Performance Evaluation and Comparis~~i~~on of Ordinary, Adaptive and Exhaustive Service in the Token Ring Network

by

Darshakkumar J Patel

Thesis submitted to the Faculty of the Graduate School
of the New Jersey Institute of Technology in partial
fulfillment of the requirements for the degree of
Master of Science in Electrical Engineering
1990

APPROVAL SHEET

Title of Thesis: **Performance Evaluation and Comparison of Ordinary, Adaptive and Exhaustive Service in the Token Ring Network**

Name of Candidate: Darshakkumar J Patel
Master of Science in Electrical Engineering, 1990

Thesis & Abstract Approved
by the Examining Committee:

Dr. Constantine N. Manikopoulos, Advisor — Date
Associate Professor
Department of Electrical and Computer Engineering

Dr. Irving Wang — Date
Assistant Professor
Department of Electrical and Computer Engineering

Dr. George Antoniou — Date
Visiting Professor
Department of Electrical and Computer Engineering

New Jersey Institute of Technology, Newark, New Jersey.

VITA

Name: Darshakkumar J Patel

Degree and date to be conferred: M.S.E.E., 1990.

Secondary education: Sir C J N Z Madresha, Navsari, India

Collegiate institutions attended	Dates	Degree	Date of Degree
Gujarat University, India	1983-88	B.S.E.E.	August, 1986
New Jersey Institute of Technology	1988-90	M.S.E.E.	December, 1990

Major: Electrical Engineering

Abstract

In the token ring network different types of services gives different delay characteristics. Results to date have shown that in asymmetric traffic, exhaustive service gives more delay to lightly loaded stations where ordinary service wastes time in circulating the token after each transmission. In general, there is a need for more efficient service which is compromised between ordinary and exhaustive service.

Ordinary and exhaustive service are analyzed in this thesis, and a new service, adaptive service, is proposed. By using timer and counter, adaptive service dynamically changes token holding time at the station. Different types of delay characteristics are derived from their respective simulation models. The results indicate that proposed adaptive service has superior delay characteristics when compared with ordinary and adaptive service in asymmetric and symmetric traffic.

Acknowledgements

I wish to express my sincere thanks to Dr. Constantine N. Manikopoulos for his valuable guidance and encouragement throughout the course of this thesis. I am specially indebted to him for his insightful and constructive criticism, without which this thesis would never have been completed. I also thank the members of my thesis committee, Dr. Irving Wang and Dr. George Antoniou, for their suggestions and comments.

Last, but not least, I would like to thank all those who have helped me in one way or another during the course of this thesis.

Contents

Chapter 1

Chapter 2

Chapter 3

Chapter 4

Chapter 5

List of Figures

List of Tables

Chapter 1

INTRODUCTION

Computer communication network is a system of interconnected computer and other intelligent devices. Here, intelligent devices mean any device with some form of intelligence capable of sending or receiving information to or from other computers or intelligent devices. One reason for this interest is that it is cost effective to provide users with their own computer instead of having the users bring their work to a single computer located centrally. Other reasons include the sharing of expensive resources, electronic mail, the development of distributed processing, sharing of database, etc. These types of computer communication networks are classified on distance[1,2].

Local Area Network (**LAN**): This type of network is characterized by an inter-station distance in the order of magnitude of a few kilometers.

Metropolitan Area Network (MAN): An interstation distance of this type of network is in the order of a few meters to tens of kilometer.

Long Haul Network: An interstation distance for long haul network ranges up to hundred or thousands of kilometer.

1.1 Local Area Networks (LAN)

The local area network is typically used within an organization (e.g., university, office, bank, hospital) and is owned by that organization. The elements of a LAN are configured as an integrated system involving communication media, medium interfaces and communications software. All network station has appropriate medium interfaces and communication software for enabling them to communicate. The medium interfaces and communication software are highly structured using specific rules, formats, protocols and interfaces. One aspect of a LAN structure is its topology, with the most common ones being the tree, bus, ring and star. Some topologies have certain advantages over others[2].

Local area networks are being used in variety of applications that include data processing, office automation, factory automations, teleconferencing, computer room networks, etc. Certain parameters like response time, volume of data, environment, communication type and type of devices are useful characteristics in describing the use of networks in the various type of applications. The medium access protocol, which controls the access to the transmission system and prevents simultaneous transmission again and again is very important in LAN. Classification of LAN is based on the medium access protocols. Token passing networks which uses the token passing technique are kind of local area networks.

1.2 Token Passing Networks

Token passing networks have been used around for many years and have long been used for both local and wide area network. There are several advantages of token passing networks[3,4]:

1. They provide higher throughput than CSMA.
2. They provide priority services.
3. They operate efficiently at higher data rates.

The token passing operation is collision free operation so that it provides conflict free transmission among the stations. The stations are ordered to form a logical ring, according to which they gain access to the transmission medium. In the token passing operation, the token, which is a special kind of packet passes from station to station around the logical ring. When the station receives the token, it can transmit the packet or pass the token to the next station. In this way at a time only one station holds the token and transmits packets, so there isn't any possibility of collision in this type of operation.

There are two types of token passing operations:

1. *Explicit Token Passing*

2. *Implicit Token Passing*

In explicit token passing operation the token is a special packet which is uniquely identified by all stations.

In implicit token passing operation the token is a undertaken by a set of events that take place in the channel.

Token Ring and Token Bus are explicit token passing networks where Expressnet and Fasnet are implicit token passing networks.

1.3 Network Performance

Two measures of network performance are commonly used:

1. *Delay*:

Message Delay: Message delay is measured as the time elapsing since the message was queued at the sending station to the moment the entire message is successfully received at the destination.

Packet Delay: Packet delay is measured as the time elapsing since the packet became ready to be transmitted (the queuing time is excluded) to the moment the entire packet is successfully received at the destination.

2. *Throughput*: Throughput of the network is total rate of the data being trans-

mitted on the network.

All these performances are calculated as their average values but some time the peak values are also measured. In some real time applications, there are certain bounds for delay, where peak delay is an important parameter. There is one more interesting parameter called offered load which works as an input parameter for describing the network characteristics. *Offered load* is the total rate of data presented to the network for the transmission.

Factors affecting the performance are as follows:

1. Capacity of the channel
2. Propagation delay of physical medium
3. Packet length
4. Local network protocols
5. Number of stations
6. Different kind of services

All these different factors give an interesting characteristics of the network.

1.4 Purpose of Study

As I discussed earlier, delay is very important parameter to justify network characteristics. Many approaches have been proposed to reduce the message and packet delays. Token holding time is the prime parameter which controls the message queue at the station. Different services define different token holding time to access the channel. I am very much interested in the effect of different services applied to the ring network.

Martini and Welzei[5] discussed delay effect of ordinary service in the token ring network. They did simulations for different type of applications on the network. Bux and Grillo[6] implemented same service in several token rings interconnected through bridges. Both of them did simulations for symmetric type of traffic. Here, symmetric traffic means every station attached on the network generates messages at the same rate.

Extensive work done by Chen and Bhuyan[7] in multiple token ring network. They did analytical study as well as simulations for both ordinary and exhaustive service. Also, they studied effect of packet length in different number of rings for symmetric traffic. Hardy, Radziejenski and Lo[8] proposed a new method in the token ring network, based on the use of stations with two latency status. The station is able to enter in a lower latency state whenever its message queue is empty. They present detailed evaluation and comparison of network performance with and without dual latency for different services in symmetric traffic. Analytical study and simulation for exhaustive service in asymmetric traffic were done by Ferguson and Aminetzah[9,10].

In this work, I am interested in the study of local delays (at individual station) and global delays(of the network) for different type of services. Both, exhaustive and ordinary service has several disadvantages for symmetric and asymmetric traffic. Because as traffic changes, they don't change their token holding strategies. Exhaustive service gives more local delay at lightly loaded stations, where ordinary service gives more global delay of the network.

The main objective behind this work is to propose and evaluate a new service,*Adaptive Service*, which is compromise between local delay and global delay. *Adaptive service dynamically changes token holding time(or packets per token) as load on the network changes at different stations, at different time.* An another purpose of this work is to compare the delays for all services with different types of traffic, different packet lengths and different number of stations.

1.5 Outline

An outline for the rest of this document is as follows.

In chapter 2, a brief theory of token passing operation, different type of services and principal of adaptive service is presented. Chapter 3, describes the simulation model for different kind of services.

Results of the simulations are presented in chapter 4. These results are presented in form of the graphs for two types of messages with different packet lengths and different number of stations. From the simulation results, conclusions are derived and presented in chapter 5.

Appendix A briefly describes LANSF, simulation package.

Appendix B describes input model for different types of traffic.

Appendix C describes the confidence intervals for graphs presented in chapter 4.

Chapter 2

TOKEN RING NETWORK

2.1 Token Ring Operation

Token ring is probably the oldest ring control technique, originally proposed in 1969 and referred to as the *Newhall Ring*[1]. It consists a set of station which are serially connected by the transmission medium. The token ring technique is based on the use of a small frame , called the token, that circulates around the ring when all stations are idle. A station wishing to transmit must wait until it detects the token passing by it. It then seizes the token by changing one bit in the token, which transform it from token to a start of the frame sequence for the frame. The station then appends and transmits the remainder of the fields needed to construct a frame.

There is no token on the ring after it is captured by a station, so other stations wishing to transmit must wait. After completion of the transmission, the station passes the token to its downstream neighbor. If next station has any messages in its buffer to transmit on the network, it seizes the token otherwise it just passes the token to the next station. In this way the token is passed from station to station in order to their physical connection exists. In general, when any packet passing by the station:

1. It may be token.
2. It may be datapacket.

If it is token:

a. If the station has any messages in its buffer to transmit, it seizes the token and transmit the packet.

b. If station's buffer is empty, it passes the token.

If it is data packet:

station checks its destination address.

a. If destination address is itself, it copies the data packet.

b. If destination address is other then itself then it passes the data packet.

When the station releases the token?

There are three type of operations for releasing the token[3].

1. *Single Message Operation*:

In this operation, the station transmits the token after receiving entire message of its last transmission.

2. *Single Token Operation*:

In this operation, station does not wait for entire message but as it gets header of its last transmission it releases the token.

3. *Multiple Token Operation*:

This operation is more quick then even single token operation. Station releases the token immediately after the completion of its last transmission.

In single message and single token operations message delay of the network will be more then multiple token operation. When the channel capacity is higher, at that time waiting time for receiving entire message or header will be significant in which station is sitting idle without transmitting any packet on to the channel.

2.2 Different Services in Token Ring Network

Token holding time is really critical issue to design fair token ring network. Token passing operations only determines when the token will be released by the station after transmitting the last packet. But how many packet should be transmitted is determine by different services like *Exhaustive*, *Gated*, *Ordinary*, and *Limited*.

1. **Exhaustive Service:**

In this service, the station transmits packets until its message queue will be empty and then it releases the token to the next station.

2. **Gated Services:**

In this service, the station can transmit only that messages which were in queue when the station captures the token.

So that, it is sure that when the station releases the token, the message queue at the station is empty in exhaustive service where in gated service it is not guaranteed. If there isn't any messages generation take place after capturing the token by the station, the queue will be empty in gated service.

When the traffic is asymmetric, this type of services give more delay to lightly loaded stations than heavily loaded stations. I will discuss this disadvantage in detail in the next section. One very serious disadvantage of exhaustive service is that if the station continuously generate messages than it will hold the token forever and not allow any other station to transmit. After considering all this disadvantages of exhaustive and gated service, ordinary service was proposed.

3. **Ordinary Service:**

In this service the station transmits at most one packet per token on the channel, if it has messages waiting in the queue.

In this way this service gives limit to transmit messages, so that it can not create problem of hogging the channel with its own transmission only. But still it gives problem to the asymmetric traffic in different way. If only one (or few) station is active on the network, then even that station can transmit only one packet. This reduces the total amount of useful bandwidth available for data transmission by circulating the token most of the time.

Limited Service:

In limited service station transmits fixed number of packets when it captures the token. Network operator or system manager decides number of packets per token. In this way ordinary service is also one kind of limited service with one packet per token. As for example, if network operator decides n packets per token than it transmit only n packets to the channel and than release the token. If the station has less then n packets than after transmitting all packets it will release the token.

2.2.1 Effect of Asymmetric Traffic

Here, I will discuss in detail about an effect of asymmetric traffic on the network with different type of services. An asymmetric traffic means some of the stations of the network are lightly loaded and some of them are highly loaded.

Let's assume that there are an average P packets in the queue at the lightly loaded station and Q packets in the queue at the highly loaded station when the station capture the token at moderate load.

So that $P << Q$, for asymmetric traffic.

In ordinary service, where only one packet will be transmitted per the token, most of the time spent in rotating the token. In this way every station gets more and more delay and especially highly loaded station's queue increases sharply. This increases overall delay(global delay) of the network sharply. No doubt, here lightly loaded stations are not getting disadvantages from highly loaded stations, but still highly loaded stations are getting delay due to the token rotation each time and increase both global and local(at highly loaded station) delay of the network.

Exhaustive service is the worst case for asymmetric load. Because as we know $P << Q$ and so that every time to decrease Q to zero(to be queue empty), it takes very long time. In this way every time it gives more and more delay to lightly loaded stations. In this case as load increases, P increases sharply. Due to this disadvantage to the lightly loaded stations, this service gives low global delay but very high local delay at lightly loaded stations.

Limited service also work as an ordinary service but rather than one packet, it sends more packets per token. This gives some what lower delay than the ordinary service because it reduces the token rotations. But still in this service token holding time does not change dynamically as load changes at different stations at different time.

2.3 Proposed Model of Adaptive Service

Let's consider a new service,*Adaptive Service*, which is compromise between exhaustive and ordinary service(or compromise between local and global delay in asymmetric traffic). This service does not work as limited service, but it dynamically changes token holding time(or packets per token) as load changes at different stations at different times.

Here is the detail of this service:

In this service every station has counter and timer. Counter counts queued messages at the station. When the station passes the token to the next station it resets its timer and when it gets the token it compares the time spent to getback in one rotation with ideal token rotation time. Here, ideal token rotation time is the time spent in rotating the token for one rotations on the network without transmitting any packet. So in this service the timer keeps track of the global activity of the network and the counter keeps track of the local activity at the station.

Now from the exhaustive service we know that if the station having long queue keeps the token for more time then it reduces the global message delay of the network. In otherwords, to reduce global delay of the network, token holding time should be proportional to the queued packet at the station.

Let's assume following:

$t^{j}_{ht,(i+1)}$	time to hold the token in $(i+1)^{th}$rotation at j^{th} station.
$Q^{j}_{(i+1)}$	queued packets in $(i+1)^{th}$ rotation at j^{th} station.
t_{pac}	time to transmit a packet on the channel.
$P^{j}_{tok,(i+1)}$	packets per token to transmit on the channel by j^{th} station in $(i+1)^{th}$ rotation.

