet in ifq
set val(seed)           5.0     ;# random number generator
set val(adhocRouting)   AODV    ;# Routing protocol used

set val(mc)             50      ;# total number of mesh clients
set val(mr)             25      ;# total number of mesh routers
set val(nimc)            1      ;# total number of interfaces in mesh clients
set val(nimr)            6      ;# total number of interfaces in mesh routers
set val(nn)             [expr $val(mc) + $val(mr)]     ;# total number of nodes

set val(cp)             "cbr-test-75-hover" ;# connection pattern file
set val(sc)             "scen-n75-seed5-p10-s10-t900-x1000-y1000";# scenario file
set val(stop)           900         ;#  simulation time
Agent/AODV set IFQ_WIN_SZ 0.1       ;#  the windows size of interface queue
```

4.3.3.2 Defining Node Properties

The following code is used to configure the mobile node with all the given values of the ad-hoc routing protocol, network stack, channel, topography, propagation model, with wired routing turned on or off (required for wired-cum-wireless scenarios) and tracing turned on or off at different levels (router, MAC, agent).

```
#global node setting

$ns_ node-config -adhocRouting $val(adhocRouting) \
         -llType $val(ll) \          ;# specifies link layer object
         -macType $val(mac) \        ;# specifies mac object
         -ifqType $val(ifq) \        ;# specifies ifq object
         -ifqLen $val(ifqlen) \      ;# specifies length of ifq
         -antType $val(ant) \        ;# specifies antenna object
         -propType $val(prop) \      ;# propagation object
         -phyType $val(netif) \      ;# specifies physical layer object
         -channel [new $val(chan)] \ ;# specifies channel object
         -topoInstance $topo \       ;# specifies topography
         -agentTrace ON \            ;# for wired cum wireless simulations
         -routerTrace ON \           ;# specifies agent level trace ON/OFF
         -macTrace OFF \             ;# specifies router level trace ON/OFF
         -energyModel  $val(em) \    ;# specifies mac level trace ON/OFF
         -initialEnergy $val(ie) \   ;# specifies initial energy of node
         -numif  $val(ni) \
         -rxPower          (in W)
         -txPower          (in W)
```

4.3.3.3 Initializing the Simulation

A flat topology is created by specifying the length and width of the topography, where val(x) and val(y) are the boundaries used in simulation. The load_flatgrid object is used to specify a 2-D terrain. Open $opt (tr) gives the value whether a trace file (file.tr) is to be generated or the results are not to be written on the hard disk and instead displayed on the screen (/dev/stdout).

```
# ============================================================
# Main Program
# ============================================================

#
# Initialize Global Variables
#

# create simulator instance
set ns_   [new Simulator]

# setup topography object
set topo  [new Topography]

# create trace object for ns and nam
set tracefd [open $opt(tr) w]
#set namtrace    [open trace.nam w]

#$ns_ use-newtrace
$ns_ trace-all $tracefd
#$ns_ namtrace-all-wireless $namtrace $val(x) $val(y)

# define topology
$topo load_flatgrid $val(x) $val(y)

#
# Create God
#
set god_ [create-god $val(nn)]
#
```

The General Operations Director or GOD is a NS-2 simulator object, which is used to store global information about the state of the environment, network, or nodes that an omniscient observer would have, but that should not be made known to any participant in the simulation.

The MobileNode is designed to move in a three dimensional topology. For simplicity, it is assumed that the MobileNode will always move on a flat terrain with Z- coordinate always set to zero. Thus the MobileNode has X, Y, Z co-ordinates that are continually adjusted as the node moves.

4.3.3.4 Finishing Up and Running the Simulation

```
#
# Tell nodes when the simulation ends
#
for {set i 0} {$i < $val(nn) } {incr i} {
    $ns_ at $val(stop).0 "$node_($i) reset";
}
#exec nam trace.nam

for {set i $val(mc)} {$i < [expr $val(mc) + $val(mr)]} {incr i} { #mis0
$ns_ at 0.0 "$node_($i) add-mark m1 red circle"
}

$ns_ at  $val(stop) "finish"

# close traces at the end
proc finish {} {
  global ns_ tracefd # namtrace
  $ns_ flush-trace
  close $tracefd
#  close $namtrace
  puts "NS EXITING..."
  $ns_ halt
#  exit 0
}

puts "Starting Simulation..."
$ns_ run
```

4.3.4 Output Trace File Format

During a simulation a trace file is generated. It serves as a log file for the events that occurred in the simulation. Each field that succeeds the first character is separated by an empty space.

An example of a trace file line is shown below.

```
s 20.666434826 _18_ RTR  --- 1551 cbr 532 [0 0 0 0] [energy
1000.000000] ------- [18:1 19:0 30 54]
```

The above trace file line means that a cbr packet of size 512 bytes was sent from the network layer of node 18 at time 20.66 seconds approx to node 19 with TTL (time to live) value of 30 hops and the next hop is node 54.

Here is the full description of trace format:

(a) ACTION: [s | r | D]: s -- sent, r -- received, D -- dropped

(b) WHEN: the time when the action happened

(c) WHERE: the node where the action happened

(d) LAYER:

  (i) AGT – application layer

  (ii) RTR – routing or network layer

  (iii) LL -- link layer

  (iv) IFQ -- outgoing packet queue (between link and MAC layer)

  (v) MAC – Media Access Layer

  (vi) PHY – physical layer

(e) Flags:

(f) SEQNO: the sequence number of the packet

(g) TYPE: the packet type

  (i) cbr -- CBR data stream packet

  (ii) DSR -- DSR routing packet (control packet generated by routing)

  (iii) AODV – AODV route request, reply or error packet

(h) SIZE: the size of packet at current layer, when packet goes down, size increases, goes up size decreases

(i) [a b c d]: a -- the packet duration in MAC layer header

  b -- MAC address of destination

  c -- MAC address of source

  d -- MAC type of the packet body

(j) Flags:

(k) [.....]:

  (i) source node IP : port number

  (ii) destination node IP (-1 means broadcast) : port number

  (iii) IP header TTL

  (iv) IP of next hop (0 means node 0 or broadcast)

4.3.5 Extracting Information from Trace File

Generation of a trace file is a lengthy process. It involves time as well as disk space. Since all the contents are written on the disk, a 900 sec simulation of 75 nodes can take time of up to an hour and a disk space of 2 GB. To save time as well as space, a different method

has been adopted in this thesis. Instead of generating a trace file at the output, the contents of the trace file are displayed on the screen (using /dev/stdout in the opt(tr) variable in tcl code). Separately a C++ code was written which reads the contents of the output of the simulation (the contents of trace file but not written on the disk) and obtains the necessary information from it. The output of the simulation was given as the input to the C++ file through a process known as "pipe". It implies the physical meaning of pipe as it outputs the contents of one file as the input to another file, just like a real world pipe is used for. The syntax of the command is given as follows.

```
$ ns temp-nc-sp5sd1.tcl | ./result ->> sp5sd1.txt
```

Here the temp.tcl is the simulation file, result is the compiled code of the C++ file whereas the results are stored in a txt file names sd5sd1 (which actually means that the results are for speed 5 and seed 1). The text file is shown in the figure below.

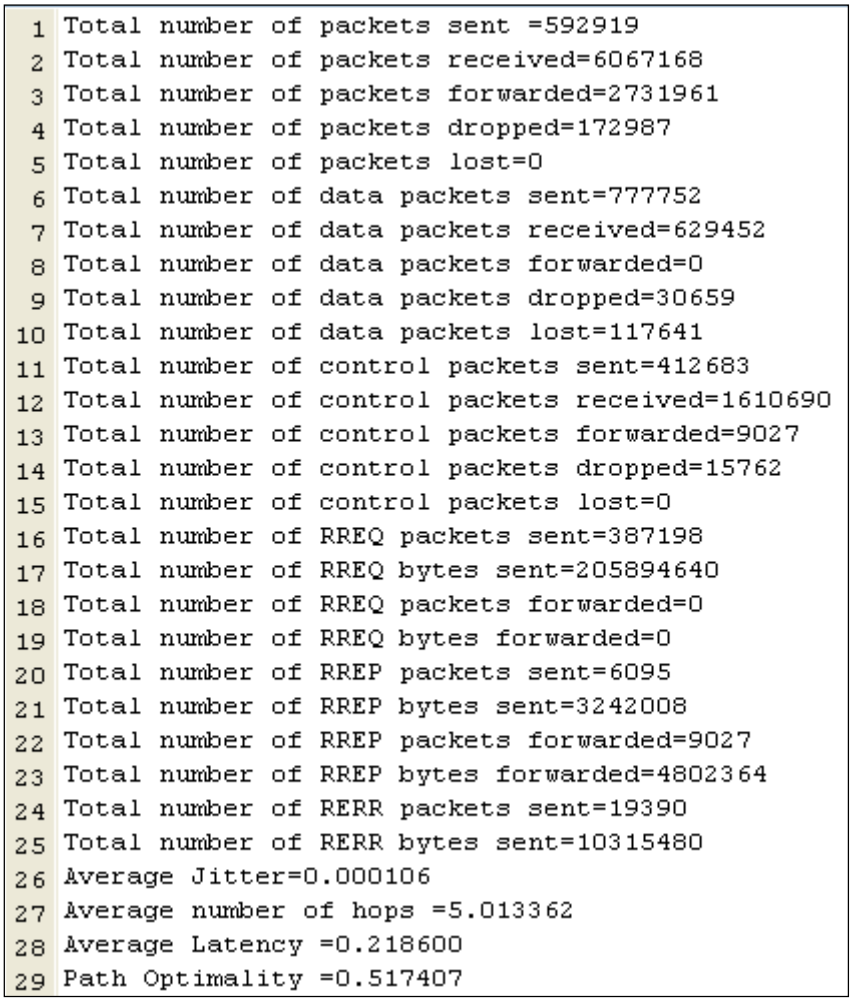


Figure 6 Contents of the output text file

4.3.6 Automating Simulations via Shell Script

For evaluating any performance parameter, a number of simulations have to be run in order to obtain the most accurate results possible. A particular simulation has to be run over at least 5 times to get accurate results. It would be quite tiresome to actually wait for a simulation to end and then to begin the next one. To automate the running of simulations a shell script was written which is nothing but a set of commands that would be executed in a sequence without the need of any intervention. This not only saves time but also a lot of human effort.

To create a shell script, write all the commands in a sequence that one would execute as if doing one by one. Include the first line as shown in the figure and save it with .sh extension. This would now be an executable file which run once will run all the commands included in it according to the sequence contained within.

