Performance Study of Multiple Channel Carrier Sense Multiple Access with Collision Detection Protocol by Simulation.

A THESIS

Submitted to the department of Computer Science and Engineering,
Bangladesh University of Engineering and Technology, Dhaka,
in partial fulfilment of the requirement for the degree

of

MASTER OF SCIENCE IN COMPUTER SCIENCE AND ENGINEERING

FARID AHMED

November , 1990

Bangladesh University of Engineering and Technology

Performance Study of Multiple Channel Carrier Sense Multiple Access with Collision Detection Protocol by Simulation

A thesis
Submitted by

Farid Ahmed
Roll No. 881815P , Registration No. 82102,
for the partial fulfillment of the degree of
M.Sc. in Computer Science and Engineering.
Examination held on : November 27,1990.

Approved as to style and contents by :

M. S. Alam 27/11/90

i. Dr. Md. Shamsul Alam — Supervisor — Chairman
Associate Professor
Department of CSE
B.U.E.T.,Dhaka.

২৯/১১/৯০

ii. Dr. Syed Mahbubur Rahman — Ex-officio — Member
Head, Department of CSE
B.U.E.T.,Dhaka.

২৯/১১/৯০

iii. Dr. Kazi Mohiuddin Ahmed — External — Member
Associate Professor
Department of EEE
B.U.E.T.,Dhaka.

Certificate

This is to certify that the work presented in this thesis paper is the outcome of the investigation carried out by me under the supervision of Dr. Md. Shamsul Alam in the Department of Computer Science & Engineering , Bangladesh University of Engineering & Technology , Dhaka . It is also declared that neither this thesis nor any part thereof has been submitted or is being concurrently submitted anywhere else for the award of any degree or diploma or for publication .

Signature of the
Supervisor

Signature of the
Author

ACKNOWLEDGEMENT

The author feels pleased to have the opportunity of expressing his heart-felt and most sincere gratitude to a number of people who have helped him in course of this research work.

First and foremost the author wishes to acknowledge his profound indebtness to Dr. Md. Shamsul Alam, Associate Professor of the Department of CSE, B.U.E.T. This research work was done under his kind supervision. His keen interest and patience and the scholarly guidance and suggestions at all stages of this research have made it possible to complete this thesis.

The author expresses his utmost gratitude to Dr. Syed Mahbubur Rahman, Associate Professor and Head, Department of CSE, without whose consistent encouragement the thesis work would not have been possible.

The author thankfully acknowledges the valuable discussions occasionally made with Dr. Md. Abdur Rashid, Associate Professor, Department of Applied Physics and Electronics, Dhaka University.

Finally the author acknowledges with sincere thanks ,the all-out co-operation and services rendered by the faculty members and the staff of the CSE Department, B.U.E.T.

Abstract

Carrier Sense Multiple Access with Collision Detection (CSMA/CD) is a very well known and efficient random access communication protocol in Local Computer Networks. In this thesis work a simulation package for the performance analysis of multiple channel unslotted CSMA/CD protocol has been developed. The performance parameters considered are delay, throughput and utilization. Independent parameters are chosen to be number of channels, number of stations, packet generation rate and the distribution of packet length. Variation of the performance parameters with different values of these independent variables are compared. From this comparison an optimum set of parameters can be obtained for a LAN system with given configurations .

The comparison of single channel and multiple channel is also investigated. All these have been done using a simulator, which has been developed using the `C´ programming language. Results of the simulator has been validated using the fairness test and the comparison with analytical result.

The developed simulator can be used to predict the delay, throughput or utilization of unslotted CSMA/CD network of any length, with any number of stations and channels and for any transmission speeds.

Table of Contents

Page

Chapter 1 Introduction

1.1 General

1.3 Literature Review

Chapter 2 SIMULATION

Chapter 3 CSMA/CD Protocol

Chapter 4 Simulation Program Development

Chapter Title Page

Chapter 5 Result & Discussions

chapter 6 Conclusions

References

Appendix : Program Listing

LIST OF FIGURES

LIST OF TABLES

LIST OF SYMBOLS

nc = Number of channels

ns = Number of stations

ps = Mean packet size

a = Ratio of propagation delay to the packettime

t_p = Propagation delay time

t_j = Jam signal time

t_d = Idle Time

t_c = Collision Time

t_s = Simulation Time

t_m = Transmission time of a packet

λ = Arrival Rate in packets per second

L= Offered Load

U = Utilization

B = Bandwidth of the channel or transmission speed

T = Throughput

N = Total number of packet transmitted

ND = Normalized Delay

Q_l = Average Length of the Queue

Chapter 1

INTRODUCTION

1.1 General

1.1.1 Simulation

A Simulation model is a mathematical logical representation of a system which can be exercised in an experimental fashion in a digital computer. Thus a simulation model can be considered as a Laboratory version of a system. It is the technique of solving a particular problem , by the observation of the performance,over time,when direct measurements of the system parameters are inconvenient or when it is desirable to view the system behavior at an increased or decreased rate of time passage, relative to the observer.

1.1.2 Computer Communication & Networks

The exchange of information between computers for the purpose of cooperative action is generally termed as Computer Communication. For the purpose of Communication the inter connected collection of autonomous computers is called computer network.

The main objectives of computer networks are :

i) Resource sharing - To make all programs, data and other resources available to anyone on the network without regard to the physical location of the resource and thee user.

ii) Fault Tolerance - Temporary failure of a channel in a multichannel Network can be overcome by using another in the network. Similarly a failed bus can be replaced by another in the system.

Now the ISO reference model of a network subdivides the network function into 7 layers namely physical layer, data link layer, network layer, transport layer, session layer, presentation layer and application layer. Network Layer can again be subdivided into 3 categories [1]. These are

i. Point to point network,
ii. Satellite & packet radio network and
iii. Local network.

Fig 1.1 shows this hierarchical relationship .

Local Area Network (LAN)

Local Area networks have three distinctive characteristics as

i. A diameter of not more than a few kilometers
ii. A total data rate exceeding1 Mbps
iii. Ownership by a single organization.

One reason of LAN is to connect together a collection of computers and peripherals, located in the same building or in adjacent buildings, not only to allow them all to intercommunicate but also to allow all of them to access a remote host or other network.

There are also three key characteristics of LAN that affect the way of its performance analysis. These are :

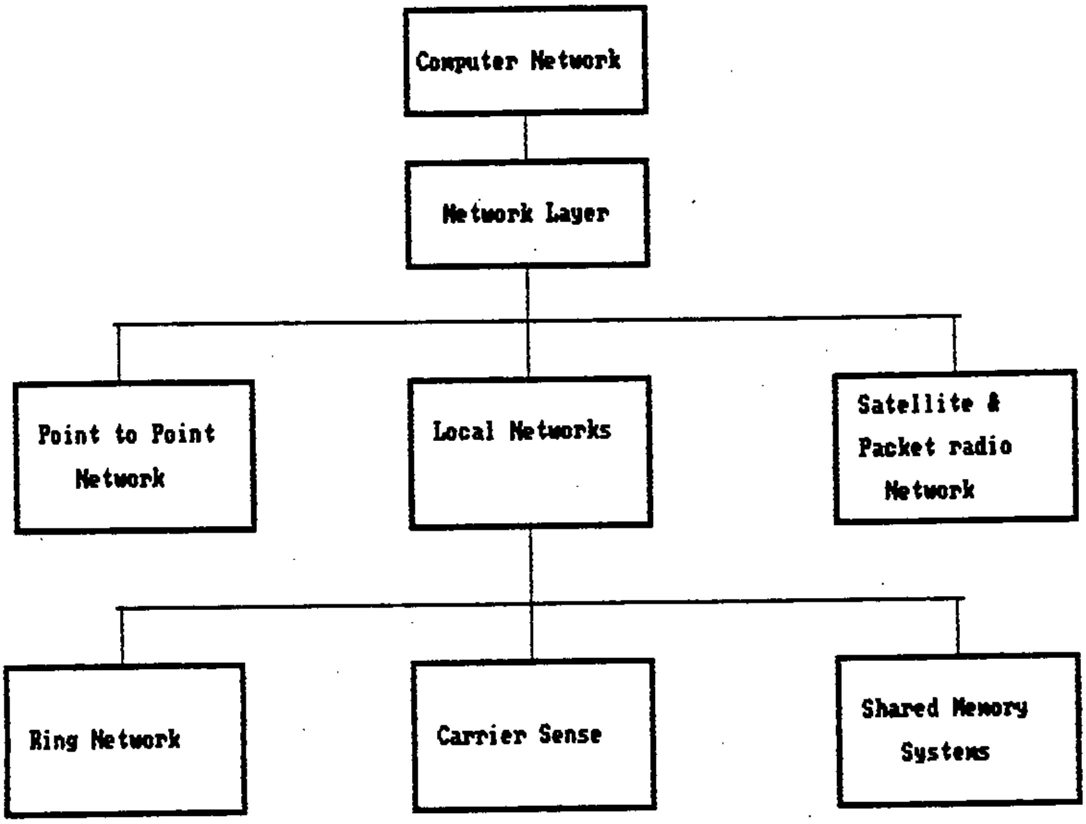


Fig 1.1 : Hierarchy of carrier Sense Networks

i) Shared access medium

ii) Medium access control protocol

iii) Switching Technique.

LAN Architecture :

Topology :

The way in which devices participating in the network are interconnected together is called Topology. The basic topologies used in the LAN are

a) Star Topology : Computers connected in this topology communicate through a central hub.

b) Ring Topology : All the connected computers form a ring .

c) Bus/Tree Topology : Computers in the network are connected to one or several buses or channels.

Fig 1.2 to Fig 1.5 show the structure of these topologies.

Transmission Media :

It is the lowest level of computer networking and provides the physical connection between the devices and the network. The widely used media in LANs are :

i) Twisted Pair

ii) Coaxial Cable

iii) Optical Fiber

Twisted pair wiring is the most common communications transmission medium and is typically used for low speed data communication. The advantage of twisted pair over the other two

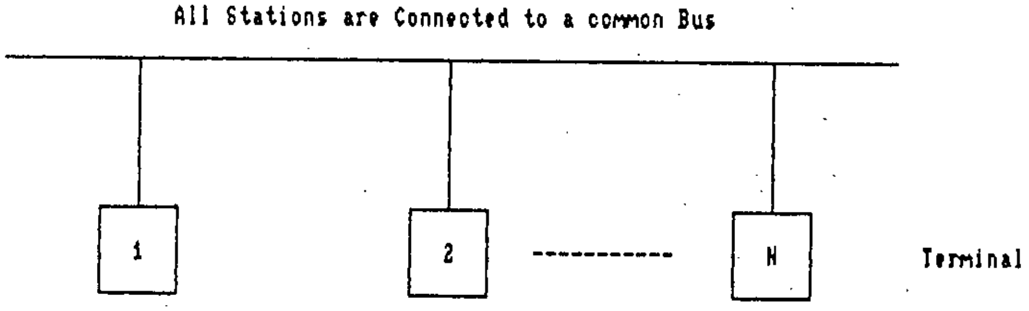


Fig 1.2 : Single Bus Topology

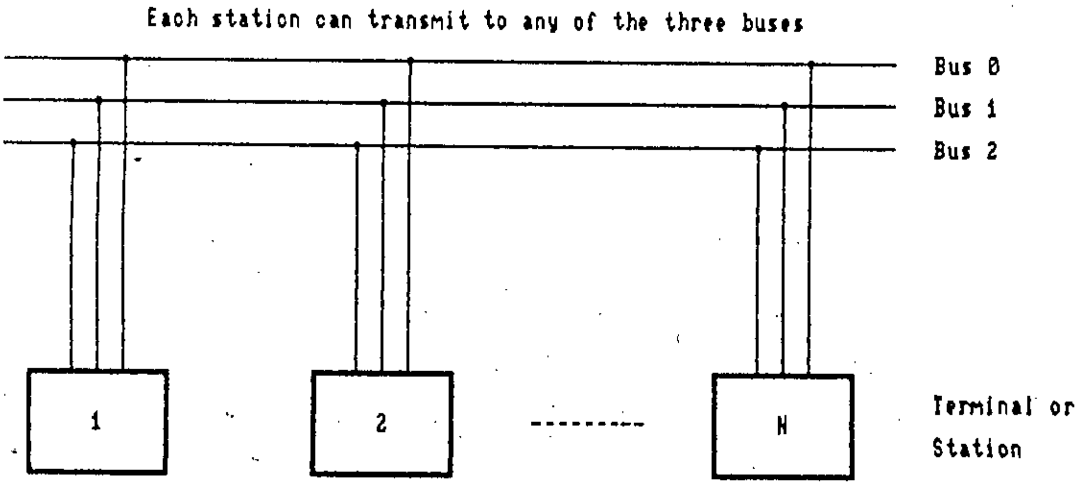


Fig 1.3 : Multiple Bus Topology

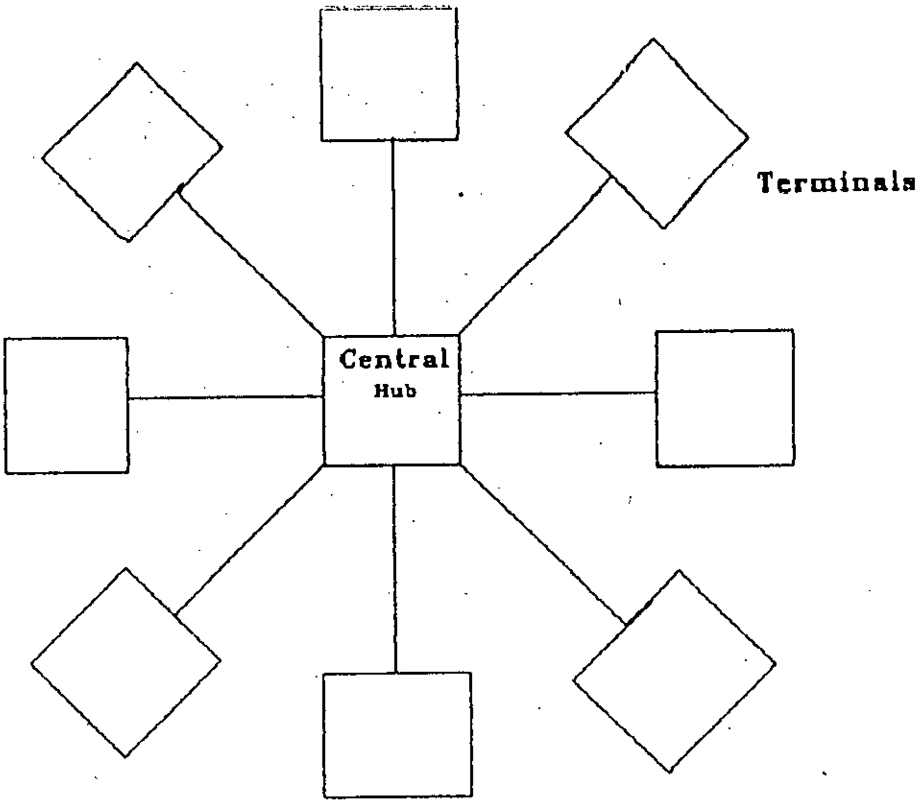


Fig 1.4 : Star Topology : All Devices are connected to a Central Hub

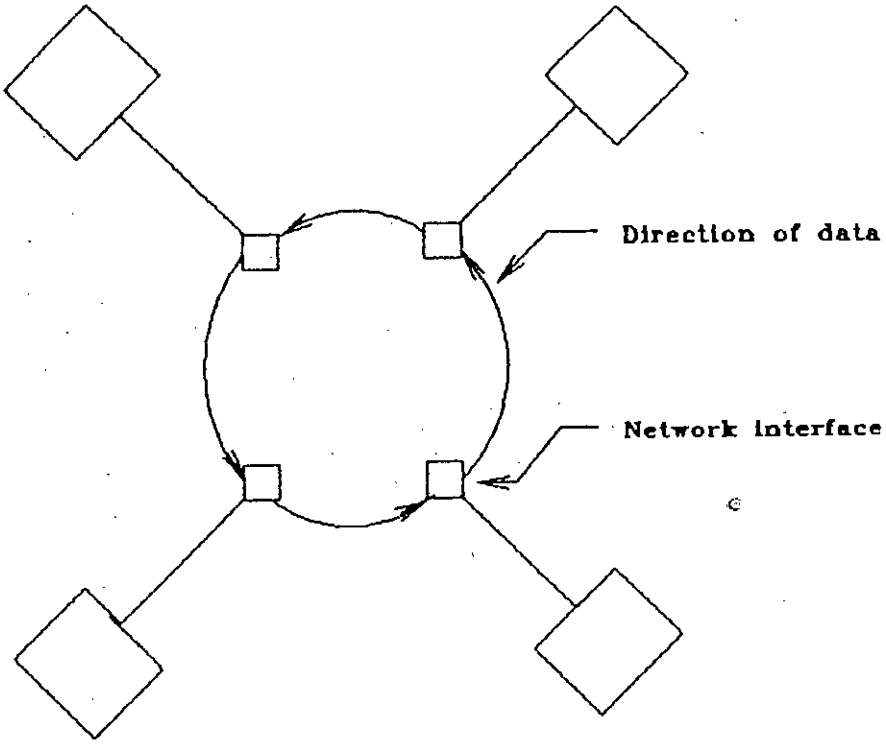


Fig 1.5 : Ring Topology :Host Computers are connected to a Ring

media is its lower cost.

Coaxial cable provides higher performance requirements. It provides high throughput and can support a larger number of devices and can span greater distances than twisted pair.

Optical fiber cable is even of greater capacity than coaxial cable and is being introduced in LAN potentially. It has been, however, little used so far due to cost and technical limitations. Optical fiber cable is immune to electrical interference and thus provides excellent security and reliability.

Switching Method :

a) Circuit Switched : A copper path must be set up between the end to end user before any data can be sent. Example : Telephone system.

b) Message Switched : No physical copper path is established in advance between sender and receiver . Instead when a sender has a block of data to be sent , it is stored in the first switching office (IMP) and then forwarded later , one hop at a time. A network using this technique is called Store and Forward network.

b) Packet Switched : Same as message switching except that there is a tight upper limit on the block size which may reduce the buffer length. The aim is that no user can monopolize any transmission line for more than a few tens of milliseconds , so that it may be well suited to handling interactive traffic.

For better delay - throughput characteristics , computer networks are usually packet switched, occasionally circuit switched , but never message switched.

Types of LAN :

Fig 1.1 shows the classification of LAN. These are:

a) Carrier sense networks :

Persistent and non-persistent CSMA, Collision free protocol, Limited contention protocol, CSMA with Collision Detection i.e. Ethernet - all these fall into this category.

b) Ring Networks :

Token rings, contention rings, slotted rings and register insertion rings are typical example of Ring Networks . Fig 1.5 describes the structure of a ring network.

c) Shared memory systems :

A collection of processors that share the common address space or use the same main memory is an example of Shared Memory System.

In the present work we are concerned with the Carrier sense network .

1.1.3 Performance Analysis

The following are the most commonly used LAN performance parameters :

Delay :- The time between the instant a packet is ready for transmission from a terminal and the instant of the start of its successful transmission. In our analysis the term Normalized Delay (ND) is used, which is defined as the ratio of total delay time faced by all the transmitted messages to the time of total number of transmitted messages.

Throughput : T, the throughput of the local network is the total rate of data (in Mbps) being transmitted between the terminals.

Utilization : The utilization of the local network medium represents the fraction of total capacity being used.

The parameter T is often normalized and expressed as function of capacity. For example if over a period of 1 sec, the sum of the successful data transfers between nodes is 1 Mbits on a 10 Mbits channel then T = 0.1 .

Thus, T can also be interpreted as utilization.

Fairness : Fairness is an interesting measure to see whether the access scheme provides equal opportunities to all the stations trying to use the channel. The fairness is investigated by examining the variations in performance characteristics measured by individual stations with varying loads. The variations are represented by the normalized maximum & minimum throughputs as well as the standard deviation varied with the load.

Factors that affect Performance :

Following is a list of those factors that affect the performance of a LAN.

i. Bandwidth

ii. Propagation delay

iii. Offered load

iv. Number of stations

v. Number of channels

vi. Type of the medium access protocol

The above factors are independent of the attached devices - those factors that are exclusively under the control of the local network designer. On the user side one can only vary the arrival rate.

1.1.4 The CSMA/CD System

The most commonly used access protocol technique for LAN bus/tree topology is carrier sense multiple access with collision detection. This is also referred to as Listen while Talk.

In this technique all stations have independent access to the medium. Any station willing to transmit some thing must at first listen or sense the carrier to see if the channel is idle. If the channel is sensed busy, the station must wait till the medium becomes free.

Once transmission from a station starts, a certain amount of time elapses to reach the data to destination station. This is called propagation delay. If no other station starts transmission within this short period of time interval following the start of the transmission, the transmission succeeds. After this propagation delay if some other stations try to transmit in the same channel ,where a transmission is already going on then these stations face delay and reschedule their attempts. If on the other hand, any other station starts transmission being unaware of the fact that another station has already started transmitting packet that has not yet been removed from the medium,the transmissions are said to collide. The CSMA/CD protocol, however, detects any such collision. When a collision occurs, all stations which are involved in this , cease transmission immediately and wait for a certain amount of time determined from a predefined statistical calculation.

1.2 Statement of the problem

The present work can be subdivided into three parts

A) Development of a simulation package for CSMA/CD protocol.

B) Determining the effect of different packet length distribution policies on the delay - throughput characteristics of CSMA/CD.

C) Study of performance analysis with the variation of number of channel and the number of stations in the system.

Performance analysis includes the following :

i) Delay - Throughput characteristics

ii) Variation of utilization with respect to the ratio `a´ , where ´a´ is defined as the ratio of propagation delay to the mean packet transmission time.

iii) Fairness Test of the protocol.

iv) Number of collision versus utilization.

v) Average queue length of each channel. Queue length is defined as the number of stations waiting for transmission when one station is transmitting over the channel successfully.

vi) Utilization of each channel.

The developed simulator shall be able to run a different number of stations, channels, transmission speeds, arrival rates and different packet length distribution policies i.e. the user whether he is an end user (terminal user) or a network administrator shall be able to see the effect of varying the parameters within their control.

Therefore the works to be done are summarized as

i) Designing the simulator

ii) Implementing & testing the simulator

iii) Use of this Simulator to observe the performance of CSMA/CD protocol by varying the number of stations,

iii) Use of this Simulator to observe the performance of CSMA/CD protocol by varying the number of stations, channels and for the following three distribution policies.

1. Negative Exponential Distribution
2. Uniform Distribution
3. Constant Packet Length

1.3 Literature Review

1.3.1 Large Scale Simulations

Many excellent computer network simulation programs have been developed over the last ten years. Simulation has been used in the overall evaluation of network switching strategies at Rand Corporation, of routing algorithms at National Physical Laboratory and of network performance in the ARPANET. These simulations have often been written for large mainframe computers in simulation language such as GPSS specially designed for the purpose.

One of the basic drawbacks to large-scale simulations of this type is that they can become quite cumbersome and thus, cannot be used to handle inter-related problems that must be solved in the detailed design of specific network functions. For example, although a aspect of a routing algorithm could be analyzed as indicated above, it would be a difficult task to simultaneously introduce, say, fluctuations, in the traffic flow.

1.3.2 Distributed Simulation

An alternative approach was described by the Computer Communications Network group of the University of Waterloo in their report "Computer Network Simulation System" CCN6 report E-25, 1974. They simulated a data switching network with a multiprocessor : one processor handled traffic generation and reception; a second processor handled switching functions. They used three PDP-11 computers interconnected by fast, direct memory access links. They claim that their simulation is 10 times more cost effective than the conventional large scale simulations on conventional computer.

1.3.3 Related Study Conducted on Simulation

Metcalfe & Boggs [2] presented a prototype of a concept for local computer networks. This technique (called ETHERNET) is particularly attractive by reason of its structural simplicity. ETHERNET combines any number of stations by means of logically passive medium for the propagation of packets. Packet transmission is similar to the "carrier sense multiple access" (CSMA) technique, that is, most possible interferences are avoided by listening to the carrier. The system is made easy to understand provided that, no one transmits at the same time. It is to meet this problem of possible collisions that Ethernet contains some special design features developed at Palo Alto.

Shoch & Hupp [3] presented the results of an Ethernet local network study focused on the measurement of the performance at

Xerox Palo Ago Research Center. Under extremely heavy (Artificially generated) the system shows stable behavior and channel utilization approaches 98 per cent as predicted .

Bux [illegible] presented the results on the performance comparison study of local area sub-networks conducted by IBM Zurich Research Laboratory, Switzerland. This provides a comparative evaluation of the performance of ring and bus systems constituting sub-networks of local area networks. Performance is measured in terms of the delay-throughput characteristics. Systems investigated include token controlled and rings as well as random access buses (CSMA with collision detection) and ordered access buses (MLMA). The investigation is based on analytical model which describes the various topologies and access mechanisms to a sufficient level of detail. The study included a comprehensive discussion of how the performance of the different networks is affected by systems parameters like transmission rate, cable length, packet length and control overhead. The traffic generated by stations is assumed to have be Poisson distributed.

Georganas & Naffah [illegible] presented the results of the performance study on "integrated office systems over LANS" carried out at University of Ottawa, Canada. It centers around the performance of three selected commercial networks Appletalk, Starlan and Ethernet. This study applied discrete-event computer simulation, using QNAP2 simulation software. Both realistic and

Price [illegible] presented a broad survey of packet switched data network simulation experiments at the physical laboratory during the years 1968-1976. Reference is made to several operating protocols. including flow control. The effect of various network enhancements and several types of component failures were studied. The data network considered in this work were parts of possible national store and forward networks. Computer simulation was used to study the behavior of the network under various traffic loads and operating with a variety of control protocols and mechanisms. Generally speaking, the network performance parameters examined in the simulation experiments were the mean values of carried load or throughput and delay in awaiting admission and in transmit. [4]

Kleinrock [7] undertook significant studies of computer network performance, by mathematical analysis and by simulation. His simulation was undertaken at a detailed level, based on a detailed description of ARPA network. Network performance was studied by the choice of routing and flow control.

Later on many researchers used simulation model using different simulation language such as SIMULA ,GPSS, SIMSCRIPT, SLAM etc. to compare the analytical model with the simulation model.

D. P. Heyman [8] has shown that throughput degrades slightly as the mean number of packets per message is increased and the ;load is kept constant.

Steven L. Beurman and Edward J. Coyle [9] have shown that significant improvement obtained in the delay characteristics of a particular network when FCFS (First Come First Serve) CSMA/CD is used instead of RSO (Random Service Order) CSMA/CD. They worked on the suitability of FCFS and RSO CSMA/CD networks in applications where they must carry delay sensitive data.