We know that

$$P^{j}_{tok,(i+1)} = \frac{t^{j}_{ht,(i+1)}}{t_{pac}} \tag{2.1}$$

and from above discussion

$$t^{j}_{ht,(i+1)} \propto Q^{j}_{(i+1)} \tag{2.2}$$

From Equ(2.1) and (2.2)

$$P^{j}_{tok,(i+1)} \propto Q^{j}_{(i+1)} \tag{2.3}$$

With regard to the disadvantages of the exhaustive service, as we discussed earlier, the station holding the token has to keep track of the recent activity on the network. It means, if other stations on the network become active, it has to reduce its token holding time(or packets per token). Timer at every station keeps track of the recent activities on the network as following.

Let's assume following:

$t^{j}_{p,(i-1)}$ time j^{th} station passes the token to $(j+1)^{th}$ station in $(i-1)^{th}$ rotation.

$t^{j}_{r,i}$ time j^{th} station receives the token from $(j-1)^{th}$ station in i^{th} rotation.

t_{ideal} time taken by the token for one rotation without transmitting any packet.

$t^{j}_{total,i}$ total time spent in i^{th} rotation for j^{th} station

$$t^{j}_{total,i} = t^{j}_{r,i} - t^{j}_{p,(i-1)} \tag{2.4}$$

$t^{j}_{l,i}$ time token arrived late in i^{th} rotation for j^{th} station.

(time spent in transmitting packets on the channel in i^{th} rotation by other stations on the network).

$$t^{j}_{l,i} = t^{j}_{t,i} - t_{ideal} \tag{2.5}$$

L^{j}_{i} number of packet token arrived late in i^{th} rotation for j^{th} station.

$$L^{j}_{i} = \frac{t^{j}_{l,i}}{t_{pac}} \tag{2.6}$$

Here, L^{j}_{i} indicates the global activity of the network and as L^{j}_{i} increases, the station has to reduces its token holding time or packets per token. In this way by reducing $P^j_{tok,(i+1)}$ we can keep control on highly loaded queues.

$$P^j_{tok,(i+1)} \propto \frac{1}{L^j_i} \tag{2.7}$$

From Equ(2.3) and (2.7)

$$P^j_{tok,(i+1)} \propto \frac{Q^j_{(i+1)}}{L^j_i} \tag{2.8}$$

For the simple case, by considering equal global and local weight for Equ(2.8)

$$P^j_{tok,(i+1)} = \frac{Q^j_{(i+1)}}{L^j_i} \tag{2.9}$$

From the above Equ. we can see that numerator controls global delay and denominator controls local delay. In this way this service gives compromise between networks global delay as well as local delay at individual station. Moreover in this service all station has different token holding time and it changes dynamically as load changes dynamically at station at different time.

Chapter 3

Configuration and Simulation Model

How ordinary, adaptive and exhaustive services are implemented in simulation experiments using LANSF are described in this chapter. Input model describes configuration and assumptions of the network model. Next, protocol codes for different services are given. Atlast, delay measurement describes how delays are calculated in the model[11].

3.1 Input Model

In LANSF input data set for the simulator consists of a number of logically separate parts. The data file start with time section followed by configuration, traffic, prtocol-specific and exit section. Sample data file for the simulation is given in appendix B.

3.1.1 Time Section

Time section specifies the number of indivisible time units(ITUs) in the experimenter time units(ETUs). In our simulation model channel capacity is defined 10 Mbps. For this reason in our model ETU is defined as 10^7 ITU, which makes calculation of other parameters simple in reference to ETU.

3.1.2 Configuration Section

Configuration section defines the network backbone as follows:

a. Number of station

b. Port allocation

c. Number of links

d. Port assigments

f. Distance matrix

In our model simulations are done for 8 and 12 stations in the ring. Every station has two ports: input port and output port, through which stations are interconnected by links. For 8(and 12) stations 8(and 12) links form the ring type of network. We defined earlier that 10 Mbps is channel capacity and ETU = 10^7 ITU and if we assume that each link has same characteristics then the rate of transmission is 1bit/ITU. Distance between two stations are expressed as the time of propagating a signal from one port to the other.

3.1.3 Traffic Section

Each station has access to the queue of messages to be transmitted. The message queues are maintained by an external *daemon* called *client* in LANSF. Typically, a message generated and queued by the client at some station represents a sequence of bits to be transmitted to another station. The protocol is supposed to process the messages by removing them from the queue and transmitting over the network to their proper destination.

The traffic pattern specified as a set of:

a. Options

b. Inter arrival time

c. Message length

d. Number of senders(receivers) with their weight

Combination of different options generate nonburst and burst traffic. By a burst we mean a number of messages that appear in the network at the same time(or almost same time). Burst are usually separated by periods of silence. To generate nonburst traffic:

a. message interarrival time may be exponentially or uniformly distributed.

b. message length may be exponentially or uniformly distributed.

For burst traffic same options are available for message interarrival time and message length within a burst and for burst itself:

a. interarrival time may be exponentially or uniformly distributed.

b. size(the number of messages within a burst) may be exponentially or uniformly distributed.

Interarrival time explicitly define load on the network. As we decrease the interarrival time between the messages(or bursts) load on the network increases. In this way by changing interarrival time we can vary load for selected range. All simulations are done for load range 1-7 Mbps. Simulations for nonburst traffic are done with different exponential interarrival time and uniform distributed message length of 2000 and 1000 bits. For burst traffic simulations are done with 10 ITU exponential interarrival time between messages, uniformly distributed message length of 2000 and 1000 bits, different exponential burst interarrival time for different load and uniformly distributed burst size of 20 messages.

Client needs two more parameters to complete the procedure of generating messages.

a. Source address(sender of message)

b. Destination address(receiver of message)

We had done simulation for both symmetric type of traffic and asymmetric type of traffic on the network. For symmetric type of traffic every station has same probability to be sender, so that messages generated by the client are distributed evenly to all stations. For asymmetric type of traffic on the network, we had defined two type of messages for different loads.:

a. For message type 1, interarrival is calculated in such a way, so that it will generate 35% of total load of the network. This messages are evenly distributed among 7(and 11) stations to form lightly loaded stations of the network.

b. For message type 2, interarrival time is calculated in such a way, so that it will generate 65% of the total load of the network. This messages are queued at only one station to form highly loaded station.

3.1.4 Protocol-specific and Exit Section

In this section, protocol-specific values like packet length, header and trailer information, token length, interpacket space and other necessary values are given. We had done simulations for fixed size of packets with 128 header bits and 32 trailer bits. Token length and interpacket space are specified as 24 bits and 16 bits.

Exit section describes the stop conditions for the simulation. Three limits can be declared to exit simulation.

a. Maximum number of messages

b. Virtual time limit

c. CPU time limit

We had done each simulation for total 20,000 messages on the network.

3.2 Protocol Codes for Simulation Model

In LANSF, protocol is expressed by program consists of two C files. One file contains mainly declarations of user defined symbolic constants and other file contains code of different processes. Here, we are mainly interested in the code of channel access to transmit packets for different services.

3.2.1 Ordinary and Exhaustive Service

Ordinary and exhaustive service are explained by examining partial pseudo-code of the program shown below:

```
Case  TRANSMIT_OWN_PACKET:
          if (any message is in the queue, then get the first, add header and trailer
            and store it in packet buffer) then
            begin
                transmit packet to the output port;
                continue at case PACKET_TRANSMITTED;
            end
          else
            continue at case PASS_TOKEN;
Case  PASS_TOKEN:
          transmit token to the output port;
          continue at case TOKEN_PASSED;
Case  TOKEN_PASSED:
          stop transfer at output port;
          continue at case INITIALIZE;
Case  PACKET_TRANSMITTED
          stop transfer at output port;
          release packet buffer;
          wait for interpacket space;
          continue at case PASS_TOKEN;
```

Above code shows how ordinary service is implemented by transmitting only one packet per token. All stations on the network has same type of codes by which they access the channel to transmit packets. Different functions, to get packet from queue, to transmit packet from output port to another station, to calculate time for delays, to release buffer and others are complicated and described in [11], although we are not interested in real programming.

Exhaustive service has almost same type of code. Only one case is different.

```
Case PACKET_TRANSMITTED:
    stop transfer at output port;
    release packet buffer;
    wait for interpacket space;
    continue at case TRANSMIT_OWN_PACKET again;
```

In this way in exhaustive service, the program will be in loop and will exit from the loop only when there is no message in the queue at the station.

3.2.2 Adaptive service

Partial pseudo code of the program for adaptive service to access the channel is as follows:

```
Case TRANSMIT_OWN_PACKET:
        if (timer is grater than zero) then
          begin
                get total token rotation time by deducting timer from current time;
                get time token arrived late by deducting ideal rotation time from
                total token rotation time;
                if (time token arrived late is zero) then
                  begin
                        if (counter is less then 20) then
                                transmit all packets;
                        else
                                20 packets per token;
                  end
                else
                  begin
                        convert time token arrived late in packets token arrived late;
                        get packets per token by dividing counter to packets token
                        arrived late;
                  end;
          end
```

```
        else
              one packet per token;
        if (any message is in the queue, then get the first ,add header and trailer
           and store it in packet buffer) then
           begin
              transmit packet to the output port;
              continue at case PACKET_TRANSMITTED;
           end
        else
              continue at case PASS_TOKEN;
Case    PASS_TOKEN:
        transmit token to the output port;
        continue at case TOKEN_PASSED;
Case    TOKEN_PASSED:
        stop transfer at output port;
        note current time into timer;
        reset the subcounter;
        continue at case INITIALIZE;
Case    PACKET_TRANSMITTED
        stop transfer at output port;
```

```
        release packet buffer;
        increase subcounter by one;
        decrease counter by one;
          if (subcounter equals to packets per token) then
              wait for interpacket space and continue at case PASS_TOKEN;
          else
              wait for interpacket space and continue at case TRANSMIT_AGAIN;
Case  TRANSMIT_AGAIN:
        if (any message is in the queue, then get the first ,add header and trailer
          and store it in packet buffer) then
          begin
              transmit packet to the output port;
              continue at case PACKET_TRANSMITTED;
          end
        else
              continue at case PASS_TOKEN;
```

First part of the case TRANSMIT_OWN_PACKET calculates packets per token. Two assumptions are made here:

1. For very first round every station can send atmost one packet per token.
2. If only one station is active on the network then even it cannot send more

then 20 packets per token.

In this way for every rotation this service checks recent global activity by timer and recent local activity by counter and then calculates packets per token.

3.3 Delay Measurement

Two delay measures are:

1. The *absolute message delay* of message M, denoted by $d_s(\mathrm{M})$, is measured as the time elapsing since the message was queued at the sending node to the moment the entire message (its last packet) is successfully received at the destination.

2. The *absolute packet delay* of packet P, denoted by $d_p(\mathrm{P})$ is measured as the time elapsing since the packet became ready to be transmitted (the queuing time is excluded) to the moment the entire packet is successfully received at its destination.

To define the above listed measures formally and to explain how the parameters of their distribution are computed, assume that we have a sequence of messages M^1,...,M^n and that message M^j consists of packets P_1^j,.. .,P_k^j with lengths l_1^j,...,l_K^j, respectively. Let $l^j = \sum_{i=1}^{k_j} l_i^j$ denote the length of M^j. Message M^j was queued at the sender at time tq^j, its i'th packet P_i^j became ready for transmission at tt_i^j and was completely received by the target station at tr_i^j. For the simulation all messages are assumed to be uniform length of 2000 and 1000 bits. The two delays mentioned above are calculated according to the following formulas:

$$d_s(M^j) = tr_{k_j}^k - tq^j \tag{3.1}$$

$$d_p(P_i^j) = tr_i^j - tt_i^j \tag{3.2}$$

The time when a packet becomes ready for transmission tt_i^j is determined as the maximum of the following two values.

1. the time when the buffer, the packet acquired into, was last released

2. the time when the message, the packet acquired from, was queued at the station.

The distribution parameters of the random variable representing the message delay of multiple messages transmitted over the network are calculated assuming that the random variable consists of discrete samples, namely, the message delay for the n messages M^1,...,M^n is computed as:

$$d_s^a(M^1, ..., M^n) = \frac{\sum_{i=1}^n d_s(M^i)}{n} \tag{3.3}$$

The absolute packet delay is interpreted in a similar way and the formula for determining the average delay is:

$$d_p^a(< P_i^j, ..., P_{k_j}^j >_{j=1}^n) = \frac{\sum_{j=1}^n \sum_{i=1}^{k_j} d_p(P_i^j)}{\sum_{j-1}^n k_j} \tag{3.4}$$

Chapter 4

ANALYSIS OF SIMULATION RESULTS

Results of the simulation are discussed in this chapter. Delay characteristics for various loads with different packet sizes and different number of stations are plotted and analyzed. For all results, delay time is expressed in milliseconds and load is expressed in megabits per second.

4.1 Simulation Parameters

Simulations are carried out for all services discussed in chapter 2. Simulations are done for 8 and 12 stations, all of them having similar characteristics. Two types of traffic are assumed: Symmetric and Asymmetric. In asymmetric traffic, 35% of the total traffic of the network is distributed among 7 stations(or 11 stations) to form lightly loaded stations. Again in symmetric and asymmetric traffic two more categories are defined: Nonburst and Burst. In burst traffic, burst of 20 messages appears on the network periodically. With exponential interarrival time, fixed message size and packet size of 2000 and 1000 bits are assumed. All simulations are done with channel capacity of 10 Mbits/second.

4.2 Discussion of Simulation Results

I will discuss the results of simulation in the order listed below:

1. Asymmetric nonburst traffic
2. Asymmetric burst traffic
3. Symmetric nonburst traffic
4. Symmetric burst traffic

First I will discuss the results of asymmetric nonburst traffic with 8 stations and 2000 bits/packet. Figure 4.1 shows the message delay Vs load at lightly loaded stations. As load increases curve of exhaustive service increases sharply because as load increases queue at lightly loaded station increases which takes significant time to be empty. In exhaustive service highly loaded station keeps token for more time and gives more delay to lightly loaded stations. In ordinary service station can send only one packet per token, so that every station keeps token for same time. In this way lightly loaded stations are not getting any disadvantage from highly loaded stations in ordinary services. In adaptive service timer and counter checks the recent activities on the network and doesn't allow to keep the token more time at highly loaded stations. From the graph we can see that it gives 50% improvement at moderate load on local delay compared to exhaustive service. Similar type of results for packet delay Vs load at lightly loaded stations are shown in figure 4.2.

Figure 4.3 shows the global message delay Vs load of the network. As we know, in ordinary service token rotates on the ring after each transmission which increases the global delay of the network where in exhaustive service token makes minimum rotations. From the graph it is clear that ordinary service gives more global delay than exhaustive service. We can see from the graph that adaptive service gives 70% improvement at moderate load on global delay compared to ordinary service.

Figure 4.1 shows that exhaustive service gives more local delay at lightly loaded stations and figure 4.3 shows that ordinary service gives more global delay of the network. From both figures, we can see that adaptive service gives compromise between the global and the local delay.

