PERFORMANCE ANALYSIS OF THE IEEE 802.5 TOKEN RING

Jeffery H. Peden

Computer Science Report No. RM-86-06
July 1986

Performance Analysis of the IEEE 802.5 Token Ring

A Thesis

Presented to

the Faculty of the School of Engineering and Applied Science

University of Virginia

In Partial Fulfillment

of the Requirements for the Degree

Master of Science (Computer Science)

by

Jeffery Harold Peden

January 1987

Abstract

Currently three IEEE local area networks (the 802.3 contention bus, 802.4 token bus, and 802.5 token ring) are receiving much attention in industry. Interest in 802.5 is due to its applicability to real-time control and monitoring functions, and its potential use of mixed media in various operating environments.

This thesis studies the performance of the 802.5 token ring under various network configurations. Measurements of interest include token cycle time, mean packet delivery delay, total throughput, and queue lengths. Performance data are presented for varying numbers of ring stations, packet sizes, and offered loads, and the ring's low protocol overhead and effective use of pipelining are discussed. The performance of the token ring when handling multiple priority traffic is also studied, and it is shown that the priority of a message has no significant impact on its relative level of service.

An analytic model of network performance is developed that predicts token cycle time, delay, maximum throughput, and queue lengths. This model is shown to be an excellent predictor of mean network performance, and comparisons are given between simulation results and analytic predictions.

Acknowledgements

I would like to thank Dr. Alfred C. Weaver, without whose help I would have gotten nowhere, and everyone else, past and present, who has been involved in my education.

Also deserving of thanks are my parents, Harold and Louise, for continuing to believe I could do it; Phil Dickens, for showing me how much fun it is to be a student; Nancy Schult, for several references, and for help with the analysis of simulation results; John Yancey, for help with TROFF; Mangala Gorur, for many hours of pleasant discussion; and Dr. Orlando J. Andy, without whom none of this would have been possible.

Finally, I would like to thank the Nasa-Lewis Research Center and the Virginia Center for Innovative Technology for the funding that made this research possible.

Table of Contents

List of Figures

Chapter 1

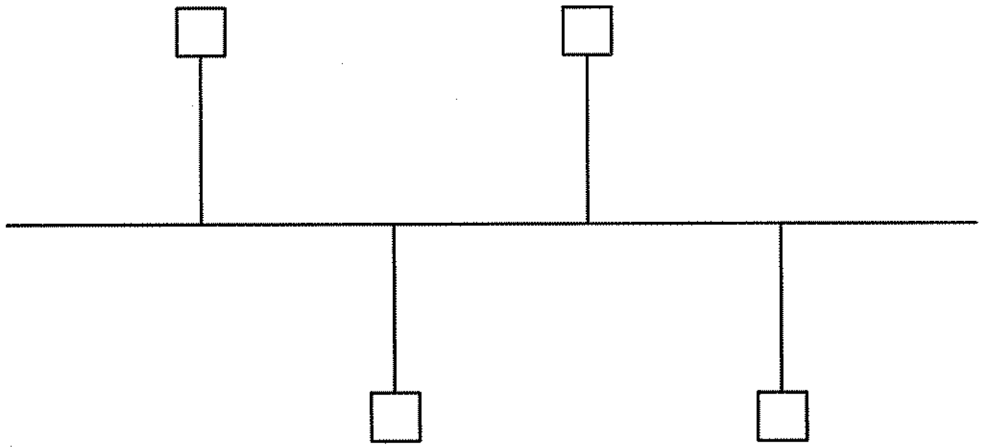
Local Area Networks

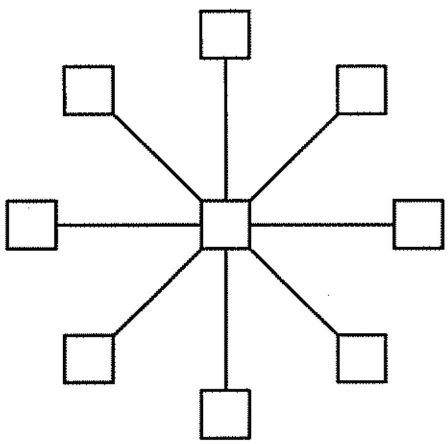
Local area networks are coming into more and more widespread use, and the multiplicity of protocols is growing rapidly. In an attempt to lend some standardization to this growing field, IEEE has established the 802 family of local area network protocols, which includes the 802.3 contention bus, the 802.4 token bus, and the 802.5 token ring.

1.1. Introduction

A local area network (LAN) is typically a high-speed packet communication network that is geographically limited [CLAR78]. This geographic limitation is seen in the fact that most LANs exist in a single building, or in a group of physically adjacent buildings (such as a university campus). One limitation on the size of a local area network can be seen in the fact that if a LAN spans over a few kilometers, the advantages of a high data transmission rate are lost due to increased propagation delays.

Network topologies can be divided into two main categories: restrained and unrestrained. An example of an unrestrained topology is a generalized graph. This structure, however, needlessly complicates many aspects of a local area network protocol (e.g., addressing), so most LANs use some form of restrained topology. Examples of restrained topologies are the star, the bus, and the ring. Diagrams of these topologies are shown in Figure 1.1.





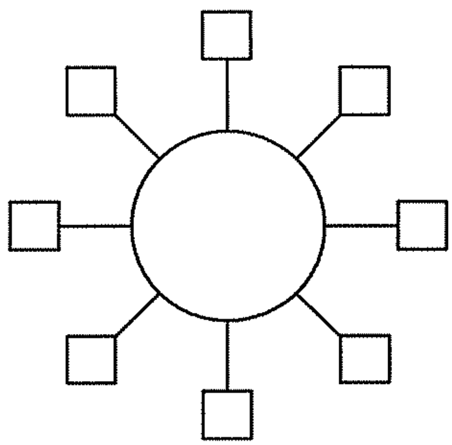


Figure 1.1 — LAN Topologies

A star network consists of a central control node and the communicating nodes. The central node has the network control functions, while the other nodes merely send and receive messages. This topology has the advantage of simplicity, but lacks robustness. For example, if the central node dies, the entire network is disabled.

Bus and ring topologies are much more decentralized. Both consist of a transmission medium with stations connecting to the medium. Protocols for these topologies are typically more complex than those for star networks, but bus and ring networks are also much more robust, in that the failure of a single node need not bring down the entire network.

1.2. IEEE 802

IEEE 802 is a family of standards for local area networks which deal with internetworking, network management, and ISO OSI layers 1 and 2 (physical layer and data link layer). The 802.1 standard covers internetworking and network management, the 802.2 standard deals with the data link layer, and 802.3, 802.4, and 802.5 specify local area network access methods [IEEE84]. A diagram showing the relationship between the standards and the ISO OSI model is shown in Figure 1.2.

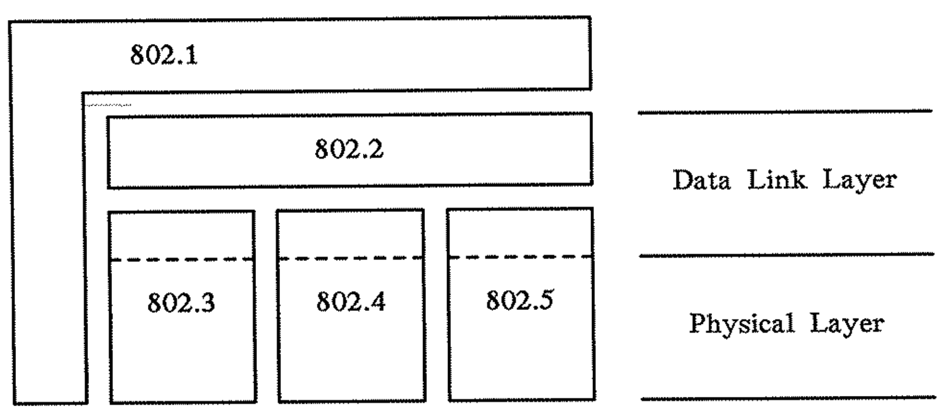


Figure 1.2 — Relationship of IEEE 802 to ISO OSI Standard

1.2.1. IEEE 802.3

The 802.3 protocol is a standard for a bus using carrier sense multiple access with collision detection (CSMA/CD) [IEEE85c]. CSMA/CD is a method whereby a station is free to transmit at any time that there is no other traffic on the bus. A station determines the state of the bus by constant monitoring (hence "carrier sense").

Collision detection is necessary since two or more stations may attempt to transmit simultaneously. Therefore stations must monitor their own transmissions for at least one round trip propagation time for a signal on the bus (the collision window).

The general operation of the 802.3 CSMA/CD bus is as follows: Whenever a station wishes to transmit a packet, it waits until the bus is clear of all other transmissions, and then immediately begins transmission. During the collision window, it monitors the bus for a collision. If no collision occurs, the remainder of the packet is transmitted. If there are more packets waiting for transmission, the process is repeated.

If a collision does occur, the stations transmitting the colliding packets immediately cease their packet transmissions, and immediately transmit "jam", which is a special bit sequence used to enforce the collision, thus ensuring that it was detected by all stations. The colliding stations then wait random amounts of time before attempting retransmission. The random variable controlling the wait time is a binary exponential backoff function. The 802.3 bus has a transmission speed of 10 Mbps.

1.2.2. IEEE 802.4

The 802.4 protocol is a method of bus transmission utilizing token-passing rather than collision detection [IEEE85a]. Token-passing is the utilization of a special fixed-length packet which is transmitted from station to station in canonical order. When a station receives the token, it may then transmit packets, but not otherwise. Stations thus connect to the bus in the form of a logical ring. Since logically adjacent stations may be in any physical order, each station must address the token to its logical successor.

When a station on the bus receives a token, and has a packet (or packets) ready for transmission, it begins transmission. Since only stations which have received the token may transmit, no collisions are possible under proper protocol operation, thus removing the need for collision detection.

The token bus implements four different priorities for packets. These priorities are Synchronous, Urgent_Asynchronous, Normal_Asynchronous, and Time_Available, with Synchronous being the highest. Each priority has an associated timer which controls the relative amount of service to be given to that class of messages.

A station terminates packet transmission (higher priority packets being sent before lower priority packets), either because its queue is empty, or because the timer which controls the amount of time the station may hold the token has expired. When the station has finished its transmission, it then executes a routine to determine if there are any heretofore inactive stations which now wish to join the logical ring (stations may also opt to leave the ring). If so, the inactive station with the highest address joins, and the token is sent to it. Otherwise, the token is sent to the station's current successor. The token bus operates at a transmission speed of 1, 5, or 10 Mbps.

1.3. IEEE 802.5 Token Ring

Like 802.4, IEEE 802.5 is also a token-passing protocol [IEEE85b]. This LAN, however, is a physical ring, unlike the token bus, which is a logical ring. Each station's logical successor is also its physical successor, and stations connect to the ring using digital repeaters. There is therefore a small delay imposed that the token bus lacks, but on the other hand, the token ring can (and does) take advantage of pipelining.

The token used by 802.5 is much shorter than that used by the token bus, because there is no need to make room for an address field. Since the token is shorter, at high loads the ring performs with more efficiency than does the token bus (see Chapter 4 for a performance analysis of the token ring). The token ring also implements eight priorities rather than four.

One major disadvantage of the token ring is that a break in the physical medium causes the network to cease operation. An advantage, however, over bus networks is that ring networks can use mixed media (such as fiber optics in high EMI environments, with twisted pair elsewhere).

The purpose of this thesis is to examine the IEEE 802.5 token ring. Chapter 3 presents relevant mathematical models, including a queuing model for the single priority case. Chapter 4 gives a performance analysis for many different ring configurations, and Chapter 5 presents appropriate conclusions and discusses possibilities for further research. Chapter 2 is a description of the 802.5 protocol.

Chapter 2

Token Ring Protocol

The intent of this chapter is to give a fairly detailed summary of the operation of the token ring LAN. The topology of the ring will be described, along with an explanation of the protocol involved in its operation. It is not the purpose of this chapter to reproduce the IEEE specifications, but instead to give a presentation of the overall workings of the ring in a detailed summary. The facts in this chapter were taken from [IEEE85b].

2.1. Introduction

IEEE 802.5 is a non-collision based protocol in which a series of bits called a *token* circulates on the ring. When a station accesses a *token*, it gains the right of transmission.

The topology of the ring is that of a closed physical loop transmission medium with stations connecting to the medium. Bits on the ring are transmitted serially and unidirectionally, and in a properly operating ring circulate one full circumference before being removed by the transmitting station. The receiving station, if any, copies the transmitted bits as it repeats them around the ring. Stations may be in *active* or *bypass* (not inserted into the ring) mode.

When a station gains the right of transmission, it modifies the *token* to become a Start_of_Frame sequence, then appends control and address information, data, and an End_of_Frame sequence. After a station finishes its transmission, it generates a new *token* so that other stations may access the ring. The amount of time that any station may transmit is controlled by a timer which limits the time a station may hold a *token*.

Eight different priority levels are available to the user. Message priorities are set by the users of the ring. The priority of the circulating *token* (which is also the priority of the ring) is raised and lowered by the ring stations depending upon the traffic and the message classes on the ring.

In case of errors on the ring, there is a monitor function that detects and aids in recovery from these errors. Each station has the potential of performing this function, and there is always a backup monitor to the active monitor.

2.2. Terms

Several terms will be used in the following sections describing the protocol and operation of the token ring. Definitions and associated acronyms are given in this section.

Station: a device which may be attached to the ring for the purpose of transmission and reception of data, and is identified by an address.

Token: a unique sequence of bits which is passed from station to station around the ring and which confers the right of transmission to a station.

Fill: a sequence of bits of any length (within the constraints imposed by timing considerations), composed of randomly generated zero and one bits. Fill is transmitted by a station holding a *token* before or after frame transmission, *token* transmission, or the sending of an abort sequence. Fill is transmitted to prevent an inactive or indeterminate state of a transmitting station.

Repeat: the action of a station receiving a bit stream from its upstream station and retransmitting the bit stream to its downstream station.

Protocol Data Unit (PDU): a message (delivered as a unit) between stations on the ring containing control information, address information, or data.

Priority: a priority is defined as the relative right of access to the ring for purposes of transmission. If PDU A is of higher priority than PDU B, PDU A is to be transmitted before PDU B.

Frame: a transmission that carries a PDU.

Abort Sequence: a bit sequence which prematurely terminates the transmission of a frame.

Broadcast Transmission: a transmission that is addressed and sent to all stations on the ring.

Octet: a sequence of eight bits.

Non-Data Symbols: symbols which are transmitted in one bit time, but are not defined as zero or one. Non-data symbols can be used to uniquely delineate certain octets which have specialized usages. Using Differential Manchester Encoding, two non-data symbols are defined, and are referred to as *non−data J* and *non−data K*.

Differential Manchester Encoding: a polarity-independent method of symbol coding where each bit symbol is split into halves of inverse polarity. A logical "1" is represented by no polarity change at the start of a bit time; a logical "0" changes polarity at the start of a bit time. See Section 2.10.1 for a discussion of this coding method.

2.3. Formats

The following two subsections contain descriptions of the *token* and frame formats, and the usages of the octets in the fields. Many of the octets are subdivided further for purposes of control and addressing.

2.3.1. Token Format

The *token* format consists of three octets: a starting delimiter, an ending delimiter, and an access control octet between them. The *token* format is shown below.

Starting_Delimiter	Access_Control	Ending_Delimiter

The Starting_Delimiter is the means whereby a station recognizes that the transmission on the ring is indeed a *token* or data frame, and consists of two triples of $(J, K, 0)$, and two zeros:

J K 0 J K 0 0 0

The non-data symbols are necessary to avoid confusion between a *token* and a data (or broadcast) transmission.

The Ending_Delimiter consists of two triples of (J, K, 1), plus an intermediate frame bit and an error detected bit:

J K 1 J K 1	I	E

Again, the non-data symbols are for the purpose of uniquely distinguishing the octet from normal data. The intermediate frame bit (I) is set to one if the message being transmitted is not the last frame of a multiple frame transmission, and to zero otherwise. If an error is detected by any station repeating or receiving the data transmission, the error detected bit (E) is set to one, otherwise it is repeated as received. It is always initially transmitted as zero by the station originating the message. It should be noted that the standard does not define what (if any) actions should be taken by a station when it receives a negative acknowledgement of a message (i.e., the E bit of its transmission is set to one). However, it is reasonable to assume that this message would be requeued for transmission.

The Access_Control octet consists of three priority bits, a *token* bit, a monitor bit, and three reservation bits as shown:

P P P	T	M	R R R

The priority bits (PPP) indicate the priority of a *token* (priorities range from 000 to 111, bits transmitted most significant bit first, with 111 being the highest priority). In a multiple priority LAN, they determine which station (or stations) may access a *token* and thus gain the right to transmit.

The token bit (T) is always transmitted as zero if the *token* alone is being repeated. If, however, the *token* is accessed by a station and modified into a start of frame sequence, it is set to one.

The monitor bit (M) is modified only by the active monitor on the ring. The monitor is the ring function that is responsible for the recovery of error situations. There is only one monitor on the ring at a time (although all stations are potentially responsible for this function), and it is referred to as the active monitor.

The monitor bit is used in order to prevent a *token* or frame with a priority greater than zero from continuously circulating on the ring. If the monitor detects a monitor bit equal to one in a frame or *token* of priority greater than zero, that frame or *token* is aborted. The monitor bit is always initially transmitted as zero, and all stations repeat this bit unchanged.

The reservation bits are used by repeating stations to request that the next *token* be issued at a requested priority. A station may only modify these bits if it holds a message of higher priority than the value of the reservation bits currently being repeated. These bits are also transmitted most significant bit first.

2.3.2. Frame Format

The frame format consists of three groups of octets: the Start_of_Frame_Sequence (SFS), the Frame_Check_Sequence (FCS), and the End_of_Frame_Sequence (EFS). The bit length of each field is shown.

SFS		FCS					EFS	
SD	AC	FC	DA	SA	INFO	CRC	ED	FS
8	8	8	16/48	16/48	0..N	32	8	8

The SFS is made up of the starting delimiter and access control octets, which are of the same format as that in a *token*. However, when these two octets are part of a data transmission frame, the token bit in the access control octet is transmitted as a one.

The FCS is composed of one Frame_Control (FC) octet, either two or six Destination_Address (DA) octets, two or six Source_Address (SA) octets, an information field of zero or more octets, and a Cyclic_Redundancy_Check (CRC) consisting of four octets.

The FC octet contains two bits describing the type of frame and six bits which function as control bits. The frame type bits are the most significant bits in the octet.

The DA and SA octets uniquely describe the address of the intended receiver of the data frame and the station originating the transmission, respectively. In a forty-eight bit address, the most significant bit in the DA indicates whether the address is an individual or group address, and the following bit indicates local or global address administration. In a sixteen bit address, only the most significant bit is a special purpose bit, and is the same as in a forty-eight bit address. Sixteen bit addresses are always locally administered, whereas forty-eight bit addresses may be locally or globally administered. Local administration is used for addresses local to a particular LAN segment, whereas global administration, for example, would be used in the case of several interconnected ring networks. All stations on the ring must have address fields of the same length; there is no mixing of sixteen and forty-eight bit addressing modes. In the source address field, the bit determining local or group addressing is always set to zero, which is an arbitrary choice on the part of the standard.

The information field (INFO) may contain zero or more octets of message, and there is no specified limit for the length of this field. However, timing constraints prevent messages from being arbitrarily long, as there is a predetermined length of time that a station may hold a *token* for the purposes of transmission.

The CRC octets contain the information to enable the receiver to perform a cyclic redundancy check on the frame transmitted. This check is done on a standard polynomial of degree thirty-two. It is transmitted with the coefficient of the highest term being the most significant bit of this field.

The ED is identical to the *token* ED, with the intermediate frame and error detected bits being set appropriately. The delimiter is considered valid if the first six bits in the ED are received correctly.

The FS (Frame_Status) field is sent as two sets of four bits. These two sets are identical and are transmitted most significant bit first: an Address_Recognized bit (A), a Frame_Copied bit (C), and two bits that are reserved for future standardization (at present they are transmitted as zero, and ignored by receiving stations):

A C r r	A C r r

The A and C bits are transmitted as zero. The Address_Recognized bit is set to one if a station recognizes the DA as its own. The Frame_Copied bit is set to one if the intended receiving station successfully copied the message into its receive buffer (i.e., if no error was detected).