T. K. Apostolopoulos and E. N. Protonotarious [10] proposed a new queuing model appropriate for the analysis of a buffered CSMA/CD protocol in their paper 'Queuing Analysis of Buffered CSMA/CD protocols'. They assumed that each user has a finite buffer capacity. The analysis was done using a two dimensional Semi- Markov Chain. The obtained solutions found were extremely accurate and exhibited excellent agreement with simulation result.

Takagi and Murata [11] have developed an stochastic analysis for the interval between two successive successful transmission in a variety of slotted persistent CSMA and CSMA/CD system. They included DFT (Delay First Transmission) and IFT (Immediate First Transmission) models.

Ko, Lye and Chua [12] have shown that the throughput delay performance can be improved if simultaneous successful transmission are allowed on the single channel of a network. The ability to accommodate simultaneous successful transmission is achieved by dynamically partitioning the network into independent segment.

S.M. Sharrock , S. Ghanta and H.C. Du [13] have proposed an efficient fully distributed protocol for integrated voice data traffic in a local area random access broadcast network in their paper ʹA CSMA/CD based , Integrated voice /data protocol with dynamic channel Allocation. The behavior of framed TDMA/CSMA was investigated via simulation and analysis.
Y. Matsumoto , Y.Takahasi and T.Hasegawa [14] have analyzed the probability distributions of Interdeparture time and response time in multipacket CSMA/CD system. They have presented an exact analysis of the unslotted multipacket CSMA/CD- DFT model and derived the laplace - steiljets transform of the probability distribution function, the moment generating function of the message response time.

Gonsalves and Tobagi [15] have presented a simulation study of several aspects of the Ethernet performance. They have found the effects of station locations and Access protocol parameters .

M. S. Alam and R.E. Swartwout [16] have established a model for the performance analysis of CSMA/CD-DFT protocol in a Multiple Bus LCN. The model was based on the analysis of two-dimensional Markov chain. The analytical was validated using the simulation results.

Chapter 2

SIMULATION

2.1 Definition

Simulation in the arena of Computer means writing a program in such a way that it will execute in a manner just like a system behaves in the real world. It is a mathematical logical representation of a system which can be executed in an experimental fashion in a digital computer.

Simulation modeling assumes that we can describe a system in terms acceptable to a computing system. In this regard a key concept is that of a system state description. If a system can be characterized by a set of variables with each combination of variable values representing a unique state or condition of the system, then manipulation of the variable values simulates movement of the system from state to state. This is precisely what simulation is: *The representation of the dynamic behavior of the system by moving it from state to state in accordance with well defined operating rules.* [17]

2.2 Purpose of Simulation.

A simulation model permits inferences to be drawn about a system

- Without building them if they are only proposed systems.
- Without disturbing them ,if they are operating systems that are costly or unsafe to experiment with.
- Without disturbing them , if the object of an experiment is to determine their range of operation.

Therefore a Simulation model can be used for

1. Design
2. Procedural analysis and
3. Performance assessment of a system .

2.3 Types of Simulation Models.

To clarify the nature of simulation, a number of the characteristics of simulations are defined. It may be classified on the basis of these characteristics. [18]

i) Static-Dynamic :

A simulator may be used to represent both dynamic and static situations. In most operations research studies, we are interested in dynamic models, for example, a simulation describing a chemical process. Occasionally certain static problems are of interest and typically, these are problems of space allocation or plant design.

ii) Aggregate-Detailed

One of the most important characteristics of a simulator is its degree of aggregation. In simulating a network protocol, for example, we can represent every operation performed by each station & channel of the system ,the change of collision state to transmission state & vise-versa . It could also be demonstrated why a collision is occurring ,how the stations behave when finding the channel busy and all other related details . On the other hand, we can construct a very aggregate models in which only the gross quantities-total system utilization ,delay and so-forth are represented.

iii) Physical-Behavioral

The system being modeled may contain only physical process, or it may involve only human behavior. The example of physical process is a chemical process. In a management Decision System only human behavior is incorporated. Most simulation models involve aspects of both, as the present network protocol simulator.

iv) Computer-Human

The kind of mechanism used to carry out the simulation procedure is an important factor. At one extreme there is the all-computer simulation in which the entire procedural model is executed on a computer. However, there are also models in which some of the behavioral subroutines have not been specified; here the analyst allows a human to carry out these subroutines on line as the simulation proceeds. Gaming is such kind of simulation in which there is a human-determined component of the model.

v) Recursive-Quasi-equilibrium

There are two approaches to the operation of dynamic models : the recursive and the quasi-equilibrium. The recursive approach requires that the state of the system at any given time be derived within the model from the conditions at earlier times. On the other hand , the models of economy generally comprises a set of simultaneous equations that must be solved in each time period . This approach is called quasi-equilibrium method.

vi) Continuous-Discrete

The variables in a simulation may change in any of four ways :

(i) In a continuous fashion at any point in time,such as the global weather change.

(ii) In a discrete fashion, but at any point in time as the present network protocol.

(iii) In a discrete fashion and only at certain points in time, such as monitoring a system in some fixed interval of time.

(iv) In a continuous fashion, but only at discrete points in time. For example if the concentration level in a chemical process reaches a point then the process may be shut down.

The nature of the variables used depends upon the situation modeled, the purpose of the model and the kind of computational facility available. In general, more aggregate models tend to use variables that are continuous in value, but often discrete in time.

vii) Deterministic-Stochastic

Where the outcome of an activity can be described completely in terms of its input , the activity is said to be deterministic. Where the effects of an activity vary randomly over various possible outcomes , the activity is said to be stochastic.

Most situations in the real world have stochastic (randomly varying) properties because of real (or assumed) ignorance of details. Sometimes these properties must be modeled explicitly, but it is often sufficient to model situations as if they were deterministic by using expected values of the variables.

2.4 Subsystems of Simulation Models.

The process for the successful development of a simulation model consists of beginning with a simple model which is embellished in an evolutionary fashion to meet problem-solving requirements. Within this process, the following stages of development can be identified.

i) Problem Formulation :

It includes the definition of the problem to be studied including a statement of the problem-solving objective. The formulation consists of both a static and a dynamic description. The static phase defines the elements of the system and the characteristics of the system. The dynamic description states the way in which the elements of the system interact to change the state of the system with respect to time.

ii) Model Building :

The abstraction of the system into mathematical-logical relationships in accordance with the problem formulation. The model of a system consists of both a static and a dynamic description. The static description defines the elements of the system and the characteristics of the elements. The dynamic description defines the way in which the elements of the system interacts to cause c of the system over time.

iii) Data Acquisition :

The identification, specification and collection of data.

The prior two phases will generate data input requirements for the model.

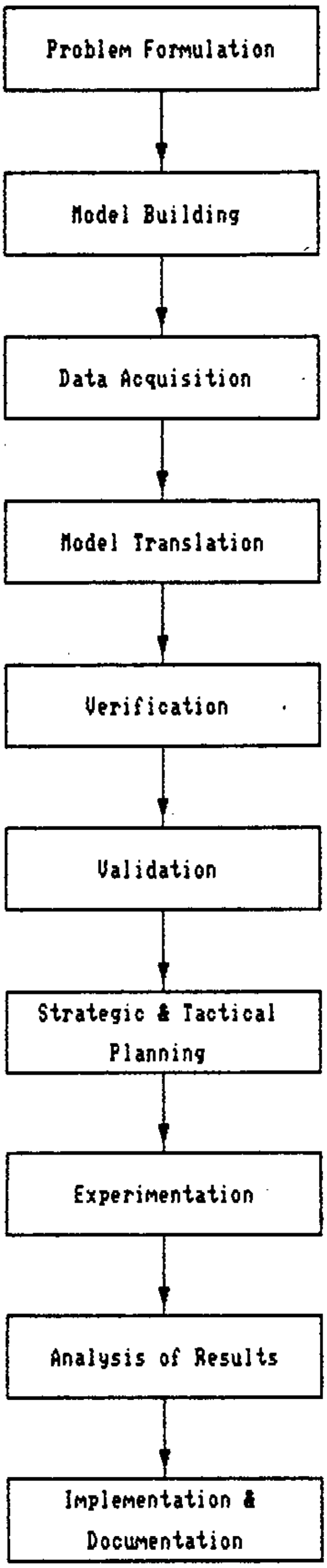


Fig 2.1: Subsystem of a Simulated Model.

iv) Model Translation :

The efficient coding of the model for computer processing. This can be done using the general purpose programming languages or the more specifically developed simulation languages.

v) Verification :

The process of establishing the fact that the computer program executes as intended. This is typically done by manual checking of calculations. Fishman and Kiviat in their book 'Analysis of Simulated Generated Time Series ' have described statistical methods which can aid in verification process .

vi) Validation :

The process of proving that a desired accuracy or correspondence exists between the simulation model and the real system. Verification and validation processes are actually concerned with the evaluation of the performance of the simulation model.

vii) Strategic and Tactical Planning :

The phase of establishing the experimental conditions for using thee model.

viii) Experimentation :

The execution of the simulation model to obtain output values.

ix) Analysis of Results :

The subsystem of analyzing the simulation outputs to draw inference and make recommendations for problem solution.

x) Implementation and Documentation :

The process of implementing decisions resulting from the simulation and documenting the model and its use.

2.5 Discrete Simulation Modeling.

Discrete simulation can be of two types.

i) Time driven simulation and

ii) Event driven simulation

In the time driven simulation system , statistics are updated in the discrete point of time. An event may or may not occur between two points of time i.e. there may be a number of periods when no event occurs.

In the event driven simulation system statistics are updated at the start or end of an event. Random units of time may elapse in this period.

Time driven modeling is equally applicable to both continuous and discrete systems, whereas event driven model is only applicable to the discrete systems.

In general, if events occur on a fairly regular basis, then time driven simulation is preferable. In event driven system, a list of events is to be maintained. In the worst case, within each unit of time, many events may occur which may result in the loss of information. Gofarian and Aucker [18] revealed this fact in their "Mean Value Estimation from Digital Computer Simulation." Conway, Johnson and Maxwell [19] , in their paper titled "Some problems of Digital Computer Simulation" have formulated the trade off between the two methods as follows :

In their paper they dealt with a hypothetical problem that has the following characteristics :

(i) The state of the system at any time can be described in terms of k variables.

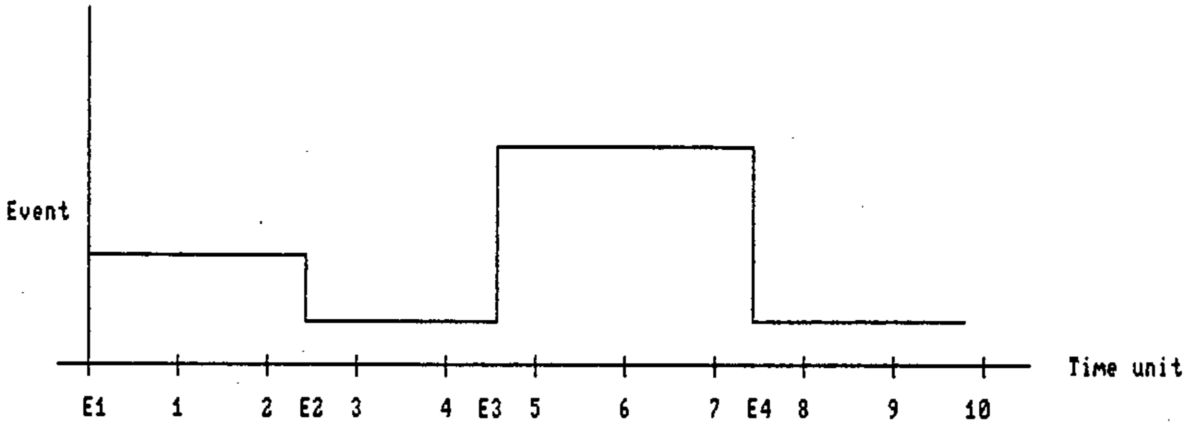


Figure 2.2: Showing the difference between event driven and time driven simulation

[Design & Use of Computer Simulation Models, J.R. Emshoff]

Time Driven : For updating system statistics for the event E_3 time is advanced from 4 to 5. No event occurs between the time unit 3 & 4 and 5 & 6.

Event Driven: For updating system statistics for the event E_3 time is advanced from 2.5 to 4.5.

(ii) Each variable has a particular value for an average of m time periods.

(iii) The simulations model is to be run for t simulated time periods.

The unit-time-advance simulator has a time interval of 1/t. In each interval, the simulator checks the current status of each of the k state variables. Thus the total simulation will involve kt comparisons. To determine the number of comparisons in the event simulator, it is necessary to computer the expected number of events that will occur during the run. Assuming that events occur independently, the probability of an event occurring in any particular time interval is k/m and the expected number of events tk/m. Each time an event occurs, the event list must be updated. With k state variables, the update can be achieved by k - 1 paired comparisons in the event list. Thus the total simulation using event-advance methods will involve tk(k - 1)/m comparisons.

Conway et al. found that an approximate estimate of the relative running efficiency of the two time-advance procedures is obtained by computing tk with tk(k - 1)/m. By comparing these formulas they concluded that event-advance simulators become more advantageous as the mean time between events increases (i.e., m gets larger), whereas the unit-advance method becomes preferable when the number of state variables increases (i.e.,k becomes larger). Thus simulators of large systems (e.g., enterprise) in which there is a high probability of something happening in a time unit should use unit-time-advance methods.

2.6 Some Existing Simulation Languages. [17]

i) GASP : A set of subroutines in FORTRAN that performs function useful in Simulation .

ii) GPSS : A complete language oriented towards problems in which items pass through a series of processing.

iii) SIMSCRIPT : A complete language oriented towards event to event simulations in which discrete logical processes are common.

iv) CSMP : A complete language oriented towards the solution of problems stated as nonlinear integral-differential equations.

v) DYNAMO : A complete language oriented towards expressing micro economic models of firms by means of difference equations.

vi) SIMULA : A well designed simulation language similar to SIMSCRIPT that compiles into ALGOL.

vii) SLAM : Simulation Language for Alternative Modeling supports the modeling of a system from diverse points of view.

2.7 Statistical Aspects of Simulation.

2.7.1 Random Number Generation :

In the Stochastic simulation models, the simulation must include the representation of random variables - variables having values that are specified according to a distribution. In simulation analysis, random variables are used to represent the behavior of uncontrollable factors in the system - whose real world counterparts fluctuate in an unpredictable but statistically describable way.

The distribution from which the random variable is to be drawn must be established from observed data or from the theoretical considerations.

The following are criterion for "Acceptable" methods for generating random numbers. These methods must yield sequences of numbers that are :

i. Uniformly Distributed

ii. Statistically Independent

iii. Reproducible

iv. Nonrepeating for any desired length (Period)

v. Capable of generating random numbers at high rate of speeds

vi. Requiring a minimum amount of computer memory capacity.

There are many methods that have been and are being used to generate random numbers satisfying the above conditions. Two of them which are mostly used are :

A) Mid-square Method : In it each successive number is the middle digits of the previous number squared. It met all the above mentioned criterion except that its period is dependent on the initial value and quite often degenerated into cycles of very short length. Because of this degeneration this method is not used very much.

B) Congruential Method.

Linear Congruential generators are of the form

$$X_{i+1} = (C_1X_i + \ldots\ldots + C_jX_{i+1-j} + C_0) \bmod m$$

where C_j 's are constants and m is an integer determined by the word size of a computer.

Linear Congruential method can again be of three types :

i) Additive

ii) Multiplicative

iii) Mixed

The Additive method is also called the Fibonacci method for its resemblance to the Fibonacci series in which any number is the summation of its two preceding numbers.. The chief drawback of this method is their requirement for additional storage and their failure to perform well in their run test. Their main advantage is in the great increase in period length independent of the computer word size that they allow.

Multiplicative generator is chosen to meet certain requirements to guarantee that the sequence of numbers has the larger period. The value of m is taken to be the word size of the computer being used. Memory requirement here is less.

Mixed Congruential method comprises the advantages and disadvantages of both the above mentioned methods.

Following is a description of the mixed congruential random number generator which is very widely used.

This method has the following recurrence relationship :

$$X_{i+1} = (cX_i + d) \bmod m$$

The expression means to take the last random number, X_i, multiply it by the constant c and add with the constant d . Then take the result, modulo m (that is, divide by m and treat the remainder as X_{i+1}). Thus the random numbers all range between zero and m - 1. If m is chosen as the largest possible integer in the computer, division to take the modulo is done implicitly by the multiplication process and some computer time is saved. Moreover , here X_i is taken randomly from an originally stored table.

2.7.2 Distribution Policies :

For the use in Simulation the generated random numbers must be transformed to random variates using any suitable distribution method. Following is a description of some of these methods for discrete random variates.

Erlang Process Generator

Let X be an Erlang random variable with probability density function given by

$$f(x) = \frac{(n\lambda)^n}{(n-1)!} x^{n-1} e^{-n\lambda x}, \quad 0 < x < \infty$$

The Erlang random variable could be expressed as the sum of independent, identically distributed exponential random variables. Let $Y_1, Y_2, \ldots, Y_n$ be independent exponential random variables each with parameter λ. Then

$$X = \sum_{i=1}^{n} Y_i$$

Now Y_i can be generated by

$$Y_i = \frac{1}{n\lambda} \ln(1 - r_i)$$

where r_i is a uniformly distributed random number under on the interval (0, 1). Then

$$x = \sum_{i=1}^{n} Y_i = - \frac{1}{n\lambda} \ln[\prod_{i=1}^{n} (1 - r_i)]$$

Since r_i and $1 - r_i$ are identically distributed, the process generator for the Erlang random variable becomes

$$x = - \frac{1}{n\lambda} \ln(\prod_{i=1}^{n} r_i)$$

Thus, to generate an Erlang random variable with parameters and n, we generate n random numbers r_i, i = 1, 2, ..., n, take the natural logarithm of their product and multiply the result by -1/n .

Normal Process Generator

Let r_1 and r_2 be two random numbers on the interval (0, 1). Then

$$x = [-2 \ln(r_1)]^{1/2} \cos(2\pi r_2) \quad \ldots\ldots(1)$$

is a standard normal random variable (Abramowitz and Stegun, 1964). To generate a normal variable with mean m and variance ^{2} we modify (1) as follows :

$$x = m + \sigma[-2 \ln(r_1)]^{1/2} \cos(2\pi r_2)$$

Chi-Square Process Generator

A chi-square random variable can be generated by noting its relationship to the standard normal variable. Let Z_1, Z_2, ..., Z_n be independent standard normal random variables. Then

$$X = \sum_{i=1}^{n} Z_i^2$$

is a chi-square random variable with n degrees of freedom.

Uniform process Generator

Let X be a uniformly distributed random variable with probability density function

$$f(x) = \frac{1}{b - a}, \quad a < x < b$$

Then

$$F(x) = \int_{a}^{x} \frac{1}{b - a}\, dy = \frac{x - a}{b - a}, \quad a < x < b$$

and

$$x = a + (b - a)r$$

is the required process generator.

Geometric Process Generator

The distribution function for the geometric random variable X is given by

$F(x) = 1 - (1 - p)^{x}$, $x = 1, 2, \ldots,$ or $r = 1 - (1 - p)^{x}$

Then X has the value x whenever

$$1 - (1 - p)^{x - 1} < r <= 1 - (1 - p)^{x},$$

or

$$(1 - p)^{x} <= 1 - r < (1 - p)^{x - 1}$$

Exponential Process Generator

$$f(x) = \alpha e^{-\alpha x}, \quad \alpha > 0,\ x >= 0,$$

$$= 0, \quad \text{elsewhere.}$$

$$F(x) = \int_{0}^{x} \alpha e^{-\ t}\, dt = 1 - e^{-\alpha x}$$

$$E(x) = \frac{1}{\alpha}$$

$$V(x) = \frac{1}{\alpha^{2}} = [E(x)]^{2}$$

By inverse transformation technique,

$$R = 1 - e^{-\alpha x} \quad \text{or} \quad 1 - R = 1 - e^{-\ x}$$

$X = -(\frac{1}{\alpha}) \ln R$, since R ia as likely to occur as 1 - R.

= -E(x) ln R.

Poisson Process Generator

If the number of events occuring in a fixed time interval T is poisson distributed, then the time between successive events is exponentially distributed. If Y_1 is the time until the first event, Y_2 the time between the first and second events, ..., Y_x the time between the (x-1)th and xth interval, and if Y_1, Y_2,

exponentially distributed random variables with parameter , then the number of events occuring in T is poisson distributed with parameter T and has the value x-1 if and only if

$$\sum_{i=1}^{x-1} Y_i < T \leqslant \sum_{i=1}^{x} Y_i$$

Hence, to generate a poisson random variable X with parameter T, We generate an exponential random variable Y_1 with parameter , and Y_1 with T. If $Y_1 > T$, then the value of x is zero. If $Y_1 < T$ we generate another exponential random variable Y_2, and compare Y_1+Y_2 with T. If $Y_1+Y_2 > T$, the value of x is one. If $Y_1+Y_2 < T$, a third exponential random variable Y_3 is generated and $Y_1+Y_2+Y_3$ is compared with T.

This process continues until the sum of exponential random variables generated exceeds T. At this point the value of the poisson random variable is the number of exponential random variables included in the sum minus one.

2.7.3 Starting and Stopping Policy :

Initial Conditions :

In every simulation model there is an initial condition . The simplest and probably the most commonly used initial state is 'empty and idle' in which the simulation begins with no entities in the system. When the purpose is to study the steady state behavior of a system then it may be desirable to start with an initial condition that might be a representation of the long term behavior of the system.

Data Truncation :

Initial transient results obtained from the simulation can be avoided in many methods . One of these methods is to delay the collection of statistics until after a "warm up" period. This is usually done by specifying a truncation limit upto which data values are included in the statistics. The most common method to specify this limit is to examine a plot of the response from a pilot run. The truncation point is selected as one where the statistics are seemed to have attained the steady state .

Run Length and the Number of Replications :

An important factor in the design of simulation model is the tradeoff between the Simulation time and the number of simulation run. The use of a few long runs as opposed to many short runs generally produces a better estimate of the steady state results because the initial bias is introduced fewer times and less data is truncated. However reduced replications may increase the estimate of variance.

There are several alternate methods for estimating the length of simulation run. Perhaps the most common method is to specify a simulation time. But the disadvantage of this method is that as the nature of the events is stochastic so number of samples collected might not be satisfactorily enough. Another method is to use a fixed number of entities . The simulation will be continued upto the completion of this fixed number. Still another approach for controlling the duration of a simulation is the use of automatic stopping rules. These rules automatically monitor the simulation results at selected intervals during the execution of the simulation. The simulation is stopped when the estimate of the variance of the mean is within a prescribed tolerance.

Chapter 3

CSMA/CD Protocol

3.1 LAN Protocols :

A LAN usually contains four major components shown in fig 3.1 [22] . These are :

A) A LAN's path may consist of coaxial cable named as channel

B) Interface between the path and the protocol logic . Typical example is RS232-C.

C) The protocol control logic component controls the LAN and provides for the end user's access to the network.

D) User work station. It can be anything from a user workstation to a mainframe computer.

We are at present concerned with the third component i.e. LAN Protocol. Local Networks employ several kinds of data link controls such as polling/selection,hub polling, contention,and time slots to manage the flow of data on a communication path. IEEE LAN 802 standards , published in late 1983 describes the four following standard protocol.

i. 802.2 : Logical Link Control (LLC)

ii. 802.3 : CSMA/CD

iii. 802.4 : Token Bus

iv. 802.5 : Token Ring

LLC is designed to provide a level 2 interface for IEEE 802.3 ,IEEE 802.4 and IEEE 802.5. It sits above these protocol at the top of the data link layer. LLC is a subset of HDLC.

In the **Token Bus** Topology the stations pass the tokens by placing the address of the next logical recipient in the header

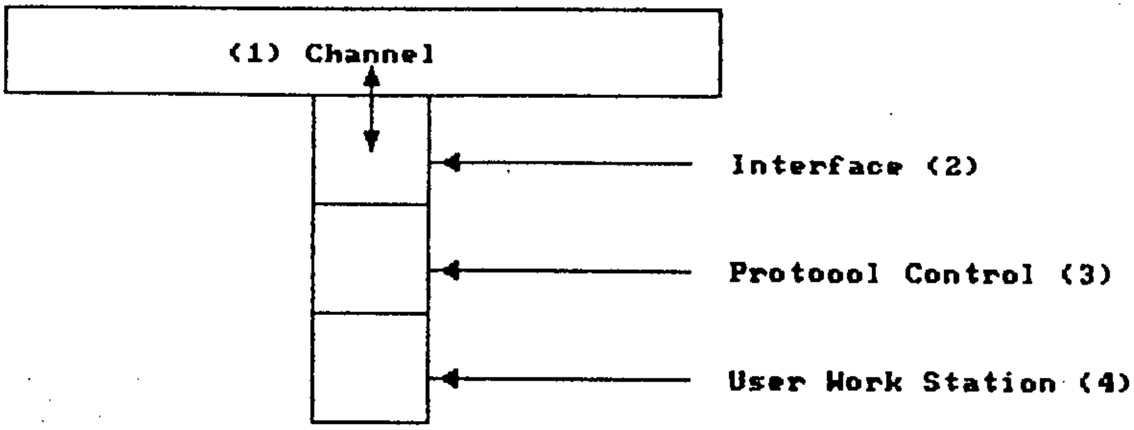


Fig 3.1: Major components of a local area network [22]

of the packet. The message passes along the bus and is monitored by all the stations ;but it is only made available to the destined station based on the sender's placement of the destination address in the destination header. In the event a token is passed to a failed station , the sender will time out ,retransmit a given number of times and eventually transmit to a successor station.

The token is passed from stations in descending numerical order based on station address. When a station hears a token frame addressed to itself , it may transmit data frames. When a station has completed transmitting data frames it passes the token to the next station in the logical ring. When a station has a token , it may temporarily delegate its right to transmit to another station by sending a request with response data frame.

After each station has completed transmitting any data frames it may have , the station passes the token to its successor by sending a token control frame. A point of note here is that the access to the system is always sequential.

Token Ring Technique is based on the use of a small token packet that circulates around the ring . When all stations are idle , the token packet is labeled as free token . A station wishing to transmit must wait until it detects a token passing by.

It then changes the token from free token to busy token by altering a bit pattern. The station then transmit a packet immediately following a busy token. There is now no free token in the ring. So other stations willing to transmit must wait. The packets on the ring will make a round trip. The transmitting stations will insert a new free token on the ring when both of the following conditions have been met.

i. The station has completed transmission of its packet.

ii. The busy token has returned to the station.

There are three kinds of Token operation

i. Single Token Operation

ii. Multiple Token Operation

iii. Single Packet Operation.

In **Carrier Sense** Protocols stations listen for a carrier and then act accordingly . Depending upon this listening the station will decide whether to transmit or to sit idle. When the sensing can detect any collision then it is called **Carrier Sense Multiple Access with Collision Detection (CSMA/CD)** Protocol. Its origin is the University of Hawaii's Aloha Network. Now before the formal analysis of the protocol rules it is necessary to give a brief idea about it.