Figure 4.4 shows the message delay Vs load at lightly loaded stations for 12 stations and 2000 bits/packet. From figure 4.1 and 4.4 we can see that as number of stations increase on the ring, token rotation time increases and gives more delay to all services. Figure 4.5 shows similar characteristics for packet delay at lightly loaded stations. Figure 4.6 shows the global message delay Vs load of the network. It is clear from figure 4.4 and 4.6 that adaptive service gives improvement on both local and global delay compared to other services. Figure 4.7 and 4.8 show the results of asymmetric nonburst traffic on the network for 12 stations and 384 bits/packet.

Figure 4.9 and 4.10 show the results of asymmetric burst traffic on the network for 8 stations and 2000 bits/packet. In burst traffic, we can see an improvement of 30% at moderate load on the local delay compared to exhaustive service and an 80% improvement at moderate load on the global delay compared to ordinary service.

Figure 4.11 to 4.18 show graphs for symmetric traffic. Figure 4.11 and 4.12 show message delay Vs load and packet delay Vs load for nonburst traffic with 8 stations and 2000 bits/packet. For symmetric traffic local and global delays are same because load is evenly distributed among the stations. From the graph it is clear that for symmetric traffic adaptive service gives almost 20% improvement over ordinary service. As I discussed earlier, for more stations delays will be more . From figure 4.13 and 4.14 we can see that for 12 stations, message delay and packet delay are more than 8 stations.

Figure 4.15 shows the message delay for 8 stations and 1000 bits/packet where figure 4.16 shows the message delay for 12 stations and 384 bits/packet. Small packets give low delay at low load than big packets but at higher load small packets give more delay. We can see from figure 4.11 and 4.15 for low load 1000 bits/packet gives low delay than 2000 bits/packet but after 6 Mbps 1000 bits/packet gives more delay. Figure 4.17 and 4.18 shows the graphs for burst traffic on the network. We can see from graph 4.11 and 4.17 that for burst traffic delay is higher because in burst traffic number of messages appear on the network at the same time.

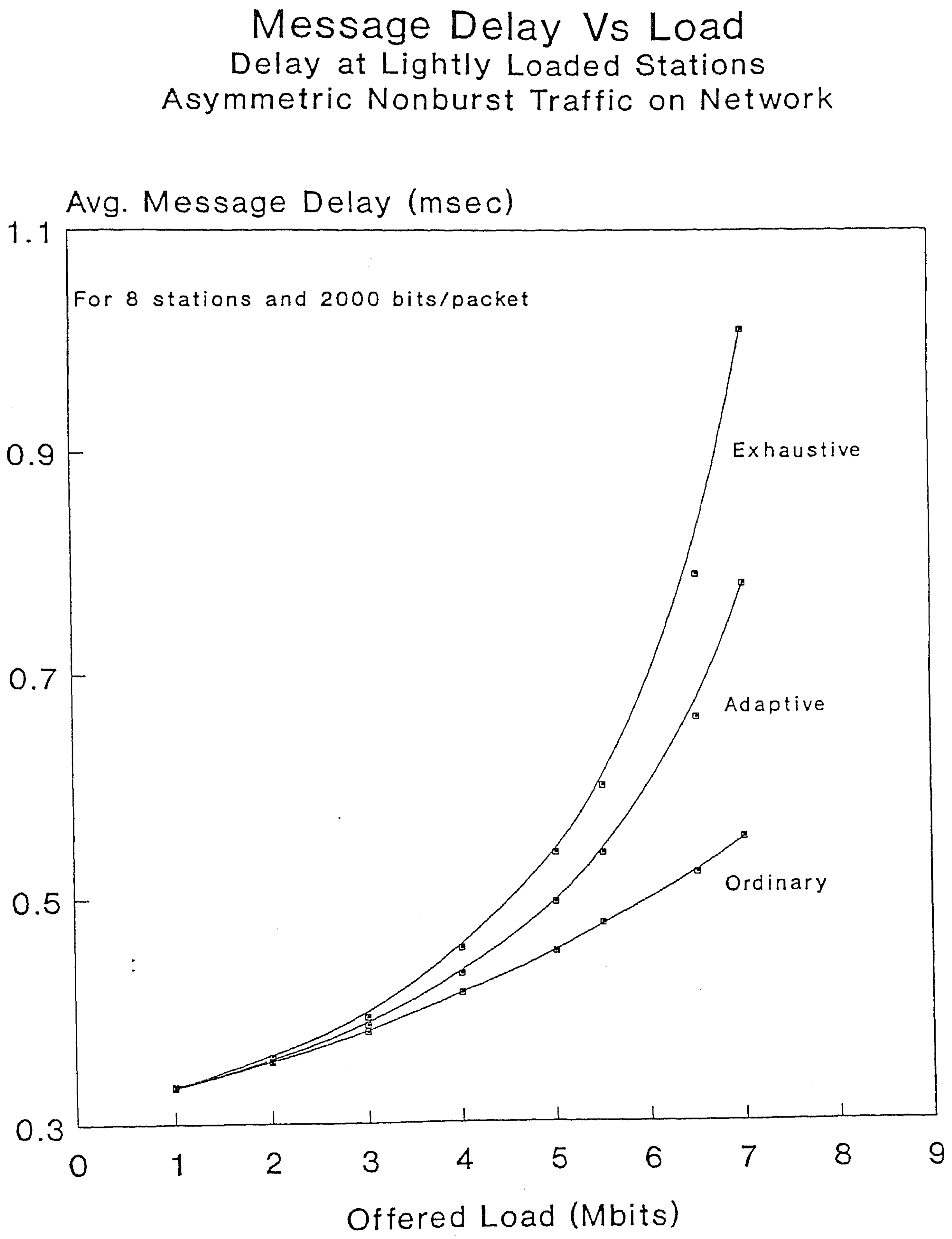

Figure 4.1: Local message delay Vs Load at lightly loaded stations for asymmetric nonburst traffic(8 stations, 2000 bits/packet)

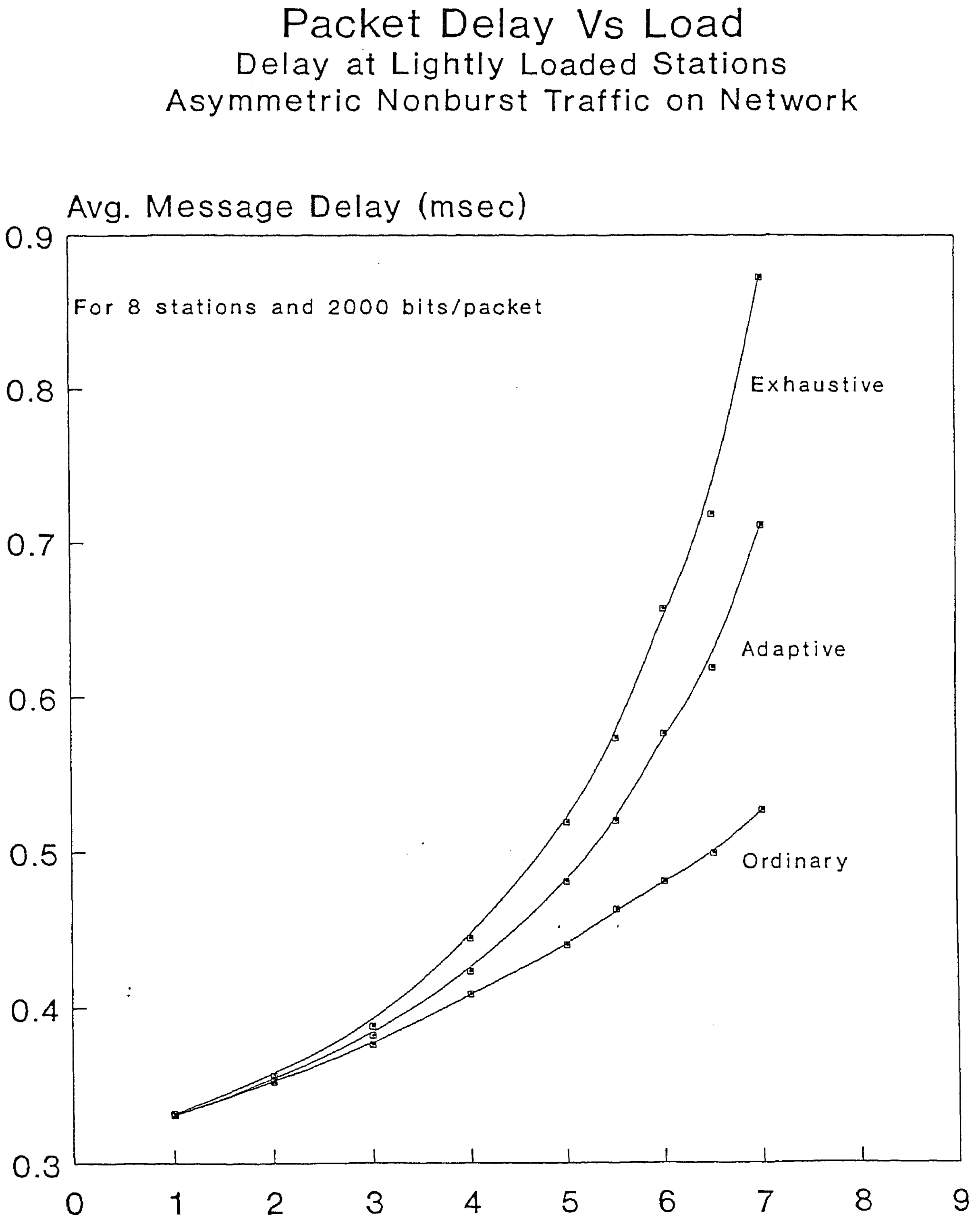

Figure 4.2: Local packet delay Vs Load at lightly loaded stations for asymmetric nonburst traffic(8 stations, 2000 bits/packet)

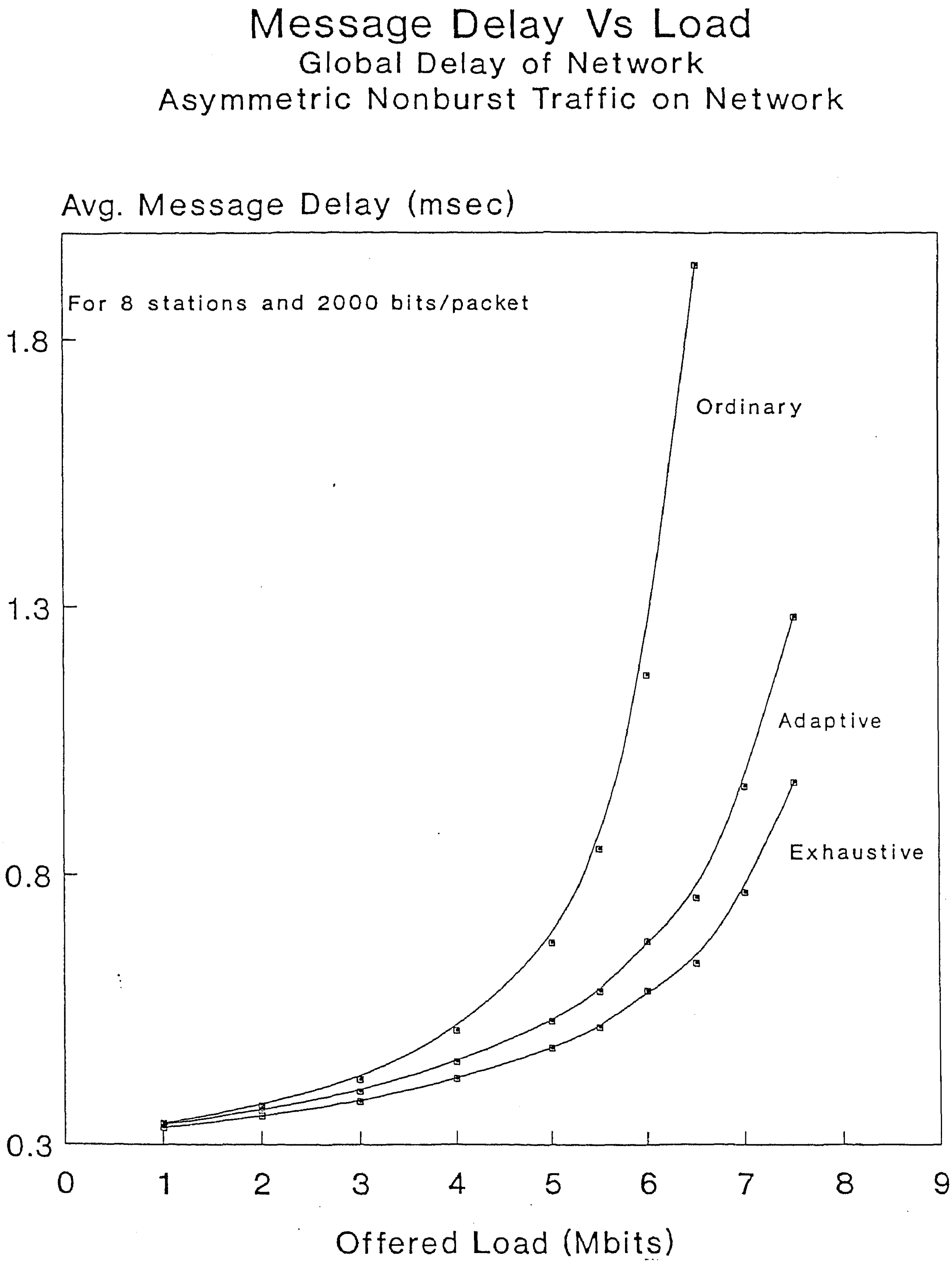

Figure 4.3: Global message delay Vs Load of the network for asymmetric nonburst traffic(8 stations, 2000 bits/packet)