2.3.3. Medium Access Control Frame Format

Medium access control frames are composed of vectors. A vector is composed of two octets containing the length of the vector, two octets identifying the function of the vector, and zero or more subvectors. Subvectors contain two one-octet fields describing their length and function, plus one or more octets containing their value. Subvectors may optionally contain other special purpose subvectors.

Medium_Access_Control (MAC) Frames are used in the management of the token ring. There exist some undefined values for these frames which are reserved for future specification.

If a station in standby mode determines that there is no active ring monitor, it enters a "claiming token" state, and sends out Claim_Token frames (CL_TK). While doing so, it also checks the source address of all CL_TK frames it receives. If the source address of the CL_TK frame matches its own address, and the initial subvector of the CL_TK matches its upstream neighbor's address, the station enters the active monitor mode and places a new *token* on the ring.

The Duplicate_Address_Test (DAT) MAC Frame is used to detect stations with identical addresses. In the ring initialization process, the DAT frame is put on the ring with the destination address equal to the transmitting station's address. If the

DAT frame returns with the address bit set to one, a duplicate address has been detected, notification is sent to the network manager (while not defined in the standard, it is reasonable to assume that the network manager is the network operating system), and the station returns to bypass mode. If a station copies a DAT frame, it ignores it.

The Active_Monitor_Present (AMP) MAC frame is used by the active monitor on the ring. It is transmitted after a successful purge of the ring or after expiration of the active monitor timer. If a station receives this frame and is in its *standby* state, it resets its Standby_Monitor_Timer (see below).

The standby monitor on the ring periodically transmits Standby_Monitor_Present (SMP) MAC frames in order to assure the other stations on the ring that it is functioning properly. If the standby monitor receives an AMP frame or an SMP frame whose Address_Recognized and Frame_Copied bits are zero, the timer which controls the enqueuing of an SMP PDU is reset. When this timer expires, a standby monitor present frame is queued for transmission.

In the case of a serious ring failure such as a broken cable or a malfunctioning station, the standby monitor transmits Beacon_MAC frames, which carry address and failure information. These frames are used to aid in the localization and correction of the ring failure.

Purge_MAC frames are transmitted by the active monitor on the ring after the monitor has claimed a *token*, or after reinitialization of the ring when a monitor bit has been set to one. These frames are used to clear the ring of any previous traffic.

2.4. Timers

To ensure synchronization of operation of the various stations on the token ring (not to be confused with synchronous data transmission), several timers are defined for use. The values of these timers are established by mutual consent of the ring users, although all have default values. Also, certain timers are only used by certain stations at any point in time.

Each station has a Token_Holding_Timer (THT) which limits the length of time that a station may transmit after having accessed a *token*. A station, having accessed a *token*, transmits frames until it has no more to send, or until the transmission of another frame would take more time than is left on the THT. Its default value is 10 ms.

The Return_to_Repeat_Timer (TRR) limits the time that a station may wait to receive the header of a message it transmitted. If this timer expires before the station sees this message header, the station returns to the repeat mode. The default value of the TRR is 2.5 ms.

The active ring monitor uses the Valid_Transmission_Timer (TVX) to check that stations which have accessed a *token* are actually using the *token*. Therefore, the time-out value of the TVX (the maximum time for which a station may be transmitting) is the sum of the THT and the TRR, with a default of 12.5 ms.

It should be noted that the above description of the TVX assumes that the values of the THT and TRR are global to the network. However, the standard makes no statement to this effect. If the values of the THT and TRR are not global, it is reasonable to assume that the value of the TVX would be the sum of the maximum THT and the maximum TRR on the ring.

The No_Token_Timer (TNT) is used to detect the absence of a *token* on the ring due to error. To ensure that the stations on the ring do not prematurely assume the absence of a *token*, its value is $TNT = N(TRR + THT)$ with a default value of one second.

The proper functioning of the ring requires that an Active_Monitor_Present frame (AMP) be transmitted around the ring every so often. To ensure this, the active ring monitor uses the Active_Monitor_Timer (TAM) to limit the amount of time between transmission of these frames. Its default value is 3 seconds.

When a station receives an AMP or Standby_Monitor_Present (SMP) frame with the Address_Recognized and Frame_Copied (A and C) bits equal to zero, it resets is Queue_PDU_Timer (TQP). The station must then enqueue its own SMP by the time the TQP has expired.

As a check on the operation of the active ring monitor, each standby monitor uses a Standby_Monitor_Timer (TSM). If a standby monitor has not detected a *token* or an AMP frame on the ring within the time-out value of this timer (default 7 seconds), an error is assumed.

2.5. Flags

Certain flags are used by the stations on the ring in order to mark particular events. The intermediate frame flag (I bit) is used to mark a frame that is not the last frame in a transmission, in which case it is set to one. When a station receives a start of frame sequence, it sets its Start_of_Frame_Sequence (SFS) flag, and the reception of a station's own address in the source address octet of a frame causes the My_Address_Received (MA) flag to be set to one.

2.6. Priority Registers and Stacks

Each station has a priority register and stack. These are used to raise and lower the priority of the *token* on the ring (if there was no method of lowering *token* priority, it is possible that within a short period of time after ring initialization, only messages of the highest priority could be transmitted).

The value of the priority bits of the latest received access control field is stored in the priority register as Pr. The value of the most recently received reservation bits is stored in the reservation register as Rr.

If the priority of an enqueued message or the value of the reservation register is greater than the value of the priority register at the time for the transmission of a *token*, the *token* is transmitted with its priority equal to the higher of the two. Since the station has raised the *token* priority, it stacks the value of the priority register as the value Sr, and stacks the priority value of the transmitted *token* as Sx. The station uses these stacks in order to lower the *token* (and ring) priority at some later time.

2.7. Normal Operation

During the following discussion, it may be useful to refer to Figures 2.1 and 2.2, which are the Finite State Machine and Transition Table for the normal operation of the ring.

A station on the ring is generally in *repeat* mode (state 0). In this mode, the station (for the most part) merely repeats bits as received. However, certain bits and fields in the bit stream may be modified by the station. If a station copies the bits being repeated into its receive buffer, it sets the Frame_Copied bits (the C bits in the FS octet of the EFS field) to one in the frame being repeated. If a station detects that the frame being repeated is in error, it sets the error bit (the E bit in the Ending_Delimiter) in the frame to one. If a station sees its own address (or the address of its group), the Address_Recognized bits in the frame (the A bits in the FS octet) are set to one.

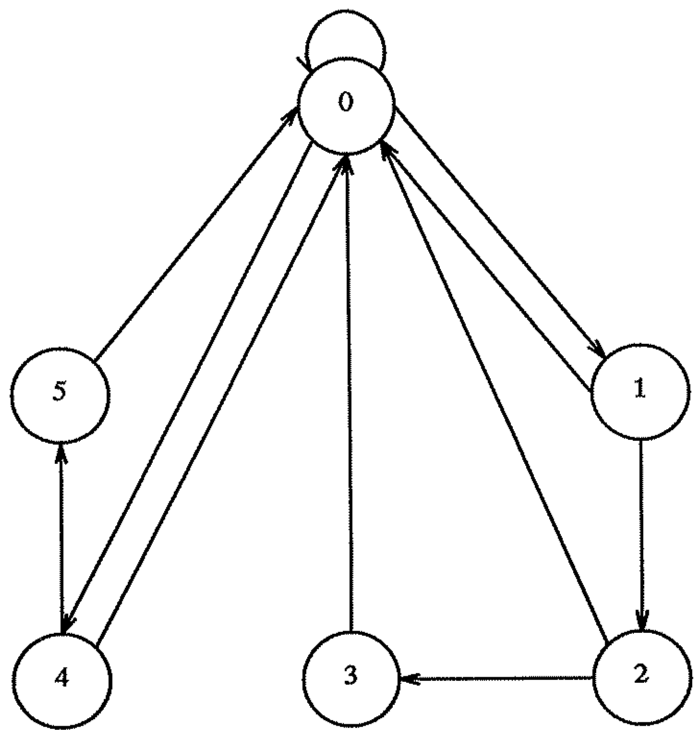


Figure 2.1 — Normal Operation Finite State Machine

When a station has a frame ready for transmission that has a priority higher than the priority of the reservation bits in the frame being repeated, the station may then set the reservation equal to the priority of its message. This informs the other stations on the ring that a station is requesting a priority *token* (one that has priority equal to the message enqueued at the station).

A usable *token* is defined as one of equal or lower priority than an enqueued message. When a station receives a usable *token*, and has an enqueued message, it then changes from *repeat* mode to *transmit* mode.

In *transmit* mode (state 1), a station transmits one or more message frames, but only those which have a priority equal to or greater than the priority of the

STATE	INPUT	OUTPUT	STATE
0	PDU_QUED ∩ TK(PPP⩽Pm)	SFS(PPP=Pr,M=RRR=0), RST(THT,MA_FLG)	1
0	PDU_QUED ∩ (FR(RRR<Pm) ∪ TK(PPP>Pm>RRR,PPP≠Sx))	RRR=Pm	0
0	FR_WTH_ERR	E=1	0
0	DA=MA	A=1	0
0	FR_COP	C=1	0
0	(QUE_EMP ∪ (PDU_QUED ∩ Pm<Sx)) ∩ TK(PPP=Sx)	SFS(PPP=Pr,M=RRR=0), POP(Sx), RST(TRR,SFS_FLG)	4
1	TK_ERR ∪ FR_PRG ∪ FR_BCN ∪ FR_CL_TK ∪ STN_ERR	TX_ABORT	0
1	PDU_END ∩ (QUE_EMP ∪ TST_THT)	EFS(I=A=E=C=0), RST(TRR,I_FLG)	2
2	MA_FLG ∩ (Pr⩾Rr ∩ Pr⩾Pm)	TK(PPP=Pr,M=0, RRR=max(Rr,Pm))	3
2	MA_FLG ∩ (Pr<Rr ∪ Pr<Pm) ∩ Pr>Sx	TK(PPP=max(Rr,Pm), M=RRR=0), STACK(Sr=Pr,Sx=PPP)	3
2	MA_FLG ∩ (Pr<Rr ∪ Pr<Pm) ∩ Pr=Sx	TK(PPP=max(Rr,Pm), M=RRR=0), POP(Sx), STACK(Sx=PPP)	3
2	TST_TRR	MSI	0
3	I_FLG ∪ TST_TRR	∅	0
4	Rr>Sr	TK(PPP=Rr,M=RRR=0), STACK(Sx=P)	5
4	Rr⩽Sr	TK(PPP=Sr,M=0, RRR=Rr),POP(Sr)	5
4	TK_ERR	TX_ABORT, STACK(Sx=PPP)	0
5	SFS_FLAG ∪ TST_TRR	∅	0

Figure 2.2 — State Transition Table for Normal Operation

accessed *token*. To transmit, the station changes the *token* to a Start_of_Frame_Sequence, but if when doing so the station sees an error in the *token*, or if during transmission the station receives certain specialized messages, the station must abort its transmission and return to *repeat* mode.

When a station completes its transmission, either because it has sent all its messages or because it has run out of time, it appends a final end of frame sequence to the PDU and changes from *transmit* mode to *transmit_fill* mode.

In *transmit_fill* mode (state 2), the station transmits a random sequence of ones and zeros until it receives its own address, or until the Return_to_Repeat_Timer (the TRR, which was set when the station entered this mode) has expired. If the TRR expires before the station receives the frame containing its address, the station returns to *repeat* mode; otherwise it transmits a new *token* (of the appropriate priority), and then changes to the *strip_frames* mode (see below) after the transmission of a new *token*. The priority of the newly transmitted *token* depends on the value of the reservation bits in the *token* just used, the current stacked priority, and the priority of any messages enqueued at the station.

The *strip_frames* mode (state 3) differs from the transmit fill mode in that the station does not repeat received bits. The station continues to transmit fill until it receives an end of frame sequence with the intermediate bit set to zero (i.e., the frame received is the last frame of the message), or until the TRR expires. However, if an intermediate frame bit equal to zero has already been received when the station enters this mode, or the TRR has already expired, the station returns directly to *repeat* mode.

When a station enters the *modify_priority_stack* mode (state 4), it transmits zeros until a new *token* can be generated with a new priority. As soon as the station has performed the necessary steps to accomplish this, the new *token* is put on the ring. It is not necessary for the transmission to have been an integral number of octets, and no ending delimiter is appended. If the station detects that the *token* (which was modified to a start of frame sequence) did not end correctly, the station aborts the transmission, stacks the priority of the *token*, and changes to *repeat* mode.

The priority of the new *token* is determined by the following steps: If the reservation priority of the received *token* is higher than the new highest stacked priority, the new *token* is generated with its priority equal to the priority indicated by the reservation bits, and with its reservation and monitor bits equal to zero. The priority of the new *token* is then put onto the priority stack, and the station changes to the *strip_start_of_frame_sequence* mode (see below).

If the reservation bits in the received *token* indicate a priority less than the new highest stacked priority, the new *token* is generated with its priority equal to the highest stacked priority, with the reservation bits unchanged, and the monitor bit equal to zero. The priority on the stack is then popped, and the station changes to the *strip_start_of_frame_sequence* mode.

In *strip_start_of_frame_sequence* mode (state 5), the station transmits fill (random bits) until reception of a start of frame sequence, or until the TRR expires, at which time the station returns to *repeat* mode.

2.8. Standby Monitor Operation

The discussion of the operation of the Standby Monitor references Figures 2.3 and 2.4, which show the Standby Monitor Finite State Machine and State Transition Table respectively.

The standby monitor is used to ensure that there is a functioning active monitor on the ring. When a station is reset (the No_Token_Timer and the Standby_Monitor_Timer are reset, and a check for a duplicate of the station's

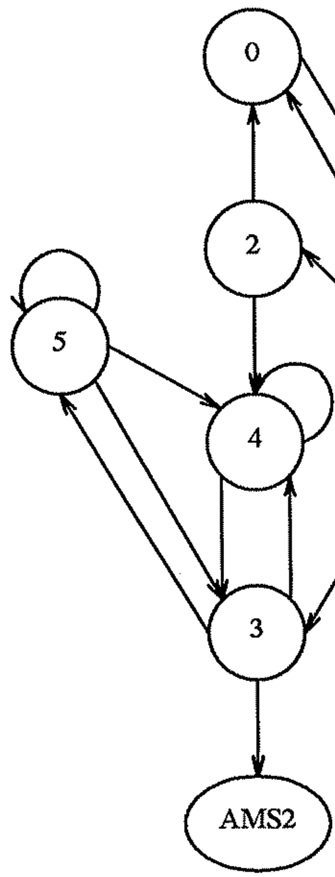


Figure 2.3 — Standby Monitor Finite State Machine

address is made), the status of its address is indicated to the network management and its entry or re-entry into the ring is made known to its immediate downstream neighbor. The station then enters the *inserted* mode (see below) from *bypass* mode.

In *bypass* mode (state 0), the station is not inserted into the ring. All timers are reset, and the latency buffer (used to ensure a minimum amount of delay in transmission) is not used.

When a station is inserted into the ring, the Standby_Monitor_Timer (TSM) is reset, and the station changes to the *inserted* mode (state 1). When inserted, the station first synchronizes its clock with the signal on the ring, and then repeats the bit transmissions on the ring. The station then looks for the presence of an Active_Monitor_Present (AMP) frame or a Purge frame. If neither are received before the TSM expires, the station indicates its address status to the network management, resets its TSM and No_Token_Timer, and changes to *claiming_token* mode. If a Beacon frame is received, the station indicates its address status and returns to *bypass* mode. If an Active_Monitor_Present frame or a Purge frame is received, the station enqueues a Duplicate_Address_Test (DAT) frame, resets its TSM, and changes to *initialize* mode.

In *initialize* mode (state 2), the station transmits its DAT frame and continues to repeat any transmissions on the ring. If the TSM expires before the station receives its DAT frame, or if a beacon frame is received, or if the DAT frame is received and indicates a duplicate address problem on the ring, the station indicates its address status to the network management and returns to *bypass* mode. If the DAT frame is received and no duplicate address problem appears to exist, the station then prepares a Standby_Monitor_Present (SMP) frame for transmission, resets its TNT and TSM timers, and changes to *standby* mode (see below).

In *claim_token* mode (state 3), the station continuously transmits Claim_Token (CL_TK) frames. If, however, the station receives a CL_TK frame from a station with an address greater than this station's address, or a Beacon or Purge frame from any other station, the station resets its TNT and TSM timers and changes to *standby* mode. If the TNT expires before reception of a CL_TL frame (its own or any other station's), the station resets its TSM and changes to *beaconing* mode (see below).

If the station receives its own CL_TL frame and the address bits in the frame equal the assumed address of the station's upstream neighbor, the station becomes the active monitor on the ring. The station activates its latency buffer, resets its TNT, and changes to *active_monitor_purge* mode (see Section 2.8).

A station in *standby* mode (state 4) monitors the transmissions on the ring in order to check the status of the active ring monitor. If *tokens* and active monitor frames are not detected at definite intervals, the station will change to *claim_token* mode.

If the TNT or the TSM expire, the station resets its TNT and changes to *claim_token* mode. If a Beacon frame is received, the station resets its TNT and TSM timers and does not change mode. Also, no mode change is made if the station receives a CL_TK frame, a purge frame, or a *token*; only the TNT is reset.

If the station receives a Standby_Monitor_Present (SMP) frame in which the Address_Recognized and Frame_Copied bits are equal to zero, the station stores the source address of the frame as its upstream neighbor's address and resets its Queue_PDU_Timer (TQP). If the station receives an Active_Monitor_Present (AMP) frame with the Address_Recognized and Frame_Copied bits equal to zero, then in addition to the actions taken above the TSM is also reset. If an AMP frame is received in which another station has modified the Address_Recognized and Frame_Copied bits, the station resets its TSM. If the TQP of a station expires before the station receives an AMP frame or a SMP frame, it prepares its own SMP frame for transmission.

A station operates in *beacon* mode (state 5) only when a serious failure of the ring has been detected. It continually transmits Beacon frames until it receives a Beacon frame. If a station receives a Beacon frame, the source address of which is not equal to its own address, the station resets its No_Token_Timer and

STATE	INPUT	OUTPUT	STATE
0	INSERT	RST(TSM)	1
1	TST_TSM	RST(TNT),MSI	3
1	FR_AMP ∪ FR_PRG	QUE(DAT_PDU),RST(TSM)	2
1	FR_BCN	MSI	0
2	TST_TSM ∪ FR_BCN ∪ FR_DAT(DA=MA,A≠0)	MSI	0
2	FR_DAT(DA=MA,A=0)	RST(TNT,TSM), QUE(SMP_PDU),MSI	4
3	FR_CL_TK(SA>MA) ∪ FR_BCN(SA≠MA) ∪ FR_PRG	RST(TNT,TSM),MSI	4
3	TST_TNT	RST(TSM)	5
3	FR_CL_TK(SA=MA,RUA=SUA)	ADD(LATENCY_BUFFER), RST(TNT)	AMS2
4	TST_TNT ∪ TST_TSM	RST(TNT),MSI	3
4	FR_BCN	RST(TNT,TSM),MSI	4
4	FR_CL_TK ∪ FR_PRG ∪ TK	RST(TNT)	4
4	FR_SMP(A=C=0)	RST(TQP),SUA=SA	4
4	FR_AMP(A=C=0)	RST(TQP,TSM),SUA=SA	4
4	FR_AMP(A≠0,C≠0)	RST(TSM)	4
4	TST_TQP	QUE(SMP_PDU)	4
5	FR_BCN(SA≠MA)	RST(TNT,TSM),MSI	4
5	FR_BCN(SA=MA)	RST(TNT),MSI	3
5	TST_TSM	RST(TSM),MSI	5

Figure 2.4 — State Transition Table for Standby Monitor

Standby_Monitor_Timer, and changes to *standby* mode. If the source address of the received beacon is the same as the station's address, the station resets its TNT, indicates its address status to the network management, and changes to *claim_token* mode. If the station's TSM expires before it receives a Beacon frame, the station resets its TSM, and notifies the network management of its address status.

2.9. Active Monitor Operation

Under normal operating conditions there is only one active monitor. Its purpose is to aid in the recovery from various error situations. The monitor also provides timing and the latency buffer for the ring. Refer to Figures 2.5 and 2.6 for the Finite State Machine and Transition Table for the operation of the active monitor.

The monitor is in *active* mode (state 0) when the ring is operating normally. If the monitor detects a *token* whose monitor bit is zero, and the *token* has a priority greater than zero, the monitor bit is changed to one. The monitor then resets its Valid_Transmission_Timer (TVX). This timer is also reset if the monitor receives a *token* with its monitor and priority bits equal to zero.