```
#! /bin/sh
chmod -R 777 .
cd tests/sd1/ncsd1sp0
ns tempnc-sd1sp0.tcl | ./result ->> ncsd1sp0.txt
cd ..
cd ncsd1sp5
ns tempnc-sd1sp5.tcl | ./result ->> ncsd1sp5.txt
cd ..
cd ncsd1sp10
ns tempnc-sd1sp10.tcl | ./result ->> ncsd1sp10.txt
cd ..
cd ncsd1sp15
ns tempnc-sd1sp15.tcl | ./result ->> ncsd1sp15.txt
cd ..
cd ncsd1sp20
ns tempnc-sd1sp20.tcl | ./result ->> ncsd1sp20.txt
cd ..
cd ..
```

4.3.7 Generating Graphs using MATLAB

Once a series of simulation have been run with different seeds and the results are gathered in a txt file, the next step comes out to visualize the results using graphs. Different graph plotting tools are available namely XGraph and TraceGraph but require some proficiency to operate. A simpler method has been used in this thesis which makes use of MATLAB and MS Excel.

The contents of the txt files (simulation results) are copied on to an excel sheet where for different set of tests belonging to the same category are averaged. A MATLAB code was written which reads the excel files, computes the necessary parameters (e.g. packet delivery ratio, goodput etc.) and draws the graphs according to scale. Some excerpts of the code are given below.