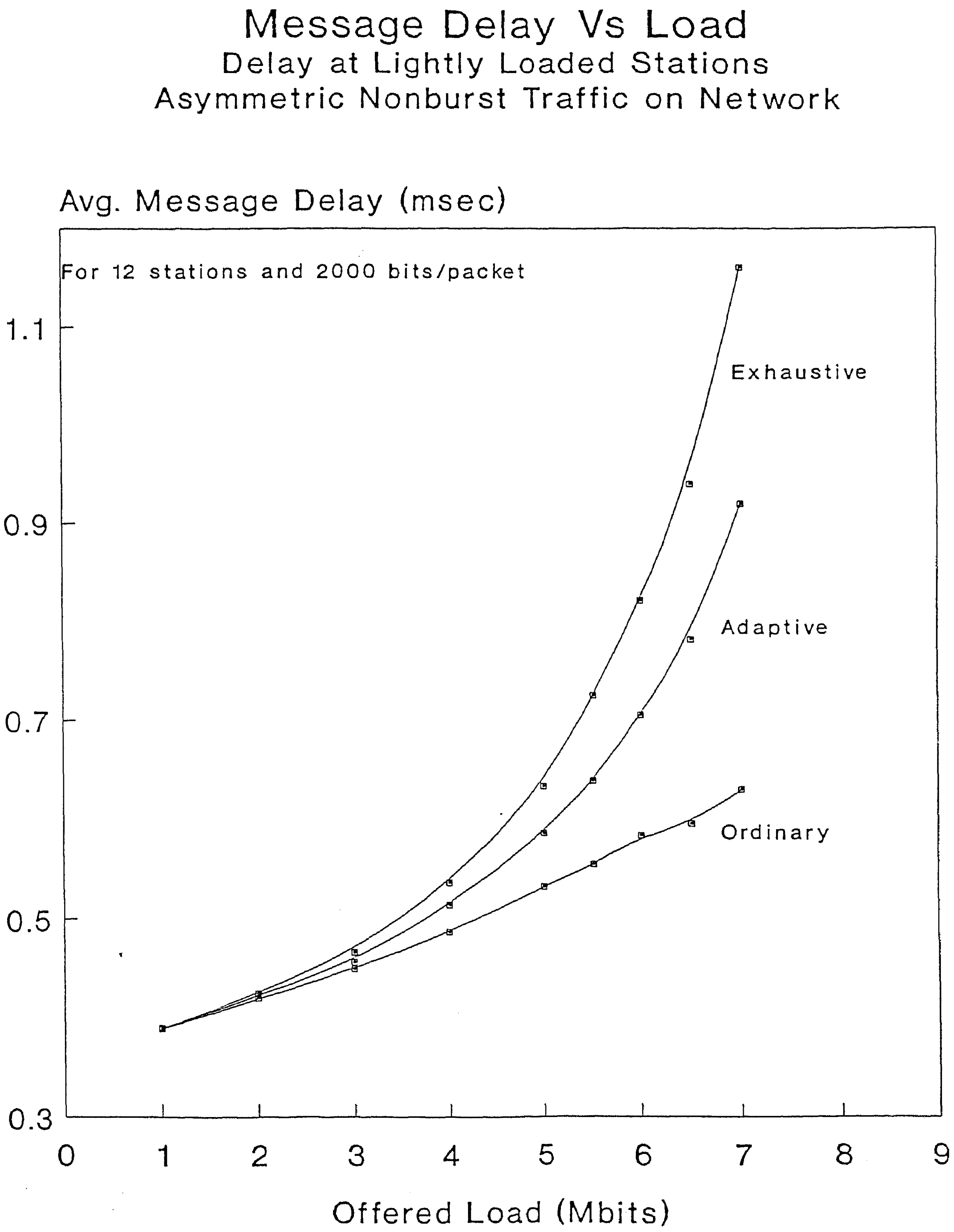

Figure 4.4: Local message delay Vs Load at lightly loaded stations for asymmetric nonburst traffic(12 stations, 2000 bits/packet)

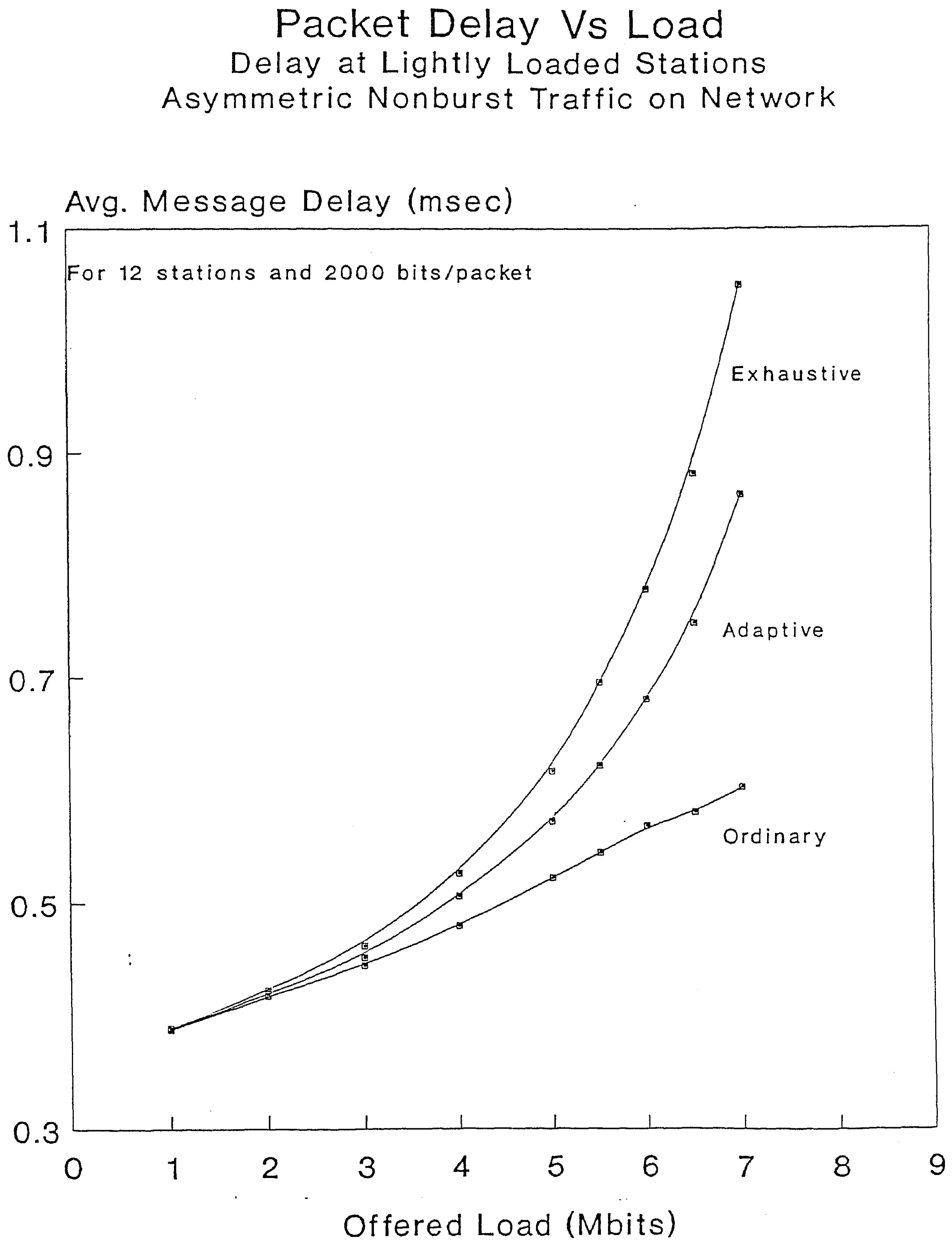

Figure 4.5: Local packet delay Vs Load at lightly loaded stations for asymmetric nonburst traffic(12 stations, 2000 bits/packet)

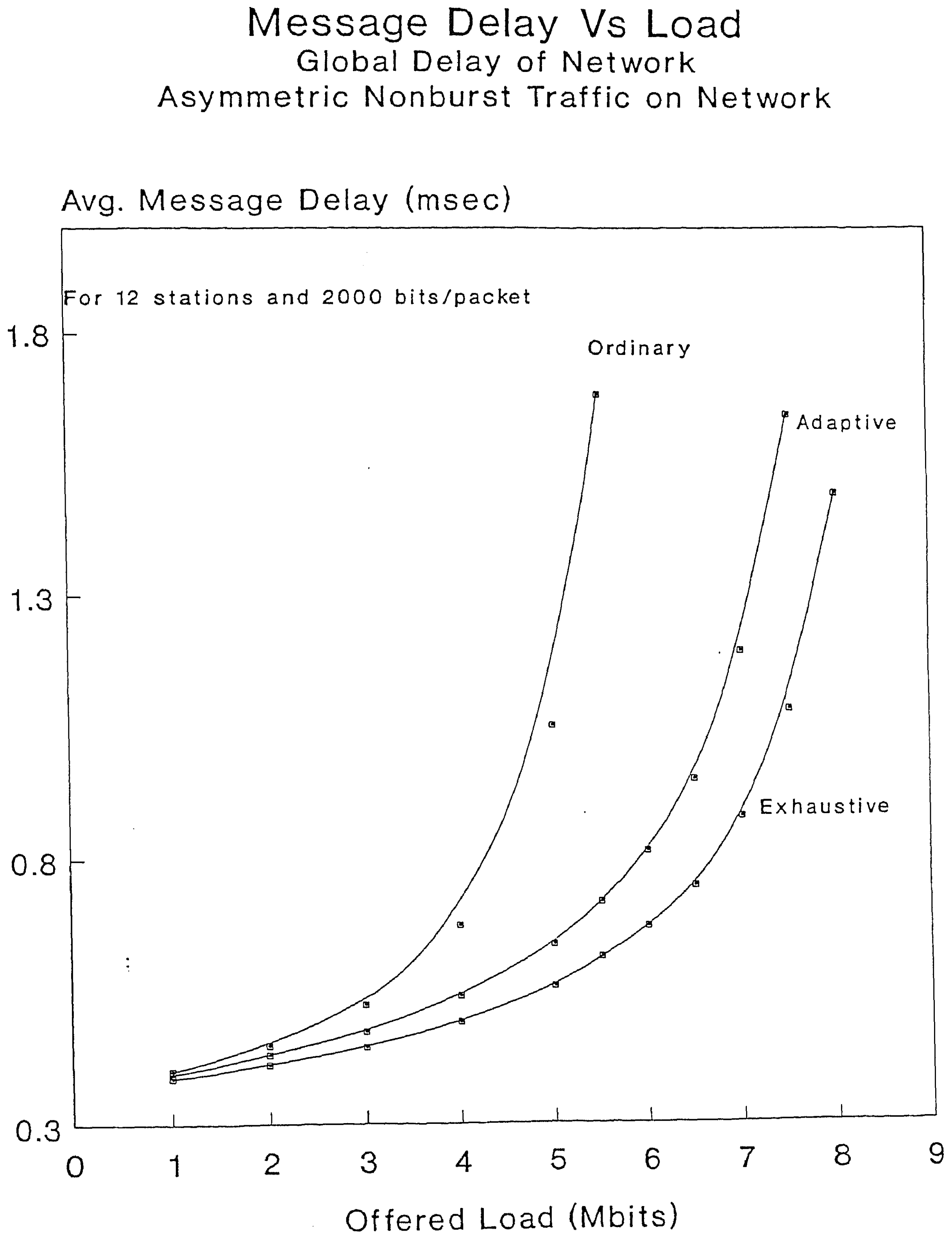

Figure 4.6: Global message delay Vs Load of the network for asymmetric nonburst traffic(12 stations, 2000 bits/packet)

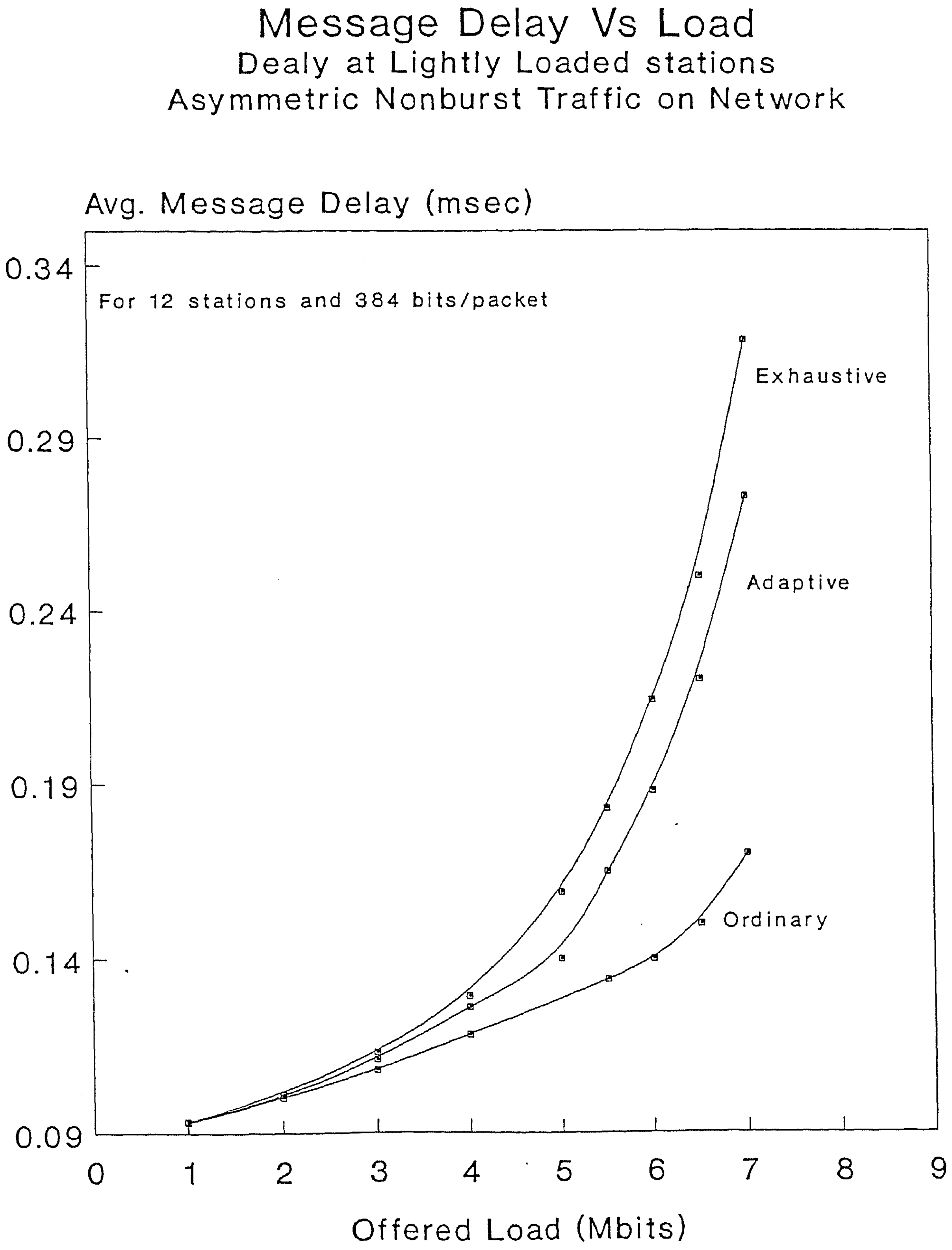

Figure 4.7: Local message delay Vs Load at lightly loaded stations for asymmetric nonburst traffic(12 stations, 384 bits/packet)