Suppose in a LAN several users try to use a transmission channel. The CSMA/CD protocol suggests a way of resolving conflicts when several sources attempt to transmit through a single channel. The unit of data transfer between any two stations in the network is called a packet. A packet is a collection of some control and data bits. At any time there will be only one packet in the channel for successful transmission. When a station wants to seize the channel by sending a packet it first listens to the carrier if the channel is busy. If the channel is not busy it starts transmission. But if the channel is found busy the station postpones transmitting to a later time according to a suitable policy. Since electrical signals travel at a finite speed , there is a propagation delay between the instant a station starts transmission and the instant another station listens it. Thus soon after a station seizes the channel another station may sense the channel free (even though the channel is busy) and send a message. This is the *'vulnerable period'*. At this time both messages are destroyed. Shortly after the transmission of a packet the station monitors the cable to verify that; what it 'hears' is what it has transmitted. From this 'hearing' it determines whether the transmission is successful or not. This is called the 'collision detection' phase. In this detection phase if there is a mismatch between the 'sent' and 'heard' messages then transmission is aborted and the time for next transmission is rescheduled. Otherwise the packet is transmitted successfully . Fig 3.2 ,3.3 and 3.4 show the timing diagram and contention interval of CSMA/CD.

3.2 CSMA/CD Protocol policies :

A set of established assumptions of this protocol is furnished below.

i. There are no errors except those caused by collisions.

ii. There is no capture effect.

iii. The random delay after a collision is uniformly distributed and large compared to the packet transmission time.

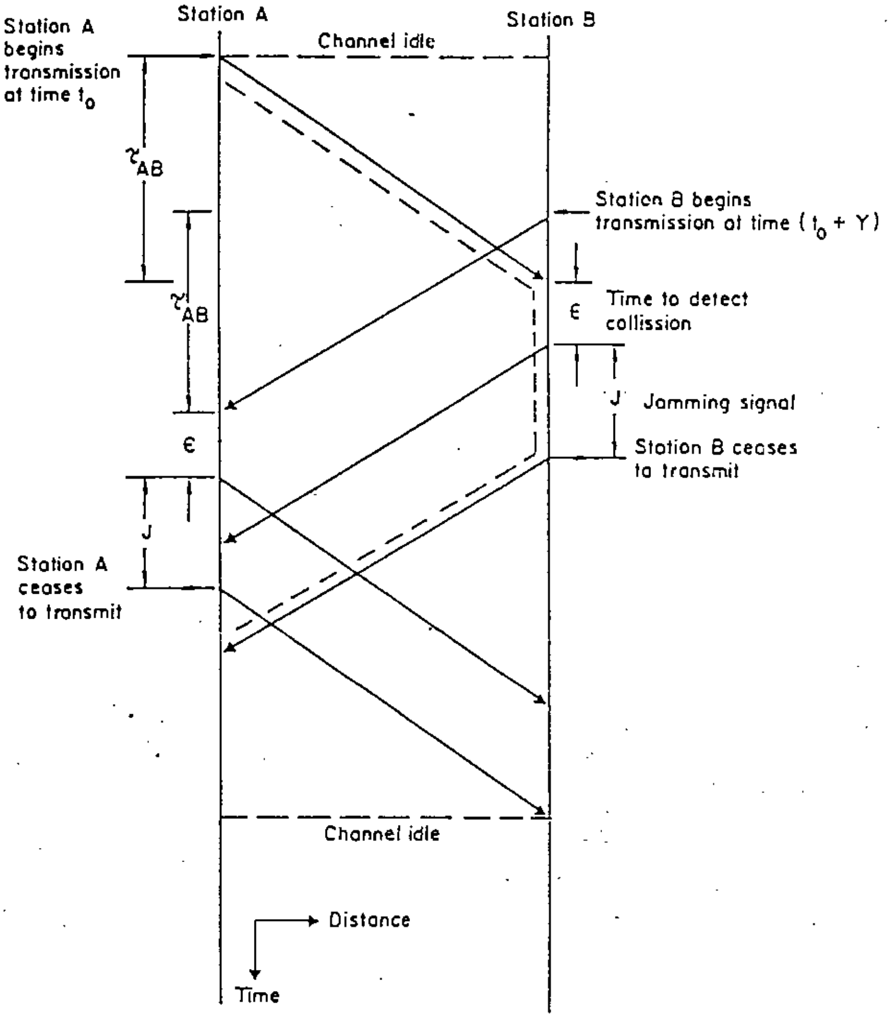


Figure 3.2 Timing Diagram for CSMA/CD, Showing Contention (HAMMOND & O'REILLY, 1986)

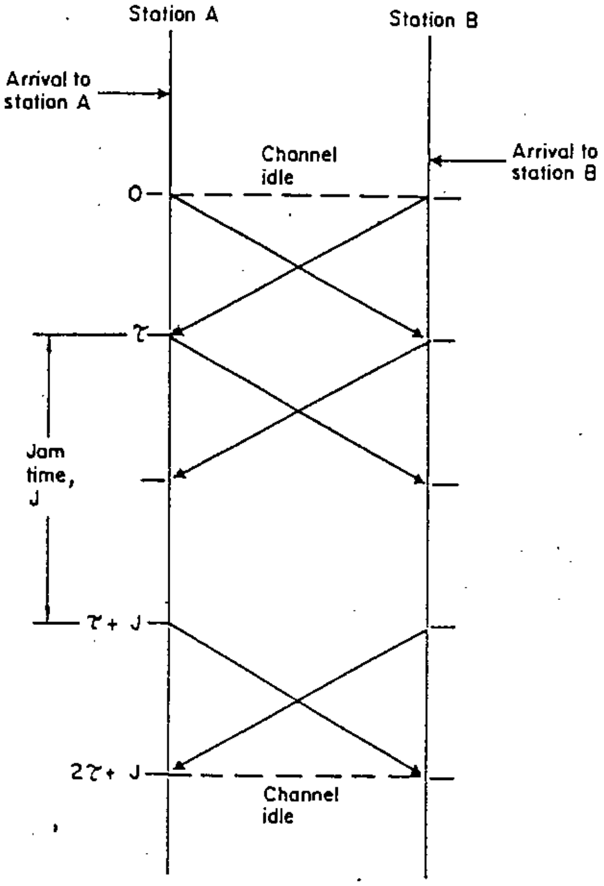


Figure 3.3 A Contention Interval for Slotted CSMA/CD (HAMMOND & O'REILLY, 1986)

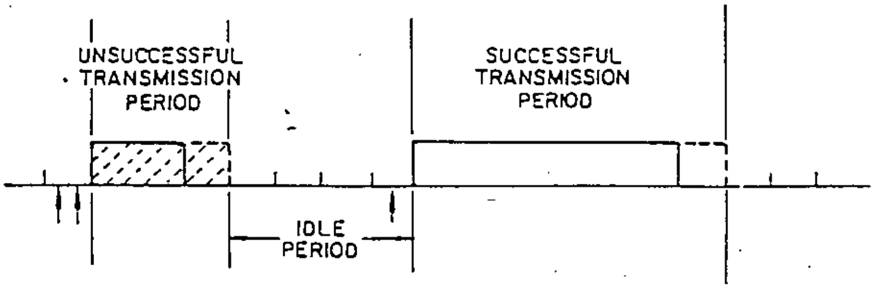


Figure 3.4 Transmission and Idle Periods in Slotted Nonpersistent CSMA - CD. (TOBAGI & HUNT, 1980)

iv. Packet generation process form a poisson process with mean G packets per packet time.

v. All packets are of constant length.

vi. A station may not transmit and listen simultaneously.

vii. Each station can sense the transmission of all other stations.

viii. The propagation delay is small compared to the packet transmission time and is identical for all stations.

ix. Sensing the state of the channel can be done instantaneously.

3.3 Classification of CSMA/CD :

Based upon these rules TOBAGI & KLEINROCK two such protocols in the context of ground radio channels. These are :

1. Non persistent CSMA/CD Protocol
2. P- persistent CSMA/CD protocol

In the nonpersistent CSMA/CD protocol a terminal with a ready packet senses the channel and proceeds as follows.

i) If the channel is sensed idle ,the terminal initiates transmission of the packet.

ii) If the channel is sensed busy , then the terminal schedules transmission of the packet to some later time and repeats the algorithm.

iii) If a collision is detected during a transmission , the transmission is aborted and the packet is scheduled for retransmission at some later time . The terminal then repeats the algorithm .

persistent CSMA/CD) a terminal which finds the channel busy persists on transmitting as soon as the channel becomes free. Thus a ready terminal senses a channel and proceeds as in non-persistent CSMA/CD , except that when the channel is sensed busy , it monitors the channel until it is sensed idle and then with probability one it transmits the packet.

P-persistent protocol is an enhancement of the one persistent protocol by allowing ready terminals to randomize the start of transmission following the instant at which the channel goes idle. Thus a ready terminal senses the channel and proceeds as in the above except that when the channel is sensed busy , the terminal persists until the channel is idle and

i) with probability p it initiates transmission of the packet

ii) with probability 1-p it delays transmission by the propagation delay time and if at this new point of time , the channel is sensed idle then terminal repeats this process (i & ii) , otherwise it schedules retransmission of the packet to some later time.

CSMA/CD can also be classified in two more ways .

Based upon the synchronization :

1. Slotted CSMA/CD : Messages are divided into fixed length slots in time. A packet can start transmission only at the beginning of a slot. i.e. Packets are Synchronized.
2. Unslotted CSMA/CD : A packet can start transmission at any point of time.

Based upon the packet size :

1. CSMA/CD with fixed packet size
2. CSMA/CD with variable packet size

3.4 Hardware Implementation of CSMA/CD :

The system comprises a number of stations or terminals transmitting through a number of channels . The stations are individual computers in the network. Each station has its transmit and receive buffer. It has been said earlier that , before transmitting a packet , a station 'listens ' for a signal and does not transmit until another station's message has passed through the channel. The sender then transmits its message from its transmit buffer. At each receiving station the arrival of a frame is detected, which responds by the sysnchronization of the incoming signal. As the bits are received they are decoded. The receiver checks the frame's destination address field to check whether the frame should be received by its node. It also checks for invalid frames by inspecting the frame check sequence to detect any damage to the frame. CSMA/CD works best on a bus, multipoint topology with bursty asynchronous transmission.

Ethernet is one of the better known and mostly used LAN protocol that includes the primary characteristics of CSMA/CD protocol . It was developed by the Xerox Corporation at its Palo Alto Laboratory in the 1970s and was modeled after the Aloha Network. It uses a shielded coaxial cable for baseband signaling
The data rate is 10 Mbits/sec with a provision of upto 1024 stations on the path. It uses a layering concept, somewhat

similar to the low levels of ISO model and transmits user data in frames or packets. Ethernet uses Manchester encoding .

The Ethernet packet contains six fields . Fig 3.5 shows the packet structure. The preamble is sent before the data to provide for channel stabilization and synchronization. The last two bits of the preamble are coded as 11. Upon reception of 11 , successive data bits are passed into the station. The destination and source address identify the receiving and sending stations, respectively. The type field is used by the end users at a higher level in the local network. It is defined in the Ethernet level to provide for a uniform convention between higher levels. The data field contains user data. The frame check sequence field provides for a cyclic redundancy value.

Layers of Ethernet : Ethernet is designed around three layers depicted in fig 3.6. [22]

i) The user or client layer is the workstation,

ii) The data link layer contains the data encapsulation and link management functions

iii) The Physical layer provides for data encoding /decoding and channel access.

Preamble	Destination	Source	Type	User Data	CRC Field
64 bit	48 bit	48bit	16 bit	368 bits to 12000 bits	32 bit

Fig 3.5: Ethernet Packet

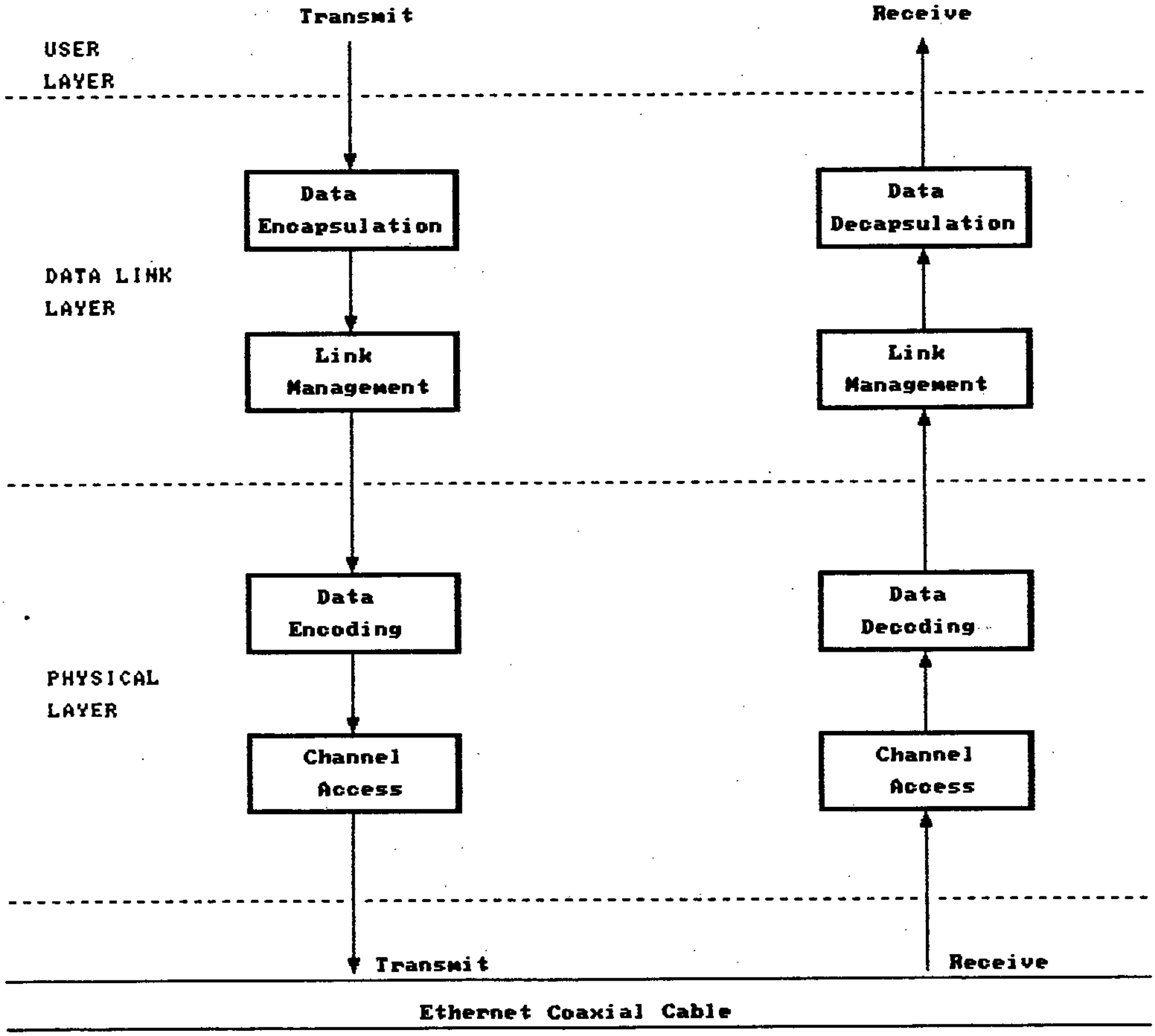


Fig 3.6: Ethernet Layers

Chapter 4

Simulation Program Development

4.1 Problem Formulation :

A discrete event driven simulation program is developed for the Multiple channel CSMA/CD system for the unslotted nonpersistent case. The following are the possible events in this system.

i) Successful Transmission of a packet of a station with no other station within its packet transmission time . In this case a station while attempting to seize a channel finds it idle. No collision occurs among the stations. Also in the period of transmission of this station there is no other station having ready packet in its queue. So no other station faces delay.

ii) Successful transmission of a packet with some other stations having ready packets within the transmission time of that packet. In this case the other ready packets face delay . So they have to reschedule their attempts. Delay faced by one of these stations is calculated as follows.

Delay=Time of the transmitting station - Time of the present station +Packettime.

iii) Unsuccessful transmission of a packet due to the simultaneous attempt of transmission of several packets. This happens when several stations ,finding a channel busy try to transmit within a period equal to the propagation delay time. This results in a collision. The colliding stations reschedule their attempts according to the negative exponential distribution. The mean time for negative exponential distribution is taken to be ten times the propagation delay time.Rescheduling means generation of new arrival time for the next attempt and a new choice of channel.

iv) A packet may subsequently face collision for many times. In our case we have allowed a maximum number of collision to be 16. After that the packet is aborted . The station then generates new time for a new packet.

4.2 Protocol Assumptions :

In the light of the general CSMA/CD protocol formulation discussed in chapter 3 the following set of assumptions and modifications are taken for the development of the simulation of the protocol.

A. The protocol is taken to be unslotted or asynchronized nonpersistent. Because the assumption of synchronized transmission is unrealistic .

B. All the stations are statistically identical. There is no priority criterion for any of the stations .

C. All the channels are also statistically identical .

D. All the channels are bi-directional.

E. There can be at most one ready packet in the buffer of any station.

F. Data packet generation rate is determined by the mean interval of generation which is assumed to be Negative Exponentially distributed with mean rate of 'L' packets per second .

G. The propagation delay 't_p' between any two stations in the system is a constant which is taken to be the delay between the two farthest stations. Of course this assumption will lead to a conservative estimates of the performance measures.

H. Transmission channel is totally noiseless .

I. Retransmission interval is negative exponentially distributed.

The following protocol policies are also assumed.

i. When a packet faces collision its retransmission is scheduled using the Negative Exponential Algorithm. This algorithm allows the initial attempt plus 15 retransmission, each delayed by a time which is negative exponentially distributed with base back-off time of 10 times the propagation delay.

ii. A packet sensing the channel busy during a successful transmission of another station is rescheduled after a random delay which is negative exponentially distributed with mean rate of twice the packet transmission time .

iii. The collision time is determined by the following method shown in fig 4.1 . Suppose the instant at which a station initiates transmission is t_a. On the assumption the busy period results in a collision , let t_b be the instant at which the first colliding station starts transmission , where for a collision to occur $t_a<t_b<t_a + t_d$. Under the assumption of identical propagation delay between any two station the station which initiated the unsuccessful busy period (Collision period) notices the collision at $t_b+ t_d$, while the other transmitting stations detect the collision at $t_a +t_d$.

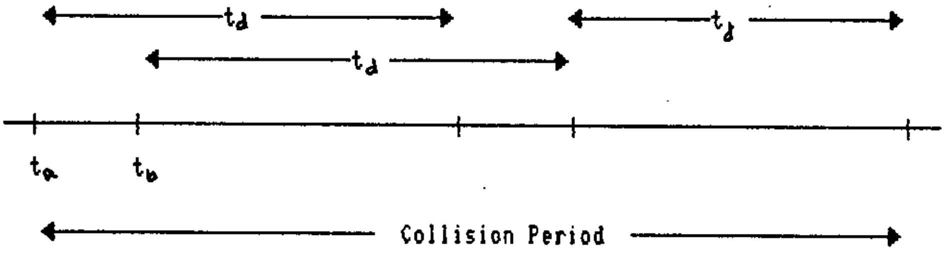


Fig 4.1 : Collision Slot in Unslotted CSMA/CD

Therefore in the unslotted CSMA/CD , the last station that aborts the colliding signal is the station which initiated the channel busy period. It must be noted that in this case the length of the unsuccessful busy period is $t_b - t_a + t_d + t_j$. The last term is the time for transmitting a Jamming signal notifying all the stations of the occurrence of a collision . So regardless of the number of colliding stations and their packet lengths , the length of the unsuccessful busy period is the same.

4.3 Model Building

4.3.1 Development of the Physical System of Multiple channel CSMA/CD :

The system is assumed to consist of "ns" stations connected to each of "nc" buses or channels. Each station has both a receive and a transmit capability on every channel. The method of transfer of access opportunity is a stochastic process. The hardware capability options that can be considered are follows :

(a) Each station is capable of receiving transmissions from all the channels simultaneously. That means the number of receive buffers is equal to the number of channels.

(b) Each station can transmit (data or control) on one bus at a time. It means that only one transmitter can be enabled at a time.

(c) Any overlapped received and transmit operations at a station is allowed.

(d) Each station process carrier sensing capability on all the channels simultaneously.

(e) Each station is able to detect packet collision on the channels on which it is transmitting.

(f) Collision detection time is negligible .

4.3.2 Development of the logical system :

Fig 4.2 shows the flowchart of the course of action a station will take when it is ready to transmit a packet through a channel. Fig 4.3 shows the flowchart of the whole simulation process considering all the stations ,channels and other system entities. The processes in the flowchart are discussed in the following topic.

4.4 Subsystems of the Simulation Process :

fig 4.3 describes the subsystems of the simulation model .

4.4.1 Declaration of variables and Data Acquisition :

Most of the symbols used in my simulation program are self explanatory. Still for further documentation all the variables used are defined in the program listing .

A) Variables related to the overall system performance

B) Variables related to the channels

C) Variables related to the stations.

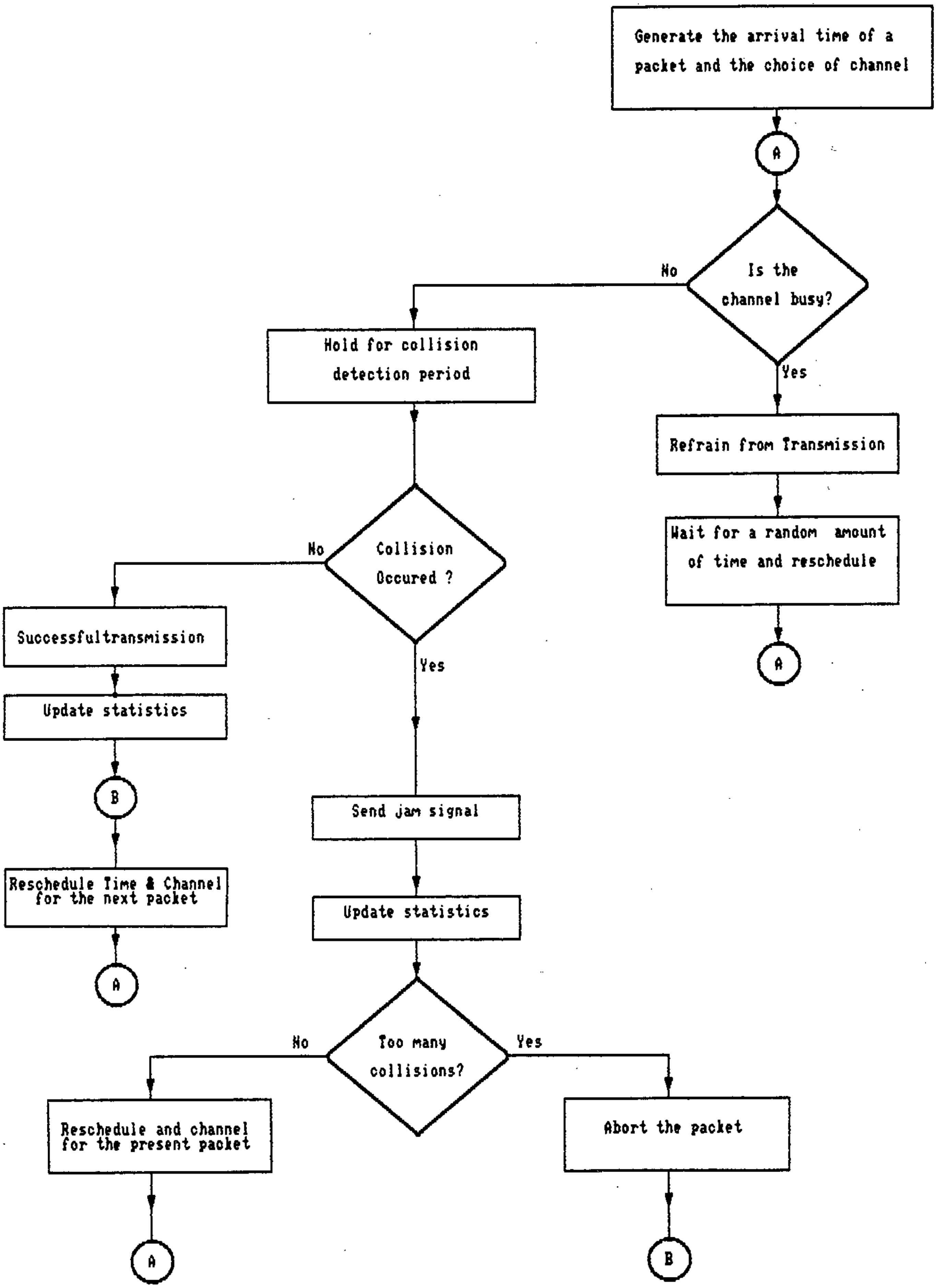
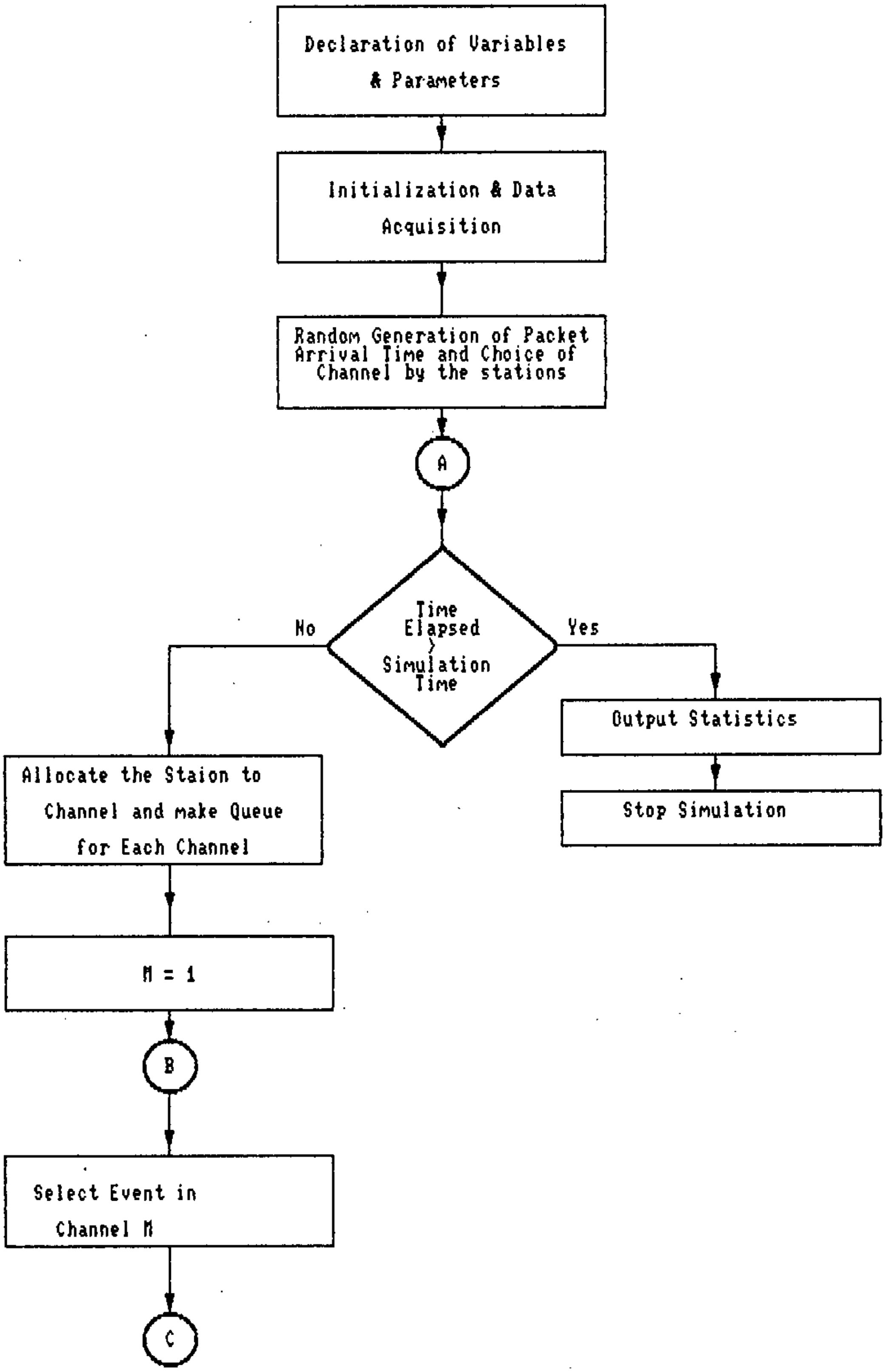


Fig 4·2 : Flowchart of the path followed by a Station



CONTINUED IN THE NEXT PAGE ....

