Simulation of Packet Switched Data Networks

Thesis submitted in partial fulfilment of the requirements
for the degree of Master of Technology
in
Computer Science

By

Dinesh Pandey

Under the Guidance of

Dr Bimal Kumar Roy
ASU, Indian Statistical Institute, Calcutta.

Indian Statistical Institute
Calcutta 700035
July 1998

CERTIFICATE

This is to certify that the project "Simulation of Packet Switched Data Networks is an authentic record of work carried out by Dinesh Pandey under my supervision. This work fulfils the requirements for the award of M.Tech. degree in Computer Science.

Bimal Roy

Dr Bimal Kumar Roy
Applied Statistical Unit,
Indian Statistical Institute,
Calcutta 700035.

Date ....

N. Chakravarti
External Examiner

ACKNOWLEDGEMENTS

I am very thankful to Dr Bimal Kumar Roy for his invaluable guidance and support. His was patient with my half baked ideas and gave me ample freedom to operate. The past one year that I have spent in his association would be difficult to forget.

I would also like to thank Mr. Saumitra Kayal whose printer this report is printed on.

Dinesh Pandey
mtc9614
M. Tech. Student,
Indian Statistical Institute,
Calcutta 700035.

Date ....29/7/1998....

Index

1
Network and Simulation

1.1 Simulation

Simulation is a word which is in common use today. The Chambers concise 20the century dictionary defines it thus:

> '**simulate** *sim'ū-āte v.t.* to feign: to have or assume a false appearance of : to mimic… *-n.* **simulā'tion** feigning: mimicry: the making of working replicas or representations of machines or the re-creation of a situation, environment, etc. for demonstration or for analysis of problems.-…'

For engineers involved in design and analysis of new systems the term simulation describes a wealth of varied and useful techniques, all connected with the mimicking of the rules of a *model* of some kind.

1.2 What is a model?

Often the processes in the real world are far too complicated to understand. In such cases it is good idea to strip the processes of some of their features, to leave us with the *models* of the original processes. If we can understand the model, then that may provide us with some insight into the process itself.

Subjects such as physics, biology, chemistry and economics use models to formulate and solve their problems. Differential equations governing the laws of mechanics can be viewed as models. For a model to work, it need not be an exact description of the reality. The famous Gallilean formulation of relativity is a model, and an inaccurate one, as Einstein pointed out later . Yet, from guiding space probes to analysing air-crashes it is Gallilean relativity which is used. Similarly the laws of economics are built upon some model of human behaviour (profit motive etc.), though actual human behaviour may be far more complex than that.

When the *model* is given a mathematical formulation and an analytical solution is not possible, quite often simulation can prove to be an useful tool. However, there are situations when it may be difficult to formulate the *model* in terms of known mathematical constructs. In such cases simulation simulation remains the only way out.

Sadiku and Ilyas [3] quote some other reasons why simulation is an attractive option:

1. It is the next best thing to observing a real system in operation.
2. It enables the analysis of very complicated systems. A system can be so complex that its description by a mathematical model is beyond the capabilities of the analyst. "When all else fails" is a common slogan for many such simulations.
3. It is straight forward and easy to understand and apply. It does not rely heavily on mathematical abstractions that require an expert to understand and apply. It can be employed by many more individuals.

4. It is useful in experimenting with new or proposed designs prior to implementation. Once constructed, it may be used to analyse the system under different conditions. Simulation can also be used in asessing and improving an existing system.

5. It is useful in verifying or reinforcing analytic solutions.

1.3 Examples of uses of simulation

(a) weather prediction

The actual physical weather phenomena defy comprehension. Still, the weather scientists build their models by stripping off all the details which may be unnecessary as far as weather prediction is concerned. Their success in this approach needs no mention.

(b) aeroplane design

Model aeroplanes in wind tunnels are used by aircraft designers to study the aerodynamics of full-scale aeroplanes.

(c) electrical circuit simulation

Electronic circuit designers use computer simulations extensively to study the behaviour of yet to be designed circuit. Many software packages are available in the market to aid them in this job.

(d) statistics

Even mathematicians use simulation. Not only may we use to mimic explicitly the behaviour of models, but we can also use simulation to evaluate the behaviour of complicated random variables whose precise distribution we are unable to evaluate mathematically. An early example (from [2]) of this method is the study of t-distributions by W. S. Gosset who published his works under the pen name of 'Student'. In 1908 he wrote:

> 'Before I had succeeded in solving my problem analytically, I had endeavoured to do so empirically [i.e. by simulation]. The material used was a... table containing the height and left middle finger measurements of 3000 criminals .... The measurements were written on 3000 pieces of cardboard, which were then very thoroughly shuffled and drawn at random ... each consecutive set of 4 was taken as a sample ... and the mean and standard deviation of each sample determined ... This provides us with two sets of ...750 z's on which to test the theoretical results arrived at. The height and left middle finger ... table was chosen because the distribution of both was approximately normal...'

1.4 Characteristics of a simulation model

(a) Continuous/Discrete Model

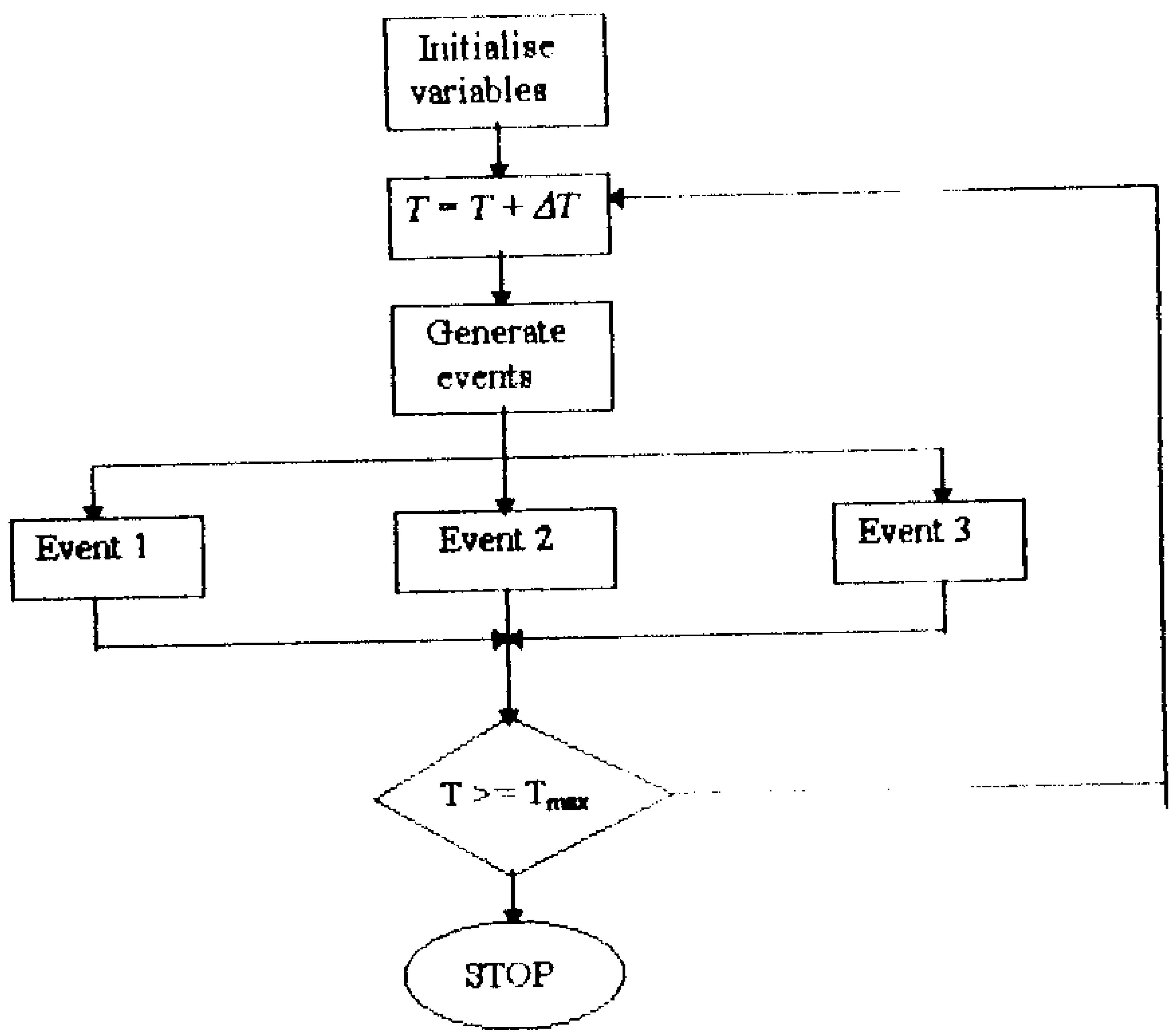
A continuous model is one in which the state variables change continuously with time The model is characterised by smooth changes in the system. Weather prediction models are continuous. A discrete model is characterised by discontinuous changes in the system. The arrival process of messages in a network is discrete since the state variable, the number of waiting messages, changes only at the arrival or departure of a message.

(b) Deterministic/Stochastic Models

This characteristics deals with the system response. A system is deterministic if its response is completely determined by its initial state and input. It is stochastic if the system response may assume a range of values for a given initial state and input. Thus, only the statistical averages of the output measures of a stochastic model are true characteristics of a system. The simulation of computer networks falls under the second category because both the inter-arrival and service time are random.

(c) Time/Event Based-Models

Since simulation is the dynamic portrayal of the states of a system over time, a simulation model must be driven by an internal clock. In time-based simulations the simulation clock advances one "tick" of Δt. Figure 1.1 shows the flowchart of a typical time-based simulation model.



1.2 Typical time-based simulation model.

Although time-based simulation is simple, it is inefficient because some action must take place at each clock "tick". An event signifies a change in the system. In an event-based simulation model, updating takes place only at the occurrence of an event, and the simulation clock is advanced by the amount of time since the last event. The need to determine which event is next in event-based simulation makes its programming complex. A problem of this approach is that the speed at which the simulation proceeds is not directly related to real-time; correspondence to real-time operation is lost. Figure 1.2 is the flowchart of a typical event-based simulation.