```
% FOR SPEED 0

a=xlsread('a0.xlsx');
m=xlsread('m0.xlsx');
n=xlsread('n0.xlsx');
w=xlsread('w0.xlsx');
z=1;
.
.
dpdr_a(1,z)=a(7,1)/a(6,1);
rpktoverhead_a(1,z)=a(11,1)/a(7,1);
latency_a(1,z)=a(28,1);
gdput_a(1,z)=(512*a(7,1))/900;
jitter_a(1,z)=a(26,1);
hops_a(1,z)=a(27,1);
.
.
%Packet delievery ratio
figure(1)
subplot(2,2,1)
bar(speed,100*V,1,'group')
legend('B=0.1','B=0.5','B=0.9','Dynamic B')
xlim([-2 22])
ylim([0 100])
```

Chapter 5

Simulation Results and Analysis

The simulations carried out to assess the performance of the proposed routing metric are presented along with a detailed analysis of each result.

5.1 Simulation Environment

The performance of D-WCETT has been evaluated through extensive simulations in NS-2 [11]. AODV [12] has been selected as the routing protocol owing to its suitability and applicability to Hybrid WMNs. Since WCETT metric was developed for link state routing protocol, a few adaptations have been made to AODV for WCETT to work on it [13] which are carried over to D-WCETT as well. The simulated network topology consists of 25 static Mesh Routers arranged in a 5x5 grid in a dense WMN covering an area of 1 square km. Moreover 50 mobile Mesh Clients, each equipped with a single radio, are placed randomly in the simulation area. NS-2 simulations were configured such that Mesh Routers were equipped with six 802.11b radios tuned to orthogonal channels, hence additionally simulating a scenario where 802.11a radios can be used which offer more non-overlapping channels than 802.11b. Concurrent UDP flows were established between randomly selected source and destination Mesh Client pairs. After experimenting with different values, we have found that IFQ Window Size of 0.1 seconds is best suited to our simulated network. The default simulation parameters are listed in Table 1. As we wanted to evaluate the behaviour of Dynamic β, it was pertinent to test it against different static β values. Hence three tests were conducted to evaluate the performance of D-WCETT routing metric in AODV protocol modified with multi-radio support and results were compared against WCETT (with static β values 0.1, 0.5 and 0.9) also implemented in AODV.

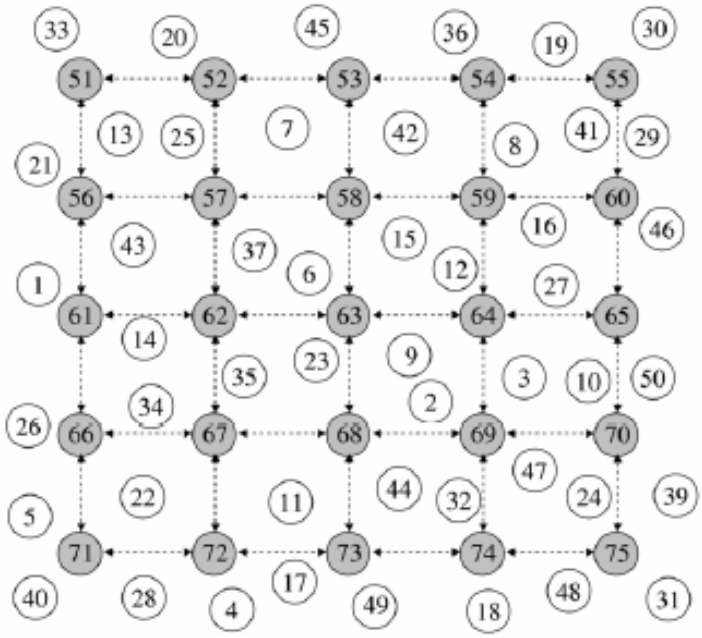


Figure 7 Simulated 75 node Topology

Table 1 Simulation Parameters

Examined Metrics	WCETT (with β 0.1, 0.5 and 0.9) and D-WCETT (Dynamic β)
Simulation Area	1000 x 1000 m
Simulation Time	900 seconds
Propagation Model	Two-Ray Ground Reflection
Mobility Model for Mesh Clients	Random Waypoint
Number of Mesh Routers	25
Number of Mesh Clients	50
Number of 802.11b radios in Mesh Client	1
Number of 802.11b radios in Mesh Router	6
Transmission Range	250 m
Traffic Type	CBR (UDP)
Packet Size	512 bytes
Packet Transmission Rate	32 pkts/sec
Number of Sources (Flows)	30
Max Number of Packets in Queue (IFQ length)	50 packets
IFQ Window Size	0.1 seconds

The following four tests were conducted to evaluate the performance of the D-WCETT routing metric.

(a) Test 1: Varying the Mesh Client speeds. To study the effect of mobility of nodes on the performance of the routing metrics, the maximum speed of Mesh Clients was varied from 0 m/s to 15 m/s in steps of 5 m/s.

(b) Test 2: Varying the Packet Transmission Rate. In order to study how data sending rate affects the performance of routing metrics, the packet transmission rate of source nodes was varied from 8 pkts/sec to 64 pkts/sec.

(c) Test 3: Varying the Number of Flows. To study the performance of routing metrics at different network load levels, the number of concurrent UDP flows was varied from 10 flows to 50 flows in steps of 10.

(d) Test 4: Varying the Number of Mesh Routers. The Mesh Routers play an important role in Hybrid WMNs and to study their effects the number of Mesh Routers in the simulation were varied from 5 MRs to 25 MRs in steps of 5.

5.2 Performance Metrics

The following three performance metrics were considered during the testing of D-WCETT:

a) **Packet Delivery Ratio**: The ratio of total number of data packets successfully received and the total number of data packets transmitted.

b) **Routing Packet Overhead**: The ratio of control packets generated and number of successfully received data packets.

c) **Latency**: The mean time (in seconds) taken by data packets to reach their respective destinations

d) **Aggregate Goodput**: The total number of application layer data bits successfully transmitted in the network per second.

5.3 Results and Analysis

5.3.1 Test 1: Varying the Maximum Mesh Client Speed

In Test 1, we have studied the effect of mobility on the routing performance and in doing that we also evaluated the metrics in a Static vs. Highly Mobile Network scenario as well. The maximum speed of the mesh clients was varied from 0 m/s to 15 m/s and thirty 128kbps CBR flows were maintained between random mesh client pairs. Five sets of simulations (with same connection patterns but different mobility patterns) were run each for WCETT (with β values

of 0.1, 0.5 and 0.9) and D-WCETT (with Dynamic β) and performance parameters were obtained.

D-WCETT achieves a higher Packet Delivery Ratio (PDR) at all speeds as compared to all β values of WCETT. At mesh client speeds of 0 m/s, WCETT (β 0.1) achieves a 93% Packet Delivery Ratio (PDR), WCETT (β 0.5) 93%, WCETT (β 0.9) 91% while D-WCETT with its Dynamic B achieves a PDR of 98%. The PDR starts to decline as the mesh client speed is increased. As the speed of clients increase, the nodes leave their mutual communication ranges more frequently, resulting in frequent route breakages. The improvement of D-WCETT over WCETT becomes more evident as the mobility of the network increases. At a client speed of 15 m/s WCETT with different β values is around 71% whereas the PDR of D-WCETT is at a remarkably high 88%. This improvement of D-WCETT is owing to its ability to change the value of β according to network load. If the network is facing congestion at a particular node, D-WCETT sets a lower value of β, which favours shorter paths, resulting in higher PDR. But at the same time areas within the network which are least congested and for whom channel diverse paths would yield optimum results are assigned higher values of β by the D- WCETT metric.

