Improving Network Convergence Time and Network Stability of an OSPF-Routed IP Network

by

Amir A. Siddiqi

B. Eng. Computer Systems, Carleton University, 2002

A thesis submitted to
The Faculty of Graduate Studies and Research
in partial fulfillment of the requirements
for the degree of

Masters of Applied Sciences in Electrical Engineering

Ottawa-Carleton Institute of Electrical and Computer Engineering

Department of Systems and Computer Engineering
Carleton University
Ottawa, Ontario, K1S 5B6, Canada

September 22, 2004

© 2004, Amir A. Siddiqi

Library and
Archives Canada

Bibliothèque et
Archives Canada

Published Heritage
Branch

Direction du
Patrimoine de l'édition

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

ISBN: 0-612-97441-3

Our file Notre référence

ISBN: 0-612-97441-3

The author has granted a non-exclusive license allowing the Library and Archives Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this thesis.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

While these forms may be included in the document page count, their removal does not represent any loss of content from the thesis.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

ABSTRACT

Improving Network Convergence Time and Network Stability of an OSPF-Routed IP Network

by

Amir Siddiqi

Loss of routing protocol control messages due to network congestion can cause peer failures in routers, leading to routing failures and network instability. We investigate the effect of network congestion on Open Shortest Path First (OSPF) routing protocol, when OSPF control traffic is not isolated from the data traffic. We attempt to decrease the network convergence time in response to a node or link failure in a network. We develop an algorithm to dynamically tune OSPF failure threshold based on the network congestion level. The tunable OSPF protocol expedites failure detections as compared to standard OSPF protocol in lightly loaded networks by reducing the failure threshold parameter; whereas the failure threshold is increased during network congestion to sustain stable routing operation and avoid spurious failure decisions due to missed or delayed hello packets. Our findings demonstrate the importance of tunable failure threshold to achieve stable and robust network operation.

ACKNOWLEDGEMENTS

First and foremost, I am thankful to my supervisor Dr. Biswajit Nandy. This thesis would not have materialized without his careful and rightful guidance. His vast knowledge and experience in the area of networking proved very beneficial during this research. His active involvement and supervision allowed me to exercise my full potential in this thesis.

I also like to express my gratitude to Prof. Ioannis Lambadaris for his technical and career advices. Furthermore, I like to thank all my group members in the Broadband Network Laboratory at Carleton University who made my two years of graduate studies a wonderful and memorable experience.

Most importantly, I am very grateful to my family for their love, encouragement and support, and for providing a worry-free environment. Their constant love and support have always been the source of my strength and the reason I have come this far.

Finally, I would like to acknowledge the financial aid of Carleton University, Ontario Graduate Scholarship Program (OGS), and The Natural Science and Engineering Research Council of Canada (NSERC) towards the completion of this degree.

TABLE OF CONTENTS

ABSTRACT.....	i
ACKNOWLEDGEMENTS	ii
TABLE OF CONTENTS.....	iii
LIST OF FIGURES	vi
LIST OF TABLES.....	viii
LIST OF ACRONYMS	ix
1 INTRODUCTION.....	1
1.1 MOTIVATION.....	1
1.2 THESIS OBJECTIVES	4
1.3 THESIS OUTLINE	4
1.4 THESIS CONTRIBUTIONS	5
2 OPEN SHORTEST PATH FIRST	8
2.1 INTRODUCTION	8
2.2 OPEN SHORTEST PATH FIRST – A BRIEF OVERVIEW	10
2.3 OSPF NETWORK CONVERGENCE.....	12
2.4 PROPOSALS TO REDUCE OSPF CONVERGENCE TIME.....	13
2.5 CHAPTER SUMMARY	17
3 EFFECT OF NETWORK CONGESTION ON OSPF PERFORMANCE	19
3.1 NETWORK CONGESTION OVERVIEW	19

3.2	SIMULATION TEST CASES	21
3.2.1	<i>Evaluation Criteria</i>	21
3.2.2	<i>Experiment 1: Merging CBR Traffic Flows</i>	21
3.2.2.1	Fixed CBR Traffic Load, Variable Router Dead Interval	23
3.2.2.2	Variable CBR Traffic Load, Fixed Router Dead Interval	31
3.2.3	<i>Experiment 2: Merging FTP Traffic Flows</i>	35
3.2.3.1	Fixed FTP Traffic Load, Variable Router Dead Interval	37
3.2.3.2	Variable FTP Traffic Load, Fixed Router Dead Interval	43
3.3	CHAPTER SUMMARY	46
4	TUNABLE OSPF PROTOCOL	47
4.1	ALGORITHM	47
4.1.1	<i>Detecting Network Congestion</i>	49
4.1.2	<i>Propagating RDI Values</i>	51
4.1.3	<i>Updating RDI Values</i>	51
4.2	DISCUSSION AND ALTERNATE STRATEGIES	56
5	TUNABLE OSPF PROTOCOL PERFORMANCE EVALUATION	57
5.1	EVALUATION STRATEGY	57
5.2	EVALUATION CRITERIA	58
5.3	SIMULATION TEST CASES	58
5.3.1	<i>Simulated Hello Packet Drop Test Cases</i>	59
5.3.1.1	Gradually Increasing Network Congestion	60
5.3.1.2	Gradually Increasing then Decreasing Network Congestion	62
5.3.1.3	Asymmetric Network Congestion	64

5.3.2	<i>Congestion Based Hello Packet Drop with Merging CBR Traffic Flows</i>	66
5.3.2.1	Network Topology	66
5.3.2.2	Route Selection	67
5.3.2.3	Traffic Overload Test.....	69
5.3.2.4	Router Buffer Size Test	72
5.3.2.5	Data Packet Size Test.....	74
5.3.3	<i>Comparison of Standard and Tunable OSPF with Heavy FTP Load.....</i>	77
5.4	CHAPTER SUMMARY	82
6	APPLICATIONS OF TUNABLE OSPF PROTOCOL	83
6.1	INTERNET NETWORK.....	83
6.2	WIRELESS MESH NETWORKS	85
6.3	CHAPTER SUMMARY	88
7	CONCLUSION	89
7.1	THESIS SUMMARY	89
7.2	FUTURE DIRECTIONS.....	91
7.2.1	<i>Tunable OSPF Implementation in a Linux Test-bed.....</i>	<i>91</i>
7.2.2	<i>Comparison of Standard and Tunable OSPF Protocols on WMN.....</i>	<i>91</i>
7.2.3	<i>Incorporating SNR in Tunable RDI Algorithm for WMN.....</i>	<i>91</i>
	REFERENCES.....	93

LIST OF FIGURES

Figure 3-1: Simulation Network Topology for Merging CBR Traffic Flows	23
Figure 3-2: Results of Heavy CBR Traffic Load, RDI=4 seconds Test Case	25
Figure 3-3: Results of Heavy CBR Traffic Load, RDI=8 seconds Test Case	26
Figure 3-4: Results of Heavy CBR Traffic Load, RDI=12 seconds Test Case	27
Figure 3-5: Results of Heavy CBR Traffic Load, RDI=16 seconds Test Case	28
Figure 3-6: Results of Heavy CBR Traffic Load, RDI=20 seconds Test Case	29
Figure 3-7: A Comparison of Number of Lost Adjacencies with Different RDI values in Heavy CBR Traffic Load.....	30
Figure 3-8: Results of Medium CBR Traffic Load, RDI=4 seconds Test Case	33
Figure 3-9: Results of Light CBR Traffic Load, RDI=4 seconds Test Case	34
Figure 3-10: Buffer Usage for Merging Node ‘M’ with Light, Medium and High CBR Traffic Flows with RDI=4 seconds.....	35
Figure 3-11: Simulation Network Topology for FTP Traffic Flows	36
Figure 3-12: Results of Heavy FTP Traffic Load, RDI=4 seconds Test Case	40
Figure 3-13: Results of Heavy FTP Traffic Load, RDI=8 seconds Test Case	41
Figure 3-14: Results of Heavy FTP Traffic Load, RDI=12 seconds Test Case	42
Figure 3-15: Results of Medium FTP Traffic Load, RDI=4 seconds Test Case	44
Figure 3-16: Results of Light FTP Traffic Load, RDI=4 seconds Test Case	45

Figure 4-1: Tunable RDI OSPF Algorithm Pseudo Code	54
Figure 4-2: Tunable RDI OSPF Algorithm Flowchart	55
Figure 5-1: Network Topology for Simulated Hello Packet Drop Test Cases	60
Figure 5-2: Simulated Hello Packet Drop Test Case 1	61
Figure 5-3: Simulated Hello Packet Drop Test Case 2	63
Figure 5-4: Simulated Hello Packet Drop Test Case 3	65
Figure 5-5: Network Topology for Congestion-based Hello Packet Drop Test Cases.....	67
Figure 5-6: Traffic Overload Test.....	71
Figure 5-7: Buffer Size Test	73
Figure 5-8: Data Packet Size Test with Constant Packet Processing Speed.....	76
Figure 5-9: Total Number of Routing Updates Sent in the Network in Data Packet Size Test Case.....	77
Figure 5-10: Simulation Network Topology for Comparing Tunable and Standard OSPF Performance with Heavy FTP Traffic Flows.....	78
Figure 5-11: Comparison of Standard and Tunable OSPF Protocols with Heavy FTP Traffic Load	81
Figure 6-1: Byte Volume for 1-day and 7-days Period on a Domestic Link.....	84
Figure 6-2: Byte Volume for 1-day and 7-days Period on an International Link.....	85
Figure 6-3: Wireless Mesh Network.....	86

LIST OF TABLES

Table 3-1: Comparison of OSPF Control Packets in Fixed CBR Traffic Load, Variable RDI Test Case	31
Table 3-2: Configuration of FTP Traffic Flows	36
Table 4-1: Network Congestion Lookup List	50

LIST OF ACRONYMS

BGP:	Border Gateway Protocol
CBR:	Constant Bit Rate
CPU:	Central Processing Unit
FIFO:	First-In First-Out
FTP:	File Transfer Protocol
HTTP:	Hypertext Transfer Protocol
IETF:	International Engineering Task Force
IGP:	Interior Gateway Protocol
IP:	Internet Protocol
IS-IS:	Intermediate-System-to-Intermediate-System
LAN:	Local Area Network
LSA:	Link State Advertisement
LSU:	Link State Update
NCT:	Network Convergence Time
OSPF:	Open Shortest Path First
PPP:	Point-to-Point Protocol
QoS:	Quality of Service
RDI:	Router Dead Interval
RFC:	Request for Comments
RIP:	Routing Information Protocol
SNR:	Signal-to-Noise Ratio
SONET:	Synchronous Optical Network
TCP:	Transmission Control Protocol
UDP:	User Datagram Protocol
WMN:	Wireless Mesh Networks

1 INTRODUCTION

1.1 MOTIVATION

The startling growth of Internet and the flexibility of IP-based packet switched networks have expedited the convergence of data communications (packet switched) and voice-based communications (traditionally circuit switched) into single IP-based core architecture. Internet has become an interconnection of various types of heterogeneous networks and is supporting a wide range of applications. Besides traditional web applications such as HTTP and FTP, Internet is experiencing an exponential growth in audio and video streaming, and other real-time applications. However, the existing Internet only offers a single best-effort class of service that is mainly designed for the traditional applications, and can adversely impact the end-to-end performance of the real-time applications. Clearly, there is a need to re-examine the current Internet architecture, and designing and building an intelligent Internet infrastructure to complement these new breed of applications.

Decreasing Network Convergence Time (NCT) in response to a critical event in a network, such as a node or link failure, is significantly important for next-generation, real-time network applications. The NCT is defined as the difference in time when the

critical event has occurred until the time the network is stabilized and all nodes have a loop-free path to every other node in the network domain. Network failures have been observed to be fairly common in the day to day operations of a network due to fiber cuts, hardware/software failures, faulty equipments, or router misconfigurations [1]. Athina et al. [2] analyzed the failures occurring on a Sprint Network over a period of 7-months from April 2002 to October 2002. The paper reveals that only 20% of the failures occurring in the network were planned due to scheduled maintenance, while the rest were unplanned failures with 30% of the unplanned failures spanning multiple links and routers. There is a need to minimize the effect of such network failures that may lead to service disruptions and adversely affect the performance of the web applications.