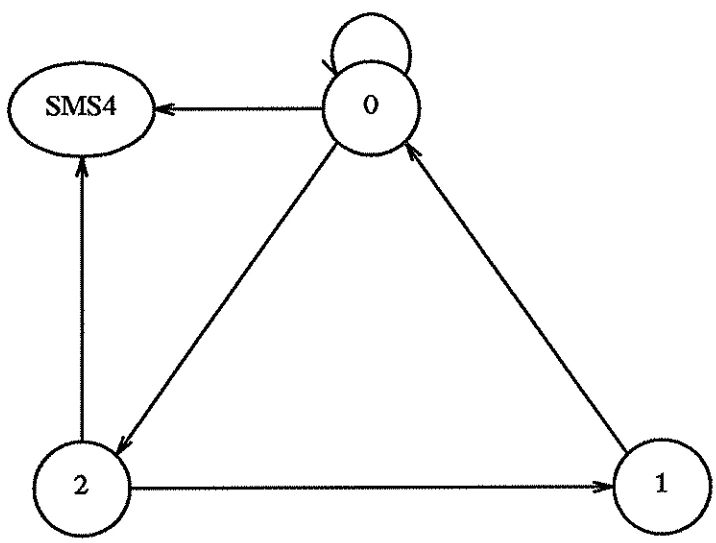


Figure 2.5 — Active Monitor Finite State Machine

If the monitor's Active_Monitor_Timer (TAM) expires, it prepares an Active_Monitor_Present (AMP) frame for transmission and resets its TAM; no mode change occurs.

If a frame or *token* is being repeated by the active monitor which has its monitor bit equal to one, it is aborted. The monitor resets its TNT and changes to purge mode. Also, if the TVX expires, the TNT is reset and the monitor changes to *purge* mode.

When in *purge* mode (state 2), the active monitor continuously transmits Purge frames in order to clear the ring of other transmissions. If the monitor receives a Purge frame that it transmitted, and which was seen by its upstream neighbor, the monitor resets its Return_to_Repeat_Timer and changes to *transmit_fill* mode.

If the TNT expires before the monitor receives a Purge frame that it transmitted, it deletes its latency buffer, indicates its address status to the network management, resets its TNT and TSM timers, and changes to *standby_monitor_standby* mode (see Section 2.7).

If the active monitor receives a Purge frame, a Claim_Token frame, a Beacon frame, or an Active_Monitor_Present frame not transmitted by itself, it deletes its latency buffer, resets its TNT and TSM timers, indicates its address status to the network management, and changes to *standby_monitor_standby* mode.

After the active monitor has finished transmitting purge frames, it enters *transmit_fill* mode (state 1) in order to clear the ring before a new *token* is transmitted. When the Return_to_Repeat_Timer expires, the monitor transmits the new *token* with its priority equal to the reserved priority, and the *token's* reservation and monitor bits equal to zero. The *token's* priority is put on the transmitted priority stack, and zero is put on the reserved priority stack. The monitor

STATE	INPUT	OUTPUT	STATE
0	TK(PPP>0,M=0) ∪ FR(PPP=ANY,M=0)	M=1,RST(TVX)	0
0	TK(PPP=0,M=0)	RST(TVX)	0
0	TST_TAM	QUE(AMP_PDU), RST(TAM)	0
0	FR_SMP(A=C=0)	SUA=SA	0
0	TK(M=1) ∪ FR(M=1)	FR_ABORT, RST(TNT)	2
0	TST_TVX	RST(TNT)	2
0	FR_AMP(SA≠MA) ∪ FR_PRG ∪ FR_CL_TK ∪ FR_BCN	DELETE(LATENCY_BUFFER), RST(TNT,TSM),MSI	SMS4
1	TST_TRR	TK(PPP=Rr,M=RRR=0), RST(TVX,TAM), QUE(AMP_PDU), STACK(Sx=PPP,Sr=0),MSI	0
2	FR_PRG	RST(TRR)	1
2	TST_TNT	DELETE(LATENCY_BUFFER), RST(TNT,TSM),MSI	SMS4

Figure 2.6 — State Transition Table for Active Monitor

then resets its TVX and TAM timers, and changes to *active* mode.

2.10. Symbol Coding and Timing

This section presents some information about the physical transmission aspects of the ring.

2.10.1. Differential Manchester Code

Differential Manchester Code is used in the encoding of bit symbols on the ring. This method of coding has two distinct advantages over the use of binary encoding methods. The first is that in the binary method of symbol coding, only two symbols are possible: zero and one. Differential Manchester Code, however, can generate two additional bit symbols known as *non−data J* and *non−data K*. These symbols differ from data symbols in that there is no mid-bit-time polarity change in their transmission. A *non−data J* has the same polarity as its predecessor, and a *non−data K* has opposite polarity. These symbols are used to uniquely distinguish the SD and ED fields of *tokens* and messages. An example of Differential Manchester Code as compared to non-return-to-zero (NRZ) Binary Code is shown below.

Without the use of Differential Manchester Code, some other method besides the use of non-data symbols would have to be used in order to distinguish the starting and ending delimiters of *tokens* and message frames from data. One such method is bit stuffing [CARL80], but this has the disadvantages of adding to the bit

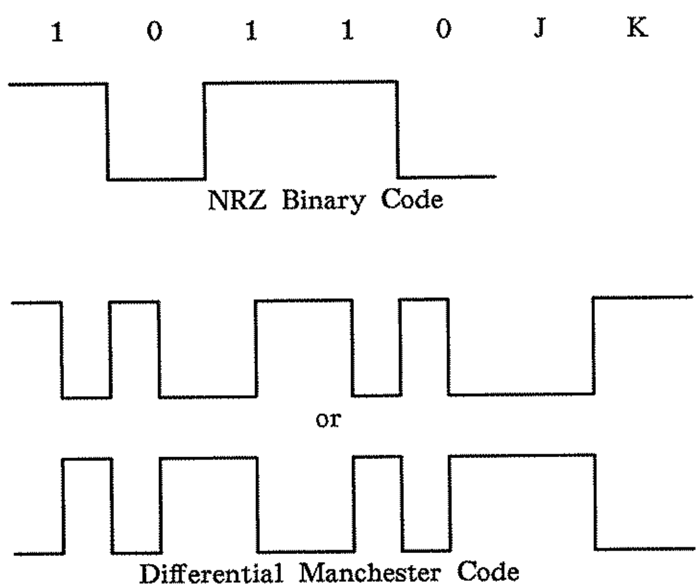


Figure 2.7 — Coding Schemes

load on the ring, as well as the circuitry required to perform the bit stuffing.

The second advantage of Differential Manchester Code is that since all symbols are split into inverse halves, timing information is inherently transmitted along with data, even if the data being transmitted is nothing but a stream of zeros. This also makes it a simple matter to detect the failure of an upstream station, since if nothing is being repeated the energy level of the ring drops; this method of symbol encoding ensures a constant energy level as long as transmission is taking place.

It should be noted that even though the non-data symbols are not divided into inverse halves, there is an equal number of non-data J and non-data K symbols on the ring, as they are always transmitted in pairs.

2.10.2. Timing and Latency

The bit transmission rate of the ring may be either one megabit per second or four megabits per second, and the standard states that the transmission rate should have a tolerance of plus or minus one one-hundredth of a percent.

In order that the *token* will completely fit on the ring, a special buffer called the latency buffer is used. This buffer maintains at least 24 bits of latency, so that the *token* will not overlap itself if all stations happen to be in repeat mode simultaneously. It should be noted that the latency buffer is used even if the size of the ring is great enough so that the ring latency is greater than the length of the *token*. All stations have a latency buffer, but it is active only in the Active Monitor for the ring.

Since the signal transmission rate on the ring may vary a small amount due to phase jitter, the latency buffer is also used to compensate for this effect. This is done by expanding or contracting the latency buffer as needed. The latency buffer may vary between 24 and 30 bits, and is initialized to 27 bits.

The latency buffer expands when the rate of the incoming signal is slightly faster than optimal, and contracts when the signal is slower, thus acting as a negative feedback device to keep the signal transmission rate within the required tolerance.

Chapter 3

Mathematical Modeling of the Token Ring

This chapter deals with the mathematical description of the operation of the IEEE token ring, and is divided into four main sections. The first section of the chapter gives precise definitions of the quantities being modeled, and the second section will present derivations for such quantities as minimum and maximum token cycle time, maximum throughput, and the relative amounts of bandwidth used by delay, protocol overhead, and data. Transmission and propagation delay will also be discussed.

The third section of the chapter will present a queuing model of the ring operation, which can be used to predict the actual token cycle time, mean station delay of a packet, and queue lengths at ring stations. The last section of the chapter will discuss the appropriateness of this model with reference to the actual protocol presented in the previous chapter.

3.1. Definitions

There are several important definitions that must be presented in any mathematical description of a network. The following symbols are used in the definitions.

$n \equiv$ *total number of packets delivered*

$C \equiv$ *medium capacity in bits per second*

$S \equiv$ *total simulation time in seconds*

$\tau \equiv$ *throughput; the fraction of all bits transmitted that comprise frames*

$i \equiv$ *packet number*, $1 \leqslant i \leqslant n$

$M_i \equiv$ *number of bits in the* i^{th} *packet*

$\overline{M} \equiv$ *mean packet length in bits for all stations*

$\gamma_i \equiv$ *propagation time for the* i^{th} *packet*

$\gamma \equiv$ *total ring propagation time*

$T_{H_i} \equiv$ *time the* i^{th} *packet reaches the head of its queue,* $0 \leqslant T_{H_i} \leqslant S$

$T_{Q_i} \equiv$ *time the* i^{th} *packet is initially enqueued,* $0 \leqslant T_{Q_i} \leqslant S$

$T_{F_i} \equiv$ *time the first bit of the* i^{th} *packet is transmitted,* $0 \leqslant T_{F_i} \leqslant S$

$T_{L_i} \equiv$ *time the last bit of the* i^{th} *packet has been received,* $0 \leqslant T_{L_i} \leqslant S$

$D_{Q_i} \equiv$ *queuing delay of the* i^{th} *packet* $= T_{H_i} - T_{Q_i}$

$E(D_Q) \equiv$ *expected queuing delay* $= \dfrac{\sum_i D_{Q_i}}{n}$

$D_{NA_i} \equiv$ *network access delay of the* i^{th} *packet* $= T_{F_i} - T_{H_i}$

$E(D_{NA}) \equiv$ *expected network access delay* $= \dfrac{\sum_i D_{NA_i}}{n}$

$D_{QN_i} \equiv$ *station delay of the* i^{th} *packet* $= D_{Q_i} + D_{NA_i}$

$E(D_{QN}) \equiv$ *expected station delay* $= \overline{D_Q} + \overline{D_{NA}}$

$D_{TR_i} \equiv$ *transmit delay of the* i^{th} *packet* $= \dfrac{M_i}{C} + \gamma_i$

$E(D_{TR}) \equiv$ *expected transmit delay* $= \dfrac{\sum_i D_{TR_i}}{n} + \dfrac{\gamma}{2}$

$D_i \equiv$ *packet delivery delay for the* i^{th} *packet* $= T_{L_i} - T_{Q_i}$

$E(D) \equiv$ *expected packet delivery delay* $= E(D_Q) + E(D_{NA}) + E(D_{TR}) = \dfrac{\sum_i D_i}{n}$

$\beta \equiv$ *total number of data and framing bits received at all stations*

We define offered load to be the number of bits that were presented to the network, computed as the total number of bits in all packets at all stations per unit time, including framing. For any particular configuration, the station-to-station propagation time will be known, and includes both signal delay and station latency. The medium capacity will also be known.

3.2. Derivations

3.2.1. Token Cycle Time

The token cycle time on the token ring is defined as the mean time it takes for the token to travel one full circumference on the ring. This is an important measure of performance (perhaps a basic measure) in that the token cycle time has a direct impact on delay and throughput. The following definitions are given for terms used in the derivation.

$N \equiv$ *number of stations on the ring, numbered 0..(N - 1),* $N \geqslant 2$

$\lambda_N \equiv$ *packet arrival rate in packets per second at the* N^{th} *station*

$\lambda \equiv$ *mean packet arrival rate in packets per second*

$Q_N \equiv$ *number of packets enqueued at the* N^{th} *station*

$E(Q) \equiv$ *expected number of packets enqueued at any station*

$E(TCT) \equiv$ *expected token cycle time*

$THT_j \equiv$ *initial value of the* j^{th} *station's Token_Hold_Timer,* $0 \leqslant j \leqslant N-1$

$TRR_j \equiv$ *initial value of the* j^{th} *station's Return_to_Repeat_Timer,* $0 \leqslant j \leqslant N-1$

$E(V_j) \equiv$ *expected token visitation time at the* j^{th} *station,* $0 \leqslant j \leqslant N-1$

$E(V) \equiv$ *expected token visitation time*

$\mu \equiv$ *mean packet service rate in packets per second* $= \dfrac{C}{\overline{M}}$

$L_j \equiv$ *latency in seconds of the* j^{th} *station,* $0 \leqslant j \leqslant N-1$

$B \equiv$ *latency buffer delay in seconds within the active monitor*

$P_j \equiv$ *propagation delay in seconds from station j to (j + 1) mod N,* $0 \leqslant j \leqslant N-1$

$F(TRR_j) \equiv$ *the effect of the TRR of the* j^{th} *station, and is defined as*

$$F(TRR_j) = \begin{cases} \sum_j L_j + \sum_j P_j + B + \dfrac{56-\overline{M}}{C}, & \overline{M} - 56 < C(\sum_j L_j + \sum_j P_j + B) \\ 0, & otherwise \end{cases}$$

The following two equations give the upper and lower bounds on the token cycle time:

$$TCT_{\max} = B + \sum_j (THT_j + F(TRR_j) + L_j + P_j) \tag{1}$$

$$TCT_{\min} = B + \sum_j (L_j + P_j) \tag{2}$$

Although not specifically required by the standard, it is reasonable to assume that all stations will have a common value for their THT timers, and another for their TRR timers. Also assuming that latency delays are the same in each station, and that propagation delay is the same between all adjacent stations,

$$TCT_{\max} = N(THT + F(TRR) + L + P) + B \tag{3}$$

$$TCT_{\min} = N(L + P) + B \tag{4}$$

These equations, however, give only the upper and lower bounds of the token cycle time. It is also useful to know the mean token cycle time when the network is under load in equilibrium. To make this derivation, we need the token visitation time for the j^{th} station, $E(V_j)$ (the time between the arrival and departure of the token at the station). The derivation of this term will be given in Section 3.3 where a queuing theory model of the token ring is developed.

$$E(TCT) = B + \sum_j \left[E(V_j) + P_j + L_j \right] \tag{5}$$

It should be noted, however, that each term $E(V_j)$ may not exceed $(THT_j + TRR_j)$; $E(V_j) \leqslant THT_j + TRR_j$ is enforced by the standard.

Under the assumption that the discrete values λ_j, THT_j, D_{TR_j}, P_j, and L_j are respectively identical at all stations, this equation may be simplified to:

$$E(TCT) = B + N\left[E(V) + P + L \right] \tag{6}$$

It is appropriate here to make a note about propagation delay. In the above derivations, this delay was represented by the terms P_j and L_j, where L_j is the station repeater latency, and P_j is the signal propagation delay from one station to the next. However, since in most cases $L >> P$, it is correct to state that $L \approx L + P$. Therefore, the above equations can be simplified even further without any meaningful loss of accuracy.

3.2.2. Throughput

Throughput, τ, is defined as the total number of bits received at all receivers per unit time per medium capacity. Thus,

$$\tau = \frac{\beta}{C \cdot S} \tag{7}$$

This is an important measure of the efficiency of the network protocol. If maximum throughput is high, then protocol overhead is low, and vice versa. In any reasonable protocol, measured throughput should rise with increasing offered load until some maximum value is reached, and should stabilize at this value unless offered load decreases.

Maximum token cycle time plays an important part in the derivation of a formula for maximum throughput. This value is also dependent on message length, medium capacity, and station latency.

It is obvious that the maximum number of bits delivered in one token cycle time is bounded by the number of bit times in one token cycle time. However, since some of this bandwidth will be used for protocol overhead (tokens), and some for propagation delay (station latency, latency buffer delay, and signal delay), the actual number of bits delivered will be less than the number of bit times in one token cycle time. The formula for maximum throughput is given below. It should be noted that throughput is a unitless fraction less than one.

$$\tau_{\max} = \frac{\sum_j floor(\mu \cdot THT_j)}{\mu \left[\sum_j (THT_j + F(TRR_j) + L_j + P_j) + B \right]} \tag{8}$$

Notice that the denominator is merely the maximum token cycle time converted to bit times, with the numerator being the maximum number of bits that can be transmitted in that number of bit times. If the discrete values for THT, TRT, P, and L are respectively identical among all stations on the ring, this equation can be simplified as below.

$$\tau_{\max} = \frac{N \cdot floor(\mu \cdot THT)}{\mu\left[N(THT + F(TRR) + L + P) + B\right]} \tag{9}$$

3.2.3. Utilization

When dealing with a network, it is desirable to be able to compute the fractions of available bandwidth used by tokens, framing, data, propagation delay, and protocol overhead. This section will present general formulae for these quantities. We define the following terms for use in these derivations.

$C \equiv$ *available bandwidth*

$t \equiv$ *fraction of bandwidth used by tokens*

$m \equiv$ *fraction of bandwidth used by message traffic*

$f \equiv$ *fraction of bandwidth used by framing*

$d \equiv$ *fraction of bandwidth used by data*

$p \equiv$ *fraction of bandwidth used by propagation delay and protocol overhead*

$b_f \equiv$ *number of framing bits per message*

$b_d \equiv$ *number of data bits per message*

The fraction of bandwidth used by token traffic is a good measure of the performance and efficiency of network operation. If this fraction is high, then the network protocol is not very efficient. To compute this statistic, one needs to convert the token cycle time from units of seconds to units of bit times (a bit time being the time needed to transmit one bit). The result, t, is the length of the token (24 bits) divided by the token cycle time.

$$t = \frac{24}{C\left[B + \sum_j (E(V_j) + P_j + L_j)\right]} \tag{10}$$

We are defining message length to be the number of data bits plus the number of framing bits in a Protocol Data Unit, $l = b_d + b_f$. The fraction of bandwidth used by message traffic, m, is

$$m = \frac{floor(\mu \sum_j E(V_j))}{\mu\left[B + \sum_j (E(V_j) + P_j + L_j)\right]} \tag{11}$$

The equations showing the fraction of bandwidth used by data alone, d, and the fraction used only by framing, f, are directly derived from the previous equation.

$$d = \frac{b_d \cdot floor(\mu \sum_j E(V_j))}{C\left[B + \sum_j (E(V_j) + P_j + L_j)\right]} \tag{12}$$

$$f = \frac{b_f \cdot floor(\mu \sum_j E(V_j))}{C\left[B + \sum_j (E(V_j) + P_j + L_j)\right]} \tag{13}$$

The protocol of the token ring is such that protocol overhead (e.g., special purpose message packets) is minimal, and quite small compared to propagation delay. Therefore, we include the fraction of available bandwidth used by this overhead in that of propagation delay. Since the portion of bandwidth given to propagation delay is a function of data and token traffic, it is appropriate that it be expressed as such.

$$p = 1 - \left[t + f + d\right] \tag{14}$$

3.2.4. Delay

The delay involved in getting a message packet from one station to another is crucial to the performance and utility of any network, and especially to local area networks, when operating under real-time constraints. However, in order to simplify the problem of computing delay, the total delay involved in delivering a packet to its destination is divided into four discrete parts: queuing delay, network access delay, transmit delay, and propagation delay.

Of these four components of delay, two (queuing delay and network access delay) require a queuing theory model. Transmit delay and propagation delay, however, are fairly simple to compute. As we are assuming that the addressing of messages on the token ring is fairly homogeneous, the mean distance that messages will have to travel is halfway around the ring. It therefore follows that the propagation delay of a packet is the time for it to travel halfway around the ring. The transmit delay has been given previously in Section 3.1.

3.3. Queuing Model

3.3.1. Token Cycle Time

In Section 3.2.1, we presented an expression for the token cycle time of the token ring which was given as a function of $E(V)$, the token visitation time. We now introduce $E(T_v)$ as the expected token vacation time of a token from any station. Therefore, the token cycle time may also be expressed as the sum of the token visitation time and the token vacation time,

$$E(TCT) = E(T_v) + E(V) \tag{15}$$

Following the derivation in [HEYM83] of the expected token vacation time, we now introduce $\gamma = B + N(L + P)$ as the round trip propagation time for the ring (note that this is equal to the minimum token cycle time), and $\rho = \frac{\lambda}{\mu}$. Since the expected token vacation time $E(T_v)$ is the same at any station, the number of waiting packets at any station (by Little's law) is $\lambda E(T_v)$.