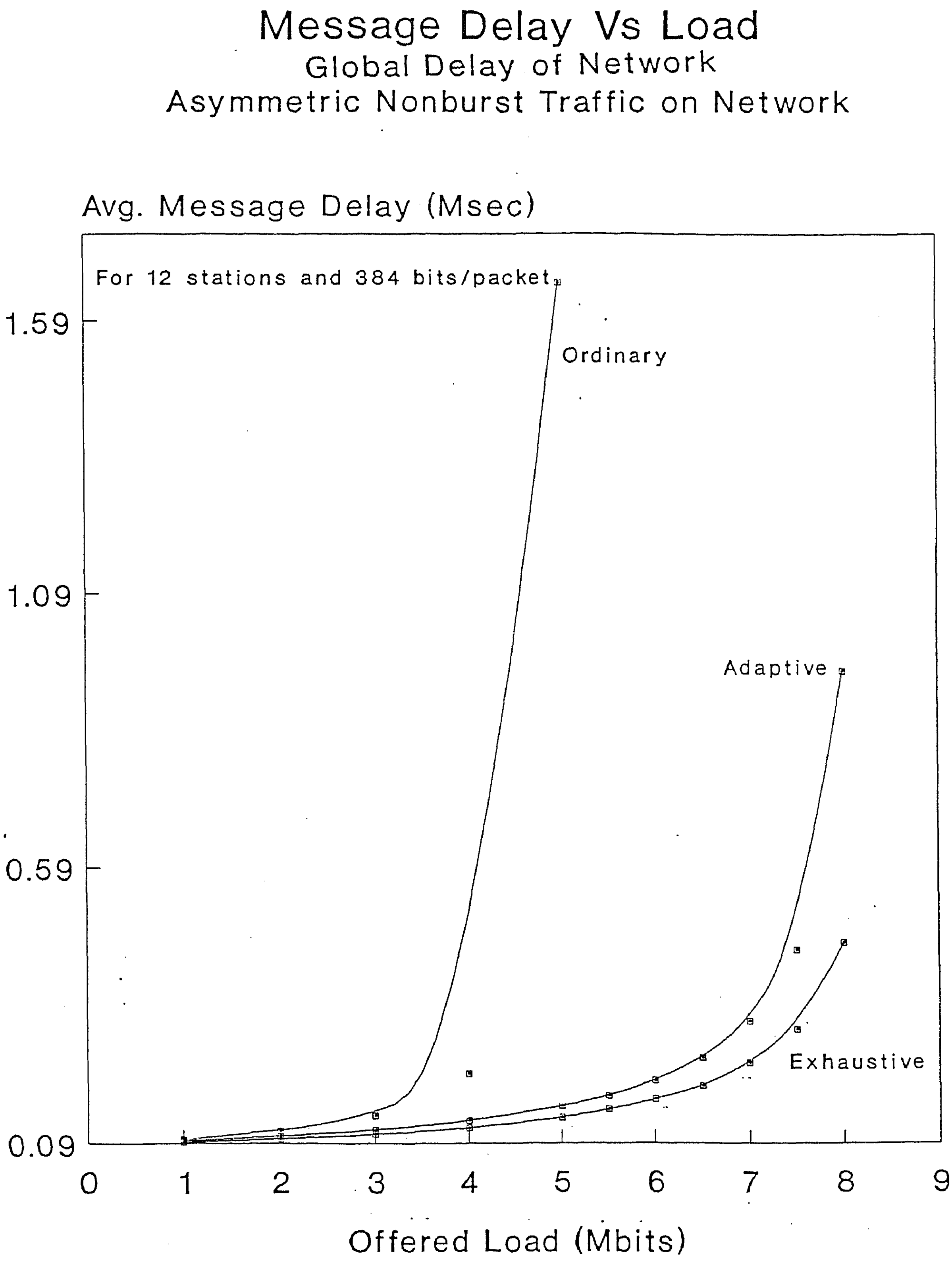

Figure 4.8: Global message delay Vs Load of the network for asymmetric nonburst traffic(12 stations, 384 bits/packet)

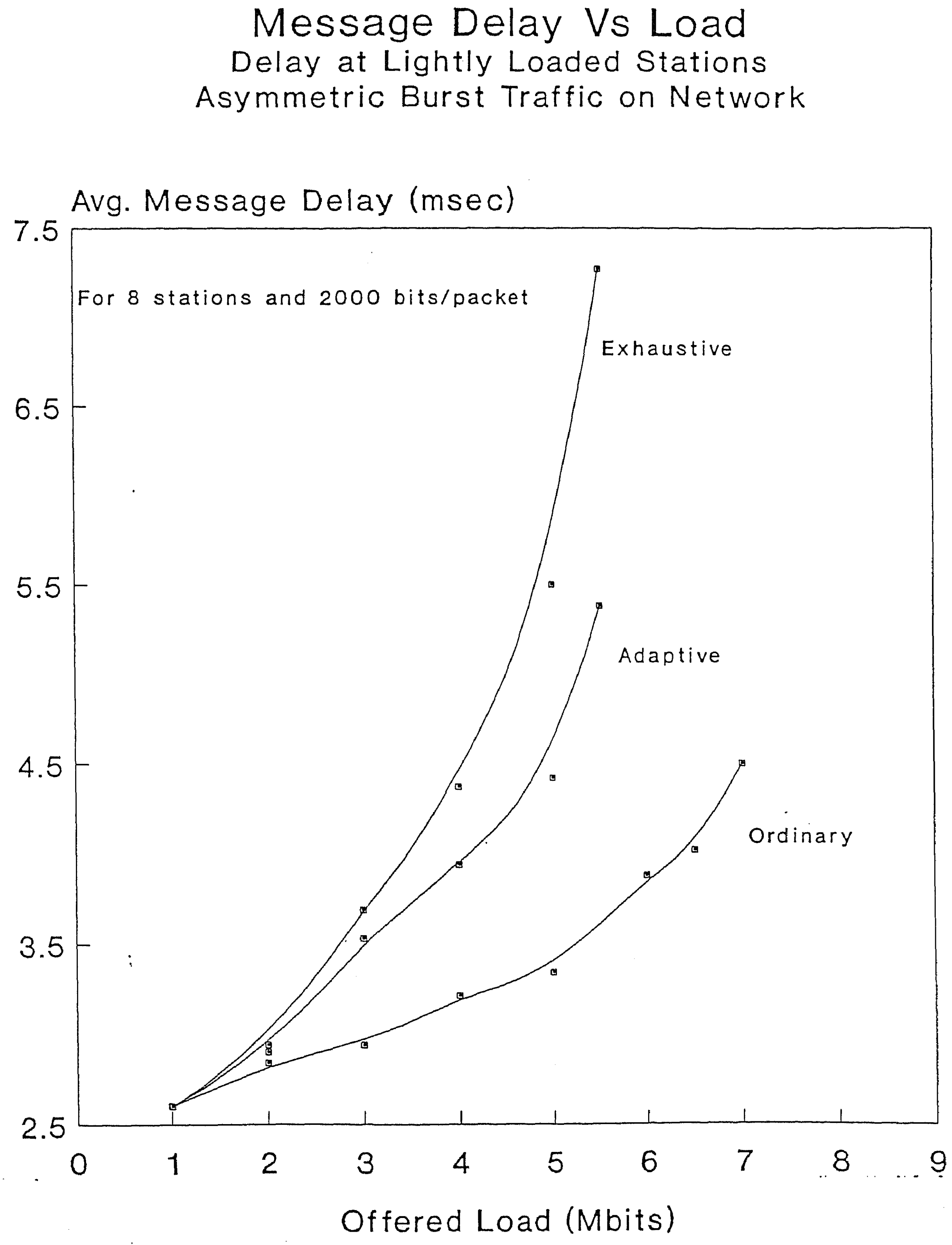

Figure 4.9: Local message delay Vs Load at lightly loaded stations for asymmetric burst traffic(8 stations, 2000 bits/packet)

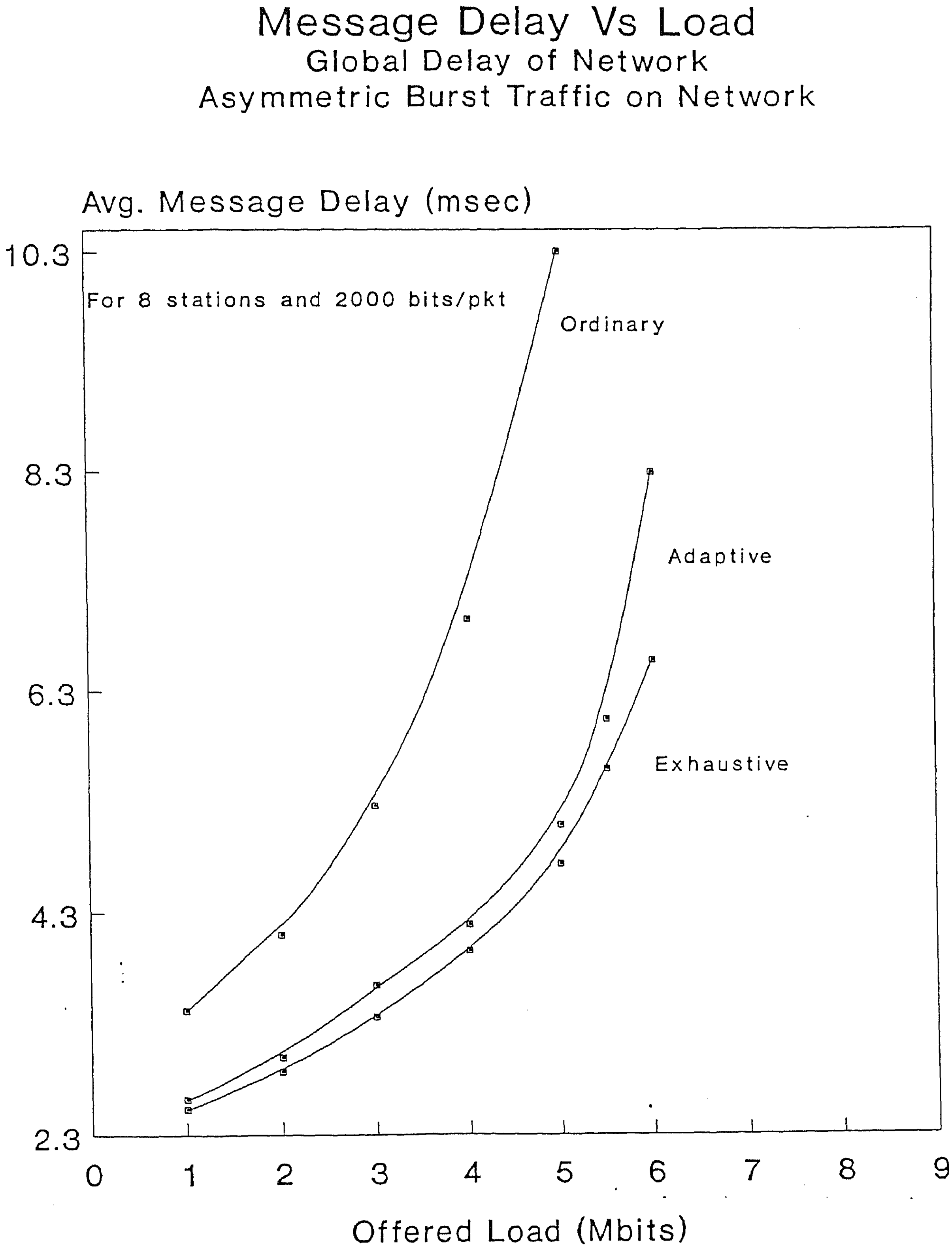

Figure 4.10: Global message delay Vs Load of the network for asymmetric burst traffic(8 stations, 2000 bits/packet)

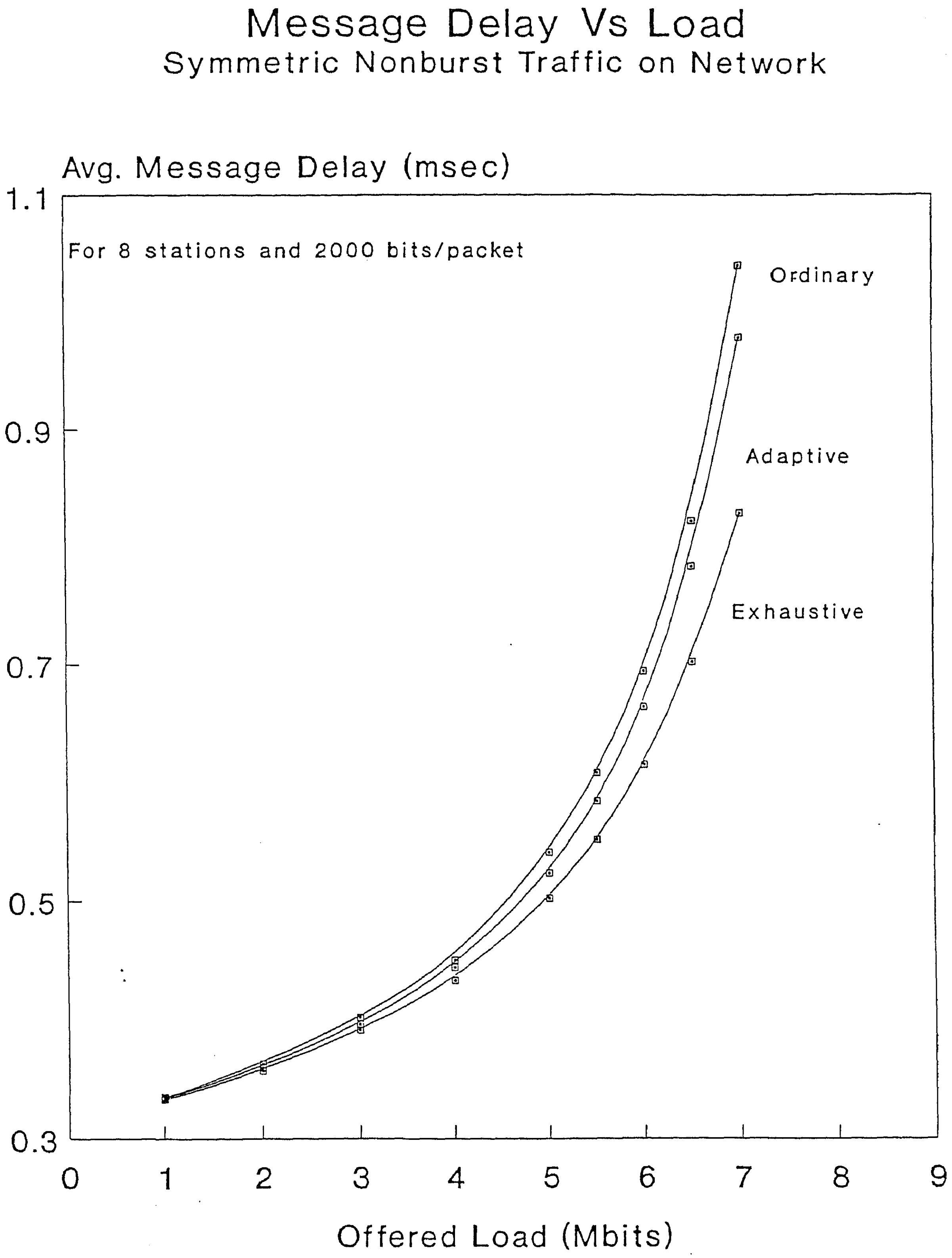

Figure 4.11: Message delay Vs Load for symmetric nonburst traffic(8 stations, 2000 bits/packet)