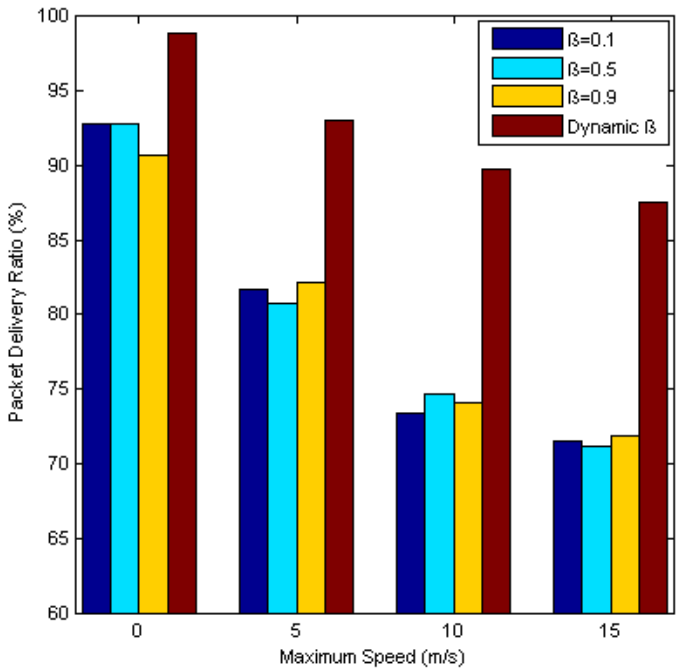


Figure 8 Packet Delivery Ratio results for varying the Mesh Clients' speed

The routing packet overhead is an indication of how many control packets are generated to

deliver a data packet. At 0 m/s client speed, routing packet overhead registers a value of 0.3 and is the same for WCETT (with different β values) and D-WCETT meaning that 3 control packets are sent on an average for 10 data packets. As expected, with the increase in mesh clients' speed, the routing packet overhead increases and more effort is needed to maintain connectivity within the network. At mesh client speed of 5 m/s, the packet overhead of WCETT with all β values has a value of 0.7 whereas for D-WCETT it is 1.1 control packets for every data packet. Finally at mesh client speed of 15 m/s, the packet overhead of WCETT with all β values has a value of 1.1 whereas for D-WCETT it is 1.8. The results indicate that D-WCETT incurs a larger routing overhead as compared to all β values with WCETT. The reason behind the increased overhead in D-WCETT's case is explained owing to D-WCETT's behaviour of forwarding subsequent route request (RREQ) packets having lower (and better) WCETT values. At Mesh Routers with multiple network interfaces tuned to orthogonal channels, at a particular instant a node may discard a RREQ on a particular channel if another channel is found to be less congested. But the same channel may be more favourable after some time and hence a subsequent RREQ would be allowed to propagate through that channel. Hence the number of control packets generated for each data packet would tend to increase.

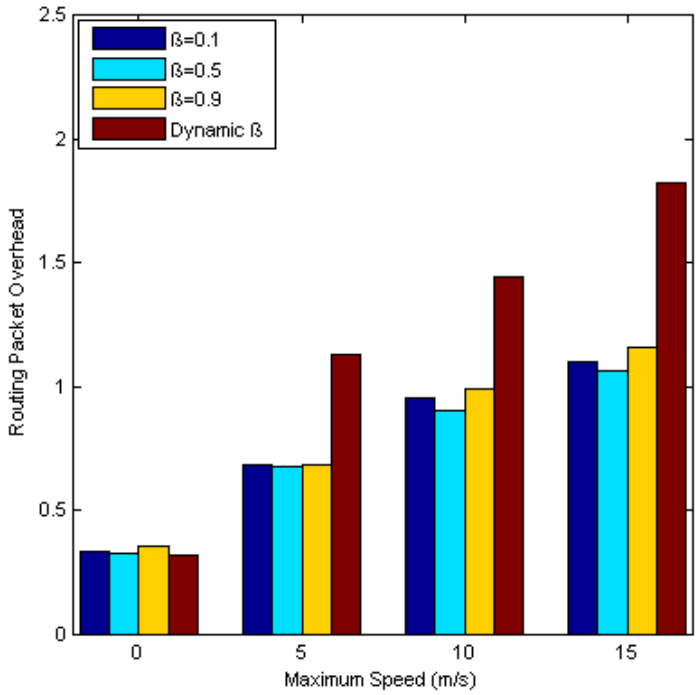


Figure 9 Routing overhead result for varying the Mesh Clients' speed

The most notable improvement of D-WCETT over the three values of β in WCETT is in terms

of the average network latency. D-WCETT considerably outperforms all values of β at all mesh client speeds. At mesh client speed of 0 m/s, DWCETT has an average network latency of 0.01 seconds as compared to an average value of 0.1 seconds for WCETT with different β values. As the mesh client speed increases, the latency values also increase as it takes more time for packets to reach their destinations owing to increased routing disruptions. At mesh client speeds of 5, 10 and 15 m/s the time taken by data packets to reach their destinations in D-WCETT is half the time taken by packets in WCETT (all β values) under similar circumstances. This is the most direct result of choosing least congested links in D-WCETT. In D-WCETT paths are created having least number of packets in interface queues. Thus after the creation of a path from source to destination, data packets traverse through the network in less time as they are navigated through least filled queues. It should also be noted that although D-WCETT incurs an additional computational overhead of measuring QDI and integrating it with β for calculation of D-WCETT, it may appear that it will take more additional time for packets to reach their destinations. However this computational disadvantage is offset through selection of least congested links which substantially reduces the time taken by packets to traverse through the network.

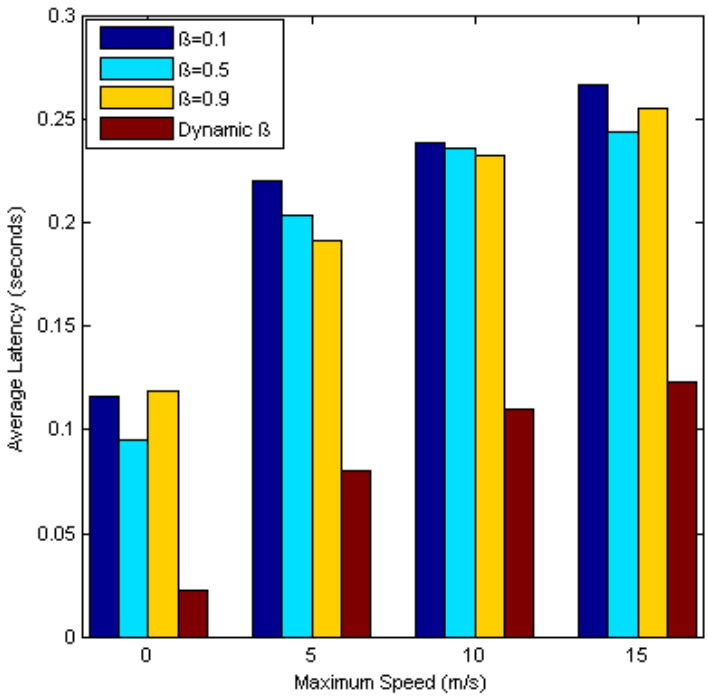


Figure 10 Latency result for varying the Mesh Clients' speed

As mentioned in previous results, PDR and Latency improves for Dynamic β whereas Routing

Overhead increases. Although D-WCETT with its Dynamic β delivers packet more efficiently and in less time but at the cost of increased network overhead. Hence it is important to have a look at the Goodput results of the routing metrics under comparison as it shall provide a good overall comparison of Dynamic β against static values of β. At an average Transmission Rate of 3.53 Mbps (Number of data packets sent on an average, as the same Connection File was used in all simulations), Dynamic β returns a Goodput result ranging from 3.50 Mbps to 3.1 Mbps with increasing mesh client speeds whereas Static β returns lesser Goodput values. At mesh client speed of 15 m/s, D-WCETT has a Goodput value of 3.1 Mbps as against a value of 2.6 Mbps for Static β of WCETT. D-WCETT maintains better Goodput values at all mesh client speeds than WCETT. The graph depicting Goodput is similar to the Packet Delivery Ratio graph which confirms the assessment that having a dynamically changing value of β with the network load produces better results than assigning a constant value of β to be used during the whole network operation.

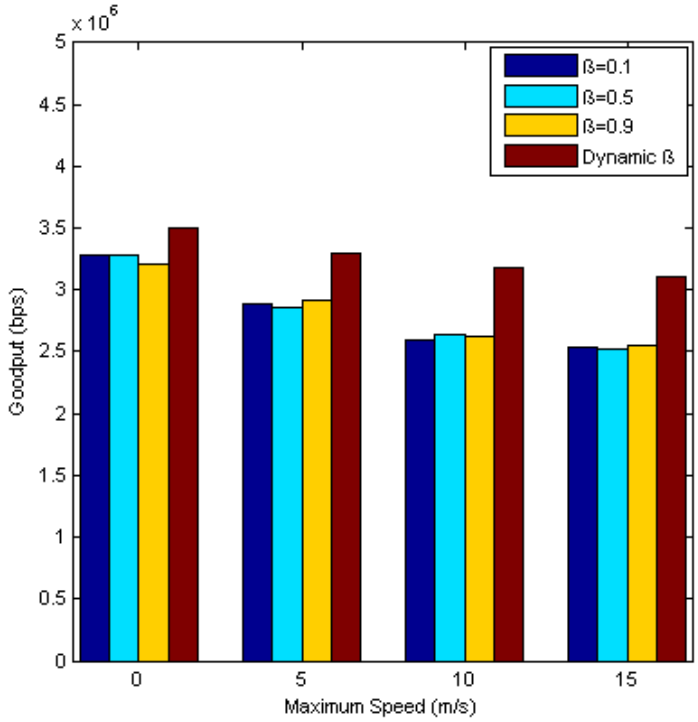


Figure 11 Goodput result for varying the Mesh Clients' speed

5.3.2 Test 2: Varying the Packet Transmission Rate:

In Test 2, the packet transmission rate was varied from 8 pkts/sec to 64 pkts/sec, and thirty CBR flows were maintained between random mesh client pairs moving at a speed of 15 m/s.

Keeping in view the results of Test 1, a high mobility scenario (with speed of 15 m/s) was selected in order to ascertain performance in a challenging scenario. The corresponding performance metrics for WCETT (with β values of 0.1, 0.5 and 0.9) and D-WCETT (with Dynamic β) are discussed in the following sections.