The time needed to transmit one packet is $\frac{1}{\mu}$, so it therefore requires $\rho E(T_v)$ seconds to transmit $\lambda E(T_v)$ packets. But since $\lambda\rho E(T_v)$ packets arrive at the station during the transmission of $\lambda E(T_v)$ packets, $\rho^2 E(T_v)$ seconds must be added to the previous transmission time. Extending this process, we arrive at the following infinite sum:

$$E(T_v)\left[\rho + \rho^2 + \rho^3 + \cdots\right] = \frac{\rho E(T_v)}{1-\rho} \tag{16}$$

Therefore, the time needed to transmit all packets at a station is:

$$E(V) = \frac{\rho E(T_v)}{1-\rho} \tag{17}$$

Thus, the expected token vacation time is the sum of the minimum token cycle time and the time to transmit all packets in all other stations:

$$E(T_v) = \gamma + (N-1)\frac{\rho E(T_v)}{1-\rho} \tag{18}$$

Simplifying, we arrive at the following equation for the token vacation time:

$$E(T_v) = \frac{\gamma(1-\rho)}{1-N\rho}, \quad N\rho < 1 \tag{19}$$

Equating (6) and (15), which are the two expressions for token cycle time, and solving for the token visitation time, $E(V)$, we obtain the following equation:

$$E(V) = \frac{\gamma(1-\rho)}{(1-N\rho)(N-1)} - \frac{\gamma}{N-1}, \quad N\rho < 1 \tag{20}$$

which simplifies to

$$E(V) = \frac{\gamma\rho}{1-N\rho}, \quad N\rho < 1 \tag{21}$$

Note that this is equal to $\frac{\rho E(T_v)}{1-\rho}$. Since the token cycle time is the sum of the token vacation time and the token visitation time, we arrive at the following equation for the token cycle time:

$$E(TCT) = \frac{\gamma}{1 - N\rho}, \quad N\rho < 1 \tag{22}$$

The expected number of packets enqueued at a station upon the arrival of a token is a straightforward application of Little's law to the expected token vacation time:

$$E(Q) = \frac{\lambda\gamma(1-\rho)}{1 - N\rho}, \quad N\rho < 1 \tag{23}$$

3.3.2. Mean Station Delay

The delay a packet experiences at a station before gaining access to the network is an extremely important consideration in the performance of any network. This delay has two components: queuing delay and network access delay. In Chapter 4 we will present simulation measurements for both of these. However, here we will deal only with a formula for the sum of these two quantities, which we will call mean station delay.

The expected station delay of a packet is proven in [LEVY75] to be the sum of the expected delay of a packet while enqueued and the expected delay awaiting the arrival of a token:

$$E(D_{QN}) = E(D) + \frac{E(T_v{}^2)}{2E(T_v)} \tag{24}$$

But $E(D) = \frac{\lambda \frac{1}{\mu^2}}{2(1-\rho)}$, from the Pollaczek-Khinchine formula for the expected delay in an M/G/1 queue (even though we have a formula for queuing delay, it is not applicable in this situation, as it is predicated on the assumption of continuous service). We have from [KONH74] that:

$$Var(T_v) = \frac{\rho\gamma(N-1)\frac{1}{\mu}}{(1-N\rho)^2}, \quad N\rho < 1 \tag{25}$$

Since $Var(T_v) = E(T_v{}^2) - E^2(T_v)$, we have

$$E(T_v{}^2) = \frac{\rho\gamma(N-1)\frac{1}{\mu} + \gamma^2(1-\rho)^2}{(1-N\rho)^2}, \quad N\rho < 1 \tag{26}$$

which gives, after substitution and simplification, the expected station delay of a packet:

$$E(D_{QN}) = \frac{N\rho\frac{1}{\mu} + \gamma(1-\rho)}{2(1-N\rho)}, \quad N\rho < 1 \tag{27}$$

Once the expected mean station delay of a packet is known, it is a simple matter to compute the expected service delay. Recall from section 3.2.4 that the expected propagation delay of a packet is the time to travel halfway around the ring, $\frac{\gamma}{2}$. Adding to this the transmission time for a packet, $\frac{1}{\mu}$, we arrive at the expected service delay:

$$E(D) = \frac{N\rho\frac{1}{\mu} + \gamma(1-\rho)}{2(1-N\rho)} + \frac{\gamma\mu + 2}{2\mu}, \quad N\rho < 1 \tag{28}$$

3.4. Discussion

The queuing model presented in this chapter has two important underlying assumptions: single priority operation and exhaustive service. This is obviously at variance with the description of the protocol given in Chapter 2, yet we maintain that it is still a useful model.

It will often be the case that in the actual operating environment, only a single level of priority will be needed, and if multiple priorities are implemented, it is reasonable to assume that high priority messages will be few. Therefore, a single priority model should be a good approximation to real world operation.

The assumption of exhaustive service was made for two reasons, the first being the inherent mathematical difficulties associated with a nonexhaustive model [KUEH79]. The second reason is that it is a reasonable assumption to make. It will be shown in Chapter 4 that a station's queue is normally emptied even under high network loads. We therefore believe that these assumptions are justified.

Chapter 4

Performance of the Token Ring

This chapter deals with the performance of the IEEE 802.5 protocol. Measures of performance include throughput, token cycle time, delay, and queue lengths. Graphs of these data will be presented and discussed, along with comparison to mathematical predictions. The results presented in this chapter were obtained via simulation. It must be understood that results obtained from simulation and data obtained from actual measurements will differ to some extent.

4.1. Single Priority Operation

4.1.1. Reference Configuration

When measuring a network (or any computer system), it is important that there be a baseline configuration against which perturbations can be compared. We have therefore constructed a reference configuration for the token ring with the following characteristics: 40 operating ring stations (one of which is the active monitor), 256 bit packets, repeater latencies of 1 bit time, a transmission medium of 1Mbps, Poisson arrivals, and single priority operation (a modification of this configuration is required for the operation of multiple priorities). Propagation delay is considered constant between stations, and error-free operation is assumed. The Token_Holding_Timer (see Chapter 2) is set to the default value of 10 ms in all simulations. No theoretical considerations motivated the choice for the reference configuration; however, it is a reasonable baseline case, and is being used only for purposes of comparison.

4.1.1.1. Token Cycle Time and Queue Lengths

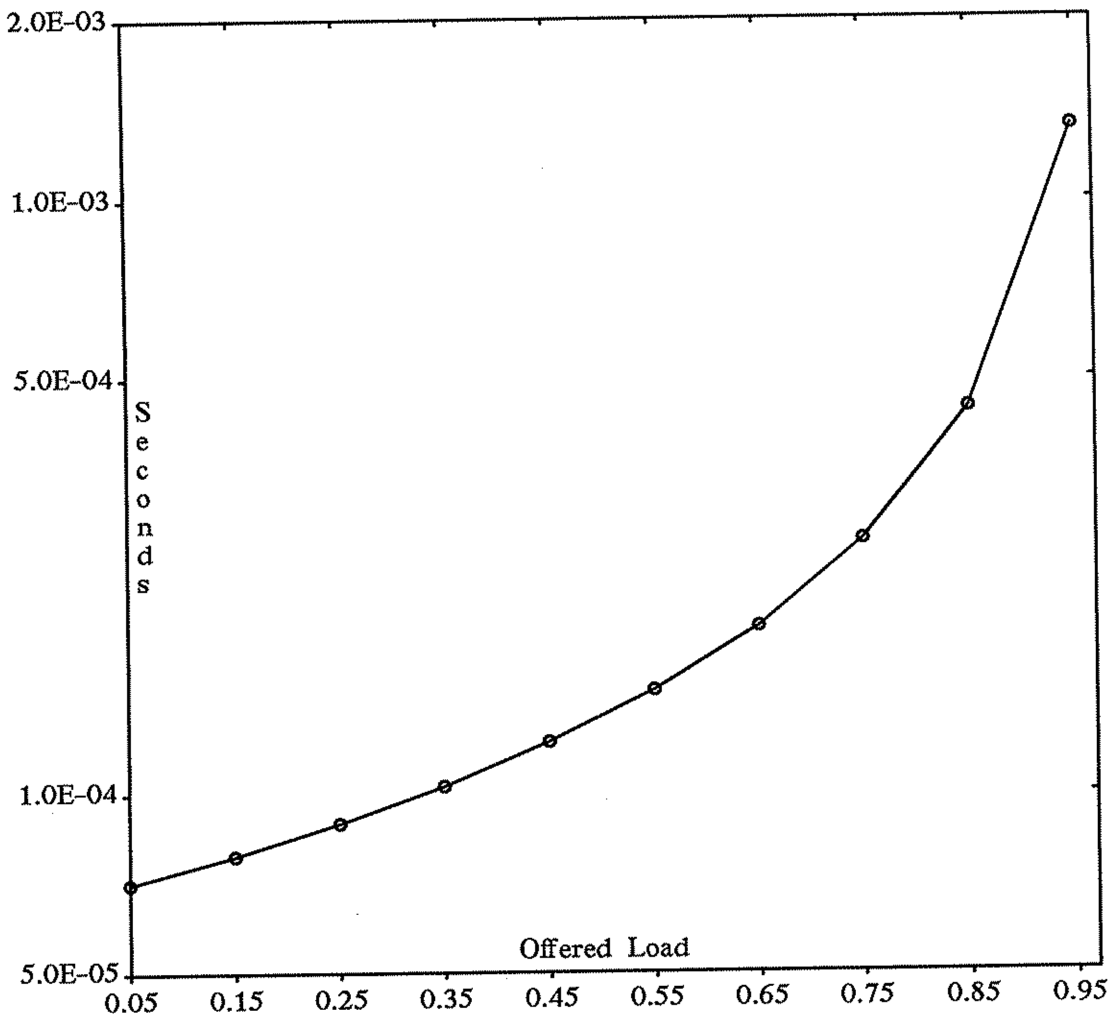
In any form of cyclic service, the cycle time is a basic measure as it has a direct effect on all other measures of performance. Figure 4.1 shows the token cycle time for the reference configuration. The values of the upper and lower bounds are 1.32 ms and 70.7 μs, respectively. Figure 4.2 shows the comparison between the simulation and prediction for the token cycle time. The upper and lower bounds for the predicted values for the token cycle time are 1.34 ms and 70.5 μs.

Queue lengths are measured when a token arrives at a station. Figure 4.3 illustrates the number of packets queued upon token arrival, and varies from 0.000344 to 0.119. Figure 4.4 shows the close correspondence between predicted and measured values for queue lengths. The prediction and simulation values are the same to 3 significant figures up to 0.95 offered load, where the prediction gives 0.121.

In Chapter 3 it was stated that even though the Token_Holding_Timer limits the service to a station, the assumption of exhaustive service was reasonable. Here we state that the mean number of packets remaining enqueued at the end of a service time to a station is zero. Therefore, the assumption of exhaustive service is shown to be valid for the reference configuration.

4.1.1.2. Delay

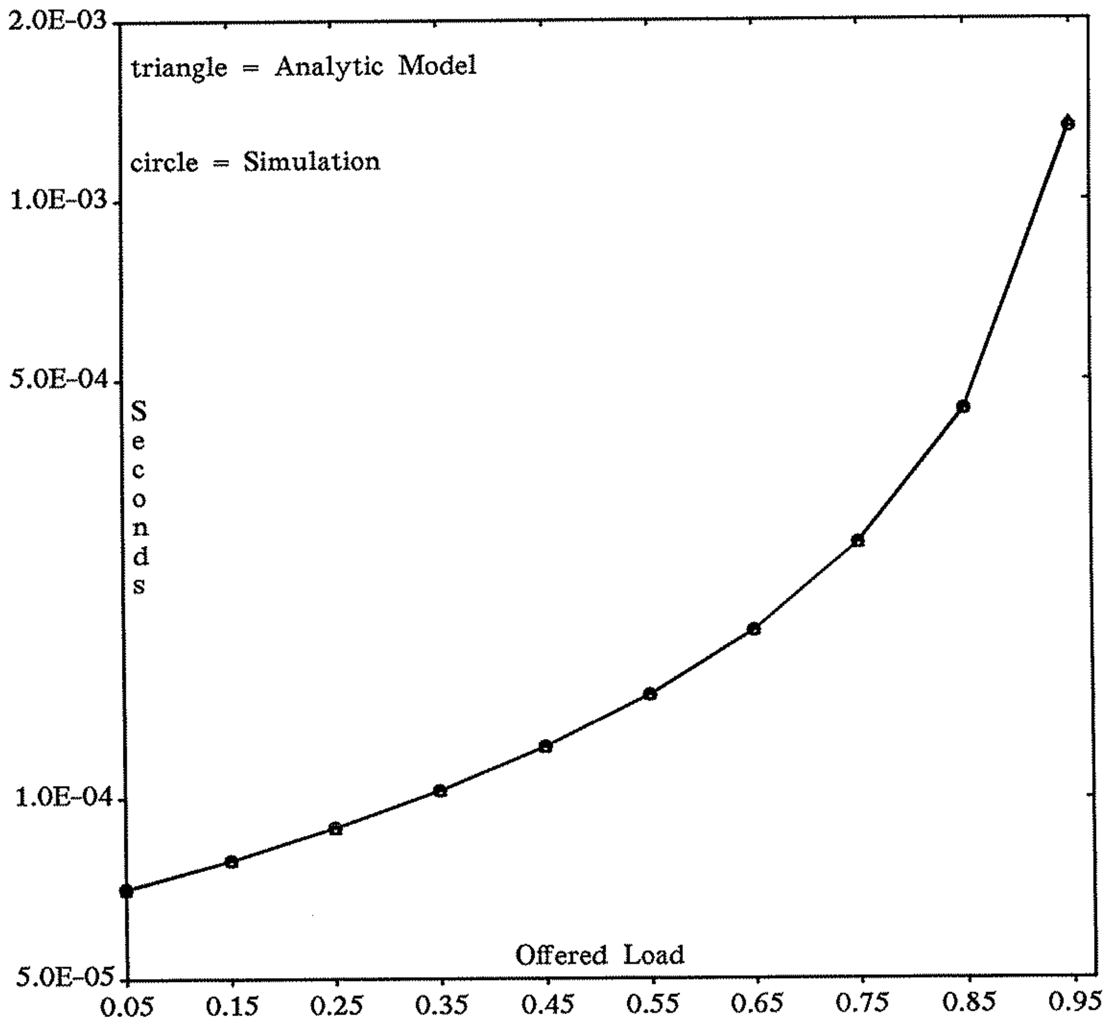
Four measures of delay are reported (queuing delay, network access delay, station delay, and service delay), of which station delay and service delay are given by the mathematical model of the previous chapter. Figure 4.5 shows the queuing delay for the reference configuration. The maximum value is 762 μs. Note that even though the delay for 0.05 and 0.15 offered load is reported to be 1 μs, both are actually considerably less. However, we consider delays of less than



