

Library copy

SIMULATION OF HYBRID-ETHERNET LAN PROTOCOL

*A dissertation submitted
in partial fulfilment of the requirements
for the award of the degree of*

**MASTER OF TECHNOLOGY
in
COMPUTER SCIENCE**

by

G. SRINIVAS REDDY

TH-6377



JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI

**SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110 067
INDIA**

JANUARY 1996

1474

SIMULATION OF HYBRID-ETHERNET LAN PROTOCOL

JAWAHARLAL NEHRU UNIVERSITY

*A dissertation submitted
in partial fulfilment of the requirements
for the award of the degree of*

**MASTER OF TECHNOLOGY
in
COMPUTER SCIENCE.**

by

G. SRINIVAS REDDY

58 p + fig + tables




JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI


**SCHOOL OF COMPUTER AND SYSTEMS SCIENCES
JAWAHARLAL NEHRU UNIVERSITY
NEW DELHI - 110 067
INDIA**

JANUARY 1996

CERTIFICATE

This is to certify that the dissertation entitled **SIMULATION OF HYBRID-ETHERNET LAN PROTOCOL** being submitted by **G.SRINIVAS REDDY** to School of Computer and Systems Sciences, Jawaharlal Nehru University, New Delhi, in partial fulfilment of the requirements for the award of the degree of Master of Technology in Computer Science, is a bonafide work carried by him under the guidance and supervision of **Dr. Rajesh C. Phoha**. This work has not been submitted elsewhere for any other purpose.


Prof. G.V. Singh
Dean, SC&SS 5-2-96
Jawaharlal Nehru University
New Delhi 110 067.


Dr. Rajesh C. Phoha
Associate Professor, SC&SS
Jawaharlal Nehru University
New Delhi 110 067.

ACKNOWLEDGEMENTS

It is a great pleasure for me to express my gratitude to Dr. Rajesh C. Phoha for invaluable suggestions, guidance and inspiration throughout my project work.

I would like to thank Prof.G.V. Singh for providing me the opportunity and necessary facilities to complete this project.

Lastly, I would like to thank my friends and classmates for their support rendered during my stay in J.N.U.

G. Srinivas Reddy

ABSTRACT

Local Area Networks (LANs) have become a vital part of modern computing. LANs so far have been developed using linear bus or ring topologies. The reason for choosing them is simple network control and routing, and minimization of hardware costs by the use of single transmission medium. However, these topologies have the disadvantages of performance degradation under increased network traffic, and limitations on performance characteristics due to lack of concurrent communication. And also their performance cannot be upgraded to accommodate increased system load.

A solution to the above problem is the Hybrid-Meshnet network architecture, which allows insertion of dedicated (direct) datalinks into existing LANs: Ethernet and Token Ring. In this thesis, Hybrid Ethernet i.e. ethernet with connected data links is discussed and its performance comparison with pure ethernet through simulation is done. The CSMA/CD protocol model is developed using the event scheduling approach. The simulation results obtained for standard Ethernet parameters as inputs, are validated against the analytical values. After the validation is successful, the simulation model is modified to simulate the Hybrid

Ethernet. The performance parameters such as throughput and average packet delay are obtained under different traffic loads.

The simulation results of Ethernet and Hybrid Ethernet are presented and they are compared. The results conclude that with connected datalinks performance of the Ethernet improves significantly.

CONTENTS

CHAPTER 1: INTRODUCTION	1-9
1.1 Local Area Networks	
1.2 MAC Protocols	
1.3 Performance Measures	
1.4 Objectives	
CHAPTER 2: CSMA/CD WITH CONNECTED DATALINKS	10-13
2.1 Hybrid Ethernet	
2.2 Ethernet Vs. Hybrid Ethernet	
2.3 Hybrid Ethernet Operation	
2.4 Advantages	
CHAPTER 3: SIMULATION	14-31
3.1 Introduction	
3.2 Discrete System Simulation	
3.3 Simulation Languages Vs. High Level Languages	
3.4 Random Number Generation	
3.5 Simulation Output Analysis	
CHAPTER 4: MODELLING	32-43
4.1 Modelling Assumptions	
4.2 Simulation Model	
4.3 Input and Output parameters	
4.4 Validation	
CHAPTER 5: PROGRAMMING APPROACH	44-54
5.1 Major Programs	
5.2 Implementation of Hybrid Ethernet	
5.3 Major Functions	
CHAPTER 6: RESULTS AND CONCLUSION	55-58
6.1 Validation of Pure Ethernet	
6.2 Simulation of Hybrid Ethernet	
6.3 Conclusion	

BIBLIOGRAPHY

CHAPTER - 1

INTRODUCTION

1.1 LOCAL AREA NETWORKS (LANs)

Local Area Networks are widely used nowadays in the offices.

A Local Area Network is defined as:

"A Local Area Network is a communication network that provides interconnection of a variety of data communicating devices within a small area".

Some of the key characteristics of local area networks are:

- high data rates (up to 100 Mbps)
- short distances (0.1-50 km)
- low error rate (10^{-8} - 10^{-11})
- geographically confined to a small area
- generally privately owned

Local Area Networks are characterized in terms of their topology.

Bus and ring topologies are widely popular. In Bus topology, all devices share a common communication medium. Only one device can transmit at a time, and transmission employs a packet containing source and destination address fields and data. The ring topology consists of a closed loop, with each node attached to a repeating element.

1.2 MEDIUM ACCESS (MAC) PROTOCOLS IN LANS

The most commonly used MAC protocols are CSMA/CD, tokenring and tokenbus.

In the CSMA/CD protocol, a terminal with a packet ready for transmission, senses the channel and proceeds as follows:

- (1) If the channel is sensed idle, the terminal initiates transmission of packet.
- (2) If the channel is busy, then depending on the persistence algorithm the terminal transmits.

There are mainly three persistence algorithms. They are nonpersistent, 1-persistent and p-persistent.

In the nonpersistence case, the station backsoff a random amount of time and then senses the medium again. It is effective in avoiding collisions; two stations wishing to transmit when the medium is busy are likely to backoff for different amounts of time. The drawback is that there is likely to be a wasted idle time following each transmission.

In case of 1-persistent algorithm, the station continues to sense the medium until it is idle, and then it transmits. This attempts to reduce the idle time by allowing a single waiting station to transmit immediately after another transmission. Unfortunately, if more than one station is waiting, a collision is guaranteed.

In case of p-persistent algorithm, the station continues to sense the medium until it is idle, then transmits with some preassigned probability. Otherwise it backs off a fixed amount of time, then transmits with probability p or continues to backoff with probability (1-p). This algorithm is a compromise that attempts to minimize both collisions and idle time.

- (3) If a collision is detected during transmission, immediately cease transmitting the packet and transmit a brief jamming signal to ensure that all stations know that there has been a collision. After transmitting jamming signal, wait a random amount of time, then attempt to transmit again.

For the proper operation of the CSMA/CD protocol, the packet size should be sufficient to permit collision detection in the worstcase. For a baseband system, with two stations that are as far apart as possible

(worstcase), the time that it takes to detect a collision is twice the propagation delay.

The most common choice in the persistence algorithms is 1-persistent, used by the Ethernet and IEEE 802 standard. With the 1-persistent scheme, the wasted idle time is eliminated at the cost of wasted collision time.

The time wasted due to collisions is short if the packet size is long relative to the propagation delay. With random backoff, two stations involved in a collision are unlikely to collide on their next tries. To ensure stability of this backoff, a technique known as binary exponential backoff is used. A station attempts to transmit repeatedly in the face of repeated collisions, but the mean value of random delay is doubled after each collision. After a number of unsuccessful attempts the station gives up and reports an error.

Token Bus is a technique in which stations on the bus or tree form a logical ring; that is, the stations are assigned positions in an ordered sequence, with the last member of the sequence followed by the first. Each station knows the identity of the stations preceding and following it. When a station receives the token, it is granted control of the medium for a

specified time, during which it may transmit. When the station completes its transmission or if its assigned time slice has expired, it passes the token on to the next station in the logical sequence. Hence steady state operation consists of alternating data transfer and token transfer phases.

In the case of token ring, a small token packet circulates around the ring: when all stations are idle, the token packet is labeled as 'free' token. A station wishing to transmit waits until it detects the token passing by, alters the bit pattern of the token from "free" token to "busy" token and transmits the packet immediately following busy token. Since there is no free token on the ring, all other stations wishing to transmit must wait. The packet on the ring will make a round trip and be purged by the transmitting station. The transmitting station inserts a "freetoken" on the ring when the station has completed transmission of its packet and busytoken has returned to the station. When transmitting station releases a new free token, the next station downstream with data to send will be able to seize the token and transmit.

The comparison between tokenring, token bus and CSMA/CD is done by many. The analysis [3] of these comparisons yielded the following conclusions.

1. Token ring is least sensitive to workload.

2. CSMA/CD offers shortest delay under light load, whereas it is most sensitive under heavy loads.

Under heavy load, the disparity between token passing and CSMA/CD is due to the instability of CSMA/CD. As offered load increases, so does throughput, until beyond some maximum value, throughput actually declines as offered load increases. This results from the fact that there is an increased frequency of collisions. More packets are offered but fewer successfully escape collision. Worse, those packets that do collide must be retransmitted, further increasing the load.