As shown in Figure 12 PDR result for varying the packet transmission rate, D-WCETT with its Dynamic β achieves better Packet Delivery Ratio (PDR) compared to all static β values at all packet transmission rates. At a lightly loaded network with Packet Transmission Rate of 8 pkts/sec, Dynamic β achieves a PDR of almost 100% whereas Static β values achieve PDR of 92%. The same result is seen for Transmission Rate of 16 pkts/sec. As the transmission rate is increased, the network is being loaded with more traffic and a drop in PDR values is expected. Here Dynamic β becomes comparable with the Static β metrics. The improvement of D-WCETT over WCETT with Static β values is due to the former's selection of least congested links in formation of a path. Whenever a path is to be formed between Source and Destination, the decision at multi-radio Mesh Routers is taken keeping in view the link with least number of packets in queue. Hence all channels of a Mesh Router are equally and judiciously utilised, leading to a higher Packet Delivery Ratio. At transmission rates of 64 pkts/sec (256kbps per flow) all links become equally congested, eliminating the possibility of any improvement.

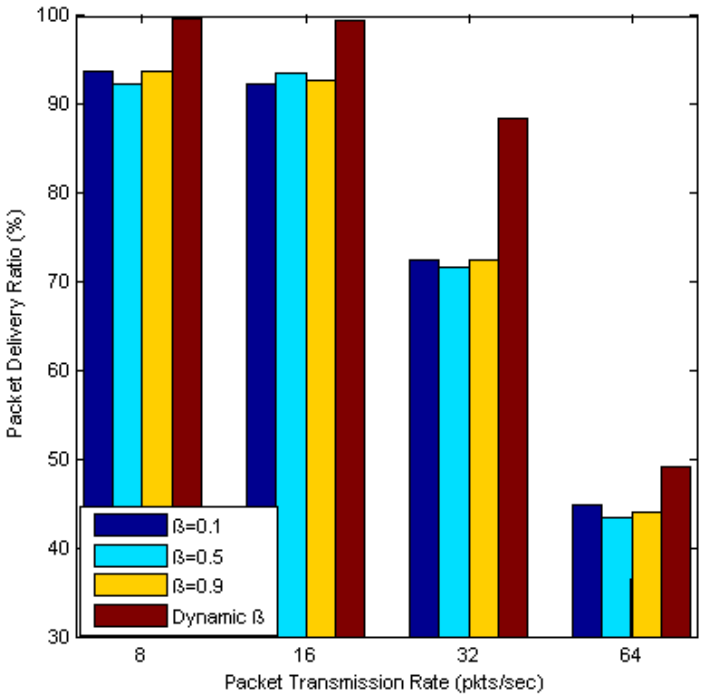


Figure 12 PDR result for varying the packet transmission rate

The routing overhead of WCETT (with β 0.1, 0.5 and 0.9) and D-WCETT (Dynamic β) is shown in Figure 13. We observe that the routing overhead of D-WCETT remains slightly lower than WCETT at 8 and 16 pkts/sec but is higher at 32pkts/sec and registers a significant increase at 64 pkts/sec. For the 75 node simulated network, we see that 30 flows of 8 pkts/sec (32kbps per flow) constitute a lightly loaded network, and even at a high mobility of 15 m/s the packet overhead remains moderate around 1 control packet per data packet. The same scenario is observed at packet transmission rate of 16 pkts/sec. However when the packet transmission rate is increased to 32 pkts/sec (30 flows of 128kbps/flow), the simulated network becomes moderately loaded. Coupled with high mobility, it generates a lot more control packets in the case of D-WCETT. Here we observe a disadvantage of Dynamic β; it generates more control packets in order to maintain connectivity between nodes. This behaviour is owing to DWCETT's forwarding of subsequently better RREQ packets at Mesh Routers. When a RREQ reaches a Mesh Router, which has multiple channels, the D-WCETT metric is computed for all links based upon their QDI value and RREQ is forwarded on the best link. However at a subsequent instance when the queue lengths have changed, a RREQ arriving at the same node may be forwarded on the currently best link, which may be different than the previous. This behaviour, combined with such high packet sending rates, leads to a storm of control packets.

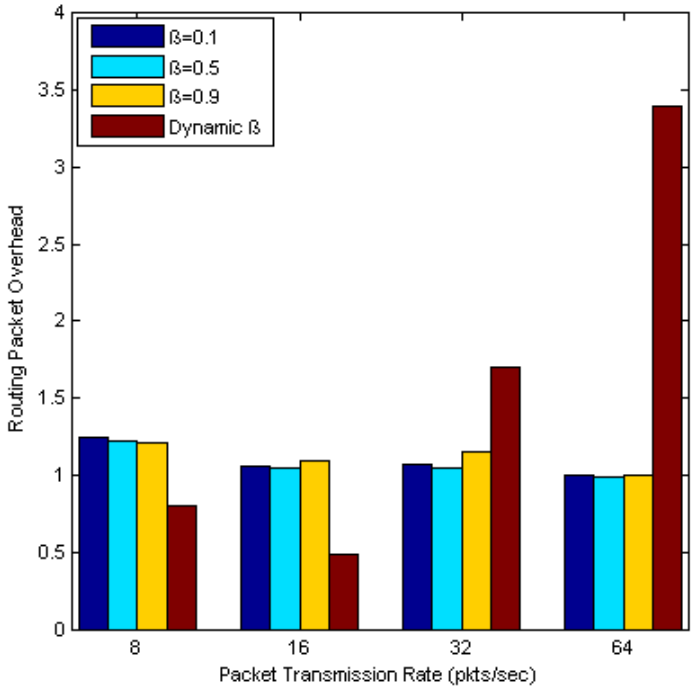


Figure 13 Routing overhead result for varying the packet transmission rate

Latency results for WCETT (with Static β values) and D-WCETT (Dynamic β) with increasing packet transmission rates of 8, 16, 32 and 64 pkts/sec are shown in Figure 14. Results show that at very low rates of 8 pkts/sec, both metrics take the same amount of time to deliver packets to their destinations. As the transmission rate is increased to 16 pkts/sec, the time taken by Static β increases whereas Dynamic β remains the same. However 8 and 16 pkts/sec constitute a fairly low network activity. At 32 pkts/sec, Static β metrics have a latency value of approx. 0.25 seconds whereas Dynamic β gives a result of 0.1 seconds. At a high transmission rate of 64 pkts/sec, we see a significant increase in Latency results as more time is taken by packets to reach their destinations. This is both due to increased network load as well as frequent route disruptions due to high mobility of nodes (at a speed of 15m/s). It is observed that time taken by packets for D-WCETT is less but only by a slight margin. It would be fair to say that D-WCETT performs only marginally better in terms of latency at higher transmission rates. From the figure it is clear that D-WCETT performs better than all static values of β at all transmission rates. The delay experienced by packets in D WCETT metric is significantly low till 32 pkts/sec but becomes comparable at higher transmission rates. In D-WCETT paths are created having the least number of packets in interface queues and hence packets take less time to reach their destinations.

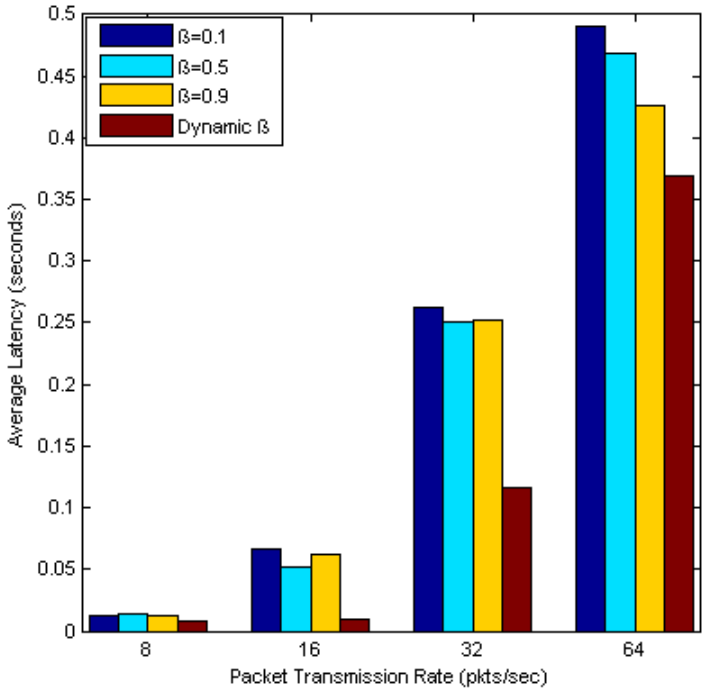


Figure 14 Latency result for varying the packet transmission rate

Goodput results for varying packet transmission rate are shown in Figure 15. It can be seen from the results that at lower transmission rates of 8 and 16 pkts/sec, all Static β metrics perform almost equally with Dynamic β achieving a slightly better Goodput. Increasing the transmission rate to 32 pkts/sec yields a better and clearer picture of the performance of the metrics under evaluation. We observe that all Static β values of WCETT perform equally well and result in approximately same Goodput. However D-WCETT achieves better Goodput by a margin of 0.5 Mbps. The difference tends to decrease at higher transmission rate of 64 pkts/sec at which we observe that improvement owing to Dynamic β becomes less influential. Thus D-WCETT is able to achieve a constantly better performance than Static β values of WCETT at all packet transmission rates in terms of Packet Delivery Ratio, Latency and Goodput. The performance is most optimum at packet transmission rate of 32 packets/sec. However we see that D-WCETT applies a heavier penalty in terms of increased routing packet overhead.

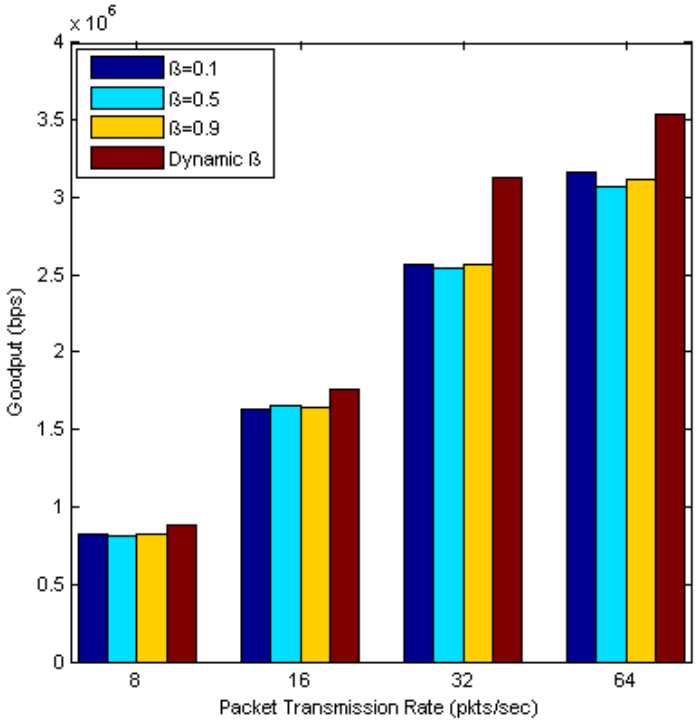


Figure 15 Goodput result for varying the packet transmission rate

5.3.3 Test 3: Varying the Number of Flows:

In Test 3, the simultaneous number of connections (concurrent flows) between nodes was varied from 10 to 50 connections. The packet transmission rate was 32 pkts/sec for each flow. The speed of mesh clients was set to a low mobility value of 1 m/s in order to emphasise on