Guaranteeing a Quality of Service (QoS) for an application requires that the NCT of the network, where the application is running, is adequate for the operation of the application. The network is considered unstable after a node or link failure, leading to routing failures and data packet drop, until the network has converged. During the transient period of network convergence, the application packets may be dropped due to invalid routes such as the ones transiting through the failed node or link, or the packets getting caught in routing loops. Any dropped application data packets need to be retransmitted once the network has converged. In today's high speed networks, even a short recovery time can cause huge packet losses. For example, if an OC48 link is down for 10 seconds, close to 3 million packets (assuming a packet size of 1KB) could be lost [3]. Furthermore, retransmission, which is considered an acceptable option in the traditional applications such as FTP, may not be feasible for the operation of most real-time applications due to an unacceptable delay incurred during retransmissions.

At the present, various kinds of networks such as Optical network, IP network, Wireless network, etc. coexist with each other. These networks exhibit very different NCT behavior depending on the capabilities of the physical layer and type of protection scheme provisioned in the network. For instance, in case of an optical SONET network, a node or link failure is detected immediately at the physical layer by the neighboring nodes. The neighbors initiate the local recovery process and the traffic transiting through the failed node or link is switched to a predetermined backup route. The backup route can be installed in the order of milliseconds and usually the network is recovered with little or no application data loss.

In the IP networks, on the other hand, a failure is detected by the adjacent nodes after expiration of a fixed amount of time without receiving any keep-alive message from a particular node. Each node in an IP network periodically sends hello messages to all adjacent neighbors. If a hello message is not received from a particular neighbor for a predetermined fixed time, called router dead interval (RDI), the host declares the neighboring node down and initiates the recovery process. The adjacency is terminated and all nodes in the network domain are notified about the failure. Each node upon receiving the update removes the failed node from its network topology and recalculates the routes to all other nodes in the network. This process of detecting the failure can take up to 40 seconds in the current IP networks.

In this thesis, we attempt to decrease the NCT of an IP network running Open Shortest Path First (OSPF) routing protocol. The results are based on simulations performed in *Opnet Modeler* Simulation Environment. We investigate the tradeoff between network stability and decreasing the NCT. Based on our findings, we propose

an algorithm to decrease the NCT while achieving a stable network operation for a point-to-point OSPF network. The proposed algorithm is implemented in the OSPF model provided by *Opnet Modeler* and the performance of the standard OSPF and proposed tunable OSPF protocols is analyzed and compared in various network scenarios.

1.2 THESIS OBJECTIVES

- Study the importance of decreasing the NCT
- Research and investigate the recent proposals to decrease the NCT
- Investigate the performance of the standard OSPF protocol, with decreased NCT, in various network congestion scenarios for different traffic types
- Propose and implement an algorithm to decrease the NCT for an OSPF network
- Analyze and compare the performance of the proposed and standard OSPF protocols in various network congestion scenarios and traffic types
- Explore the viability and applications of the proposed algorithm in the current and emerging IP networks

1.3 THESIS OUTLINE

The rest of this thesis is organized as follows:

Chapter 2: In chapter 2, we review the fundamentals of Open Shortest Path First (OSPF) routing protocol. We introduce the term ‘Network Convergence Time’ and explain how the NCT is calculated for an OSPF network. We also present a

study of the related past works in the field of network convergence and network stability.

Chapter 3: In chapter 3, we investigate the performance of the standard OSPF protocol in various network congestion scenarios for Constant Bit Rate (CBR) and FTP traffic flows. We show that the optimal RDI value, required for decreased NCT, is different in different network congestion levels.

Chapter 4: In chapter 4, an algorithm is described to dynamically tune the RDI value based on the network congestion level.

Chapter 5: In chapter 5, we analyze and compare the performance of the standard and proposed tunable OSPF protocols in various network congestion scenarios and for different data traffic types. We also investigate the effect of router buffer size and data packet size on the performance of standard and tunable OSPF protocols.

Chapter 6: In chapter 6, we discuss the applicability and viability of the proposed tunable OSPF protocol in the current and emerging IP networks.

Chapter 7: In chapter 7, we summarize our findings and suggest some future directions for this work.

1.4 THESIS CONTRIBUTIONS

The contributions of this thesis are two-fold. In the first part, we study the tradeoff in the standard OSPF routing protocol between network stability and decreasing

the NCT. In the second part, we propose an algorithm to decrease the NCT of an OSPF-routed IP network while assuring a stable and robust routing operation.

The following are the main contributions of this thesis:

- A detailed analysis of the behavior of standard OSPF routing protocol in varying network congestion scenarios for CBR and FTP traffic flows is presented
 - The performance of the OSPF protocol is shown to vary significantly with the variation in the network congestion for all types of traffic flows
- A development of an algorithm to tune the RDI value based on the network congestion level for a point-to-point OSPF network
- A detailed comparison of the standard and the proposed tunable RDI OSPF protocols in various congestion scenarios is presented
 - The tunable OSPF protocol is found to be comparable or outperformed the standard OSPF protocol in all the scenarios
 - The proposed tunable OSPF protocol is found to be more stable than the standard OSPF protocol.
 - The proposed tunable protocol is shown to provide a less disruptive flow of data and facilitate shorter download times for FTP traffic as compared to the standard OSPF protocol
- The effect of router buffer size and data packet size on the performance of standard and tunable OSPF protocols is investigated

- The protocols are found to be insensitive to the router buffer sizes in high network congestion and depict similar relative performance
- The network congestion level is shown to decrease with the increase in the data packet size with constant packet processing speed
- The protocols are found to behave comparably with large data packet size
- The tunable OSPF protocol demonstrated significant performance improvement over the standard OSPF with small data packet size
- The applications of the proposed tunable OSPF protocol in the current and emerging IP networks are discussed

2 OPEN SHORTEST PATH FIRST

In this chapter, we review the fundamentals of OSPF routing protocol, emphasizing more on the issues related to network convergence. We also present a study of the related past works in the field of OSPF network convergence and network stability.

2.1 INTRODUCTION

Internet connects millions of computers World-wide and allows them to communicate with each other. Special purpose computers, called routers, exist in the Internet that make this communication possible by delivering the message from one computer destined to another computer. Internet today is mostly a best-effort service based on datagram packets. Best effort service implies that there is no guarantee that a message will be delivered successfully to its final destination. A message is divided into packets and then each packet is routed independently through the Internet. Each router upon receiving a packet extracts the packet header that contains the destination address and consults its routing table to find out the neighbor who should be forwarded this packet. The process of forwarding the packet continues until the packet has reached its final destination or maximum hop count has reached. To populate the routing tables and determine the next hop for a message destined for any computer around the World, each

router runs routing protocols that allow the routers to collaborate and gather information about the topology of the network. Common examples of routing protocols are Open Shortest Path First (OSPF), Routing Information Protocol (RIP), Border Gateway Protocol (BGP) and Intermediate System-Intermediate System (IS-IS).

Routing protocols can be classified into Distance Vector and Link-State based on the method of learning to find the topology of the network. In the distance vector routing protocol, each node advertises to all its neighbor routers the distance to every other reachable router. Each router, upon receiving the advertisement, compares the distance to every router in its routing table to the distances contained in the advertisement. If the advertisement contains routers that are not present in the host routing table, the host adds those routers in its routing table and makes the advertisement sender as the next hop. For routers that exist in the host routing table, the host compares the current distance with the distance contained in the advertisement. If going through the router, from which the advertisement has been received, results in a shorter aggregate distance, the host router modifies its routing table to make the advertisement sender as the next hop. The protocol continues in this fashion until it has built the topology of the entire network.

Distance vector protocols suffer from slow convergence in order of tens of minutes and formation of routing loops [4]. A number of techniques, such as “split horizon” and “triggered updates”, are proposed to speed up the convergence [5]. A router only knows about the distance to the destinations and has no information about the exact path that the packets are going to take. This can result in two nodes picking each other as the next hop and essentially forming a routing loop. BGP is a distance vector protocol widely deployed in the current Internet infrastructure. BGP avoids forming the routing

loops by including the entire paths along with the distance estimates. The router avoids using those neighbors whose path to a destination passes through the router itself as the next hop for that destination.

In link-state routing algorithm, each node in a network domain builds an entire view of the network topology. Each node is responsible for maintaining the status of the links incident to it. The status information is flooded to all nodes in the network domain and each node develops a topology of the network using this information. Every router then builds a single source routing path tree with itself as a root and compute possible paths to all other nodes in its routing domain. Routing is consistent as long as all the routers in the domain have the same view of the network. The network is said to be converged when all nodes in the network have a consistent view of the network. Any change in the network topology must be flooded to all the nodes in the network domain in order to maintain a consistent view of the network at all the nodes. The network is considered unstable from the time when the change has occurred until the time the network has converged and all the nodes have re-established a consistent view of the network topology.

2.2 OPEN SHORTEST PATH FIRST – A BRIEF OVERVIEW

Open Shortest Path First (OSPF) is the most commonly employed link state Interior Gateway Protocol (IGP) in the current Internet infrastructure. ‘Open’ stands for the fact that the OSPF protocol is not proprietary to any organization. ‘Shortest Path First’ symbolizes the fact that the OSPF protocol is based on Dijkstra SPF algorithm. John Moy, the writer of OSPF request for comments (RFC), first introduced version 1 of

this protocol in RFC 1131. Currently, however, OSPF version 2 is in use. OSPF Version 2 was introduced in RFC 1247 and its latest update is RFC 2328.

To find routing paths, OSPF assigns weights to each link in the network. It is proposed to make the weights inversely proportional to the link bandwidth [6]. To find an optimal route, OSPF selects the route that minimizes the aggregate weight. If two or more routes have an equal aggregate weight, all these routes are included in the routing table for load balancing purposes.

The operation of OSPF is centered on the exchange of hello messages. During initialization, an OSPF node sends hello messages to all its neighbors. The neighbors supporting the OSPF protocol recognize the hello packets and realize that these hello packets are from a new host. The neighbors then exchange their routing tables by sending link state advertisements (LSA) and data description packets. The OSPF neighbors are said to be adjacent, once the neighbors have identical topological databases. To keep the adjacency alive, the neighbors periodically send hello messages to all adjacent neighbors. Not receiving a hello packet from a particular neighbor for a fixed amount of time, called router dead interval (RDI), indicates that the neighbor is down and the adjacency is terminated. The neighbor, detecting the failure, then sends the failure information to all other neighbors as link state updates (LSU). Each node upon receiving the LSU removes the failed node from its database, recalculates routing paths to all other nodes and forwards the LSU to all its neighbors. Once all nodes in the network domain have received the failure information and updated their routing tables, the network is said to be converged. After the network is converged, all the nodes have a consistent view of the network. From the time of the failure until the network is

converged, the nodes may have inconsistent network view and routing failures can occur. No state information is kept for the failed node and thus when the node comes up again, the entire synchronization process is repeated.

2.3 OSPF NETWORK CONVERGENCE

Network Convergence Time (NCT) is defined as the difference in time when an event has occurred in the network until the time the network is stabilized and all nodes have a loop-free path to every other node in the network domain. The NCT can be calculated as the sum of the event detection time, information flooding time, update processing time, paths computation time and route installation time. The NCT is determined by the diameter of the network (maximum number of hops between the any two routers), network congestion, routing load on processors, and several other factors. A low NCT indicates a more stable network. This is because a network can quickly come back to the stable state when perturbed [7].

In an OSPF network, a failure is detected by the neighboring nodes after expiration of the RDI without receiving any hello message from the adjacent node. The RDI value is a constant in the current OSPF standard and is usually set to four times the hello interval [6]. The default values for the hello interval and the router dead interval are 10 and 40 seconds respectively. With these values, detecting a failure in a network can take up to 40 seconds. From the time of failure until the network has converged, the data traffic is continued to be forwarded to the failed node or link. The data packets, forwarded to the failed node or link, are dropped and need to be retransmitted. Recently, it has been suggested to decrease the hello and router dead intervals to expedite the

failure detection time which consequently decrease the overall NCT. The values of 2 seconds for the hello interval and 8 seconds for the RDI are proposed in [8].

Routing protocol control packets usually share resources such as link bandwidth and router buffers with the data traffic. Congestion in the shared resources can thus, hinder the propagation of the control packets. The hello packets can get queued with the other data packets resulting in delayed transmission or packet drop. The adjacent neighbors will erroneously presume that the node is down if no hello packet has been received within the RDI expiration time and initiate the recovery process. In this thesis, we will use the term ‘spurious’ or ‘premature’ lost adjacency to refer to the failure decision that is made when the node is in fact, up and running. The probability of delaying or dropping the hello packets increases with the increase in network congestion. Decreasing the hello and router dead intervals make it more likely to declare spurious failures. As such, the hello interval and the RDI are two important parameters in an OSPF network. A small value of the hello interval allows faster failure detections but results in excessive control traffic whereas its large value prolongs real failure detection time. A small RDI value expedites failure detections but may result in premature failure decisions due to missed or delayed hello packets, especially in a congested network; whereas a large RDI value results in longer real failure detection time.