1.3 Performance Measures

The performance analysis is of concern in the design and selection of a network for a specific application. Given a certain collection of devices, with certain traffic characteristics, the requirement is that the local area network has adequate throughput for the expected load.

The factors significant in determining the performance of a local area network are the data rate of the medium and the average signal propagation delay between stations on the network. The product of data rate and average signal propagation delay is an important factor for determining the local area network performance.

The performance parameters for the analysis of LANs are throughput and Normalized average packet delay as function of offered load to the network. The network offered load is the total of offered load by the individual stations. It depends on interpacket transmission intervals of these stations and mean packet length.

Throughput is the number of packets transmitted per unit time. The part of channel utilization under successful data transmission. The average packet delay T_d is defined as the time difference between the time at which a station first attempts to transmit the packet to the time at which transmission of packet is successfully completed. This includes the channel waiting time, collision recovery time, back off time and the packet transmission time.

If T_p is the mean packet transmission time in case of no collision, T_p is the ratio of packet size in bits to the channel speed in bits per sec. The Normalized average packet delay(D) is defined as:

$$D = \frac{\text{The average packet delay}}{\text{Mean packet transmission time}} = \frac{T_d}{T_p}$$

If T_i is the mean interval between the time at which a station completes transmission of one packet and the time at which it initiates next, and if there are N identical stations in the network and, if the

network is a closed system, the offered load G is defined as

$$G = \frac{NT_p}{T_i + T_p}$$

If the network is an open system, the offered load G is

$$G = \lambda T_p$$

Where λ is the arrival rate of new request at the

channel $= N \lambda_i$

Where λ_i is the arrival rate of packet at a particular station.

In order to design the networks, it is necessary to have good performance evaluation techniques. With adequate performance modelling it is possible to anticipate and correct incorrect performance while avoiding cost of altering an existing network.

Performance of any type of network can be measured by using the three different ways. The first one being actually building the operating network and then measuring its performance by running the network at different loads. It is costly as the performance is measured only after network has been setup. The second approach is to use analytical methods. This approach has the advantage of being so simple that it requires less effort and time required to measure performance. But, the various simplifying assumptions, for example, mathematical calculations always

assume symmetrical network, made the mathematical analysis may not perfectly reflect the performance of actual network. The third approach is to simulate the network using computer model. This approach allows modelling of networks with desired degree of complexity, rather than being simple like analytical approach, and hence it produces exact performance results. The disadvantages of this approach are the time spent on designing, writing, executing a sophisticated simulation program and the efforts necessary to validate the simulation and the efforts necessary for the analysis of obtained statistical simulation output.

1.4 OBJECTIVES

Linearbus and ring topology networks have significant disadvantage in certain applications. Since Ethernet uses the bus topology establishing private, secure and interactive communications is difficult. The maximum throughput achievable for the CSMA/CD protocol (Ethernet) is one. The objective is to study the performance of Hybrid Ethernet, which allows the insertion of direct data links between the stations on the ethernet. The objective is to show that the maximum throughput achievable with Hybrid Ethernet exceeds one and average packet delay decreases with load even with small increase of hardware links cost.

Ethernet Parameters	Values
Channel speed	10 Mbps.
Slot time	64 byte
Jam Signal size	32
Inter frame gap	9.6 microsecs.
Max. Retransmission attempts	16
Source or destination Address size	48 bits
Maximum frame size	512 bits
Maximum frame size	1518 bytes

TABLE 1.1

IEEE 802.3 Standard Ethernet parameters

CHAPTER - 2

CSMA/CD WITH CONNECTED DATA LINKS

2.1 HYBRID ETHERNET

Most of the local area networks are based on simple bus or ring topology. However linear bus and ring topologies have disadvantages in certain applications. The disadvantages are:

1. Limitations on performance characteristics due to the lack of concurrent communication.
2. Interactive communication is difficult to achieve.
3. Private and secure communication between two hosts on the networks is not possible.
4. Under heavy load, the performance degrades and uncertainty of data delivery increases.
5. Upgrading the performance for increased system load is difficult to achieve.

The above limitations can be overcome by using Hybrid topology network, which allows the insertion of direct data links between hosts on network.

2.2. ETHERNET VS.HYBRID ETHERNET

The basic operating characteristics of CSMA/CD protocol are as follows. Even when there is no packet waiting to be transmitted, the CSMA/CD MAC sublayer monitors the physical medium for traffic. When the station becomes ready to transmit, the behaviour of the difference mechanism depends on protocol used. Ethernet uses the 1-persistent CSMA/CD protocol. So in Ethernet, if the channel is idle, the packet is transmitted. If the channel is busy, the station defers transmission until the channel is sensed idle and then immediately transmits. If a collision is detected, the station aborts the packet being transmitted and sends a jamming signal. After the jamming signal has been transmitted the station involved in collision waits random amounts of time and then tries to send their packets again.

CSMA/CD protocol has got the advantages of being simple in design and since all hosts are distributed along the bus, there are no central elements to fail. The failure of a particular active element only affects communication of that station. Even though it has excellent advantages, the research on it revealed a number of disadvantages. The first, as the traffic increases, many active elements collide and only a small fraction of true communication band width is used for the transmission of valid data. The other is, as the number of stations connected to the network increases, the

number of collisions increases and hence less and less valid data gets through the network, and also uncertainty of data delivery increases.

Hybrid Ethernet adds the advantages of mesh networks to the existing ethernet while retaining its simple network control. In Hybrid Ethernet each station will have two communication channels. The first is the ordinary Ethernet and the other consists of an arbitrary collection of data links. The configuration of Hybrid Ethernet is shown in Fig. 2.1. In the Hybrid Ethernet Architecture a LAN can be installed similar to an ordinary Ethernet LAN and may be upgraded, depending on the application requirements, to have direct data links between stations. With the addition of direct data links, the network can now accommodate interactive, secure communication between node pairs and improves the overall network performance.

2.3 HYBRID ETHERNET OPERATION

In hybrid Ethernet, two types of transmission are possible.

1. The broadcast data transmission, i.e. through the ether.
2. The direct transmission i.e. through connected data link.

When a frame arrives at the network interface unit from the upper layer for transmission over the hybrid ethernet LAN, either the direct

transmission or broadcast transmission is chosen by the network controller.

A direct transmission method is chosen only when direct data link to the destination station is available. When data link to the destination station is not available, then the frame will be broadcasted over the ether. Thus, broadcast transmission is used only when a direct transmission is not available.

2.4 ADVANTAGES

Hybrid Ethernet overcomes some of the disadvantages of the Ethernet and its advantages over Ethernet are:

1. It supports flexibility in upgradation. The performance and reliability of network can be adjusted to suit the application environment.
2. Since the data is transmitted both through the ether and connected data links, it provides the concurrent communication. Therefore, the maximum throughput achievable may exceed one, which is not possible in Ethernet.
3. Highly secured and interactive data transmissions are possible to maintain by having direct data link to the specific destination node which is adjacent to the source node.

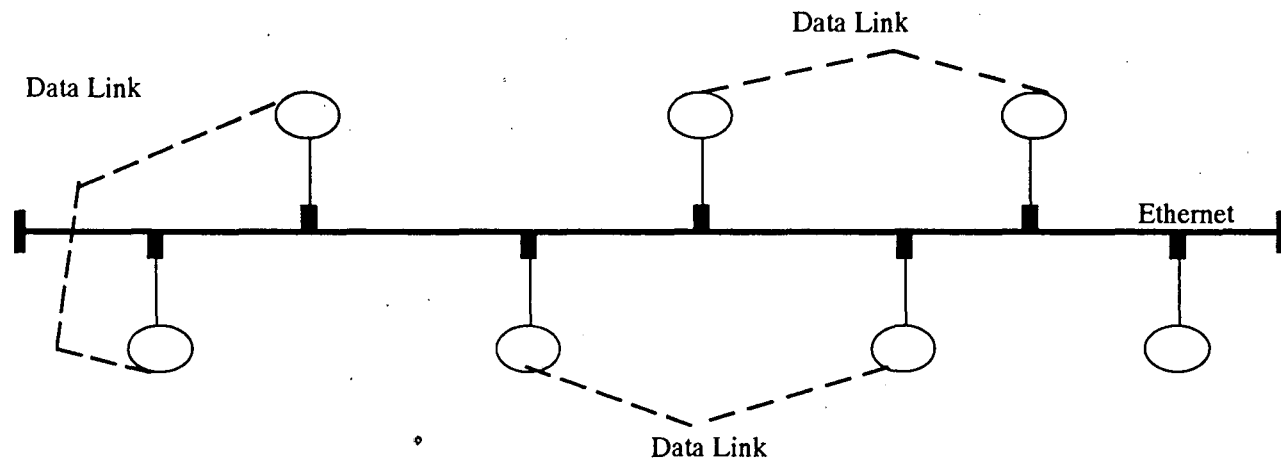


FIGURE 2.1: AN EXAMPLE CONFIGURATION OF HYBRID-ETHERNET

SIMULATION

3.1 INTRODUCTION

Simulation allows to copy the behaviour of a system under study. Simulation evolved as a powerful technique for solving a wide variety of problems, which are not easily solved. Simulation allows to copy the behaviour of a real life system, however complex, and get a measure of its performance. Simulation is becoming increasingly popular in the class of dynamic systems like communication networks with random traffic inputs. Simulation provides the means to visualize a system that is not yet built, to analyse a system to determine critical elements and to act as a design accessory in order to evaluate proposals. Simulations use simulating models and based on these, perform experiments which enable the analyst to determine the behaviour of a system.

