

Thesis  
2001  
Samia, MI  
Spec. Coll.

**OPTIMIZATION OF  
IP MULTICAST NETWORKS PERFORMANCE**

**By**

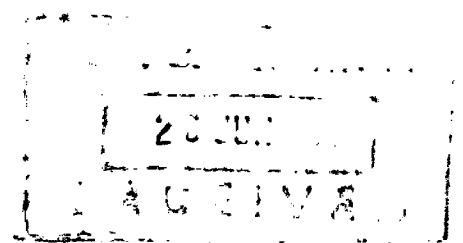
**Mireille I. SAMIA**

A Thesis

Submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science in Computer Science

Department of Computer Science  
Faculty of Natural and Applied Sciences  
Notre Dame University

June 2001

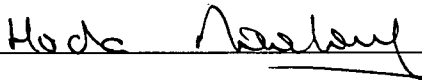


# OPTIMIZATION OF IP MULTICAST NETWORKS PERFORMANCE

by

**Mireille I. SAMIA**

Approved by



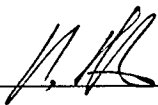
---

Hoda Maalouf: Assistant Professor of Computer Science  
Advisor



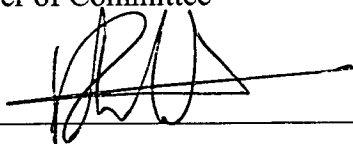
---

Fouad Chedid: Associate Professor of Computer Science. Chairperson  
Member of Committee



---

Khaldoun El-Khalidi: Assistant Professor of Computer Science  
Member of Committee



---

Ramez Maalouf: Assistant Professor of Mathematics  
Member of Committee

# **OPTIMIZATION OF IP MULTICAST NETWORKS PERFORMANCE**

**By**

**Mireille I. SAMIA**

A Thesis

Submitted in partial fulfillment  
of the requirements for the degree of  
Master of Science in Computer Science

Department of Computer Science  
Faculty of Natural and Applied Sciences  
Notre Dame University

June 2001

# Abstract

Internet Multicast service extends the Internet point-to-point unicast delivery to a multi-point delivery. Multicasting means that a datagram is sent to multiple sites at more or less the same time.

To support real-time multimedia, the minimization of delay without losing the reliability of the network is required. To achieve this goal, we have proposed to change the architecture of a multicast one-stage network to a multicast hierarchical two-stage network.

We have proved that by giving additional functionalities (such as back-up and error correction) to the intermediate sites (i.e. m-routers) in a two-stage network, we can minimize the delay. In all our analysis, packet loss and retransmission were considered.

We also proved that for a given network specification, there is an optimal number for the intermediate sites. For this optimal number of intermediate sites, the network delay drops by 35%.

# Acknowledgments

Thank you God for enlightening my way and for giving me the strength to bear what I had to and have to.

Thank you God for giving me the opportunity to express my deep appreciation to my family: my father, my mother, my sister Noelle for their patience, their prayers, their help, and their continued support, and to Bijou for *her* cuteness. I cannot imagine what life could have been without my sister Noelle.

Thank you God for adding new members to my family and letting me express my gratitude to all of them. I was deeply touched by the deep attention of Dr. Hoda Maalout, who helped me patiently and skillfully. This thesis would have not been without her constant guidance, motivation, and encouragement. I would like to thank her sincerely and respectfully. I would like also to express a special gratitude to Mrs. Rosemarie Marcos for believing in me, for her encouragement and support. I am very grateful to Dr. Boulos Sarru' for his support, and for lending me his wisdom. My many thanks also to Mr. Nabil Char for his support and his kindness.

Thank you God for letting me express my thanks to all NDU staffs, particularly all the faculty members for their kindness, their patience and their time.

Thank you God for helping me understand that 'Thank You' is much more than two words! To all who have directly or indirectly helped me in making one of my dreams comes true:

I will never forget You

and

I wish I could do the same for You!

Thank You!

# Table Of Contents

	<b>Page</b>
<b>LIST OF FIGURES</b> .....	7
<b>LIST OF ABBREVIATIONS</b> .....	8
 <b>CHAPTER</b>	
 <b>1. INTRODUCTION AND PROBLEM DEFINITION</b>	
1.1. Introduction to the General Problem .....	9
1.2. Problem Definition .....	10
1.3. Research Objectives .....	13
1.4. Thesis Organization .....	13
 <b>2. TCP/IP AND INTERNET MULTICASTING</b>	
2.1. TCP/IP .....	14
2.1.1. internet .....	14
2.1.2. Internet Protocol (IP).....	15
2.1.2.1. Purpose of IP .....	15
2.1.2.2. Internet Datagram .....	15
2.1.2.3. Internet Control Message Protocol (ICMP) .....	17
2.1.2.4. User Datagram Protocol (UDP) .....	18
2.1.2.5. Transport Control Protocol (TCP) .....	20
2.2. Internet Multicasting .....	21
2.2.1. IP Multicasting .....	21
2.2.2. Multicasting Advantages .....	22
2.2.3. Extending IP to Handle Multicast .....	23
2.2.4. Internet Multicast Backbone (MBone) .....	23
2.2.5. IP Multicast Addresses .....	24
2.2.6. Internet Group Management Protocol (IGMP) .....	25
2.2.7. Distance Vector Multicast Routing Protocol (DVMRP) .....	26
2.3. Research Motivation .....	26

---

<b>3.</b>	<b>MULTICAST TWO-STAGE NETWORKS VS. MULTICAST ONE-STAGE NETWORKS</b>	
3.1.	Assumptions Used in the Simulation.....	28
3.2.	Multicast in One-Stage Networks .....	29
3.2.1.	Architecture of Multicast One-Stage Networks.....	29
3.2.2.	Multicast One-Stage Network Analysis .....	30
3.2.3.	Multicast One-Stage Network's Simulation Results .....	31
3.2.3.1.	The arrival rate $\lambda$ is variable .....	31
3.2.3.2.	$\mu$ is variable, or $\gamma$ is variable, or $\sigma$ is variable .....	32
3.3.	Multicast in Two-Stage Networks .....	34
3.3.1.	Architecture of Multicast Two-Stage Networks .....	34
3.3.2.	Multicast Two-Stage Network Analysis .....	35
3.3.3.	Multicast Two-Stage Network's Simulation Results .....	35
3.3.3.1.	The arrival rate $\lambda$ is variable .....	36
3.3.3.2.	Intermediate Site Optimal Number .....	37
3.3.3.3.	$\mu$ is variable, or $\gamma$ is variable, or $\sigma$ is variable .....	38
3.4.	Comparison of Multicast One-Stage Networks and Multicast Two-Stage Networks.....	40
3.4.1.	Benefits of the Intermediate Sites Optimal Number .....	40
3.4.2.	Impact of the Arrival Rate $\lambda$ on the Delay .....	43
<b>4.</b>	<b>CONCLUSION</b>	
4.1.	Final Results.....	44
4.2.	Future Works.....	45
<b>APPENDIX A:</b>	<b>REFERENCES</b> .....	47
<b>APPENDIX B:</b>	<b>SIMULATION SOURCE CODE USING MATLAB</b> .....	49

# List of Figures

Figure	Page
1.1. Principle of Packet-Switching .....	10
1.2. Multicast Two-Stage Network .....	12
1.3. Multicast One-Stage Network .....	12
2.1. internet .....	14
2.2. Datagram Encapsulation .....	16
2.3. ICMP Message Encapsulation .....	18
2.4. UDP Layer .....	19
2.5. UDP Message .....	19
2.6. TCP Layer .....	20
2.7. Multicast Datagram Tunneling .....	24
2.8. Internet Multicast Backbone .....	24
2.9. Class D Address .....	25
2.10. One-Stage Topology: One-Stage Network.....	27
2.11. Hierarchical Topology: Two-Stage Network .....	27
3.1. Multicast One-Stage Network .....	30
3.2. Average Delay <sub>1</sub> Variation: from the Source of the data to the Dummy Sites .....	31
3.3. Total Delay Variation: from the Source of the data to the Destination Servers .....	32
3.4. Total Average Delay as function of $\mu$ .....	33
3.5. Total Average Delay as function of $\gamma$ .....	33
3.6. Total Average Delay as function of $\sigma$ .....	33
3.7. Multicast Two-Stage Network .....	34
3.8. Average Delay Variation: from the Source of the data to the Intermediate Sites .....	36
3.9. Total Average Delay Variation: from the Source of the data to the Destination Servers .....	37
3.10. Total Average Delay Variation as a function of Intermediate Sites .....	38
3.11. Total Average Delay vs. $\mu$ .....	39
3.12. Total Average Delay vs. $\gamma$ .....	39
3.13. Total Average Delay vs. $\sigma$ .....	39
3.14. Impact of the number of Intermediate Sites on the Delay ( $\lambda=0.9, \mu=1, \gamma=\sigma=2$ ).....	41
3.15. Impact of the number of Intermediate Sites on the Delay ( $\lambda=0.9, \mu=\gamma=\sigma=1$ ).....	41
3.16. Impact of the number of Intermediate Sites on the Delay ( $\lambda=0.9, \mu=\gamma=1, \sigma=2$ ) .....	42
3.17. Two-Stage Network vs. Multicast One-Stage Network .....	43
4.1. Remaining Destination Sites added to the Ultimate Intermediate Site C.....	45
4.2. Remaining Destination Sites added to the Intermediate Sites A, B, C .....	46



# List of Abbreviations

BER	Bit Error Rate
DVMRP	Distance Vector Multicast Routing Protocol
ICMP	Internet Control Message Protocol
IGMP	Internet Group Management Protocol
IP	Internet Protocol
LAN	Local Area Network
MBone	Internet Multicast Backbone
MTU	Maximum Transfer Unit
NAK	Negative Acknowledgement
PC	Personal Computer
PSE	Packet-Switching Exchange
sd	Session Directory
TCP	Transport Control Protocol
TTL	Time To Live
UDP	User Datagram Protocol
WAN	Wide Area Network

# **CHAPTER 1**

## **Introduction and Problem Definition**

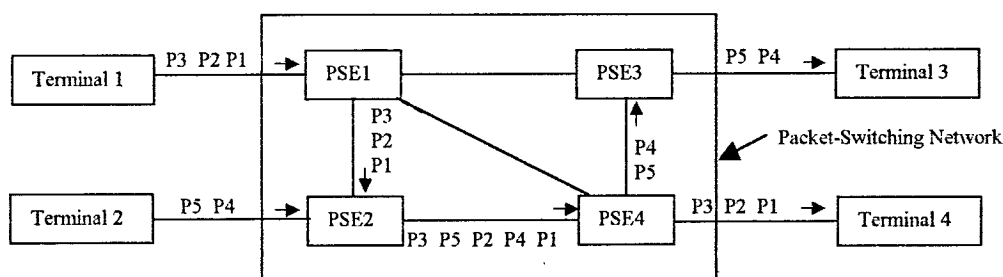
### ***1.1 INTRODUCTION TO THE GENERAL PROBLEM***

The aim of this research is to optimize the communication network performance of Internet Multicast systems with packet loss constraints. Internet is the world's largest computer network. It is an international interconnected collection of smaller networks and computers. Multicasting means that a piece of data (i.e. a packet) can be sent to multiple sites at more or less the same time, instead of having to send that packet to one site at a given time. The performance of IP communication is defined as the overall quality of service of this communication system [9]. The problem we are analyzing in this thesis is very crucial, because multicasting is used in many real-time applications such as videoconferencing, audio conferencing, etc.[8] These applications are very sensitive to packet loss or increased delay.

## 1.2 PROBLEM DEFINITION

The Internet (or TCP/IP network) is a special case of packet-switching networks. Packet-switching networks divide the data to be transferred into packets. In it, multiple communications among computers can proceed concurrently. However, as the activity increases, a packet-switching network can become overloaded, and computers using it must wait before sending other packets.