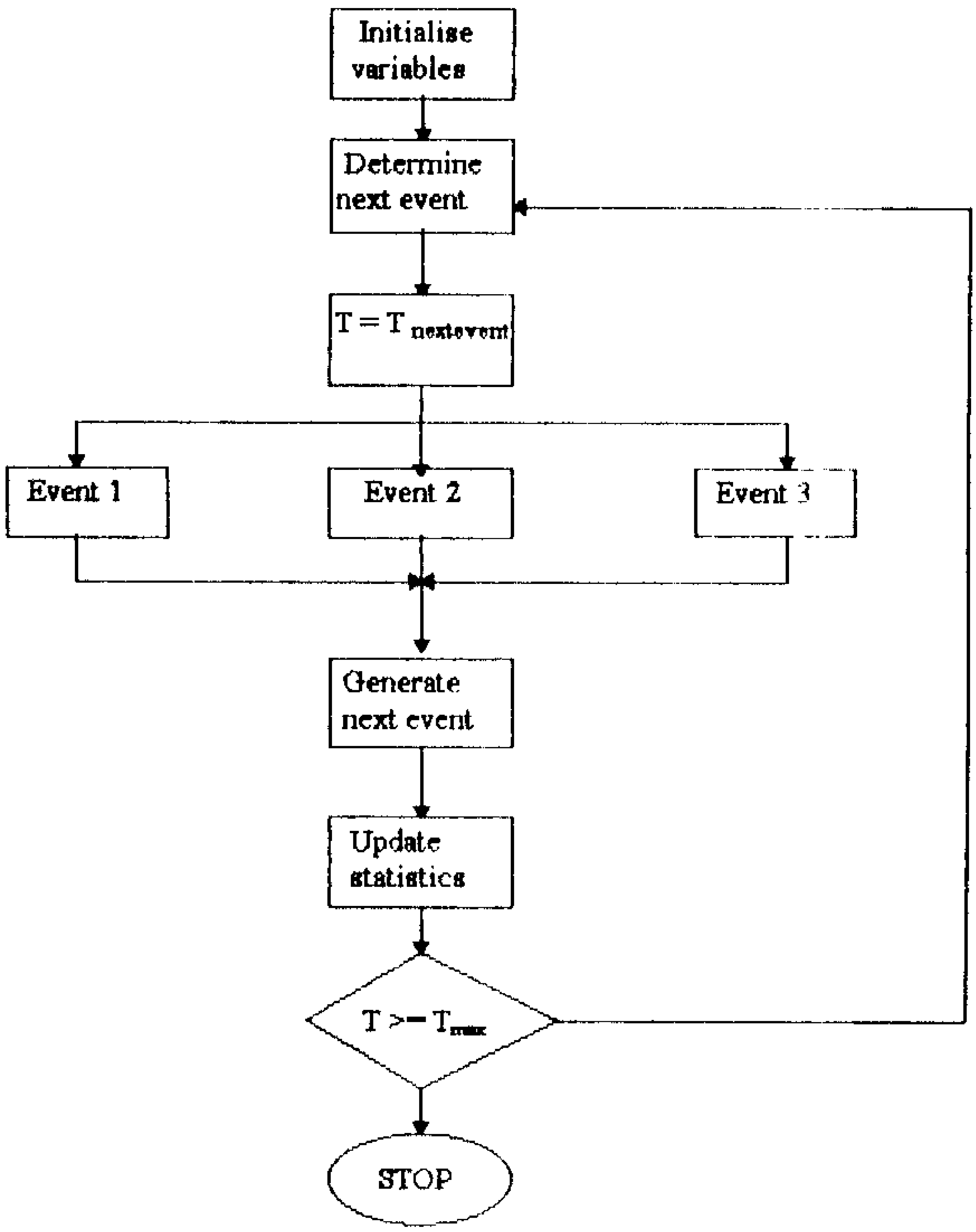


Figure 2.2 Typical event based simulation model

1.5 Why simulate a computer network?

The problems of computer network designer are many. He has to make essentially inanimate and unintelligent machines co-operate amongst themselves for exchange of information. The conditions under which the network will operate is at best an intelligent guess. The complexity of a network and the underlying protocol makes analysis difficult to use. Simulation, however, gives us an ideal tool for this kind of situation.

In this project two types of networks have been studied. The first one, the ALOHA net is a simple broadcast network that was once used for connecting many computer users to a central computer. The second is the ubiquitous Ethernet. This logical successor of ALOHA was developed by Xerox PARC and is almost exclusively used for low cost local area networks.

2
The ALOHA Protocol

2.1 The Channel Allocation Problem

When many users are using the same medium to communicate the problem of channel allocation comes up. It is like many people talking amongst themselves. They use the same channel (i.e. the surrounding air and voice signals). If everyone tried to talk at the same time no one will be able to communicate. The common social etiquette helps us in resolving this problem and even large groups are able to communicate purposefully. Computers, however have no etiquette and it is the network designers problem to devise a protocol to give them at least a semblance of civility. Though there are many workable solutions the ideal protocol still remains elusive.

2.2 Origin of ALOHA

In the 1970s, Norman Abramson and his colleagues at the University of Hawaii devised a new and elegant method to solve the channel allocation problem. The aim there was to provide cheap and easy access for a large number of terminal users to central computing facilities. Although Abramson's work, called the ALOHA system used ground based radio broadcasting, the basic idea is applicable to any system in which uncoordinated users are competing for the use of a single channel. In this project a satellite communication system using the ALOHA protocol is simulated.

2.3 The Protocol

The basic idea of an ALOHA system is simple: let users transmit whenever they have data to be sent (Figure 2.1). There will be collisions, of course,

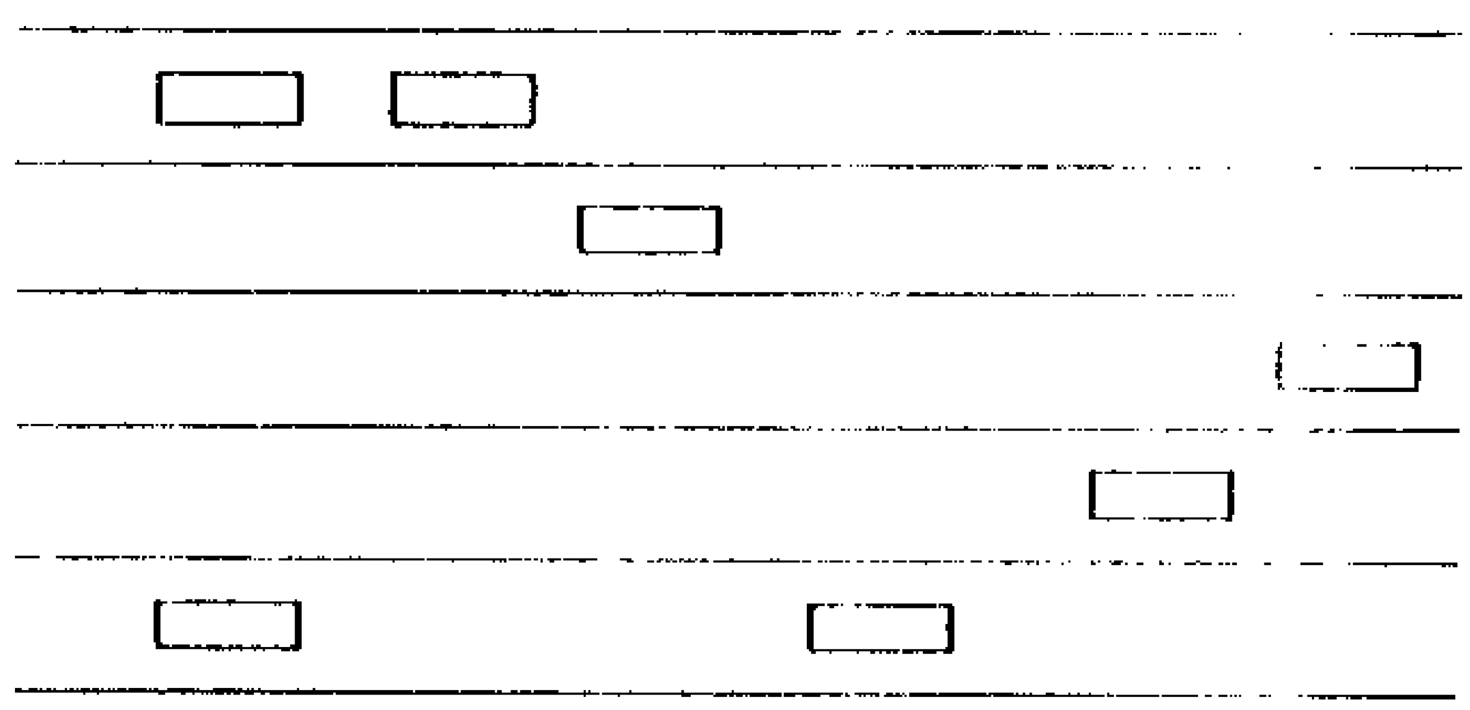


Figure 2.1 In pure ALOHA frames are transmitted at completely random times

and the colliding packets will be destroyed. However, due to the feedback property (stations are always listening to the packets being transmitted) a sender can always find out whether or not his packet was destroyed. In a local area network the feedback is immediate, with a satellite, there is a delay of 270 msec before the sender knows if the transmission is successful. If the packet was destroyed, the sender just waits a random amount of time and starts transmission again. The waiting time must be random or the same packets will collide over and over, in lockstep. This protocol is also sometimes referred to as pure ALOHA because many researchers have extended Abramson's work and have developed different versions of this system.

2.4 The Simulation

The simulation was done using an event-based model, which is ideally suited for this type of systems. The program is self driven and need no user input during the course of its execution. The generation of new packets at the transmitting stations is a Poisson process which is simulated using the built in function `rand()`. This function generates uniformly distributed numbers, which are then converted to follow the desired probability distribution. In this case we primarily need exponential distribution to simulate the random inter-arrival time for the Poisson process

The generation rates are same at all the stations and all the packets have the same size. The variables being monitored are delay, channel traffic, collision rate and throughput. Figure 2.2 (on the next page)gives a detailed flowchart of the simulation.

Due to the inherent complexity of the problem some degree of object orientation is used in the simulation. Each station is an object capable of maintaining itself (its queues, event times etc.). The satellite and the intervening space is modelled as another object. The satellite object does more than what a satellite is supposed to do (a communication satellite simply acts as a reflector or a repeater). When a station transmits (in the simulation) it effectively deposits the ID of the packet to the satellite object. Since many stations will be transmitting the satellite object is in the best position to know when a collision occurs.

Five events have been used to simulate the model.