A Model of a system is a collection of entities and their relationships. Entities are described by attributes. For example, customer in a queuing model is an entity, and customer is described by waiting time, service time, these are attributes. The description of values of all the attributes of an entity at a specified instant is called the state of an entity.

Continuous systems are those in which the change in the state of the system entities are continuous with time, and discrete systems are those in which changes in the state of the system entities occur at discrete points in time.

3.2 DISCRETE SYSTEM SIMULATION

Discrete systems are those systems in which changes in the objects are discontinuous. Each change in the state of the system is called an event. Therefore the simulation of a discrete system is often referred to as discrete - event simulation. In simulating any dynamic system, continuous or discrete, there must be a mechanism for the flow of time. For this we must advance time, keep track of the total elapsed time, determine the state of the system at the new point of time, and terminate the simulation when the total elapsed time equals or exceeds the simulation period. For continuous systems, time is advanced in small increments as long as needed. In simulation of discrete systems, there are two fundamentally different models for moving a system through time.

In Discrete Event Simulation, the state of the simulated system is stored in a set of system state variables. Event routines cause state variables to be modified. An eventlist is maintained to control the execution sequence of these event routines. Event list consists of events in

chronological order. Event Routines can add or delete items from the event list and pseudo random number generators in the event routines provide requisite randomness for modifying and scheduling of future events. Running a simulation is the repeated execution of a loop where at each iteration the most imminent event is executed in turn.

3.2.1 Fixed time step model

In time step model a timer or clock is simulated by the computer. This clock is updated by a fixed time interval and the system is examined to see if any event has taken place during this time interval. All events that take place during this period are treated as if they occurred simultaneously at the tail end of this interval. The fixed time step simulation works as shown in flowchart 3(a).

3.2.2 Event to Event model (next event model)

In this simulation model the computer advances time according to the occurrence of the next event. It is advanced from event to event. The system state does not change in between. Only those points in time are kept track of when something of interest happens to the system. Event to event model is preferred to fixed time step model because in this model no computer time is wasted in scanning those points of time when nothing takes place. This waste is bound to occur if a very small value of t is

picked. On the other hand if t is so large that one or more events must take place during each interval then the model becomes unrealistic and may not yield meaningful results. The implementation of event to event model is more complicated than the fixed time step model. The event to event model is described in flow chart 3(b).

One of the three approaches used for modelling a discrete event system is event scheduling approach. In the event scheduling approach, events are scheduled in advance. Whenever an event is scheduled, data identifying the type of the event and time at which event has occurred is placed in event list. The simulation proceeds as follows.

1. The list of scheduled events is searched to find the event with earliest scheduled time.
2. The simulation time is advanced to that earliest scheduled time.
3. State changes and scheduling of new events associated with the occurrence of the event occurred are performed.
4. Repeat the steps 1 to 4 until the simulation time completes.

3.3 SIMULATION LANGUAGES Vs.HIGH LEVEL LANGUAGES

There are languages such as SIMSCRIPT, CSL and GPSS which are designed especially for the simulation. These languages are designed to

optimize the features unique to simulation. These languages

1. provide a convenient representation of elements that appear in simulation model.
2. generate automatically the pseudo random numbers for statistical distribution.
3. facilitate the collection of data and statistics of simulated system.

Even though, these languages provide many features required for the simulation, they have not become popular. The main reasons for not using them widely are:

1. Users are not familiar with these languages.
2. These languages are not easily available to the users.
3. These languages are highly complex in dealing with the problem.

On the otherhand, highlevel languages such as PASCAL, FORTRAN, C and C++ have the advantage of being familiar with the users, easy availability, and the model can be chosen in the way you like where as the special purpose simulation languages cannot. General purpose highlevel languages are being used widely to simulate the discrete event systems because of their advantages.

3.4 RANDOM NUMBER GENERATION

Queueing networks are discrete stochastic systems. In order to simulate them, a source of generating random variates is required. It is possible to generate random variables for the given type of probability distribution. In order to prepare the input parameters for the simulation, the first step is to have a deterministic algorithm that produces the uniform random variables. Sequences generated in deterministic manner, which appear to be random are called pseudo random sequences. Many statistical tests have been devised to test the randomness properties of the generated sequence. Important property to be satisfied by the generated sequence is that they should be uniform and independent. A good pseudo random generator must be capable of producing different sequences of numbers which are:

1. Uniformly generated between 0 and 1.
2. Statistically independent.
3. Reproducible.
4. Non repeating for any required length.

3.4.1 Uniform random number generator

Several methods exist for generating the uniform random numbers. These methods are based on some recursive relation. Each new random number is generated from the previous value by applying some scrambling

operation. Multiplicative congruential generator is one method for generating the random numbers. It consists of computing

$$X_{n+1} = (CX_n) \text{ mod } M$$

Where X_{n+1} is the (n+1)th random number, X_n is the previous random number and C is a constant multiplier. The value of X_0 is called seed of the random number generator. With this generator, the maximum period of 2^{b-2} is obtained when

$$M = 2^b; b > 4$$

$$C = 8K + 5; K = 0, 1, 2, \dots$$

$$X_0 \text{ is odd}$$

The value of M is chosen to be equal to the largest prime number which is less than 2^b . For a computer system the value M is equal to largest prime number that can be represented in it. If the multiplier C is a primitive root modulo M, then the generator will have a maximal period of M-1.

3.4.2 Non uniform Random number generation

It is possible to generate random numbers for any distribution using the method of inverse transformation from the uniform random numbers.

If X_i is a sequence of random numbers which are uniformly



distributed in the interval (0,1) and if Y has probability density function $f(y)$ and cumulative distribution function $F(y)$ then the sequence of random numbers Y_i are generated by the operation

$$Y_i = F^{-1}(X_i) \quad \text{----- 3.1}$$

For example, for an exponentially distributed sequence

$$f(y) = (1/\lambda) e^{-y/\lambda} \quad \text{where } \lambda \text{ is the arrival rate}$$

and $F(y) = 1 - e^{-y/\lambda}$

From the inverse transformation method, if x_i 's are the uniform random numbers then the exponentially distributed random sequence y_i are generated by $y_i = -\lambda \log(1-x_i)$

$$y_i = -\lambda \log(x_i) \quad \text{----- 3.2}$$

Thus equation (3.2) will generate the exponentially distributed random numbers. Two ways of generating random variables is discussed below:

1. **Exponential Distribution:** To generate an exponential distribution, first of all a random number 'r' is generated with uniform distribution i.e. the number lies between 0 and 1. Then calculate the value of E_i using the following equation.

$$E_i = (\text{Mean}) * \ln(r)$$

E_i gives the instance of the desired exponentially distributed random variate. This process of generation of exponentially distributed random variate is described in flow chart 3(c).

2. **Poisson Distribution:** Poisson distribution is a discrete distribution in which the probability of an event occurring exactly "k" times during a time interval t is given by the probability mass function

$$G_k(t) = ((t*L)^k) * (1 / (k!)) * (e^{-(t*L)})$$

Where L is the average number of times the event occurs in a unit period. The procedure required to get a poisson distributed random variate is to form the product of successive uniformly distributed random numbers, until the following equation is satisfied.

$$U_i < \exp (-L)$$

Where U_i is a uniformly distributed (0,1) random number, L is the mean of the distribution. The desired random variate instance N_i will be one less than the required number of uniformly distributed random numbers as shown in flowchart 3(d).

3.5 SIMULATION OUTPUT ANALYSIS

In the simulation process the question which comes into mind is how long to run the simulation? Hence the approaches to estimating and controlling simulation output accuracy is needed. The subject which deals with this is called simulation output analysis.

If the simulation run length is determined by the problem itself, then this type of simulation is called terminating or transient simulation.

For this type of simulation, the question becomes how many times the simulation must be repeated to achieve a specified accuracy.

In the steady state simulation, both the initial conditions and the length of the simulation are determined by the modeler, and the measure of the interest is the limiting value reached as the length of simulation run goes to infinity. Practically, run lengths are finite and our problem is to determine how close the mean estimated from the sample values is to the true mean of the distribution.

Most simulations in the computer system design environment are steady state simulations. The performance measure of interest is the mean (average) value of simulation output variable.

For a discrete time random process, the simulation produces a sequence of n sample values $X_1, X_2 \dots X_n$ whose mean (denoted by \bar{X}) is

$$\bar{X} = \sum_{i=1}^n x_i/n$$

As n approaches infinity, \bar{X} converges to a limiting value $E[X]$ called expectation of X . Call $E[X]$ as distribution mean (μ).

The variance is a measure of dispersion of a distribution. The variance of a set of values x_1, x_2, \dots, x_n

$$s^2 = \sum_{i=1}^n (x_i - \bar{x})^2 / (n-1)$$

s^2 is called sample variance.

As n approaches infinity, s^2 converges to limiting value $E[s^2] = E[(x-\mu)^2]$ denoted by σ^2 . The square root of variance is standard deviation.