The basic principle of packet-switching is shown by (fig. 1.1)[3].



*fig. 1.1: Principle of Packet-Switching*

A packet-switching network consists of a number of packet-switching exchanges (PSE). A PSE sends the packets to their destinations using any appropriate link at the maximum available bit rate. Each terminal (e.g. PC, router) of the network is connected to the nearest PSE by a data line. To prevent long queuing delays and to ensure fast transmit time, each message is divided into packets that are transmitted separately through the network.

Terminal1 has a message to send to Terminal4, and Terminal2 has a smaller message to send to Terminal 3. The message sent by Terminal1 is divided into three packets P1, P2, and P3 before it is transmitted to PSE1. From PSE1, the packets may traverse any available link that provides a possible path to the destination (i.e. Terminal4). In fig.1.1, the packets are transmitted to PSE2. At the same time, the message sent by Terminal2 is divided into P4 and P5 that are transmitted to PSE3, then to Terminal3 [3].

If an error is detected, the corresponding packet must be retransmitted. Then, a terminal may receive packets after a substantial delay. Packet delays, loss, and duplicate make the internet service unreliable, because delivery is not guaranteed. In fact, the internet protocol IP is a connectionless service because each packet is treated independently. It is a best-effort, unreliable packet (called datagram) delivery system [6].

The notion of interactivity is important in IP multicast because it brings up live multimedia. The real world happens in real-time, where things are changing by the second. The real-time factor comes into play because it makes the Internet multicast important due to real-time multimedia communications (videoconferencing, whiteboards, etc.). Consequently, the need for reliable stream delivery is crucial. Whenever a recipient does not receive a transmitted packet, it communicates with the source of the data by sending back a negative acknowledgement (NAK) message. Retransmission is hence required. Duplicate packets are detected because each packet is assigned a unique sequence number. The receiver remembers which sequence numbers it has received so far. A packet sequence number is sent back in positive or negative acknowledgements.

A real-time traffic requires minimal delays, and low packet loss rates. High packet loss rates and large delays affect the quality of Internet multicast service. In this research, the presence of packet loss is considered. The performance of a multicast two-stage network (fig. 1.2) is studied and is compared to a multicast one-stage network (fig. 1.3). The flow of packets entering one of these networks may be with delays or losses. In a multicast two-stage network, routers are involved in the communication special process. Special services implemented in these routers (or intermediate sites) will enhance the overall network performance.

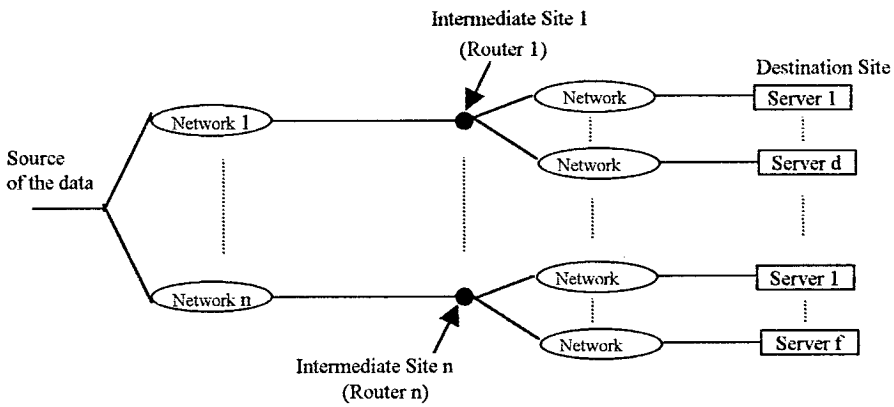


fig. 1.2: Multicast Two-Stage Network

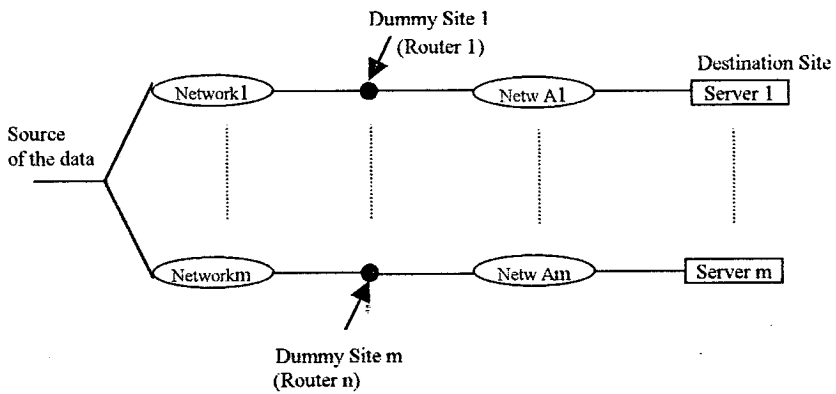


fig. 1.3: Multicast One-Stage Network

A router, also called IP gateway, connects a Local Area Network (LAN) to a Wide Area Network (WAN) (such as the Internet). It moves packets from one network to another. Only routers provide connections between physical networks in an internet [1].

### ***1.3 RESEARCH OBJECTIVES***

The work in this thesis can be divided into two main sections depending on whether the network is a multicast one-stage or a multicast two-stage. The aim of this research is to compare these two networks (as previously stated) and to optimize the performance of the multicast two-stage network by finding the optimal number of intermediate sites (or intermediate routers). In fact, we aim to prove that when we add special functions to the intermediate sites (routers), a two-stage network gives better performance than a one-stage network. Consequently, our research objectives are:

- a) the minimization of the delay without forgetting the importance of reliability in the network
- b) finding the optimum number of intermediate sites that makes two-stage better than one-stage network.

### ***1.4 THESIS ORGANIZATION***

This thesis is organized in 4 chapters. Chapter two is divided into two main sections. The first section presents a general overview of the basic concepts of TCP/IP protocol and the second section presents those of Internet Multicasting.

Chapter three is divided into 4 sections. Section 3.1 explains the general assumptions considered in our network models. Section 3.2 studies the multicast one-stage network, explains its architecture, analyzes its performance and presents the results of the related simulation. Section 3.3 studies the multicast two-stage network and its architecture. The results of the related simulation are also discussed in this section. Section 3.4 compares the two systems and shows the importance of the multicast two-stage network.

The final chapter (4) summarizes the main results of the thesis and provides possible extensions to it.

# CHAPTER 2

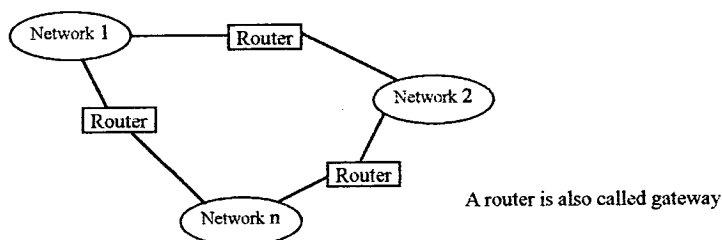
## TCP/IP and Internet Multicasting

### 2.1 TCP/IP

The internet technology permits computers to communicate independently of their physical network conventions. The rule for communication is provided by TCP/IP, which treats all networks equally. TCP/IP protocols contain the details of message formats, describe how a computer responds while receiving a message, and specify how a computer responds when errors or other abnormal conditions occur.

#### 2.1.1 internet

The interconnection of more than one network connected by a Wide Area Network (WAN) to function as one larger network is called *Internetworking*. The whole network is known as *internet* (fig. 2.1) [11].



*fig. 2.1: internet*

Internetworking gives users access to additional data and to a greater number of other sites. The internet hides the details of network hardware by using a system based on network-level interconnection. A network level interconnection provides a mechanism that delivers packets from their original source to their ultimate destination in real time. Routers (or IP gateways) transfer packets from one network to another [1].

Each machine on a TCP/IP internet is assigned a unique internet address (IP address) which is independent of the machine's hardware address and which is used in all communication with that machine. Routers use the netid (i.e. network identification) portion of an IP address to decide where to send a packet. A router connecting  $n$  networks has  $n$  distinct IP addresses, one for each network connection. If a host computer moves from one network to another, then it must change its IP address [1].

### **2.1.2 Internet Protocol (IP)**

#### **2.1.2.1 Purpose of IP:**

IP is a connectionless protocol responsible for communication between interconnected networks and for moving data between them. It chooses a path over which the data should be transferred. It includes a set of rules that characterizes how hosts and routers should process packets, how and when error control messages should be generated, and when the packets should be dropped. In other words, IP is responsible for routing datagrams between interconnected networks and for error reporting [6].

IP data unit is known as IP datagram (i.e. datagram).

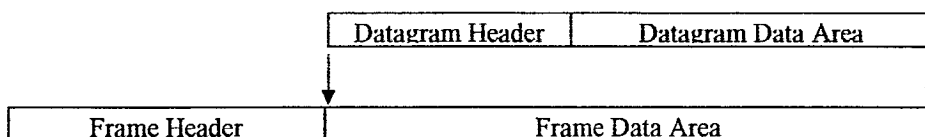
#### **2.1.2.2 Internet Datagram**

On a physical network, a frame is the unit of transfer. A frame is divided into a header and data. The header contains the physical source and destination addresses.

As seen previously, on an internet, an Internet datagram, also called IP datagram or datagram, is the basic unit of transfer. A datagram is divided into header and data. The header gives information such as the source and destination addresses. IP does not specify the format of the data area; it can be used to transport arbitrary data.



The idea of carrying a datagram in one network frame is called encapsulation. Whenever a computer sends a datagram to another, the entire datagram travels in the data portion of the network frame. Thus, the physical network considers the datagram header and datagram data area as data (fig.2.2) [1].



*fig. 2.2: Datagram Encapsulation*

The optimal case is when the entire IP datagram fits into one physical frame.

In a packet-switching network, the amount of data that can be transferred in one physical frame is fixed to a certain limit. This limit is called the maximum transfer unit (MTU). Limiting datagrams to fit the smallest possible MTU in the internet makes transfer inefficient when those datagrams pass across a network that can carry larger size frames. For this reason, TCP/IP software chooses an initial datagram size and divides larger datagrams into smaller pieces or fragments, when the datagram needs to traverse a network that has a small MTU. The process of dividing a datagram is called fragmentation. Fragmentation occurs at a router, which receives a datagram from a network with a large MTU, and must send it to a network with smaller MTU. Fragments must be reassembled to produce a copy of the original datagram. Each fragment contains a header, which is a copy of the header of the original datagram, followed by the data that can be included [1].

The datagrams are not reassembled immediately after passing across a network with small MTU. Reassembly of the fragments is produced at the ultimate destination. If a fragment is lost, then the datagram cannot be reassembled. Consequently, the receiver starts a reassembly timer as it receives the first fragment. If the reassembly timer expires before all fragments are received, then the receiver drops the received fragments. For this reason, the receiving machine starts a reassembly timer when it receives an initial fragment. If the timer expires before all fragments arrive, the receiving machine drops the received fragments.

The Threshold is a numeric value of which the minimum Time To Live (TTL) field is specified. It is required to allow the datagram to pass. Time To Live (TTL) specifies the maximum life (i.e. maximum time), in seconds, the datagram could remain in the internet system. The maximum life of a datagram is specified by the sender. Each router along the path from source to destination decrements the TTL by one when it receives the datagram. As time passes, the datagram TTL passing through routers and hosts is decremented. Whenever TTL expires, the datagram is dropped. When the routers are overloaded, long delays are produced. For this reason, each router records the datagram arrival time, and decrements its TTL by the number of seconds the datagram remains inside the router waiting for service [1].

When a host receives an IP datagram, the network interface software delivers it to the IP software. The host is required to discard the datagram whenever the datagram's IP destination address is not equal to the host's IP address [1].