2.4 PROPOSALS TO REDUCE OSPF CONVERGENCE TIME

Decreasing the NCT is a major research topic these days and a number of techniques have been proposed to decrease the NCT of an OSPF network. This section

gives an overview of the related work and suggested improvements in some recent publications.

1) Explicit Marking and Control Traffic Prioritization

Reference [9] proposes processing the hello packets at a higher priority as compared to other OSPF control packets and application data packets. Assigning higher priority ensures that the hello packets are given differentiated treatment in the routers during the times of network congestion. In order to facilitate the priority queuing, the hello packets can be explicitly marked to differentiate the hello packets from the other OSPF control packets.

Prioritizing the hello packet allows reducing the hello and router dead intervals to decrease the overall NCT. In the absence of priority scheme, reducing the hello and router dead intervals increases the probability of making spurious adjacency failure decisions as the hello packets are likely to be delayed or dropped with the other data and OSPF control packets.

Simulations results given in the paper show that prioritizing the hello packets results in significant improvement in the scalability and stability of the network. Based on the promising results, authors further proposed prioritizing other selected OSPF control packet such as the data description packets. Prioritizing the data description packets will reduce the adjacency synchronization time and thus, improve the NCT. It is recommended to incorporate this priority information in the packet header, such as by using the reserved DiffServ bits, to enable the prioritized treatment without any modifications in the protocol.

2) Keeping Backup Paths

Keeping a backup path can replace slower path recalculations by faster path switching in a situation when a router or link fails. This approach is most beneficial when the number of shared links is minimized [10]. The shortest path of the network is deployed as the primary path, and the other path is deployed as a backup.

The technique of keeping backup paths is more applicable during the link failures as the connecting nodes immediately detect the failure (via Multi-Access Reachability Protocol) and thus, can initiate the path switching in real time. The path switching during a link failure can be made fast enough to make the failure oblivious from the high level layers. In case of a node failure, the failure is detected after the expiration of RDI and employing the backup routes would allow saving the time required for LSU transmissions to all the nodes and SPF re-calculations.

3) Local Restoration Algorithm – Incremental SPF

References [11] and [12] detail algorithms to localize the domain where the failure information needs to be transmitted. The algorithms restore all the paths traversing the failed link or node by only performing local updates on the neighboring routers. The objective is to restore loop-free routing after a node or link failure, while spreading the failure information to as few routers as possible. The unidirectional restoration algorithm is documented in [11] whereas [12] extends the algorithm for bidirectional links.

4) Intelligent Flooding

OSPF requires that all nodes in a network have identical link state databases and thus, all the nodes in the network domain need to be notified about any change in the network topology. OSPF reduces the updates flooding frequency by defining a *hold time* which is the time a node must wait, after receiving an update, before forwarding the update to its neighbors. The intent of the hold time is to accumulate as many updates as possible before sending a composite update to the neighbors and reduce the number of duplicate updates. However, an evident implication of this approach is increasing the NCT. A possible solution proposed in [13] is to immediately forward the updates containing critical information, such as a node or link failure, whereas delaying the propagation until the hold time for the updates that are not critical, such as a node discovery. The scheme would allow further increasing the hold time to reduce the flooding frequency as the updates waiting until the hold time to be transmitted will not be carrying any time-critical information.

5) Database Backup and Synchronization – Graceful Recovery

A node failure results in losing all the locally stored topology database information that must be rebuilt when the node recovers. It is proposed in [14] to periodically backup the topology database at the local node. In case of a failure and recovery, the backup topology database information can be re-used for topology rediscovery. However, the information derived from the backup database needs to be resynchronized with the neighbors' topological databases. By using the backup database, the recovered node only requires the updates occurred between the last database backup

time and the node recovery time to be completely synchronized. As such, no LSA transmissions are required if the backup database is completely up to date.

6) Using Any OSPF Control Packets to Maintain Adjacency

If the link between two OSPF routers is congested, for example due to a high volume of LSA transmissions, it is possible for several consecutive hello packets to get dropped or delayed beyond the RDI. When the RDI expires, the adjacent node will assume that the neighbor node is down even though, because the LSA are being received from that node, it is clearly up. It is proposed in [14] to accept the arrival of any OSPF control packet from the adjacent neighbor, as a keep-alive signal, to retain the adjacency.

6) Reducing OSPF Hello Interval to Achieve Faster Failure Detections

Reference [15] explores the possibility of reducing the OSPF hello interval to decrease the failure detection time. The paper presents the results of experimenting with different hello interval values in different network congestion levels. The aim of the experiments was to find an optimal hello interval value that reduces the failure detection time while keeping the spurious failure decisions within acceptable limits. It is concluded that the optimal value of the hello interval is strongly influenced by the network congestion level and the number of links in the topology.

2.5 CHAPTER SUMMARY

In this chapter, we explained the concept of Network Convergence Time (NCT) and investigated the factors contributing to network convergence. We also presented a study of the recent related work in the field of network convergence. Most of these recent works have mainly focused on reducing the flooding information and locally

restoring the network in case of node or link failures. Only one proposal investigated the possibility of reducing the failure detection time by reducing the hello interval of an OSPF network.

A major factor in detecting the failures is the router dead interval (RDI) which essentially determines the time it takes to detect the failure in the network. The effect of varying the RDI value is not investigated in any of the prior studies. In the next chapter, we will explore the possibility of expediting the failure detection time by decreasing the RDI value and analyze its implications on the stability of the network.

3 EFFECT OF NETWORK CONGESTION ON OSPF PERFORMANCE

In this chapter, we first review the factors contributing to network congestion in an IP network. We then investigate the performance of standard OSPF protocol under different network congestion scenarios with Constant Bit Rate (CBR) and FTP traffic flows. We find an optimal router dead interval (RDI) value for different network congestion levels and show that the optimal RDI value varies with the change in the network congestion level for all traffic types.

3.1 NETWORK CONGESTION OVERVIEW

Network resources such as router processing power and link throughput are limited, and the network can be overloaded from time to time resulting in deterioration of the overall network performance. This situation is called congestion. Congestion can be brought up by several factors. If the incoming traffic rate at a router is greater than the capacity of output lines, a router queue will build up. For example, if a number of streams of packets are merging at one router, delivering packets at top speed, en route for the same output interface, the router queue will build up. Once the queue is full, any

further incoming packets are dropped. Making a queue size bigger may help up to a point, but it can be shown that routers with infinite queues make the congestion worse, not better, because by the time packets get to the front of the queue, they have already timed out, and duplicates have been sent [16]. Furthermore, most next-generation, real-time applications have stringent end-to-end delay requirements that limit the size of the router queues.

Slow processors can also cause congestion. If the routers' CPUs are slow, queues can build up, even though there is excess link capacity. This is because, a CPU is responsible to determine the output interface for each packet by consulting the routing table and forwarding the packet to the output queue. If the CPU is unable to keep up with the speed of the incoming packets, a queue will build up.

Congestion tends to feed itself and become worse [16]. If a queue is full, the router must ignore newly arriving packets. When a packet is discarded, the sending router may time out and retransmits the packet, perhaps ultimately many times. Since the sender can not discard the packets until it has been acknowledged, congestion at the receiver's end forces the sender to keep the packet in its outgoing buffer. As such, the congestion backs up and becomes worse.

Congestion is a network-wide issue, involving the behavior of all the elements in the network such as the hosts, the routers and the links. To alleviate network congestion, the network must be carefully examined and the congestion bottleneck must be determined. The congestion will persist until all the components in the network are in balance.

3.2 SIMULATION TEST CASES

In this section, we investigate the performance of standard OSPF routing protocol in various network congestion scenarios. The results are based on simulations performed in *Opnet Modeler 9.0* Simulation environment. We first analyze the performance of the standard OSPF protocol for CBR data traffic and later extend the analysis for FTP traffic flows.

3.2.1 EVALUATION CRITERIA

We investigate the performance of the standard OSPF protocol by varying magnitude of the applied traffic load. For each test case, we find the optimal RDI value which is its lowest value for which the network stays stabilized all the time for a given congestion level. Any RDI values greater than the optimal value prolong real failure detection time whereas any RDI values lower than the optimal value result in spurious failure decisions, causing routing failures and network instability. The optimal RDI value is determined based on the number of adjacencies lost. As there are no real node or link failures simulated in the network, the number of lost adjacencies essentially depicts the number of spurious failure decisions made over the course of simulation. Thus, the lower number of adjacency failure decisions depict superior network protocol performance.

3.2.2 EXPERIMENT 1: MERGING CBR TRAFFIC FLOWS

In this experiment, we analyze the performance of the standard OSPF protocol with CBR traffic flows. The CBR traffic is unreliable and unidirectional. It is predictable as the packet size, packet interval and stream duration are known. As such, the real-time audio and video streaming traffic can be approximated by the CBR traffic.

Most audio and video streams are based on User Datagram Protocol (UDP) and thus, are also unreliable and unidirectional. Due to absence of flow control in UDP, the source transmits the traffic at a constant and predictable rate. Due to these similarities, we decided to model the real-time UDP streaming traffic with the CBR traffic due to ease of its implementation in *Opnet Modeler*.

The simulation network topology is shown in Fig. 3-1 that consists of 4 nodes connected by PPP-DS3 links supporting a maximum bandwidth of 44.7Mbits/second. Two traffic flows are defined: The first CBR traffic flow from Node A to Node C starts at time=200 seconds and has a duration of 1100 seconds; whereas the second CBR traffic flow from Node B to Node C starts at time=500 seconds and has a duration of 500 seconds. As such, the two flows meet at a common interface at Node M from time=500 seconds until time=1000 seconds. Three traffic loads are defined: heavy traffic load in which each traffic flow is 50Mbits/second, medium traffic load where each flow is 25Mbits/sec and light traffic load where each flow is 15Mbits/second.

We performed two experiments in this setup. In the first experiment, the traffic load is fixed at heavy and the simulations are run with different RDI values. In the second experiment, the RDI value is fixed at 4 seconds and the traffic load is varied. The hello interval of 2 seconds is used throughout the experiment, as recommended in [8], to expedite real failure detections in the network. The data and control traffic flows are assigned equal priority and thus, share the router FIFO queue and link bandwidth.

Figure 3-1: Simulation Network Topology for Merging CBR Traffic Flows

3.2.2.1 Fixed CBR Traffic Load, Variable Router Dead Interval

Fig. 3-2 to Fig. 3-6 show the results of varying the RDI value on the performance of standard OSPF protocol when the CBR traffic load is fixed to heavy. We vary the value of RDI from 4, 8, 12, 16 and 20 and observe the number of OSPF control packets dropped, number of adjacencies lost, and number of link state updates (LSU) received in each case. When we compare the number of control packets dropped in each case, we notice that the case of RDI=4 seconds resulted in twice as many control packets dropped as compared to other cases with higher RDI values. To explain this behavior, we look at the number of adjacencies lost and number of LSU packets received in each case. We notice that the case of RDI=4 seconds resulted in 131 spurious adjacency losses. The rate of losing adjacencies increases from time=500 seconds to time=1000 seconds when both traffic generators are active and accordingly, the rate of control packets dropped also

increased during that time. Each adjacency loss and later discovery resulted in LSU packets being sent to notify the other nodes in the network about the update. As such, more control packets are being sent that resulted in further worsening the network congestion and more control packets being dropped. We notice that for RDI=4 seconds, adjacencies are lost continuously during the entire time the two traffic flows are active. As shown in Fig. 3-3(b), the case of RDI=8 seconds showed in only 4 adjacencies being lost and subsequently, the number of LSU packets being sent also decreased. The case of RDI=12 seconds resulted in only one adjacency losses as shown in Fig 3-4(b). The graphs for the cases of RDI=16 seconds and RDI=20 seconds, given in Fig. 3-5 and Fig. 3-6 respectively, showed no adjacency losses and thus LSU packets are received only in the beginning of the simulation, during the initial data exchange between the nodes.

When we compare the number of adjacencies lost with different RDI values, we notice that increasing the RDI value from 4 to 8 shows a sudden decrease in the number of lost adjacencies (from 131 to 4) but increasing the RDI value further only has a nominal effect. The rationale for this non-linear behavior is that an adjacency is lost only if no hello packet is received within the RDI expiration time. Receiving a hello packet just before the RDI expiration time results in retaining the adjacency and receiving a hello packet just after the RDI expiration time results in a lost adjacency. To get further insight for this non-linear behavior, we ran the simulations for the RDI value of 6 seconds that resulted in 25 lost adjacencies. The graph of the number of lost adjacencies for different RDI values is shown in Fig. 3-7 which confirms the non-linear behavior of the number of lost adjacencies with respect to varying the RDI value.