Fig 4.3 : Flowchart of the Simulation Process

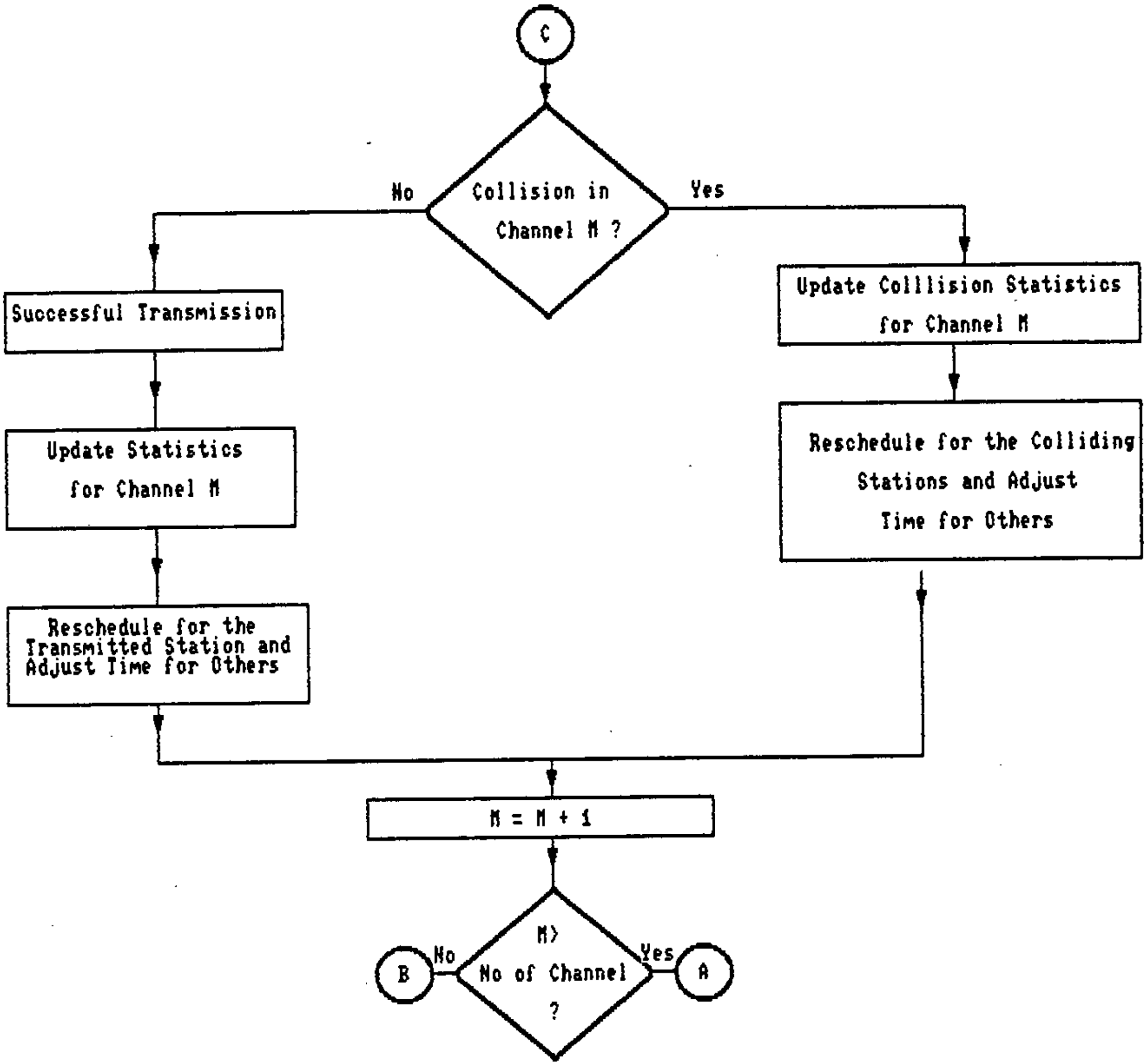


Fig 4.3: Flowchart of the Simulation Process

To make the program more interactive the following variable values are taken from the user response.

i) Simulation Time
ii) Length of the LAN
iii) Arrival Rate in packets per second
iv) Packet size in bits per packet
v) Transmission speed
vi) Number of stations
vii) Number of Channels

4.4.2 Random Number Generator & Distribution Policy

We have used the most commonly used multiplicative congruential method.

$X_i=(cX_{i-1} + b) \bmod m$

Here b is taken to be 0

c is a constant which has some experimental values. A value most commonly used is = 7^5

m= Largest prime less than $2^{31} = 2^{31} - 1$.

To perform a higher degree of randomness another step is done in this module. That is , for one value of X_{i-1} 8 values of X_i are calculated . And from these 8 values one value is taken at random.

The generated random numbers are used in the fixation of the arrival time of a packet of a station and also for the choice of channel. The size of the packets in variable packet size is also randomly chosen over a mean value.

Now the distribution policies considered are

1) Negative Exponential Distribution
2) Uniform Distribution and
3) Constant Packet Length

4.4.3 Packet Arrival Time and Channel selection Generator :

Arrival Process is taken to be negative exponentially distributed. The mean rate of packet arrival is varied in the program to have the desired results . Each station chooses a channel randomly. At the start of the simulation the system goes through this module. This module is also executed in every replication of the simulation.

A queue is made for each channel in this module. The first station in the queue is the station with the least arrivaltime.

4.4.4 Event Selector :

For each channel a possible event is determined in the following way.

The arrival time of a packet of all the stations is compared with that of the first station in the queue. If any station is within the propagation delay time of the first station then a collision is said to be occurred. More than two stations may be engaged in such a collision. If there is no station within the propagation delay of the first station then no collision occurs and the first station in the queue is allowed to transmit successfully.

At this time another event may also occur. Suppose a station has started transmission ; But there may be some other stations trying to send packet during the transmission period of this station. This event is determined by comparing the arrivaltime of all stations with that of the first station . If the difference of this two time is less than the current packet transmission time than the stations except the first station in the queue have to wait.

4.4.5 Collision Module :

From the collision window shown in fig 4.1 it is clear that the time lost in collision for a station is not a constant . This real situation is considered in the present simulation . In most of the simulation done earlier this was taken to be a constant. When a station faces a collision it reschedules the transmission time of its packet according to the negative exponential distribution. The mean is taken to be the 10 times the propagation delay. It might have been taken otherwise. Binary Back-off Algorithm is used most frequently. In this model , a station after facing a collision chooses another channel at random. The following statistics are updated in this module :

i. Time Elapsed

ii. Delay time of each station

iii. Delay time of each channel

iv. Number of aborted packets

v. Queue Length

vi. Number of Collision of each station and channel

4.4.6 Successful Transmission Module :

If in a channel there are no other stations within the propagation delay time of the first station of that channel then that station starts transmission successfully. The successful station then schedules the time and channel for its next packet . The time for all other stations are updated If however there are some other station shaving ready packet within the packet transmission time of the first station then all these stations wait for a random amount of time and reschedules transmission according to the negative exponential distribution having mean time of 2*packettime . The following statistics are updated in this module :

i. Time Elapsed

ii. Delay time of some of the stations

iii. Delay time of each channel

iv. Number of transmitted packets of each station and each channel

v. Queue Length

4.4.7 Output of statistics Module :

Here the updated statistics generated in the previous two modules are collected . Stations related outputs ,channel related outputs and the overall system outputs are considered separately.

4.5 Starting and Stopping Policies :

The simulation process is started at idle state. All the parameters are initialized to the idle state in the initialization block. The length of the simulation time is chosen in such a way that the total packet transmitted from any of the channel is about 1000. This limit has been chosen using trial and error method.

For each set of parameters 5 runs are carried on. The final statistics generated is the average of these 5 runs. So the number of replication of the simulation run and the length of the simulation run have been found within the satisfactory limit. The validation techniques and the experimental process of this simulation are discussed in the following chapter.

Chapter 5

Result & Discussions

5.1 Definition of the Performance Parameters :

1. Utilization : This represents the percentage of the total capacity of the channels in a CSMA/CD system which is being utilized. For a single channel it is defined as

$$\text{Utilization} = \frac{\text{Total Transmission time}}{\text{Total Simulation Time}}$$

Let the packet transmission time of the ith packet is $= t_i$, The number of total transmitted packet = N

Idle Time = T_d , Collision Time = T_c

Then

$$\text{Utilization} = \frac{\sum_{i=1}^{N} t_i}{T_d + T_c + \sum_{i=1}^{N} t_i} \qquad (5.1)$$

Now Idle Time is the time during which there is no transmission through the channel. Collision Time is the time lost in collision in a channel.

Utilization of the system is taken to be the average of the utilizations of all the channels. i.e. Utilization of the system ,

$$U = \frac{\sum_{i=1}^{nc} U_i}{nc} \qquad (5.2)$$

Where, U_i is the utilization of the ith channel. nc is the total number of channel.

2. Delay : A packet faces delay if it cannot be transmitted at its scheduled point of time. So delay for a packet is defined as the difference in time between the instant a packet is scheduled for transmission after the generation and the instant it actually starts its transmission. Normalized delay is the ratio of this time to the actual time needed for transmission. That means it represents how many actual transmission time unit is needed .

This parameter is calculated as follows :

$$\text{Normalized Delay} = \frac{\text{Total Delay faced by all the packets}}{\text{Total Transmission Time of all the packets}}$$

$$= \frac{\sum_{i=1}^{N} d_i}{\sum_{i=1}^{N} t_i} \qquad (5.3)$$

Where d_i is the delay faced by the ith transmitted packet.

3.Throughput : It is the actual number of Mega bits per second transmitted. It is calculated by incrementing the number of bits transmitted after the transmission of a packet. Suppose in t_s Second simulation time the number of bits transmitted is equal to M on all the channels . Then ,

$$\text{Throughput} = \frac{M}{t_s} \text{ bits/sec} \qquad (5.4)$$

4. Bandwidth : This is expressed as the transmission speed in Mega Bits per Second of a channel.

5. **Capacity** : Maximum possible utilization of a channel is called its capacity.

6. **Maximum Offered Load** : It is the theoretical maximum offered load expressed as the percentage of total bandwidth and calculated in the following way .

$$\text{Load} = \frac{\text{Mean Arrival Rate} * \text{Mean Packet Size in bits per packet} * \text{Number of stations}}{\text{Transmission Speed} * \text{Number of channel}}$$

$$\text{or, } L = \frac{\lambda * ps * ns}{B * nc} \qquad (5.5)$$

Let us take an example.

Mean Arrival Rate = 1000 packets /sec

Packet Size = 5000 bits/packet

Number of stations = 20

Number of channels = 4

Transmission speed = 10 MBPS

Then ,

$$L = \frac{1000 * 5000 * 20}{10000000 * 4} = 2.5 = 250\%$$

7. **Queue Length** : It is defined as the number of stations waiting for transmission within the transmitting period of a successfully transmitting station in the same channel. Suppose in a channel a station is trnsmitting successfully with its

packettime = t_m. Now if the number of stations trying to transmit within this time in the same channel is equal to 'Q' , then we say that these 'Q' stations are queued in that channel. So the queue length is equal to 'Q'.

5.2 Validation of the simulator :

The random number generator is tested by running it separately. The period of generation is satisfactorily within the tolerance. The nature of the performance parameters found from the simulation run is found to be within a very close approximation to the analytical results and results found from simulation carried elsewhere on the same. The slight variation is because of the variation in retransmission policy adopted. In my model a packet, if suffers more than 16 collisions is aborted. So a separate run was given for the simulator withdrawing this condition. The resulting delay - throughput characteristics was compared with the results in [20] . The nature of the curves found from my simulation was found to be in close approximation to that of the above mentioned paper. The result is shown in Fig 5.17 which is discussed below . The difference is due to the difference in reschedule policy and the packet abortion policy.

The simulator was also tested using experimental results and comparing these with the theoretical results which are discussed in the following chapter.

5.3 Experimentation :

A set of different experiments is done using the simulator for the performance analysis of the CSMA/CD Protocol. These are :

A. Fairness test of the protocol

B. Effect of number of channel on the delay - utilization characteristics of CSMA/CD

C. Effect of number of stations on the delay - utilization characteristics of CSMA/CD

D. Effect of varying the offered load by varying the arrival rate of packets

E. Effect of the packet size

F. Effect of the packet size distribution policy.

Now using the Simulator the following results are obtained.

A) Effects of Number of channels(Buses) on the performance :

Two Characteristic curves are discussed here. These are :

I. Utilization vs Number of Channels (Fig 5.1)

II. Normalized Delay vs Number of Channels (Fig 5.2)

Only the Constant packet distribution is considered.

Number of stations=20

Number of channels are varied from 1 to 8

Packet size is taken to be 500 microseconds

Fig 5.1 & Fig 5.2 are drawn for 3 different values of arrival rate having values 500, 1000 and 1500 packets per second .

the following observations are made from Fig 5.1 :

1. For the same packet arrival rate utilization decreases with the increase of number of channels. This is because the total load offered to the system remains constant . So the increase in capacity of the system with the increase of channel will not be utilized to that extant. According to the definition of offered load , the load which is 500% for a single channel will only be 100% for 5 Channels.

2 With the increase of arrival rate the rate of decrease of utilization with increasing number of channels is getting smaller. The following statistics supports this fact .

For 500 packets/sec ,
Utilization for 1 channel = .85
Utilization for 8 channels = .46
Difference = .85-.46 =.39

For 1000 packets/sec ,
Utilization for 1 channel = .895
Utilization for 8 channels = .53
Difference = .89-.53 =.36

For 1500 packets/sec ,
Utilization for 1 channel = .92
Utilization for 8 channels = .60
Difference = .92-.60 =.32

Fig 5.2 shows the variation of Normalized Delay with the number of channel. It is found that :

I. For the same offered load increasing number of channel will decrease the delay. Because in multiple channel a packet can have the option of choosing one from many channels.

II. If we increase packet arrival rate then delay increases in all cases. But the rate of increase is smaller for larger number of channels. If arrival rate is changed from 500 to 1500 then

Increase in delay for 1 channel = 7.6 - 5.2 = 2.4

Increase in delay for 8 channels = 2.84 - 1.96=.88

B) Effects of Number of stations on the performance :

Here also two Characteristic curves are discussed . These are :

I. Utilization vs Number of stations (Fig 5.3)

II. Normalized Delay vs Number of stations (Fig 5.4)

Only the Constant packet distribution is considered.

Number of channels is kept fixed at 4.

Number of stations is varied from 5 to 40 in step of 5

Packet size is taken to be 500 microseconds

Arrival Rate = 1000 packets per second

Variation of utilization with respect to the the number of stations (Fig 5.3) shows that Utilization increases with the increase in station . But the rate of increase in utilization is getting smaller with increased number of stations. The change of utilization from station 5 to 10 = .63 - .53 =.10. The same for increase of stations from 35 to 40 is = .81 - .79 = .02.

Fig 5.4 shows the change of normalized delay under the above mentioned parameter values.

It is clear that as more number of stations are entered into the system ,more packets try to occupy the same number of channels . The result is the increase in delay.

C) Fairness Test of the Protocol :

The protocol is assumed to have no bias to any of the channels or stations. This assumption is validated by the simulation run in the following way.

1. Fairness to the channels : A simulation run is performed with the following set of parameters.

No. of stations =20

No. of channels =4

Mean Packet size =5000 bits/packet

Mean Packet arrival rate =400 packets/second

For constant packet Distribution Table 1 shows the result found from the simulation run .

From the tabular data the mean and standard deviation of the performance parameters are found as follows. (Table 2)

Constant Packet Distribution

Number of Channels = 4

Number of Stations = 20

Arrival Rate=400 packets per second

Mean Packet Size=5000 bits per packet

Channel Number	Packet Transmitted	Channel Utilization	Normalized Delay	Queue Length	Channel Throughput
0	1003	0.73	4.89	1.96	7.29
1	1071	0.81	5.33	1.94	8.06
2	1065	0.78	5.27	1.64	7.77
3	1044	0.76	5.06	1.67	7.62

Table 1 : Performance Parameters of the channels in a simulation run

Performance Parameter	Mean	Standard Deviation
No. of Transmitted Packets	1030	18.89
Channel Utilization	.7775	.021
Normalized Delay	5.14	.162
Queue Length	1.91	.066

Table 2 : Mean & Standard Deviation of Performance Parameters of channels

Constant Packet Distribution

Arrival Rate=400 packets per second
Mean Packet Size=5000 bits per packet

Station No.	Message Transmitted	Message Aborted	No. of Collisions	Delay (microsecs)	Transmission Time
0	212	0.67	6.33	5.00	106250
1	222	0.17	9.67	4.73	111083
2	193	0.50	5.83	5.70	96500
3	207	0.33	8.83	5.14	103500
4	200	0.17	8.83	5.39	100000
5	214	0.17	9.50	4.99	107083
6	205	0.50	8.00	5.23	102750
7	208	0.33	8.83	5.18	104000
8	204	0.33	7.17	5.31	102000
9	212	0.17	9.67	5.08	106333
10	211	0.50	5.17	5.09	105750
11	207	0.33	8.33	5.26	103500
12	195	0.33	6.50	5.58	97750
13	221	0.33	7.83	4.77	110583
14	219	0.50	6.17	4.83	109583
15	214	0.33	9.33	5.04	107417
16	208	0.50	5.83	5.23	104333
17	204	0.33	8.33	5.26	102250
18	221	0.33	8.67	4.81	110667
19	202	0.17	7.67	5.48	101000

Table 3 : Performance Parameters of the stations in a simulation run.

The pie chart (Fig 5.5) and the standard deviation figures clearly shows the fairness of the protocol.

2. Fairness to the stations : Parameter values were taken as above. Table 3 shows the result. Mean and Standard Deviations of the performance parameters have been found as follows.

Performance Parameter	Mean	Standard Deviation
No. of packet transmitted	208.95	.0382
Average Delay	5.15	.257
Average no. of Collision	7.82	.138

Table 4 : Mean & Standard Deviation of the performance Parameters of the stations.

D) Effect of packet size Distribution and variable packet size

The following three distribution policies are considered .

i) Negative Exponential

ii) Uniform Distribution

iii) Constant Packet Length

Variable nature of the packet size is represented by the ratio ´a´ which is defined as the ratio of propagation delay to the mean packet transmission time. Following set of data is used.

No. of stations =20

No. of channels =4

Packet size is varied from 50 bits/packet to 5000 bits /packet so that the ratio ´a´ is varied from .01 to 1.

Arrival Rate = 500 packets/second.

Fig 5.7 and 5.8 show the relation between utilization and the parameter 'a' for the three distributions. As the arrival rate is kept fixed , so variation of idletime is very small. Therefore dependency of utilization on the collisiontime becomes prominent(eq. 5.1). At higher value of 'a' , packet transmission time is small ; so collisiontime is comparable to packettime . Hence utilization decreases. Therefore, The general trend of the graph is that utilization decreases with increasing value of 'a' for all the three aforementioned distributions.

Another important finding of this experimentation is that for the same value of 'a' utilization is maximum for Constant packet length ; Utilization is minimum for Negative Exponential Distribution. The Utilization for Uniform Distribution is in between these two.

Fig 5.9 shows the dependency of average collision on the ratio 'a'. Higher values of 'a' gives more collision . Of course the behavior is somewhat random.

E) Effect of packet arrival rate :

All the three above mentioned packet distribution policies are considered.

No. of stations =20

Packet length = 5000 bits per second Arrival rate for each of the above packet length is varied in such a way so that the system offered load varies from 20% to 400% for channel numbers 1 to 4 .

Fig 5.10 to 5.13 show the most important Delay - Throughput

Characteristics of CSMA/CD protocol. Four graphs are drawn from the results for four different number of channels keeping all the parameters constant. The number of channels chosen are 1,2,3,4 for 20 stations in the system. Following inferences can be drawn from each graph.

i) For the same utilization Negative exponential distribution of packets results in highest delay, Whereas constant distribution gives lowest delay and uniform distribution is in between these two.

ii) Of course the variation is not so prominent , especially for lower utilization. This is obvious , because at lower utilization the channel remains free for a larger time. So the chances of collision are rare. So delay is small and not so dispersed among different distribution.

iii) The delay for all three distributions are found to increase with increasing offered load This is because more packets are now in the system waiting for transmission

iv) As the number of channels is increased the normalized delay is found to decrease for the same utilization for all three distributions.

For Constant packet length and at 80% Utilization the normalized delay figures for channel 1,2,3 and 4 are 12.5, 9.4,6.7 and 5.6 respectively.

Figures 5.14, 5.15 and 5.16 show the delay utilization curve for a small packet size of 500 bits per packet. Number of channels taken in the three cases are 2, 4, and 8. If we compare the results obtained with the mean packet size of 5000 bits per second we see that the utilization decreases with 500 bits per second case for the same number of channel and stations and other parameters. So decreasing the mean packet size is found to decrease the utilization.

To investigate the effect of arrival rate on the delay throughput characteristics for different number of channel a separate run was undertaken. Packet length is taken to be negative exponentially distributed with mean of 5000 bits. The run was done using 1,2,4,and 8 channels. Arrival rate in the system was varied in such away so that for 8 channels the offered load varies from 10% to 100%. This is equivalent to 80% to 800% for the single channel case which means very high offered load . As a result for 1 channel the utilization is found to be maximum with a value of 92% . The normalized delay also has the maximum value of 21.5. With the increase of number of channels the normalized delay decreases . For 2 , 4 and 8 channels the maximum delay for the given maximum load are 13.5,7.6 and 3.42 respectively. Another improvement in adding new channels is the significant increase in the throughput. With the given maximum offered load throughput values are 9.2, 16.1, 27.8 and 51 Mbps for 1, 2, 4 and 8 channels respectively.

F) Delay - Utilization Characteristics with no limit on the number of collision :

All the above characteristics were found using a maximum limit on the number of collision a packet may face. This limit was taken as 16. That means a packet after facing collision for 16 times is aborted. Therefore the delay utilization curves were found almost monotonic. So a separate run was given letting a packet suffer as many collision as possible before successful transmission. Interestingly enough this time , with the increase of load the utilization was found to decrease with sharp increase in delay. The results found completely matches with the theoretical result and other simulation results. Figure 5.18 clearly explains this feature. The maximum utilization point is very close for all three distributions, having .81 for constant,.79 for Uniform and negative exponential distribution. After this maximum utilization , as the offered load is further increased utilization was found to decrease with increasing delay. For this curve, the maximum delay is found for the Negative exponential Distribution which is equal to 6.65, the minimum is for constant having 5.14. This result suggests a maximum value of the offered load. That means in a given system, increasing offered load will result in a better delay - utilization curve upto a certain limit. If after that limit the load is increased by increasing the arrival rate then the delay - utilization characteristics degrades sharply.

G) Effects of 'a' on the Capacity of the channel :

In this experimentation packet size was varied in such a way so that 'a' can be varied from .01 to 1. For each value of 'a' the arrival rate was varied so that the load varies from 20% to 2000% to get the maximum possible utilization for a certain parameter set. By plotting the maximum utilization for a particular value of 'a' Fig 5.19 was found. The curve shows that the efficiency of the system decreases as the increase of 'a'. For Ethernet the maximum utilization or efficiency is found from the following formulae [22]

$$\text{efficiency} = \frac{1}{1 + 5.4*a}$$

The experimental result and the theoretical result is compared in the following table.

a	Efficiency from Simulation	Efficiency from Analytical result
.01	.927	.9487
.02	.873	.9025
.0333	.794	.8476
.05	.729	.7874
.1	.624	.6494
.2	.452	.48
.4	.293	.3164
.8	.167	.188
1	.15	.1563

Table 5 : Comparison of Analytical & Simulation results of the maximum utilization vs 'a'.

The slight variation is due to the difference in model of Ethernet and the present protocol.