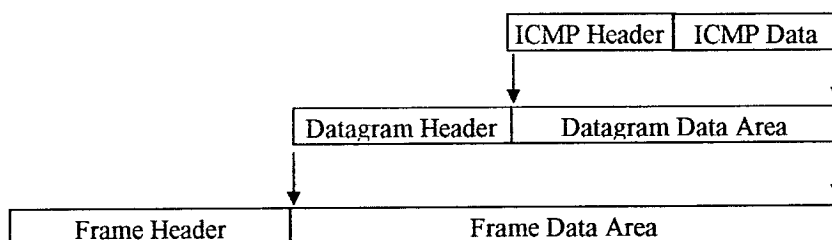
When an IP datagram arrives at a router, the network interface software delivers it to the IP software. If the datagram destination IP address is equal to the router's IP address, then the IP software passes the datagram for processing. Otherwise, the datagram must travel further. IP software uses the routing table whenever it has to decide how to forward a datagram. Routing tables store information about possible destination and how to reach them. Because hosts and routers route datagrams, both have IP routing tables, which are consulted whenever a datagram must be sent [1].

### **2.1.2.3 Internet Control Message Protocol (ICMP)**

If a router cannot deliver a datagram, then it must inform the original source to avoid or to correct the problem. Each router operates independently, because they deliver the received datagrams without coordinating with the original sender. If the destination machine is disconnected from the network or if the TTL counter expires, IP will not be able to deliver datagrams.

ICMP is a special-purpose message mechanism, an error reporting mechanism, which must be included in every IP implementation [11]. ICMP allow routers in an internet to report errors or information about unexpected conditions to other routers or hosts, which must relate the error to an individual application program or take other action to correct the problem. It provides communication between IP software on one machine and IP software on another.

ICMP messages require two levels of encapsulation. ICMP message travels across the internet in the data portion of an IP datagram, which itself travels across the internet in the data portion of a frame (fig.2.3) [1].



*fig. 2.3: ICMP Message Encapsulation*

IP is connectionless. Consequently, a router cannot reserve buffer space before receiving datagrams. As a result, a problem known as congestion occurs. Congestion is a traffic overload [4].

When datagrams arrive too quickly to a host or router to process, they are put in a queue temporarily in memory. If the traffic continues, the host or router is overloaded and it should drop datagrams that arrive. In this case, a machine uses ICMP to report congestion to the original source and to request the source to reduce its current rate of datagram transmission.

Also when a router detects that a host uses a non-optimal route, it sends the host an ICMP message, called a redirect, requesting the host to change its route [1].

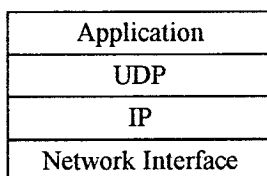
Also when a router drops a datagram because its TTL count has reached zero or expired, it sends an ICMP message back to the datagram's source [11].

#### **2.1.2.4 User Datagram Protocol (UDP)**

At the Internet Protocol (IP) layer, a destination address identifies a host computer.

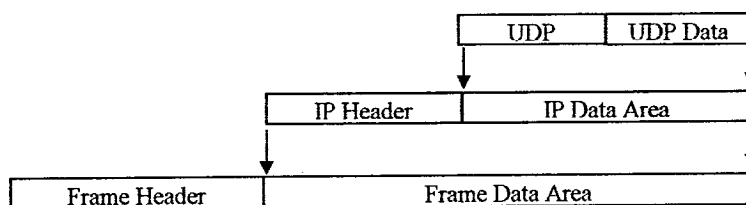
Each machine contains a set of abstract destination points called protocol ports. To communicate with a foreign port, a sender must know the IP address of the destination machine and the protocol port number of the destination within that machine. Each message contains the number of the destination port on the destination machine and the source port number on the source machine to which a reply should be sent.

The User Datagram Protocol (UDP) uses IP to carry messages between machines and adds the ability to distinguish among multiple destinations within a given host computer by providing protocol ports. UDP messages can be lost, duplicated. Each UDP message is called user datagram, which consists of a UDP header and a UDP data area. UDP lies in the layer above the IP layer (fig. 2.4) [1].



*fig. 2.4: UDP Layer*

The IP layer is responsible only for transferring data between a pair of hosts in an internet. UDP layer is responsible only for differentiating among multiple sources or destinations within one host. A UDP message is encapsulated in IP datagram data area as it travels across an internet (fig. 2.5) [1].



*fig. 2.5: UDP Message*

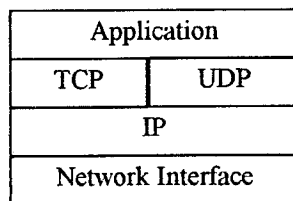
UDP software accepts UDP datagrams from many application programs and passes them to IP for transmission. Also it accepts arriving UDP datagrams from IP and passes them to the appropriate application program. Each application program must negotiate with the operating system to obtain a protocol port and an associated port number before it can send a UDP datagram. UDP is useful when reliable delivery is not needed.

### 2.1.2.5 Transport Control Protocol (TCP)

TCP is a connection-oriented protocol responsible for reliable delivery of messages [2]. It specifies the data format and acknowledgements format exchanged by computers and the procedures used by the computers to guarantee data delivery.

TCP is responsible for breaking up the message into datagrams, reassembling them at the other end, resending anything that gets lost.

TCP resides above IP (fig. 2.6) [1]



*fig. 2.6: TCP Layer*

TCP uses the connection identified by a pair of endpoints, and not the protocol port. TCP defines an endpoint to be a pair of integers (host, port), where host is the IP address for a host and port is a TCP port on that host. A TCP port can be shared by multiple connections on the same machine.

The unit of transfer between TCP software on two machines is called a segment. TCP divides the data stream into segments for transmission. Segments are exchanged to establish or to close connections, to transfer data, to send acknowledgements. Usually, each segment travels across an internet in a single IP datagram.

TCP uses a sliding window mechanism which makes it possible to send multiple segments before an acknowledgement is received, and allows the receiver to restrict transmission to reduce the transmission rates when congestion occurs. TCP allows the window size to vary over time. TCP must reduce transmission rates when congestion occurs.

TCP software at each end maintains two windows per connection, one slides along the data stream being sent, and the other slides as data arrives.

## ***2.2 Internet Multicasting***

The traditional best-effort IP Unicast delivery is a point-to-point unicast delivery, in which for a single source, there is exactly one destination [6]. A Unicast address is designed to transmit a datagram to exactly one destination [10]. In other words, Unicast protocols are used to send packets to one site at a time.

Because of the need of live multimedia, Internet Multicast service extended IP Unicast delivery to a multi-link packet transmission [8]. A Multicast address is used to enable the delivery of datagrams to a set of hosts, which are members of a multicast group.

### **2.2.1 IP Multicasting**

As previously stated, Multicasting is a network routing facility, which sends packets to more than one site at more or less the same time.

On a multicast network, a single packet is sent to an arbitrary number of receivers, instead of having to send that packet once for every destination. Multicast datagrams can be lost, delayed, or duplicated.

Senders need not know about every receiver, and receivers need not know about the sender's identity to receive its multicast. Instead, sources send packets to a multicast group, which contains the subscribed receivers [6].

Membership in a group determines whether the host will receive datagrams sent to the multicast group. Also a host may send datagrams to a multicast group without being a member.

Each multicast group has a unique multicast address. Some IP multicast well-known addresses are assigned by the Internet, and correspond to groups that always exist even if no member belongs to that group. Other addresses are temporary and are discarded when the number of members belonging to that group is zero.

IP multicast, used in the internet, use special multicast routers, called mrouter, to forward multicast datagrams.

The TTL field in a multicast datagram limits propagation through routers.

The TCP/IP standard for multicasting defines IP multicast addressing, specifies how hosts send and receive multicast datagrams, and describes the protocol routers use to determine multicast group membership in a network.

### 2.2.2 Multicasting Advantages

Multicasting has many advantages described next:

- It extends the point-to-point transmission of the traditional unicast delivery to a point-to-multipoint transmission [8].
- It conserves bandwidth by forcing the network to do replication only when necessary [7].
- It reduces the network load by sending a single packet from the source to an arbitrary number of receivers by replicating the packet within the network at points along a multicast delivery tree rooted at the packet's source [1].
- It offers a range of scope-control mechanisms. Applications can limit the distance with which a multicast packet travels. The TTL field in the IP address header can be used to limit the range (or scope) of a multicast transmission. The TTL value is decremented every time the packet passes through an mrouter. A threshold is assigned to each multicast link. If a multicast packet's TTL is less than the threshold, then the packet is dropped [1].

- It offers a flexible group membership. Individual hosts are free to join or leave a multicast group at any time. There is no restriction on the physical location or the number of members in a multicast group. A host may be a member of one or more multicast groups at any time. A host does not have to belong to a group to send messages to members of a group. A group membership protocol is employed by routers to know about the presence of group members [6].

### 2.2.3 Extending IP to Handle Multicast

IP software on the host must have an interface allowing an application program to declare that it wants to join or leave a multicast group.

Whenever multiple application programs join the same group, IP software must remember to pass each of them a copy of datagrams sent to that group.

If all application programs leave a group, a host must remember that it no longer participates in that group. The host must run group membership protocol that informs the local multicast routers (m routers) of its group membership status.

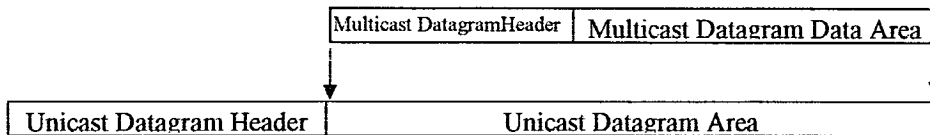
### 2.2.4 Internet Multicast Backbone (MBone)

MBone is an interconnected set of sub-networks and routers that support IP multicast delivery [7]. It is a virtual multicast network embedded within the traditional unicast. Thus, it is a network that runs on top of the Internet, because it shares the same physical media-wires, routers, and other equipment, as the Internet. It is composed of networks of multicast routing capability connected to other networks by tunnels.

A tunnel is needed when two or more machines wish to participate in multicast applications and when one or more routers do not run multicast routing software.

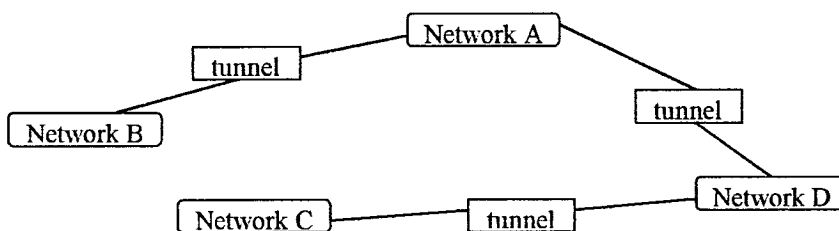
The tunnels allow multicast packets to travel through routers that can handle only unicast traffic by hiding multicast packets in traditional unicast packets (fig.2.7)[10].





*fig. 2.7: Multicast Datagram Tunneling*

The scheme of moving multicast packets by putting them in unicast packets is called tunneling. The multicast packets remain hidden in unicast packets until they reach a router or a host that can handle multicast packets where they are processed as multicast packets (fig. 2.8) [10].



*fig. 2.8: Internet Multicast Backbone*

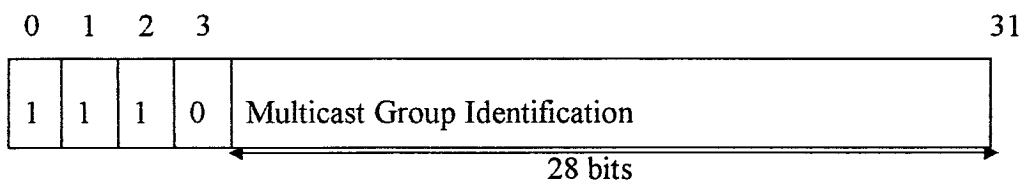
The MBone and the Internet have different topologies. For this reason, multicast routers execute a separate routing protocol to decide how to forward multicast packets.