(a) Number of control packets dropped for RDI=4 seconds

(b) Number of adjacencies lost for RDI=4 seconds

(c) Number of LSU packets received for RDI=4 seconds

Figure 3-2: Results of Heavy CBR Traffic Load (50Mbits/sec), RDI=4 seconds Test Case

(a) Number of control packets dropped for RDI=8 seconds

(b) Number of adjacencies lost for RDI=8 seconds

(c) Number of LSU packets received for RDI=8 seconds

Figure 3-3: Results of Heavy CBR Traffic Load (50Mbits/sec), RDI=8 seconds Test Case

(a) Number of control packets dropped for RDI=12 seconds

(b) Number of adjacencies lost for RDI=12 seconds

(c) Number of LSU packets received for RDI=12 seconds

Figure 3-4: Results of Heavy CBR Traffic Load (50Mbits/sec), RDI=12 seconds Test Case

(a) Number of control packets dropped for RDI=16 seconds

(b) Number of adjacencies lost for RDI=16 seconds

(c) Number of LSU packets received for RDI=16 seconds

Figure 3-5: Results of Heavy CBR Traffic Load (50Mbits/sec), RDI=16 seconds Test Case

(a) Number of control packets dropped for RDI=20 seconds

(b) Number of adjacencies lost for RDI=20 seconds

(c) Number of LSU packets received for RDI=20 seconds

Figure 3-6: Results of Heavy CBR Traffic Load (50Mbits/sec), RDI=20 seconds Test Case

Figure 3-7: A Comparison of Number of Lost Adjacencies with Different RDI values in Heavy CBR Traffic Load (50Mbps/sec)

Table 3-1 shows a comparison of the number of OSPF control packets for the different RDI values. We notice that the total number of OSPF control packets sent and the number LSU received in the network are the highest for RDI = 4 seconds case where there are the most number of lost adjacencies. There is a significant decrease in all the numbers for RDI = 8 seconds and RDI = 12 seconds cases where there are only 4 and 1 lost adjacencies respectively. As expected, the cases of higher RDI values, with no lost adjacencies, depict same numbers, slightly lower than the RDI = 12 seconds case.

Based on these results, we can say that 16 seconds is the optimal RDI value for this network operating under heavy CBR traffic load. Any values lower than 16 seconds would result in spurious adjacency losses whereas any RDI values higher than 16 seconds increase the real failure detection time. With RDI = 16 seconds, there will be no spurious adjacency losses and the network will stay stabilized throughout the simulation.

RDI	Number of Adjacencies Lost	Number of OSPF Control Packets Sent	Number of OSPF Control Packets Dropped	Number of LSU Packets Received
4	131	6699	356	584
8	4	4566	150	75
12	1	4464	148	40
16	0	4416	143	26
20	0	4416	143	26

Table 3-1: Comparison of OSPF Control Packets in Fixed CBR Traffic Load (50Mbps/sec), Variable RDI Test Case

3.2.2.2 Variable CBR Traffic Load, Fixed Router Dead Interval

In this experiment, the RDI value is fixed at 4 seconds and the CBR traffic load is varied. We performed the simulations with medium and light traffic flows and compared the results with the result of the heavy traffic flow given in Fig. 3-2. Fig. 3.8 and Fig. 3-9 show the results of medium and light traffic loads respectively on the performance of standard OSPF protocol. When we compare the number of control packets dropped in each case, we notice that the case of heavy traffic load resulted in 356 control packets dropped as compared to 46 packets dropped in the medium traffic load case; whereas there is no control packet dropped in the case of light traffic load. The reason for the difference can be understood by analyzing the number of spurious lost adjacencies. In case of high traffic load, the adjacencies are being lost continuously during the entire time the two traffic flows are merging. Each lost adjacency and later discovery requires sending updates in the network to notify the other nodes about the update. As more

control packets are being sent, the network congestion worsens and resulted in more control packets being dropped. The network running under medium traffic load lost only 15% of the adjacencies as compared to the high traffic load case. As expected, the number of control packets dropped also decreased with medium traffic load as compared to the high traffic load case due to fewer number of control packets being sent. With light traffic load, no adjacency is lost and no control packets are dropped as the network is running well below its operating limit.

The reason for the variability in the control packet drop rate can be seen from Fig. 3-10 which shows the buffer usage for the merging node 'M' for the three congestion cases. We notice that the router buffer stays 100% full for the high traffic load from time=500 seconds to time=1000 seconds resulting in data overflow. For medium traffic load case, the buffer is on average 85% full from time=500 seconds to time=1000 seconds and only reaches 100% at couple of occasions. For the light traffic load case, the buffer is on average only 5% full when the two traffic flows are merging and never reaches its full capacity and thus, no control packet is dropped.

Based on these results, we can say that 4 seconds is the optimal RDI value for the light traffic load case. With this value, the network will detect a real node or link failure in 4 seconds while avoiding any spurious adjacency lost decisions. Similar analysis showed that 8 seconds is the optimal RDI value for the medium traffic load and recall that in section 3.2.2.1, we have already shown that 16 seconds is the optimal RDI value for the network running under the high traffic load.

(a) Number of control packets dropped for medium traffic load

(b) Number of adjacencies lost for medium traffic load

(c) Number of LSU packets received for medium traffic load

Figure 3-8: Results of Medium CBR Traffic Load (25Mbits/sec), RDI=4 seconds Test Case

(a) Number of control packets dropped for light traffic load

(b) Number of adjacencies lost for light Traffic load

(c) Number of LSU packets received for light traffic load

Figure 3-9: Results of Light CBR Traffic Load (15Mbits/sec), RDI=4 seconds Test Case

Figure 3-10: Buffer Usage for Merging Node ‘M’ with Light (15Mbps/sec), Medium (25Mbps/sec) and High (50Mbps/sec) CBR Traffic Flows with RDI=4 seconds

3.2.3 EXPERIMENT 2: MERGING FTP TRAFFIC FLOWS

In this experiment, we investigate the performance of the standard OSPF protocol when the traffic flows consist of FTP packets. As opposed to the CBR traffic considered in the preceding section, the FTP traffic is bursty in nature. The packet and request inter-arrival times are random and can be modeled by exponential distribution. We used the FTP model provided by *Opnet Modeler* and configure the parameters to create different network congestion scenarios.

Fig. 3-11 shows the simulation topology employed in the experiment. All the links are PPP-DS1 supporting a maximum bandwidth of 1.544Mb/second. Users in LAN1 and LAN2 request files located at the FTP server and the FTP Server initiates the downloading of the files upon receiving the requests. The users in LAN1 make requests from time=200 seconds to time=1300 seconds whereas users in LAN2 make requests from time=500 seconds until time=1000 seconds. During the experiment, we vary the

network congestion level by varying the file size located at the FTP server. Table 3-2 summarizes the parameters configured to create different network congestion scenarios.

Figure 3-11: Simulation Network Topology for FTP Traffic Flows

FTP Traffic Type	Request Inter-arrival Time λ (sec)	File Size (Kbytes)
Heavy	50	500
Medium	50	250
Light	50	100

Table 3-2: Configuration of FTP Traffic Flows

To analyze the performance of the standard OSPF protocol with FTP traffic, we employ the same technique used in the preceding section with the CBR traffic. First we keep the FTP traffic rate constant at heavy load and vary the value of RDI to find the optimal RDI value. Then, we keep the RDI value constant at 4 seconds and vary the FTP load between heavy, medium and light traffic loads.

3.2.3.1 Fixed FTP Traffic Load, Variable Router Dead Interval

In this experiment, the FTP traffic load is fixed to heavy traffic load to simulate heavy network congestion and the simulation is run with different RDI values until the optimal RDI value is found. We run the simulations with the RDI values in the multiple of 4 starting with a value of 4 seconds. We terminate the simulations when an optimal RDI value has been found as we know from the results of the CBR traffic that the simulations with any RDI values higher than the optimal RDI value show similar results as the simulation results with the optimal RDI value.

Figures 3-12 to 3-14 show the results of varying the RDI value on the heavy FTP traffic load. We notice that the case of RDI=4 seconds resulted in 58 lost adjacencies as compared to 9 lost adjacencies when the RDI value is increased to 8 seconds. When the RDI value is increased to 12 seconds, no adjacency losses are reported and thus 12 seconds is the optimal RDI value for this system operating under heavy FTP load.

Each lost adjacency and later discovery is flooded in the network as an update. The network is considered unstable from the time when an adjacency is lost until the time the network is converged and routing failures may occur during this transient time. To investigate the relationship between the number of lost adjacencies and the stability of the network, we plotted the IP traffic received (packets/second) graphs for LAN1 and LAN2

for the different RDI values. We notice that for any RDI value from time=200 seconds to time=500 seconds, IP traffic is received on LAN1 at a constant rate of 500 packets/seconds. This is because there is minimal congestion in the network from time=200 seconds to time=500 seconds and no adjacency loss is reported during this time for any RDI value. As such, the network stayed stable from time=200 seconds until time=500 seconds and no routing failure occurred during this time resulting in uninterrupted data transfer.

From time=500 seconds to time=1000 seconds, users in both LAN1 and LAN2 are requesting files from the FTP server that builds congestion in the network. As we can see from the adjacency loss graph, depending on the RDI value, adjacencies are lost during this time. For RDI=4 seconds, there are 23 adjacency losses from time=500 seconds to time=1000 seconds, 6 adjacency losses for RDI=8 seconds and no adjacency losses for RDI=12 seconds. The number of lost adjacencies directly impacts the stability of the network which can be seen from the corresponding IP traffic received graphs for LAN1 and LAN2. We notice that with RDI = 4 seconds, network is unstable from time to time due to high number of adjacency losses. The network instability causes routing failures that resulted in IP traffic being dropped and we see large variations in the IP packet received graphs for LAN1 and LAN2 as shown in Fig 3-12(b) and Fig. 3-12(c) respectively. For RDI=8 seconds case with fewer adjacency losses, network is more stable than the RDI=4 seconds case. We notice only one major variation in the IP traffic received (packets/seconds) on LAN1 and LAN2 as shown in Fig. 3-13(b) and Fig. 3-13(c) respectively. For RDI=12 seconds, there are no adjacency loss and thus the IP traffic received graphs for LAN1 and LAN2 show no variations. The IP traffic is

received continuously on both LAN1 and LAN2 as shown in Fig. 3-14(b) and Fig. 3-14(c) respectively, depicting a more stable network environment.

From time=1000 seconds to time=1500 seconds, only users in LAN1 are making requests for files from the FTP server. However, due to transfer backlog created during the high request time between time=500 seconds to time=1000 seconds, we notice that heavy network congestion persists for some time. Adjacencies are continued to be lost for RDI=4 seconds, though at a lower rate. We notice that the lower adjacency lost rate improves the network stability and thus IP traffic is received on LAN1 and LAN2 at a more constant rate. We only see one traffic drop for RDI=4 seconds case and no drop for higher RDI values cases due to no adjacency loss. Notice that traffic is received on both LANs until the end of simulation due to file transfer backlog even though no requests are made after time=1300 seconds.

(a) Number of adjacencies lost

(b) IP traffic received (packets/sec) on LAN1

(c) IP traffic received (packets/sec) on LAN2

Figure 3-12: Results of Heavy FTP Traffic Load (file size=500KB, $\lambda=50$), RDI=4 seconds Test Case

(a) Number of adjacencies lost

(b) IP traffic received (packets/sec) on LAN1

(c) IP traffic received (packets/sec) on LAN2

Figure 3-13: Results of Heavy FTP Traffic Load (file size=500KB, $\lambda=50$), RDI=8 seconds Test Case

(a) Number of adjacencies lost

(b) IP traffic received (packets/sec) on LAN1

(c) IP traffic received (packets/sec) on LAN2

Figure 3-14: Results of Heavy FTP Traffic Load (file size=500KB, $\lambda=50$), RDI=12 seconds Test Case

3.2.3.2 Variable FTP Traffic Load, Fixed Router Dead Interval

In this experiment, we fix the RDI value to 4 seconds and vary the network congestion level by varying the file size located at the FTP server. We experimented with medium and light FTP traffic flows and compared the results with the results of the heavy FTP traffic given in Fig. 3-12. Fig. 3-15 and Fig. 3-16 show the results of applying medium and light traffic loads respectively on the performance of standard OSPF protocol when the RDI value is fixed at 4 seconds. We notice that for heavy traffic load, there are 58 lost adjacencies as compared to 24 and 2 lost adjacencies for medium and light traffic loads respectively. As such, RDI=4 seconds is not an optimal RDI value for any of the considered FTP traffic flows.