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.1 — Token Cycle Time

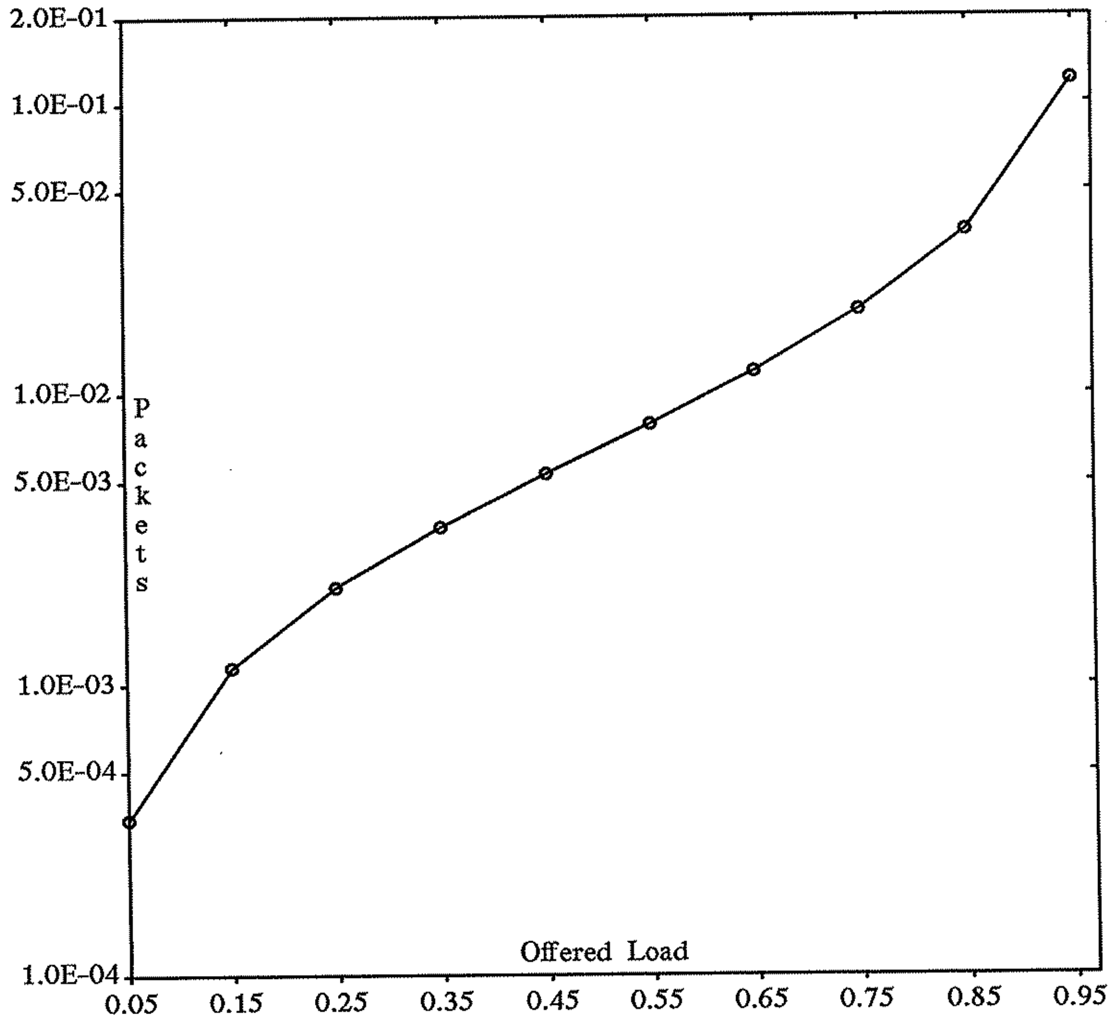


Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.2 — Token Cycle Time

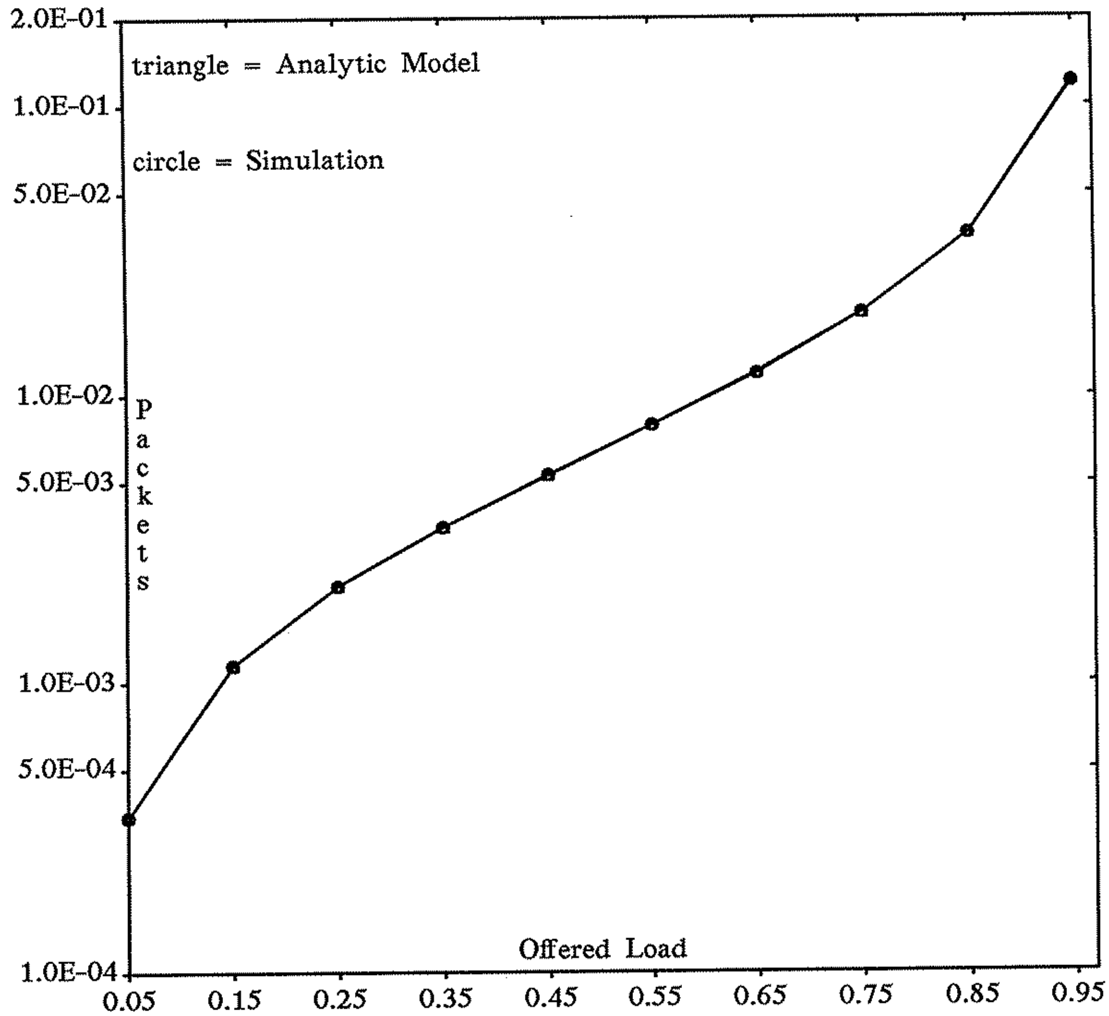
Simulation vs. Analytic Model



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.3 — Queue Lengths

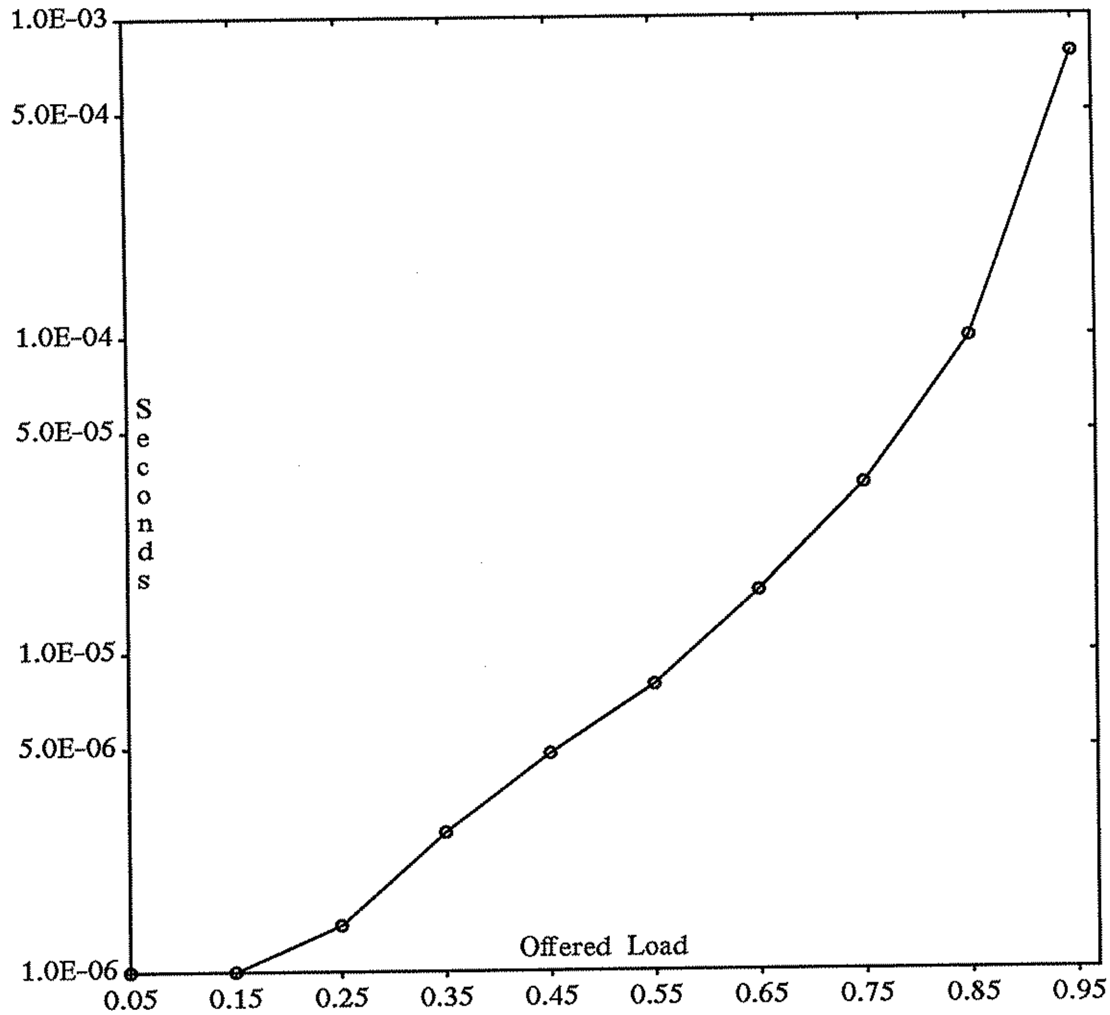


Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.4 — Queue Lengths

Simulation vs. Analytic Model



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.5 — Queuing Delay

1 μs to be essentially zero, and will show them as being either zero or the minimum positive value on the graph.

Network access delay is given in Figure 4.6. It varies from 40.7 μs to 2.15 ms, and greatly outweighs queuing delay, even at high loads.

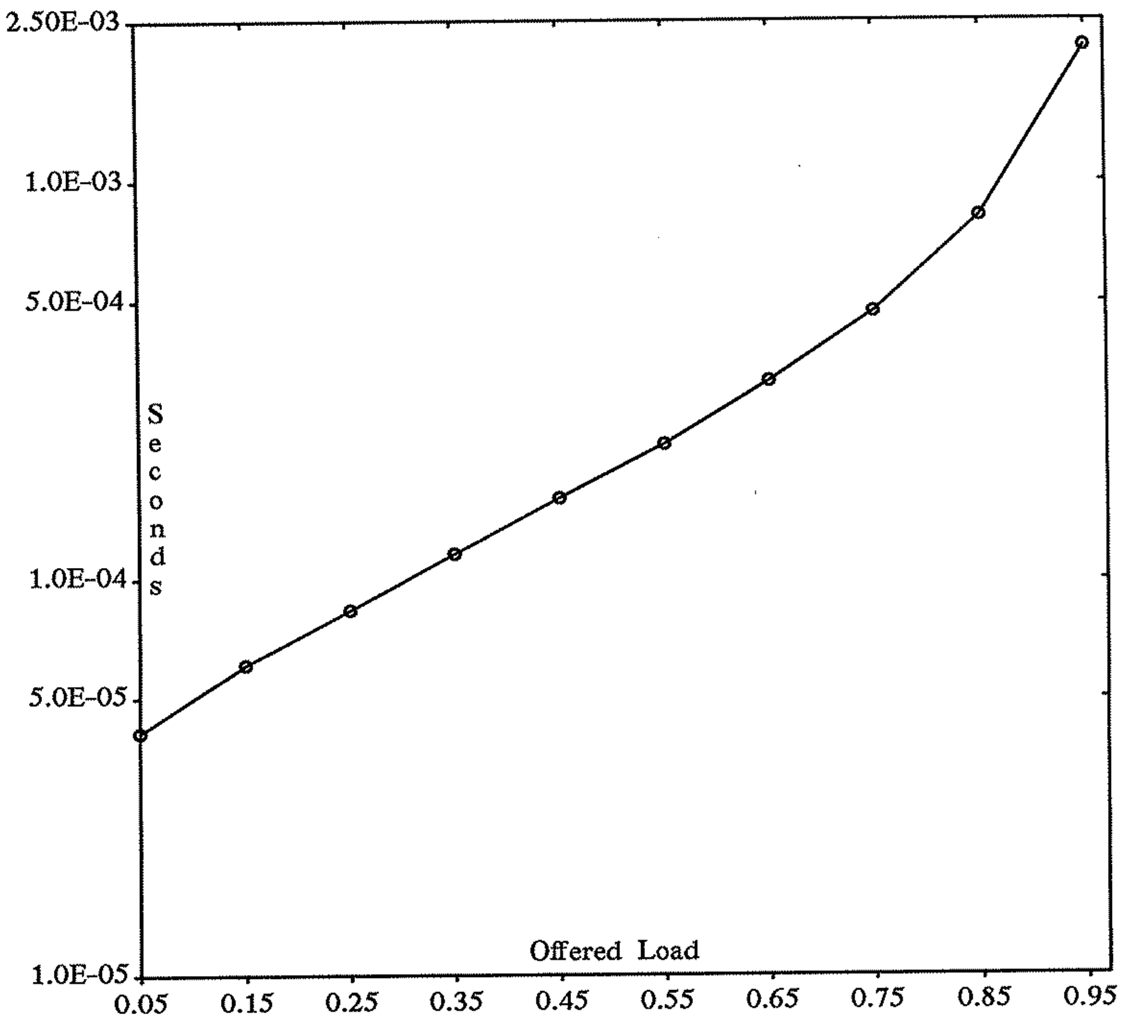
Station delay, which is the sum of queuing delay and network access delay, is given in Figure 4.7. The low and high values are 40.8 μs and 2.91 ms respectively. This compares to the predicted station delay, which has a low value of 42.0 μs, and a high value of 3.09 ms. Figure 4.8 shows the comparison of the two graphs. Service delay is the total delay experienced by a packet from the time it is enqueued until its last bit is delivered at the receiving station (see Chapter 3). Its graph is shown in Figure 4.9, with minimum and maximum values of 318 μs and 3.18 ms.

The graph of predicted values vs. measurement for service delay is shown in Figure 4.10, and obviously shows the most variance between the simulation and mathematical model. However, both the simulator and the queuing model treat service delay as the sum of various quantities, and any errors are cumulative. The minimum and maximum values of the predicted service delay are 332 μs and 3.38 ms.

4.1.1.3. Throughput, Tokens, and Packets

The throughput of the reference configuration is linear through 0.95 offered load, with a slope of approximately 0.967. Throughput varies from 0.0479 to 0.918, and is shown in Figure 4.11.

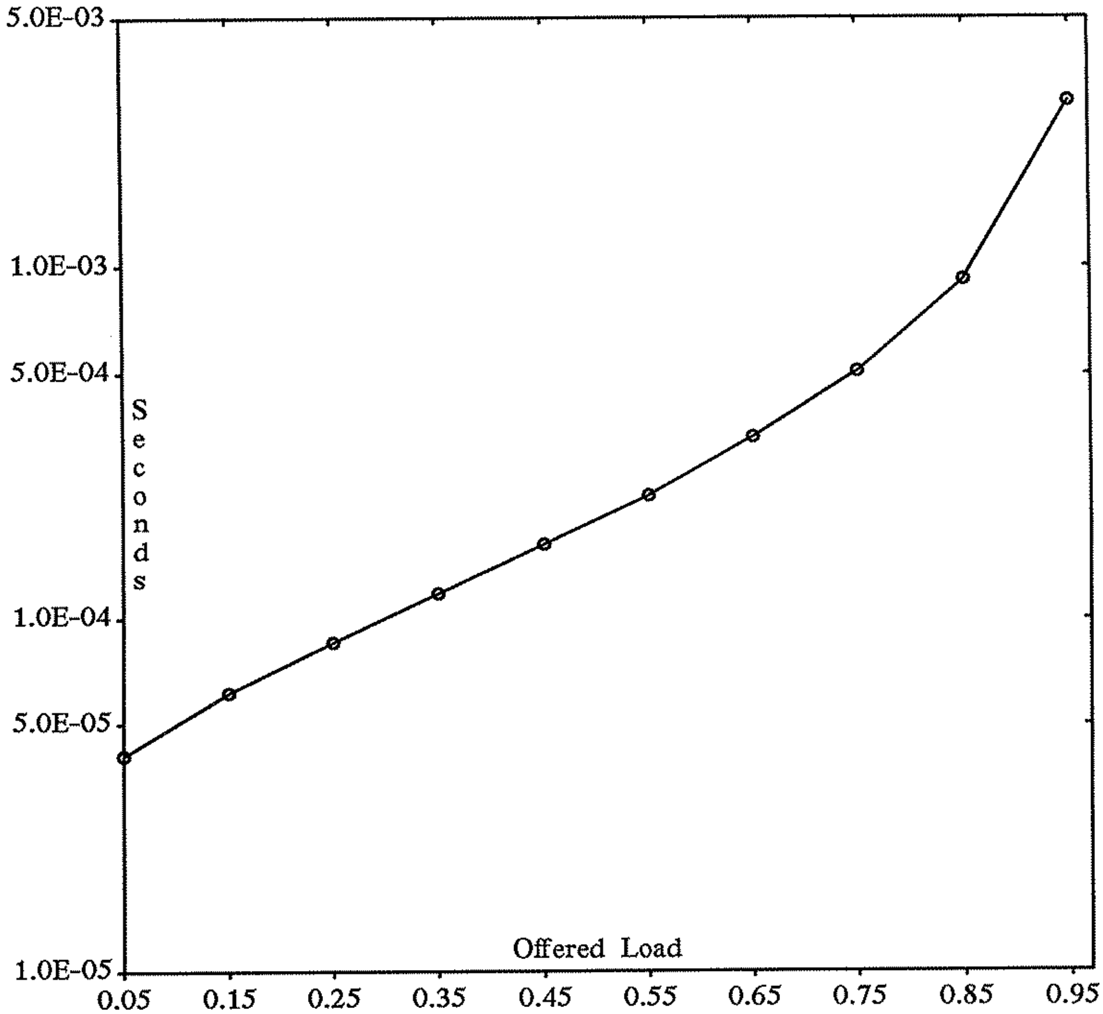
Figure 4.12 shows the number of tokens received per station per second, along with the number of tokens accessed per station per second. The curve showing tokens received varies from 14100 down to 728, while the curve of the number