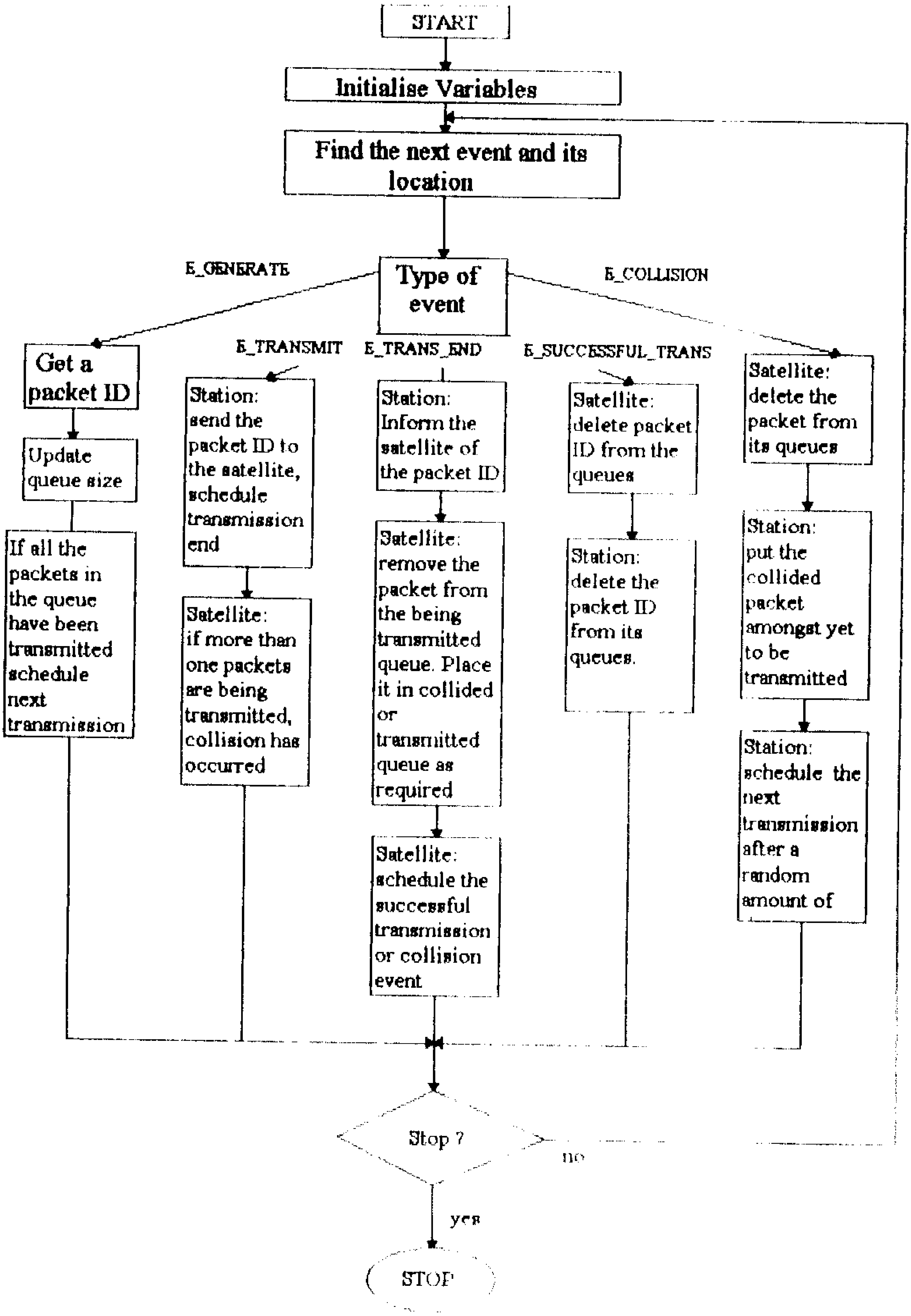
1. E_GENERATE event for a station to generate a packet.

2. E_TRANSMIT event for a station to transmit the next queued packet.

3. E_TRANS_END the station has finished transmitting the packet.

4. E_SUCCESSFUL_TRANS at this event the satellite informs a station that a packet sent by it has been successfully transmitted. This event is scheduled exactly 270ms after E_TRANS_END for the packet.

5. E_COLLISION event for the satellite to inform a station about a collided packet. This event is also scheduled 270ms after E_TRANS_END. At this event the station delays its next transmission by a random amount of time (subject to a maximum).


START
Initialise Variables
Find the next event and its location
Type of event
E_GENERATE
E_TRANSMIT
E_TRANS_END
E_SUCCESSFUL_TRANS
E_COLLISION
Get a packet ID
Update queue size
If all the packets in the queue have been transmitted schedule next transmission
Station: send the packet ID to the satellite, schedule transmission end
Satellite: if more than one packets are being transmitted, collision has occurred
Station: Inform the satellite of the packet ID
Satellite: remove the packet from the being transmitted queue. Place it in collided or transmitted queue as required
Satellite: schedule the successful transmission or collision event
Satellite: delete packet ID from the queues
Station: delete the packet ID from its queues.
Satellite: delete the packet from its queues
Station: put the collided packet amongst yet to be transmitted
Station: schedule the next transmission after a random amount of
Stop ?
no
yes
STOP

3

The Ethernet

3.1 History

Ethernet has an interesting history. The real beginning was the ALOHA system constructed to allow radio communication between machines scattered over the Hawaiian islands. Later carrier sensing was added, and in 1976 Xerox PARC built a 2.94-Mbps system to connect over 100 personal workstations over a 1-km cable. This system was called Ethernet after the *luminiferous ether*, through which electromagnetic radiation was once thought to propagate. The Xerox Ethernet was so successful that Xerox, DEC, and Intel drew up a standard for 10-Mbps Ethernet. Later IEEE standardised the system (IEEE 802.3). Even now Ethernet remains the most popular networking product and is best suited for setting up small low-cost networks.

3.2 The Protocol

The protocol followed by Ethernet is called CSMA/CD : Carrier Sense Multiple Access with Collision Detection. In this technique, all stations attached to the network make their transmission decisions on the basis of the status of the transmission medium. This differs from ALOHA in the sense that in ALOHA no station is aware of the other's intention until 270 milliseconds have elapsed. Since Ethernet is used exclusively in Local Area Networks, it is possible to monitor the current activity on the transmission medium Of course, there is a non-zero transmission delay but that is very small compared to ALOHA.

All stations on the Ethernet make their decisions based on the activity on the transmission medium as seen from their interface. The transmission initiated by various stations may overlap and may cause a collision. This necessitates retransmission of all collided packets. The next section describes the CSMA/CD protocol in some detail.

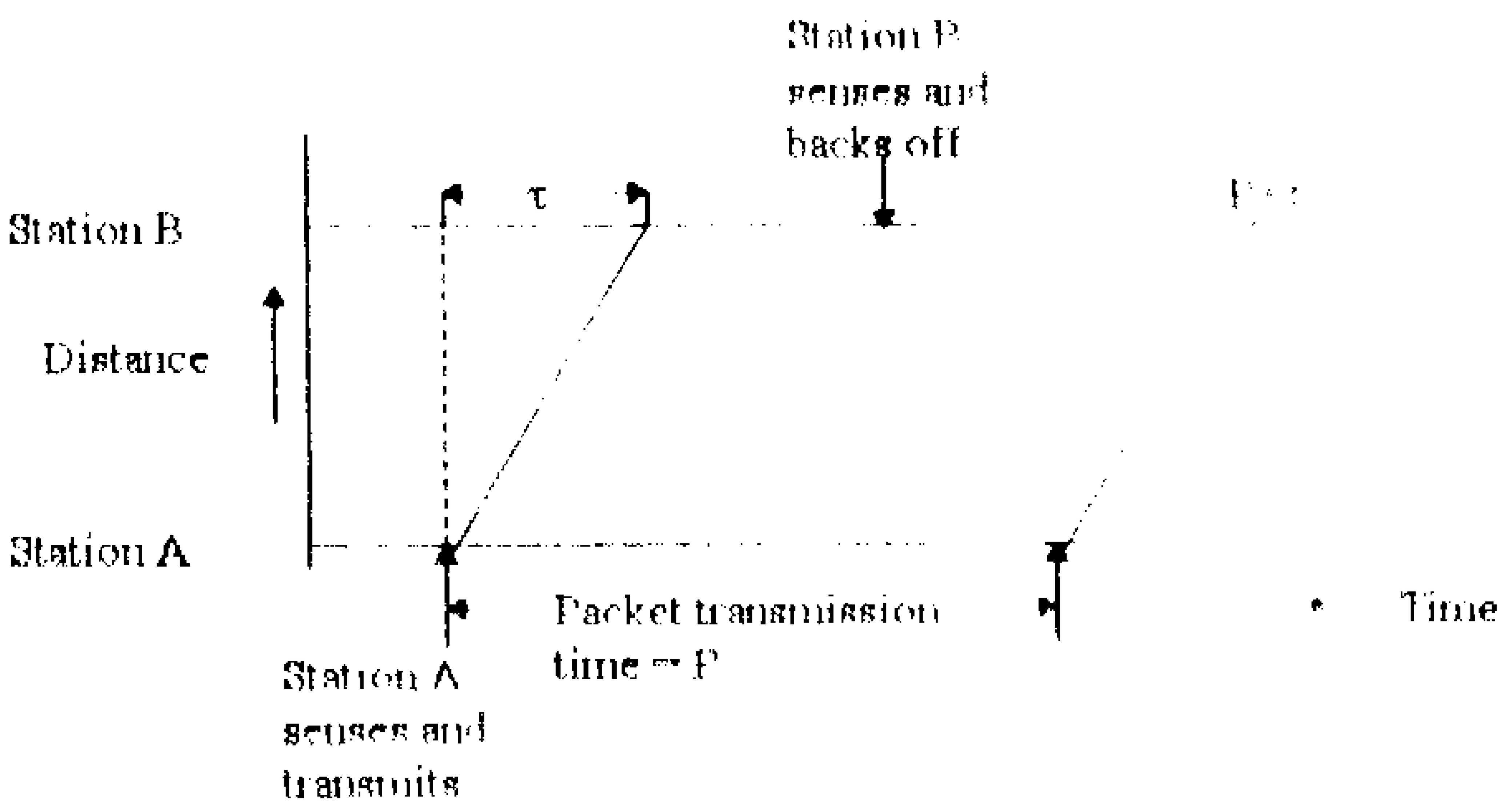
3.3 CSMA/CD

In CSMA/CD based local-area networks, the transmission medium is open ended. All stations are attached to the transmission medium using a passive interface. Access to the transmission medium is decided solely by the stations that are attempting to transmit. Before transmitting its packet of information, a station senses the state of the transmission medium to see if it is already busy (in transporting information packets) or idle. If a station finds the medium busy at its interface, then the transmission of its information packet must be delayed. However, if the medium is sensed free at the interface, the transmission of the information packet may proceed.