When we compare the IP traffic received pattern on LAN1 and LAN2, we observe that the network stability deteriorates with the increase in number of lost adjacencies. There are sudden drops in the IP traffic received for both LAN1 and LAN2 when there is high number of adjacency losses. For heavy traffic load, we see that files are received at the LANs until time=1500 seconds due to request backlogging at the server even though no requests are made from LAN2 after time=1000 seconds and from LAN1 after time=1300 seconds. For the medium traffic load, the request backlog is lower than the heavy traffic load due to the smaller file size facilitating faster transfers. For light FTP traffic load, the file size is the smallest and there is no backlog created after the requests are over. There is no traffic flow to LAN1 after time=1300 seconds and to LAN2 after time=1000 seconds as shown in Fig. 3-16(b) and Fig. 3-16(c) respectively. Furthermore, the network is stable all the time for the light traffic load with no sudden traffic drop as are seen in the corresponding graphs with the heavy and medium traffic loads.

(a) Number of adjacencies lost

(b) IP traffic received (packets/sec) on LAN1

(c) IP traffic received (packets/sec) on LAN2

Figure 3-15: Results of Medium FTP Traffic Load (file size=250KB, $\lambda=50$), RDI=4 seconds Test Case

(a) Number of adjacencies lost

(b) IP traffic received (packets/sec) on LAN1

(c) IP traffic received (packets/sec) on LAN2

Figure 3-16: Results of Light FTP Traffic Load (file size=500KB, $\lambda=50$), RDI=4 seconds Test Case

3.3 CHAPTER SUMMARY

In this chapter, we investigated the performance of the standard OSPF routing protocol in various network congestion scenarios. The intent of the experiments is to prove that the performance of the protocol varies significantly based on the traffic load applied on the network. We experimented with deterministic CBR traffic that closely models the real-time streaming UDP traffic as well as with random and bursty FTP traffic flows. The simulation results clearly demonstrated that no single RDI value is optimal for different network congestion scenarios. For networks with predictable traffic load, we can find the optimal RDI value through simulations or by analyzing the history and can use that value during the network operation. However, for systems with unpredictable and frequently varying traffic pattern, such as the current Internet network traffic, making the RDI value tunable based on the network congestion level is a promising option, which is discussed in the next chapter.

4 TUNABLE OSPF PROTOCOL

In the preceding chapter, we investigated the performance of the standard OSPF protocol in various network congestion scenarios and found that no single router dead interval (RDI) value is optimal for all the scenarios. In this chapter, we describe an algorithm to tune the RDI value based on the network congestion level. As the congestion varies, the algorithm dynamically modifies the RDI value to ensure optimum OSPF performance for all network congestion levels.

4.1 ALGORITHM

In this section, we describe an algorithm to dynamically tune the RDI value based on the network congestion level for a point-to-point OSPF network. An OSPF network stores the hello interval and the RDI locally at each interface and the OSPF standard requires that these intervals must be the same on both ends of the point-to-point link. An OSPF hello packet sent from an interface contains the hello interval and the RDI of the sending interface. The receiving interface compares these intervals contained in the packet with the interval values stored locally. In case of any discrepancy, the hello packet is dropped. As such, any change in the RDI value must be agreed upon, propagated and installed at both connecting interfaces of a point-to-point link.

To maintain consistency in the RDI values, a tuning RDI algorithm needs to consider and incorporate asymmetric traffic flows in the opposite directions, and ensure that the RDI value is changed only after considering the network congestion of both sides of the link. Each link in a point-to-point OSPF network is independent and thus the RDI values of the two different links can be different. However as stated earlier, the RDI values at the opposite interfaces of a same link must be the same. To facilitate the consistency, we introduce the notion of master and slave in the connecting interfaces. The designation can be made simply by choosing the interface with a higher IP address as the master. Any update in the RDI value will be done only by the master interface. A slave, on the other hand, can suggest a new RDI value to the master and if the request is accepted, the master propagates the new value at both ends of the link.

Tuning the RDI value based on the network congestion level for a point-to-point OSPF network consists of the following steps:

- Detecting the network congestion level at both connecting interfaces of all the point-to-point links in the OSPF network
- Determining the optimal RDI value for the network congestion level, detected in step 1, for each interface of all the links in the OSPF network
- Determining the state of the connecting interface for each link
- Determining whether to increase or decrease the current RDI value by considering the state of both the local and the connecting interface
- Propagating and installing the RDI value at both ends of the link

The steps involved in the tunable RDI algorithm are discussed in detail in the following sections:

4.1.1 DETECTING NETWORK CONGESTION

To determine the network congestion level, we maintain a history of the last twenty hello packet arrivals at each interface. Based on the history information, we estimate the number of hello packets sent by the sending interface and calculate the hello packet drop rate. i.e.

Number of hello packets received: $H_r = 20$

Number of hello packets sent: $H_s = (t_2 - t_1) / \text{hello interval} \dots\dots\dots (1)$

Hello packet drop rate = $1 - (H_r / H_s) \dots\dots\dots (2)$

Where t_1 corresponds to the time when the first hello packet in the current sequence of twenty hello packets is received and t_2 corresponds to the arrival time of the twentieth hello packet. In our implementation, we divide the network congestion into five states (no congestion, low congestion, medium congestion, high congestion and very high congestion) based on the hello packet drop rate. For each congestion level, a specific value of RDI is defined. Upon arrival of every twentieth hello packet at an interface, we calculate the hello packet drop rate and determine the desired RDI value from the look-up table given in Table 4-1. The current RDI value is then compared with the desired RDI value and the current value is incremented or decremented, as and if required, by one hello interval. These desired RDI values act as upper and lower bounds

to prevent the RDI value grow or drop unboundedly. This approach of maintaining a history of hello packets to determine the network congestion level and incrementing or decrementing the RDI value by one hello interval at a time circumvents abrupt changes in the RDI value and thus, facilitates network stability due to the following reasons:

- The network congestion level is detected by the history of last 20 hello packets and thus, any short-lived traffic variations average out and their effect is mostly mitigated
- The RDI value is only changed by one hello interval once after every cycle of 20 hello packets. As such, a traffic burst can only change the RDI value, at the most, by one hello interval after every cycle of 20 hello packets

Hello Packet Drop Rate	Network Congestion State	Desired RDI
[0, 0.1]	No Congestion	6
(0.1, 0.25]	Low Congestion	8
(0.25, 0.5]	Medium Congestion	12
(0.5, 0.75]	High Congestion	16
(0.75, 1]	Very High Congestion	20

Table 4-1: Network Congestion Lookup List

4.1.2 PROPAGATING RDI VALUES

Hello packets are used for the propagation of new RDI values. An OSPF hello packet contains the hello interval and the RDI of the originating interface. In the current OSPF standard, these values contained in the hello packet must match the values stored locally at the receiving interface and the hello packet is discarded in case of any discrepancy. In tunable RDI algorithm, any new RDI value from the master or suggested RDI value from the slave is contained in the hello packet. The receiving interface recognizes that the RDI contained in the packet is different from the local RDI and runs the tunable RDI algorithm. This approach of using hello packets for the propagation of new RDI values offers the following advantages:

- Allows the implementation of the tunable RDI algorithm without modifying the existing OSPF packet formats
- Allows the implementation of tunable RDI algorithm without adding any extra communication overhead
- Facilitates backward compatibility with the standard OSPF protocol by simply ‘turning off’ the tunable RDI switch

4.1.3 UPDATING RDI VALUES

An increase in the RDI value requires signaling from only one interface to ensure that the RDI is increased when either interface is experiencing congestion. If the master interface determines that the local RDI value is less than the desired value, the master increments the local RDI value by one hello interval and sends the new value to the slave

in the next hello packet. The slave updates its local RDI value upon receiving that hello packet. If the slave is experiencing congestion and wants to increase the RDI value, the slave sends a RDI value incremented by one hello interval in the next hello packet destined for the master. The slave however, does not modify its local RDI value. The master upon receiving the hello packet realizes that slave is requesting to increase the RDI value and accepts the request unconditionally. The master then increments the local RDI value by one hello interval and sends the new value to the slave in the next hello packet resulting in the slave updating its RDI value.

Decreasing the RDI value requires acceptance from both interfaces to ensure that the RDI value is decreased only if both interfaces are experiencing low congestion. As such, only one interface needs to initiate the request while the other interface, upon receiving the request, will determine if it is feasible to decrease the RDI value. In our implementation, the slave initiates the decrease RDI request. When the slave determines that the current RDI value is higher than the desired one, the slave sends a RDI value decremented by one hello interval in the next hello packet destined for the master. Upon receiving the request, the master realizes that the packet contains a decrease RDI request and consults its history to determine the possibility of reducing the RDI value. If the desired RDI value at the master is also less than the current RDI value, the request is approved. The master then lowers its local RDI value by one hello interval and sends the new value to the slave in the next hello packet resulting in the slave reducing its RDI value. However, if the desired RDI value at the master is higher or equal to the current RDI value, the slave request is ignored.

If the hello packet destined for the master containing the slave suggestion is dropped, the slave will never find out that the master has not received the suggestion. The slave will presume that the request is rejected by the master. To handle this situation in our implementation, the slave sends the suggested RDI value to the master in up to three consecutive hello packets and if still the slave has not received the desired RDI value from the master, the request is presumed to be rejected.

The pseudo code and the flowchart for the tunable RDI algorithm, when a hello packet is received at an interface, are given in Fig. 4.1 and Fig. 4.2 respectively.

Event: hello packet received

```
/* Parse hello packet and extract RDI value from it (msg.rdi) */

If interface.rdi  $\neq$  msg.rdi          /* msg.rdi differs from locally stored RDI value */
    If interface is a slave
        interface.rdi=msg.rdi          /* Slave accepts new value from master */
    Else if interface is a master      /* msg.rdi is a suggestion from slave */
        If msg.rdi > interface.rdi    /* Suggestion is to increase RDI value */
            interface.rdi = msg.rdi    /* Always accept increase RDI request */
        Else                          /* Suggestion is to decrease RDI value */
            /* Determine desired RDI value for current congestion from lookup table */
            If desired_rdi < interface.rdi /* Congestion level has decreased */
                interface.rdi = msg.rdi /* Accept the decrease RDI request */
            End if
        End if
    End if
End if

/* Check if the RDI value can be increased or decreased. The adjacent interface will
notice any change in the next hello packet */

/* Calculate packet drop rate and find desired RDI value from the lookup table */
If interface.rdi < desired_rdi        /* Congestion level has increased */
    If interface is a master          /* Master increases its RDI value */
        interface.rdi = interface.rdi + interface.hello_interval
    Else if interface is a slave      /* Slave suggests a higher RDI value*/
        interface.suggested_rdi = interface.rdi + interface.hello_interval
    End if
Else if interface.rdi > desired_rdi    /* Congestion level has decreased */
    If interface is a slave          /* Slave suggests a lower RDI value */
        interface.suggested_rdi = interface.rdi - interface.hello_interval
    End if
End if
```

Figure 4-1: Tunable RDI OSPF Algorithm Pseudo Code

Figure 4-2: Tunable RDI OSPF Algorithm Flowchart

4.2 DISCUSSION AND ALTERNATE STRATEGIES

In the stated algorithm and in our implementation, we keep a history of hello packets to determine the network congestion state which delays the algorithm reaction time until the completion of each history cycle of 20 hello packets. An alternative approach to determine the network congestion state is to monitor average router queue length at each interface which will enable more timely detection of any change in the network congestion state and thus, facilitate lower algorithm response time.

It is commonly believed that adaptive routing strategy is unnecessary, arguing that most networks currently in operation are under-loaded by design. However, a recent study at a major service provider showed spikes of over 90% load on some links over periods of many minutes and spikes of over 50% load over periods of hours [17].

Some modern routers employ priority scheduling and queuing for the control packets [18] that alleviate the problem of spurious adjacency loss decisions by giving hello packets high priority during the network congestion. However, most of the currently deployed routers do not support such differentiated treatment and mainly rely on single FIFO queuing for both data and control traffic [19].

As a proof-of-concept, we have only implemented the tunable RDI algorithm in the tunable OSPF protocol. However, the algorithm can be readily extended to also control the frequency of hello packets. If the network is congested, the hello interval can be increased to lessen the transmission rate of the hello packets and thus, avoid worsening the network congestion. On the other hand, if the network is lightly loaded, the hello interval can be decreased to expedite real failure detections in the network.