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

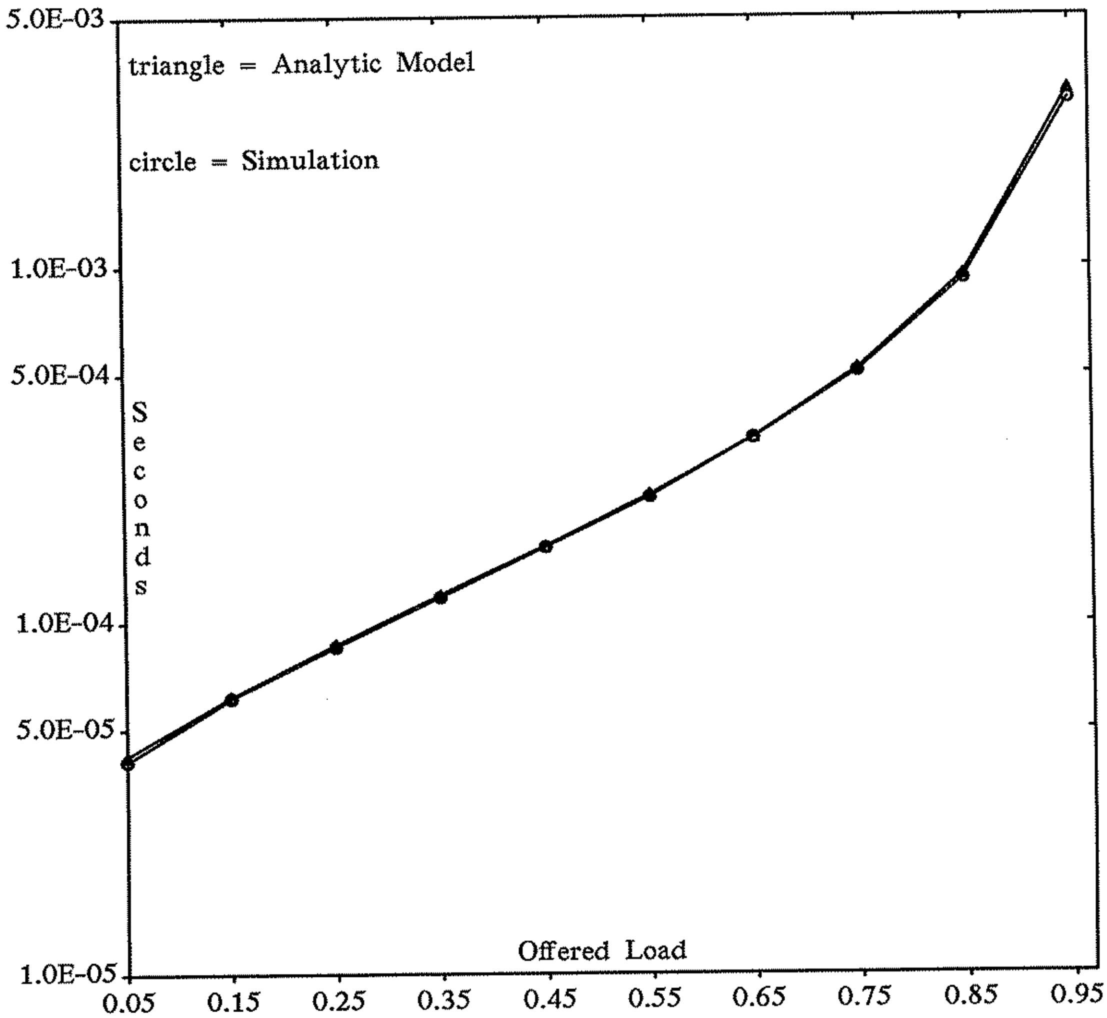
Figure 4.6 — Network Access Delay



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.7 — Station Delay

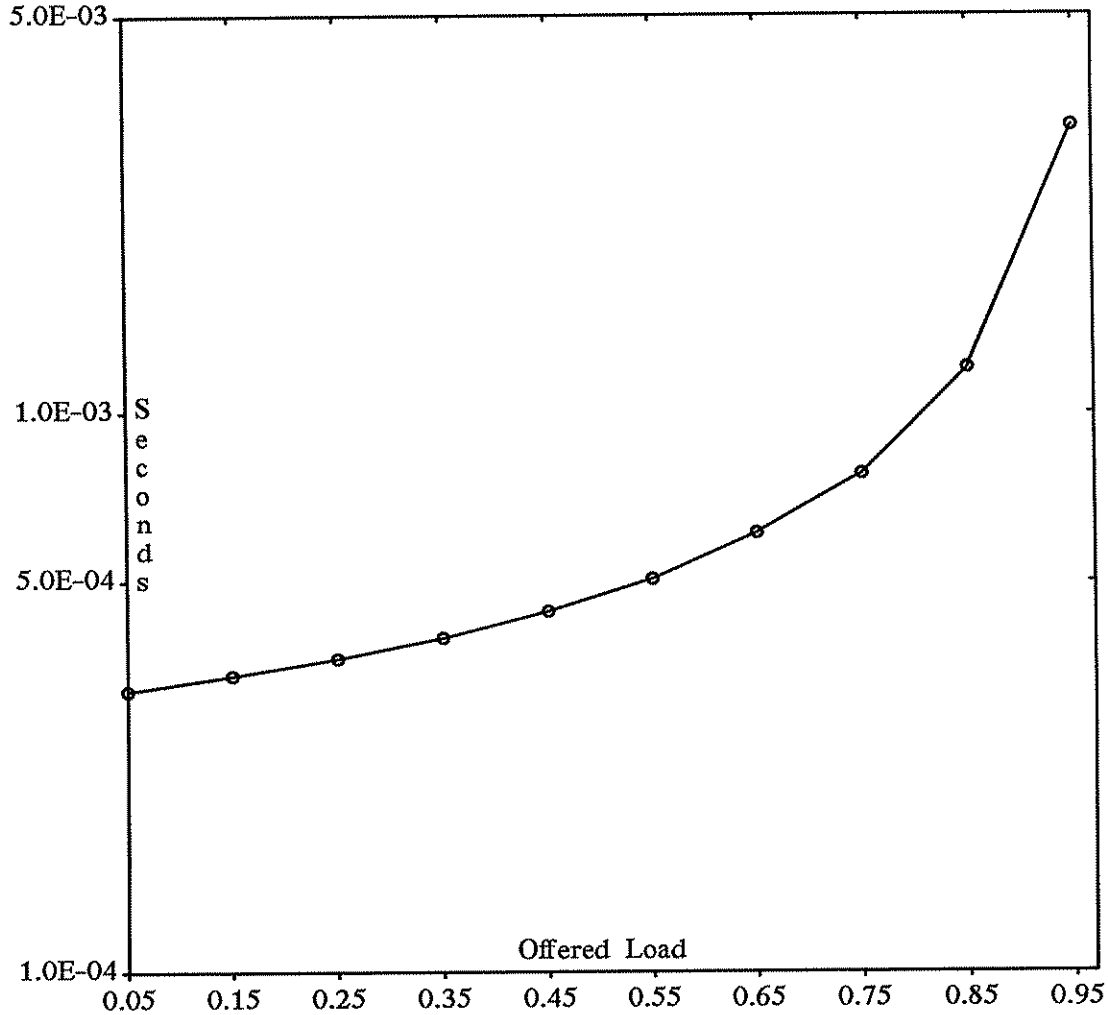


Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.8 — Station Delay

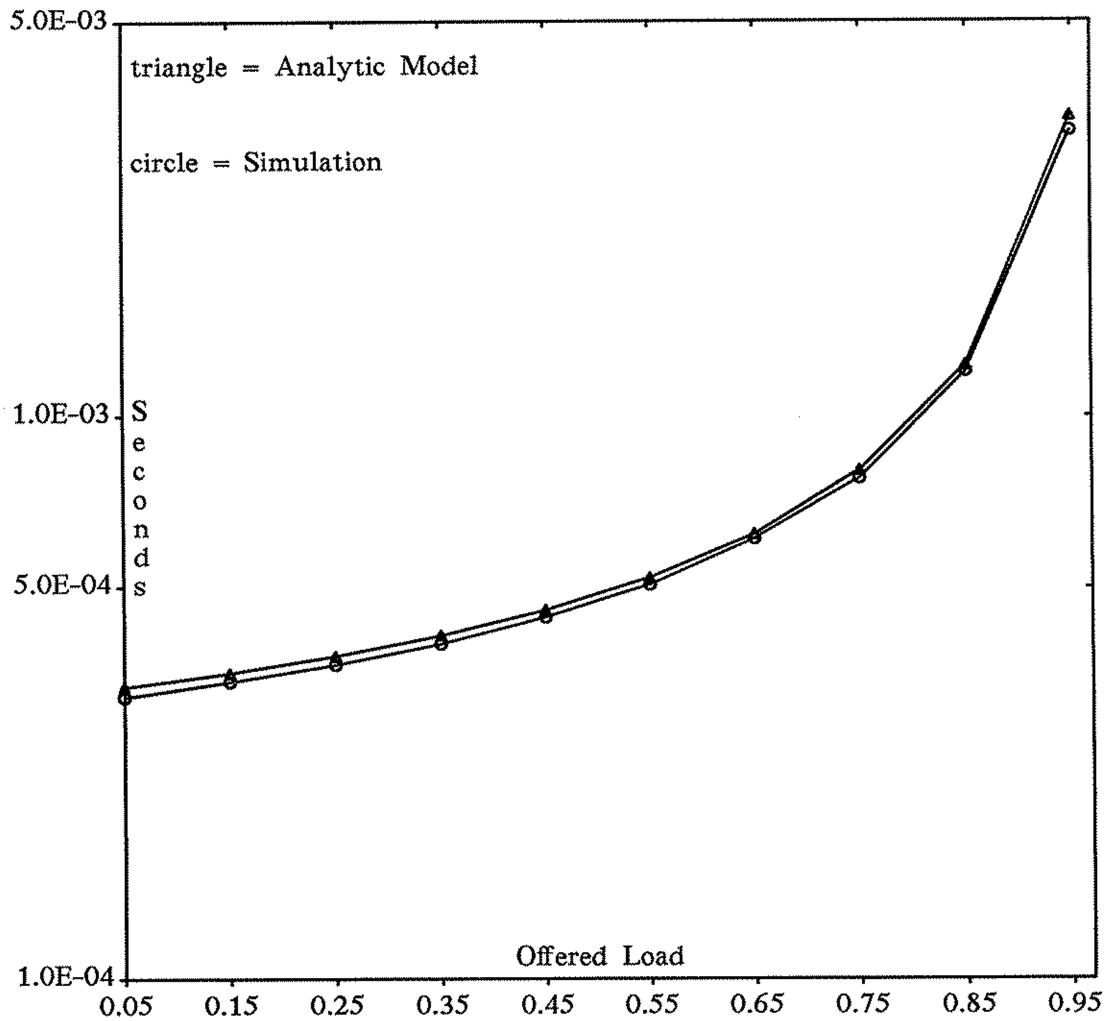
Simulation vs. Analytic Model



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.9 — Service Delay

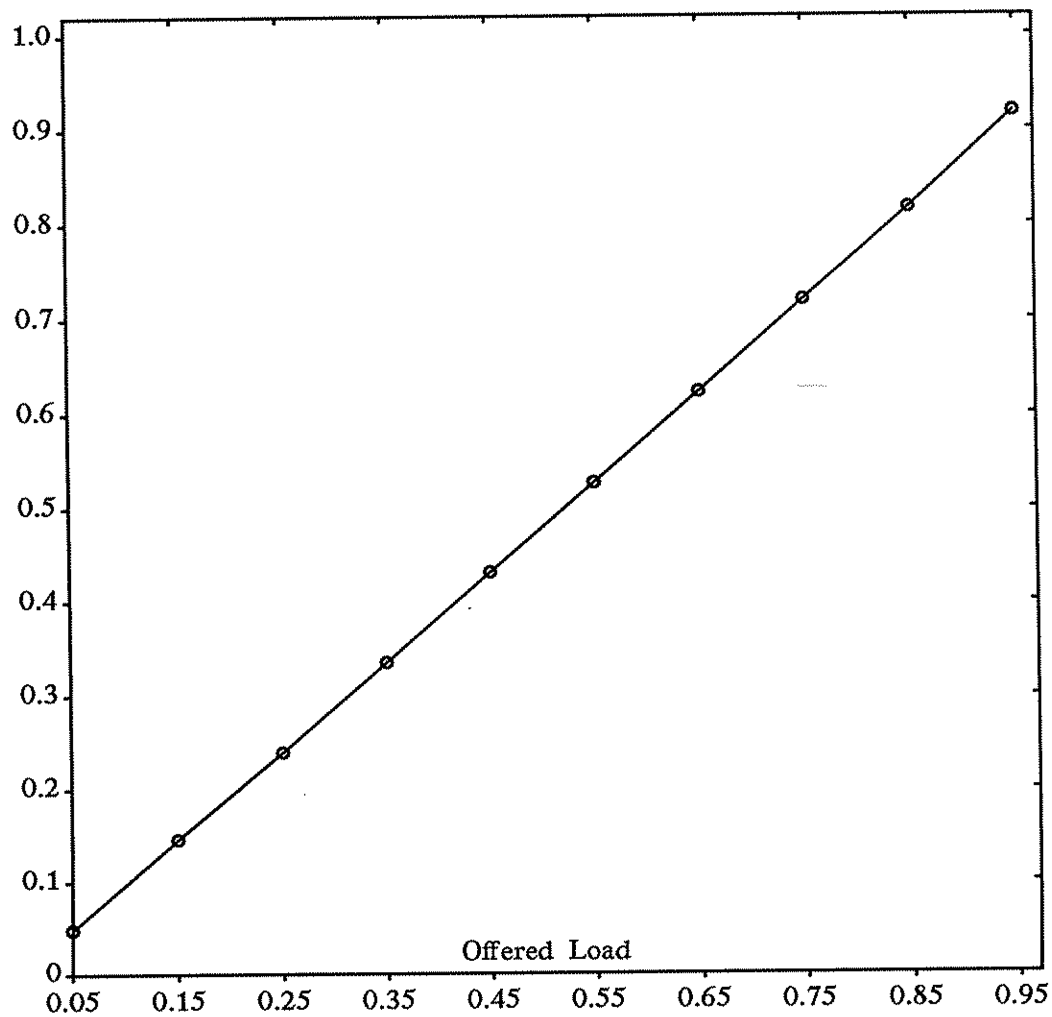


Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.10 — Service Delay

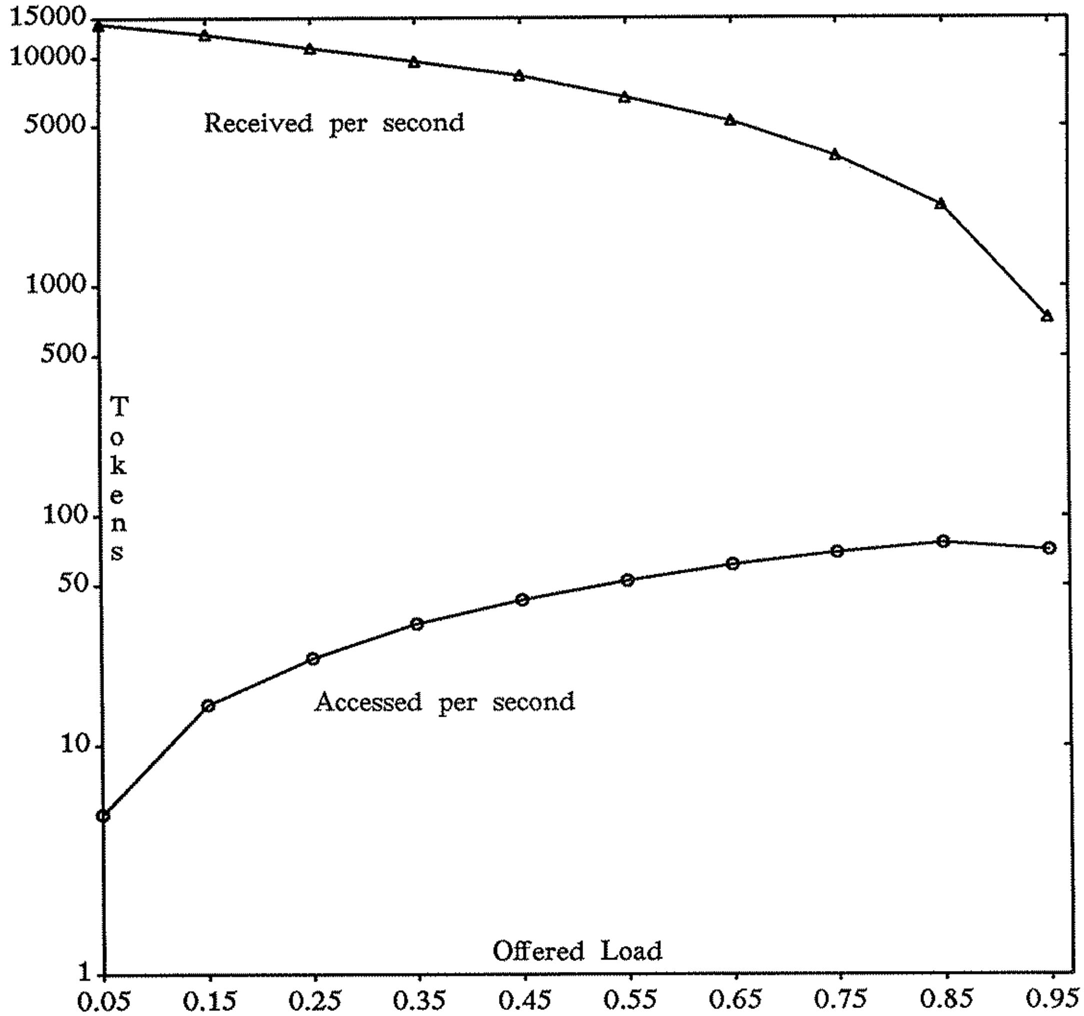
Simulation vs. Analytic Model



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.11 — Throughput



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.12 — Token Traffic

accessed goes from 4.8 to 71.2, with a maximum value of approximately 76 tokens accessed per second. Although not shown, if offered load is sufficiently high so that the Token_Holding_Timers of the stations are completely exhausted at each transmission, the curve showing tokens received and the curve showing tokens accessed will meet at a value of about 2.5 tokens received and accessed per station per second.

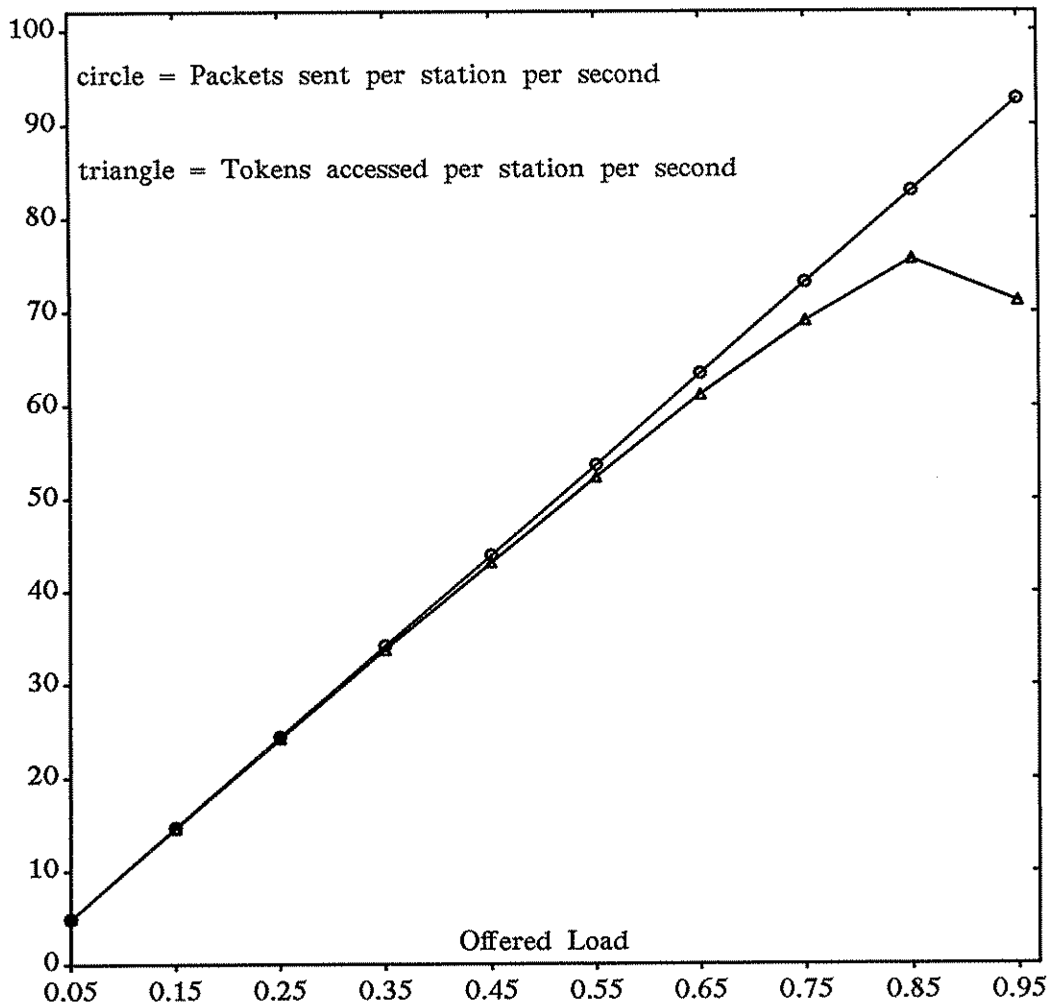
A graph depicting the number of tokens accessed per station per second (see Figure 4.12) along with the number of packets sent per station per second is given in Figure 4.13. The graph of packets sent is linear through 0.95 offered load, with a slope of 0.976, and varies from 4.8 to 92.7 packets per station per second. This graph will level off at about 98 packets per station per second when the load is sufficiently high.

4.1.2. Effects of Varying the Number of Stations

This section presents the results obtained by varying the number of ring stations. Note that the only parameter that has been varied is the number of active stations. Simulations were run for configurations of 3, 5, 10, 20, 40, 80, 120, 160, 200, 225, and 250 stations (250 stations is the maximum number allowed by the standard). The results of most interest are token cycle time, station delay, queue lengths, and throughput.

4.1.2.1. Token Cycle Time and Delay

The token cycle time for the ring is very close to the values predicted by the analytic model (see Chapter 3) until configurations are reached that combine a high offered load with a large number of stations. Under these conditions the values given by the simulation are greater than those given by the analytic model. This is explained by the actions taken by the protocol at the end of packet transmis-



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

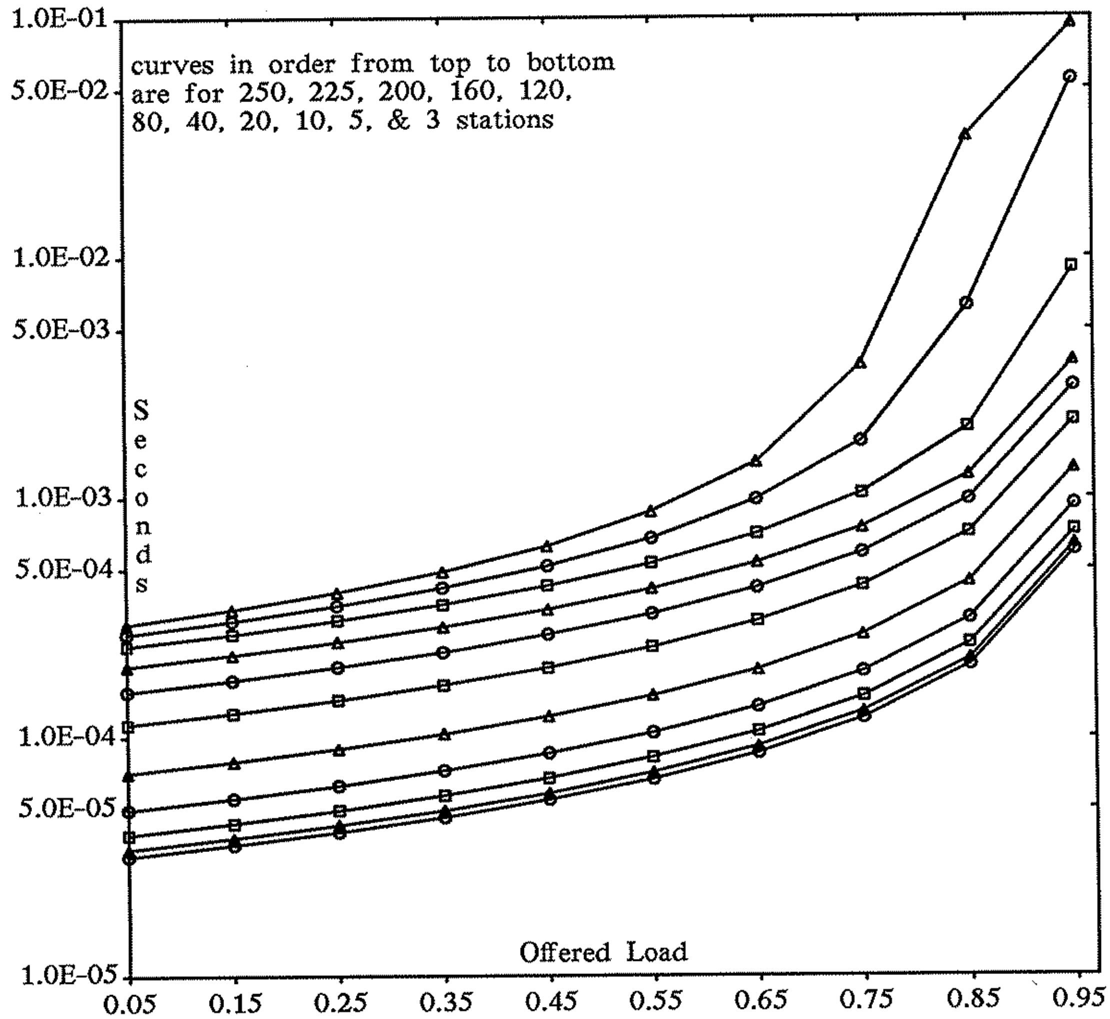
Figure 4.13 — Packets vs. Tokens

sion.