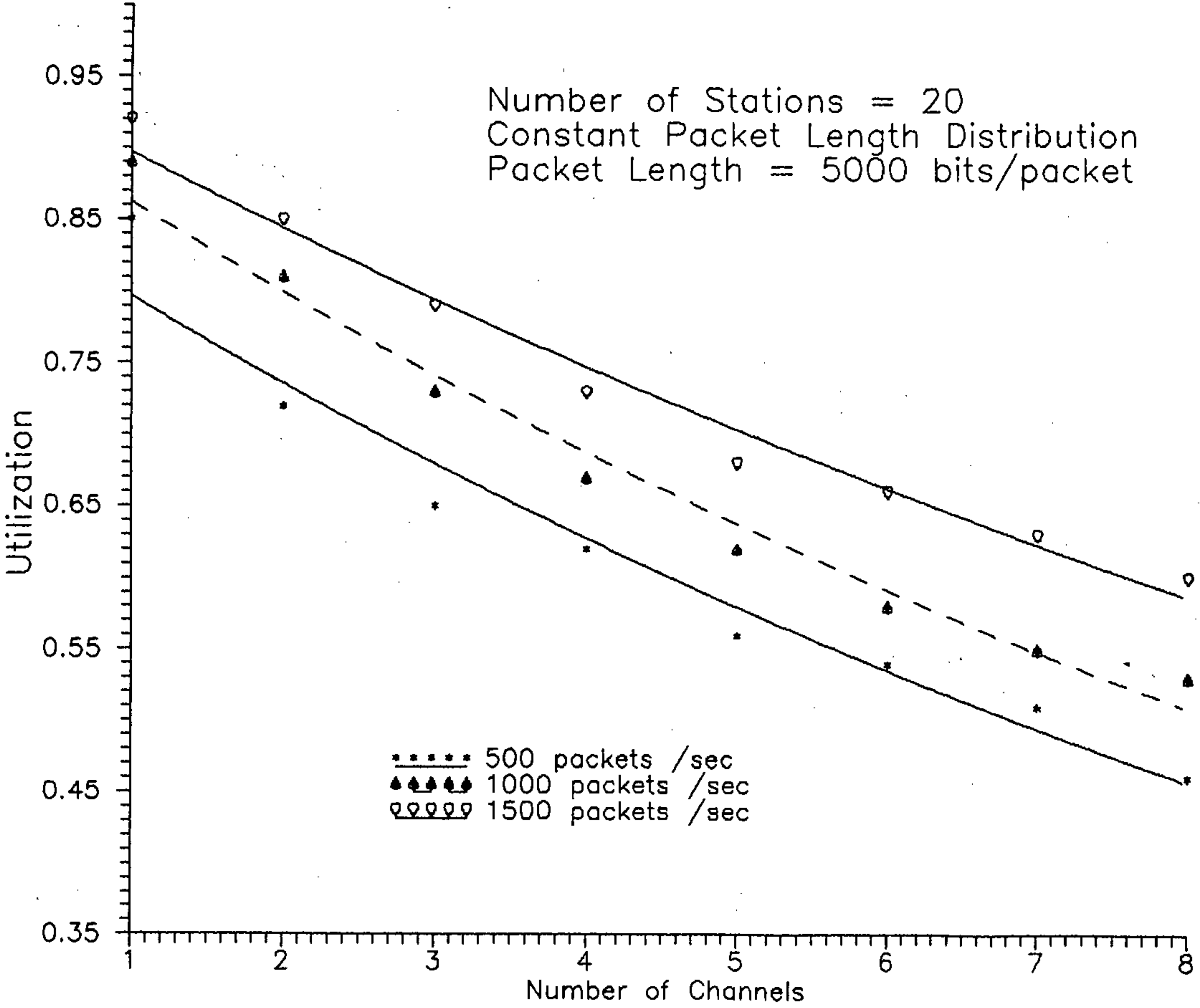


Fig 5.1 : Utilization vs Number of channel for constant packet length

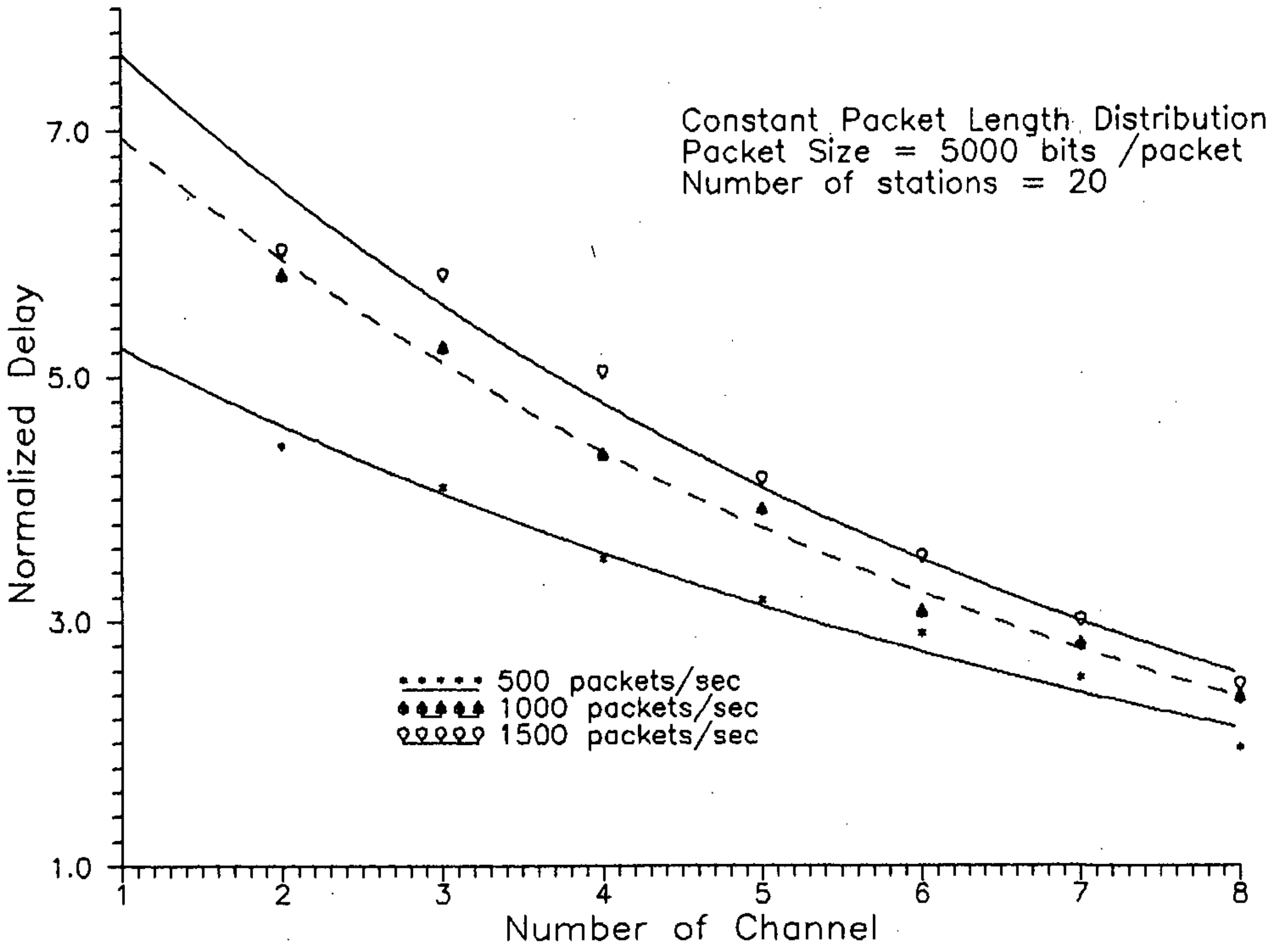


Fig 5.2 : Delay vs Number of Channel for Different Arrival Rate

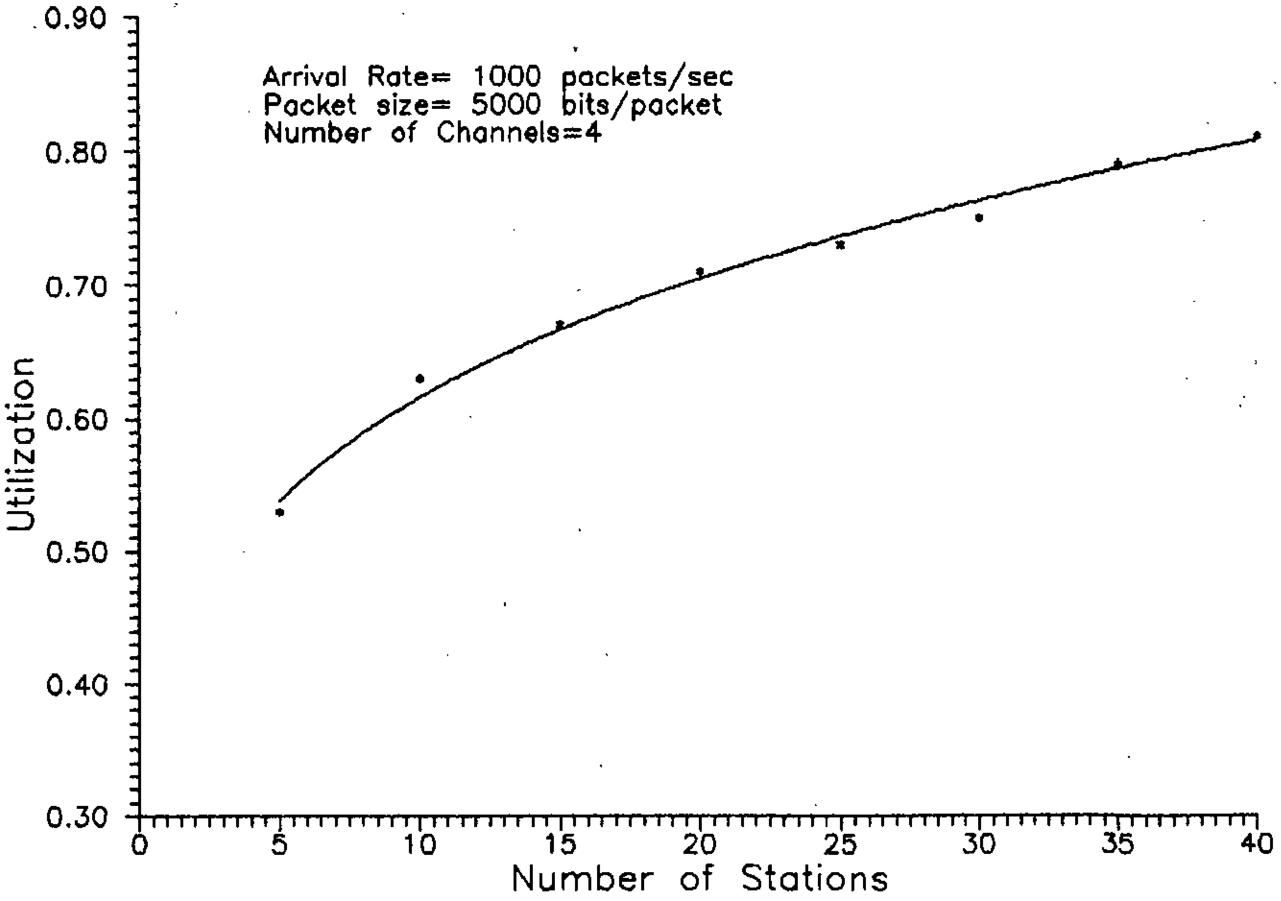


Fig 5.3 : Effect of Number of Stations on Utilization

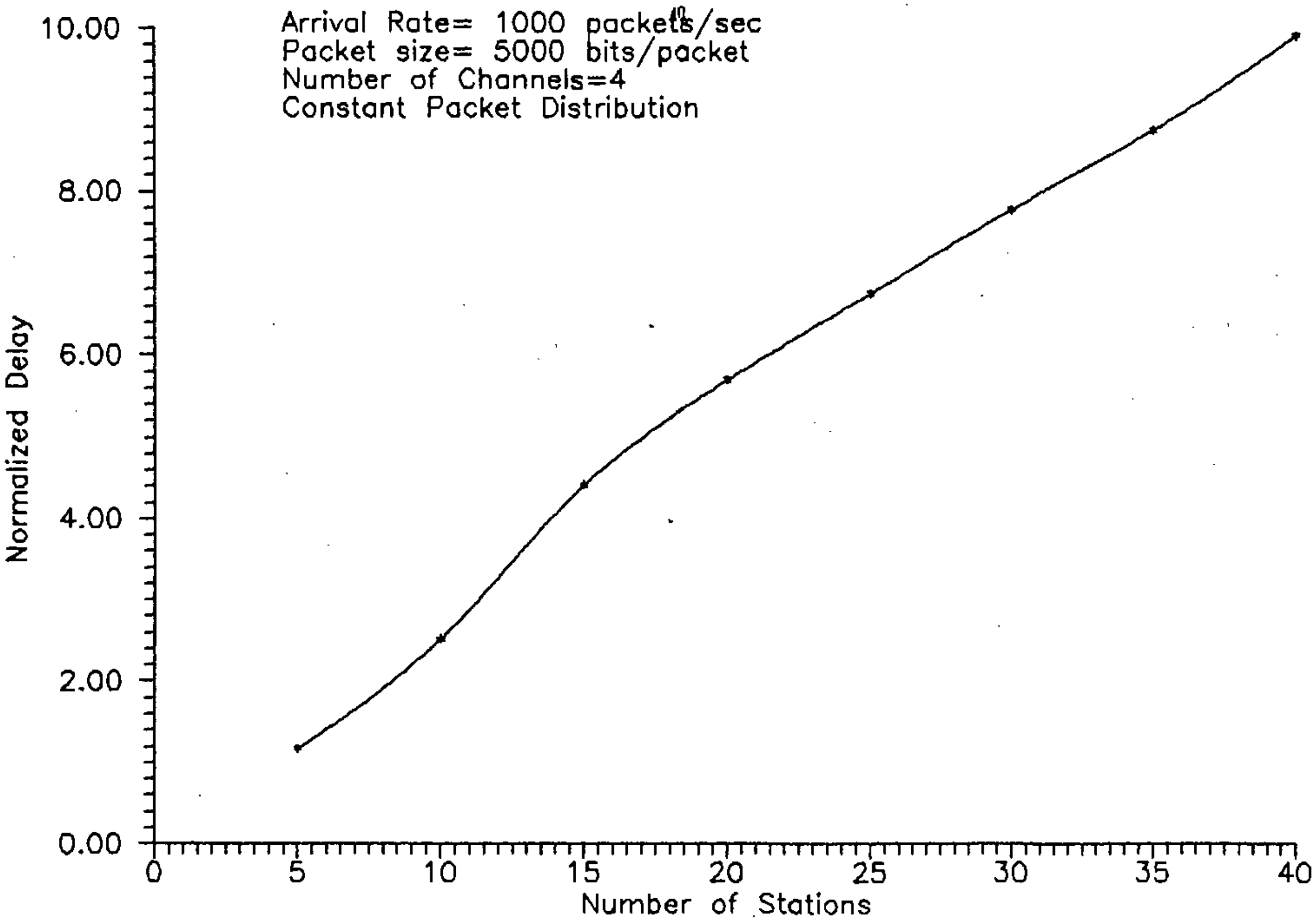


Fig 5.4 : Effect of Number of Stations on Delay

Percentage of Transmitted packets
among the channels.

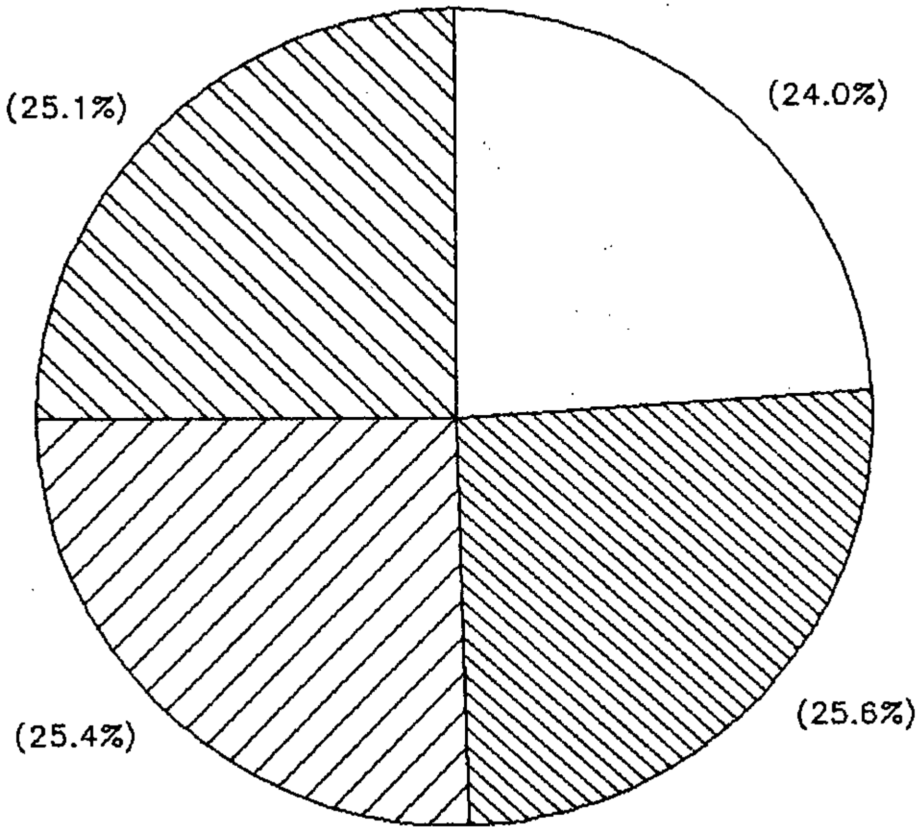


Fig 5·5

Percentage of Transmitted packets among the stations.