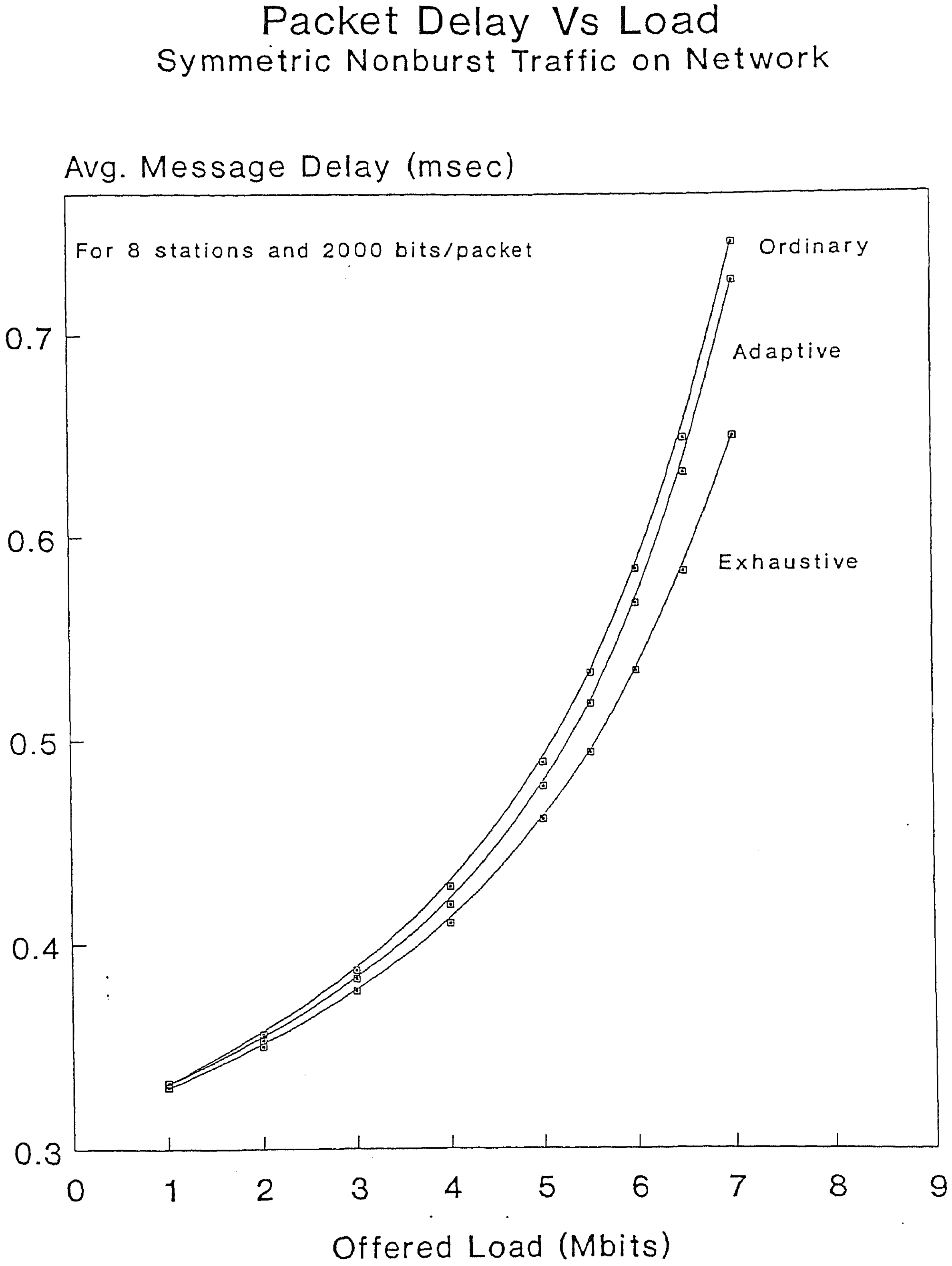

Figure 4.12: Packet delay Vs Load for symmetric nonburst traffic(8 stations, 2000 bits/packet)

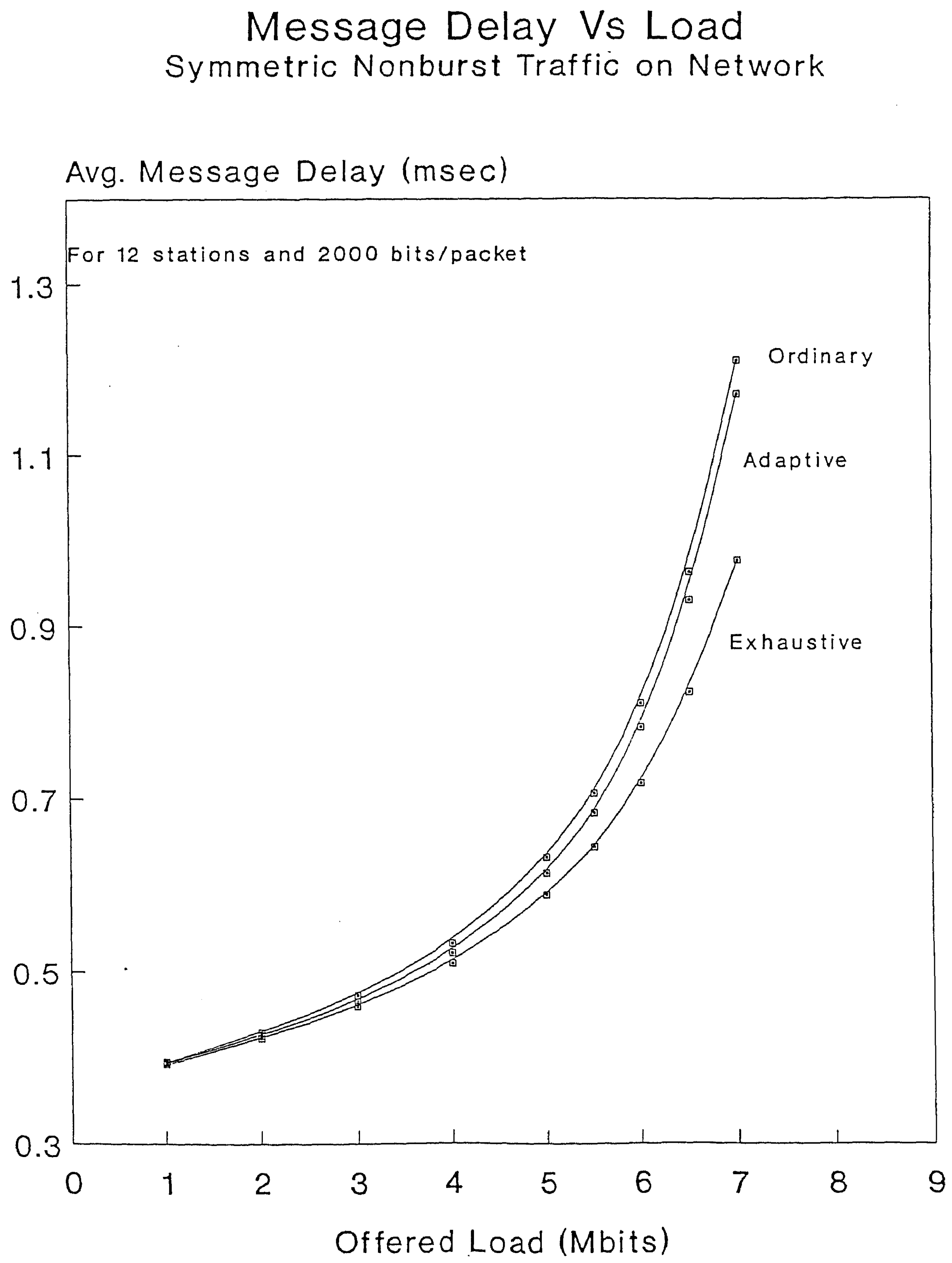

Figure 4.13: Message delay Vs Load for symmetric nonburst traffic(12 stations, 2000 bits/packet)

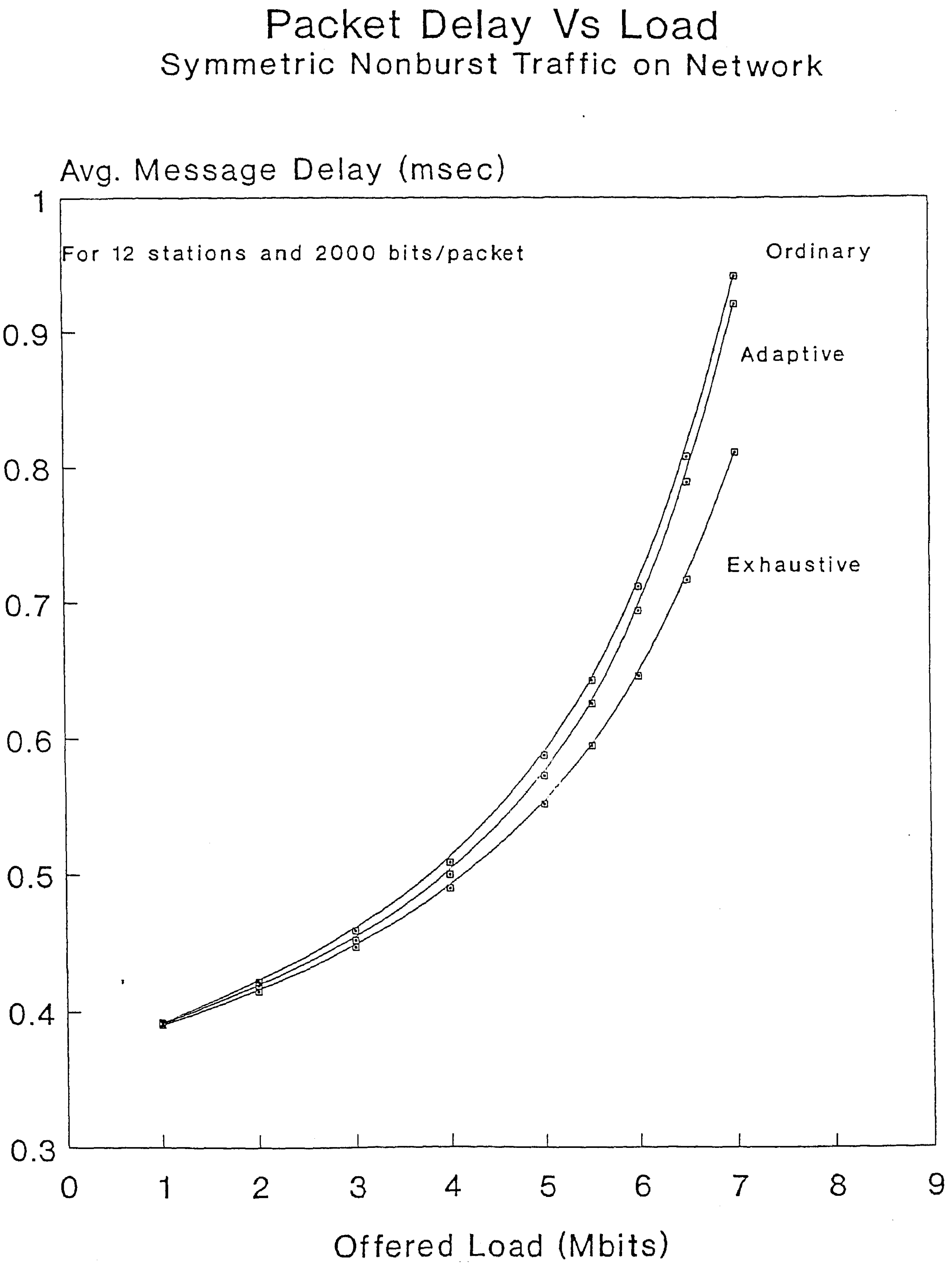

Figure 4.14: Packet delay Vs Load for symmetric nonburst traffic(12 stations, 2000 bits/packet)

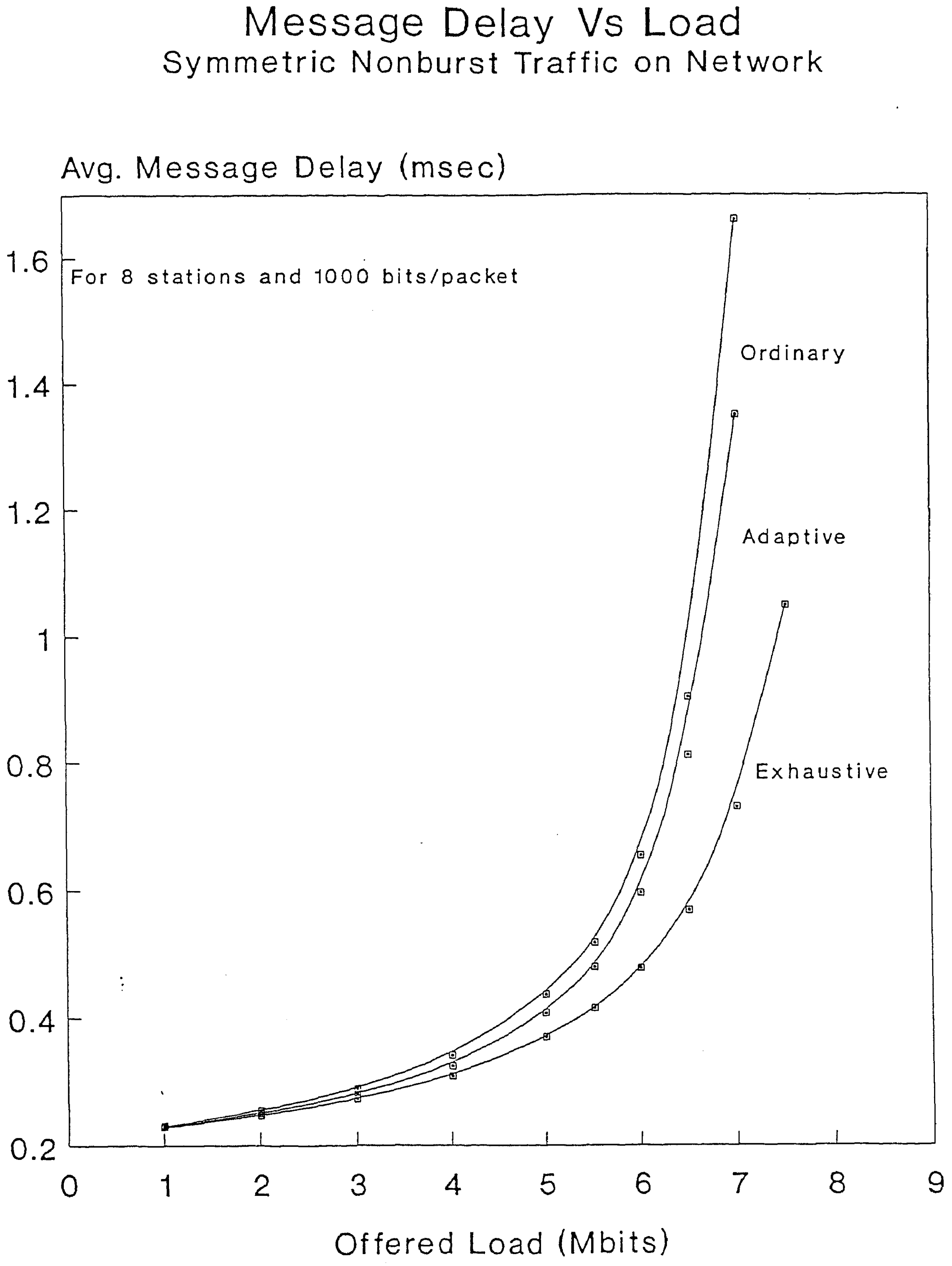

Figure 4.15: Message delay Vs Load for symmetric nonburst traffic(8 stations, 1000 bits/packet)