MBone carries audio and video multicasts, live satellite weather photos, etc.[7] The Session Directory (sd) tool provides users with a listing of the active multicast sessions on the MBone and allows them to define or join a conference. It can be used to automatically download and install authorized upgrades and bug-fixes to a computer software, without the user's intervention.

### 2.2.5 IP Multicast Addresses

A multicast address is assigned to a set of receivers defining a multicast group. Senders use the multicast address as the destination IP address of a packet that must be transmitted to all group members.

IP multicast uses class D addresses of the form (fig. 2.9) [10]:



*fig. 2.9: Class D Address*

The first four bits contain 1110 and identify the address as a multicast address.

Bits 4 through 31 identify a multicast group, which does not identify the origin of the group, nor does it contain a network address.

When expressed in dotted decimal notation, multicast addresses range from: 224.0.0.0 through 239.255.255.255

224.0.0.0 is reserved and cannot be assigned to any group.

224.0.0.1 is permanently assigned to all hosts group, which includes all hosts and routers participating in IP multicast.

No IP multicast address is assigned to refer to all hosts in the internet.

### **2.2.6 Internet Group Management Protocol (IGMP)**

A host, wishing to participate in IP multicast on a local network, must have software allowing it to send and receive multicast datagrams.

To participate in a multicast that spans multiple networks, the host must inform local multicast routers. The local routers contact other multicast routers, passing on the membership information and establishing routes. Before a multicast router can propagate multicast membership information, it must determine that one or more hosts on the local network have decided to join a multicast group. To do so, multicast routers and hosts that implement multicast must use the IGMP to communicate group membership information. IGMP uses IP datagrams to carry messages. It is an integral part of IP. It runs between hosts and their immediately neighboring m routers.

IGMP has two phases [1]:

1. When a host joins a new multicast group, it sends an IGMP message to the “all hosts” multicast groups declaring its membership. Local multicast routers receive the message and propagate the group membership information to other m routers across the internet.

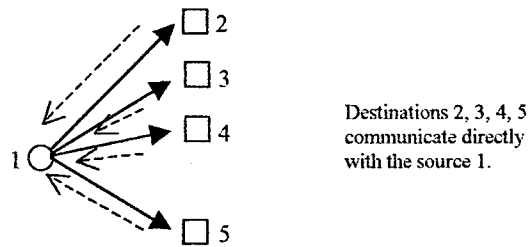
2. Membership is dynamic. Then, local m routers periodically polls hosts on the local network to determine which hosts remain member of which groups. If no host reports membership in a group after several polls, the m router assumes that no host on the network remains in that group, and stops advertising group membership to other m routers.

### 2.2.7 Distance Vector Multicast Routing Protocol (DVMRP)

Multicast routers execute a DVMRP to define delivery paths that enable the forwarding of multicast datagrams across an internetwork, and to pass group membership information among them. DVMRP uses IGMP messages to carry this information. It defines IGMP message types that allow routers to declare or leave membership in multicast groups, and to receive routing information from other routers [10].

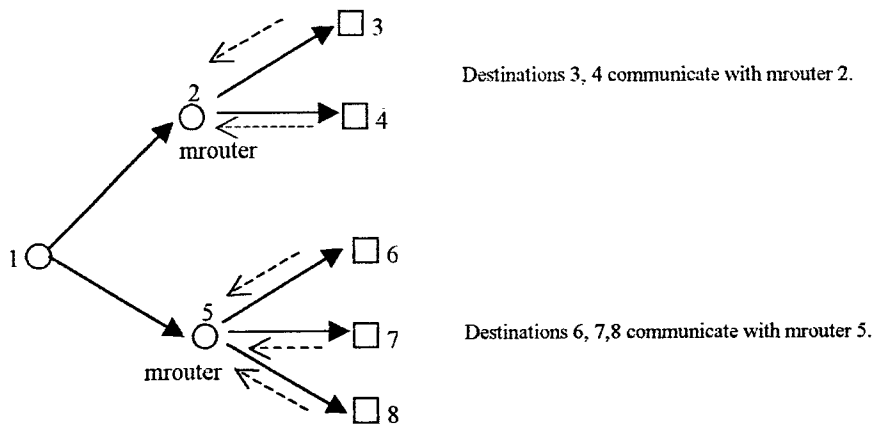
## 2.3 Research Motivation

Let us consider a Multicast Backbone network (or Mbone). In such a network, the system response time is the time to transmit the messages to all the members in a group. By changing the propagation topology of a multicast transmission from a one-stage topology to a hierarchical one, we would be able to control the total delay. In fact, the propagation topology of messages in our original system has the following configuration (fig. 2.10):



*fig. 2.10: One-Stage Topology: One-Stage Network*

While, if we decide to use the intermediate sites (or mrouter), the topology becomes (fig. 2.11).



*fig. 2.11: Hierarchical Topology: Two-Stage Network*

The introduction of hierarchy in such a network can help reduce the system response time by providing the mrouter with some additional functions such as back-up and error recovery functions.

This issue will be our main objective in this thesis. In fact, chapter 3 will analyze the response time in two-stage network and will compare it to that of a one-stage network. Also, chapter 3 will try to find an optimum configuration for the Mbone. In particular, it will try to find the optimum number of mrouter (or intermediate sites) for a given network.

We should finally note that the loss of packets is considered in the present system and it represents a major constraint on the network performance. So, we are aiming to minimize the delay and to improve the reliability of a real-time IP Multicast network.

# CHAPTER 3

## **Multicast Two-Stage Networks vs. Multicast One-Stage Networks**

In this chapter, we compare the performance of multicast two-stage networks to multicast one-stage networks using MATLAB simulations. Our simulation models involve 100 sites spread either in one-stage network or two-stage network. The presence of packet loss is taken into account and hence, retransmission of lost packets are simulated and analyzed.

### ***3.1 Assumptions Used in the Simulation***

The assumption that are considered in both models are:

- The bit error rate (BER) is equal to  $10^{-4}$  and the packet size is 1000 bits.

Since the probability of corrupted packet is equal to

$$1-(1-\text{BER})^{\text{PacketSize}}=1-(1-10^{-4})^{1000}=0.0952$$

Then, the packet retransmission number is equal to

$$1/(1-\text{Probability of Corrupted Packet})=1.1052$$

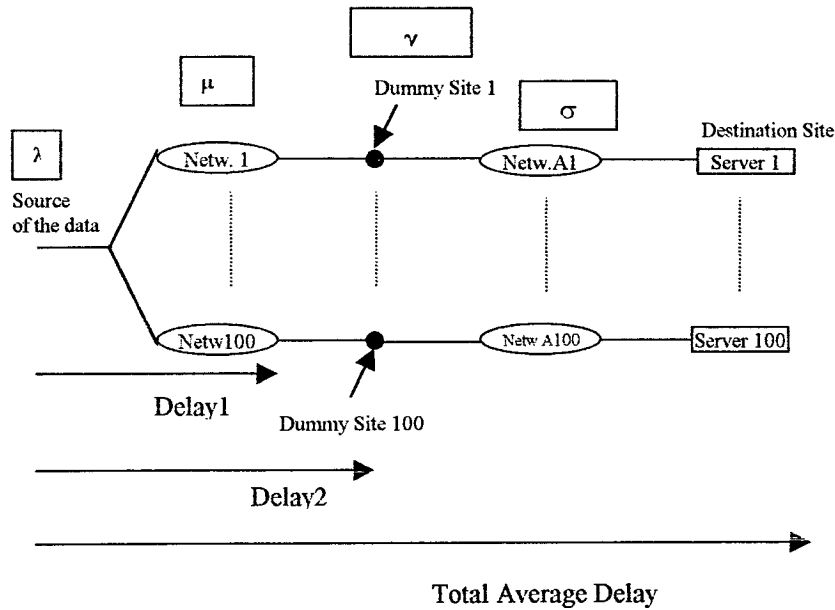
- Packets are generated at the source as a Poisson process with arrival rate  $\lambda$ . The arrival rate  $\lambda$  varies between 0.1 and 0.9.

- The service rate at the first (respectively the second) stage of the network is  $\mu$  (respectively  $\sigma$ ).  $\mu$  and  $\sigma$  are taken larger (or equal) to 1 so that the system response time does not diverge. In fact,  $\lambda$  must be smaller than  $\mu$  and  $\sigma$ .
- The service rate at the intermediate sites (or routers) is taken equal to  $\gamma$ .  $\gamma$  is also larger or equal to 1.
- $\mu$ ,  $\gamma$ , and  $\sigma$  are all considered to have an exponential distribution.
- In order to be able to compare the two systems, both networks involve two hops of networks connected by routers. In the one-stage model, the routers are not involved in the communication protocol proposed, i.e., they do not have back-up function nor do any retransmission. Only source nodes retransmit lost packet. For this reason, we call them “Dummy Sites”. Moreover, in the two-stage model, the routers are active. They have back-up functions and they retransmit any packet which is lost in the second stage of the network.

## ***3.2 Multicast in One-Stage Networks***

### **3.2.1 Architecture of Multicast One-Stage Networks**

In this research, the architecture of a multicast one-stage network is defined in figure 3.1 below:



*fig. 3.1: Multicast One-Stage Network*

The architecture of a multicast one-stage network consists of a source node trying to transmit a multicast IP message to 100 destination sites. Then, the source node is connected to 100 networks. Each of the 100 networks is connected to a dummy site (i.e. a router). It is called a dummy site, because it does not support error correction nor backup of data. Each dummy site is connected to another network hop (Netw.A), before reaching the destination site (or server).

### 3.2.2 Multicast One-Stage Network Analysis

We need to multicast packets to 100 networks. The source of the data sends each of the packets to the 100 networks at more or less the same time. Because IP datagrams can be delayed, lost or duplicated, the probability of error in packet's delivery is possible. In this case, a negative acknowledgement (NAK) is sent back to the source of the data asking it to retransmit the corresponding packet to the corresponding destination. NAK message includes the corresponding sequence number of the packet. Then, an additional delay for the retransmission of the packet is added to the total delay.

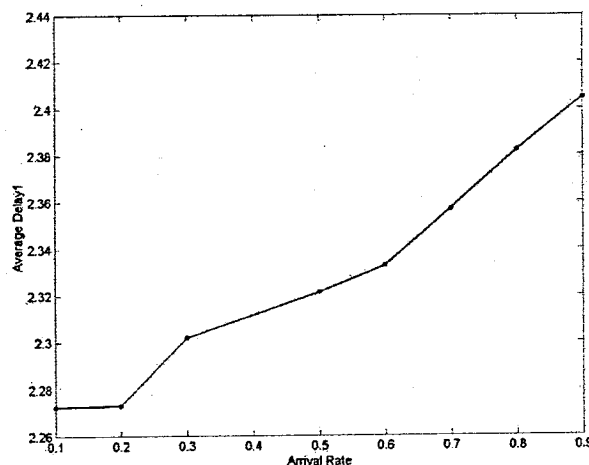
At the first hop of the network, each of the 100 networks transmits the received packets to the dummy site (i.e. a router) connected to it, where only a service rate  $\gamma$  is considered. If the packet is corrupted, the router (i.e. dummy site) does not discover the error, and transmits the packet forward to the second hop of the network, and finally to destination site. At Netw.A, a service rate  $\sigma$  is taken into account and error-correction is supported. Whenever a packet is not in its normal condition, Netw.A requests the retransmission of the packet from the source of the data, because the router (i.e. dummy site) is not able to backup data. Then, a NAK message is sent back to the source of the data. Consequently, a NAK delay is created as well as a delay for the retransmission of the packet.