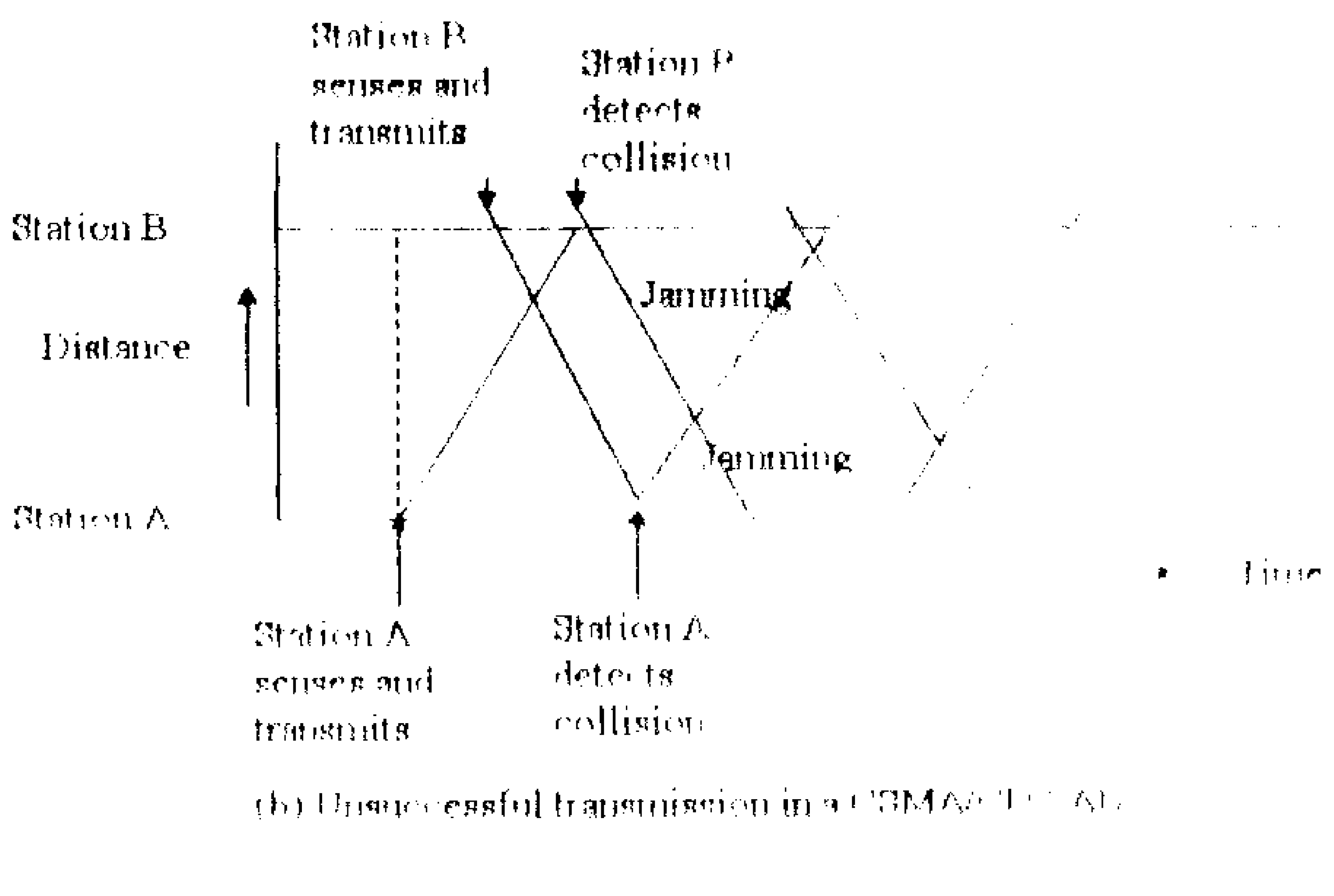
Although each station transmits only when it senses the transmission as free, a collision still may take place. This is because a transmission decision is made only on the basis of the local information not on the basis of the overall situation on the transmission medium. When a station transmits its packet it takes a small but finite amount of time before the information reaches all the stations. If two stations sense the medium as free and start transmission the packets will definitely collide within a time equal to the end to end propagation delay.

During transmission of their packets, all stations monitor transmitted information packets on the transmission medium. The information is compared to what the station is transmitting. If these two match then it is assumed that the transmission is successful. However, if the transmitted information differs from what is on the medium, it is assumed that a collision has taken place and a retransmission is needed.

In CSMA/CD-based LANs, when a collision has been detected , all transmitting stations abort their transmission. The station that first detects a collision starts transmitting a collision enforcement signal also known as *jamming signal*. This is to inform all the stations that a collision has taken place and they should wait until after the jamming signal is over and a predefined silence period has elapsed. After the silence period, business starts as usual. A typical situation is shown in figure 3.1



(a) Successful transmission in CSMA/CD LAN



(b) Unsuccessful transmission in a CSMA/CD LAN

3.4 Non-persistent and p-persistent CSMA/CD

The decision about transmission of a packet is made solely by a station and depends upon the status of the transmission medium as seen by the point of interface. The decision may vary slightly depending upon which version of CSMA/CD is being used even if the status of the transmission medium is the same.

Let us assume a station has a packet to transmit. It senses the transmission medium at its interface and finds it free. In the case of non-persistent CSMA/CD, the station will definitely transmit its packet. However, in the case of p-persistent CSMA/CD, the packet is transmitted with a probability p, and the transmission is delayed by τ seconds (end-to-end propagation delay) with a probability $(1 - p)$. If p happens to be equal to 1, as happens to be the case in most Ethernet implementations, the packet will also be transmitted immediately after the transmission medium is sensed free.

On the other hand, if the transmission medium is sensed busy, no packet should be transmitted. In the case on non-persistent CSMA/CD, the station backs off and senses the transmission medium again after a random duration of time. In the case of p-persistent CSMA/CD, the station keeps on checking the medium until it becomes free. As soon as the medium becomes free, the station transmits its packet with a probability p and delays it with a probability $(1 - p)$. Obviously, if $p = 1$ in the p-persistent case the station will immediately transmit after the medium becomes free. In this case, if more than one stations was checking the transmission medium at the same time, all of them will transmit almost at the same time and will collide with a probability 1.

3.5 The Simulation

The simulation program is written in C (code is given in appendix B). As in the case of ALOHA the inter-arrival time between two packets follows an exponential distribution. The built-in function `rand()` is used to generate uniformly distributed random numbers, which is then converted to exponentially distributed numbers by the transformation

$$y = -log(x)/\lambda$$

where x is follows the distribution U(0,1), λ is the arrival rate. Then, y will follow an exponential distribution.

The flowchart of the simulation is presented in Figure 3.2. It follows the general principles of an event based model.

3.6 Assumptions

The following assumptions have been made in the simulation process of the CSMA/CD local area networks:

- Arrival at all stations follow a Poisson process.
- All stations generate traffic at the same rate

- Packet lengths are equal.
- The transmission medium is assumed to be error free, and any errors are only due to collisions.
- The spacing between stations is the same.
- The propagation delay is about 5 microsecond per kilometre of the transmission medium.

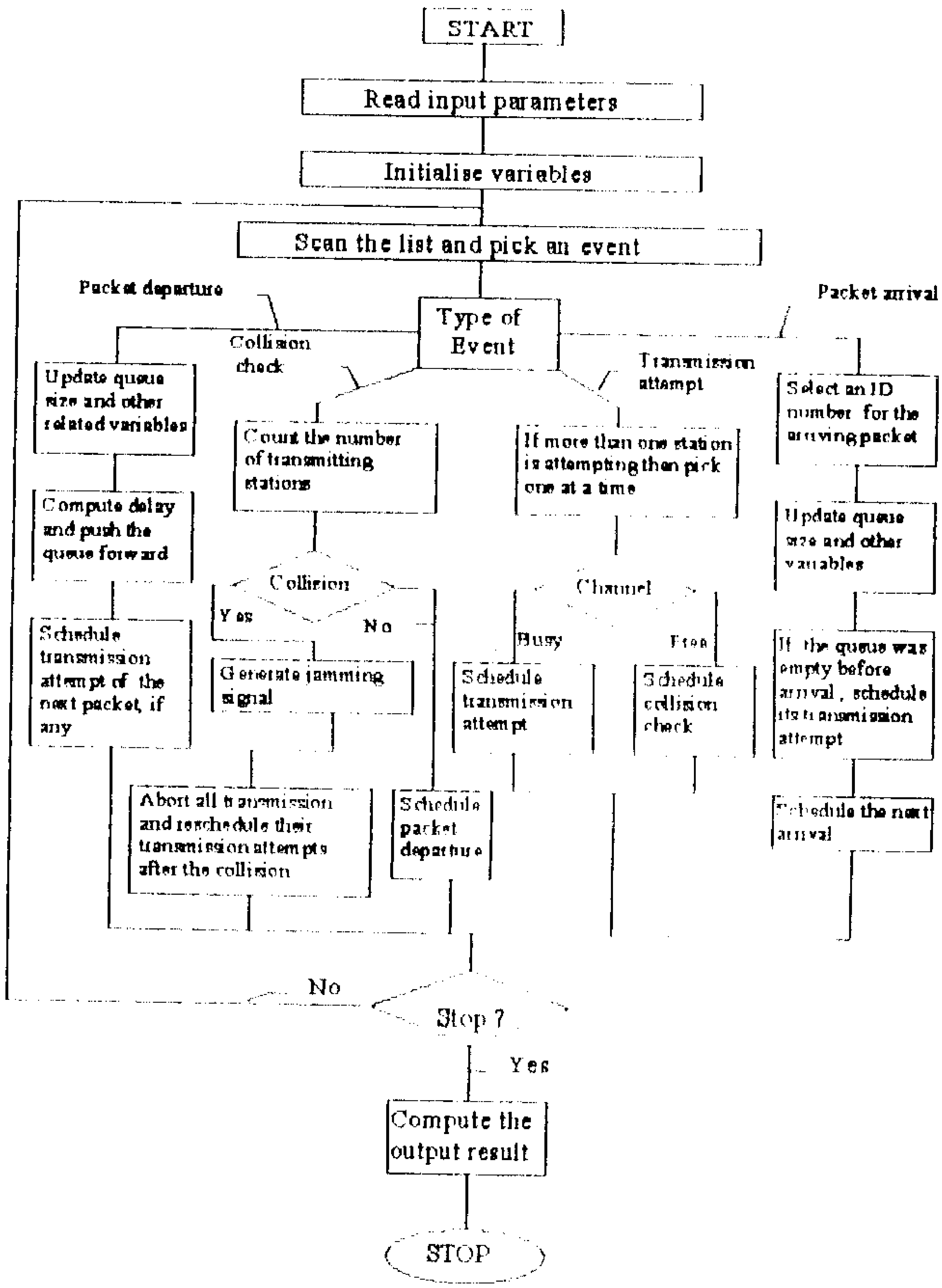


Figure 3.2 Flow chart of the CSMA/CD simulation program.