5 TUNABLE OSPF PROTOCOL

PERFORMANCE EVALUATION

In this chapter, we present results of the performance evaluation of the tunable OSPF protocol. We developed a number of test cases of varying traffic patterns and different traffic types, and analyzed and compared the performance of the tunable and standard OSPF protocols. Furthermore, the effect of data packet size and router buffer size on the performance of the protocols is also investigated.

5.1 EVALUATION STRATEGY

To analyze and compare the performance of the tunable and standard protocols, we employ the method of '*Correlated Sampling*'. In correlated sampling, the same set of random numbers is used to simulate all the systems under study. Thus, for each replication r , the outputs of all the systems are correlated with each other. The intent of using correlated sampling is to induce a positive correlation between the outputs of each system (for each r), and thus to achieve a variance reduction in the point estimator of mean difference [20].

To achieve a statistically accuracy, each experiment is repeated 30 times with different seeds for the random number generator and the results are reported with 95% confidence interval (CI) shown with bars around the average.

5.2 EVALUATION CRITERIA

The performance of the tunable and standard OSPF protocols is evaluated in terms of number of lost adjacencies and number of total updates sent in the network by the two protocols during the course of the simulation. As there is no real node or link failure simulated in the network, all adjacencies lost decisions are spurious, and thus the protocol reporting a lower number of lost adjacencies is considered superior. Each spurious adjacency loss (and rediscovery) is flooded in the network domain as an update, and results in excessive bandwidth usage and routing instabilities. Each node, upon receiving the update, removes the failed node from its routing table in case of a failure update or adds the new node in its routing table in case of discovery update, and re-runs the SPF algorithm. This results inconsistent network view on the nodes until the network is converged, and excessive bandwidth and CPU consumption. As such, the protocol flooding lower number of updates in the network is considered superior.

5.3 SIMULATION TEST CASES

We developed two sets of test cases: (i) simulated hello packet drop and (ii) congestion-based hello packet drop. In the former case, a number of hello packets arriving at an interface are explicitly dropped, with a known probability, to simulate network congestion in a controlled environment. For example, a network congestion of

10% at an interface implies that 10% of the hello packets arriving at that interface selected randomly will be dropped. In the congestion-based hello packet drop test case, we generate data traffic of increasing magnitude from a traffic source and observe the behavior of the tunable and standard OSPF protocols when both the data and control traffic flows are assigned equal priority.

We run each simulation for 1500 seconds and use a batch size of 100 seconds i.e. for every 100 seconds, we count the number of adjacencies lost and the number of updates sent by the standard and tunable OSPF protocols in that particular batch. The hello interval and the RDI are configured to 2 seconds and 8 seconds respectively as recommended in [8] to expedite real failure detections in a network.

5.3.1 SIMULATED HELLO PACKET DROP TEST CASES

In simulated hello packet drop test cases, we explicitly drop a predefined percentage of hello packets arriving at an interface to simulate network congestion in a controlled environment in the absence of any data traffic.

The simulation network topology is shown in Fig. 5-1 which exhibits a two nodes point-to-point OSPF network. As each link in a point-to-point OSPF network is independent, this simple topology is deemed sufficient for simulation in a controlled environment. The simulation is run in the absence of any data traffic and a predefined percentage of OSPF hello packets received at an interface are dropped explicitly to simulate network congestion. The tunable RDI algorithm classifies Node B, which has a higher IP address, as the master while Node A with lower IP address becomes the slave.

Figure 5-1: Network Topology for Simulated Hello Packet Drop Test Cases

5.3.1.1 Gradually Increasing Network Congestion

In this test case, we start the simulation with no congestion (0% hello packet drop) and increase the hello packet drop rate by 10% every 100 seconds on the master interface until the packet drop rate reaches 70% at time 700 seconds as shown in Fig. 5-2(a). The congestion level at the master is then kept at 70% until the end of simulation. The hello packet drop rate at the slave interface is kept at 0% throughout the simulation.

Fig. 5-2(b) shows the number of adjacencies lost per batch for the tunable and standard OSPF protocols. We notice that initially until batch 3, tunable OSPF resulted in a slightly higher lost adjacency rate per batch; however after batch 3, standard OSPF resulted in a much higher adjacency loss rate. The reason for the initial slightly higher loss rate in tunable OSPF is that, in the beginning of the simulation, the hello packet drop rate of 0% allowed tunable OSPF to reduce the RDI value from 8 to 6 seconds as shown in the RDI graph of Fig. 5-2(c). As such, when the congestion developed, the tunable OSPF declared adjacency down after 3 consecutive hello packet drops as compared to 4 hello packet drops in the standard OSPF protocol. However, the tunable RDI algorithm responded to the increasing hello packet drop rate and gradually increased the RDI value and thus, resulted in lower adjacency loss rate per batch later in the simulation. The standard OSPF is oblivious to network congestion and continued to lose adjacencies at a higher rate until the end of the simulation.

(a) Hello packet drop rate at the master and slave interfaces

(b) Number of adjacencies lost per batch by tunable and standard OSPF (95% confidence interval)

(c) RDI graph for Tunable OSPF

**Figure 5-2: Simulated Hello Packet Drop Test Case 1
(Gradually Increasing Network Congestion)**

5.3.1.2 Gradually Increasing then Decreasing Network Congestion

In this test case, we gradually build congestion on the master interface, starting at 20% at time 100 seconds and increase the packet drop rate by 10% every 100 seconds until the congestion reaches 50% at time 500 seconds. The congestion is then decreased by 10% every 100 seconds until the congestion at the master reaches 10% at time 800 seconds. The congestion at the master is then kept at 10% until the end of simulation. The hello packet drop rate at the slave interface is set to 0% throughout the simulation as shown in Fig. 5-3(a).

Fig. 5-3(b) shows a graph of the number of lost adjacencies per batch by the tunable and standard OSPF protocols. We notice that standard OSPF results in higher adjacency loss rate in the first half whereas tunable OSPF resulted in slightly higher adjacency loss rate in the latter half of the simulation. The reason for the first half has already been explained in the first test case. For the latter half, the hello packet drop rate of 10% allowed tunable OSPF to decrease the RDI to 6 seconds as shown in the RDI graph of Fig. 5-3(c). With a RDI of 8 seconds for the standard OSPF, the tunable OSPF has a higher probability of declaring adjacency down in response to hello packets drop as compared to standard OSPF. However, on the other hand, the tunable OSPF will detect a real failure in 6 seconds as opposed to 8 seconds by the standard OSPF and thus, decrease the overall network convergence time.

(a) Hello packet drop rate at the master and slave interfaces

(b) Number of adjacencies Lost per batch by tunable and standard OSPF (95% confidence interval)

(c) RDI graph for Tunable OSPF

**Figure 5-3: Simulated Hello Packet Drop Test Case 2
(Gradually Increasing then Decreasing Network Congestion)**

5.3.1.3 Asymmetric Network Congestion

In this test case, we start with no congestion at the master and slave interfaces and increase the packet drop by 10% every 100 seconds on both interfaces. However after time 600 seconds, the packet drop at the slave interface is decreased by 10% every 100 seconds until the hello packet drop rate at the slave reaches 10% at time 900 seconds and then is kept at that level. On the master interface, packet drop rate keeps increasing by 10% every 100 seconds until the congestion level is raised to 80% at time 800 seconds and then is kept at that level as shown in Fig. 5-4(a).

The intent of this test case is to analyze the algorithm for asymmetric traffic flows. As the slave interface where the hello packet drop rate is decreasing will request to decrease the RDI value, the master interface with persisting high congestion will reject the decrease RDI request and keep the RDI high.

Fig. 5-4(b) shows the number of lost adjacencies per batch and Fig. 5-4(c) shows the corresponding RDI graph for the tunable RDI case. It is evident that the RDI is increased gradually and is kept high until the end of simulation. As the network congestion is decreased, the slave interface requests the master interface to decrease the RDI value but the requests are rejected by the master due to persistent high network congestion at the master interface. The adjacencies loss rate graph of Fig. 5-4(b) depicts the same pattern as in the first test case and has already been explained.

(a) Hello packet drop rate at the master and slave interfaces

(b) Number of adjacencies lost per batch by tunable and standard OSPF (95% confidence interval)

(c) RDI graph for Tunable OSPF

**Figure 5-4: Simulated Hello Packet Drop Test Case 3
(Asymmetric Network Congestion)**

5.3.2 CONGESTION BASED HELLO PACKET DROP WITH MERGING CBR TRAFFIC FLOWS

In congestion-based hello packet drop test cases, we analyze the performance of the standard and tunable OSPF protocols with increasing magnitude of CBR data traffic when both the data and control traffic flows are assigned equal priority. Furthermore, we also investigate the effect of router buffer size and data packet size on the performance of standard and tunable OSPF protocols.

5.3.2.1 Network Topology

The simulation network topology for congestion-based hello packet drop with CBR traffic flows is shown in Fig. 5-5. All the links connecting the routers are PPP-DS3 supporting a maximum bandwidth of 44.7Mbits/second. The packet processing speed at the routers is set to the link speed. Both the data and control flows are assigned equal priority and thus, share the router FIFO queue and the link bandwidth. Routers 'B' and 'D' are connecting to fixed-rate traffic sources that are generating data traffic at a constant rate of 22Mbits/second (50% of the link bandwidth) from time 100 seconds to time 1500 seconds in all the simulation scenarios. The traffic generated by a source connected to router 'A' is varied during the experiment to build the desired level of network congestion and we observe the performance of tunable and standard OSPF protocols when all the traffic flows are destined for router 'E'.

Figure 5-5: Network Topology for Congestion-based Hello Packet Drop Test Cases

5.3.2.2 Route Selection

To find an optimal route between a source and a destination, OSPF assigns weights to all the links in the network and selects the route that minimizes the aggregate weight. In our configuration, we configure the link weight inversely proportional to the supported link bandwidth as proposed in [6]. As all the links in the topology are similar and thus support equal bandwidth, each link has an equal weight. The optimal route, in this case, will be the one traversing lower number of hops.

Under normal conditions, the data traffic originating from router ‘A’ takes path ABE which we refer to from here on as the primary path. As the network resources, such as the link bandwidth and the router buffers are shared among the data and control traffic, increasing the data traffic results in control traffic being delayed or dropped. When

congestion develops in the network, the router buffers the incoming packets in the queue and once the queue is full, any further incoming packets are dropped. Persistent network congestion at an interface may force a number of consecutive hello packets to drop. The adjacent interface will presume that the neighbor router is down and initiate the recovery process, once the RDI has expired without receiving any hello packet.

In the topology shown in Fig. 5-5, router 'B' interface is a congestion point and the link between router 'B' and router 'E' is congested when the combined traffic influx from router 'A' and the traffic generator connected to router 'B' exceeds 100%. During congestion, it is possible for router 'B' to drop four consecutive hello packet destined for router 'E'. Not receiving a hello packet within the RDI from router 'B' results in router 'E' advertising router 'B' down, and the recovery process is initiated. All nodes in the network are notified about router 'B' (spurious) failure. Router 'A', upon receiving the update, removes router 'B' from its topology and re-runs the SPF algorithm. The SPF algorithm finds path ACDE as the shortest path which we refer from here on as the secondary path. The data traffic from router 'A' is thus, redirected to the secondary path.

The re-routing of the data traffic gradually alleviates the congestion at router 'B' and eventually a hello packet from router 'B' reaches router 'E' resulting in the re-discovery of router 'B'. Router 'E' sends an update to all nodes in the network notifying about the discovery of a (new) node, router 'B'. Each node upon receiving the discovery update adds router 'B' to its routing table and re-runs the SPF algorithm. The SPF algorithm running at Router 'A' again selects the path ABE as the optimal path to reach router 'E' and the data traffic is re-routed back to the primary path. This process of losing adjacency and later rediscovery and thus, switching the data traffic between the

primary and secondary paths continues as long as the network congestion persists in the network.

5.3.2.3 Traffic Overload Test

In traffic overload test, we compared the performance of the standard and tunable OSPF protocols when the traffic source connected to router 'A' is generating a CBR data traffic of 25%, 50%, 75%, 100%, 150% and 200% of the link bandwidth. The node buffer size and the data packet size are set to 1K byte and 64 bytes respectively.