### 3.2.3 Multicast One-Stage Network's Simulation Results

#### 3.2.3.1 The arrival rate $\lambda$ is variable

We assume that the arrival rate of packet  $\lambda$  is variable. As stated earlier, it varies between 0.1 and 0.9. The service rates  $\mu$  at the first network,  $\gamma$  at the dummy site, and  $\sigma$  at the second hop of the network are fixed, and all equal to 1.

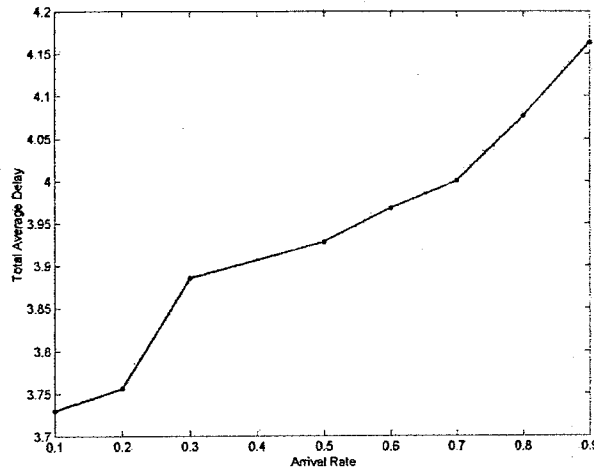
We first calculate the average delay of the network from the source of the data till the routers (i.e. dummy sites) as a function of the arrival rate  $\lambda$ . The following graph is generated (fig. 3.2)



*fig. 3.2: Average Delay1 Variation: from the Source of the data to the Dummy Sites*



Then, we calculate the total average delay of the *whole* network from the source of the data till the destination servers. The following graph is generated (fig. 3.3)



*fig. 3.3: Total Delay Variation: from the Source of the data to the Destination Servers*

Like in any network where the arrival process of packets is Poisson and the service process at the different servers are exponential, we suspect that when the arrival rate  $\lambda$  increases, the delay in the network increases. We should note that the “complex” impact of NAK delay and retransmission delay did not affect the “expected” shape of the delay curve, i.e., it is still an increasing function. In fact, we can easily prove that the stability condition ( $\lambda < \mu$ ,  $\lambda < \sigma$ ,  $\lambda < \gamma$ ) of the network with packet loss and retransmission is the same as the stability condition for the same network with FIFO (First In First Out) processing at the routers (i.e. with dummy sites).

### **3.2.3.2 $\mu$ is variable, or $\gamma$ is variable, or $\sigma$ is variable**

In this paragraph, we analyze the impact of the variation of the service rates on the total delay in the network. We start by varying the service rate of the first network  $\mu$ , with  $\gamma$  and  $\sigma$  fixed, and equal to 1. The arrival rate of packets in all the three diagrams (fig. 3.4, 3.5, 3.6) is fixed and equal to 0.9.

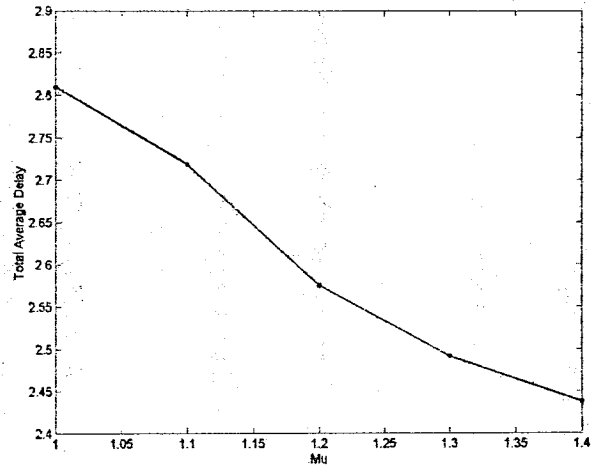


fig. 3.4: Total Average Delay as function of  $\mu$

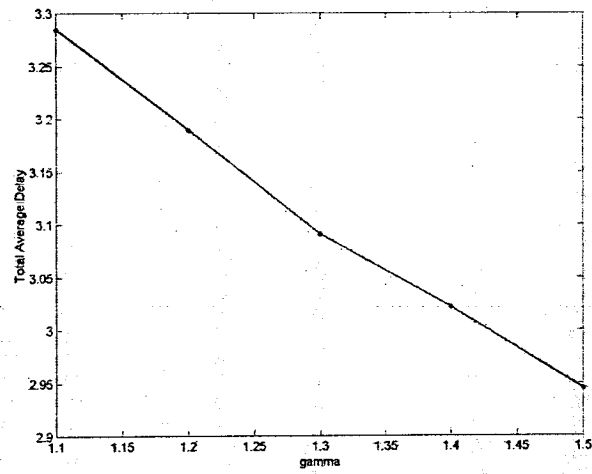


fig. 3.5: Total Average Delay as a function of  $\gamma$

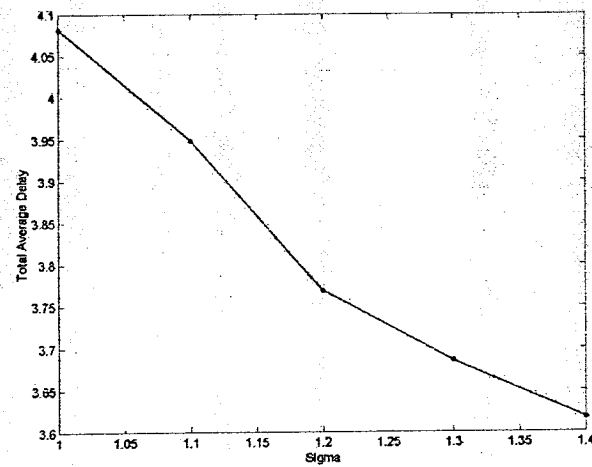


fig. 3.6: Total Average Delay as a function of  $\sigma$

From the above figures (3.4, 3.5, and 3.6), we can conclude that whenever we assume that  $\mu$ , or  $\gamma$ , or  $\sigma$  is variable, the total average delay of the network will decrease as the variable value is increased. In fact, for the given values of  $\mu$ ,  $\gamma$ , and  $\sigma$ , we notice that a 40% increase in the servers' speed can decrease the total average delay of 10% approximately.

### 3.3 Multicast in Two-Stage Networks

#### 3.3.1 Architecture of Multicast Two-Stage Networks

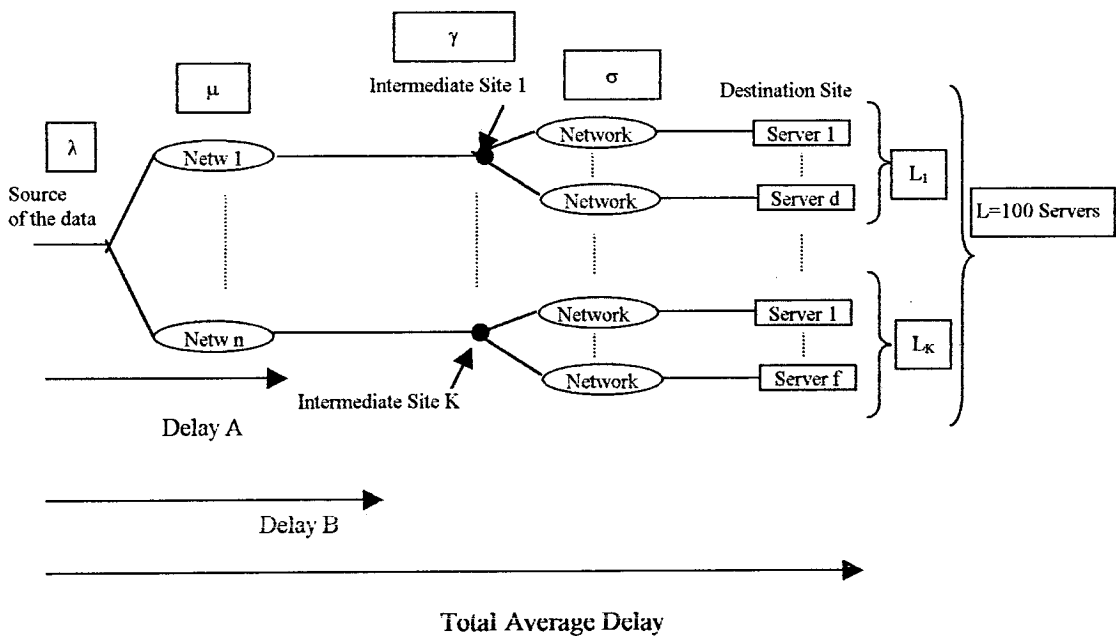


fig. 3.7: Multicast Two-Stage Network

In figure 3.7, the architecture of a multicast two-stage network is presented. It consists of a source node connected to a number of networks. Each of these networks is connected to an intermediate site (i.e. a router). Each router is connected to several networks, which lead to a number of destination sites.

The total number of destinations is still equal to 100. Here, our major objective is to find the optimal number of intermediate sites so that the overall performance of the network is optimized.

### 3.3.2 Multicast Two-Stage Network Analysis

We need to multicast packets to 100 destination sites. Packets are forwarded from the root (i.e. source of the data) along a distribution tree to each receiver in the multicast group. Each of the originated packets is sent from the source of the data to Netw where only a service rate is considered. Then, from each of the Netw, it is sent to a router (i.e. intermediate site) where a service rate is considered and backup of a packet and error-correction are considered. Whenever a packet is corrupted, lost or duplicated, a NAK message is sent back to the source of data with the packet sequence number to be retransmitted. Each of the intermediate sites (or multicast routers) forwards the datagram to other sites where a service rate is considered and error-correction is supported. Whenever a delivery error occurs, a NAK message is sent back to the related intermediate site asking it to retransmit the corresponding packet. The intermediate site will again retransmit the requested packet to the destination site.

### 3.3.3 Multicast Two-Stage Network's Simulation Results

In a two-stage network, the variables used (i.e.  $\lambda$ ,  $\mu$ ,  $\gamma$ ,  $\sigma$ ) are similar to those in one-stage network. Let  $K$  and  $L$  be the number of routers (i.e. intermediate sites), and the number of destination sites, respectively.

Note that  $L_1 + L_2 + L_3 + \dots + L_k = L = 100$ .

In this section, we aim to prove that for a certain value of  $K$  and  $L$ , a multicast two-stage network is better than a multicast one-stage network when back-up and error recovery are used at routers (or intermediate sites).

In [5], it was proven that for general multi-link networks, two-stage networks are better than one-stage networks. The optimum value of  $K$  as a function of  $L$  (assuming all other parameters in the system are constant) is given by  $K \cong L^{1/r}$ , where  $r > 1$ .

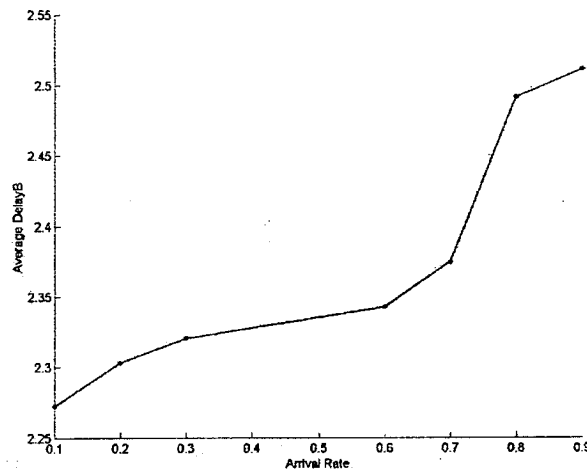
All we need to do is to show that the above result still hold for IP-multicast networks. In our simulations, we considered  $L=100$  nodes.

### 3.3.3.1 The arrival rate $\lambda$ is variable

In this section, we assume that  $\lambda$  is variable and varies between 0.1 and 0.9. The service rates are fixed and all equal to 1. At first, we assume that the number of intermediate sites  $K$  is equal to 10. Then, every intermediate site is responsible for 10 servers, because the total number of destination sites is equal to 100.