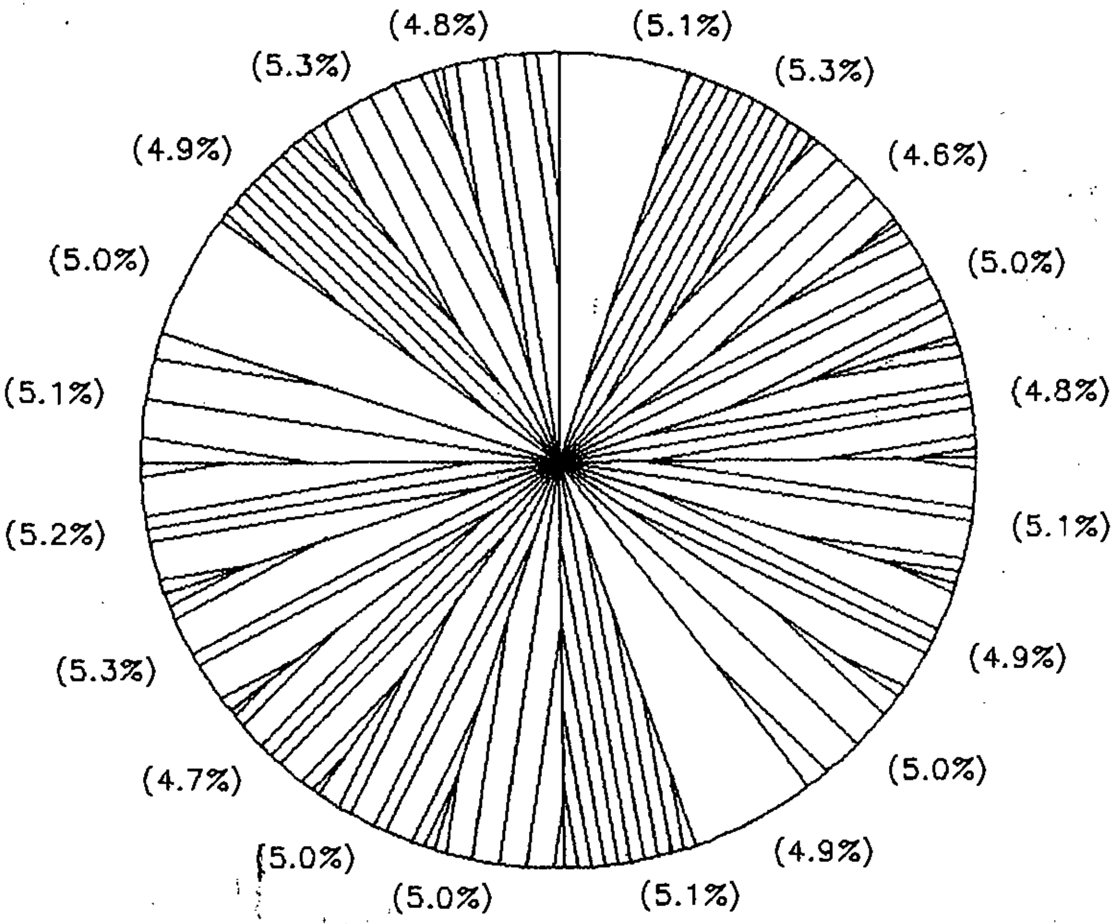


Fig 5·6

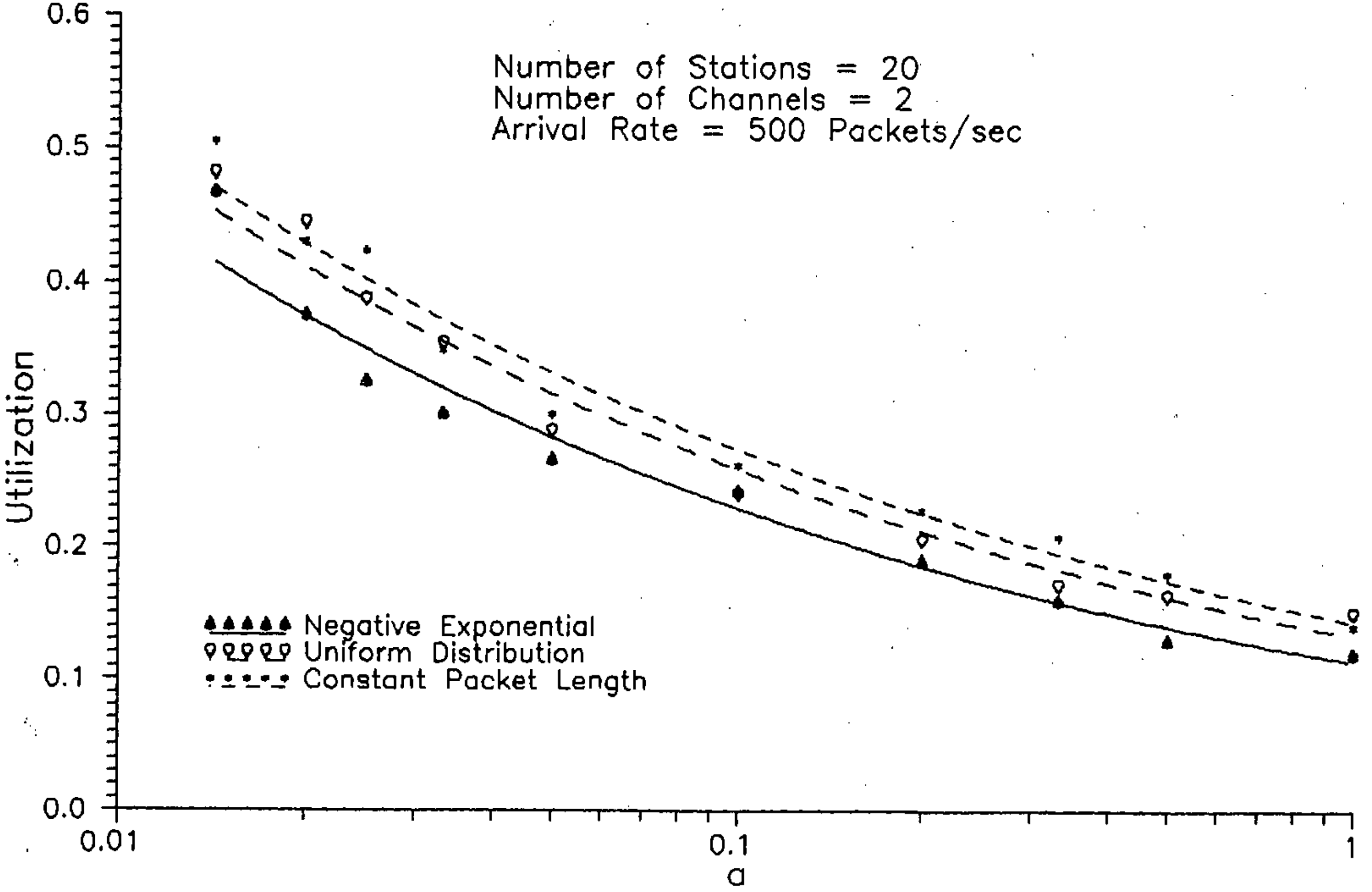


Fig 5.7 : Utilization vs 'a' for Different Packet Distribution

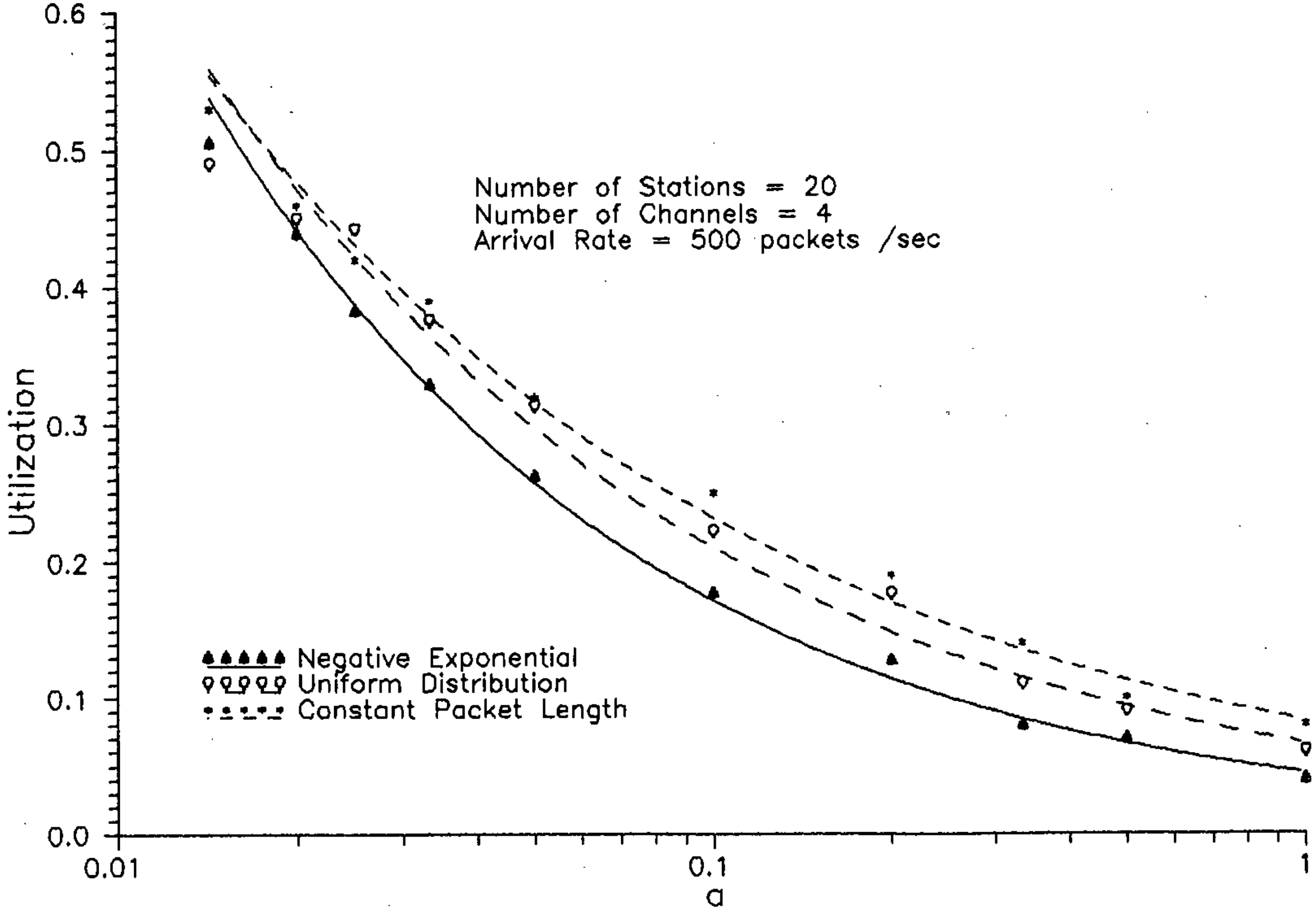


Fig 5.8 : Utilization vs 'a' for Different Packet Distribution

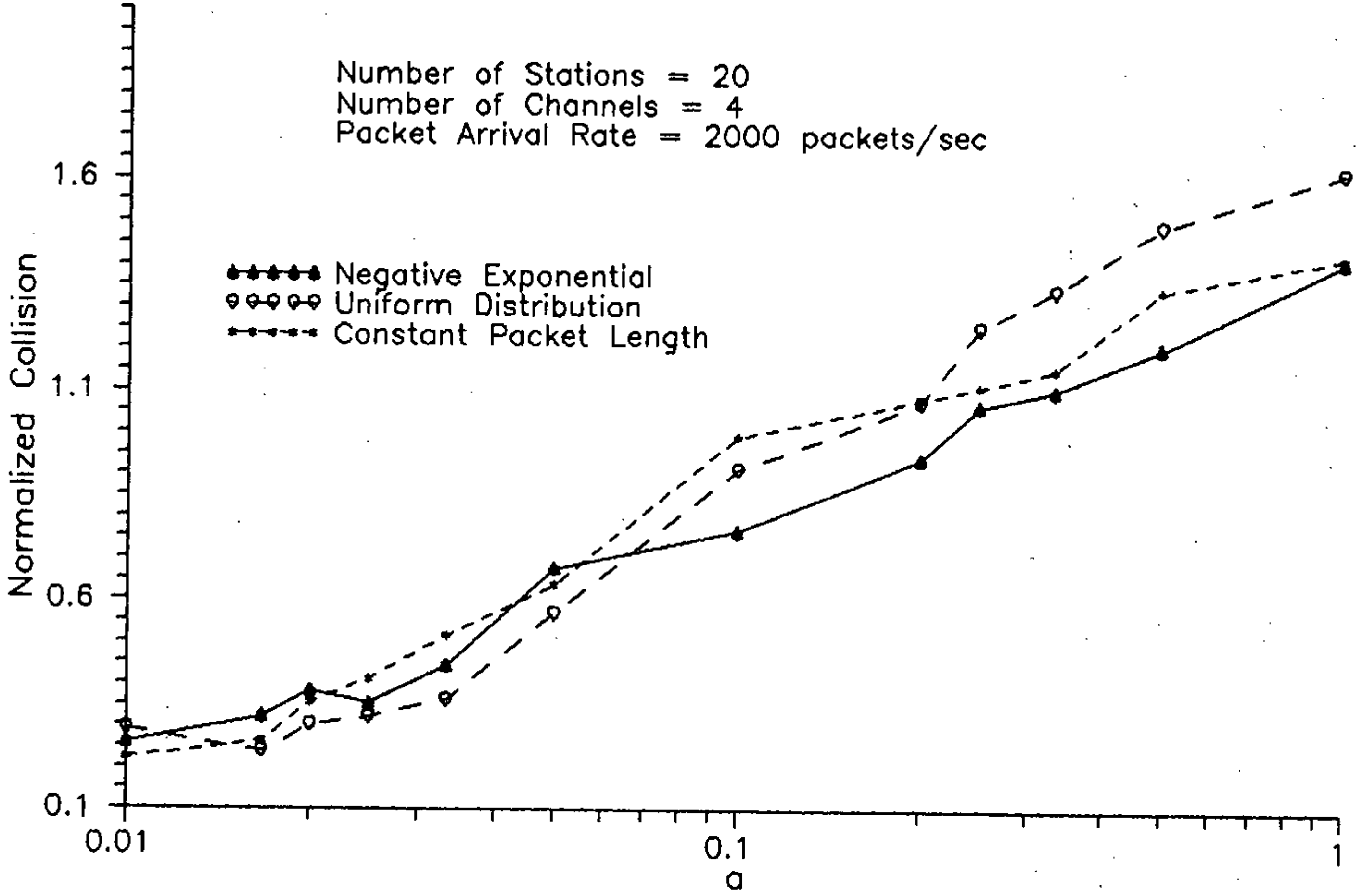


Fig 5.9 : Normalized Number of Collisions vs 'a'