3.5.1 Confidence interval

How close the sample mean obtained from finite length simulation is to the distribution mean μ or how long run lengths have to be to obtain a sample mean arbitrarily close to μ is to be estimated. The answer to this can be found by computing a measure called confidence interval.

Suppose we have a set of n sample values Y_1, Y_2, \dots, Y_n from a distribution with true (but unknown) mean μ . The sample mean is

$$\bar{Y} = \sum_{i=1}^n Y_i / n$$

By defining $1 - \alpha$ as the probability that the absolute value of the difference between sample mean and μ is equal to or less than H .

$$P[|\bar{Y} - \mu| \leq H] = 1 - \alpha$$

Then a confidence interval for the mean is defined as

$$P[\bar{Y} - H \leq \mu \leq \bar{Y} + H] = 1 - \alpha$$

The interval $\bar{Y} - H$ to $\bar{Y} + H$ is called the confidence interval. H is called the confidence interval half width, and $1 - \alpha$ is called the confidence level or

confidence coefficient. Typical values of which are 0.90 or 0.95. The confidence level $1-\alpha$ is specified by the analyst. H then is determined by the sample values, number of samples, and the value of α . H is a function of random variables and so is a random variable itself.

When $Y_1, Y_2 \dots Y_n$ are independent random variables from a normal distribution with mean μ , H is given by

$$H = t_{\alpha/2; n-1} s/n^{1/2}$$

where $t_{\alpha/2; (n-1)}$ is the upper $\alpha/2$ quantile of a distribution with $(n-1)$ degree of freedom and s^2 is the sample variance

$$s^2 = \sum_{i=1}^n (Y_i - \bar{Y})^2 / (n-1)$$

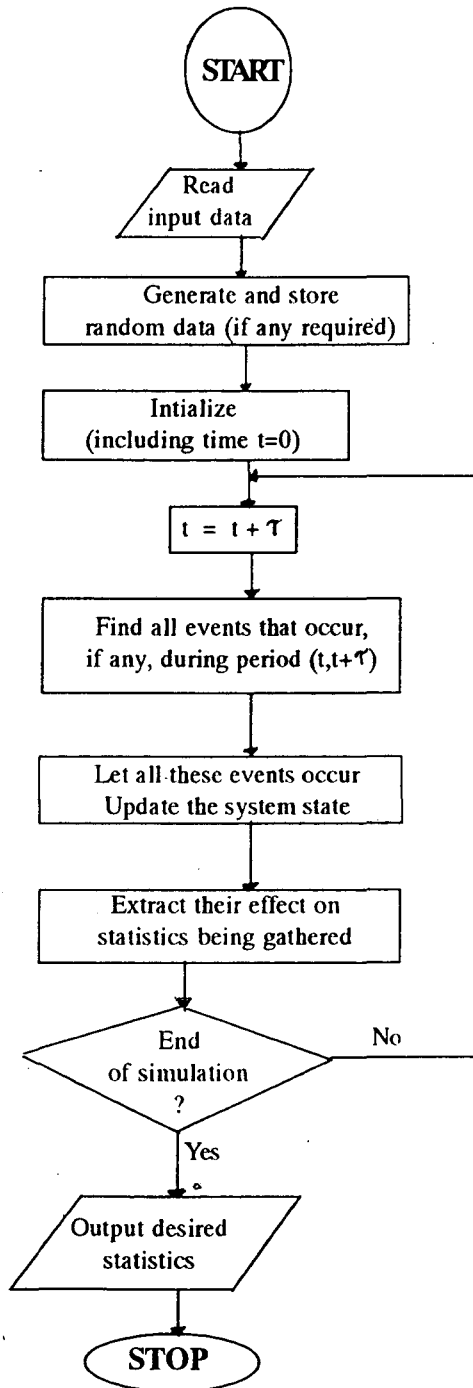
Estimation of a confidence interval is what the simulation output analysis is all about. A number of methods for estimating a confidence interval for the mean of simulation output variable have been described. These include methods called. (1) Replication (2) Batch means (3) Regeneration (4) Auto regression (5) spectral analysis (6) standardized time series.

Approaches to confidence interval estimation can be classified as fixed sample size procedures or as sequential procedures. In a fixed sample size procedure a simulation experiment of total fixed length is performed

and the confidence interval is estimated from the results of the experiment upon its completion. In a sequential procedure, the desired accuracy is specified, confidence interval estimates are computed at selected intervals and the experiment is continued until the desired accuracy is obtained.

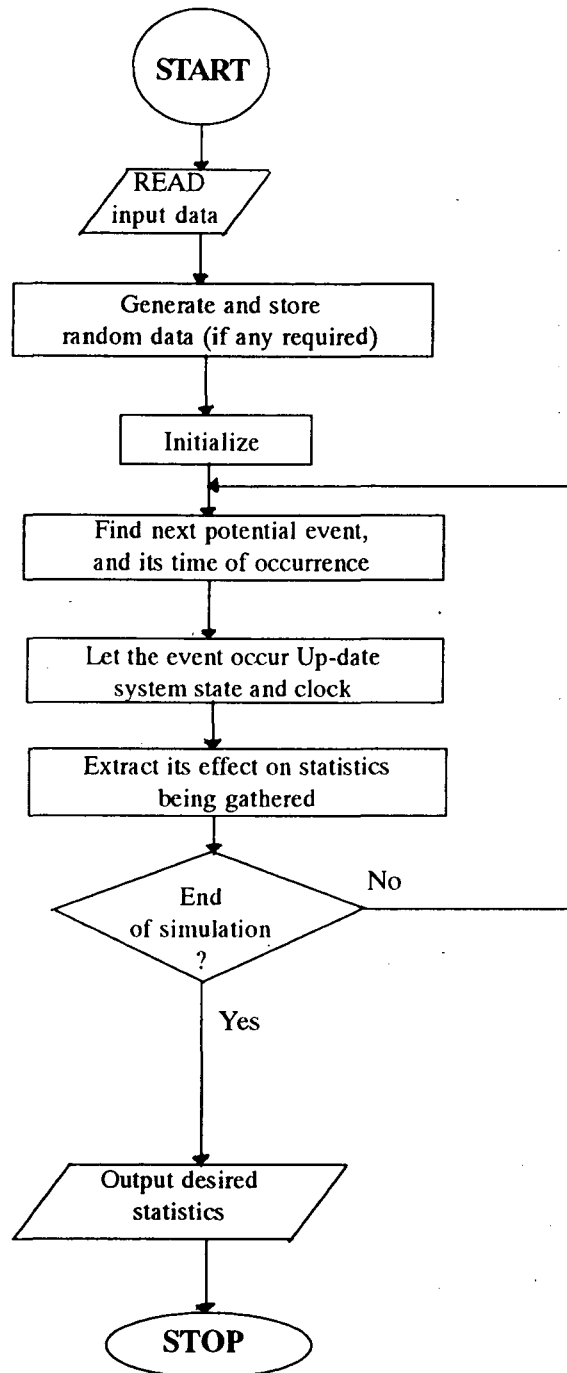
In the batch means analysis divide one long run in to a set of K subruns of length m , called batches, computing a separate sample mean for each batch, and using these batch means compute the grand mean and confidence interval.

Assuming deletion is used to achieve steady state initial conditions for the first batch, each subsequent batch begins with the system in the steady state. Since warm-up effects have to be dealt with only once, rather than K times as in the case of replication, the batch means method is potentially more efficient i.e., fewer sample values are needed to achieve a given accuracy.



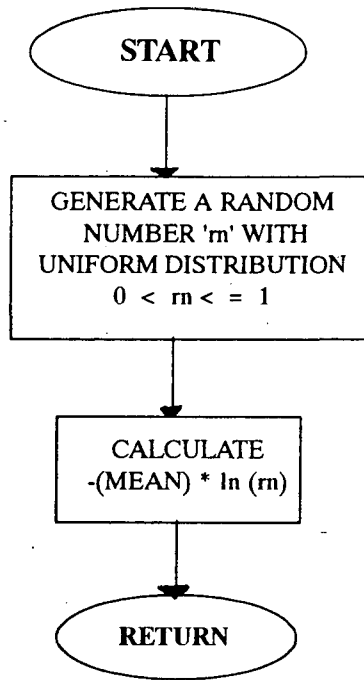
FLOW CHART 3(a)