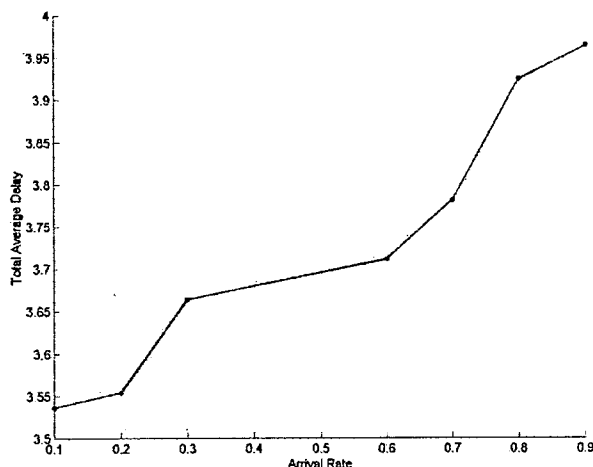
As a reminder, the presence of packet loss and the need to retransmit corrupted packets are taken into account as stated in Section 3.1.

We calculate the average delay in the first network (i.e. from the source of the data to the intermediate sites). The following graph is generated (fig. 3.8)



*fig. 3.8: Average Delay Variation: from the Source of the data to the Intermediate Sites*

We also calculate the total average delay of the multicast two-stage network from the source of the data till the destination servers. The following graph is generated (fig. 3.9)



*fig. 3.9: Total Average Delay Variation: from the Source of the data to the Destination Servers*

As stated earlier, the network delay is an increasing function of the arrival rate. Although back-up and retransmission functions increase the delay in the system, they do not affect the shape of the delay curve.

### **3.3.3.2 Intermediate Site Optimal Number**

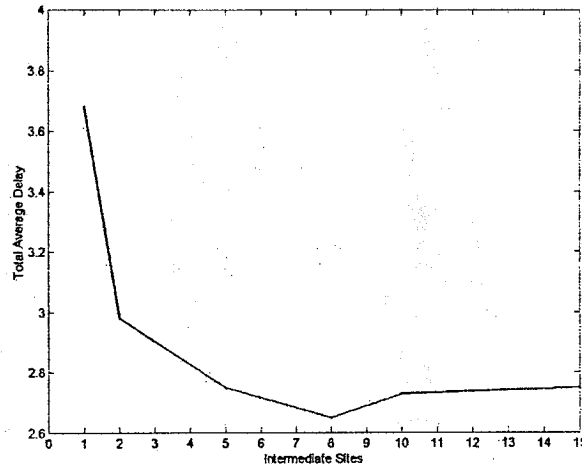
To find the optimal number of intermediate sites of the multicast two-stage network, we fix the values of  $\lambda$ ,  $\mu$ ,  $\gamma$ , and  $\sigma$ .

As state previously in section 3.3.3, the number of intermediate sites  $K$  varies as we change  $r$ :

number of intermediate sites =  $\text{round}(\text{total sites number})^{1/r}$ , where  $r > 1$

$K = \text{round}(L)^{1/r}$ , where  $r > 1$

We assume that  $\lambda$  is equal to 0.9. The first network service rate  $\mu$  is equal to 1, while  $\gamma$  and  $\sigma$  are both equal to 2. The following graph shows the variation of the delay as a function of the number of intermediate sites  $K$ .



*fig. 3.10: Total Average Delay Variation as a function of Intermediate Sites*

We deduce that the optimal number of intermediate sites is equal to 8, because the total average delay of the network at that point represents the minimal value of the curve. In fact, the delay drops around 35% of its original value. So with 100 multicast users, we should have 8 intermediate sites. The first 7 intermediate sites are connected each to 12 end-users (i.e. the cluster size is 12 users) and the 8<sup>th</sup> intermediate site is connected to 16 end-users.

### **3.3.3.3 $\mu$ is variable, or $\gamma$ is variable, or $\sigma$ is variable**

In the following graphs (fig. 3.11, 3.12, 3.13),  $\lambda$  is fixed to 0.9 and one of the service rates is variable while the two others are fixed.

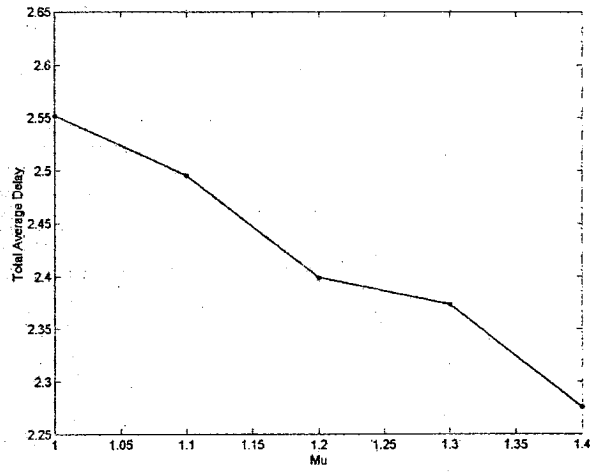


fig. 3.11: Total Average Delay vs.  $\mu$

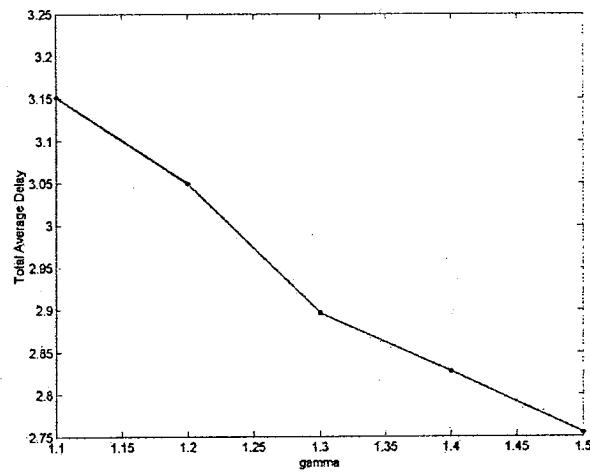


fig. 3.12: Total Average Delay vs.  $\gamma$

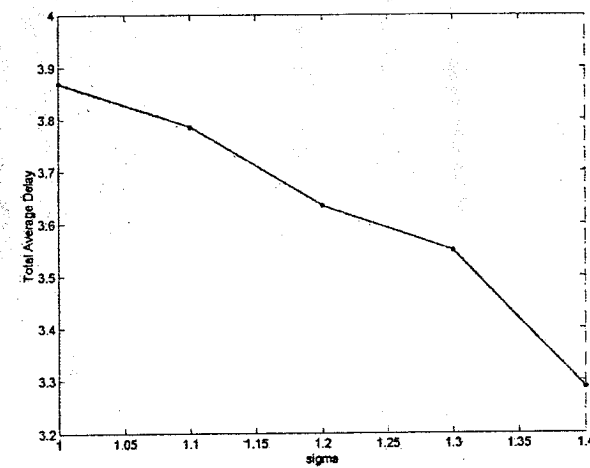


fig. 3.13: Total Average Delay vs.  $\sigma$



From the above figures (3.11, 3.12, 3.13), we can conclude that when we increase the speed of any of the servers, the network delay decreases. However, it seems that the impact of change in the value of  $\sigma$  (i.e. the last network service rate) is higher than the other servers (see table below).

	40% increase in $\sigma$	40% increase in $\mu$	40% increase in $\gamma$
% of decrease in the delay	15%	11%	13%

### 3.4 Comparison of Multicast One-Stage Networks and Multicast Two-Stage Networks

#### 3.4.1 Benefits of the Intermediate Sites Optimal Number

Note that in fig. (3.14, 3.15, 3.16), the curves of the multicast one-stage are only added to the graphs to see the difference between the two total average delay curves. The change in the number of intermediate sites does not affect the delay performance in one-stage network since their numbers are fixed to 100, and they are actually dummy sites.

In sub-section 3.3.3.2, we found that the intermediate sites optimal number of the multicast two-stage network is equal to 8, whenever the arrival rate  $\lambda$  is equal to 0.9, the service rate  $\mu$  is equal to 1, and the service rate  $\gamma$  and  $\sigma$  are both equal to 2. Also by considering the fig. 3.14 below, we can notice that the total average delay of the *multicast two-stage network* has **decreased around 35%** from that of the *multicast one-stage network*.

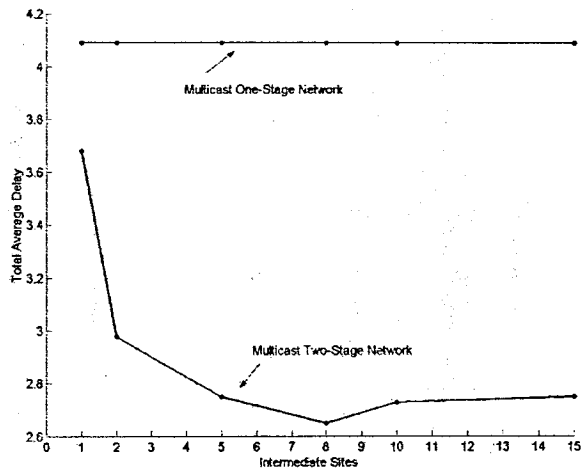


fig. 3.14.: Impact of the number of Intermediate Sites on the Delay  
 ( $\lambda=0.9, \mu=1, \gamma=\sigma=2$ )

Taking the same value for the arrival rate  $\lambda$  (i.e. 0.9), and changing the service rates  $\mu$ ,  $\gamma$ , and  $\sigma$  to 1, we found that  $K$  optimal number changes to 5 (fig. 3.15), and we can notice that the total average delay of the *multicast two-stage network* has decreased around 26% from that of the *multicast one-stage network*.

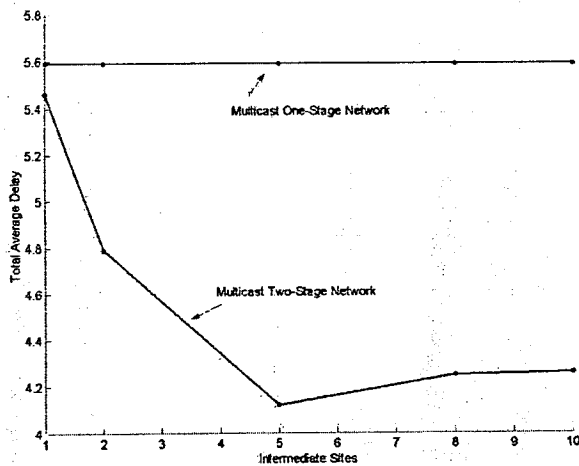


fig. 3.15.: Impact of the number of Intermediate Sites on the Delay  
 ( $\lambda=0.9, \mu=\gamma=\sigma=1$ )

Finally, if we consider the arrival rate  $\lambda$  equal to 0.9, and the service rates  $\mu, \gamma$  are both equal to 1, while  $\sigma$  equal to 2, we found that  $K$  optimal number is equal to 9 (fig. 3.16). Also we can deduce that the total average delay of the *multicast two-stage network* has **decreased around 28%** from that of the *multicast one-stage network*.