simultaneous connections and to see more clearly the effect of varying the number of concurrent flows. The corresponding performance metrics for WCETT (with β values of 0.1, 0.5 and 0.9) and D-WCETT (with Dynamic β) are discussed in the following sections.

The Packet Delivery Ratio (PDR) results for varying the number of concurrent flows are shown in Figure 16. For a lightly loaded network with only 10 concurrent flows, all metrics perform equally well. However, as soon as the load is increased beyond 20 flows, the improvement of D-WCETT becomes apparent. At 30 flows, D-WCETT eyes an improvement of around 10% compared to Static β, whereas 40 flows yields the most optimum improvement which comes out to be around 15% better for D-WCETT. With an aggregate network load of more than 6 Mbps (50 flows x 128 Kbps/flow) D-WCETT is still able to maintain a reasonable PDR of almost 85%. Another important point to note is that with increase in number of flows, permanently favouring a lower static value of β does not necessarily yield better performance as the results indicate that all static β values (0.1, 0.5 and 0.9) achieve comparable PDR values. This strengthens our argument that keeping β dynamic yields better results as the network alternates between path length and channel diversity according to changing network conditions.

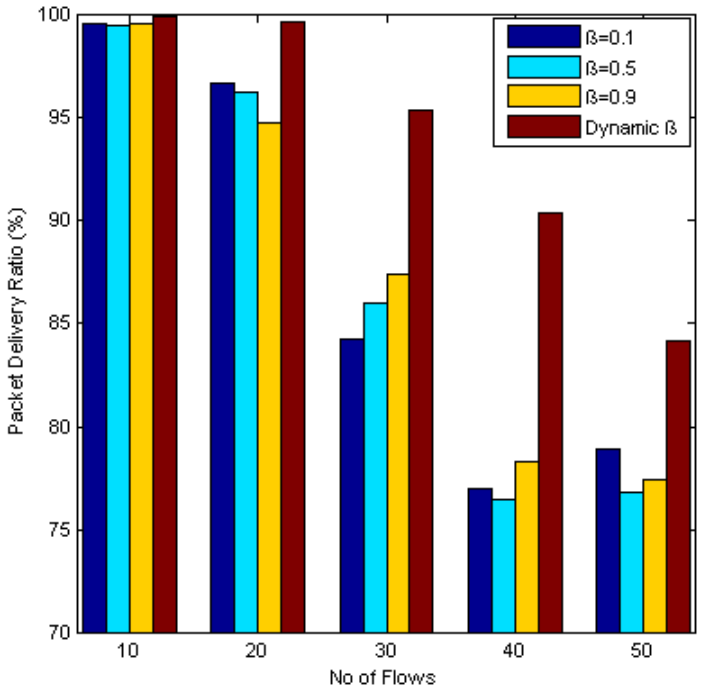


Figure 16 PDR result for varying the Number of Flows

The routing packet overhead results for WCETT with β values of 0.1, 0.5 and 0.9 against

Dynamic β of D-WCETT are shown in Figure 17. It can be seen that 10 and 20 number of flows constitute a lighter load for the simulated 75 node network across a one square kilometre area. The routing overhead of D-WCETT remains slightly lower than WCETT at 10 and 20 flows. It becomes higher at 30 and 40 number of flows and increases significantly at 50 flows. Even at 30 number of flows, the network poses a significant data rate of 3.75 Mbps, resulting in a highly congested network. Nodes drop packets and routes start to become invalid, necessitating new route discoveries leading to generation of more control packets. Dynamic β aggravates the situation further as frequent changes in IFQs lead to a continuously changing metric value, which leads to frequent route changes. Hence D-WCETT results in higher routing overhead with increasing number of flows.

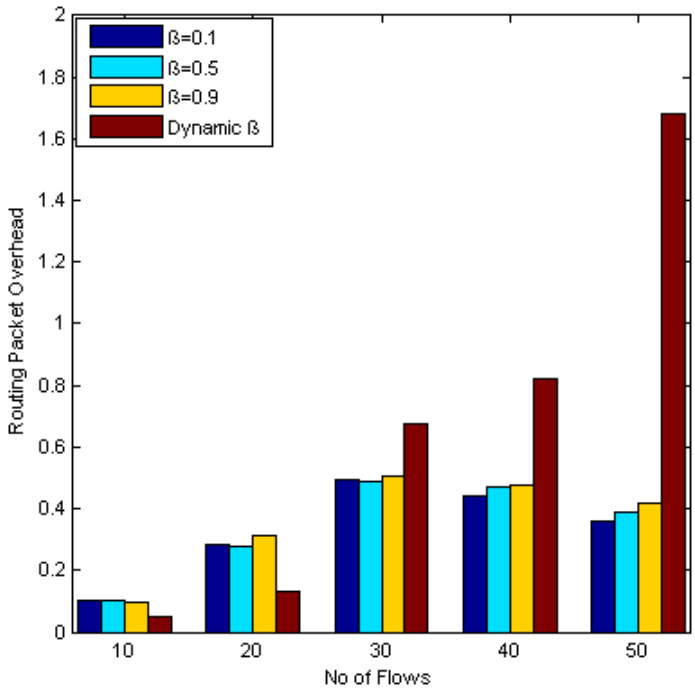


Figure 17 Routing overhead result for varying the Number of Flows

Figure 18 illustrates the Latency results for varying number of flows. At 10 flows, all metrics yield comparable results with Average Network Latency value of 0.01 seconds. At 20 flows, the time taken by packets increases to an average of 0.08 seconds for Static β values of WCETT whereas for D-WCETT it increases to 0.02 seconds. A better comparison is obtained at 30 flows, where the time taken by packets in case of WCETT shoots up to 0.23 seconds, while

Dynamic β maintains a healthy value of 0.05 seconds. The latency values at 40 and 50 number of flows show that D-WCETT maintains the improvement over WCETT, averaging a better time return of almost 0.2 seconds (200msec). Thus selecting a Dynamic β based on current network load information leads to lesser time periods and improved routing efficiency. The paths that are eventually formed by D-WCETT have least congested queues which is reflected in decreased latency values for Dynamic β. Data packets in D-WCETT need half as less time to reach their destination nodes as compared to WCETT at all static β values.

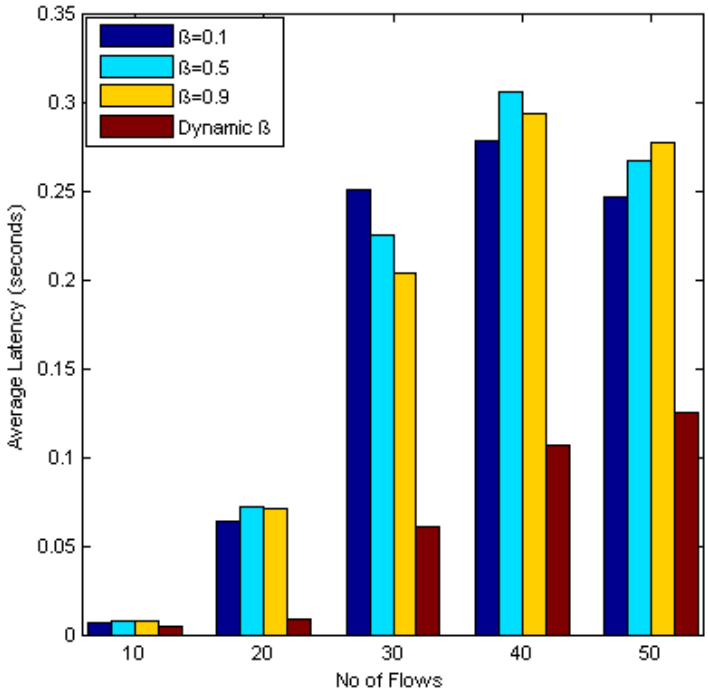


Figure 18 Latency result for varying the Number of Flows

As a measure of overall network performance, Goodput results for varying number of flows are shown in Figure 19. From the Fig. we see that the D-WCETT shows improvement in Goodput as the number of concurrent flows are increased. At 10 number of flows, Static values of β and Dynamic β perform equally well. At 20 number of flows, Dynamic β achieves a slightly better Goodput. The improvement increases further at 30 number of flows and is optimum at 40 number of flows at which Dynamic β gives a 0.5Mbps more Goodput as compared to Static β metrics. For 50 number of flows, the improvement of D-WCETT is there but decreases slightly as the network becomes heavily loaded.

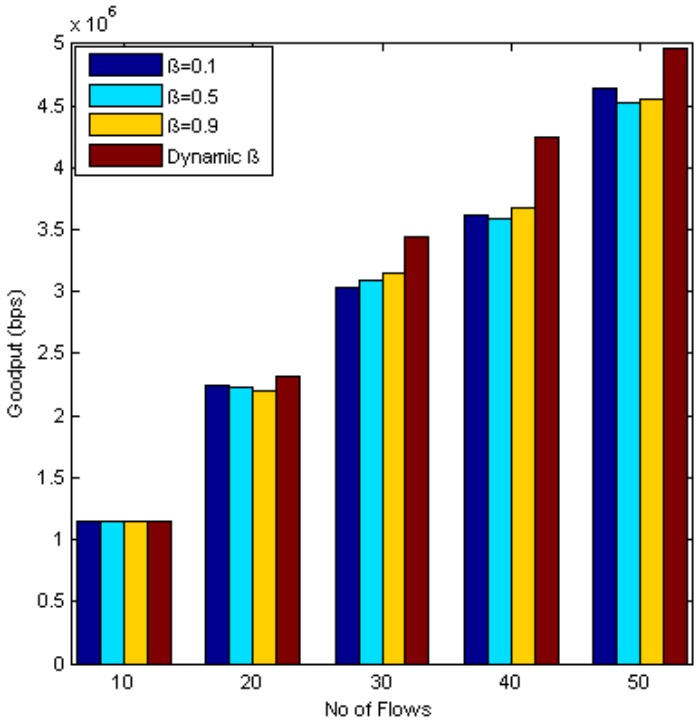
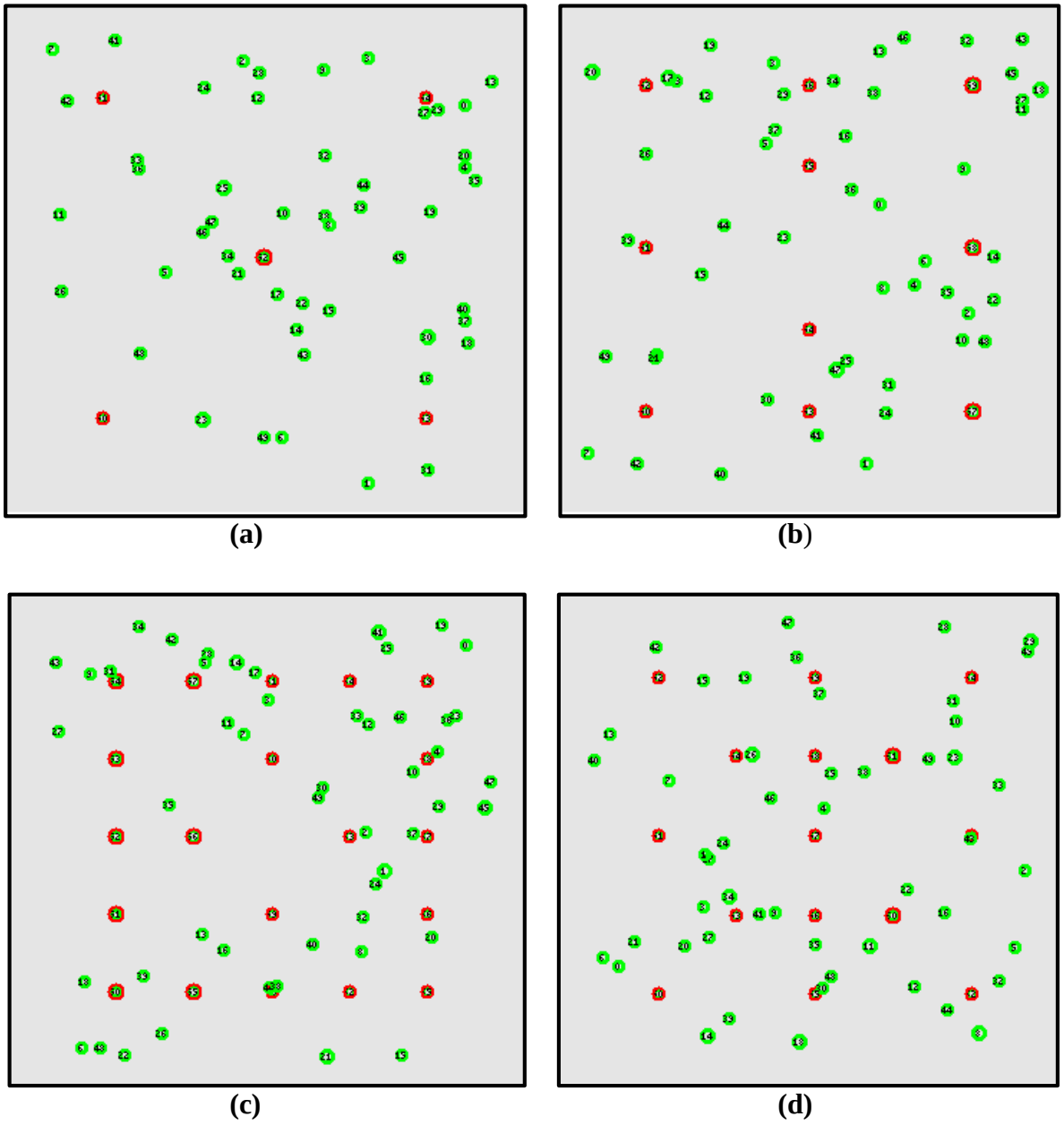


Figure 19 Goodput result for varying the Number of Flows

5.3.4 Test 4: Varying the Number of Mesh Routers:

In Test 4, the number of Mesh Routers (MR) in the simulated network was varied. The objective of the test was to analyse the performance of routing metrics from the perspective of Hybrid WMNs and its applicability. D-WCETT makes judicious use of channels available to a multi-radio Mesh Router by measuring IFQ, calculating QDI and computing D-WCETT metric value for all available channels and forwards the RREQ on the least congested channel. Hence it was felt necessary to study the effect of Mesh Routers on the performance of the metrics under evaluation, specifically D-WCETT. Number of Mesh Routers was varied from 5 to 25 whereas the number of Mesh Clients was constant at 50. The NAM screenshots of simulated topologies of 55 nodes (5 MRs), 60 (10 MRs), 65 (15 MRs), 70 (20 MRs) and 75 nodes (25 MRs) are shown in Figure 20. The speed of Mesh Clients was set to moderate value of 5 m/s and 30 concurrent flows were established between source-destination pairs. The packet transmission rate was 32 pkts/sec for each flow. The corresponding performance metrics for WCETT (with β values of 0.1, 0.5 and 0.9) and D-WCETT (with Dynamic β) are discussed in

the following sections.



(a) **(b)** **(c)** **(d)**