FIXED TIME STEP SIMULATION



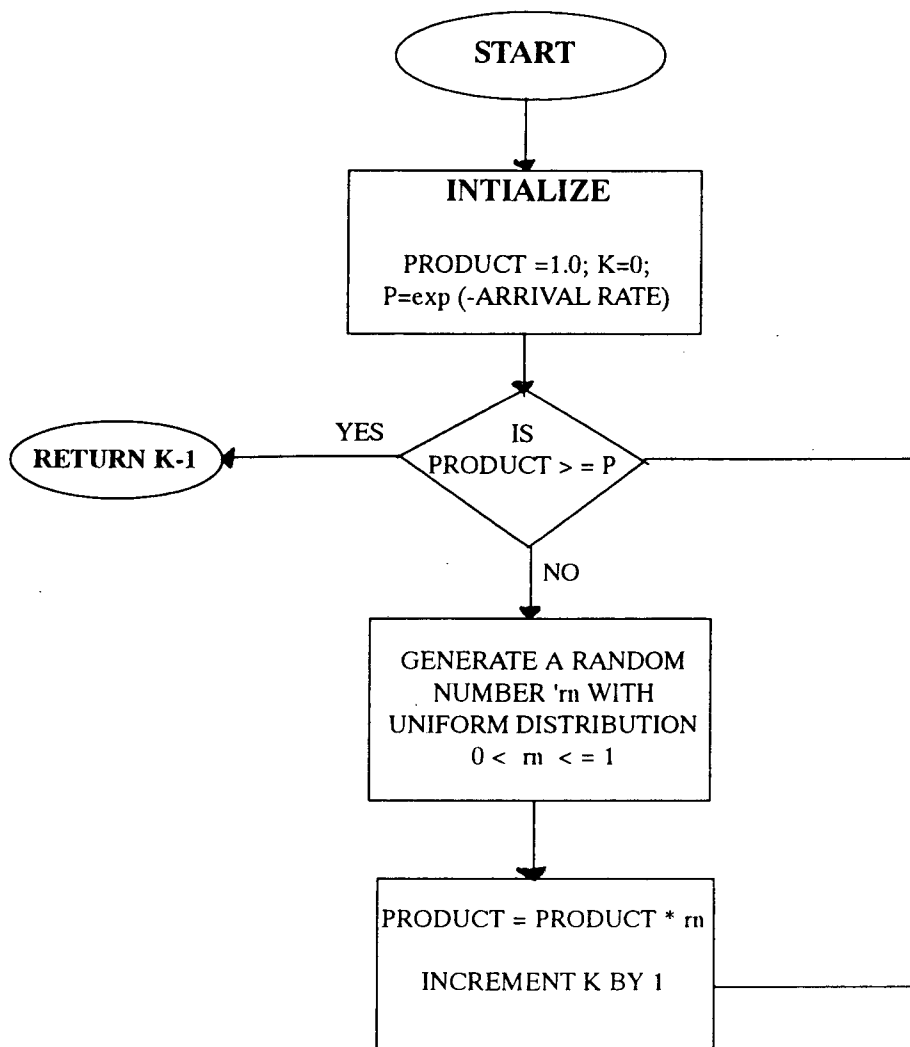
FLOWCHART 3(b)

NEXT EVENT SIMULATION



FLOWCHART 3(c)

**EXPONENTIALLY DISTRIBUTED
RANDOM VARIATE**



FLOWCHART 3(d)

POISSON DISTRIBUTED RANDOM VARIATE

CHAPTER - 4

MODELLING

In this chapter modelling of Hybrid Ethernet is described. The different modelling assumptions that have been taken in modelling the hybrid ethernet are listed in the following section.

4.1 MODELLING ASSUMPTIONS

The following assumptions have been used in performance modelling of Hybrid Ethernet:

1. Mediums of transmission (i.e. Ether and connected data links) are assumed to be completely error free.
2. The processing time for the packets at each station in the network is negligible.
3. A single destination is assumed for each packet.
4. Size of the buffer is limited at each station.
5. The packet arrivals at the stations in the network are assumed to be poisson.
6. The interarrival times of packets are assumed to be exponentially distributed.

7. The length of the packets is fixed i.e. packet transmission time is fixed.
8. For the packets at each station, first come first served basis is assumed.
9. The backoff algorithm assumed is truncated binary exponential back off with base back off time equal to double the propagation delay .
10. No concurrent communication through the direct data links.

4.2 SIMULATION MODEL

The discrete event scheduling approach has been used for the performance modelling of hybrid ethernet. This approach is more suitable as the network has many arrivals and fewer activities. For example, consider a network having fifty nodes and each node has traffic. If in case active scanning approach is used, the modelling requires a total number of fifty arrival clocks to be scanned before advancing the simulation time to next event time. And also the process interaction approach does not offer any additional advantages. From the programming point of view, the event scheduling approach is easy to implement.

Before developing the model for Hybrid Ethernet it is required to first simulate the ethernet model and validate it against the standard

results. This ensures that the results generated for the Hybrid Ethernet are accurate. The ethernet protocol chosen for the simulation is 1-persistent CSMA/CD protocol.

The Ethernet model assumes that a set of stations are connected to a single bus and each station has independent packet arrivals and processing queue. It is assumed that Packets are transmitted on ether in both the directions. A station always attempts to transmit the first packet inside the processing queue before attempting to transmit the next packet in its processing queue.

As stated earlier, discrete event simulation is used for developing the model. In the discrete event model, events which occur in the model at discrete events of time are kept in the event list. An event list is a list of nodes arranged in ascending order of time, with each node having the data relating to the type of event, the time at which it will occur and a pointer to the next node in the event list. The next imminent event to occur is the the type of the event in the headnode of the event list and the time at which it occurs is the event time in the headnode. The event list contains the following type of events:

1. Channel idle at the upstream station.
2. Channel idle at the downstream station.

3. Next packet arrival at each station.
4. Collision back off period completion events for the stations which are in the backlog state.
5. Packet transmission completion event.

In the model, the simulation clock is initially maintained as zero and the event list is initialized. During the simulation run, whenever an event occurs, simulation clock is incremented to the time at which it has occurred and the action corresponding to that event is performed.

The Arrival requests for the stations are generated from the exponentially distributed inter arrival times. Whenever a packet arrival event occurs, the packet is placed in the processing queue. If the processing queue is full, then the packet is discarded. The next arrival request for the station is scheduled from the current arrival request time and the interarrival time between these requests. Inter arrival time is generated by exponential distribution with mean arrival rate of packets at a station.

The events, channel idle at upstream node and channel idle at downstream node are scheduled each time after a collision and a successful transmission. When channel idle at upstream station (channel idle at downstream station) occurs then the channel status as seen by the current

upstream station (current downstream station) is set to idle. If that station does not attempt to acquire the channel then channel idle event for nodes adjacent to current upstream (downstream) station is scheduled. If an arrival occurs at the current upstream (Current downstream) station then next arrival request is generated and is scheduled in event list for that station. Also if the station is not in backlog state and channel status as viewed from there is idle, then attempt to acquire the channel. A station tries to acquire the channel in one of the following conditions:

1. It senses the channel as idle, i.e. the channel status seen by the station is idle at that instant and if any of the following conditions (a) or (b) is satisfied.
 - a. The station is in the backlog state and its backlog completion event ends.
 - b. The station is not in backlog state and the processing queue is not empty.
2. Channel status is idle from that station and an arrival has occurred at that instant.

The channel acquisition attempt by a station can cause either a collision or a successful transmission. In case no collision is guaranteed then that station is said to have acquired the channel. The collision detection is made as follows.

When a station i attempts to acquire the channel for packet transmission, then list of scheduled events at other stations and backlog status of the other stations, and number of packets present in the processing queues of other stations are checked to find the possibility of a collision. The collision window between the station i , which is attempting to acquire the channel and any other station j is defined as the sum of current simulation time and the propagation delay between the two stations. A collision occurs under any of the three following conditions.

1. If any other station also made channel acquisition
2. If any other station j which is not in backlog state has got the packet arrival event to occur within the collision window.
3. If any other station j is in backlog state and its backlog completion event occurs within the collision window.

The above checks are made at all other stations. If the channel acquisition attempt results in a collision then the farthest upstream and downstream stations participating in collision are determined. The collision detection time and the transmission stop times for the farthest upstream and downstream stations participating in collisions are determined. The time at which the channel again becomes idle at upstream and downstream stations is found from the transmission stop times and these are scheduled in the event list. For those stations which participated in the

collisions the backoff periods are generated using truncated binary exponential backoff algorithm and these times are scheduled in the event list.

In case the collision has not occurred, it means that station has acquired the channel then schedule the packet transmission completion event at the current simulation time plus the packet transmission time.

The truncated binary exponential backoff algorithm is used to delay the retransmission attempt of a station after it involves in collision. the delay due to the backoff algorithm is calculated as

1. Increment the Retransmission attempt by one.
2. If Retransmission attempts > 16, then discard the packet.
3. Compute $k = \min(\text{Retransmission attempt}, 10)$
4. Generate random number (r) between $[0, 2^k - 1]$
5. Backoff delay = $r * \text{slot time}$.

After the first collision of a packet, retransmission is done after a backoff delay of 0 or 1 slot times, and after the second collision of the packet, retransmission is after a backoff delay between 0 to 3 slot times, like that retransmission is after a delay from 0 to 1023 slot times for attempt 10-16. After 16 unsuccessful attempts discard the packet.

If the channel tends to become overloaded, then backoff algorithm stabilizes it. As the load on the channel increases, the number of collisions increases. Collisions in effect increases the backoff delay, reducing the load on the channel.

When the packet transmission completion event occurs then the statistics are updated and the channel idle events at the current upstream or current downstream stations is set depending on the successfully transmitted station is upstream or downstream at event time equal to current simulation time plus the inter frame gap between the packets.

The Hybrid Ethernet is simulated as follows. Whenever an arrival request comes from the station generate the destination for that arrival, using uniform random number generator. Before passing the packet to processing queue as in case of ethernet model, Hybrid Ethernet verifies whether that station has got direct data link to the destination station. If it has, then it assumes that packet is successfully transmitted. Otherwise, the packet is stored in processing queue like pure ethernet and pure ethernet model follows for transmitting the packet.