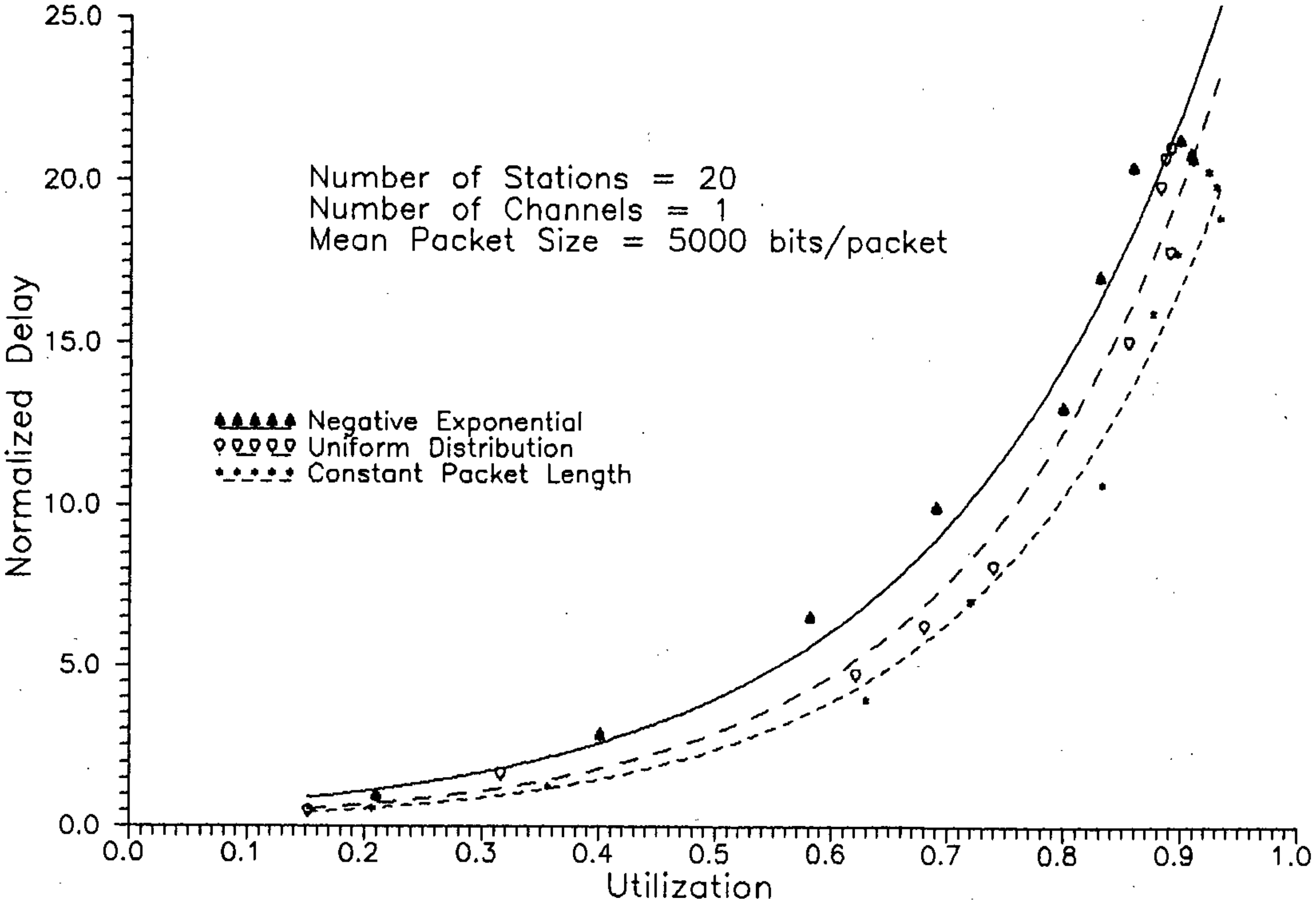


Fig 5.10 : Delay vs Utilization for Different Packet Distribution

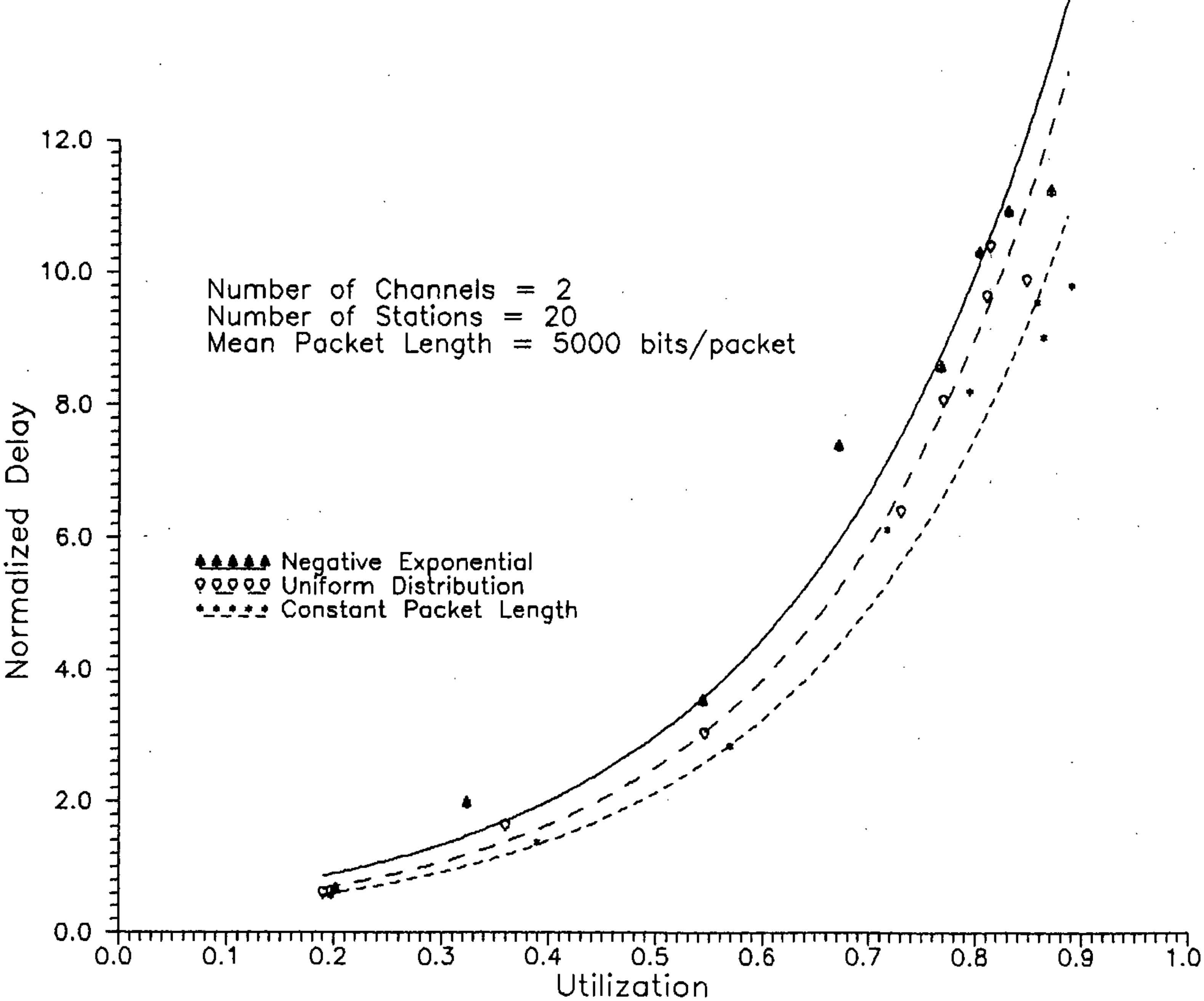


Fig 5.11 : Delay vs Utilization for Different Packet Distribution

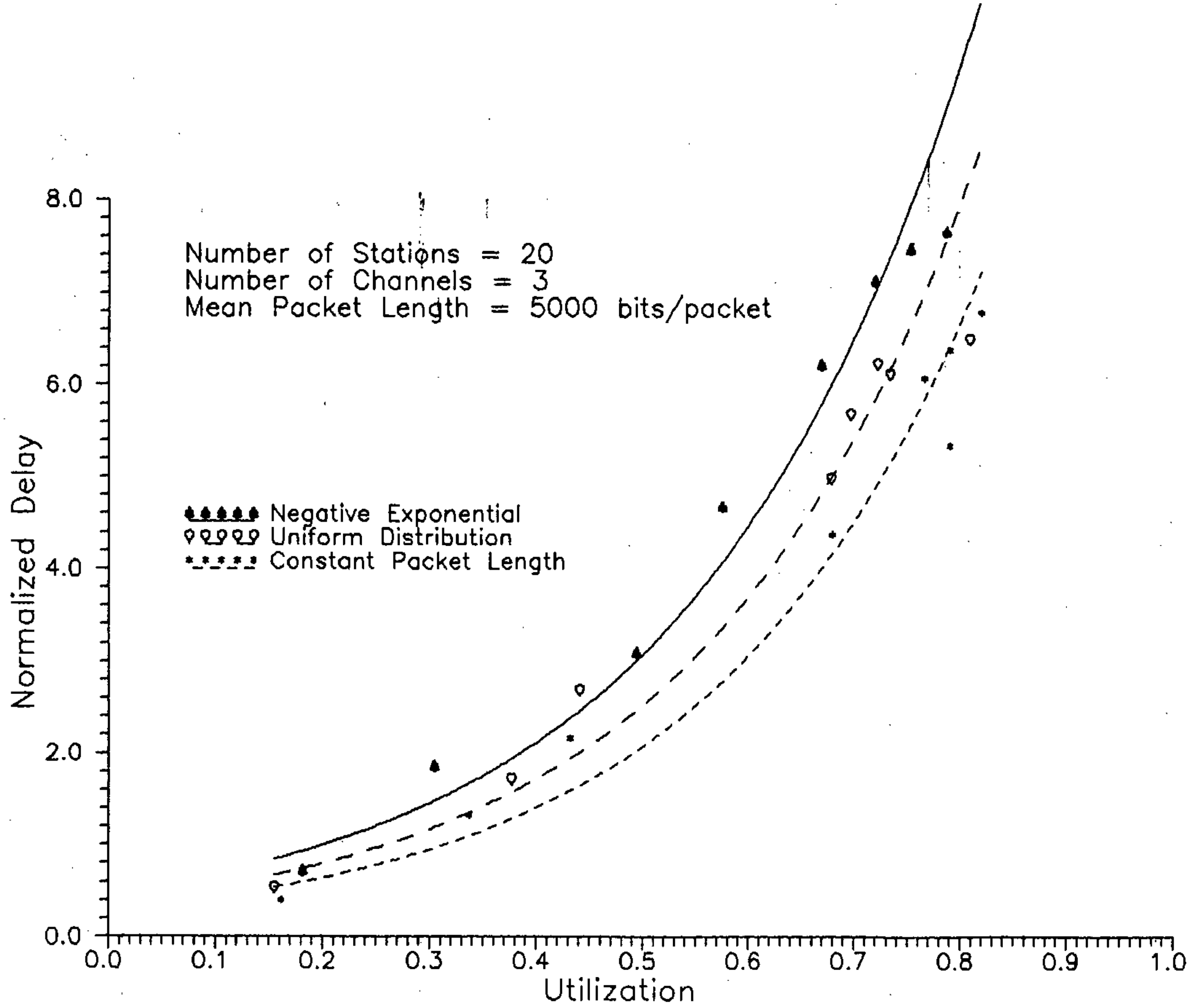


Fig 5.12 : Delay vs Utilization for Different Packet Distribution

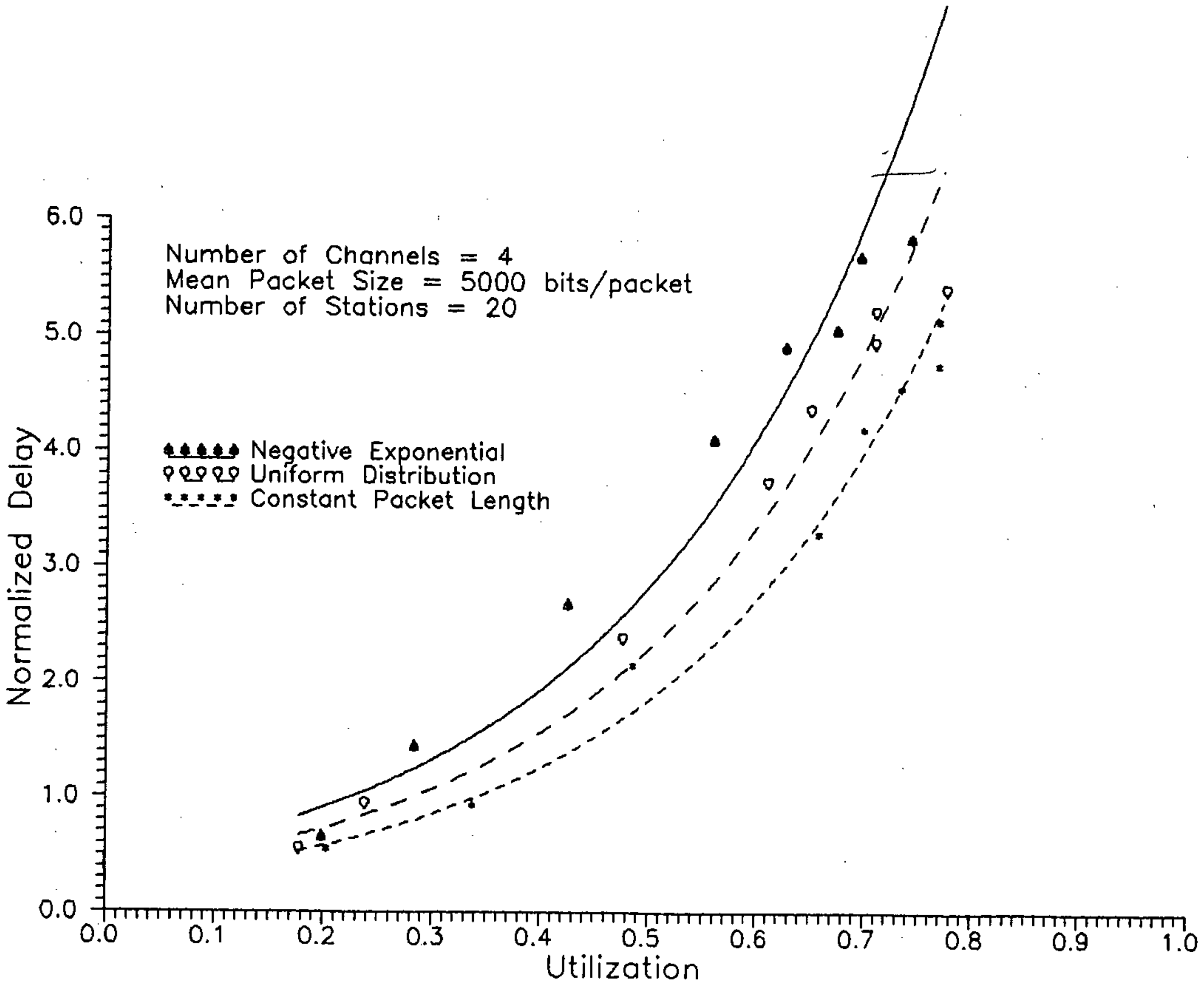


Fig 5.13 : Delay vs Utilization for Different Packet Distribution

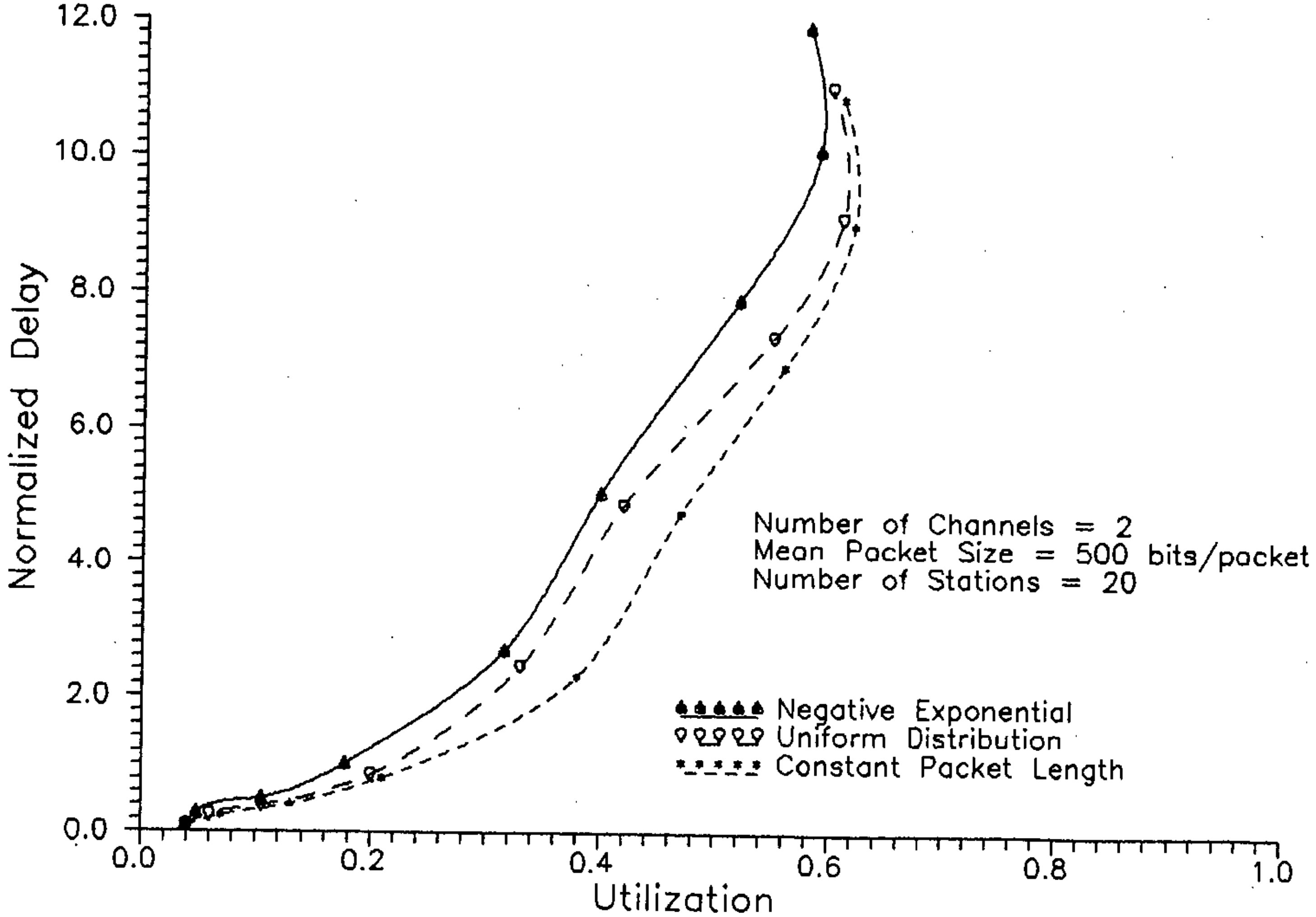


Fig 5.14 : Delay vs Utilization for Different Packet Distribution

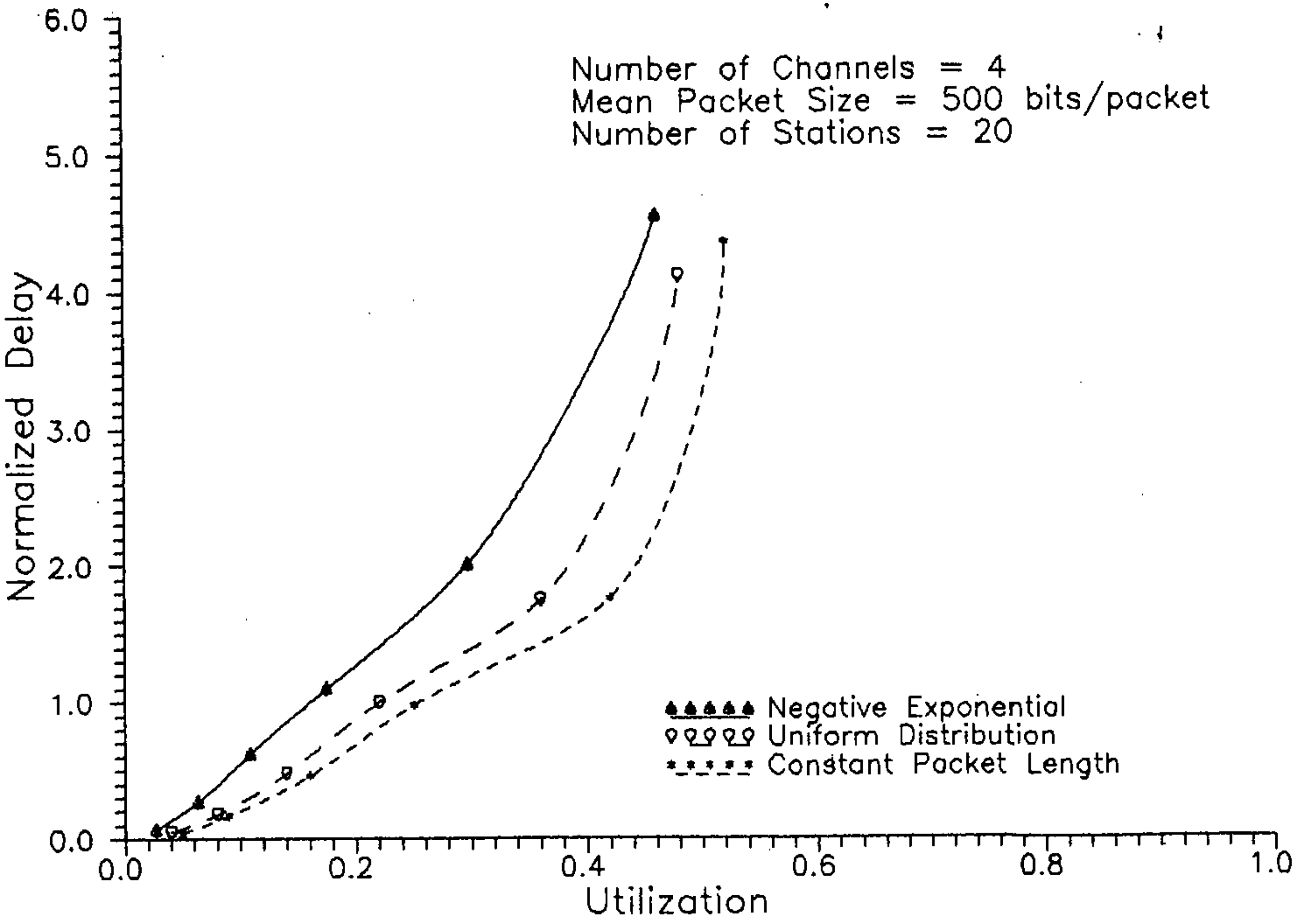


Fig 5.15 : Delay vs Utilization for Different Packet Distribution

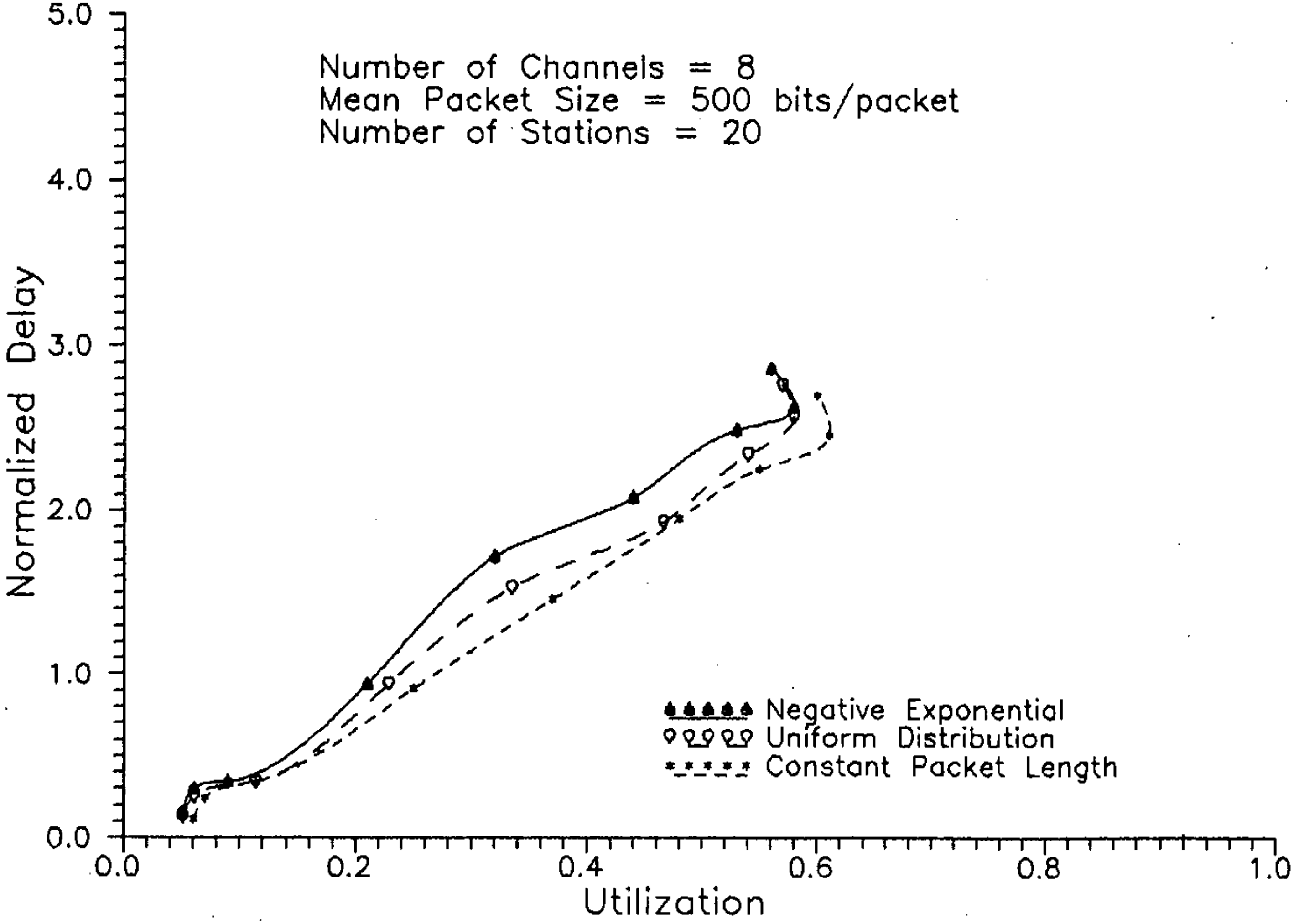


Fig 5.16 : Delay vs Utilization for Different Packet Distribution

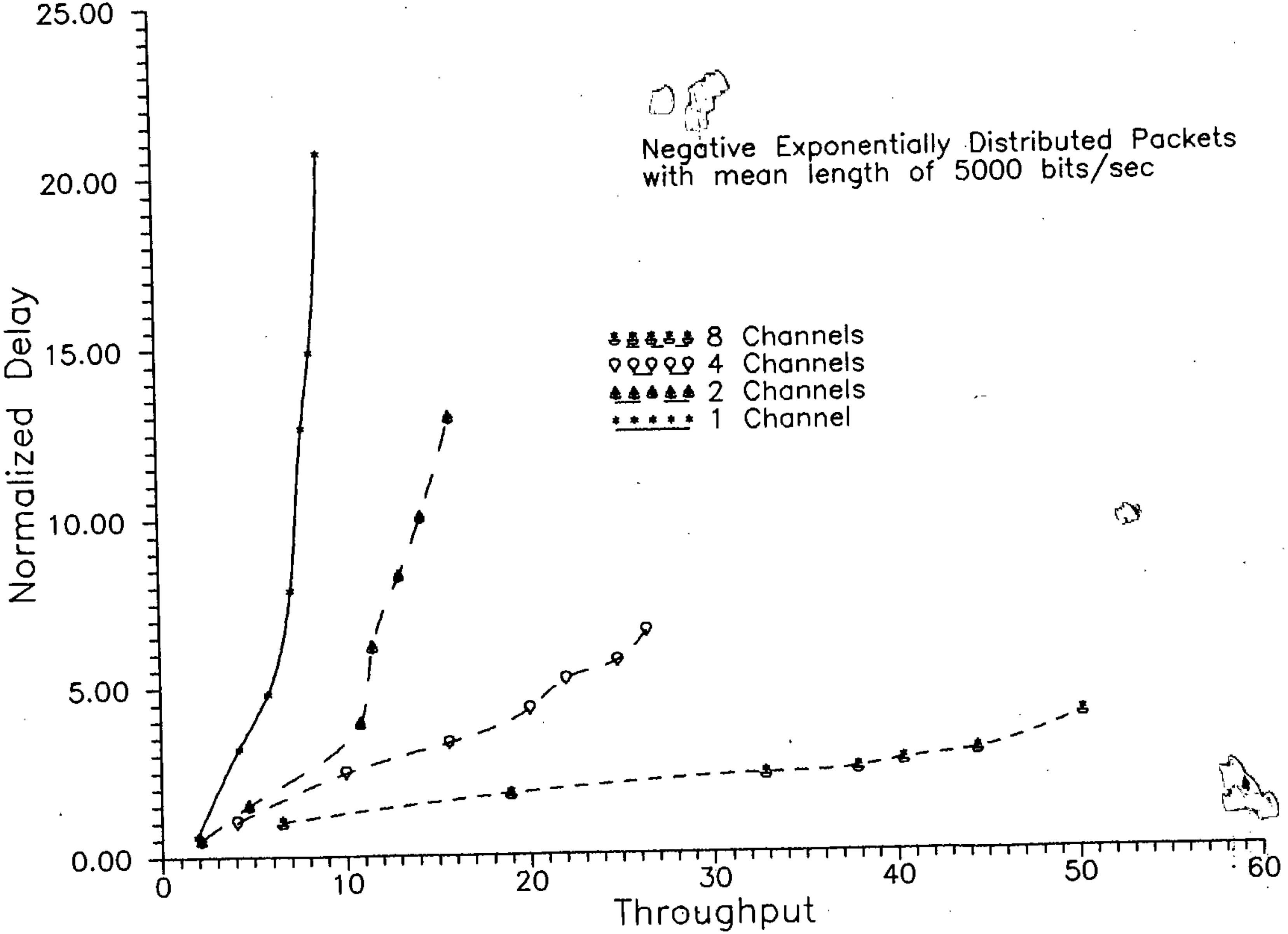


Fig 5.17 : Normalized Delay vs Throughput for Different Channel Number

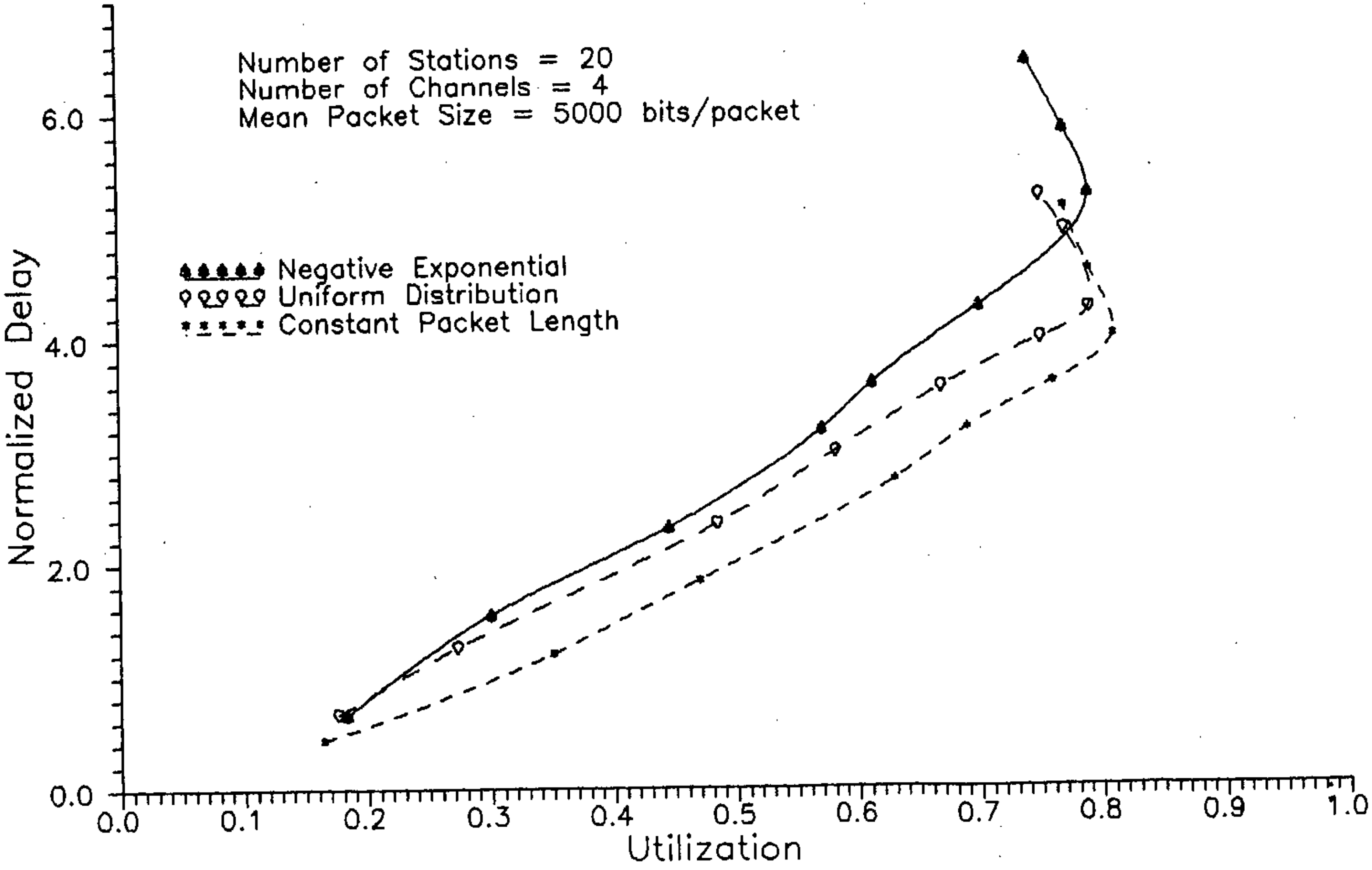


Fig 5.18: Delay vs Utilization for different Distribution

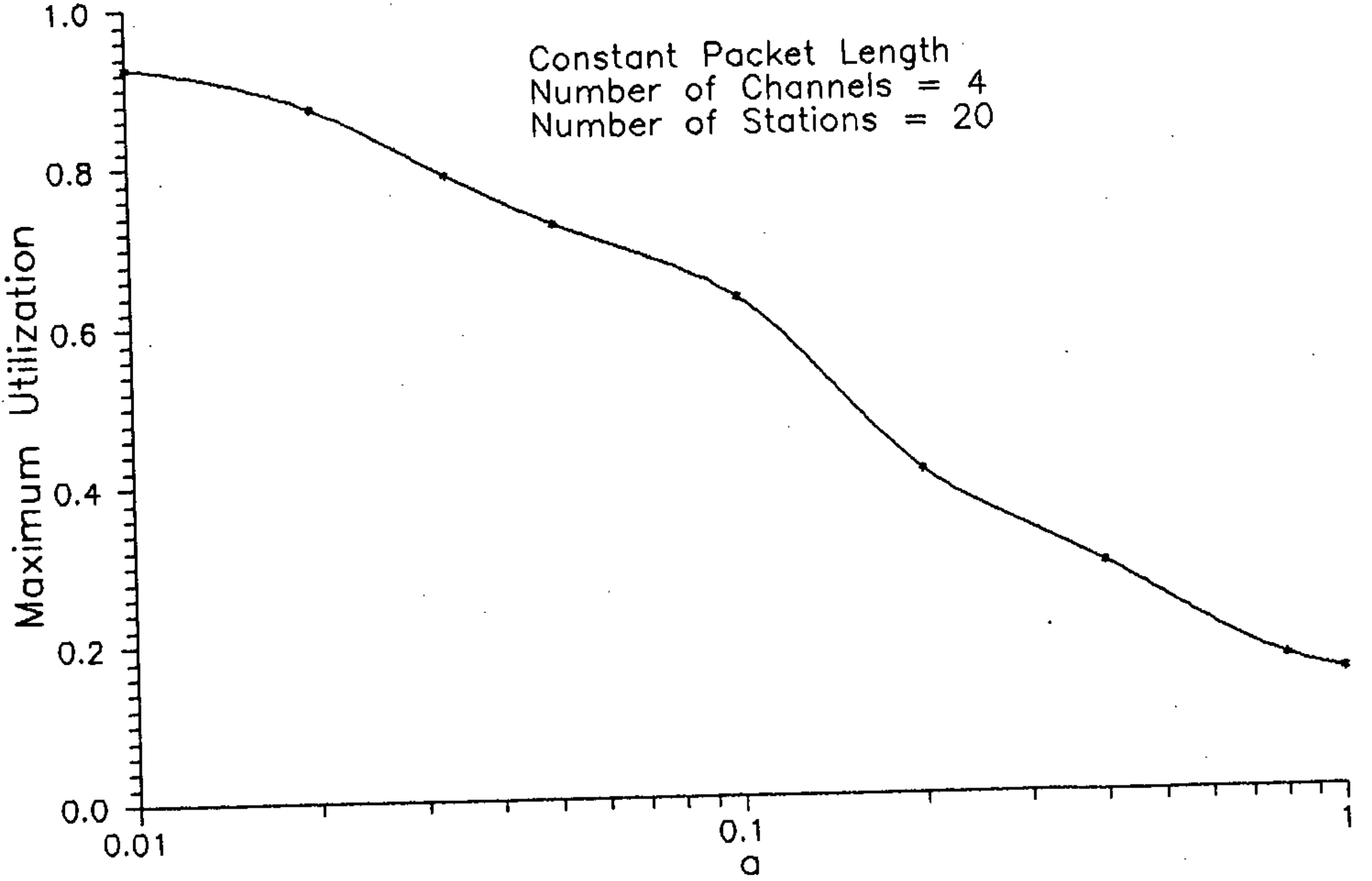


Fig 5.19 : Maximum Utilization of the system vs 'a'

Chapter 6

Conclusions

6.1 Conclusions :

A simulator has been developed which can simulate the behavior of the Unslotted Non-persistent CSMA/CD Protocol. This is intended to be used to predict the performance of this protocol under different operating conditions. The performance parameters can be delay, throughput, queue length, and Utilization. The variable parameters are Number of channels, Number of stations, packet length,packet length distribution policy, arrival rate ,arrival distribution policy,length of the LAN . The following conclusions are drawn from the simulator :

I. It was found from the simulation result that negative exponential distribution gives least utilization and highest delay among the three distribution policies described. On the other hand Constant distribution gives maximum utilization and minimum delay for the same parameter values.

II. At light load , the increase of number of channel decreases the utilization and delay for the same offered load. Increasing number of channel improves the performance at high load.

III. It was found that the increase of the size of the packet increases utilization .

IV. As the number of packet arrival rate increases , collision is found to increase. The delay is also found to increase with increasing arrival rate . It was also found that ,for 5000 bits packet size , maximum utilization was obtained at 400% offered load beyond which the delay - utilization curve deteriorates.

The delay throughput curve differs slightly from the theoretical results [25] because of two different assumptions :

I. Unslotted and Nonpersistent protocol was assumed instead of slotted persistent case.

II. Rescheduling with Negative Exponential Distribution was assumed instead of Binary Back-off policies [1].

6.2 Recommendations:

I. The simulator can be modified to accommodate slotted and persistent CSMA/CD and also other LAN Protocols. This can be done by changing the event selector module, collision module and successful transmission module.

II. The developed simulator assumes the stations to be homogeneous with same arrival rate. But the stations can be made heterogeneous with the provision of a priority queue . Variable arrival rate for different stations can also be accommodated.

III. Any Rescheduling method can be accommodated in this simulator. It can have geometric distribution, or Bernoulli Distribution.

IV. The output obtained now is in the form of statistical representation. Animation techniques can be used to the simulator for graphical lively representation.

V. For Multiple Channel LAN other probable candidates as Token Bus may also be investigated.

VI. It was found from simulation that at higher offered load Delay increases sharply and utilization decreases. Future work may be done on the improvement of performance at high load.

References

REFERENCES :

[1] Tanenbaum, A.S., "Computer Networks, " Prentice Hall Inc. , 1981.

[2] Metcalfe, R. M. and Boggs, D.R., "Ethernet Distributed packet switching for Local Computer Networks," Communications of the ACM, Vol 19,No. 7, pp 395-405.

[3] Shoch, J.F. and Hupp , J. A. , "Measured Performance of an Ethernet Local Network," Communications of the ACM, Vol 23, No. 12, pp 771-780.

[4] M. I. Nimalkumar Guy, "Simulation of Computer Network and Performance Evaluation," M.Sc. thesis ,December 1988, Asian Institute of Technology.

[5] Sventek , J. Greiman, W., O'Dell , M. and Jansen, A. " Token Ring Local Area Networks ," Computer networks 4, pp 425-459.

[6] Rogeberg, T. Tetrasim " A program system for the simulation of Telephone Networks," Computer Networks and Simulation II, North Holland Publishing Company pp 194-226.

[7] Kleinrock, L. "Analytic and Simulation methods in Computer Network Design," AFIPS Conference Proc. Vol 36, pp 569-579.

[8] Heyman, D.P., "The effects of random message sizes on the performance of the CSMA/CD Protocol," IEEE Trans. Commun.,Vol Com-34 ,June 1986,pp 547-553.

[9] Beurman, S.L., Coyle, E.J., "The Delay Characteristics of CSMA/CD Networks ," IEEE Transactions on Comm. vol 36 No. 5 May 1988, pp 553-563.

[10] Apostolopoulos, T.K and Protonotarious, E.N.,"Queuing Analysis of Buffered CSMA/CD Protocols ," IEEE Transactions on Comm. vol 34 No. 9, September 1986, pp 898-905.

[11] Takagi, H., and Murata, M., "Output Processes of Persistent CSMA and CSMA/CD systems," Computer Networks ISDN System 13, North Holland 1987, pp. 333-346.

[12] Ko, C.C., Lye, K.M., Chua, K.C. and Yap, F.T., "Analysis of a CSMA/CD based Protocol with Dynamic segmentation ," Computer Networks and ISDN Systems 18 (1989/90) 1-18.

[13] Sharrock, S.M., Ghanata, S., Maly, K.J., " A CSMA/CD-based Integrated voice/data Protocol with Dynamic Channel Allocation, " Computer Networks and ISDN Systems 18 (1989/90) 1-18.

[14] Matsumoto, M., Takahashi, Y.,and Hasegawa, T. "Probability Distributions of Interdeparture Time and Response Time in Multipacket CSMA/CD Systems," IEEE Trans. Commun.,Vol 38 No. 1,January 1990.

[15] Gonsalves, Timothy A. and Tobagi, F.A., "On the performance Effects of Station Locations and Access Protocol Parameters in Ethernet Networks," IEEE Transactions on Comm. vol 36 No. 4 April 1988, pp 441-448.

[16] Alam, M.S. and Swartwout, R. E. , "Modeling CSMA/CD-DFT in shared Multiple Bus Local Computer Network ," Confc. Proc. on Modeling and Simulation, April 23-24,1987 ,pp 1165-1171.

[17] Pritsker, A.A.B., Pedgen,C.D., "Introduction to Simulation and Slam, " John Wiley and Sons, Inc., 1979.

[18] Emshoff, J.R., Sisson, R.L., "Design and Use of Computer Simulation Models, " McMillan Publishing Co.,NY 1970.

[19] Conway, Johnson and Maxwell ,"Some Problems on Digital Computer Simulation."

[20] Alam, M.S. ,"Analysis of m-CSMA/CD in Multiple Bus Local Computer Networks ," IEB Journal, Vol 17 No. 3-4, July-October,1989, pp 17-24.

[21] Ahmed ,F., Alam,M.S. ,"Effect of Packet Length Distribution on the performance of Unslotted CSMA/CD Protocol , " Paper presented in the AIT-BUET Seminar, August,1990.

[22] Black, U.D., "Data Communications and Distributed Networks ," Prentice - Hall Inc. NJ 07632.

[23] Charles H. Sauer, "Simulation of Computer Communication system," Prentice -Hall ,Inc., Englewood Cliffs,New Jersy 07632.

[24] Tobagi F.A. and Hunt V.B.,"Performance Analysis of Carrier Sense Multiple Access with Collision Detection," Computer Networks 4(1980) 245-259.

[25] Stallings, W. ,"Local Networks - An Introduction, " McMillan Publishing Company, New York.

[26] Gordon, G. "System Simulation ," Prentice Hall of India Private Limited ,New Delhi,1980.

[27] Fishman, G.S., "Principles of Discrete Event Simulation," John Wiley and Sons, Inc. ,1978.