Figure 20 Topologies for Test 4 with varying Mesh Routers (Red Dots) (a) 5 Mesh Routers (b) 10 Mesh Routers (c) 15 Mesh Routers (d) 20 Mesh Routers

PDR values for varying Mesh Routers are shown in Figure 21. D-WCETT achieves a higher Packet Delivery Ratio (PDR) at all test points. We observe that at 5 Mesh Routers the PDR of WCETT metrics (Static β) remain around 30%, 50%, at 10 MRs, 70% at 15 MRs, 75% at 20 MRs and reaches to 80% at 25 MR. Dynamic β is able to achieve a higher PDR at all test points, averaging a roughly 8-10% increase in all scenarios. The results show the importance of multi-radio nodes in a densely populated network. As the number of Mesh Routers is increased, PDR increases occurs. The increase in PDR is exponential from 5 to 20 MRs after which it starts to

become linear. As the MR density increases, the result start to become positive. The results indicate that for the simulated network topology, having 20 MRs is an optimum value. Also the improvement of D-WCETT is owing to its ability to change the value of β according to network load.

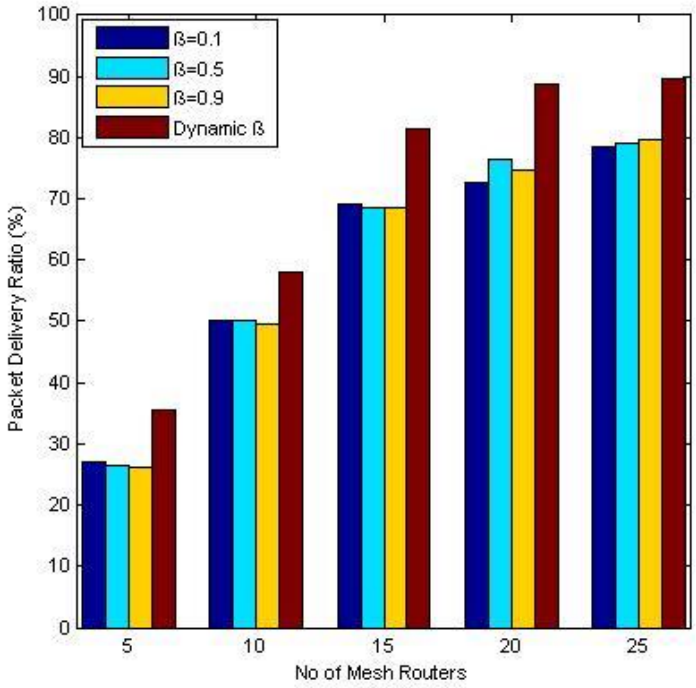


Figure 21 PDR result for varying the Number of Mesh Routers

Routing overhead results for WCETT and D-WCETT with varying Mesh Routers are shown in Figure 22. Routing overhead is significantly high for all metrics at less number of MRs and steadily improves as the MR density increases. This is due to the fact that lack of required MR density affects routing in a severe manner. Due to node mobility, areas within the network become isolated at particular instances more frequently, resulting in route disruptions. This in turn increases the control packets generated to maintain network connectivity. As the MR density increases, this phenomenon happens less often and routing efficiency improves. The results indicate that Dynamic β does not fare well against Static values of β. Dynamic β incurs significantly higher Packet Overhead as compared to Static β at all test points. The negative effect of route disruptions due to node isolation has a multiplied effect as in such cases varying queue lengths within the multiple channels of a node leads to greater number of control packets sent per channel. Hence the increased control packet generations negatively influences the routing efficiency for D-WCETT.

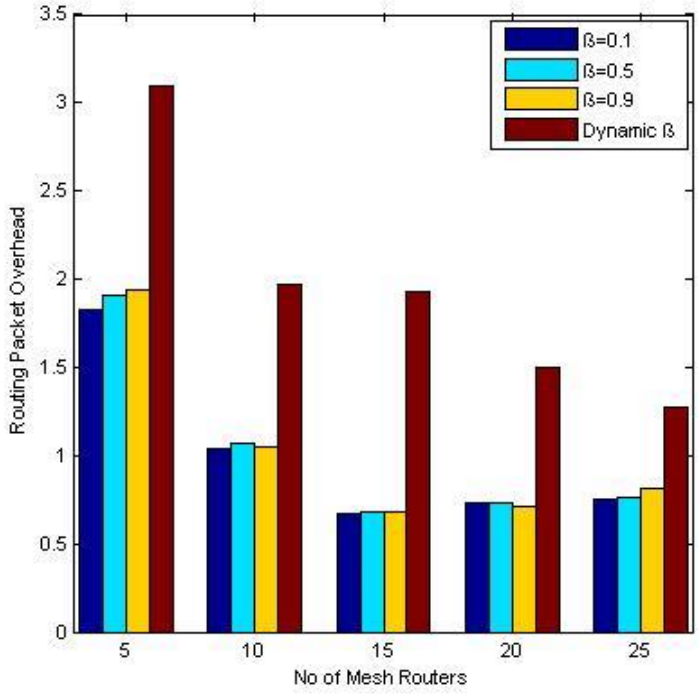


Figure 22 Routing overhead result for varying the Number of Mesh Routers

Latency results for WCETT and D-WCETT with varying Mesh Routers are shown in Figure 23. At lesser number of MRs, the time required for data packets to reach their destinations is quite high. As the MR density increases, the latency results improve. For Static β values, 5 MRs return an average time of 0.8 seconds whereas this value decreases to 0.2 seconds at 25 MR. For Dynamic β the latency values are better, almost half of their Static β counterparts. Results of Test 4 indicate the importance of Mesh Routers in a Hybrid WMN. Mesh Routers form the backbone of the network and greatly influence the routing efficiency. As MR density increases, there are less route breakages which improves PDR, packet overhead as well as Latency. Moreover from Figure 23 we see that Dynamic β gives much better Latency results as compared to Static β values. The main objective of Dynamic β is to select links with least level of congestion and this has a direct bearing on the time spent by packets in queues and hence time taken by packets to reach their destinations.

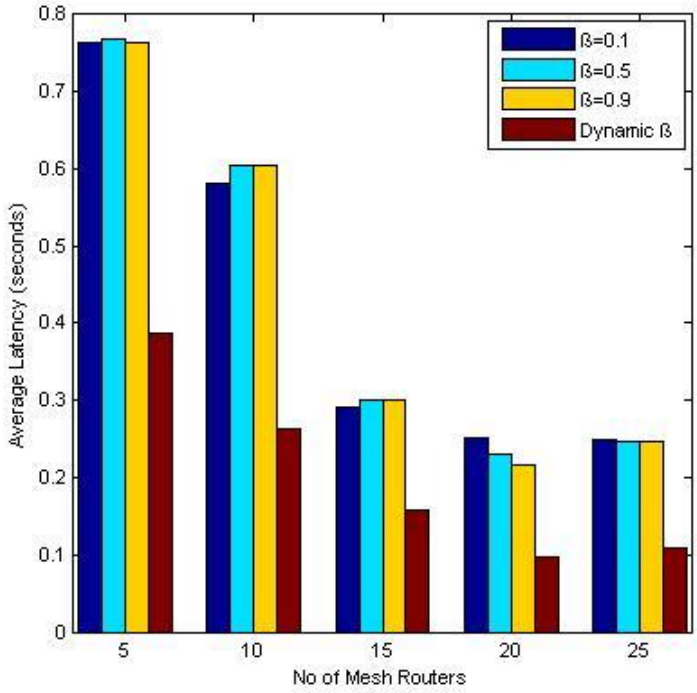


Figure 23 Latency result for varying the Number of Mesh Routers

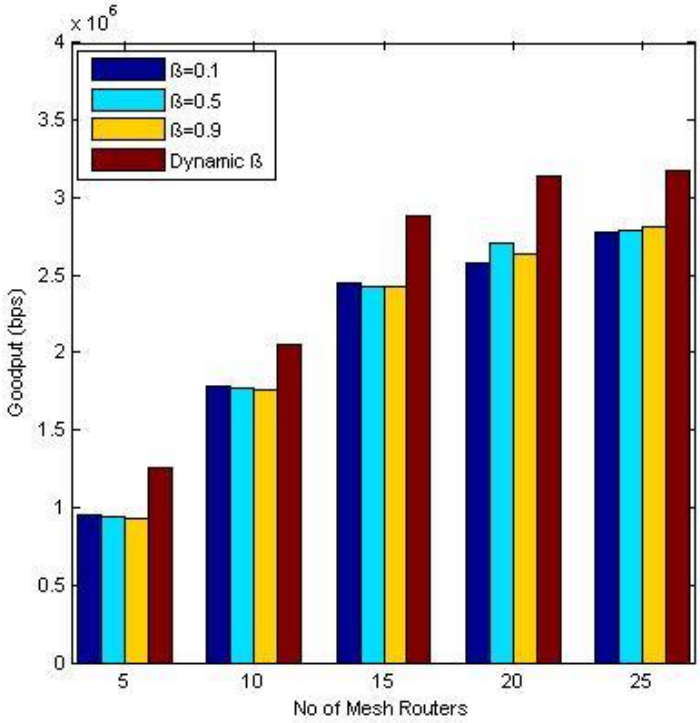


Figure 24 Goodput result for varying the Number of Mesh Routers

Goodput results for WCETT and D-WCETT with varying Mesh Routers are shown in Figure 24. The results for Dynamic β and Static β are consistent with previous results and indicate that as MR density is increased from 5 to 25, the average number of data bits successfully transmitted in the network increases. With an average transmission rate of 3.5 Mbps with 30 flows each of 32 pkts/sec, the network performs poorly at 5MRs and resultant Goodput is around 1Mbps for Static β. As the number of Mesh Routers is increased, an increase in Goodput return is observed. Results also indicate that Dynamic β achieves a much better goodput value as compared to Static β, with an average improvement of 0.5 Mbps at all test points. This improvement of D-WCETT is owing to selection of least congested links along a path, which has a multitude effect in terms of better Goodput, PDR and Latency. However increased routing overhead is observed for D-WCETT.

Chapter 6

Conclusion and Future Work

The thesis is concluded by summarizing the main contributions and pointing out some future research.

6.1 Conclusion

Hybrid WMNs present a promising technology for robust and reliable communication. They can be employed for public safety, disaster recovery and crisis management communications. The key characteristics of Hybrid WMNs are robustness, high interoperability, fault tolerance and support for wide range of applications. Owing to low cost standard commodity hardware, Hybrid WMN have the potential to provide low cost wireless broadband network. Hybrid WMNs consist of static Mesh Routers and mobile Mesh Clients, both of which contribute to the network formation and extension. Hybrid WMN presents a highly dynamic environment owing to client mobility and variable nature of wireless links. The heterogeneity and dynamic nature of these networks necessitate a challenging task of designing routing protocols for Hybrid WMNs. Traditional mobile ad-hoc routing protocols like AODV have been designed for single radio, highly mobile networks and suffer performance degradation when applied in a Multi-radio Hybrid WMN. Hence routing protocols have been proposed for multi-radio networks and a few have been adapted to Hybrid WMNs as well. Routing protocols for WMNs have been studied in this thesis along with the different routing metrics that form the core of these protocols. Hop count is the standard routing metric in most MANET routing protocols. However it has been shown that hop count does not produce optimal results. New routing metrics have been proposed since then, which try to improve upon the previous and also cater to a different set of routing demands.