4.3 INPUT AND OUTPUT PARAMETERS

4.3.1. Input Parameters

- Network Configuration
 - Number of station in the network
 - Number and configuration of direct data links
 - Percentage of packets generated sent through the direct datalinks
- Mean inter arrival time of packets
- Packet size: Length of the packet in bits
- Channel speed: Rate at which data is transferred on channel and is given in Mbps.
- Length of Ether: 10Km.
- Jam time: When a station recognizes that a collision has occurred, it transmits a jam signal by immediately abandoning its data transmission to insure that all the other stations on the network know about collision has occurred.
- Interframe spacing: The delay between the time at which a station recognizes as the channel is free and the time at which that station initiates a transmission. The gap between the two transmissions makes sure that the receiving station has time to prepare for a new transmission.

- Propagation speed.
- Slot time: It is rescheduling time used by stations in backing off after a collision has occurred. For Ethernet, its value is specified as 512 bit times.

4.3.2 Output Parameters

- Offered load: The offered load on the network is the sum of individual loads of all stations connected to the network. It is the number of attempts in packet transmission time.
- Average packet delay: The difference between the time at which a station first attempts to transmit a packet to the time at which transmission of packet is complete.
- Throughput: The part of the channel utilization used for successful data transmission.
- Normalized delay: The ratio of Average packet delay to the packet transmission time.

4.4 VALIDATION OF MODEL

The validation of the simulation model is done with the analytical values obtained from the SMV model [6] given in [7]. The throughput S , as a function of the offered load G , for the case of 1-persistent CSMA/CD is given below.

$$S = \frac{(P_{20}+P_{21})e^{-aG}}{\left[\frac{(1-P_{11})P_{20} + P_{10} P_{21}}{G} + (2a+b) (1-P_{10}-P_{11}) + \left\{ (1 - e^{-aG}) \left(2a + b + \frac{1}{G} \right) + e^{-aG} \right\} (P_{20}+P_{21}) \right]}$$

$$\text{where } P_{10} = e^{-G(1+a)} + \frac{1}{2} e^{-G(a+b)} (1 - e^{-2aG})$$

$$P_{11} = G e^{-G(1+a)} + \frac{1}{4} e^{-G(a+b)} \{ (1 - e^{-2aG}) [1 + 2G (a+b)] - 2aG e^{-2aG} \}$$

$$P_{20} = e^{-G(a+b)}$$

$$P_{21} = G (a+b) e^{-G(a+b)}$$

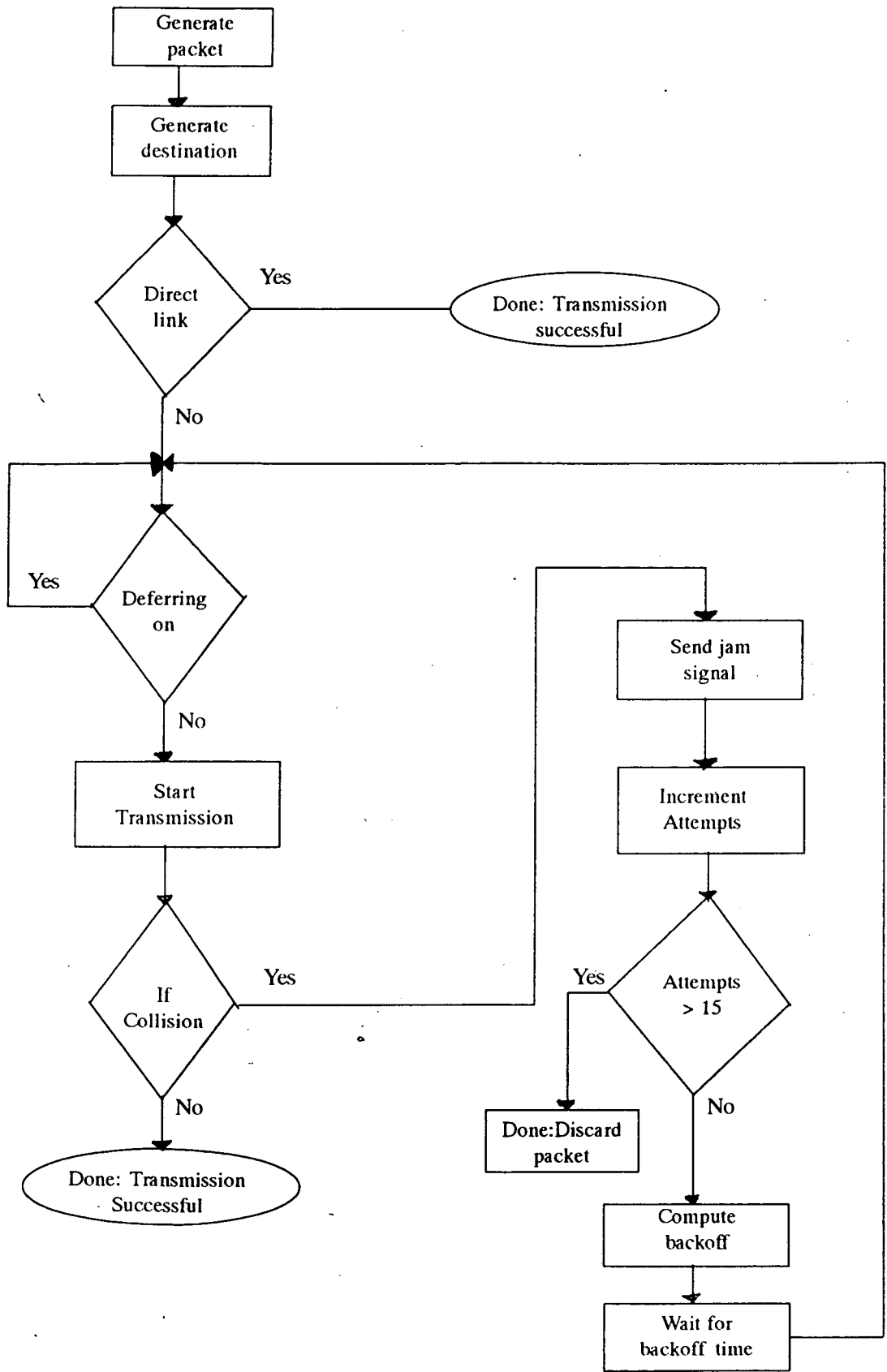


FIG 4.1: FLOW DIAGRAM FOR HYBRID ETHERNET

PROGRAMMING APPROACH

The different programs for the simulation of Hybrid Ethernet are written in C language. The major programs are listed in the following section.

5.1 MAJOR PROGRAMS

intl.c: This program contains all the routines needed for initialising parameters.

rand.c: This contains the functions to generate uniform random numbers between 0 and 1, generating random numbers between 0 and n, any integer, and also for generating exponential random numbers.

evlist.c: This contains the functions which prepare the event list in the ascending order of time and also for preparing a list to maintain the direct links between stations.

arrival.c: This consists of routines to generate initial arrival requests and to generate a next arrival request for a given station from its previous arrival request time and exponential random generator.

csmaed.c: This consists of routines for acquiring the channel incase the channel is idle, to check whether a collision occurs or not after a

station has acquired the channel and also a routine to determine when the channel is going to be idle at the next transmitting station adjacent to the acquired station.

biexp.c: This contains the backoff exponential algorithm to backoff the stations in case the packets transmitted by these stations collide and discard algorithm to discard the packet in case the number of retransmission attempts exceed 16.

stat.c: It contains the functions to determine different statistical variables and also the batch means analysis module to determine the confidence intervals for the simulation run.

makefile: This has got all the files and their inter- dependencies.

The header files required in the project have been kept in the files with .h extension.

5.1.1 Initialization Routines

These routines take network parameters like the number of stations, the channel speed, the packet size and the specification of direct links between the stations etc.

5.1.2 Generation of Random Numbers

The 'C' standard library, math.h, there exists a built in function which generates the random number between 0 and RAND-MAX, which

is an integer. By using this function, random numbers between 0 and 1 are generated.

The exponential random variable is generated by using the exponential distribution function and uniform random variable generator function, for the given arrival rate.

In order to determine the destination of a packet and also for getting random backoff we need a random variable between 0 and n. It is generated with help of rand() function.

5.1.3 Generation of Event list and direct link list

Event list is maintained as linked list. In the event linked list, each event list node is defined to be having the three fields, the event type, the time at which the event occurs and the pointer to the next node. The definition of event list node is:

```
typedef struct eventlistnode
{
    int eventtype;
    float eventtime;
    struct eventlistnode * next;
} evlistnode;
```

In the event list structure, the event types and their meanings are as follows:

Event type (ET)	Interpretation
0	Channel idle at upstream node
1	Channel idle at down stream node
$2 \leq ET \leq N + 1$	Arrival at a station (ET-N-2)
$N + 2 \leq ET \leq 2N + 1$	Backlog completion event of station (ET-N-2).
$2N + 2$	Packet Completion event

Where N is number of stations

The Direct link list is maintained as an array of linked lists. Each direct link list node contains two fields, one is station number and the other is pointer to the next. The definition of direct link list node is

```
typedef struct directlinknode
{
    int stationno;
    struct directlinknode * next;
} directlinklist;

directlinklist * directlink [MAXSTNS];
```

5.1.4 Generation of Arrival time

Arrivals at the stations are poisson in nature and the interarrival times between the packets are exponentially distributed. Since the exponential distribution follows memory less property, in order to generate next arrival time only the present arrival time and interarrival time are enough. The interarrival time is a random number obtained by using function which generates exponential random variable for a particular arrival rate. The next arrival time is the present arrival time plus the interarrival time.