Appendix

```
/************************************************************/
*  Simulation of Unslotted  Nonpersistent CSMA/CD Protocol  *
/***********************************************************/

# include <stdio.h>
# include <stdlib.h>
# include <math.h>
# include <alloc.h>
# include <time.h>
# include <conio.h>

# define NOOFSTATION 20     /*Number of Stations*/
# define PD 5           /*Propagation Delay in microsec */
# define JAMSIGNAL 25           /*Length of Jam Signal = 25 Bits */
# define NC 8                  /*Number of Channel */
# define RUN 5                 /*Replication of Simulation */

        /*************************************/
        /*  Global Declaration of Variables  */
        /*************************************/

float length,aratio,ndpp,time_elapsed,mspp;
float simulationtime,coldelay,mptime,packettime;
float bpp,jamtime,pd,load,util,avcol,throughput;

int ts,sr,lambda,ps,psdf,steady,distribution;
int total_packet_transmitted;

float s_totdelay,s_avdelay,s_maxdelay,s_totavdelay;
float s_totcollision;
double ran,y;
char *dist;

        /* Station related variables */

int rs,m,r,r1,s,ns=NOOFSTATION,a=300,nc=NC;
int channel_of_station[NOOFSTATION];
int pos[NOOFSTATION],transmitted[NOOFSTATION];
int aborted[NOOFSTATION],collision[NOOFSTATION];
float start[NOOFSTATION],arrivaltime[NOOFSTATION];
float delai[NOOFSTATION],dela,mttime[NOOFSTATION];
float s_avdelai[NOOFSTATION];
float s_normdelay,s_avcollision;

FILE *outfile,*fp,*fpc,*fps,*fpsize,*fpdelay,*fpu,*fpcol;

int station[NC][NOOFSTATION];
float transmission_time[NC][NOOFSTATION];

        /* channel related variables */

int msgout[NC],msgin[NC],chcol[NC],sn[NC];
float utilization[NC],total_collisiontime[NC],totdelay;
```

```
float tput[NC],maxdelay[NC],msgtime[NC],avdelay[NC];
float netavdelay[NC],totavdelay[NC] ,normdelay[NC];

float elapsedtime[NC],idletime[NC],avcollision[NC];
float tot_eventtime[NC];
float delay_per_packet,avqlength[NC],qlength[NC];
float chdelay[NC];

float   avutilization[NC],avchdelay[NC],,avmsgtime[NC],
    avqlength[NC],avtput[NC],avidletime[NC],
    avelapsedtime[NC],avtotal_collisiontime[NC];

int avtransmitted[NOOFSTATION];
float   avaborted[NOOFSTATION],avcoll[NOOFSTATION],
    avs_avdelai[NOOFSTATION],avmttime[NOOFSTATION];

int     avmsgout[NC],avchcol[NC];
float   avu,avc,avt,avd;
int     avp;

  /*
   length=Length of the Network
   aratio=Ratio of the Propagation Delay to the Packet Transmission Time
   lambda=mean arrival rate  in packets per second
   utilization[m]=utilization of the channel m
   qlength[m]=current length of the queue of channel m
   simulationtime= Simulation Time
   tput= Normalized Throughput
   msgout[m]=Total messages flown out of the channel m
   maxdelay=Maximum Delay for any of the message
   r=no. of stations in collision at one time
   r1=no. of stations waiting during the successful
   transmission of another station
   mptime = Mean Packet Transmission time
   ndpp = Normalized Delay per packet
   bpp = Bits per packet
   psdf = Packet Size Distribution factor

      */

        /********************************************/
        /* Programmer Defined Function Declarations */
        /********************************************/

        double randu();
        float ptime();
        void reversevideo(void);
        void hide(void);
        void Input_Parameters(void);
        void open_files(void);
        void print_headers();
        void initialize_avg_parameters(void);
        void initialize_time_and_channel(void);
```

```
   void arrange_time(void);
   void allocate_to_channel(void);
   void initialize_station_parameters(void);
   void initialize_system_parameters(void);
   void initialize_channel_parameters(void);
   void successful_transmission(void);
   void call_event(void);
   void collision_occured(void);
   void successful_transmission(void);
   void station_output_statistics(void);
   void channel_output_statistics(void);
   void system_output_statistics(void);
   void simulate(void);
   void find_maximumtime(void);
   void printout(void);

   /*********************************/
   /* Functions for screen Dispaly */
   /********************************/

   void reversevideo(void)
   {
   textattr(0x70);
   cprintf("       ");
   gotoxy(wherex()-7,wherey());
   }

   void hide(void)
   {
   textcolor(BLACK);
   textbackground(BLACK);
   }

  /**************************************************/
  /* Function for handling the Input_Parameters     */
  /**************************************************/

void Input_Parameters(void)

{
clrscr();

printf("Give the Time of Simulation (in Seconds): ");
reversevideo();
scanf("%f",&simulationtime);
simulationtime*=1000000;
normvideo();
printf("\n");

printf("\nLength of the Network (in Kilometer)?  ");
reversevideo();
scanf("%f",&length);
normvideo();
```

```
 printf("\n");

/* printf("\nNo. of Stations in the system ? ");
 scanf("%d",&ns);*/

 /* printf("\nNo. of Channels in the system ? ");
 scanf("%d",&nc);*/

 printf("\n Mean Packet  size  (in bits per packet) :? ");
 reversevideo();
 scanf("%d",&psdf);
 normvideo();
 printf("\n");

 printf("\nMean Arrival Rate (in Packets per Second) : ");
 reversevideo();
 scanf("%d",&lambda);
 normvideo();
 printf("\n");

 printf("\nTransmission speed (in Mb per Second) : ");
     reversevideo();
 scanf("%d",&ts);
 normvideo();
 printf("\n");


 }

/*******************************************/
/* Function for Random number Generation */
/*******************************************/

double randu()
{
double ma=2147483647.0,x;
float a1=16807.0;
int table[8],i3;

for (i3=0;i3<=7;i3++)
    {
    x=a1*rs;
    rs=(int)fmod(x,ma);
    table[i3]=rs;
    }

    table[rs%8]=rs;
    return((double)table[rs%8]/ma);
 }

/********************************/
/* Function for file handling */
/********************************/
```

```
 void open_files(void)
 {

 if ((outfile=fopen("c:\\out\\csm.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fp=fopen("c:\\out\\dist.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fps=fopen("c:\\out\\station.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fpc=fopen("c:\\out\\channel.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fpsize=fopen("c:\\out\\size.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fpdelay=fopen("c:\\out\\mdelay.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fpu=fopen("c:\\out\\mUtilize.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

if ((fpcol=fopen("c:\\out\\mColision.dat","w+"))==NULL)
{
perror("The file cannot be opened ");
exit(1);
}

}        /* End of openfiles */
```

```
/*****************************************/
/* Function for printing the file headers */
/****************************************/

void print_headers(void)
{

fprintf(outfile,"   Statistics of Unslotted CSMA/CD Protocol\n");
fprintf(outfile,"  ========================================\n");

fprintf(fpsize,"Transmission Speed=%d  Megabits per Second\n",ts);
fprintf(fpsize,"No. of Terminals=%d\n\n",ns);
fprintf(fpsize,"No of channels=%d\n",nc);
/*
fprintf(outfile,"\n\nPacket Size    Arrival   Utilization
Av.Delay \n");
fprintf(outfile,"(Bits)  (Packet/Sec)  (Normalised)
(Normalised)\n\n");
*/

fprintf(fpsize,"\n Arrival  offered load  Distribution   Utilization

Avg Delay\n");

fprintf(fpdelay,"Offered Load   Utilization   Delay   Throughput\n");


fprintf(fpu,"Utilization     aratio       Distribution\n");

fprintf(fpcol,"Av Collision    aratio    Distribution\n");

}



void initialize_avg_parameters(void)
{
int m1;
avu=avt=avd=avc=0.0;
avp=0;

    for (m1=0;m1<=nc-1;m1++)
{
avmsgout[m1]=0;
avutilization[m1]=0;
avchdelay[m1]=0;
avchcol[m1]=0;
avqlength[m1]=0;
avtput[m1]=0;
avidletime[m1]=0;
avmsgtime[m1]=0;
avelapsedtime[m1]=0;
avtotal_collisiontime[m1]=0;
}
```

```
    avtransmitted[m1]=0;
    avaborted[m1]=0;
    avcoll[m1]=0;
    avs_avdelai[m1]=0;
    avmttime[m1]=0;
    }

    }

      /**************************************************/
      /* Generation of Arrival time and Choice of channel*/
      /*   for the stations                             */
      /**************************************************/

    void initialize_time_and_channel(void)
    {
    int i;
    randomize();
    rs=random(20000)+1;

/*  printf("\nInitializing\n");  */

    for (i=0;i<=ns-1;i++)
    {
    arrivaltime[i]=-(float)log(randu())*mspp ;
    channel_of_station[i]=floor(randu()*nc);
    }
    }

    void arrange_time(void)
    {
    int i,n,k,j;
    float small;

    for (i=0;i<=ns-1;i++)
    {
    start[i]=arrivaltime[i];

    }

        for (i=0;i<=ns-1;i++)
        {
        pos[i]=i;
        small=start[i];

        for (j=i+1;j<=ns-1;j++)
        {
        if(small > start[j])
        {
        small=start[j];
        pos[i]=j;
        }
        /* end of j loop */
```

```
        }

        start[pos[i]]=start[i];
        start[i]=small;

        for(n=0;n<=i-1;n++)

        for (k=0;k<i;k++)
        {
        if(pos[i]==pos[k])

        pos[i]=k;
        else
        ;
        }
        }
        }   /* end of arrange_time */

/*******************************************/
/* Function for allocating the channels   */
/* among the stations                     */
/*******************************************/

void allocate_to_channel(void)
{
int sl,cn,j,i;
for(i=0;i<=nc-1;i++)
sn[i]=0;

        for (i=0;i<=ns-1;i++)
        {
        j=pos[i];

        cn=channel_of_station[j];

        sl=sn[cn];

        transmission_time[cn][sl]=start[i];

        sn[cn]+=1;

        station[cn][sl]=j;
        }
 }

  /******************************************/
  /* The following function                */
  /* initializes the station parameters    */
  /******************************************/

void initialize_station_parameters(void)
{
int i;
```

```
for (i=0;i<=ns-1;i++)
{
    aborted[i]=0;   collision[i]=0;

    delai[i]=0;     mttime[i]=0;

    transmitted[i]=0;
}
}

/********************************/
/* Function for initializing the */
/* overall system parameters     */
/********************************/

void initialize_system_parameters(void)
{
 util=0;
 avcol=0;
 throughput=0;
 time_elapsed=0;
 total_packet_transmitted=0;

 }

/*************************************/
/* The following function           */
/*initializes the channel parameters */
/*************************************/

void initialize_channel_parameters(void)
{
int i;

for (i=0;i<=nc-1;i++)
{
    msgout[i]=0;
    total_collisiontime[i]=0;
    elapsedtime[i]=0;
    idletime[i]=0;
    msgtime[i]=0;
    tot_eventtime[i]=0;
    tput[i]=0;
    chcol[i]=0;
    chdelay[i]=0;
    qlength[i]=0;

}
}

 /*******************************************/
 /* Size of the packet for the transmitting
```

```
 station is calculated  as follows */
 /*****************************************/

float ptime()
{
switch(distribution)

{
    case 1:             /* Negative Exponential */

        {
        bpp=-log(randu())*psdf ;
        if(bpp<=a)
        bpp+=(float)a;
        dist="Negative Exponential Distribution";
        break;
        }

     case 2:                   /* Uniform */

         {
         bpp=(float)(2*psdf-a)*randu();
            if(bpp<=a)
         bpp+=(float)a;
         dist="Uniform Distribution";
         break;
         }

     case 3:                  /* Constant */

         {
         bpp=psdf;
         dist="Constant Packet Distribution";
         break;
         }

 }
    return((float)(bpp/ts));
 }

 /**************************************************/
     /* The following function calculates the values of
   event determining parameters */
/**************************************************/


 void call_event(void)
     {
     float diftime;
     int i;
     packettime=ptime();
    /* printf("packettime=%f\n",packettime);getch(); */
```

```
     for (i=0;i<=s-1;i++)
     {

/*   if(qlength[i]<1)
      */

     if(transmission_time[m][i]<=packettime )
     qlength[m]+=1;

     diftime=transmission_time[m][i]-transmission_time[m][0];

     if(diftime<=pd)
     r+=1;

     else
     if(diftime>pd && diftime <= packettime)
     r1+=1;

     }
     }

 /**************************************************/
 /* Function for updating the collision statistics */
 /**************************************************/

   void collision_occured(void)

 /* This function updates the system statistics for the
    occurence of a collision  */

    {
     float collisiontime,inittime,prevtime,dtime;
     int num,j,i;
     float delayunit;

     inittime=transmission_time[m][0];

     /*
 fprintf(fp,"Collision among %d stations in channel %d\n",r,m);

 fprintf(fp,"Colliding stations are rescheduling their attempts\n");
     */
     collisiontime=transmission_time[m][1]-inittime+pd;

     chcol[m]+=1;

     elapsedtime[m]+=collisiontime;

     total_collisiontime[m]+=collisiontime;

    for(i=0;i<r;i++)
    {
```

```
    j=station[m][i];
    collision[j]+=1;
    /*
fprintf(fp,"Total collision of station %3d =%3d\n",j,collision[j]);

      */
    /* if no. of collisions is less than 10 then using
     binary exponential backoff algorithm the next arrival
     time is calculated */


    /* The above is the delay for collision only */

    num=(int)pow((double)2,(double)(collision[j]-1));
    randomize();

    prevtime=arrivaltime[j];
    arrivaltime[j]=-(float)log(randu())*10*pd;
    channel_of_station[j]=floor(randu()*nc);
    delayunit=transmission_time[m][1]-prevtime+pd+arrivaltime[j];

    delai[j]+=delayunit;
    chdelay[m]+=delayunit;




    }               /* end of for */

 /* Update the time for noncolliding stations */

 for(i=r;i<=s-1;i++)
 {
 j=station[m][i];
 dtime=transmission_time[m][i]-inittime;

 if(dtime>=collisiontime)
 arrivaltime[j]-=collisiontime+inittime;

 else
{
      num=(int)pow((double)2,(double)(collision[j]-1));
  prevtime=arrivaltime[j];

  arrivaltime[j]=-(float)log(randu())*10*pd;
  collision[j]+=1;

  delayunit=inittime+collisiontime-prevtime+arrivaltime[j];
  delai[j]+=delayunit;
  chdelay[m]+=delayunit;

 }
 }
```

```
}          /* end of collision block */


/********************************************************/
/* Function for updating the statistics related to the  */
/*     Successful Transmission                      */
/*******************************************************/

 void successful_transmission(void)

   {
    int z,j,i;
    float previoustime,reduction;
    float delayunit;
/*
After a successful transmission, time elapsed will be
incremented by the packettime. During this transmission
some stations trying to transmit will find the channel busy
They will reschedule their attempts.
 */

   z=station[m][0];
   previoustime=transmission_time[m][0] ;
   reduction=packettime+previoustime;
   /*
   fprintf(fp,"Station %d transmits successfully\n",z);
   */
   msgtime[m]+=packettime;
   tput[m]+=bpp;
   mttime[z]+=packettime;

   elapsedtime[m]+=packettime;

   if (qlength[m]>=1)
   qlength[m]-=1;
   transmitted[z]+=1;

   msgout[m]+=1;
   total_packet_transmitted+=1;
   /*
   fprintf(fp,"%d packets transmitted so far through
   the channel %d\n",msgout[m],m);
   */

    if(r1> 0)
    {
    /*
    fprintf(fp,"%d stations try to transmit when one station\n",r1)

    fprintf(fp,"has already started transmission\n");
   */

    for(i=1;i<=r1;i++)
   {
```

```
j=station[m][i];

 /* Waiting time for the stations are calculated .
 At this phase the waiting time is simply the difference
 between the arrival of a packet and the completion of the
 present packet */

 delayunit=reduction-arrivaltime[j];
 delai[j]+=delayunit;
 chdelay[m]+=delayunit;

 /* New Arrival time for r1 stations  are generated.
 Obviously this time is measured from the instant
 of the completion of transmission of the present
 station  */

arrivaltime[j]=-log(randu())*2*packettime;
    channel_of_station[j]=floor(randu()*nc);
delai[j]+=arrivaltime[j];
chdelay[m]+=arrivaltime[j];

 }              /* end for(i=1;i<=r1;i++) */


 /* new arrival time for the present transmitting station */


 arrivaltime[z]=-(float)log(randu())*mspp;
 channel_of_station[z]=floor(randu()*nc);

 /* updation of arrival time for the rest */

 for(i=r1+1;i<=s-1;i++)
{
j=station[m][i];
arrivaltime[j]-=reduction ;}

 }                    /* end if(r1>0) */

     /* if some stations are neither within collision time
nor within transmission time of a transmitting station
then no station faces any delay . Update the arrival time
of all other stations   */

 if(r<=1 && r1<1)          /* Successful transmission with
             no waiting station */

   {

   for(i=1;i<=s-1;i++)
    {
     j=station[m][i];
     arrivaltime[j]-=reduction;
```

```
            }

          arrivaltime[z]=-(float)log(randu())*mspp;
          channel_of_station[z]=floor(randu()*nc);
        }


        }

    /***************************************************/
    /* The following function does the simulation task */
    /***************************************************/

    void simulate(void)

    {
    int j;

    arrange_time();

    allocate_to_channel();

    for (m=0;m<=nc-1;m++)        /* Loop for the channels
                     starts here */
    {
    s=sn[m];

    if (s>0)
    {

    r=0;
    r1=0;

    /*
     fprintf(fp,"\nAllocation to the channel %d\n",m);
     fprintf(fp,"================================\n");
    */
    /* Idletime is the time when the channel remains free */

     idletime[m]+=transmission_time[m][0];

     elapsedtime[m]+=transmission_time[m][0];

     call_event();

/*  fprintf(fp,"\n    Time         Station      r       r1    \n");
     for(j=0;j<=s-1;j++)
    {
    fprintf(fp,"%10.2f %12d %7d %7d \n",transmission_time[m][j],
    station[m][j],r,r1);
    }
    */
```

```
if(r>1)                           /* then collision */
       collision_occured();


 else                 /* successful transmission */
    successful_transmission();

    /*
 printf("Elapsed time of channel %d =%10.0f\n",m,elapsedtime[m]);

printf("idle Time of channel %d = %10.0f\n",m,idletime[m]);
getch();  */

/*
fprintf(fp,"Elapsed time of channel %d =%10.0f\n",m,elapsedtime[m]);

fprintf(fp,"idle Time of channel %d = %10.0f\n",m,idletime[m]);
*/

}               /* end if s>0 */

/*getch();*/


}               /* end for m=0 to nc-1 */

}                              /* end simulate */

/*************************************************/
/* Function for finding the maximum elapsed time */
/*************************************************/

void find_maximumtime(void)
{
int ch;
float temp;
time_elapsed=elapsedtime[0];
for(ch=1;ch<=nc-1;ch++)
{
temp=elapsedtime[ch];
if(time_elapsed < temp)
    time_elapsed=temp;

}
}                       /* end find_minimumtime()   */

    /**********************************/
    /* Listing of station parameters */
    /*********************************/

void station_output_statistics(void)
{
```

```
     int i1,i;
     /*
     fprintf(fps,"                         *********%s**********\n",dist);
     fprintf(fps,"\nArrival Rate=%d packets per second\n",lambda);
     fprintf(fps,"Mean Packet Size=%d bits per packet\n",psdf);

     fprintf(fps,"Station    Message     Message    No. of
     Delay   Transmission\n");
     fprintf(fps,"  No.  Transmitted    Aborted   Collisions
     (microsecs)  Time\n");    */
     s_totdelay=0;
     s_avdelay=0;
     s_maxdelay=0;
     s_totavdelay=0;
     s_totcollision=0;

         for(i1=0;i1<=ns-1;i1++)
     {
     if(s_maxdelay<delai[i1])
     s_maxdelay=delai[i1];
     s_totdelay+=delai[i1];
     s_totcollision+=collision[i1];

     if(transmitted[i1] !=0)
     {
     s_avdelai[i1]=(float)delai[i1]/(transmitted[i1]*mptime);
     s_totavdelay+=s_avdelai[i1];
     }

     else
     {
     s_avdelai[i1]=delai[i1]/mptime;
     s_totavdelay+=s_avdelai[i1];
     }
     }

    for(i1=0;i1<=ns-1;i1++)
    {

    avtransmitted[i1]+=transmitted[i1];
    avaborted[i1]+=aborted[i1];
    avcoll[i1]+=collision[i1];
    avs_avdelai[i1]+=s_avdelai[i1];
    avmttime[i1]+=mttime[i1];

    }

    s_avdelay=s_totdelay/ns;
    s_totavdelay/=ns;

    if (total_packet_transmitted !=0)

    delay_per_packet=(float)s_totdelay/total_packet_transmitted;
```

```
else

delay_per_packet=s_totdelay;

ndpp=(float)delay_per_packet/mptime;
/* ndpp= Normalized Delay per packet */

s_normdelay=(float)s_totavdelay/mptime;

if(total_packet_transmitted !=0)

s_avcollision=(float)s_totcollision/total_packet_transmitted;

else

s_avcollision=s_totcollision;

/*printf("Avg Collision =%12.2f\n",avcollision);*/
/*getch(); */
  /*
for(i=0;i<=ns-1;i++)

fprintf(fps,"%4d %10d %10d %10d %12.2f %11.0f\n",i,
transmitted[i],aborted[i],collision[i],s_avdelai[i],mttime[i]);
    */
}



/*******************************************/
/* Listing of channel related parameters */
/******************************************/

void channel_output_statistics(void)
{
int j1;

for(j1=0;j1<=nc-1;j1++)
{
if(msgout[j1] !=0)
{
chdelay[j1]=(float)chdelay[j1]/(msgout[j1]*mptime);

chcol[j1]=(float)chcol[j1]/msgout[j1];

qlength[j1]=(float)qlength[j1]/msgout[j1];
}

if(elapsedtime[j1] !=0)
utilization[j1]=(float)(msgtime[j1])/elapsedtime[j1];

tot_eventtime[j1]=msgtime[j1]+idletime[j1]+total_collisiontime[j1];
```

```
    if(tot_eventtime[j1] !=0)
    tput[j1]=(float)tput[j1]/tot_eventtime[j1];
            /*
    fprintf(fpc,"%4d %12d %12.2f %12.2f %12.2f %12.2f\n",j1,msgout[j1],
    utilization[j1],chdelay[j1],qlength[j1],tput[j1]);
    */
    }

    /*
    fprintf(fpc,"Channel   Idle    Transmission   Collision   Elapsed\n"

    fprintf(fpc,"Number     Time        Time         Time     Time \n");

    for(j1=0;j1<=nc-1;j1++)
    {
    fprintf(fpc,"%4d %12.0f %12.0f %12.0f %14.0f\n",j1,idletime[j1],
    msgtime[j1],total_collisiontime[j1],elapsedtime[j1]);
    }
      */

    }
    /* end  channel_output_statistics(void) */

    /**************************************************/
    /* Function for overall system output statistics */
    /*************************************************/

    void system_output_statistics(void)
    {
    int c;

    for (c=0;c<=nc-1;c++)
    {
    avcol+=chcol[c];

    util+=utilization[c];

    throughput+=tput[c];
    }

/*  avcol=(float)avcol/nc;*/

    util=(float)util/nc;

    throughput/=nc;

/*  fprintf(fpdelay,"%10.2f %13.4f %10.2f %10.2f\n",load,
    util,ndpp,throughput);*/

    /*
    fprintf(fpu,"%12.4f %12.5f %10d\n",util,aratio,distribution);
    fprintf(fpcol,"%12.4f %12.5f %10d\n",avcol,aratio,distribution);
```

```
     fprintf(fps,"Throughput of the system=%10.4f  Mbps\n",throughput);
     fprintf(fps,"Utilization of the system=%10.4f\n",util);
     fprintf(fps,"Total Processing   Time=%10.0f  microseconds\n",
     tot_eventtime);

     fprintf(fps,"Total no. of messages transmitted=%d\n",
     total_packet_transmitted);

     fprintf(fps,"Average Delay per message transmitted =%f\n",s_totavdel);


     fprintf(fps,"Average Collision per message transmitted =%f\n",avcol);

     */
/*   fprintf(outfile,"%9d     %11d  %14.4f %14.2f \n",psdf,lambda,
     util,ndpp);

     fprintf(fpsize," %7d %10.2f %10d %14.4f %12.2f \n",lambda,
     load,distribution,util,ndpp);
     */
     }          /* end system_output_statistics(void) */

        void printout(void)
     {
     int i,j,j1;
     /*
     fprintf(fps,"                 *********%s**********\n",dist);
     fprintf(fps,"\nArrival Rate=%d packets per second\n",lambda);
     fprintf(fps,"Mean Packet Size=%d bits per packet\n",psdf);

     fprintf(fps,"Station   Message    Message   No. of
     Delay    Transmission\n");
     fprintf(fps,"  No.    Transmitted   Aborted  Collisions
     (microsecs)  Time\n");
     for(i=0;i<=ns-1;i++)
     fprintf(fps,"%4d %10d %10.2f %10.2f %12.2f %11.0f\n",i,
     avtransmitted[i]/RUN,avaborted[i]/RUN,avcoll[i]/RUN,
     avs_avdelai[i]/RUN,avmttime[i]/RUN);*/
     /*

     fprintf(fpc,"                 *********%s**********\n",dist);

     fprintf(fpc,"\nArrival Rate=%d packets per second\n",lambda);

     fprintf(fpc,"Mean Packet Size=%d bits per packet\n\n",psdf);

     fprintf(fpc,"Channel   Packet     Channel    Normalized
     Queue   Channel\n");
     fprintf(fpc,"Number   Transmitted   Utilization   Delay
     Length  Throughput\n");*/

     for(i=0;i<=nc-1;i++)
     {
     avmsgout[i]/=RUN;
     avutilization[i]/=RUN;
```

```
    avchdelay[i]/=RUN;
    avchcol[i]/=RUN;
    avqlength[i]/=RUN;
    avtput[i]/=RUN;
    avidletime[i]/=RUN;
    avmsgtime[i]/=RUN;
    avelapsedtime[i]/=RUN;
    avtotal_collisiontime[i]/=RUN;
    /*
    fprintf(fpc,"%4d %12d %12.2f %12.2f %12.2f %12.2f\n",i,avmsgout[i],
    avutilization[i],avchdelay[i],avqlength[i],avtput[i]);
    */
    }

    /*

    fprintf(fpc,"Channel   Idle   Transmission    Collision   Elapsed\n");

    fprintf(fpc,"Number    Time      Time            Time           Time \n");


    for(j1=0;j1<=nc-1;j1++)
    {
    fprintf(fpc,"%4d %12.0f %12.0f %12.0f %14.0f\n",j1,avidletime[j1],
    avmsgtime[j1],avtotal_collisiontime[j1],avelapsedtime[j1]);
    }
      */
    avu=(float)avu/RUN;
    avd=(float)avd/RUN;
    avt=(float)avt/RUN;
    avc=(float)avc/RUN;
     fprintf(fpu,"%12.4f %12.5f %10d\n",avu,aratio,distribution);

    fprintf(fpcol,"%12.2f %12.5f %10d\n",avc,aratio,distribution);
    /*

    fprintf(fps,"Throughput of the system=%10.2f  Mbps\n",avt);

    fprintf(fps,"Utilization of the system=%10.4f\n",avu);*/

/*  fprintf(fps,"Total Processing   Time=%10.0f  microseconds\n",
    tot_eventtime);

    fprintf(fps,"Total no. of messages transmitted=%d\n",avp);

    fprintf(fps,"Average Delay per message transmitted =%10.2f\n",avd);

    fprintf(fps,"Average Collision per message transmitted =%f\n",avc);

    fprintf(outfile,"%9d     %11d  %14.4f %14.2f \n",psdf,lambda,
    util,ndpp);
    */
    fprintf(fpsize," %7d %10.2f %10d %14.4f %12.2f \n",lambda,
    load,distribution,avu,avd);
```

```
    }

     /****************/
     /* Main Program */
     /****************/


main()

     {
     float prcnt;
     int count=0,z,ilambda,ips;
     int m1,runno;

     open_files();
     Input_Parameters();
     print_headers();

     pd=PD*length;
     jamtime=(float)JAMSIGNAL/ts+pd;
     coldelay=jamtime+pd;


     clrscr();
     printf("\nSimulation Time=%12.2f\n",simulationtime);

     for(distribution=1;distribution<=1;distribution++)
     {

     for(ips=1;ips<=1;ips++)       /* packet size is varied here */
     {
     psdf=10*pd*10*ips;
     mptime=(float)psdf/(float)ts;   /* mean packet transmission time */
     aratio=pd/mptime;

     for (ilambda=0;ilambda<=8;ilambda++)
     {
     lambda=20*pow(2,(double)ilambda);
     mspp=(float)1000000/(lambda*nc);

     load=(float)psdf*lambda*ns/(1000000*ts);

     initialize_avg_parameters();




     for (runno=1;runno<=RUN;runno++)
```

```
    {
    initialize_time_and_channel();

    initialize_system_parameters();

    initialize_station_parameters();

    initialize_channel_parameters();
    /*
    elapsedtime  will account for the summation
    of collision time, idle time,jamming signal
    time and actual transmission time
    */

    while (time_elapsed<=simulationtime )
    {
    simulate();
    find_maximumtime();
    }


/*  printf("\n Done !!\n");getch(); */

    station_output_statistics();

    channel_output_statistics();

    system_output_statistics();

    avu+=util;
    avd+=ndpp;
    avt+=throughput;
    avc+=avcol;
    avp+=total_packet_transmitted;

        for (m1=0;m1<=nc-1;m1++)
    {
    avmsgout[m1]+=msgout[m1];
    avutilization[m1]+=utilization[m1];
    avchdelay[m1]+=chdelay[m1];
    avchcol[m1]+=chcol[m1];
    avqlength[m1]+=qlength[m1];
    avtput[m1]+=tput[m1];
    avidletime[m1]+=idletime[m1];
    avmsgtime[m1]+=msgtime[m1];
    avelapsedtime[m1]+=elapsedtime[m1];
    avtotal_collisiontime[m1]+=total_collisiontime[m1];
    }



    }
```

```
printout();
clrscr();
printf("\n RUNNING !!! Please don't  interrupt");
count+=1;
prcnt=(float)100*count/9;
printf("\n  %6.2f percent completed ",prcnt);


}            /* end (ilambda=1;ilambda<=10;ilambda++)*/

}                /* end (ips=1;ips<=20;ips++) */

}                    /* Distribution */

fcloseall();

}                   /* end main() */
```