This thesis proposes D-WCETT, a routing metric with load aware, dynamic path selection in multi-radio Hybrid WMNs. The proposed metric is an extension of WCETT metric and takes transmission time and channel diversity into account while at the same time assigning weight to them on the basis of link congestion. The proposed routing metric removes the limitation of WCETT, which was unable to cater for the changing network topology, while dynamically building routes based on the prevalent network load information. The network load is measured through interface queue lengths of each channel at the node. The IFQ value is then normalized to take into account links of different bandwidths and β is calculated from this parameter which is then used to compute the value of D-WCETT metric. This calculation of β is dynamic and reflective of the current channel load. Based upon the value of this metric, a node decides on which interface to forward the route request. Owing to its dynamic calculation, D-WCETT is able to select least congested links with less number of packets in queues.

The results clearly indicate the superior performance of the D-WCETT metric against different static values of β from WCETT metric when implemented on AODV protocol in a multi-radio Hybrid WMN. Higher packet delivery ratio and better latency indicate significant improvement of D-WCETT over all static β values of WCETT. However due to its characteristic of adapting to current network conditions, D-WCETT incurs additional routing overhead. The better performance of D-WCETT is attributed to the integration of β with network load.

6.2 Future Work

This research work has the potential to be further extended. A number of future avenues of work have been identified. Simulation results need to be backed up by emulation results if possible. Hence one future work is to implement D-WCETT metric on a small-scale hardware test bed. The results of such test would help to validate the simulation results and also point out any challenges faced in a real-world deployment.

Moreover, a comparative analysis of the proposed load dependent routing metric D-WCETT with other contemporary load balancing metrics can be performed. This research work has compared WCETT with different static β values against a dynamic β value (D-WCETT). A number of routing metrics have been proposed that offer performance enhancement over WCETT including MIC, iAWARE, LARM etc. Also there are a few routing metrics such as WCETT-LB, ETT-LB, CAMRLB, Load Aware Airtime Link Cost Metric that are descendants of WCETT but their performance has not been evaluated against WCETT. A comparison of

the proposed D-WCETT metric can be done against WCETT (β=0.5), WCETT-LB, LARM and MIC can be performed to assess the performance of D-WCETT against the abovementioned metrics. This can most suitably be done in a simulated environment. Such a comparison will highlight the significance of the different approaches and validate the importance of load dependent dynamic path selection in routing.

References

[1] I. F. Akyildiz, X. Wang, and W. Wang, “Wireless mesh networks: a survey,” Computer Networks, vol. 47, no. 4, pp. 445–487, 2005.

[2] R. Baumann, S. Heimlicher, M. Strasser, and A. Weibel, “A survey on routing metrics,” TIK Report, vol. 262, 2007.

[3] D. S. De Couto, D. Aguayo, J. Bicket, and R. Morris, “A high-throughput path metric for multi-hop wireless routing,” Wireless Networks, vol. 11, no. 4, pp. 419–434, 2005.

[4] R. Draves, J. Padhye, and B. Zill, “Routing in multi-radio, multi-hop wireless mesh networks,” in Proc. of the 10th Annual Int. Conf. on Mobile Computing and Networking, 2004, pp. 114–128.

[5] Y. Yang, J. Wang, and R. Kravets, “Designing routing metrics for mesh networks,” in Proc. of IEEE Workshop on Wireless Mesh Networks (WiMesh), 2005.

[6] A. P. Subramanian, M. M. Buddhikot, and S. Miller, “Interference aware routing in multi-radio wireless mesh networks,” in Proc. of 2nd IEEE Workshop on Wireless Mesh Networks (WiMesh), 2006, pp. 55–63.

[7] L. Ma and M. K. Denko, “A routing metric for load-balancing in wireless mesh networks,” in Proc. of 21st Int. Conf. on Advanced Information Networking and Applications Workshops, vol. 2, 2007, pp. 409–414.

[8] A.-N. Le, D.-W. Kum, Y.-Z. Cho, and I.-S. Lee, “LARM: A load-aware routing metric for multi-radio wireless mesh networks,” in Proc. of IEEE Int. Conf. on Advanced Technologies for Communications, 2008, pp. 166–169.

[9] J. Y. Choi and Y.-B. Ko, “Multi-path routing with load-aware metric for tactical ad hoc networks,” in Proc. of IEEE Int. Conf. on Information and Communication Technology Convergence, 2010, pp. 370–375.

[10] L. Zhao, A. Y. Al-Dubai, and G. Min, “An efficient neighbourhood load routing metric for wireless mesh networks,” Simulation Modelling Practice and Theory, vol. 19, no. 6, pp. 1415–1426, 2011.

[11] NS, “The Network Simulator,” http://www.isi.edu/nsnam/ns/, 1989.

[12] C. E. Perkins and E. M. Royer, “Ad-hoc on-demand distance vector routing” in Proc. of 2nd IEEE Workshop on Mobile Computing Systems and Applications, 1999, pp. 90–100.

[13] A. A. Pirzada, M. Portmann, R. Wishart, and J. Indulska, “Safemesh: A wireless mesh network routing protocol for incident area communications,” Pervasive and Mobile Computing, vol. 5, no. 2, pp. 201–221, 2009.