Fig. 5-6 shows the results of overloading traffic on the standard and tunable OSPF protocols. Up to 50% traffic load, there is no congestion at router 'B' and thus, the two protocols behave comparably. However, the performance of the standard OSPF protocol deteriorates considerably as the traffic load is increased beyond 50%, due to congestion at router 'B'. Fig. 5-6(a) and Fig. 5-6(b) compares the number of adjacencies lost and the number of routing updates sent in the network respectively by the tunable and standard OSPF protocols. We notice that as the aggregate traffic load exceeds beyond 100%, the performance of standard OSPF protocol deteriorates considerably. The tunable OSPF protocol resulted in about 3 times less number of adjacencies losses as compared to the standard OSPF protocol when the traffic load is increased beyond 150%. Also, it is evident that the number of adjacency losses in the standard OSPF protocol increased linearly with the increase in the traffic load. The tunable OSPF, however, is adaptable to the applied traffic load and thus, the number of adjacency losses by the tunable OSPF protocol is almost the same for both the 150% and 200% traffic load cases.

Each spurious failure and rediscovery is flooded in the network as an update and thus, as expected, the number of routing updates increases with the increase in the number of lost adjacencies as shown in Fig. 5-6(b).

To demonstrate the effect of number of spurious lost adjacencies on the data traffic and network stability, Fig. 5-6(c) shows average packet drop rate for selected traffic loads with the tunable and standard OSPF protocols. The two protocols behave comparably for 50% traffic load whereas the standard OSPF dropped 15% and 25% more packets as compared to the tunable OSPF for 100% and 150% loads respectively, due to the spurious lost adjacencies and frequent rerouting, causing routing failures and network instability.

(a) Total number of adjacencies lost

(b) Total number of routing updates sent in the network

(c) Average packet drop rate for selected traffic loads

Figure 5-6: Traffic Overload Test
(buffer size=1K bytes, packet size=64 bytes, 30 replications, 95% CI)

5.3.2.4 Router Buffer Size Test

In buffer size test, we analyze the performance of the tunable and standard OSPF protocols for buffer sizes of 1K bytes, 5K bytes and 10K bytes. The CBR traffic load from router 'A' is fixed at 75% of the link bandwidth and the data packet size is set to 64 bytes.

Fig. 5-7(a) and Fig. 5-7(b) show the number of routing updates sent per batch by the tunable and standard protocols respectively as the buffer size is varied. We notice that varying the buffer size has not significantly affected the results and the averages for different buffer sizes are intersecting at many points in both the graphs. The 95% CI area around the averages is overlapping at many points in the graphs and thus, no clear distinction can be made in terms of performance.

The rationale behind the insensitivity can be seen from Fig. 5-7(c) that shows the buffer usage at router 'B' over the course of simulation. With a combined traffic influx of 125% of the link budget (50% fixed and 75% variable), router 'B' buffer usage stayed close to 100% in all the cases, irrespective of the buffer size, when the traffic source connected to router 'A' was active.

Based on these results, we conclude that the router buffer size does not play a vital role in the performance of the OSPF protocol under high network congestion.

(a) Number of routing updates per batch for tunable OSPF (microscopic view)

(b) Number of routing updates per batch for standard OSPF (microscopic view)

(c) Buffer usage at Node B

Figure 5-7: Buffer Size Test
(75% CBR load, packet size=64 bytes, 30 replications, 95% CI)

5.3.2.5 Data Packet Size Test

Fig. 5-8 shows the results of varying the data packet size on the performance of tunable and standard OSPF protocols when the CBR traffic load from router 'A' is fixed at 75% with buffer size of 10K bytes. We run the simulations with data packet size of 64 bytes, 256 bytes, 512 bytes, 1024 bytes and 1500 bytes with constant packet processing speed.

The simulation results show that the number of routing updates sent in the network decreases with the increase in the data packet size for the tunable and standard OSPF protocols as shown in Fig. 5-8(a) and Fig. 5-8(b) respectively. The rationale for the decrease is that, keeping the traffic rate constant while increasing the packet size requires transmitting fewer numbers of packets per second in the network. For example, for a fixed traffic rate of 34 Mbits/second (75% of PPP-DS3 link bandwidth), 64 bytes packet requires a packet transmission rate of 66,400 packet/second as compared to 16,600 packets/second for 256 bytes packet size. As the packet processing speed is constant, lowering the packet transmission rate results in lesser number of packets in the network, which decreases the network congestion level.

The phenomenon of the decrease in the network congestion level with increase in the data packet size can be verified from the router 'B' buffer usage graph given in Fig. 5-8(c). The graph shows 100% usage for 64 bytes packet, 90% usage for 256 bytes packet, 80% usage for 512 bytes packet, 60% average usage for 1024 bytes packet and 50% usage for 1500 bytes packet, when the traffic source connected to router 'A' is active.

(a) Number of routing updates per batch for tunable OSPF (microscopic view)

(b) Number of routing updates per batch for standard OSPF (microscopic view)

(c) Buffer usage at Node B

**Figure 5-8: Data Packet Size Test with Constant Packet Processing Speed
(75% CBR load, buffer size=10K bytes, 30 replications, 95% CI)**

To compare the relative performance of the tunable and standard OSPF protocols, Fig. 5-9 shows the graph of the total number of routing updates sent in the network by the two protocols. It is evident that number of routing updates sent in the network decreases with the increase in the data packet size. There is no control packet drop in the simulations with the bigger packet sizes due to under usage of the Node B buffer and consequently, the two protocols behave comparably. However for 64 bytes packet size, the buffer overflow at Node ‘B’ causes control packets to drop and the tunable OSPF resulted in 28% improvement over the standard OSPF in terms of total number of routing updates sent in the network.

Figure 5-9: Total Number of Routing Updates Sent in the Network in Data Packet Size Test Case (75% CBR load, buffer size=10K bytes, 30 replications, 95% CI)

5.3.3 COMPARISON OF STANDARD AND TUNABLE OSPF WITH HEAVY FTP LOAD

In this section, we compare the performance of the standard and tunable OSPF with heavy FTP traffic load. We employ the network topology and FTP traffic configurations of section 3.2.3 where we investigated the performance of the standard OSPF protocol in heavy FTP traffic load. We now revisit the same test case and compare the performance of the standard and tunable OSPF protocols. To recall the test case, the topology of the network is shown again in Fig. 5-10. Users in LAN1 and LAN2 request files located at the FTP server. The size of each file located at the server is 500K bytes whereas the user requests are made with an inter-arrival time distributed exponentially with $\lambda=50$. The requests from LAN1 start at time=200 seconds and continue until

time=1300 seconds whereas users in LAN2 requests files from time=500 seconds until time=1000 seconds.

The RDI value is set to 8 seconds (4 times the hello interval) for the standard OSPF protocol whereas the initial RDI value is configured to 8 seconds for the tunable OSPF protocol.

**Figure 5-10: Simulation Network Topology for Comparing Tunable and Standard OSPF
Performance with Heavy FTP Traffic Flows (file size=500KB, $\lambda=50$)**

Fig. 5-11 compares the performance of the standard and tunable OSPF protocols with the heavy FTP traffic load. We see that the tunable OSPF protocol resulted in 3 lost

adjacencies as compared to 9 lost adjacencies by the standard OSPF protocol. In standard OSPF protocol, most of the adjacencies are lost between time=500 seconds to time=1000 seconds when the users in LAN1 and LAN2 are making simultaneous requests. Both the standard and tunable OSPF protocols lost their first adjacency at time=600 seconds due to increase in the network congestion. However, the tunable OSPF lost its second adjacency at time=1270 seconds. The standard OSPF protocol lost 5 more adjacencies until time=1000 seconds. The reason for the difference is that the tunable OSPF protocol algorithm adjusted to the increase in the network congestion level and accordingly increased the RDI value. As such, there are no more adjacency losses until the high network congestion level persisted. The standard OSPF protocol is oblivious to the network congestion and continued losing adjacencies at regular intervals during time=500 seconds until time=1000 seconds.

After users in LAN2 stopped making file requests at time=1000 seconds, network congestion level starting decreasing. The tunable RDI algorithm reacted to change in the congestion level and gradually lowers the RDI value. From time=1000 seconds to time=1500 seconds, the standard OSPF protocol lost 3 more adjacencies as opposed to 2 more adjacencies by the tunable OSPF protocol. The adjacency losses by the tunable OSPF protocol are caused by the varying network congestion level. With the decreasing network congestion level, the tunable RDI algorithm gradually decreases the RDI value. As the RDI value is decreased, it becomes more likely to lose adjacency due to delayed or dropped hello packets.

When we compare the IP traffic received (packets/seconds) by the users in LAN1 and LAN2 with the standard and tunable OSPF protocol as given in Fig. 5-11(b) and 5-

11(c) respectively, we notice that the tunable OSPF protocol provides a more stable network operation. Due to 6 adjacency losses by the standard OSPF protocol between time=500 seconds until time=1000 seconds, we notice a sudden drop in the IP traffic received by both LAN1 and LAN2 at around time=800 seconds. The reason for the drop is that losing a number of adjacencies in a short interval results in inconsistent network view at the nodes which leads to routing failures and consequently data traffic being dropped. There is no such drop in the received traffic when the tunable OSPF is the employed routing protocol. As such, the deployment of tunable OSPF protocol facilitates a more stable network environment.

When we analyze the IP traffic received pattern from time=1000 seconds to time=1500 seconds, we note that both protocols are showing comparable performance. However we see that in the case of the tunable OSPF protocol, the file transfer finished at time=1425 seconds whereas the file transfer with the standard OSPF protocol continued until the end of simulation at time=1500 seconds. The reason for the early finish by the tunable OSPF protocol is that tunable OSPF protocol provided a more stable network environment with no interruptions and routing failures. As such, the file transfer completed early as compared to the standard OSPF protocol where the routing failures resulted in dropped packets. The dropped packets need to be retransmitted after the network has stabilized that resulted in longer overall file transfer time.

(a) Number of adjacencies lost with standard and tunable OSPF protocols

(b) IP traffic received by LAN1 by standard and tunable OSPF protocols

(c) IP traffic received by LAN2 by standard and tunable OSPF protocols

Figure 5-11: Comparison of Standard and Tunable OSPF Protocols with Heavy FTP Traffic Load
(file size=500KB, $\lambda=50$)

5.4 CHAPTER SUMMARY

In this chapter, we investigated and compared the performance of the standard and proposed tunable OSPF routing protocols. We first compared the performance of the two protocols in a controlled test environment where the hello packets are explicitly dropped to simulate network congestion. We proved that the performance of the standard OSPF protocol deteriorates considerably with the increase in the network congestion. The tunable OSPF protocol, on the other hand, adjusts to the network congestion level and thus, facilitates a more stable routing operation.

We also compared the performance of the two protocols in the presence of data traffic of varying magnitudes when the network resources are shared between the data and control traffic. We experimented with CBR traffic flow that closely models the real-time UDP streaming traffic and with random FTP traffic flows. We found that the standard and proposed tunable OSPF protocols behaved comparably in the lightly-loaded network scenarios whereas the tunable OSPF protocol outperformed the standard OSPF protocol in the heavy network congestion cases. We observed that the proposed tunable protocol is most beneficial for gradually varying network congestion environment.

In the next chapter, we will look at the salient characteristics of the current and emerging IP networks and discuss the feasibility and viability of the proposed tunable OSPF protocol in these networks.

6 APPLICATIONS OF TUNABLE OSPF PROTOCOL

In this chapter, we explore the applicability and viability of the proposed tunable OSPF protocol in the current and emerging IP networks. We study the salient features and traffic patterns followed by the existing and emerging IP networks and investigate the feasibility of deploying the tunable OSPF routing protocol in these networks.

6.1 INTERNET NETWORK

Tunable RDI algorithm is best-suited for networks in which traffic load varies considerably and the frequency of variations is low. The algorithm works best for the traffic variations occurring in gradual steps where each step lasts in the order of minutes or hours. The gradually varying traffic allows the tunable OSPF algorithm to adapt to the network congestion level. In a frequently varying network traffic load, it is possible for the tunable OSPF algorithm to never reach the optimal RDI value and always stay in the transient state due to frequent changes in the network congestion levels.

Internet traffic is known to follow the pattern of high traffic load during regular working hours, moderate load during the evening hours to low traffic load during

nighttime. Kevin et al. [21] analyzed the Internet traffic on a MCI WorldCom's IP backbone over a period of 24 hours and 7 days. Fig. 6-1 show the results of the Internet traffic measured over a domestic United States link over a 24 hours and 7 days period. The byte volume is increased by 500% between the hours of 5AM to 10AM hitting a peak of 50Mbps/second at noon. This pattern is repeated daily, with a 20% reduction of byte traffic over weekends. A similar pattern can be seen on an international link between United States and United Kingdom as shown in Fig. 6-2. In both graphs, we observe a gradual change in the average byte volume with time that in turn, results in a gradual variation in the network congestion level, which is an ideal pattern for the operation of the tunable RDI algorithm. As such, our algorithm is suitable to work with the pattern followed by Internet traffic.