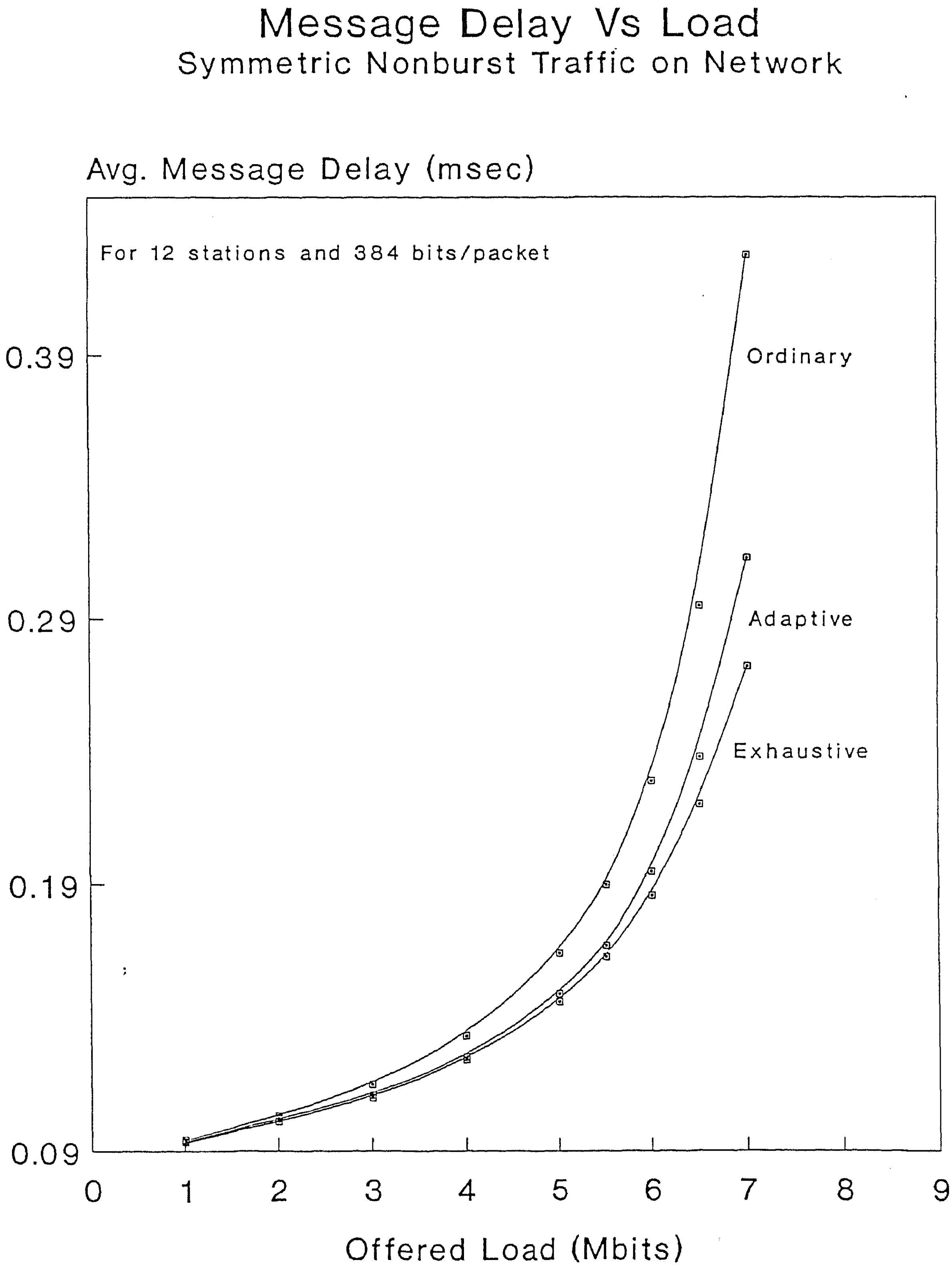

Figure 4.16: Message delay Vs Load for symmetric nonburst traffic(12 stations, 384 bits/packet)

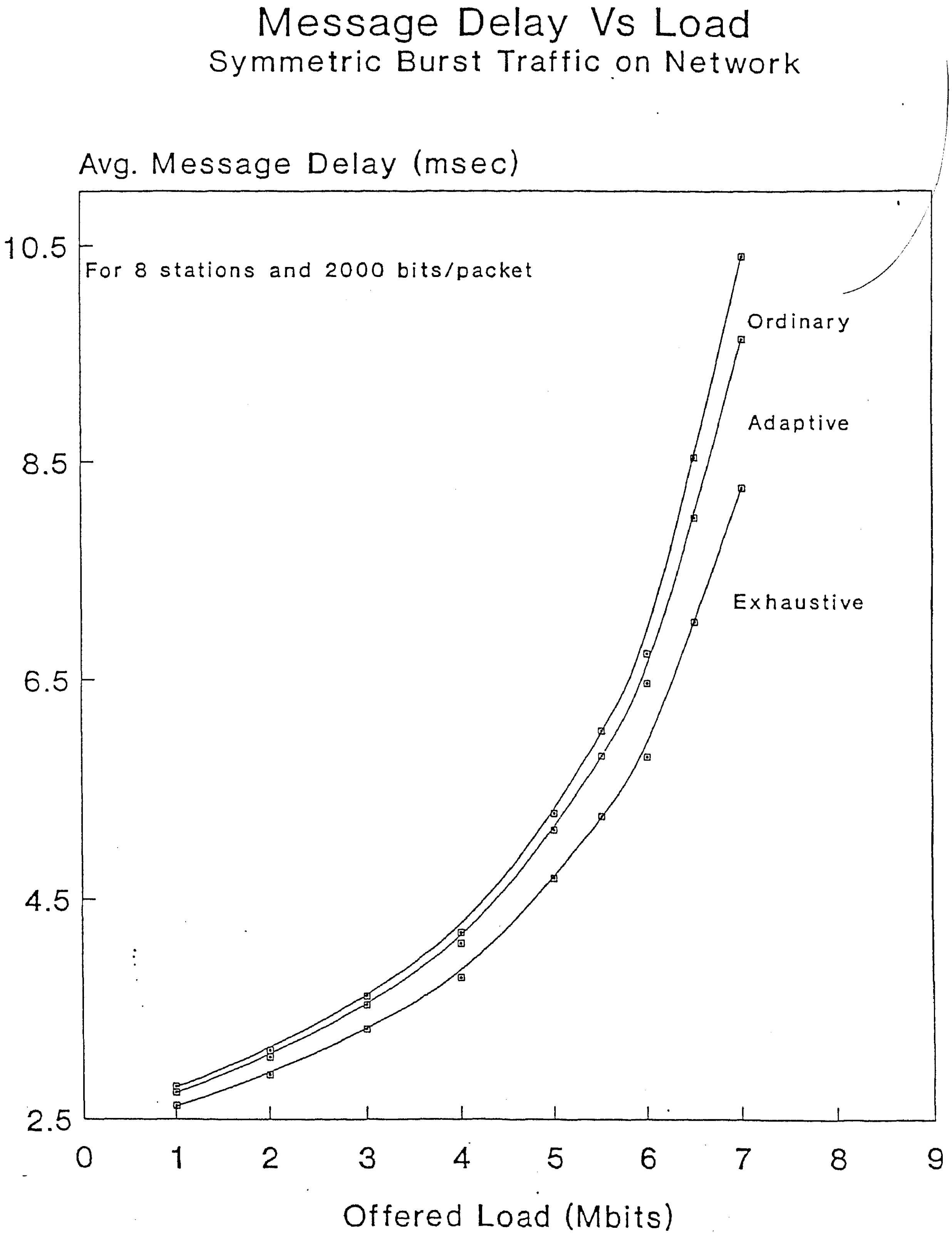

Figure 4.17: Message delay Vs Load for symmetric burst traffic(8 stations, 2000 bits/packet)

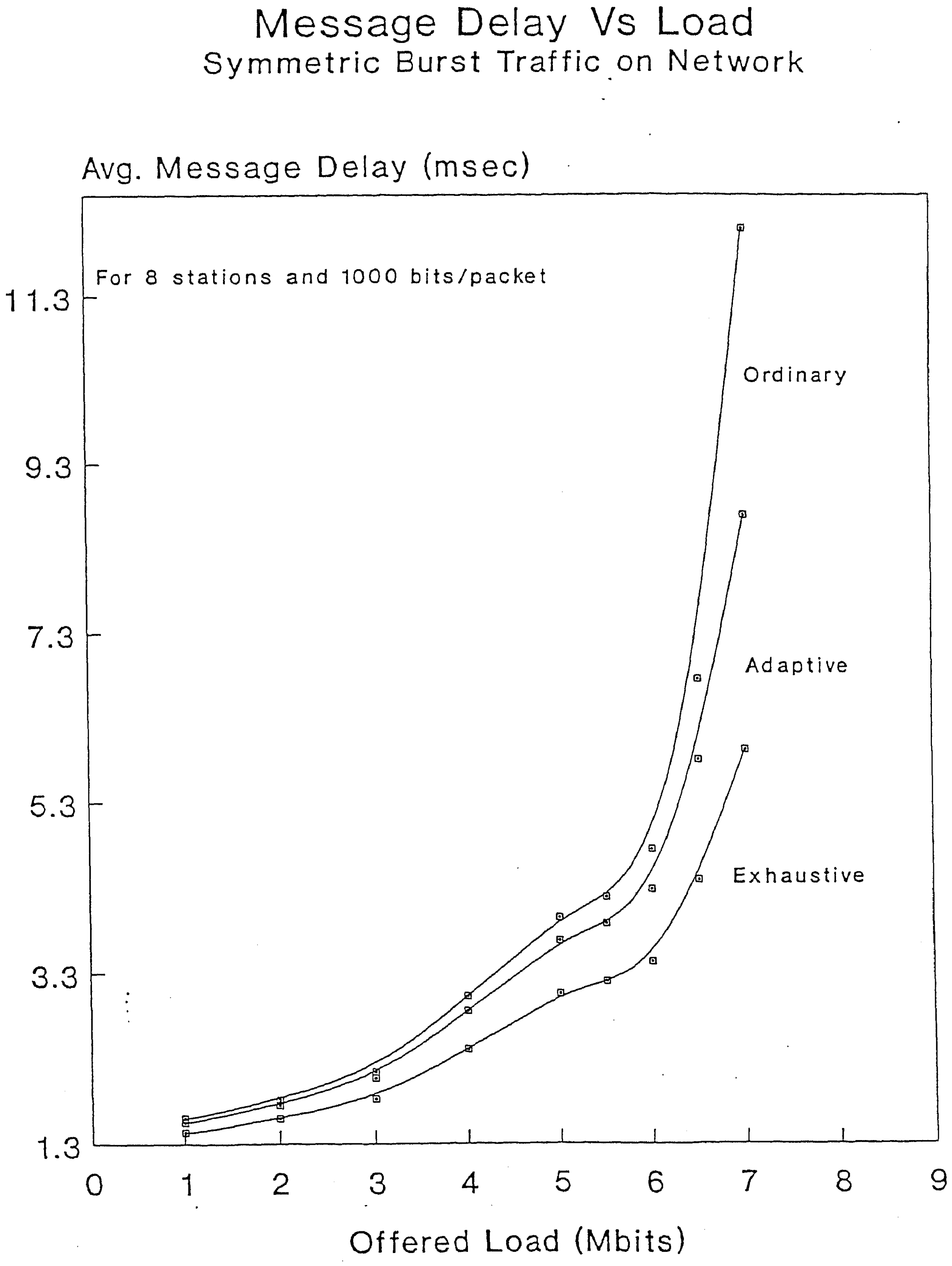

Figure 4.18: Message delay Vs Load for symmetric burst traffic(8 stations, 1000 bits/packet)

Chapter 5

CONCLUSION

A study of different services in the token ring network is presented in this thesis. The main purpose of this study was to investigate the delay characteristics of ordinary, exhaustive and adaptive service. For asymmetric traffic on the network, exhaustive service gives more local delay at the lightly loaded stations and ordinary service gives more global delay of the network. To this end, two currently existing services were analyzed and a new service, adaptive service, was proposed. By using timer and counter at every station, adaptive service keeps track of recent local and global activities on the network. It was shown that in asymmetric traffic, adaptive service gives 40-50% improvement at moderate load on the local delay compared to exhaustive service and gives 70-80% improvement at moderate load on the global delay compared to ordinary service. Also for symmetric traffic, it gives improvement of almost 20% at moderate load compared to an ordinary service. Moreover, adaptive service dynamically changes token holding time as load on the network changes at different stations, at different time.

In general, it can be said that adaptive service is compromise between ordinary service and exhaustive service or compromise between local delay and global delay in asymmetric traffic.

Appendix A

LANSF

LANSF is a configurable simulator designed to model a certain class of physical systems, namely communication networks. The attributes of a physical system that can be specified in LANSF are divided into two categories. The first category contains static elements, i.e. the system architecture and topology. the second category consists of dynamic attributes that describes the temporal behavior of the modeled system, i.e. the traffic pattern, and performance measures. Among the data communication elements available in LANSF are propagation delay, collision detection and data transfer. It also maintains event queues and statistics.

The simulation involves two tasks, system and protocol modelling and network configuration. System and protocol modelling requires C programming using the C-Library functions provided by LANSF. These tools are procedures and data structures expressed in C. Network configuration on the other hand, is specified in a data file which is interpreted by the designed system and protocol, this does not require any C programming. The programming interface to LANSF is UNIX/C. The program consists of four files

1. protocol.c
2. protocol.h
3. options.h
4. input data

A.1 Program File protocol.c

File protocol.c defines executable part of the protocol specification and functions representing protocol process executed by stations. It also has two other subroutines which must be included with the protocol module. The first extension function is in_protocol which performs a protocol specific initialization of the simulator. It starts with reading the values of the global protocol_specific parameters. In particular the values describing the propagation delay, the minimum and maximum packet length, the lengths of inter-packet space.The protocol specific parameters are read in the same order in which they occur in input data file using these three functions read_integer, read_real, read_big. the second extension function that must be in protocol.c is out_protocol. Its purpose is to write the protocol-specific parameters and result to the output file. The program file protocol.c consists of a number of simulated process running at each node specified in the input file. The execution of these C-functions is scheduled by the event handlers. Processes are awakened (scheduled) by either built in LANSF servers (TIMER,BUS events) or signals from other processes. The signalling mechanism provides a method for inter-process communication, and can be extended to simulate layered protocols as processes.