5.1.5 Acquisition of Channel and Collision Detection

Acquisition of Channel

A station is said to have acquired the channel if no other station sends any packet within the collision window. The Algorithm to be performed at the time of acquiring the channel by a station is:

Remove the channel idle events from event list if any exists.

For each station except the station trying to acquire channel

 check whether collision occurs or not

 If collision, store the corresponding station number in
 collision list

If collision list > 0

set the backoff completion events for stations in collision list
in the event list.

Find extreme nodes which participated in collision

Find transmission stop time at these extreme nodes

Set the channel idle events at the transmission

stoptimes plus interframe time in event list

Otherwise,

Set the packet completion event in the event list.

Collision Detection:

When a station i is trying to transmit a packet, Collision is said to occur if any other station j sends a packet within collision window. Collision window is defined as the sum of current simulation clock and the propagation delay between the station i and any other station j . The station j participates in collision if any one of the following conditions (1) or (2) or (3) is satisfied.

1. If the station j has got one or more packets in its processing queues and is not backlogged.
2. If it is backlogged and its backlog completion event ends within the collision window for the station j .
3. If it is not backlogged and it has no packets in its processing queue but it has an arrival within the collision window.

If the collision occurs, collision flag is to set to YES.

5.1.6 Calculation of binary exponential backlog completion time

The Algorithm for computation of backlog completion time is presented below:

1. If $k > 16$ discard the packet.
2. If number of retransmission attempts (k) < 10
 $\text{max} = 2^k - 1$
else $\text{max} = 1024$
3. Find a random number between $[0, \text{max}]$
4. Set backlog completion time = current simulation time +
(random number * slot time).

5.1.7 Determining the statistical parameters

The difference between the time at which the packet successfully transmitted and the time at which the arrival has occurred gives the delay experienced by that packet. This delay is given as an input parameter to the batchmeans analysis module `bms()` to determine the average packet delay and the confidence interval for the delay. The normalized packet delay is determined by dividing average packet delay with packet transmission time.

The throughput of the network is found by dividing the number of successful transmissions divided by the number of packets generated within the simulation time.

5.2 IMPLEMENTATION OF HYBRID ETHERNET

The main algorithm is as follows:

- (1) Intialize the different network parameters
- (2) Set the initial arrival events in the event list
- (3) Set the simulation clock to zero.
- (4) Set the simulation clock to the topevent time.
- (5) Case of topevent type.

Channel idle at upstream station:

It the current upstream station is not backlogged and it has one or more packets in its queue.

or

If it is backlogged and its backlog completion event completes

then

try to acquire the channel

else

Generate Channel idle event at the next upstream station and set it in the event list.

Channel idle at downstream station:

These steps are similar to upstream station.

Arrival at any station:

Generate destination station

Test whether direct link exists for this or not

If exists,

Increment the packets transmitted by one.

Otherwise

Put the packet in the processing queue of that station

If station is not backlogged and channel status as seen by that station is idle,

then

Try to acquire channel

Generate next packet arrival request and set it in event list.

backlog completion event at any station:

check for channel status as seen by that station

If idle,

try to acquire the channel

otherwise,

continue sensing until channel becomes idle

Packet Completion event:

Set the channel idle events at upstream station and downstream station.

Update the statistics

Make the successfully transmitted station as current upstream station and to its next station as current downstream station.

(6) If (current simulation clock < Total simulation time)

Go to step 4.

else

Print the statistics.

5.3 MAJOR FUNCTIONS:

Major functions defined in the programs are listed as follows:

Initialize(): Initializes the network parameters and network configuration

randf(): Produces a floating point random number between (0,1).

randn(int n): Generates an integer randomly between 0 and n-1, which is a uniform random number.

rand(int lambda): Generates the exponential random number for the given arrival rate.

genelist(evlistnode x): Add the node x to the event list in ascending order of time and updates pointer to the event list, which is

global variable.

gen-arrival(int stnno): Returns the next arrival time for a station by adding the simulation clock and exponentially random distributed inter arrival time generated for the station stnno.

Acquire-channel(): When a station is ready to transmit, it tries to acquire the channel. It acquires channel and schedules packet completion event if there is no collision, otherwise it schedules backoff completion event in event list.

Detectcollision(int i): It checks whether station i transmits a packet before it knows that a packet is already in transit on channel.

Gen-backlog(int stn): For a station stn from its retransmission attempt(k), it calls the function randn(2^k-1). The random number generated is multiplied by slot time and added this to current simulation clock gives the backlog completion time. This function returns the computed backlog completion time.

Updt-stat(): This function finds the difference between packet completion time at the packet arrival time and calls the function bms() with this time. It increments the number of successfully transmitted packets.

Bms(double delay): It calculates the average packet delay.

Printstat(): This function is used to print the values of various statistics such as throughput and average packet delay.

RESULTS AND CONCLUSION

In this chapter, the sample results produced when running the simulation program are presented to show the performance of Hybrid Ethernet. Initially, the simulation of pure Ethernet is done and its values are validated against the analytical values produced by [6]. All the sample values obtained after simulating more than 3000 packets on an average per each station. The effects of transients on the average performance parameters is taken in to consideration by leaving the initial 3000 packets generated and the batch means analysis is used with a batch size of 300. It is assumed that there is no correlation among successive output samples while generating confidence intervals.

6.1. VALIDATION OF PURE ETHERNET

The following assumptions are used when simulating the pure ethernet

1. There are 50 nodes over the network.
2. Ether length is 10 km.
3. propagation speed is $2/3$ of speed of light
4. Channel speed is 10 Mbps.

5. Packet length 50,000 bits
6. Jamming signal size is 32 bits.

The performance parameter values obtained through simulation are compared with the analytical values. The graph drawn between throughput and offered load for the simulation results and analytical results is shown in Figure 6.1 and the values are given in table 6.1, From the figure it is clear that the performance obtained from the simulation results are better than those found by approximate analytical calculations.

6.2 SIMULATION OF HYBRID ETHERNET

Table 6.2 shows the results of the Hybrid Ethernet under different load conditions with 5 direct links and with 5%, 10%, 20% of load directly transmitted. The graph for these results is shown in figure 6.2. It also shows the results obtained for pure ethernet at different load conditions. From the figure, it is evident that with the Hybrid Ethernet, throughput achievable is more than one. And also, as the percentage of traffic transmitted through direct data links increases, the throughput increases.

The average packet delay under different load conditions for both Hybrid Ethernet and pure Ethernet is shown in Figure 6.3 and values are given in Table 6.3. Under heavy loads, the average time delay reduced

significantly for Hybrid Ethernet compared to the pure ethernet.

The performance of Hybrid Ethernet, by varying the number of data links and keeping the percentage of load directly transmitted as constant, is also studied. When 5% load is directly transmitted, the throughput values obtained for 1DL (i.e. one data link), 5DL, and 7DL are plotted in figure 6.4 and these are given in Table 6.4. The average packet delay is plotted in Figure 6.5 and values are given in say table 6.5. From the results obtained it is clear that as the number of direct links increases, the throughput increases and the average packet delay decreases.

6.3 CONCLUSION

The performance parameter values found at different load conditions, drawn the following conclusions about Hybrid Ethernet's performance.

1. Under light traffic load, the performance of Hybrid Ethernet is similar to pure Ethernet.
2. Under heavy loads, the performance of hybrid Ethernet greatly improved over pure Ethernet.
3. The maximum throughput achievable in hybrid Ethernet may exceed one.

4. Unlike pure Ethernet, performance does not degrade under heavy loads.

Thus, it can be stated that the performance of conventional Ethernet Local Area Network significantly improves with the insertion of direct data links.