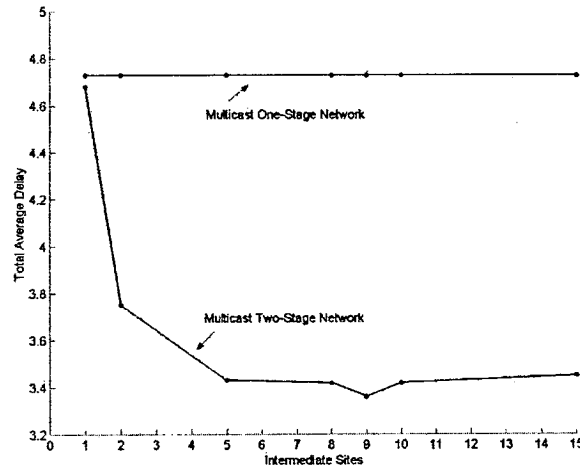


fig. 3.16.: Impact of the number of Intermediate Sites on the Delay

$$(\lambda=0.9, \mu=\gamma=1, \sigma=2)$$

In the graphs above (figs. 3.14, 3.15, 3.16), comparing the total average delay of the multicast two-stage network to that of the multicast one-stage network, a remarkable delay reduction is deduced. In fact, we notice that when we increase the speed of the routers and second stage network, we can get the best improvement (i.e. 35%). Also we can notice that the optimal number of  $K$  increases as well (from 5 to 8 to 9). A logical explanation for this phenomena is: A decrease in the number of  $K$  decreases the load on the first stage, while it increases the load on the second stage. In fact, if the number  $K$  is very small, the  $K$  routers will have to bear a large load resulting from the handling of error correction and back-up. This will lead to a larger overall delay since the routers will be like a bottleneck. While an increase in the value of  $K$  will decrease the load on the intermediate sites since the workload will be divided among all the routers. This will result in a decrease in the overall delay especially if the service rate in the routers and the second stage network were larger.

Consequently, the use of the intermediate site optimal number in *multicast two-stage network* enhances the performance of the network. Then, *multicast two-stage network* is more efficient than the *multicast one-stage network*.

### 3.4.2 Impact of the Arrival Rate $\lambda$ on the Delay

The arrival rate  $\lambda$  is variable, the service rate  $\mu$  is fixed to 1, while the service rates  $\gamma$ , and  $\sigma$  are both fixed to 2. In the following graph (fig. 3.17), we assume that the number of intermediate sites  $K$  is equal to 8 (i.e. the optimal number for  $K$  for these network specifications).

As we can see in fig. 3.17, a multicast two-stage network performs better than a multicast one-stage network. In the present configuration, the percentage of improvement is around 35%.

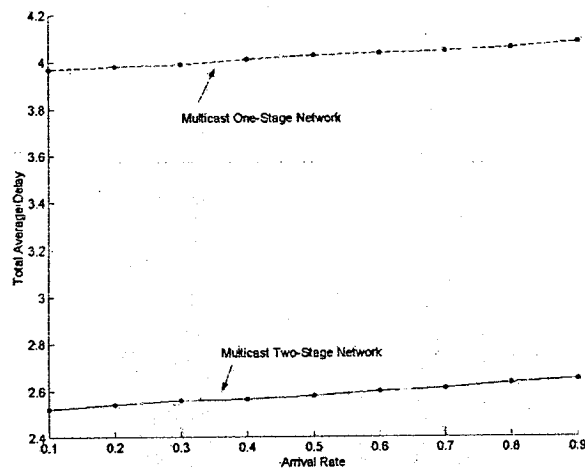


fig. 3.17: Multicast Two-Stage Network vs. Multicast One-Stage Network

# CHAPTER 4

## Conclusion

### *4.1 Final Results*

The problem of IP multicast was analyzed in this thesis. Simulation models were built using MATLAB software in order to analyze the performance and investigate new proposed ideas.

We found that in a TCP/IP network, it is better to multicast packets to different sites using a multicast two-stage network rather than a multicast one-stage network, because the network load is reduced and the delay is minimized. The presence of abnormal packet conditions is considered as well as the packet retransmission is analyzed in our simulation models.

We showed (using simulations) that the total average delay of the *multicast two-stage network* is **less around 35% than** the total average delay of the *multicast one-stage network*.

We noticed that in multicast two-stage network, the workload is distributed among the source nodes and the intermediate sites. This resulted in better system performance including delay, reliability, and throughput.

In fact, in a multicast two-stage network, the intermediate sites divides the total number of sites to clusters. Every intermediate site is responsible for one cluster of nodes. By taking the number of intermediate sites equal to its optimal value, we can enhance the performance of the network by minimizing its total average delay, and keeping its reliability.

### 4.2 Future Works

In multicast two-stage network, if the total number of destination sites cannot be divided equally to be connected to one of the intermediate site, the remaining destination site(s) is/are added to the ultimate intermediate site (fig.4.1). We could investigate the option of dividing the remaining destination sites to more than one intermediate site instead of adding them to the ultimate intermediate site (fig. 4.2).

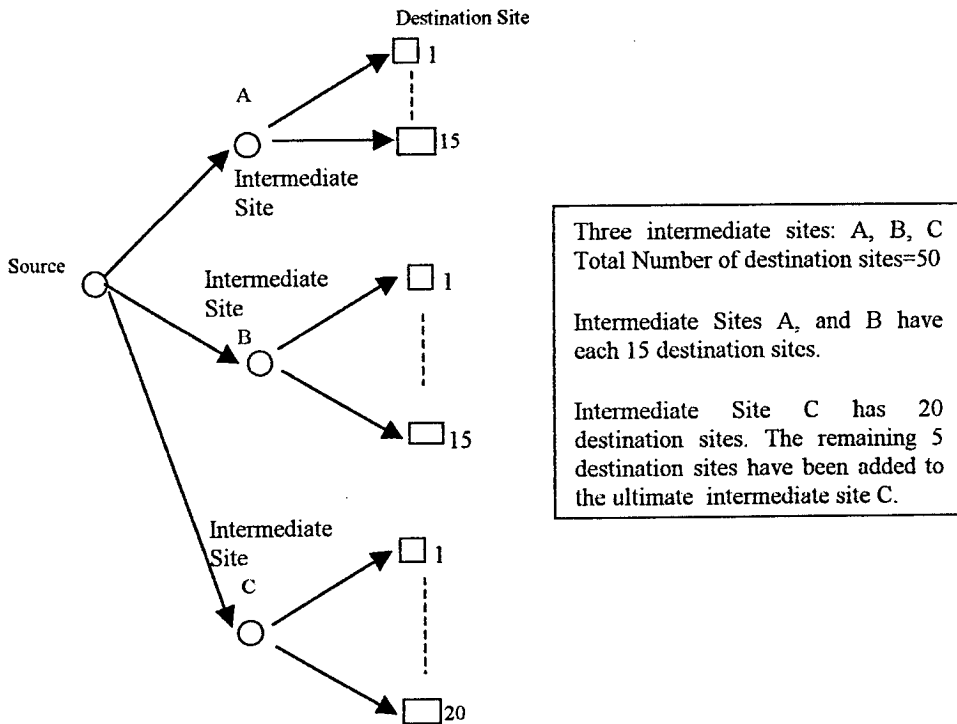
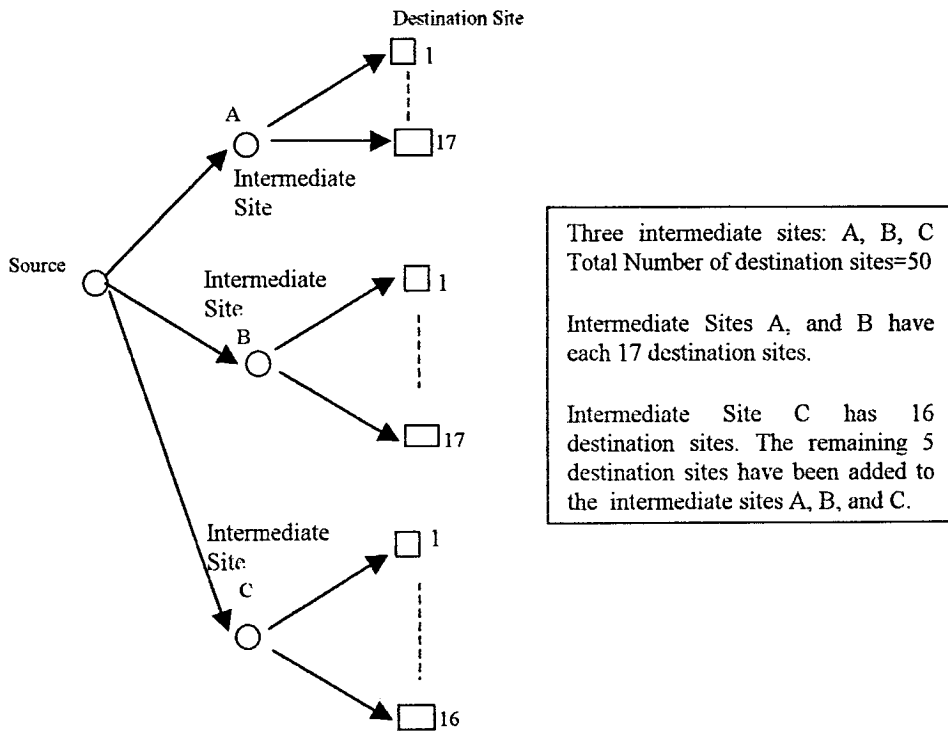


fig. 4.1: Remaining Destination Sites added to the Ultimate Intermediate Site C



*fig. 4.2: Remaining Destination Sites added to the Intermediate Sites A, B, C*

Whenever an abnormal condition of a packet occurs in one of the final sites in multicast two-stage network, the corresponding packet must be retransmitted from the corresponding intermediate site where a backup copy of it exists. Instead of backing up the whole packets in the intermediate site, we investigate the option to attach to every intermediate site a server where the backup copy could be temporarily saved. The backup copy will contain only important information and data needed to restore an abnormal packet. This method could be used with forward error detection techniques. Note that backward error detection techniques were only considered in this thesis.

## References

- [1] Comer, D.E. *Internetworking with TCP/IP. Vol. 1-Principles, Protocols, and Architecture*. Second Ed., Englewood Cliffs, NJ: Prentice Hall, 1991.
- [2] Floyd, S., et al. *A Reliable Multicast Framework for Light-Weight Sessions and Application Level Framing*. IEEE/ACM Transactions Networking, December 1997.
- [3] Green, D.C. *Data Communication*. Second Ed., Longman, 1995.
- [4] Kasera, Sneha, et al. *Scalable Fair Reliable Multicast using Active Services*. IEEE Net., Jan/Feb 2000.
- [5] Maalouf, H.W. *Optimisation of the Communication Network Performance of Distributed Systems with Resequencing Constraints*. PhD Thesis. Digital Communications Section. Department of Electrical and Electronic Engineering, Imperial College, University of London, 1998.
- [6] McCanne Steven. *Scalable Multimedia Communication with Internet Multicast Light-Weight Sessions, and the MBone*. University of California, Berkeley.
- [7] McCanne Steven, Amir Elan, and Katz Randy. *Receiver-Driven Bandwidth Adaptation for Light-Weight Sessions*. University of California, Berkeley, 1997.
- [8] McCanne Steven, et al. *RMX: Reliable Multicast for Heterogeneous Networks*. University of California, Berkeley.
- [9] Meyer, J.F. *Performability: A Retrospective and Some Pointers to the Future*. 1992.



- 
- [10] Semeria, C., and Maufer, T. *Introduction to IP Multicast Routing*. March 1997.
- [11] Tanenbaum, Andrew. *Computer Networks*. Third Ed., Prentice Hall, 1996.