As a station transmits packets, it is absorbing the same packets as they return. While doing so, it examines certain fields in the packet framing, among them the source address of the packet (see Section 2.7). The standard states that the station must wait for the source address field of a packet to arrive before it may retransmit the token (up to the limit of the Return_to_Repeat_Timer — see Section 2.4). This has the effect of changing the configuration of the network by adding to the average ring latency, and it is this added latency in the ring that is not accounted for by the analytic model.

Another interesting effect is seen in the slope of the graph of 250 stations. At about 0.85 offered load, the curve changes shape from concave upwards to concave downwards. This is an expected effect. In simulations not shown, the curve eventually flattens at the quantity given by Equation 1 in Chapter 3 (maximum token cycle time). What is being seen is the effect of the Token_Holding_Timer (see Chapter 2) occasionally limiting the transmission of packets at a station (residual queue lengths are shown in simulation reports to be non-zero for high offered loads and a large number of stations).

Figure 4.14 shows the token cycle time for varying numbers of stations. For 250 active ring stations, the token cycle time varies from about 300 μs to about 93 ms. For 3 stations, it varies from about 32 μs to 600 μs. The station delay for packets has many of the same characteristics as the token cycle time. This is because the cycle time for the ring has a direct effect on the delay of a packet. Therefore, when the token cycle time increases rapidly, so does the station delay. Notice too the change from concave upwards to concave downwards of the graph for 250 stations. This shows a direct effect of the cycle time on this measure of delay. However, the station delay of a packet will not reach some maximum



Configuration:

Varying Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.14 — Token Cycle Time
Varying the Number of Stations

value as will the token cycle time; there is no timer-controlled limit on delay (or queue lengths) as there is on token cycle time.

Station delay can be seen in Figure 4.15. It varies from 21 μs to 2.5 ms for 3 stations, and from 157 μs to 73 ms for 250 stations.

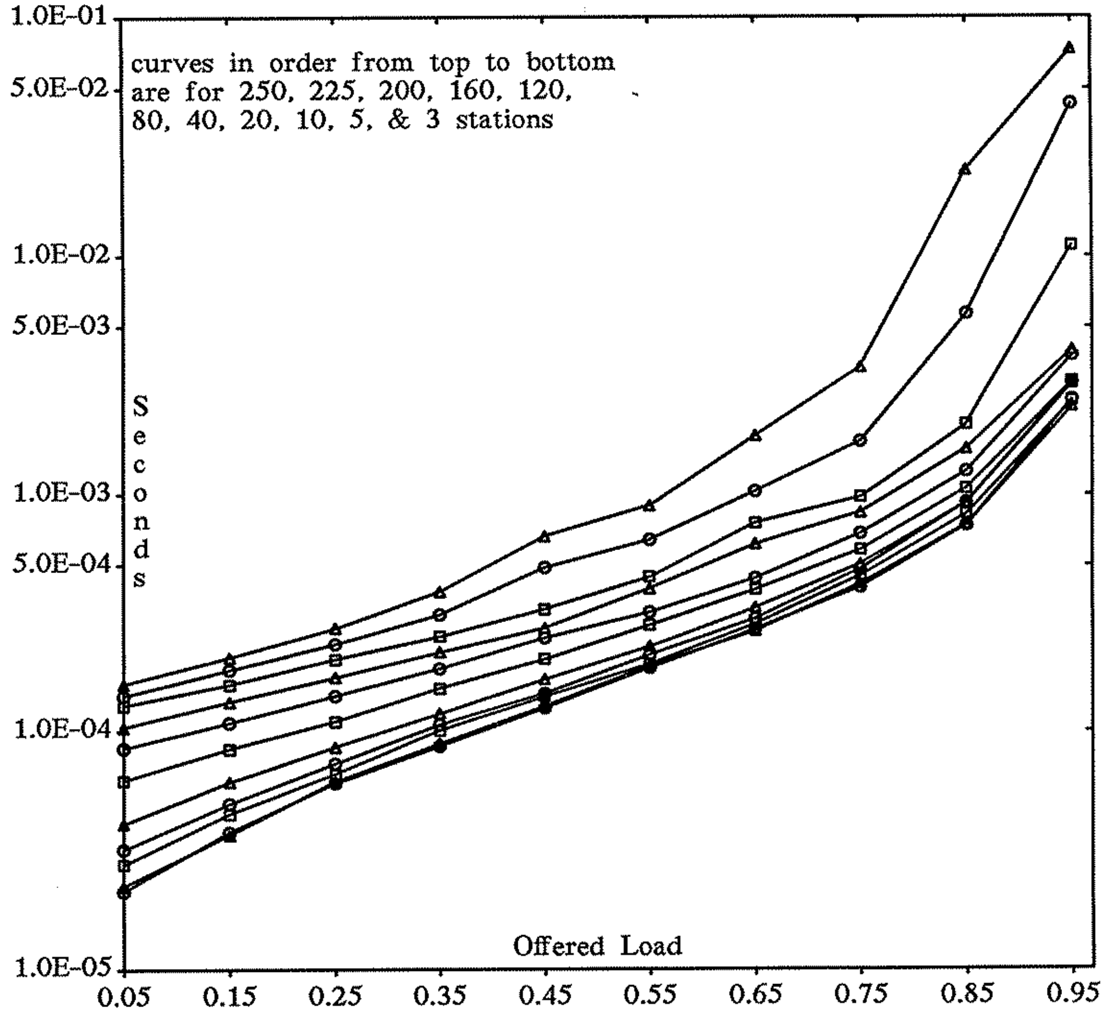
4.1.2.2. Queue Lengths

Queue lengths have the interesting (and expected) property of growing in length as the number of ring stations becomes smaller. The reason for this is that as the number of stations grows smaller, in order for the offered load to remain the same, each station contributes a larger and larger share, thus decreasing interarrival times for packets. Figure 4.16 shows queue lengths for 3 through 250 stations.

Queue lengths also increase for 200, 225, and 250 stations due to the effect of the increased latency of the ring. The limiting effect of the Token_Holding_Timer can be seen in the change from concave upwards to concave downwards in the graph for 250 stations. Queue lengths for 3 stations vary from about 0.0023 to 0.51 packets. For 250 stations, queue lengths go from 0.00023 to 1.34 packets.

4.1.2.3. Throughput

Throughput is the least affected of the measures reported here. As can be seen in Figure 4.17, there is almost no variation in throughput until an offered load of 0.55 is reached (the variation at 0.05 offered load is less than 0.0005). At an offered load of 0.95, the throughput for 250 stations is about 0.904, and for 3 stations is about 0.942, a variation of less than 0.04. Note that a throughput of 0.9 is extremely high, even for a local area network.

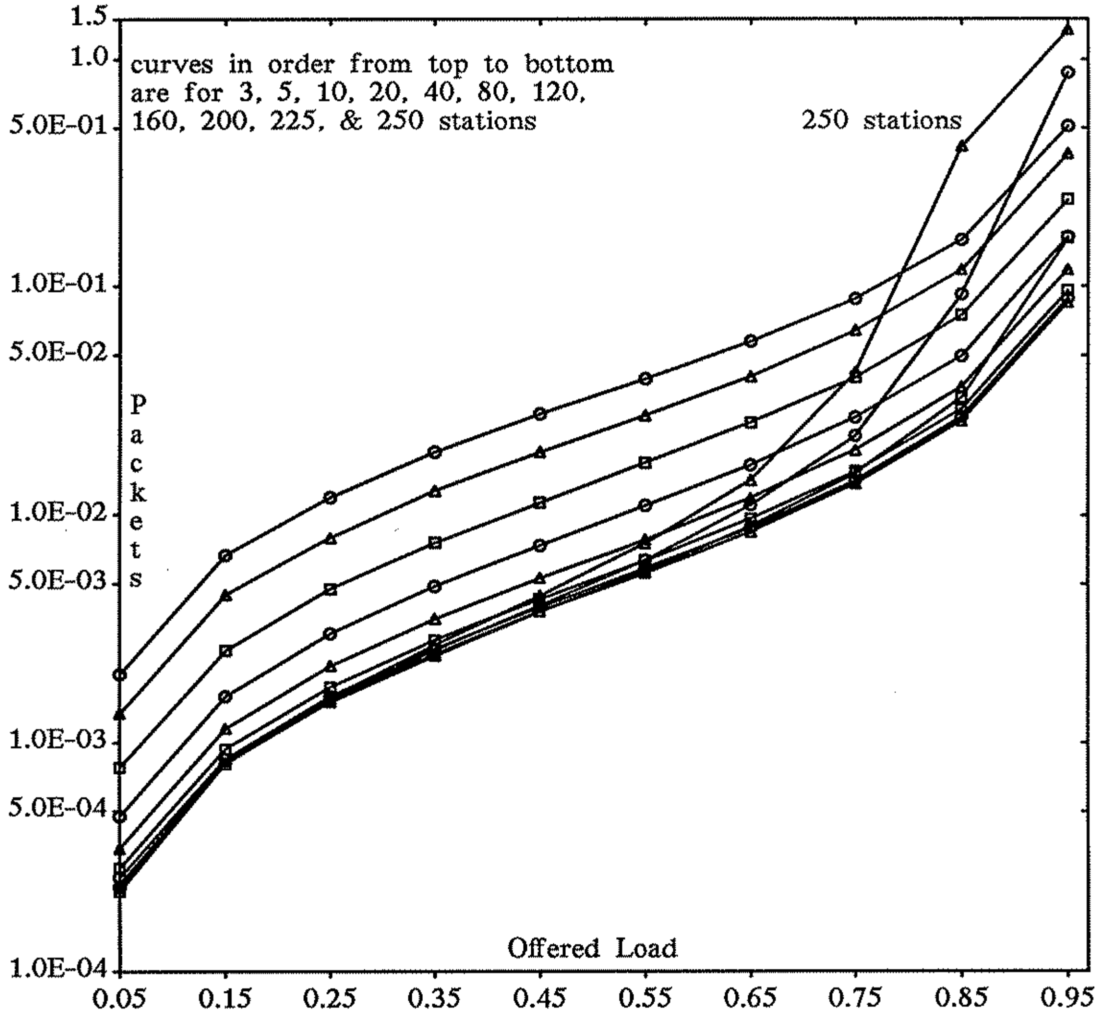


Configuration:

Varying Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.15 — Station Delay

Varying the Number of Stations

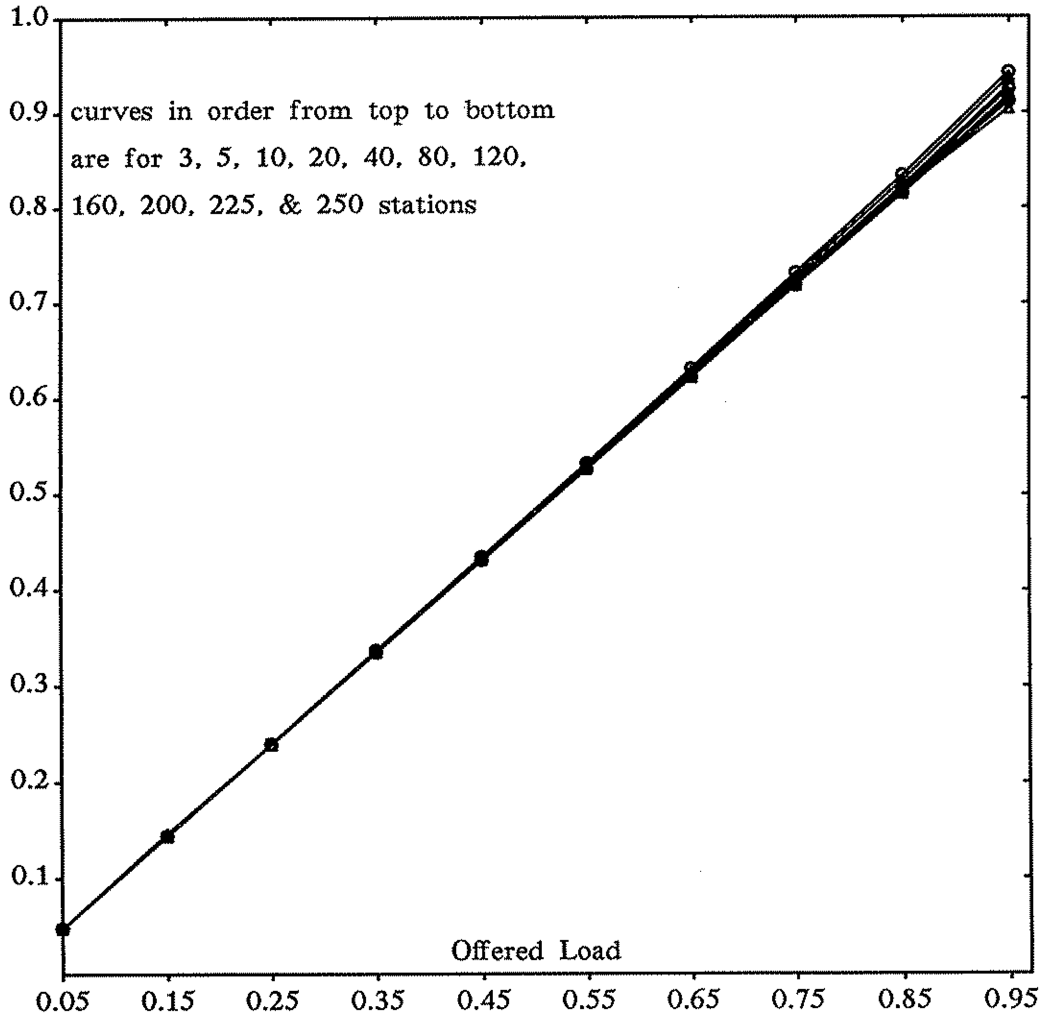


Configuration:

Varying Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.16 — Queue Lengths

Varying the Number of Stations



Configuration:

Varying Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.17 — Throughput

Varying the Number of Stations

4.1.3. Effects of Varying Packet Lengths

This section presents the results obtained by varying packet lengths. Simulations were run using packet lengths of 104 (the minimum possible), 256, 512, 1024, 2048, 4096, and 8192 bit packets. The packet length is the only parameter varied.

4.1.3.1. Delay and Queue Lengths

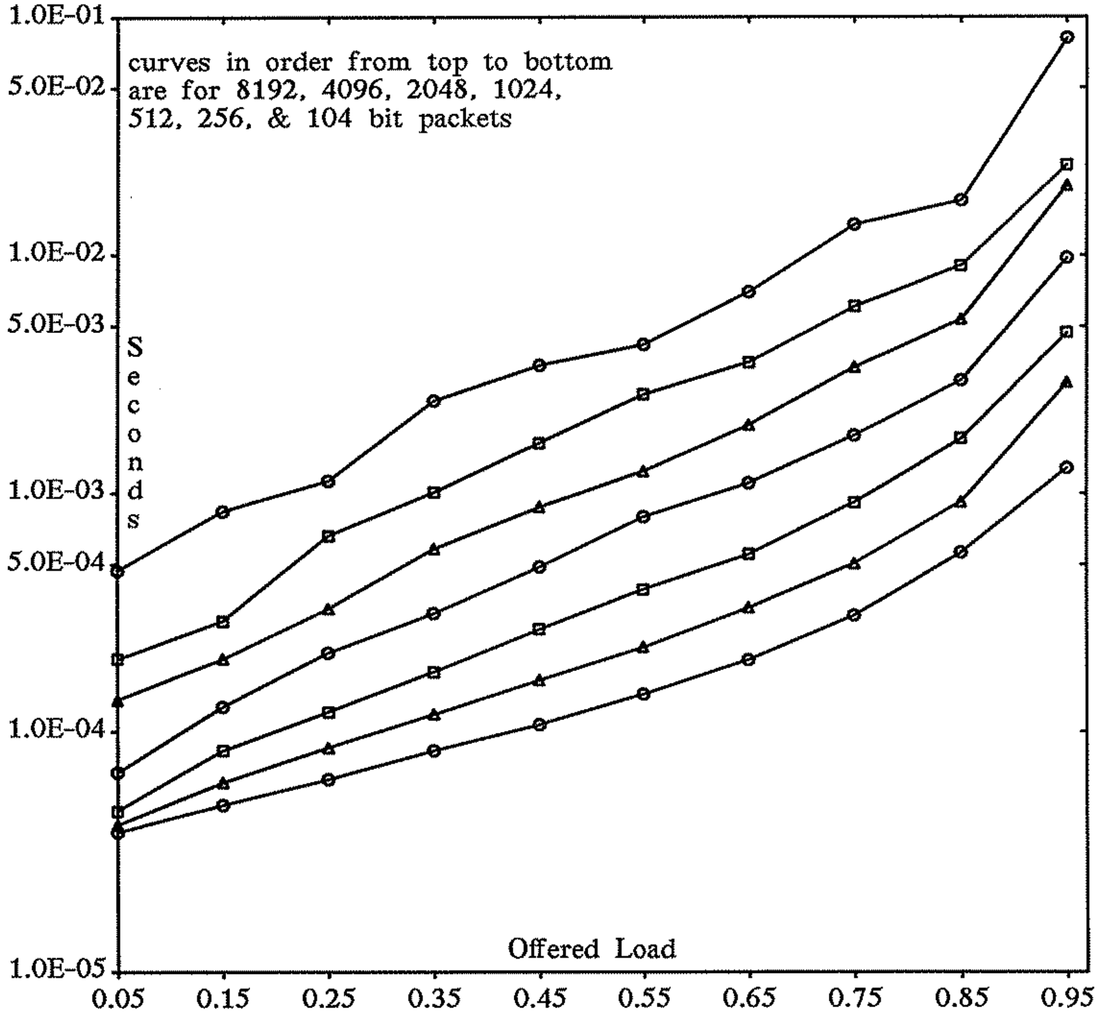
The effect of packet length on station delay can be seen in Figure 4.18. Note that doubling the packet length also increases the delay by about a factor of two. The delay for 104 bit packets varies from approximately 38 μs to about 1.28 ms, and the delay for 8192 bit packets goes from 470 μs to 82 ms.

The opposite effect can be seen in Figure 4.19, showing queue lengths. As the packet size doubles, the queue length (measured in packets) is cut approximately in half. Queue lengths for 104 bit packets range from 0.00086 to 0.30 packets. Those for 8192 bit packets vary from 0.000010 to 0.0039 packets.

The above results obtained by varying the packet length are expected. Note that even though the mean station delay doubles when the packet size does, this delay is being measured for packets. If the "delay per bit" is considered, then it is readily seen that there is no significant change in delay. Much the same argument applies for queue lengths. Since queue lengths are measured in packets, if the packet size is doubled, then the queue length is halved, thereby leaving the same number of bits enqueued.