A.1.1 Header File protocol.h

The protocol.h file contains declarations of non-standard station attributes and the definitions of protocol-specific symbolic constants. The contents of protocol.h are inserted into the declaration of the structure STATION[7]. All variables defined in protocol.h are actually declared as attributes of STATION and made visible to protocol.c and a copy of this file should be present in each protocol directory.

A.1.2 Header File options.h

File optios.h has all the local options like precision of numbers, types of port variables representing port transmission rates, the length of additional information carried by messages and packets, types of transmission link (either ethernet type or unidirectional link) and number of moments to be calculated for standard statistics. A copy of this file should be present in each protocol subdirectory.

A.2 Input Data File

The data file starts with time section and configuration section which define the network backbone. It starts with number of stations followed by specification of number of ports for each station, Link number and type, total number of ports and its transmission rate, distance matrix describing the distance between the nodes, number of messages, message length, mean interarrival time, number of senders, receivers and optional flood group or broadcast type messages. The next section of the input data file consists of protocol-specific parameters. To read it LANSF calls the function in_protocl from program file protocol.c followed by exit conditions namely total number of messages to be generated, simulation time and CPU time limit.

Appendix B

INPUT MODEL

Input model describes configuration and assumptions of the network model.

B.1 Input Model

ETU = 1000000 ITU

Network Configuration:

Number of stations	12	
Ports	2/12	* Each station has 2 ports *
Number of Links	12	

** Link 0 is unidirectional links interconnecting station 0 and 1 **

Archive time	120		
Number of stations	2		
Port assignment	0	1	1
	1	0	1

Distance matrix:

@@	10

@@

Asymmetric nonburst traffic:

Number of message type 2

** Message type 0 ***

Options =0

Mean interarrival time =0.0057142

Minimum length =2000

Maximum length =2000

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 11, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

*** Message type 1 ***

Options =0

Mean interarrival time =0.0030769

Minimum length =2000

Maximum length =2000

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 1, stations (11,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Protocol specific parameters:

Minimum packet length =2000

Maximum packet length =2000

Header =128

Trailer =32

Token length =24

Token passing timeout =2000000

Bounds:

Maximum number of messages =20000

Virtual time limit =0

CPU time limit =0

Different types of traffic has different traffic sections.

Asymmetric burst traffic:

Number of message type 2

** Message type 0 ***

Options =1+4+8

Mean message interarrival time =0.000001

Minimum message length =2000

Maximum message length =2000

Mean burst interarrival time =0.1142857

Minimum burst size =20

Maximum burst size =20

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 11, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

*** Message type 1 ***

Options =1+4+8

Mean message interarrival time =0.000001

Minimum message length =2000

Maximum message length =2000

Mean burst interarrival time =0.0615384

Minimum burst size =20

Maximum burst size =20

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 1, stations (11,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Symmetric nonburst traffic:

Number of message type 1

** Message type 0 ***

Options =0

Mean interarrival time =0.002

Minimum length =2000

Maximum length =2000

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Symmetric burst traffic:

Number of message type 1

** Message type 0 ***

Options =1+4+8

Mean message interarrival time =0.000001

Minimum message length =2000

Maximum message length =2000

Mean burst interarrival time =0.04

Minimum burst size =20

Maximum burst size =20

Number of selection group =1

Number of flood group =0

** Selection group 0 ***

Number of senders 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Number of receivers 12, stations (0,1) (1,1) (2,1) (3,1)
(4,1) (5,1) (6,1) (7,1)
(8,1) (9,1) (10,1) (11,1)

Appendix C

Confidence Intervals

Confidence intervals can be effectively utilized to prove the validity of results obtained during the simulation run of a computer network[12]. Consider an $M/M/1$ queue with an arrival rate λ and a service rate μ. It is assumed that the arrival rate, λ, is Poisson, and that S_n is the time required to observe n arrivals. A $100(1-\alpha)$ percent confidence interval for λ is [12]:

$$(\lambda_L, \lambda_U) = \left[\frac{\chi^2_{2n;1-\alpha/2}}{2S_n}, \frac{\chi^2_{2n;\alpha/2}}{2S_n}\right] \tag{C.1}$$

$\chi^2_{2n;1-\alpha/2}$ and $\chi^2_{2n;\alpha/2}$ may be obtained from tables of the critical values of the χ^2 distribution (chi-square distribution) [12].

Let Y_m be the sum of m service times, then the $100(1-\alpha)$ percent confidence interval for μ is [12]:

$$(\mu_L, \mu_U) = \left[\frac{\chi^2_{2m;1-\alpha/2}}{2Y_m}, \frac{\chi^2_{2m;\alpha/2}}{2Y_m}\right] \tag{C.2}$$

As in the above case, $\chi^2_{2m;1-\alpha/2}$ and $\chi^2_{2m;\alpha/2}$ may be obtained from tables of the critical values of the χ^2 distribution (chi-square distribution) [12]. To obtain a $100(1-\alpha)$ percent confidence interval for the mean delay, w, we make use of the

F distribution, with $2m$ and $2n$ degrees of freedom [12]:

$$(w_L, w_U) = \left[\frac{w}{f_{2m,2n;\alpha/2}}, \frac{w}{f_{2m,2n;1-\alpha/2}}\right] \tag{C.3}$$

$f_{2m,2n;\alpha/2}$ may be obtained from tables of the critical values of the F distribution. $f_{2m,2n;1-\alpha/2}$ is the reciprocal of $f_{2m,2n;\alpha/2}$ [15]:

$$f_{2m,2n;1-\alpha/2} = \frac{1}{f_{2m,2n;\alpha/2}} \tag{C.4}$$

95% confidence intervals were derived for various plots. These are shown in the following tables. It should be noted that confidence intervals have not been derived at higher loads. The quantities in the denominators of equation (C.3) is equal to unity at higher loads, because a large number of packets have been sampled. Thus, the confidence interval is zero, and it may be concluded that enough samples have been taken to give an accurate point estimate of the quantity.

In the following tables, M is the message delay obtained from simulation. (M_L, M_U) is the derived 95% confidence interval, and $\frac{M_L+M_U}{2}$ is the mid-point of the confidence interval.

Similarly, P is the mean packet delay obtained from simulation, (P_L, P_U) is the derived 95% confidence interval, and $\frac{P_L+P_U}{2}$ is the mid-point of this interval. The validity of a result may be tested by comparing it to the mid-point of the confidence interval.

Table C.1 to C.10 shows confidence interval for some of the graphs plotted in chapter 4. Same type of confidence interval can be derived for remaining graphs.

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	0.354	(0.3278, 0.3823)	0.3550
	0.415	(0.3843, 0.4482)	0.4162
	0.477	(0.4417, 0.5152)	0.4784
Adaptive	0.355	(0.3287, 0.3834)	0.3561
	0.432	(0.4000, 0.4666)	0.4333
	0.539	(0.4991, 0.5821)	0.5406
Exhaustive	0.358	(0.3315, 0.3866)	0.3591
	0.455	(0.4213, 0.4914)	0.4563
	0.599	(0.5546, 0.6469)	0.6008

Table C.1: 95% Confidence Intervals for message delay in Figure 4.1

Service	P	(P_L, P_U)	$\frac{P_L+P_U}{2}$
Ordinary	0.352	(0.3259, 0.3802)	0.3530
	0.408	(0.3778, 0.4406)	0.4092
	0.463	(0.4287, 0.5000)	0.4644
Adaptive	0.353	(0.3269, 0.3812)	0.3540
	0.423	(0.3917, 0.4568)	0.4243
	0.52	(0.4815, 0.5616)	0.5150
Exhaustive	0.356	(0.3296, 0.3845)	0.3571
	0.444	(0.4111, 0.4795)	0.4453
	0.573	(0.5306, 0.6188)	0.5747

Table C.2: 95% Confidence Intervals for packet delay in Figure 4.2

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	0.372	(0.3444, 0.4018)	0.3731
	0.51	(0.4722, 0.5508)	0.5115
	1.17	(1.0833, 1.2636)	1.1735
Adaptive	0.363	(0.3361, 0.3920)	0.3641
	0.453	(0.4194, 0.4892)	0.4543
	0.676	(0.6259, 0.7301)	0.6780
Exhaustive	0.352	(0.3259, 0.3802)	0.3530
	0.421	(0.3898, 0.4547)	0.4222
	0.585	(0.5417, 0.6318)	0.5867

Table C.3: 95% Confidence Intervals for message delay in Figure 4.3

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	0.42	(0.3889, 0.4536)	0.4212
	0.486	(0.4500, 0.5249)	0.4874
	0.584	(0.5407, 0.6307)	0.5857
Adaptive	0.423	(0.3917, 0.4568)	0.4243
	0.513	(0.4750, 0.5540)	0.5145
	0.705	(0.6528, 0.7614)	0.7071
Exhaustive	0.425	(0.3935, 0.4590)	0.4263
	0.535	(0.4954, 0.5778)	0.5366
	0.821	(0.7602, 0.8867)	0.8234

Table C.4: 95% Confidence Intervals for message delay in Figure 4.4

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	0.448	(0.4148, 0.4838)	0.4493
	0.674	(0.6241, 0.7279)	0.6760
	1.05	(0.9722, 1.1340)	1.0530
Adaptive	0.43	(0.3981, 0.4644)	0.4313
	0.541	(0.5009, 0.5843)	0.5426
	0.637	(0.5898, 0.6880)	0.6389
Exhaustive	0.412	(0.3815, 0.4450)	0.4132
	0.492	(0.4556, 0.5314)	0.4935
	0.557	(0.5157, 0.6016)	0.5587

Table C.5: 95% Confidence Intervals for message delay in Figure 4.6

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	2.94	(2.722, 3.175)	2.949
	3.21	(2.972, 3.466)	3.220
	3.34	(3.092, 3.607)	3.350
Adaptive	3.53	(3.268, 3.812)	3.541
	3.94	(3.648, 4.255)	3.952
	4.42	(4.092, 4.773)	4.433
Exhaustive	3.69	(3.416, 3.985)	3.701
	4.37	(4.046, 4.719)	4.383
	5.5	(5.092, 5.940)	5.516

Table C.6: 95% Confidence Intervals for message delay in Figure 4.7

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	4.1	(3.796, 4.428)	4.112
	6.95	(6.435, 7.506)	6.970
	10.3	(9.537, 11.124)	10.330
Adaptive	3.0	(2.777, 3.240)	3.008
	4.19	(3.879, 4.525)	4.202
	5.08	(4.703, 5.486)	5.095
Exhaustive	2.87	(2.657, 3.099)	2.878
	3.95	(3.657, 4.266)	3.961
	4.73	(4.379, 5.108)	4.744

Table C.7: 95% Confidence Intervals for message delay in Figure 4.8

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	0.363	(0.3361, 0.3920)	0.3641
	0.455	(0.4213, 0.4914)	0.4563
	0.608	(0.5630, 0.6566)	0.6098
Adaptive	0.36	(0.3333, 0.3888)	0.3611
	0.444	(0.4111, 0.4795)	0.4453
	0.584	(0.5407, 0.6307)	0.5857
Exhaustive	0.357	(0.3306, 0.3856)	0.3581
	0.433	(0.4009, 0.4676)	0.4343
	0.522	(0.4833, 0.5638)	0.5235

Table C.8: 95% Confidence Intervals for message delay in Figure 4.9

Service	P	(P_L, P_U)	$\frac{P_L+P_U}{2}$
Ordinary	0.356	(0.3296, 0.3845)	0.3571
	0.428	(0.3963, 0.4622)	0.4293
	0.533	(0.4935, 0.5756)	0.5346
Adaptive	0.353	(0.3269, 0.3812)	0.3540
	0.419	(0.3880, 0.4525)	0.4202
	0.518	(0.4796, 0.5594)	0.5195
Exhaustive	0.35	(0.3241, 0.3780)	0.3510
	0.41	(0.3796, 0.4428)	0.4112
	0.494	(0.4574, 0.5335)	0.4955

Table C.9: 95% Confidence Intervals for packet delay in Figure 4.10

Service	M	(M_L, M_U)	$\frac{M_L+M_U}{2}$
Ordinary	3.12	(2.888, 3.369)	3.129
	4.2	(3.888, 4.536)	4.212
	6.74	(6.240, 7.279)	6.760
Adaptive	3.06	(2.833, 3.304)	3.069
	4.1	(3.796, 4.428)	4.112
	6.47	(5.990, 6.987)	6.489
Exhaustive	2.9	(2.685, 3.132)	2.908
	3.79	(3.509, 4.093)	3.801
	5.79	(5.361, 6.253)	5.807

Table C.10: 95% Confidence Intervals for message delay in Figure 4.14

Bibliography

[1]Pimental, Juan, "Communication networks for manufacturing"
Prentice-Hall, New Jersey, 1990

[2]Stallings, William," Local networks"
Macmillan publishing company, New York, 1990

[3]Karvelas, Dionysios, "Performance analysis of timer based medium access schemes in symmetric token passing networks"
Ph.D. dissertation, Dept. of Elect. Engg., Uni. of Toronto, Toronto, Canada, March 1990

[4]Tanenbaum, Andrew, "Computer networks"
Prentice-Hall, Inc., New Jersey, 1988

[5]Martini, P. and Welzel, T., " The Analysis of a Token Ring Backbone"
IEEE tran. on comm., April 1988

[6]Bux, W. and Grillo, D., "End to end performance in local area networks of inter connected token rings"
Advances in local area networks, Kummerele, K., Limb, J. O. and Tobagi, F. (Eds.), pp. 496-516, IEEE press, 1987

[7]Chen, C. H. and Bhuyan, L. N., "Design and Analysis of Multiple Token Ring

Networks"

Proceeding of INFOCOM, 1988

[8]Hardy, R. H. S., Radziejewski, I. and Lo, E., "Performance evaluation of a token ring network with dual latency stations"

Proceeding of INFOCOM, 1988

[9]Ferguson, N. J. and Aminetzah, Y., "Exact results for nonsymmetric token ring systems"

IEEE tran. on comm., com-33, pp. 223-231, march 1985

[10]Ferguson, N. J. and Aminetzah, Y., "Mean waiting time for a token ring with station dependent overheads"

Pickholtz, R. L. (ed), Local area and multiple access networks, chapter 3, pp. 43-67, computer science press, 1986

[11]Gburzynski, Pawel and Rudinki, Piotr, "The LANSF protocol modeling environment"

Dept. of computing science, Uni. of Alberta, Alberta, Canada

[12]Trivedi, K. S. "Probability and statistics with reliability, queuing, and computer science applications"

Prentice-hall, Inc., New Jersey 1982

[13]Bux, W., "Token ring local area networks"

Proceedings of IEEE, vol.77, no.2, pp.238-255, feb. 1989

[14]Carsten, R. T., Newhall, E. E. and Posner, M. J. M., "A simplified analysis of scan

times in an asymmetric newhall loop with exhaustive service"

IEEE tran. on comm., vol. com-25, pp. 951-957, sept. 1977

[15]Stallings, William," Handbook of local network standards"
Macmillan publishing company, New York, 1990