% APPENDIX B

-----

```
k=input('k = ');  
pr=zeros([1,100]);  
for ii=1:4:100  
    pr(ii)=isprime(ii);  
end  
P(1)=0;  
for iii=2:100  
    rt(iii)=rand;  
    pp=rt(iii)*2/k;  
    P(iii)= P(iii-1)+pp;  
end
```

-----

```
i=1:100;  
packetsNb=100;  
PacketSize=1000;  
BER=0.0001;  
ErrorProbab=1-((1-BER)^PacketSize);  
RetransNb=(1/(1-ErrorProbab));
```

-----

```
Q=input('Total Sites Nb (Q) = ');  
%r=input('r = ');  
%IntNodes=round(Q^(1/r))  
IntNodes=input('IntNodes = ');
```

```
global UsersNb;  
UsersNb=floor(Q/IntNodes);  
NetUsers=IntNodes*UsersNb;  
TotNetUsers=Q-NetUsers;
```

-----

```
Mu=input('Mu = ');  
maxload=-1/Mu*log(1-rand);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%Multicast Two-Stage Network *  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Tnow=zeros(IntNodes,packetsNb);  
for jy=1:IntNodes  
    for y=1:packetsNb  
        servicel(jy,y)=-1/Mu*log(1-rand);  
        Tnow(jy,y)=P(y)+servicel(jy,y);  
    end  
end  
SortedTnow=zeros(IntNodes,packetsNb);  
OrderTnow=zeros(IntNodes,packetsNb);  
for si=1:IntNodes  
    [SortedTnow(si,i),OrderTnow(si,i)]=sort(Tnow(si,:));  
end
```

```
Net1TotalDelay=zeros(IntNodes,packetsNb);  
for jb=1:IntNodes  
    for packetb=1:packetsNb  
        Net1TotalDelay(jb,packetb)=Tnow(jb,packetb)-P(packetb);  
    end  
end
```

```
Netw1AverageDelay=zeros(1,packetsNb);  
Netw1AverageDelay=mean(Net1TotalDelay);
```

```
Netw1Avg=0;
```

```
NetwIAvg=mean (NetwIAverageDelay)
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%Multicast One-Stage Network %  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Tnow_win=zeros (Q,packetsNb);  
for v=1:Q  
    for ic=1:packetsNb  
        servicewl (v,ic)=-1/Mu*log (1-rand);  
  
        NAKdelaywl (v,ic)=0;  
        TransmDelaywl (v,ic)=0;  
        Delaywl (v,ic)=0;  
  
        if (pr(ic)==1)  
            NAKdelaywl (v,ic)=-1/Mu*log (1-rand);  
            TransmDelaywl (v,ic)=-1/Mu*log (1-rand);  
        end  
  
        Delaywl (v,ic)=RetransNb*(TransmDelaywl (v,ic)+NAKdelaywl (v,ic));  
  
        Tnow_win (v,ic)=P (ic)+Delaywl (v,ic)+servicewl (v,ic);  
    end  
end
```

```
SortedTnow_win=zeros (Q,packetsNb);  
OrderTnow_win=zeros (Q,packetsNb);  
for sa=1:Q  
    [SortedTnow_win (sa,i),OrderTnow_win (sa,i)]=sort (Tnow_win (sa,:));  
end
```

```
NetlTotalDelay_win=zeros (Q,packetsNb);  
for jn=1:Q  
    for packetn=1:packetsNb  
        NetlTotalDelay_win (jn,packetn)=Tnow_win (jn,packetn)-P (packetn);  
    end  
end
```

```
NetwIAverageDelay_win=zeros (1,packetsNb);  
NetwIAverageDelay_win=mean (NetlTotalDelay_win);
```

```
NetwIAvg_win=0;  
NetwIAvg_win=mean (NetwIAverageDelay_win)
```

```
%-----  
ga=input ('ga = ');
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%  
%Multicast Two-Stage Network %  
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

```
Tnowg=zeros (IntNodes,packetsNb);  
for ja=1:IntNodes  
    for a=1:packetsNb  
        service2 (ja,a)=-1/ga*log (1-rand);  
  
        sDelay (ja,a)=0;  
        if (a>1) & (OrderTnow (ja,a)<OrderTnow (ja,(a-1)))  
            sDelay (ja,a)=Tnowg (ja,(a-1))-SortedTnow (ja,a);  
            if (sDelay (ja,a)<0)  
                sDelay (ja,a)=0;  
            end  
        end  
  
        NAKdelayg (ja,a)=0;  
        TransmDelayg (ja,a)=0;  
        Delayg (ja,a)=0;  
  
        if (pr(a)==1)  
            NAKdelayg (ja,a)=-1/ga*log (1-rand);  
        end  
    end  
end
```

```

    TransmDelayg(ja,a)=-1/ga*log(1-rand);
end

Delayg(ja,a)=RetransNb*(TransmDelayg(ja,a)+NAKdelayg(ja,a));

Tnowg(ja,a)=SortedTnow(ja,a)+sDelay(ja,a)+service2(ja,a)+Delayg(ja,a);
end
end

SortedTnowg=zeros(IntNodes,packetsNb);
OrderTnowg=zeros(IntNodes,packetsNb);
for sj=1:IntNodes
    [SortedTnowg(sj,i),OrderTnowg(sj,i)]=sort(Tnowg(sj,:));
end

Net2TotalDelay=zeros(IntNodes,packetsNb);
for jk=1:IntNodes
    for packetk=1:packetsNb
        Net2TotalDelay(jk,packetk)=Tnowg(jk,packetk)-P(OrderTnow(jk,packetk));
    end
end

Netw2AverageDelay=zeros(1,packetsNb);
Netw2AverageDelay=mean(Net2TotalDelay);

Netw2Avg=0;
Netw2Avg=mean(Netw2AverageDelay)

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Multicast One-Stage Network %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Tnowg_win=zeros(Q,packetsNb);
for wj=1:Q
    for b=1:packetsNb
        servicew2(wj,b)=-1/ga*log(1-rand);

        ssDelay(wj,b)=0;
        if (b>1)&(OrderTnow_win(wj,b)<OrderTnow_win(wj,(b-1)))
            ssDelay(wj,b)=Tnowg_win(wj,(b-1))-SortedTnow_win(wj,b);
            if ssDelay(wj,b)<0
                ssDelay(wj,b)=0;
            end
        end

        Tnowg_win(wj,b)=SortedTnow_win(wj,b)+ssDelay(wj,b)+servicew2(wj,b);
    end
end

SortedTnowg_win=zeros(Q,packetsNb);
OrderTnowg_win=zeros(Q,packetsNb);
for cj=1:Q
    [SortedTnowg_win(cj,i),OrderTnowg_win(cj,i)]=sort(Tnowg_win(cj,:));
end

Net2TotalDelay_win=zeros(Q,packetsNb);
for jw=1:Q
    for packetw=1:packetsNb
        Net2TotalDelay_win(jw,packetw)=Tnowg_win(jw,packetw)-
(OrderTnowg_win(jw,packetw));
    end
end

Netw2AverageDelay_win=zeros(1,packetsNb);
Netw2AverageDelay_win=mean(Net2TotalDelay_win);

Netw2Avg_win=0;
Netw2Avg_win=mean(Netw2AverageDelay_win)

%-----
s=input('Sigma = ');

```

```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Multicast Two-Stage Network %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

```

Nodes=IntNodes;
Tnowl=zeros(IntNodes,packetsNb);
for c=1:IntNodes
    if ((c==Nodes)&(TotNetUsers~=0))
        UsersNb=UsersNb+TotNetUsers;
    end

    for e=1:packetsNb
        for u=1:UsersNb
            stDelay(c,e,u)=0;
            if (e>1)&(OrderTnowg(c,e)<OrderTnowg(c,(e-1)))
                stDelay(c,e,u)=Tnowl(c,(e-1),u)-SortedTnowg(c,e);
                if stDelay(c,e,u)<0
                    stDelay(c,e,u)=0;
                end
            end

            sNACKdelay(c,e,u)=0;
            sTransmDelay(c,e,u)=0;
            srDelay(c,e,u)=0;

            if (pr(e)==1)
                sNACKdelay(c,e,u)=-1/s*log(1-rand);
                sTransmDelay(c,e,u)=-1/s*log(1-rand);
            end

            srDelay(c,e,u)=RetransNb*(sTransmDelay(c,e,u)+sNACKdelay(c,e,u));

            sservice(c,e,u)=-1/s*log(1-rand);

            Tnowl(c,e,u)=SortedTnowg(c,e)+srDelay(c,e,u)+sservice(c,e,u)+stDelay(c,e,u);
        end
    end
end
end

```

```

Net3TotalDelay=zeros(IntNodes,packetsNb);
for l=1:IntNodes
    for t=1:UsersNb
        for packetl=1:packetsNb
            if (Tnowl(l,packetl,t)~=0)
                Net3TotalDelay(l,packetl,t)=
                    Tnowl(l,packetl,t)-P(OrderTnow(l,OrderTnowg(l,packetl)));
            end
        end
    end
end
end

```

```

Netw3AverageDelay=zeros(1,packetsNb);
if (TotNetUsers~=0)
    for t=1:IntNodes
        tt=1;
        Netw3AvgDelay(t,1:packetsNb)=0;
        while (tt<=UsersNb)
            Netw3AvgDelay(t,1:packetsNb)=
                Netw3AvgDelay(t,1:packetsNb)+Net3TotalDelay(t,1:packetsNb,tt);
            tt=tt+1;
        end;
        if (t==IntNodes)
            Netw3AverageDelay(t,1:packetsNb)=Netw3AvgDelay(t,1:packetsNb)/UsersNb;
        else
            Netw3AverageDelay(t,1:packetsNb)=Netw3AvgDelay(t,1:packetsNb)/floor(Q/IntNodes);
        end
    end
else
    Netw3AverageDelay=mean(Net3TotalDelay);
end
end

```

```

PacketMaxTime=zeros(1,packetsNb);
PacketMaxTime=max(Tnow1);

Netw3Avg=0;
Netw3AvgLoad=0;

if (TotNetUsers~=0)
    Netw3Avg=mean(mean(Netw3AverageDelay))
    Netw3AvgLoad=Netw3Avg+maxload*IntNodes*2/Q+maxload*(floor(Q/IntNodes))*2/Q
elseif (Q==IntNodes)
    Netw3Avg=mean(Netw3AverageDelay)
    Netw3AvgLoad=Netw3Avg+maxload*IntNodes*2/Q+maxload*(floor(Q/IntNodes))*2/Q
end

Netw3Avg=mean(mean(Netw3AverageDelay))
Netw3AvgLoad=Netw3Avg+maxload*IntNodes*2/Q+maxload*(floor(Q/IntNodes))*2/Q

PacketMaxT=max(max(PacketMaxTime))

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Multicast One-Stage Network %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
Tnow1_win=zeros(Q,packetsNb);
for d=1:Q
    for f=1:packetsNb
        sservicew3(d,f)=-1/s*log(1-rand);

        fstDelay(d,f)=0;

        if (f>1)&(OrderTnowg_win(d,f)<OrderTnowg_win(d,(f-1)))
            fstDelay(d,f)=Tnow1_win(d,(f-1))-SortedTnowg_win(d,f);
            if fstDelay(d,f)<0
                fstDelay(d,f)=0;
            end
        end

        Delayww(d,f)=0;
        NACKdelayww(d,f)=0;
        TransmDelayww(d,f)=0;

        if (pr(f)==1)
            NACKdelayww(d,f)=-1/s*log(1-rand);
            TransmDelayww(d,f)=-1/s*log(1-rand);
        end

        Delayww(d,f)=RetransNb*(TransmDelayww(d,f)+NACKdelayww(d,f));

        Tnow1_win(d,f)=
            SortedTnowg_win(d,f)+sservicew3(d,f)+2*Delayww(d,f)+fstDelay(d,f);
    end
end

Net3TotalDelay_win=zeros(Q,packetsNb);
for jwf=1:Q
    for packetwf=1:packetsNb
        Net3TotalDelay_win(jwf,packetwf)=
            Tnow1_win(jwf,packetwf)-P(OrderTnowg_win(jwf,OrderTnowg_win(jwf,packetwf)));
    end
end

Netw3AverageDelay_win=zeros(1,packetsNb);
Netw3AverageDelay_win=mean(Net3TotalDelay_win);

PacketMaxTime_win=zeros(1,packetsNb);
PacketMaxTime_win=max(Tnow1_win);

Netw3Avg_win=0;
Netw3Avg_win=mean(Netw3AverageDelay_win)

Netw3AvgLoad_win=0;

```

Netw3AvgLoad\_win=Netw3Avg\_win+maxload\*2

PacketMaxTimewin=max(max(PacketMaxTime\_win))