**Fig. 6-1(a) Byte volume for 1-day period
on a domestic link [21]**

**Fig. 6-1(b) Byte volume for 7-days period
on a domestic link [21]**

Fig 6-2(a) Byte volume for 1-day period on an international link [21]

Fig 6-2(b) Byte volume for 7-days period on an international link [21]

6.2 WIRELESS MESH NETWORKS

Wireless Mesh Networks (WMN) is gaining significant attention these days as a convincing candidate for next-generation Internet access networks. WMN requires minimal upfront capital investment, because the technology can be installed incrementally, one node at a time, just as needed. As more nodes are installed, the reliability and network coverage increases [22].

In conventional wireless access networks, all user nodes are required to be within the wireless transmission range of the base station which is connected to the Internet. If a node is out of range of the base station, it cannot be serviced. In WMN, however, service coverage is not limited to the direct connectivity with the base station. The central node, called a gateway, is connected to the Internet that does not require direct connectivity to all the user nodes as a base station does. Each user node in a WMN operates not only as a host, but also as a router. A node, acting as a router, may forward packets on behalf of

other nodes that are not within direct wireless transmission range of the gateway. Each node in a WMN can be stationary or comparatively mobile. The network is dynamically self-organizing and self-configuring, with the nodes in the network automatically establishing and maintaining routes among themselves.

The gateways in a WMN are added one at a time as needed. Adding more gateways increase not only the capacity of the network but also its reliability. A mesh network ensures availability of multiple paths among the nodes and the gateway. If one node fails, the traffic can be rerouted along an alternate path. Similarly, if one gateway fails, the other gateway will take over, and the system will continue to run with a slightly decreased performance. Fig. 6-3 shows an example of WMN consisting of fixed and mobile nodes.

Figure 6-3: Wireless Mesh Network

WMN relies on routing protocols, such as OSPF, to find an optimal route. These protocols are traditionally designed for wired networks and are mostly oblivious to the peculiar characteristics of a wireless network. Unlike wired links, the bandwidth of a wireless link can fluctuate frequently and considerably depending on the variation in the radio environment. Factors such as interference, fading and shadowing affect the state of the radio link. Because WLAN operate in the unlicensed spectrum, frequencies from outside networks can interfere with the deployment. Interference may include man-made noise sources, such as microwave ovens or other wireless LANs or may include line-of-sight obstructions. The signal-to-noise ratio (SNR) decreases with the deterioration in the state of the radio link and subsequently, the number of packet error increases.

To keep the SNR within acceptable limits with the deterioration in the state of the radio link, most WMN support multi-rate bandwidth based on the radio channel characteristics [23]. As the state of link deteriorates, the supported bandwidth is decreased to reduce the transmission rate, resulting in decrease in the packet error rate. For example, in 802.11b, the supported bandwidth decreases in the steps of 11Mbps/second to 5.5Mbps/second, to 2Mbps/second, and to 1Mbps/second if the condition of the radio link deteriorates. Similarly the supported bandwidth increases in steps with the improvement in the wireless link. As such, the network congestion level on a radio link in a WMN can suddenly change due to change in the supported bandwidth, even without any change in the network traffic load.

As the supported bandwidth decreases on a radio link, it becomes more likely that hello packets are dropped due to a sudden increase in the network congestion level of the

link. A standard OSPF protocol, which is oblivious to the network congestion, may start declaring adjacencies down due to delay or drop of the hello messages. The tunable OSPF protocol, on the other hand, adapts to the varying network congestion level and thus, will assure a more robust and stable routing operation.

6.3 CHAPTER SUMMARY

In this chapter, we studied the traffic pattern followed by the Internet traffic in the current and emerging IP networks and discussed the suitability of deploying tunable OSPF protocol in these networks. We showed that traditional wired IP networks follow gradually varying traffic pattern which is well-suited for the operation of tunable OSPF protocol that builds history over time and ignores sudden traffic variations. We also explained the operation of emerging Wireless Mesh Networks (WMN) that varies its supported bandwidth in response to variations in the radio environment. The tunable OSPF protocol is intuitively demonstrated to improve the stability and robustness of the WMN.

7 CONCLUSION

This chapter concludes this dissertation by summarizing the main findings and suggesting some future directions for our work.

7.1 THESIS SUMMARY

This thesis explores the idea of reducing the failure detection time and consequently, the network convergence time (NCT) of an OSPF network by reducing the hello and router dead intervals of an OSPF network. We showed that reducing the RDI value results in expediting the failure detection time but introduce the problem of spurious failure decisions during the times of network congestion. We investigated the tradeoff between network stability and decreasing in the NCT with Constant Bit Rate (CBR) traffic that closely models the real-time UDP streaming traffic and with random FTP traffic flows. We showed that the optimal RDI value, which decreases the NCT while assuring a stable network operation, varies with the network congestion level.

An algorithm is proposed for a point-to-point OSPF network to dynamically tune the RDI value based on the network congestion level. Results obtained from simulations of the proposed algorithm proved reduced numbers of spurious adjacency losses under

high network congestion as compared to standard OSPF while improving the real failure detection time in low network congestion. The algorithm may lead to higher NCT under heavy congestion but improves the stability of the network.

We also studied the effect of buffer size and data packet size on OSPF performance and found that the tunable and standard OSPF protocols are insensitive to the buffer size in high network congestion and depict similar relative performance. Decreasing the data packet size, however, increases the network congestion level when the packet processing rate is kept constant. The standard and the proposed tunable OSPF protocols behave comparably for larger data packet size due to low network congestion, whereas the tunable OSPF protocol resulted in significant performance improvement over the standard OSPF when the data packet size is decreased due to high network congestion.

We investigated and compared the performance of the standard and tunable OSPF protocol in a real-life network scenario where the users are simultaneously downloading big files from a FTP server. We noticed that employing the tunable OSPF routing protocol allowed uninterrupted data transfer and facilitated shorter download times as compared to the standard OSPF protocol.

The applicability of the proposed tunable OSPF protocol is also studied in the current and emerging IP networks. We demonstrated that the proposed tunable protocol is suitable for the gradually varying pattern followed by Internet traffic. Furthermore, it is intuitively explained that the performance and stability of the emerging Wireless Mesh Networks (WMN), which is subject to radio channel anomalies, can also be improved by employing the proposed tunable OSPF protocol.

7.2 FUTURE DIRECTIONS

7.2.1 TUNABLE OSPF IMPLEMENTATION IN A LINUX TEST-BED

A possible extension of our work is to implement and test the tunable OSPF protocol in a test-bed of Linux routers. Zebra [24] is free open-source software available under GNU General Public License that provides the implementation of many routing protocols including OSPF for Linux machines. The tunable RDI algorithm can be incorporated in the OSPF implementation provided by Zebra and the performance of the tunable and standard OSPF protocols can be analyzed and compared in a test-bed of Linux routers. The evaluation in a real test-bed will allow comparing the two protocols in more realistic traffic and packet loss models.

7.2.2 COMPARISON OF STANDARD AND TUNABLE OSPF PROTOCOLS ON WMN

In this thesis, we have discussed the performance of the tunable OSPF protocol in a WMN operating in a varying radio environment. A possible extension of this thesis is to configure a simulated WMN network and compare the performance of WMN operating in a varying radio environment with the standard and tunable OSPF routing protocols.

7.2.3 INCORPORATING SNR IN TUNABLE RDI ALGORITHM FOR WMN

A possible enhancement of the tunable RDI algorithm operating on a WMN is to estimate the SNR value and use the SNR value to predict a change in the network congestion level. As discussed in section 6-2, the supported bandwidth in a multi-rate

WMN changes in steps. The tunable OSPF gradually adapts to any change in the network congestion and may initially make spurious failure decisions.

A possible enhancement to further improve the performance of the tunable OSPF protocol operating on a WMN is to predict the change in the supported bandwidth. A history of packet errors and successful data packet arrivals can be kept at each interface and the SNR of the system can be estimated. As such, the change in the supported bandwidth can be forecasted and the RDI value can be adjusted before the actual drop in the supported bandwidth. This approach of estimating the SNR and forecasting the bandwidth change will further stabilize the operation of the WMN operating in a varying radio environment.

REFERENCES

- [1] G. Iannaccone, C. Chuah, R. Mortier, S. Bhattacharyya, and C. Diot, "Analysis of link failures in an IP backbone," Proceedings of ACM SIGCOMM Internet Measurement Workshop, November 2002
- [2] A. Markopoulou, G. Iannaccone, S. Bhattacharyya, C. Chuah, and C. Diot, "Characterization of Failures in an IP Backbone," Proceedings of IEEE INFOCOM, March 2004
- [3] S. Lee, Y. Yu, S. Nelakuditi, Z-L. Zhang, and C. N. Chuah, "Proactive vs. Reactive Approaches to Failure Resilient Routing," Proceedings of IEEE INFOCOM, March 2004
- [4] C. Labovitz, A. Ahuja, A. Abose, and F. Jahanian, "An experimental study of BGP convergence," Proceedings of ACM SIGCOMM, 2000
- [5] C. Huitema, "Routing in the Internet," Prentice Hall, Inc., Englewood Cliffs, California, 1995
- [6] Cisco Systems, "OSPF design guide," [online], <http://www.cisco.com/warp/public/104/1.html>
- [7] A. Basu and J. Riecke, "Stability issues in OSPF routing," Proceedings of ACM SIGCOMM, pp. 175-187, 2000
- [8] C. Alaettinoglu, V. Jacobson and H. Yu, "Towards millisecond IGP convergence," [online], <http://www.nanog.org/mtg-0010/ppt/cengiz.pdf> , NANOG 20, October 2000

- [9] G. Choudhury, V. Sapozhnikov, G. Ash, A. Maunder and V. Manral, "Prioritized treatment of specific OSPF packets and congestion avoidance," Internet Draft, draft-ietf-ospf-scalability-07.txt, May 2004
- [10] J. Pu, E. Manning and G. C. Shoja, "Routing reliability analysis of partially disjoint paths," in IEEE Conference on Communications, Computers, and Signal Processing, 2001, Vol. 1, August 2001, pp. 79-82
- [11] P. Narvaez, K. Siu and H. Tzeng, "Local restoration algorithm for link-state routing protocols," in Eight International Conference on Computer Communications and Networks, 1999, pp. 352-357
- [12] J. Wu, X. Lin, J. Cao and J. Weijia, "An extended fault-tolerant link-state routing protocol in the Internet," in Eighth Internal Conference on Parallel and Distributed Systems, 2001, pp. 331-337
- [13] D. Katz, "Why are we Scared of SPF? IGP Scaling and Stability", [online], <http://www.nanog.org/mtg-0206/ppt/katz.ppt>
- [14] G. Choudhury, "Prioritized Treatment of Specific OSPF Packets and Congestion Avoidance," Internet Draft, draft-ietf-ospf-scalability-08.txt, June 2004.
- [15] M. Goyal, K. Ramakrishnan, and W. Feng, "Achieving faster failure detection in OSPF networks," Proceedings of ICC 2003, Vol. 1, May 2003, pp. 296-300
- [16] A. Tanenbaum, "Computer Networks," Prentice Hall, pp. 374-375, 1996
- [17] C. Chuah and C. Diot, "A tier-1 ISP perspective: design principles & observations of routing behavior," in IPAN Workshop on Large-Scale Communication Network, 2002
- [18] V. P. Kumar, T. V. Lakshman, and D. Stiliadis, "Beyond Best Effort: Routers Architectures for the Differentiated Services of Tomorrows Internet", IEEE Communications Magazine, May 1998, pp.152-164

- [19] A. Shaikh, L. Kalampoukas, R. Dube and A. Varma, "Routing Stability in Congested Networks: Experimentation and Analysis", Proceedings of ACM SIGCOMM, Aug 2000, pp. 163-174
- [20] J. Banks, J. Carson, B. Nelson and D. Nicol, "Discrete-event system simulation," Prentice Hall, pp. 456-467, 2000
- [21] K. Thompson, G. J. Miller and R. Wilder, "Wide-area Internet traffic pattern and characteristics," IEEE network Vol. 11, Nov. 1997, pp. 10-23
- [22] J. Jun and M. Sichitiu, "The nominal capacity of wireless mesh networks," IEEE Wireless Communications, Vol. 10, Oct. 2003, pp. 8-14
- [23] A. Acharya and A. Misra, "High-performance architectures for IP-based multihop 802.11 networks," IEEE Wireless Communications, Vol. 10, Oct. 2003, pp. 22-28
- [24] <http://www.zebra.org>