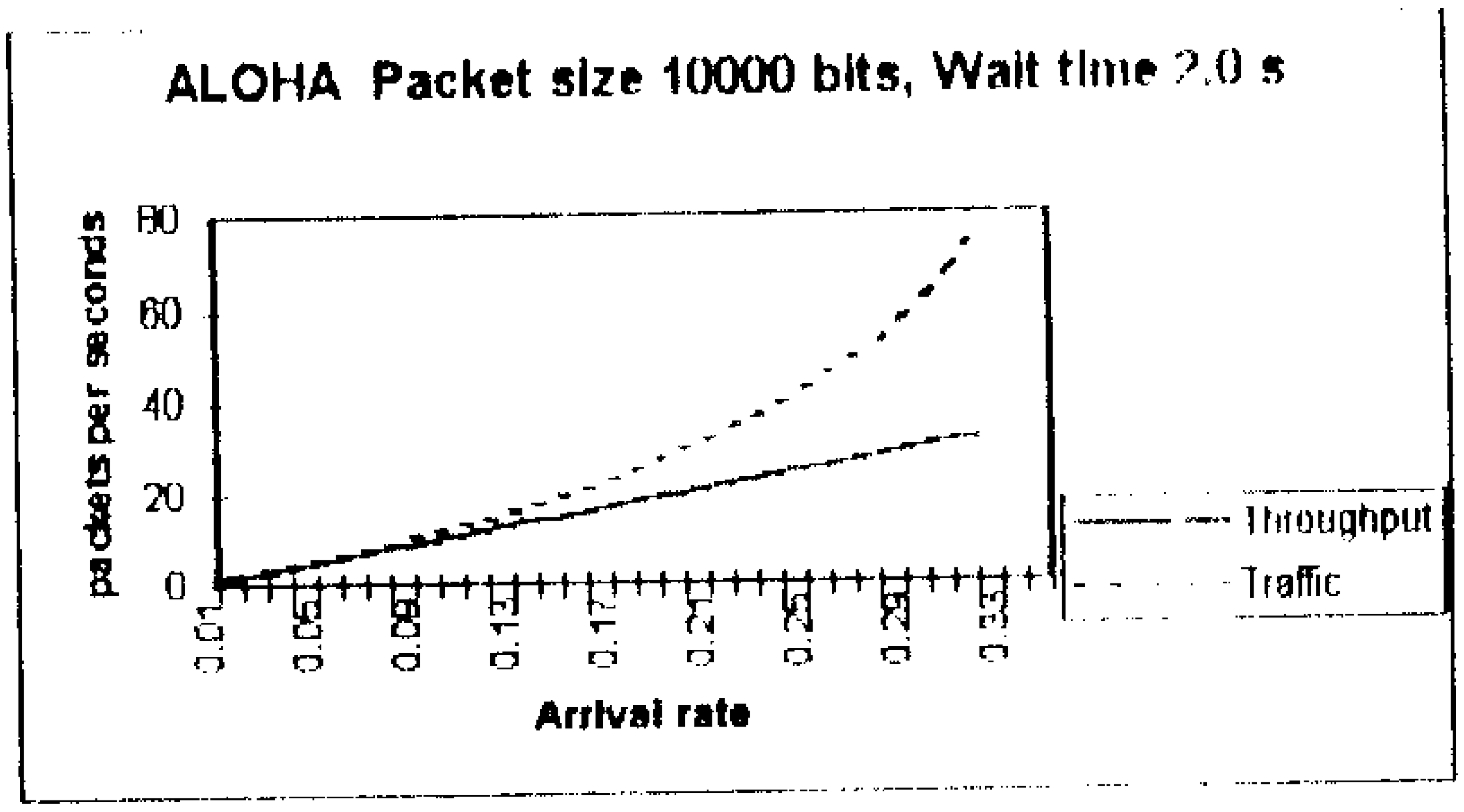
4
Results

4.1 The ALOHA Simulation

The simulation program was run on a Pentium 133 machine running Linux. For all the runs some of the parameters were kept constant. The transmission rate was kept fixed at 1.5 MHz. The number of earth stations was 100.

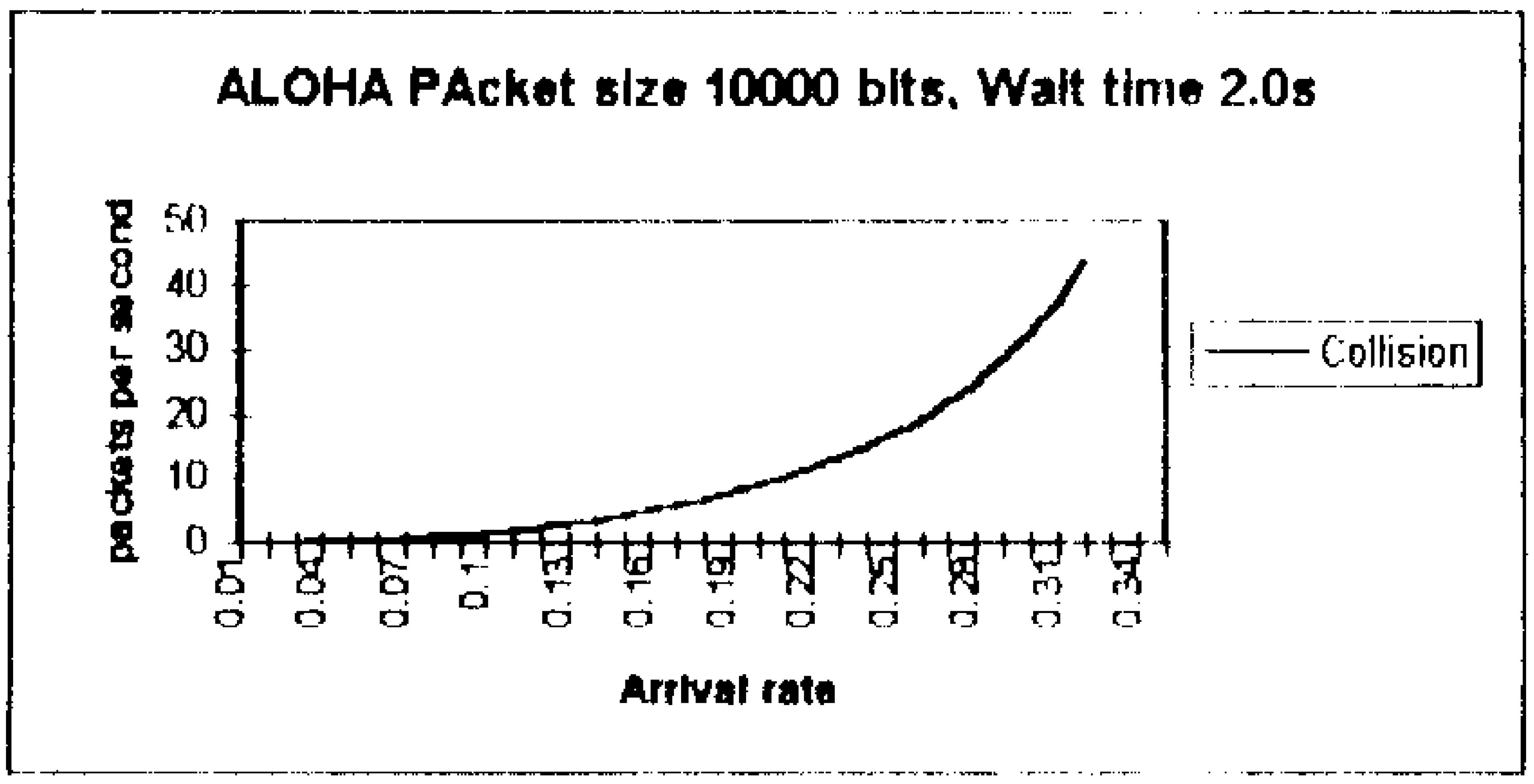
The performance measures that were monitored were *throughput*, *delay*, *traffic*, and the *collision rate*. *Collision rate* is the number of packets which collided from amongst those that were transmitted. The plots are for *throughput*, *traffic*, *delay*, and *collision rate*.

The first plot gives the variation of utilisation and traffic with change in packet size. The arrival rate is kept fixed at 0.01 packets / second at each of the 100 transmitting stations. As we can see here (and in most of the graphs that follow) the throughput is a linear function of arrival rate (up-to a certain point) and is in fact equal to the applied load (which is 100 times the arrival rate). This is very important for the long term stability of the system.