Number of Stations = 50

Channel Speed = 10 Mbps

Packet Size = 50000 bits

Traffic Load	Throughput	
	Analytical	Pure ethernet simulation
0.05	0.019	0.018
0.1	0.098	0.095
0.3	0.27	0.29
0.5	0.413	0.46
0.8	0.548	0.57
1.0	0.6	0.62
3.0	0.76	0.739
5.0	0.819	0.774
8.0	0.867	0.815
10.0	0.886	0.835
40.0	0.937	0.93
50.0	0.936	0.925
60.0	0.932	0.917
70.0	0.927	0.91

TABLE 6.1

Number of Stations = 50

Channel Speed = 10 Mbps

Packet Size = 50000 bits

5 Direct Links

Traffic Load	Throughput			
	5% DT	10% DT	20% DT	Pure ethernet
0.05	0.077	0.077	0.078	0.018
0.1	0.095	0.136	0.141	0.095
0.3	0.29	0.29	0.3	0.29
0.5	0.55	0.55	0.57	0.46
0.8	0.72	0.72	0.73	0.57
1.0	0.78	0.78	0.79	0.62
5.0	1.08	1.08	1.08	0.774
8.0	1.10	1.12	1.21	0.815
10.0	1.19	1.19	1.33	0.835
40.0	1.75	1.86	2.05	0.93
50.0	1.93	1.936	2.2	0.925
60.0	2.1	2.1	2.3	0.917
70.0	2.11	2.20	2.57	0.91
80.0	2.13	2.28	3.08	0.91
90.0	2.18	2.34	3.4	0.907
100.0	2.28	2.422	3.5	0.903

TABLE 6.2

Number of Stations = 50

Channel Speed = 10 Mbps

Packet Size = 50000 bits

5 Direct Links

Traffic Load	Average Packet Delay			
	5% DT	10% DT	20% DT	Pure ethernet
0.5	0	0	0	0
3.0	0.05	0.05	0.05	0.14
5.0	0.1	0.1	0.1	0.2
10.0	0.24	0.235	0.22	0.48
20.0	0.52	0.51	0.46	1.17
30.0	0.79	0.75	0.67	1.62
40.0	1.1	1.08	0.85	1.96
50.0	1.4	1.28	1.03	2.6
60	1.5	1.33	1.21	3.5
80.6	1.85	1.56	1.33	8.3

TABLE 6.3.

Average Packet delay for different load conditions

Number of Stations = 50

Channel Speed = 10 Mbps

Packet Size = 50000 bits

5% of Traffic Directly Transmitted

Traffic Load	Through put			
	1% DL	5% DL	20% DL	Pure ethenet
0.1	0.1	0.095	0.1	0.095
1.0	0.6	0.78	0.8	0.62
5.0	0.7	1.08	1.1	0.774
10.0	0.77	1.19	1.2	0.835
20.0	0.9	1.38	1.43	0.89
30.0	0.99	1.63	1.8	0.92
50.0	1.08	1.93	2.16	0.925

TABLES 6.4

Throughput with 5% of load directly transmitted and different network configurations for different Traffic loads

Number of Stations = 50

Channel Speed = 10 Mbps

Traffic Load	Average Packet Delay			
	1% DL	5% DL	12% DL	Pure ethenet
1.0	0	0	0	0.071
3.0	0.06	0.05	0	0.14
5.0	0.16	0.1	0	0.2
10.0	0.41	0.24	0.06	0.48
20.0	0.79	0.52	0.16	1.17
30.0	1.08	0.79	0.2	1.62
40.0	1.78	1.1	0.24	1.96
50.0	2.2	1.4	0.33	2.6
80.0	3.9	1.85	0.56	8.3

TABLE 6.5

Average packet Delay with 5% load directly transmitted and different network configuration for different Traffic loads.

Throughput vs. Traffic load of pure ethernet

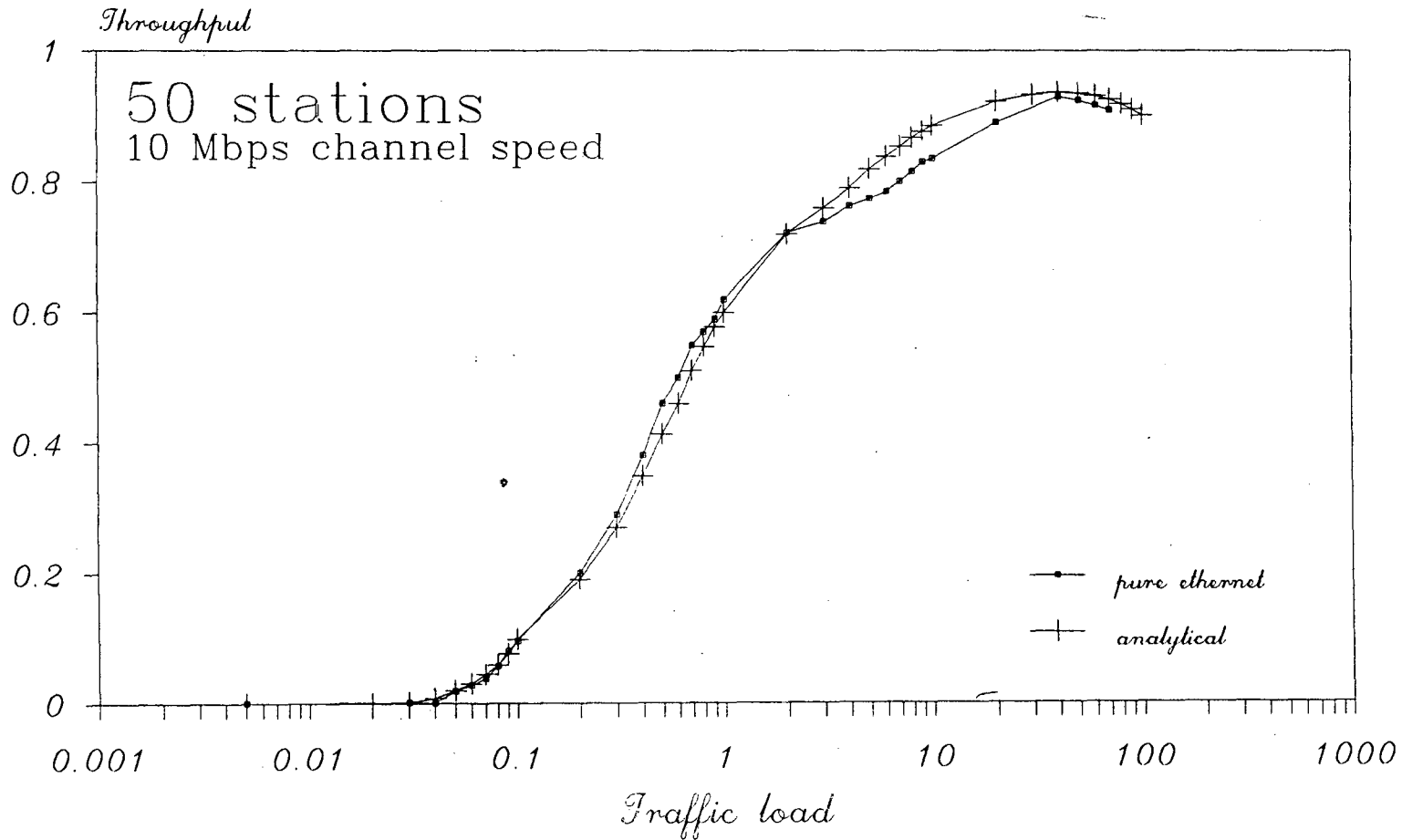


Fig. 6.1

BIBLIOGRAPHY

- [1] W. Stallings, "Local networks performance", IEEE communications Magazine, 22 (1984) 27-35.
- [2] W. Stallings, "Local Networks", Computing Surveys, 16: 3-41, March 1984.
- [3] W. Bux, "Performance issues in local area networks", IBM system Journal, 23 (1984) 351-374.
- [4] R.M. Metfalfe and D.R. Boggs, "Ethernet: distributed packet switching for local computer networks", Communications of ACM 19 (1976) 395-404.
- [5] BSTF.A. Tobaji and V.B. Hunt, "Performance analysis of carrier sense multiple access with collision detection", Computer networks 4(1980) 245-259.
- [6] K.Soharby, M.L. Molle, and A.N. Venetsanopoulos, "Comments on 'throughput analysis for persistent CSMA systems'", IEEE Trans. Commun. vol. COM-35, pp.240-243, Feb. 1987.
- [7] Gerd E.Keiser, "Local Area Networks", McGraw Hill International Editions, 1989.
- [8] Chouel-Shin Kang and James H. Herzog, "A Mesh Token Ring Hybrid Architecture LAN", ACM Computer Communications Review, Volume 18, No.4, August 1988, pp. 146-154.
- [9] Chouel-Shin Kang and James H. Herzog, "Hybrid Meshnet: A New Approach to Mesh LANs", Proc. of the IEEE 13th Conference on Local Computer Networks, Minneapolis, Minnesota, October, 1988, pp. 463-470.

- [10] Chouel-Shin Kang, " A New LAN Architecture for the 1990s", WOCON'89, Proc. of the World Conference on Information Processing/Communication, Seoul, Korea, June 1989, pp. 216-223.
- [11] Chouel-Shin Kang and E.K. Park, "A Performance Study of A Token Ring with Connected Data Link(s)", Proc. of ISMM International Conference on Computer Applications in Design, Simulation and Analysis, New Orleanse, Louisiana, March 1990, pp. 180-183.
- [12] Andrews S. Tanenbaum, "Computer Networks", Prentice Hall of India, 1988.
- [13] H. Kobayashi, "Modelling and Analysis", An Introduction of system Performance Evaluation Methodology", Addison-Wesley publishing company, 1978.
- [14] Francis Neelamkavil, "Computer simulating and Modelling", John Wiley and Sons, 1987.
- [15] Mc Dougall, "Simulating Computer Systems: Techniques and tools", MIT Press.
- [16] D.Bertasakas and R. Gallager," Data Networks", Prentice Hall, 1989.
- [17] Kernighan and Ritchie, "The C programming Language", Prentice Hall of India, 1990.
- [18] Tanenbaum, Langsam and Augestein, "Data Structures using C", Prentice Hall of India 1994.