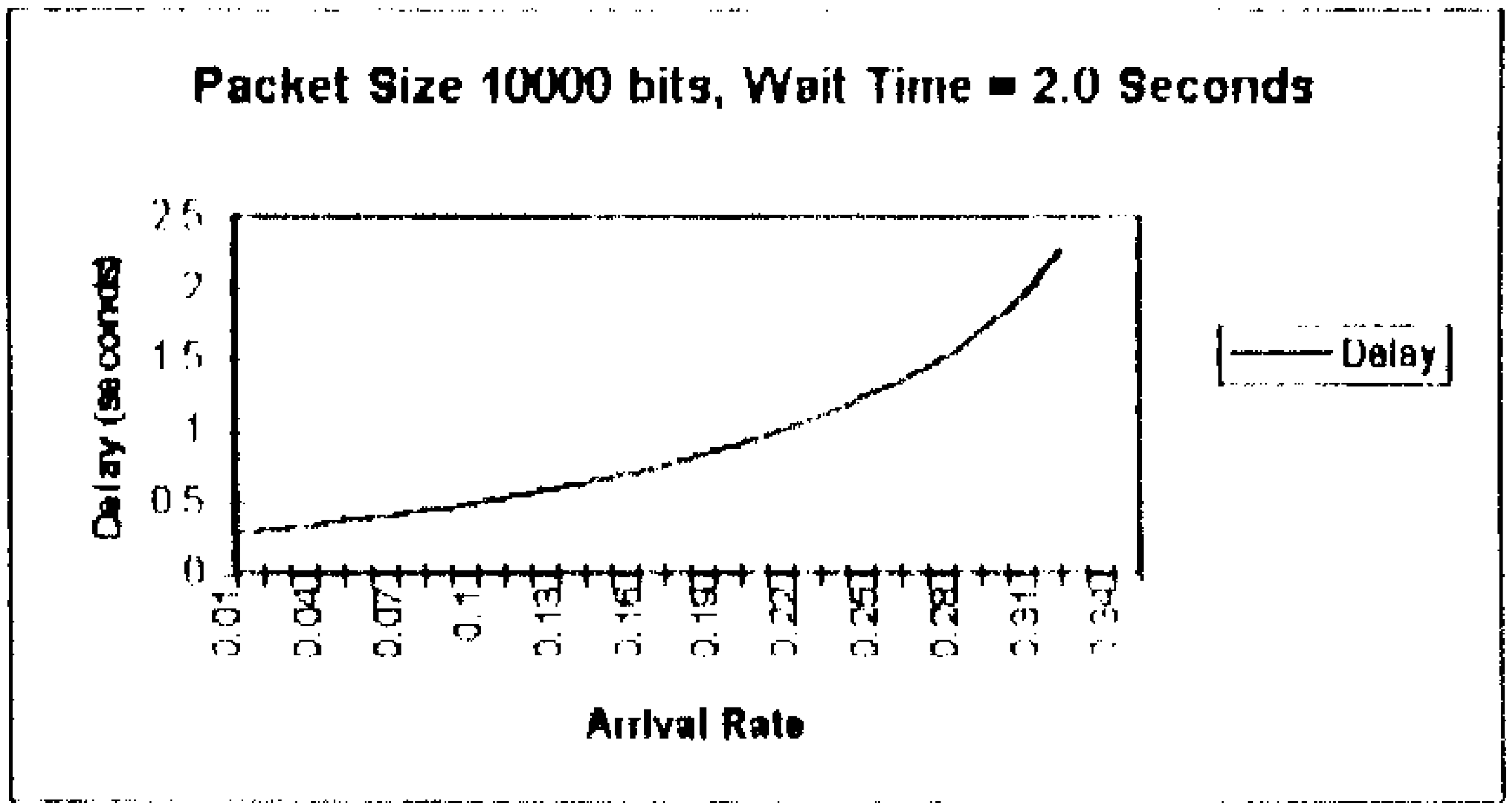


Wait-time is the maximum time for which the station waits after one of its packets collides. We can also see from the graph that the traffic increases faster than throughput This is because many of the packets that are transmitted end up colliding. Such packets have to be retransmitted. At a certain point the arrival rate becomes more than the capacity of the system to transmit. This is the point where the system becomes unstable, i.e., it is not able to transmit as fast as the packets are generated. For this case the system becomes unstable at an arrival rate of 0.33 packets per second per station.

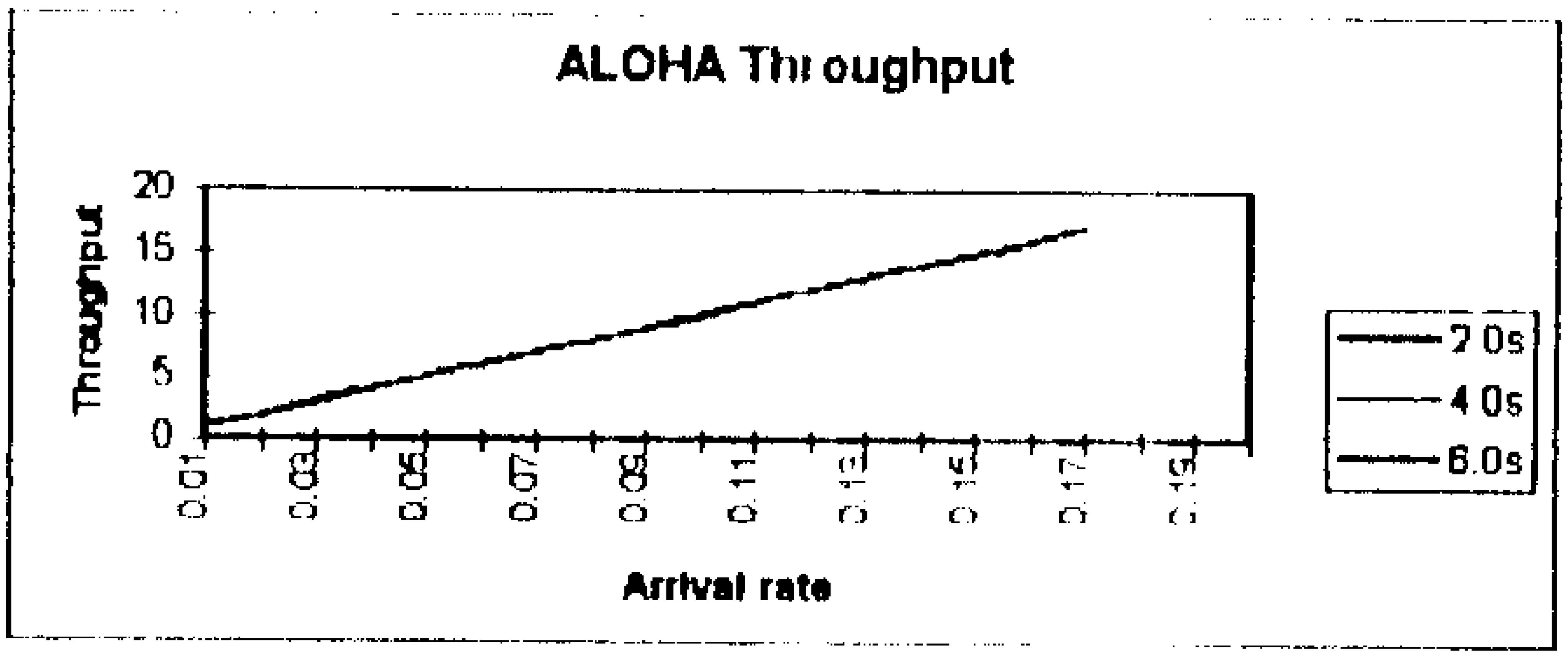
Collision rate is simply the difference between the traffic and the throughput.



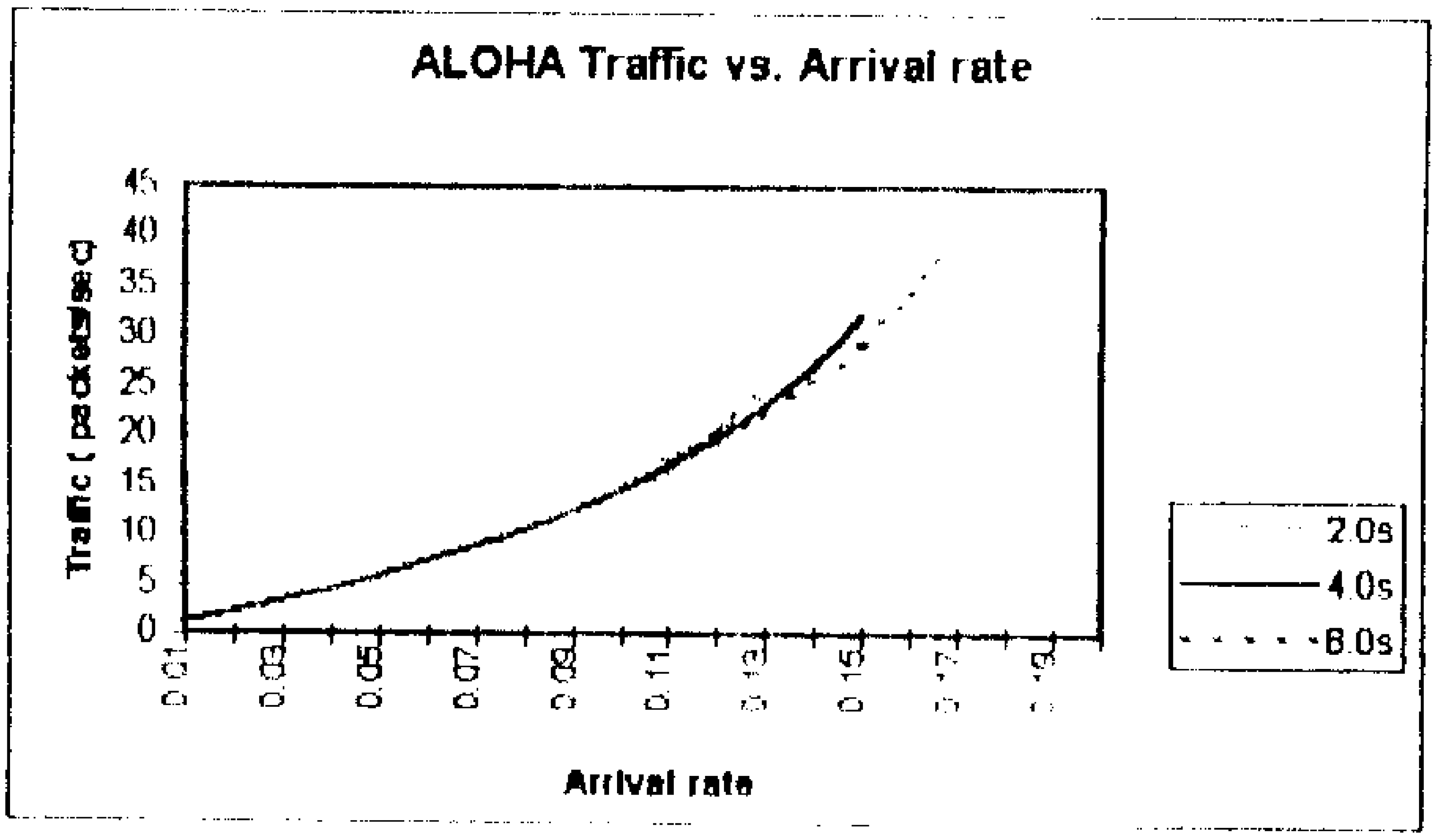
Next we look at how delay varies with arrival rate:



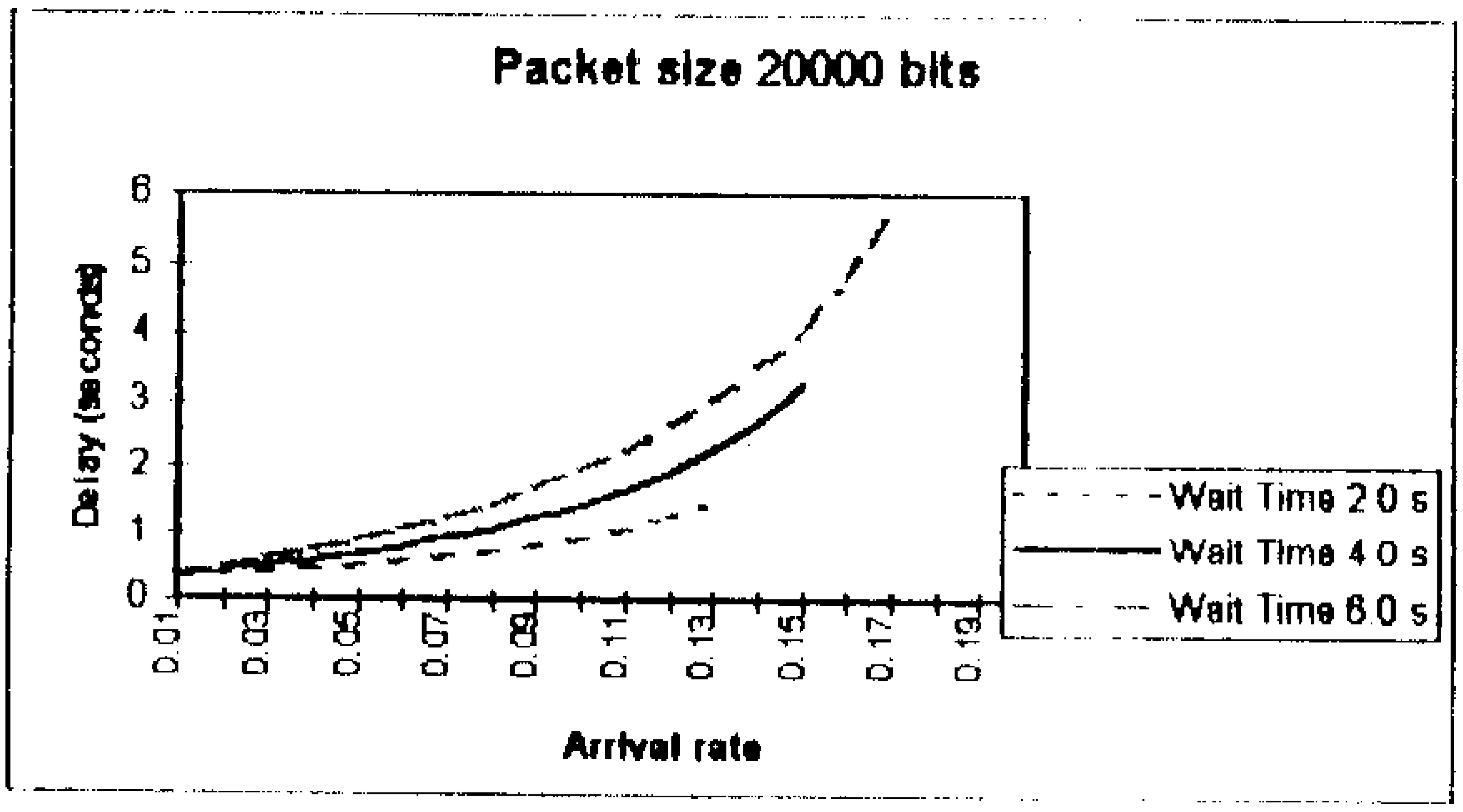
To see how the performance parameters vary with changes in wait-time, the simulation was run with packet size = 20000 bits, varying the arrival rate for three different values of wait-time (2.0s, 4.0s, 6.0s) Increasing the wait-time increases the capacity of the system to handle load but this is at the cost of delay. On the next page are the comparative plots of throughput, delay, and collision rate for these three cases



This plot is not very instructive because all the three curves are coincident. The 2.0s curve extends up to an arrival rate of 0.13, 4.0s extends up to 0.15 and 6.0s curve extends up to 0.17. The arrival rate vs. traffic makes the situation clearer.

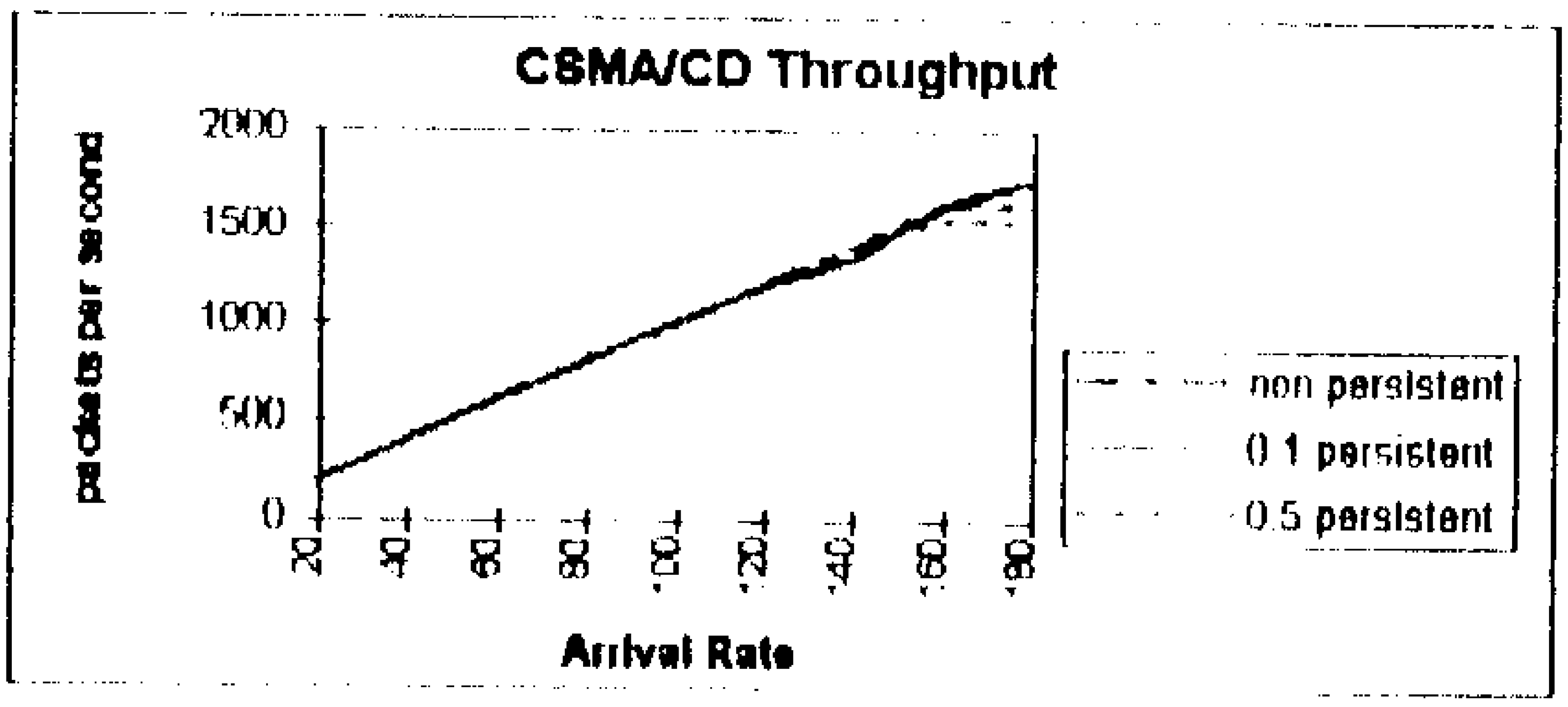


The difference between the three cases is more pronounced in this case. As we can see, for lower wait-time the traffic increases faster. The distinction becomes further clear when we look at the delay values for the three cases (next page).

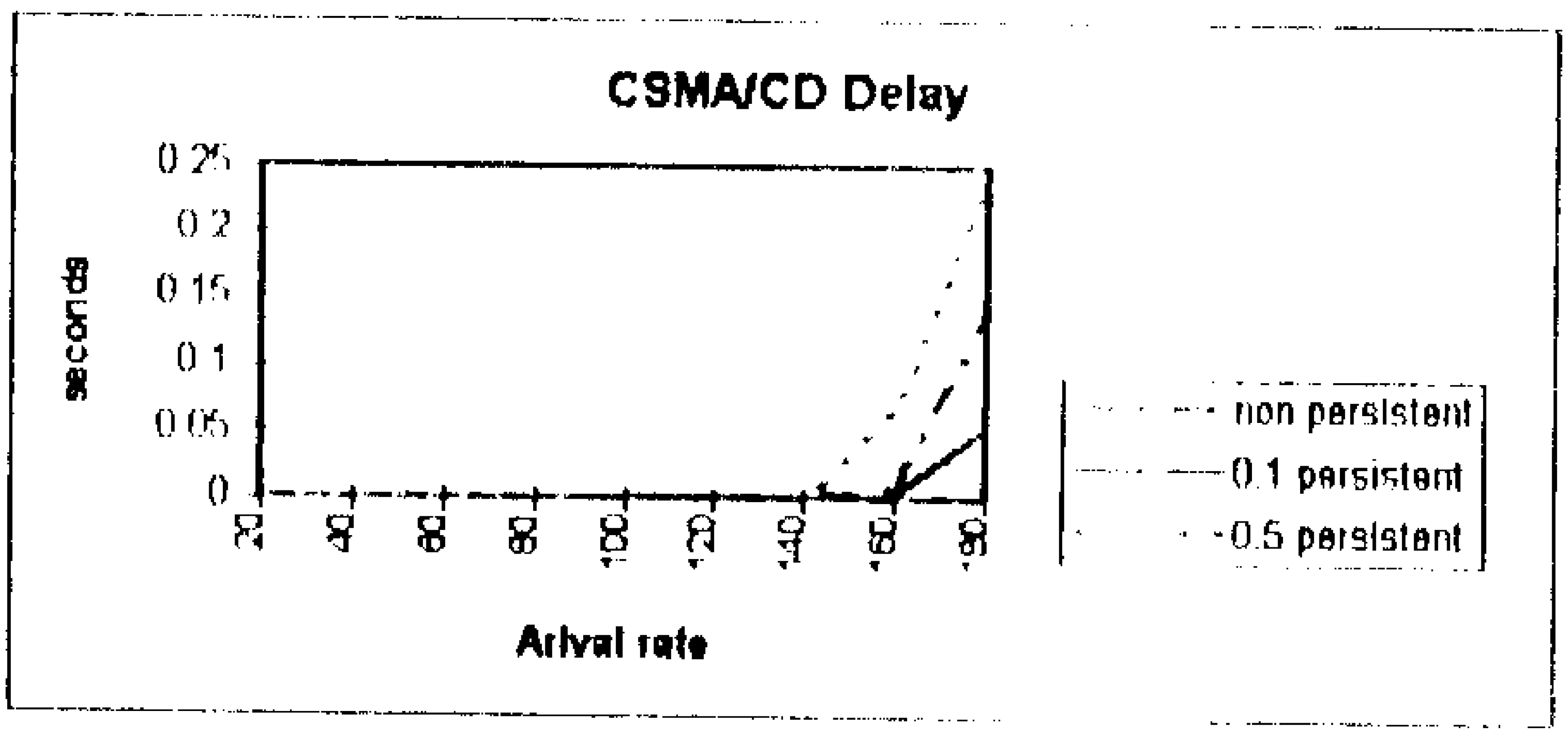

Packet size 20000 bits
Delay (seconds)
0
1
2
3
4
5
6
0.01
0.03
0.05
0.07
0.09
0.11
0.13
0.15
0.17
0.19
Arrival rate
Wait Time 2.0 s
Wait Time 4.0 s
Wait Time 6.0 s

4.2 The Ethernet Simulation

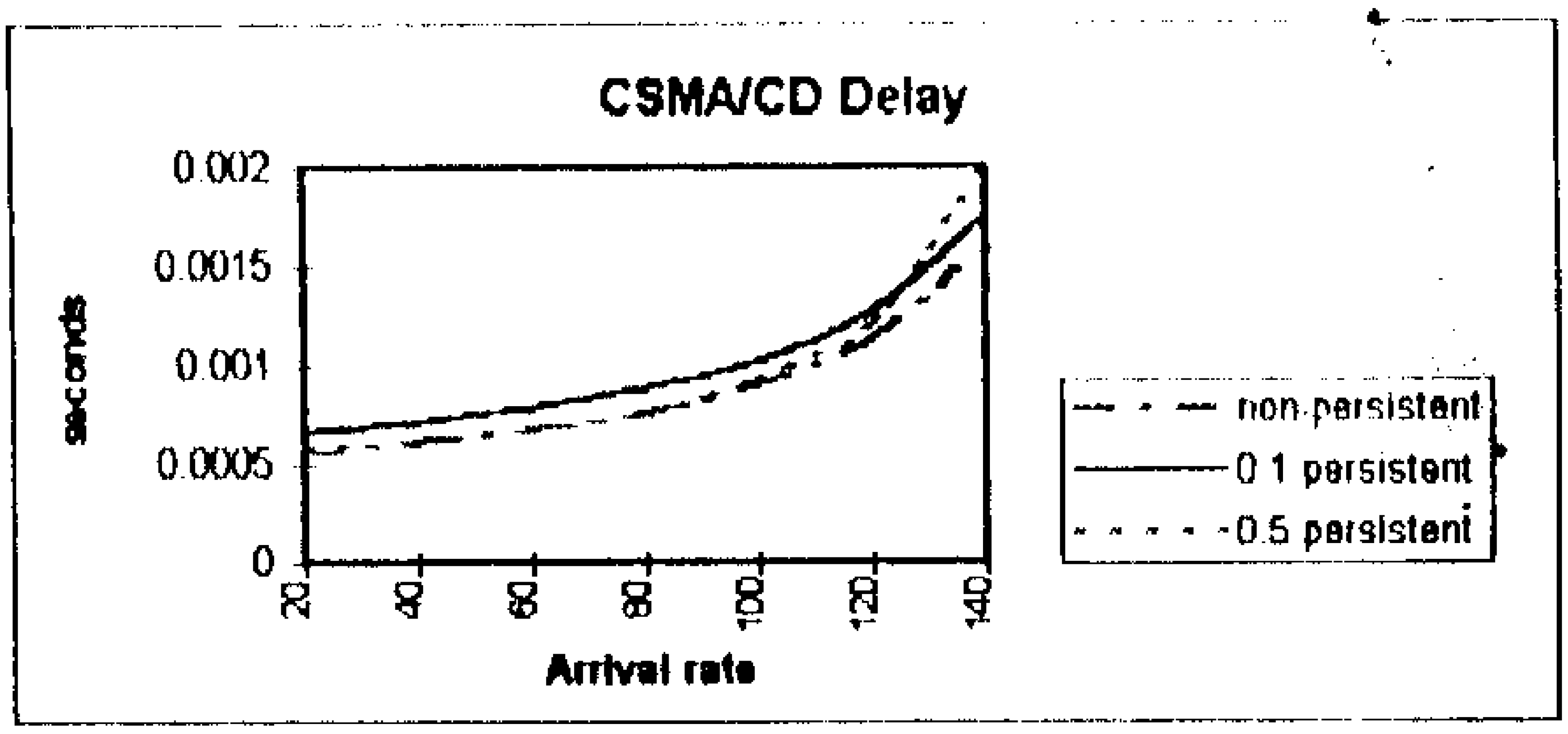
In this simulation the packet size was kept fixed at 1000 bits. The channel capacity was 2 MHz. Results were obtained by varying the persistence. The first graph plots the throughput against the arrival rate. As we can see the throughput is approximately 10 times the arrival rate. This is because there are 10 stations on the network generating packets with the same arrival rate. After the arrival rate becomes 180 packets per second the system becomes unstable and is unable to function properly.



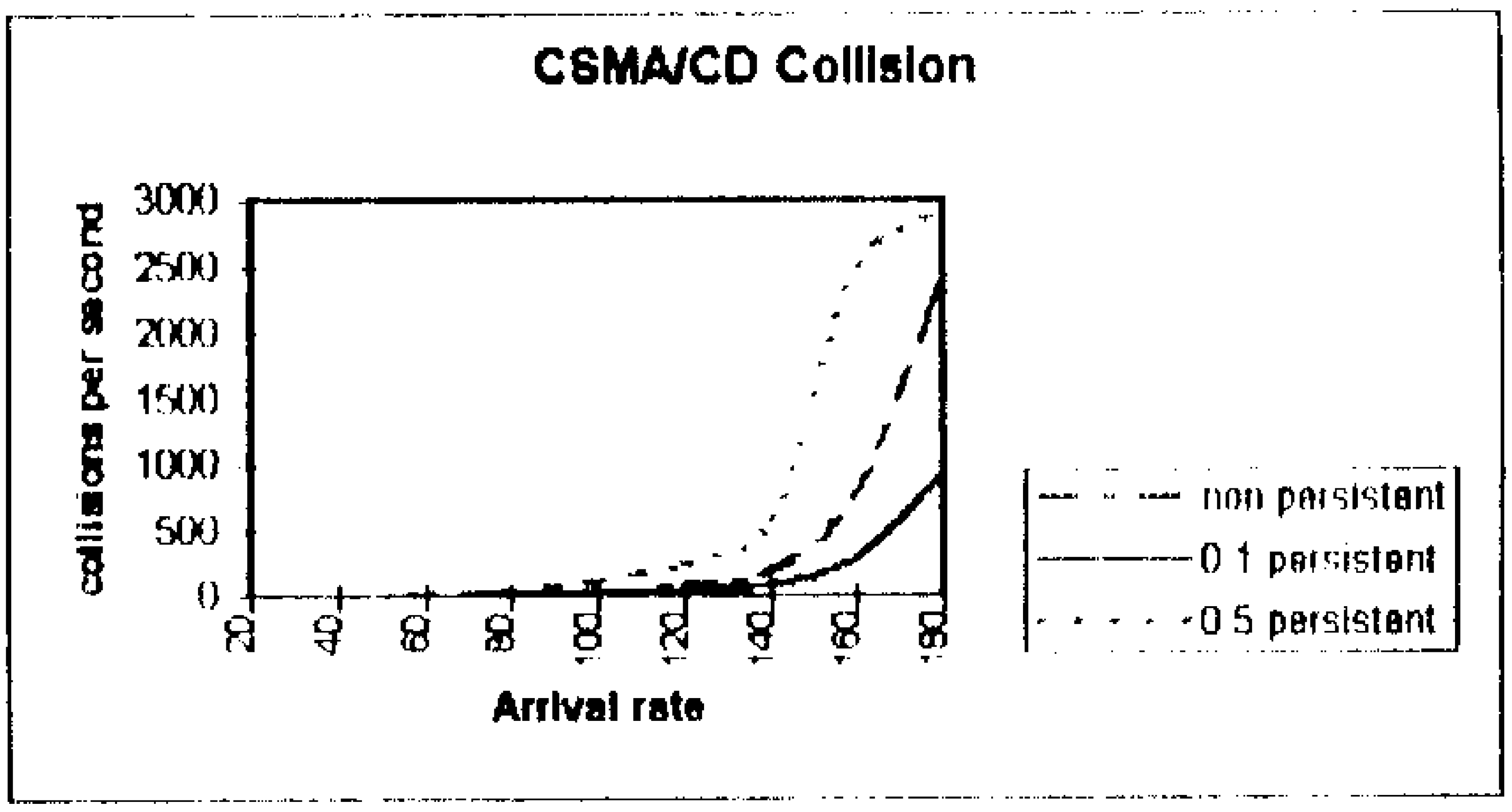
Next, we take a look at the delay in the system. Again the delay for all three persistence values is plotted on the same graph



Since at higher values of arrival rate the delay increases drastically, we take a closer look to see how the delay increases with the arrival rate



The next plot shows how the collision rate varies with arrival rate.



Appendix

References

1. *Computer networks*, **Andrew S. Tennenbaum, Prantice-Hall of India.**

2. *Elements of simulation*, **Byron J.T.Morgan, Chapman and Hall.**

3. *Simulation of Local Area Networks*, **Matthew N.O.Sadiku, Mohammad Ilyas,CRC Press.**

4. *Design and analysis of computer communication networks*, **Vijay Ahuja,McGraw-Hill Book Company.**