4.1.3.2. Throughput

A rather large effect is seen on throughput when the packet size is varied. It can be seen in Figure 4.20 that as packet size increases, throughput also increases. This is explained by the fact that although the number of tokens received by a

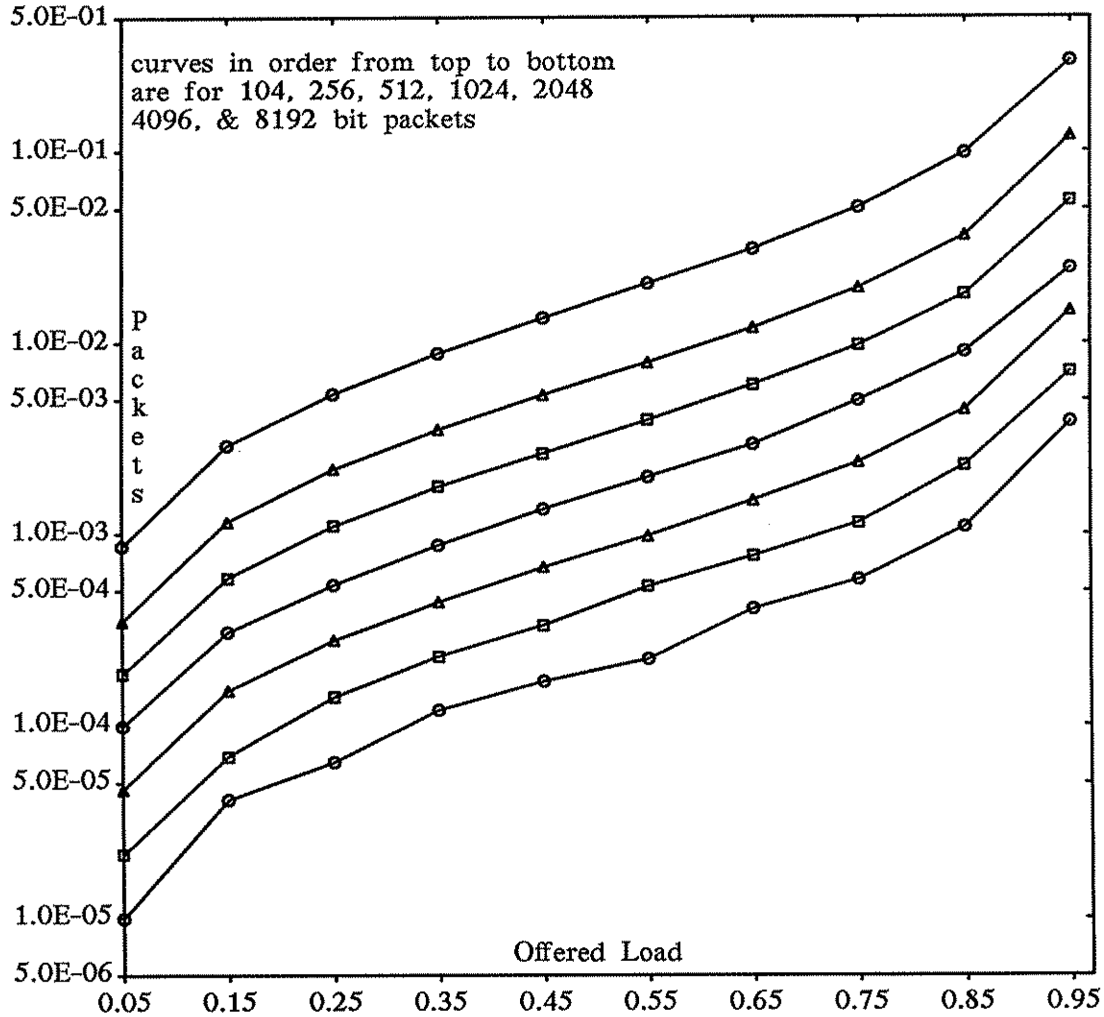


Configuration:

40 Stations
Varying Packet Lengths
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.18 — Station Delay

Varying Packet Size

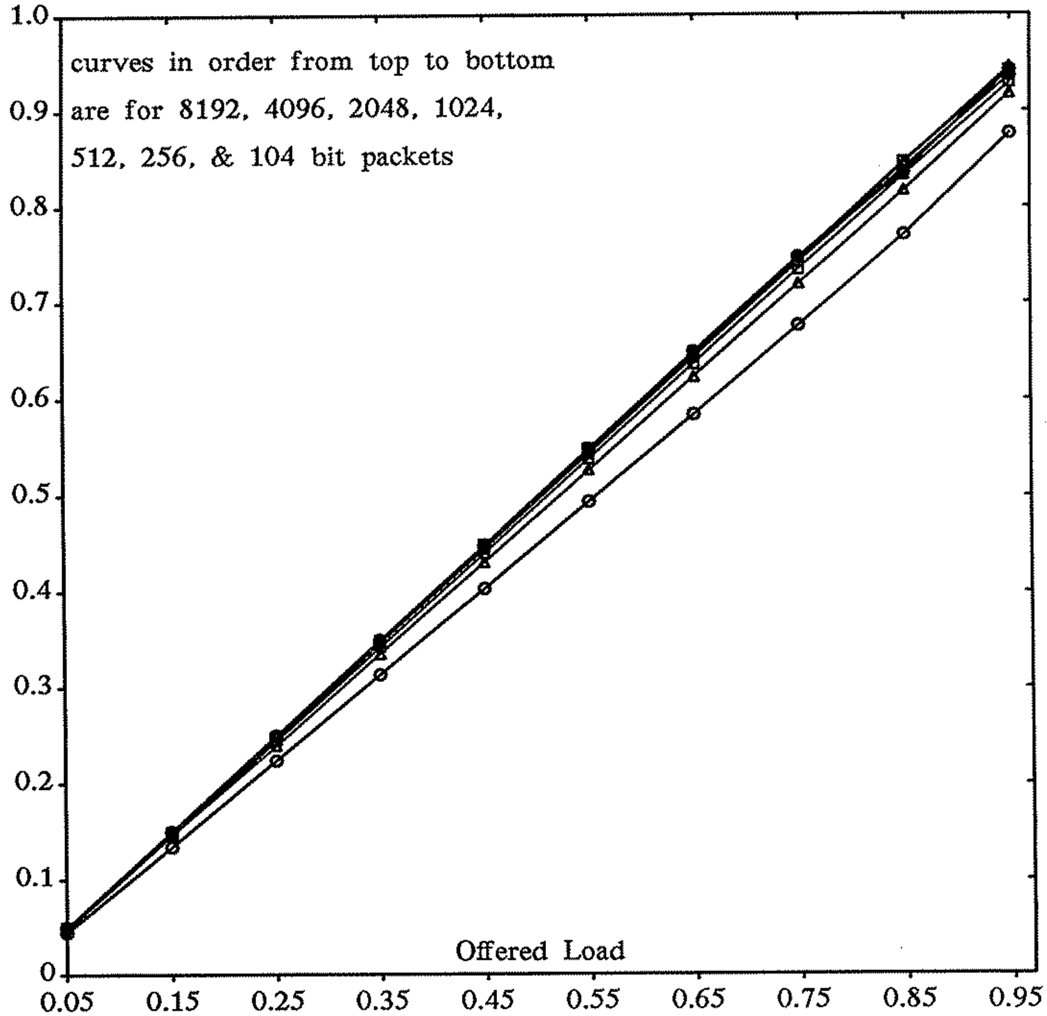


Configuration:

40 Stations
Varying Packet Lengths
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.19 — Queue Lengths

Varying Packet Size



Configuration:

40 Stations
Varying Packet Lengths
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.20 — Throughput

Varying Packet Size

station stays about the same regardless of packet size (since the token cycle time is unaffected), the number of tokens accessed is down considerably, resulting in fewer tokens used per bit delivered. The fewer tokens used to deliver a bit has the effect of raising throughput.

This same effect can be seen in all throughput curves. When throughput and the number of tokens accessed per second are examined together, it is noticed that as the curve for token accessed per second reaches its maximum, the slope for the throughput curve changes from slightly less to slightly greater than 1 (but always staying less than the number of total bits offered to the network). Throughput varies from 0.0447 to 0.876 for 104 bit packets, and from 0.499 to 0.942 for 8192 bit packets.

4.1.4. Arrival and Packet Length Distributions

Typical assumptions made when dealing with network performance are that packet arrivals and packet lengths have exponential distributions. However, this is not always the case, so various simulations were made using different arrival and packet length random distributions. The distributions used were exponential, uniform random, and constant, for both arrivals and packet lengths. Each distribution was paired with each of the other three, and simulation results show that varying the arrival rate distribution has no effect on the measurements being taken. Varying the distribution of packet size also has no impact on performance.

One exception to the above statement concerning effects on performance was observed in connection with constant packet arrivals. If arrivals are such that packets arrive at all stations simultaneously, then mean delay increases by approximately a factor of two. However, there is still no effect on token cycle time, queue lengths, throughput, etc.

4.2. Multiple Priority Operation

4.2.1. Reference Configuration

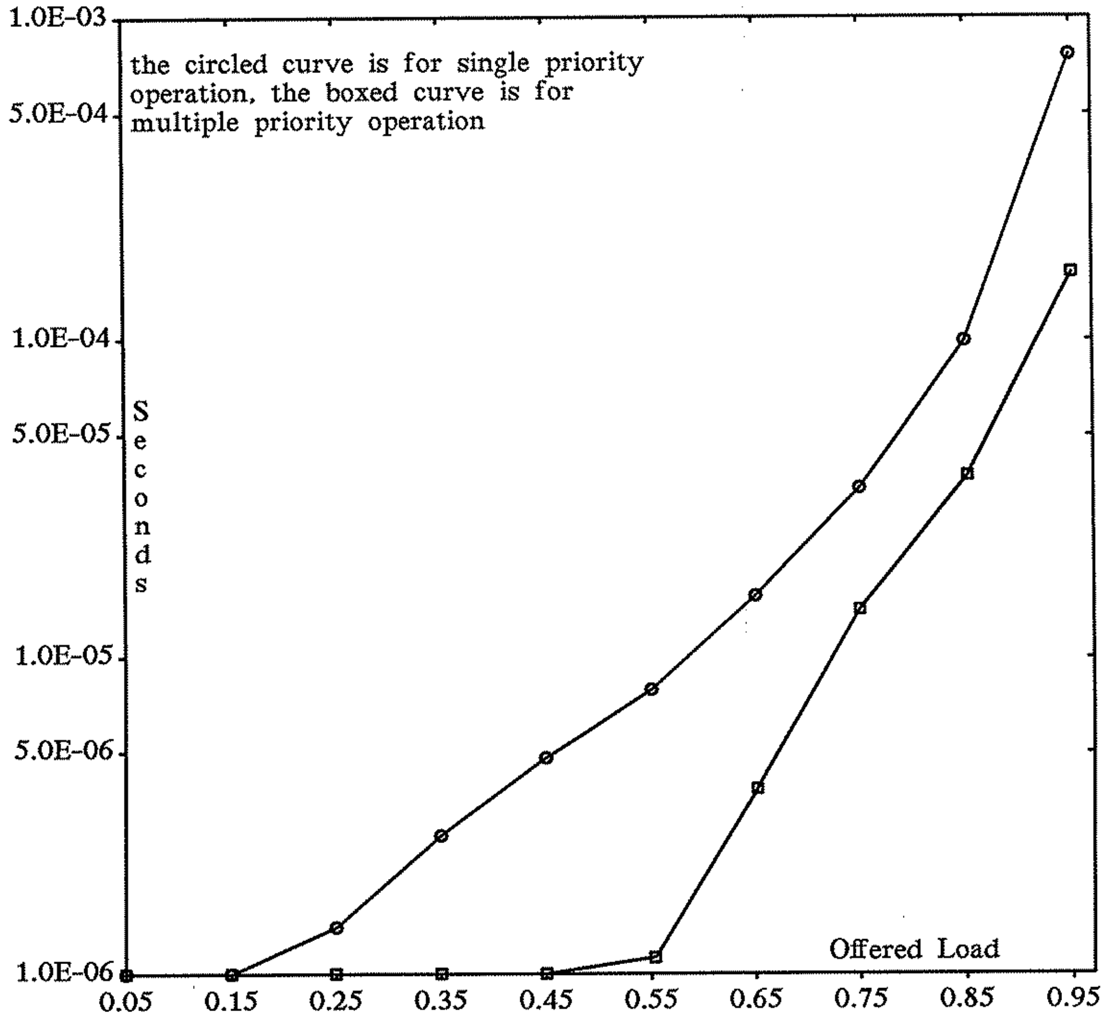
Heretofore we have examined the operation of the token ring under the condition that only a single priority of message is offered. The latter part of the chapter deals with the case where multiple priorities are being transmitted.

In order to examine the functioning of multiple priority operation of the token ring, we have established a reference configuration identical to that given in Section 4.1, with the exception that the offered load is divided equally among the eight priority levels allowed by the standard (see Section 2.1).

Interestingly, the performance of the token ring is virtually unaffected when multiple priority messages are offered (in the case of the reference configuration). Although the queue length at each priority's queue is less than that for the single priority operation, the sum of the eight queues is equal to the queue lengths for the single priority configuration (each queue is approximately one eighth the total). Token cycle time shows no difference at all. Also unaffected are throughput, token traffic, and the number of packets sent per tokens accessed.

4.2.1.1. Delay

The only noticeable effect seen in the operation of multiple priorities (in the reference configuration) is that mean delay experiences a slight increase (less than a factor of two), while queuing delay drops. Queuing delay can be seen in Figure 4.21. The drop in queuing delay is explained by the fact that packets are equally divided among the eight priorities, and each priority has its own queue. Note that not only is multiple priority queuing delay less than single priority queuing delay, but that the shape of the curve is different, being much flatter at low loads. The high value for multiple priority queuing delay is 160 μs.



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.21 — Queuing Delay

Multiple Priority Operation

Network access delay, station delay, and service delay can be seen in Figures 4.22, 4.23, and 4.24 respectively. All three are somewhat greater than the delays for the single priority configuration, the increase being much more apparent at medium to high loads. This is due to the added overhead of managing multiple priorities. Note that even though the difference in delay between single and multiple priority configurations seems to be large, it is actually not a significant increase in terms of the total delay experienced by a packet. It can be seen in the curve for service delay that the total delay for a packet is on the order of 3 ms for single and multiple priority operation.

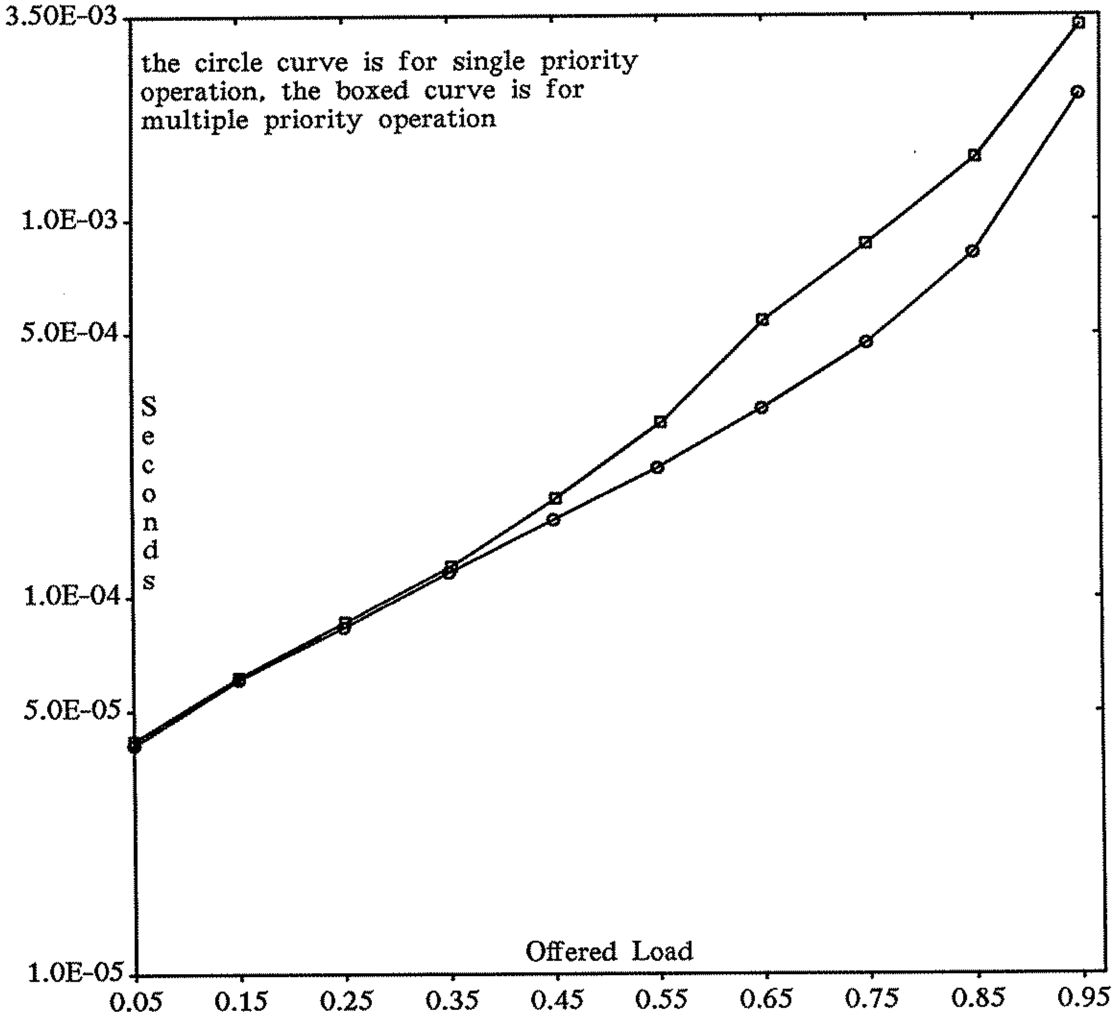
Network access delay ranges from 42 μs to 3.2 ms. Station delay goes from 42 μs to 3.5 ms, and service delay varies from 320 μs to 3.7 ms.

4.2.2. Effects of Varying the Number of Stations

Varying the number of active stations on the ring (simulations were run for configurations of 3, 5, 10, 20, 40, 80, 120, 160, 200, 225, and 250 stations) had no noticeable affect on performance except for very large and very small numbers of stations. For 200, 225, and 250 stations, the same increase in queue lengths was seen as for the single priority case (each priority's queue holding about one-eighth the total number of packets enqueued).

4.2.2.1. Delay

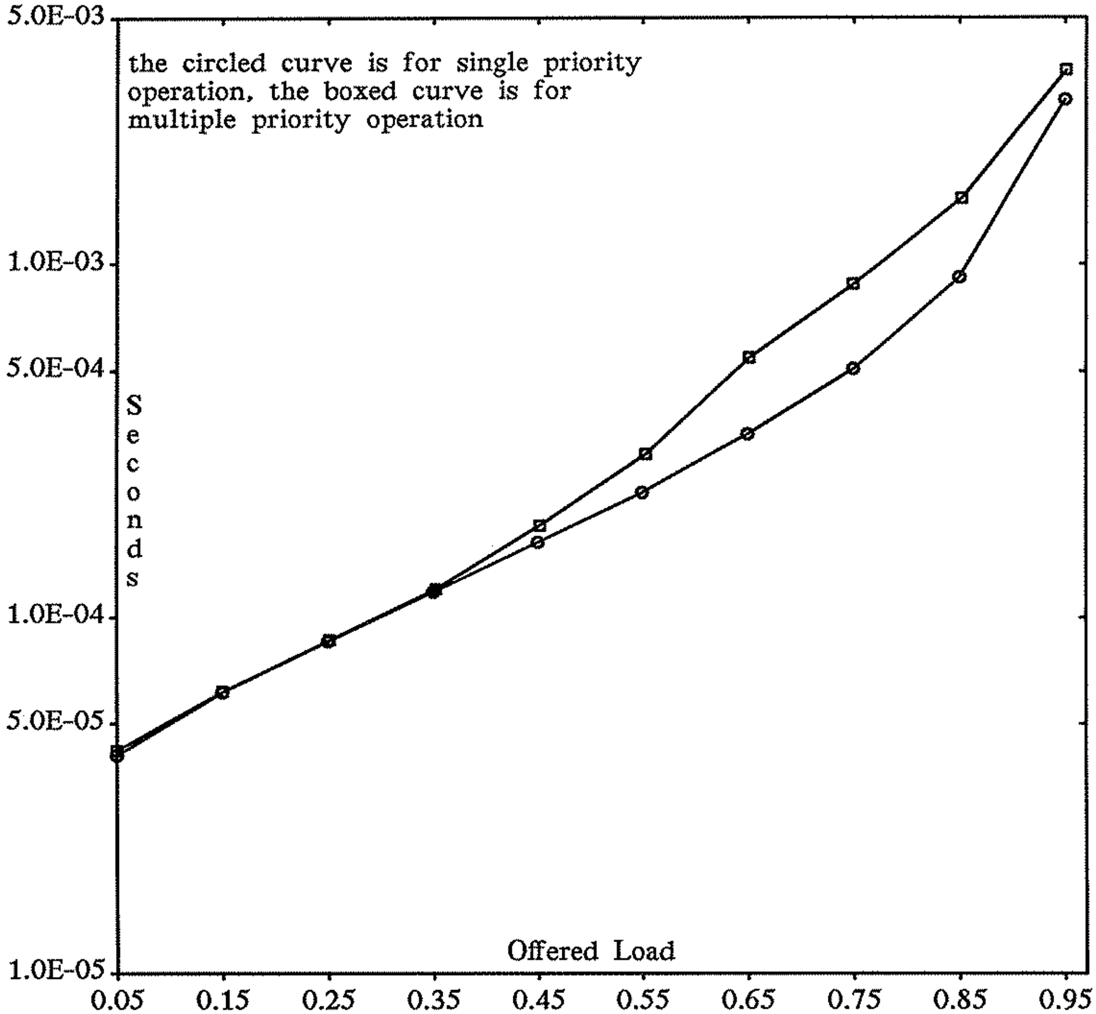
The effects seen for 3, 5, and 10 stations concern the delays experienced by the eight priorities. When the number of active ring stations is greater than approximately 10, the share of load for each station is low enough so that the effects of the different priorities are not noticeable. However, when the number of stations is about 10 or less, higher priority packets begin to experience less delay than lower priority packets.



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.22 — Network Access Delay
Multiple Priority Operation

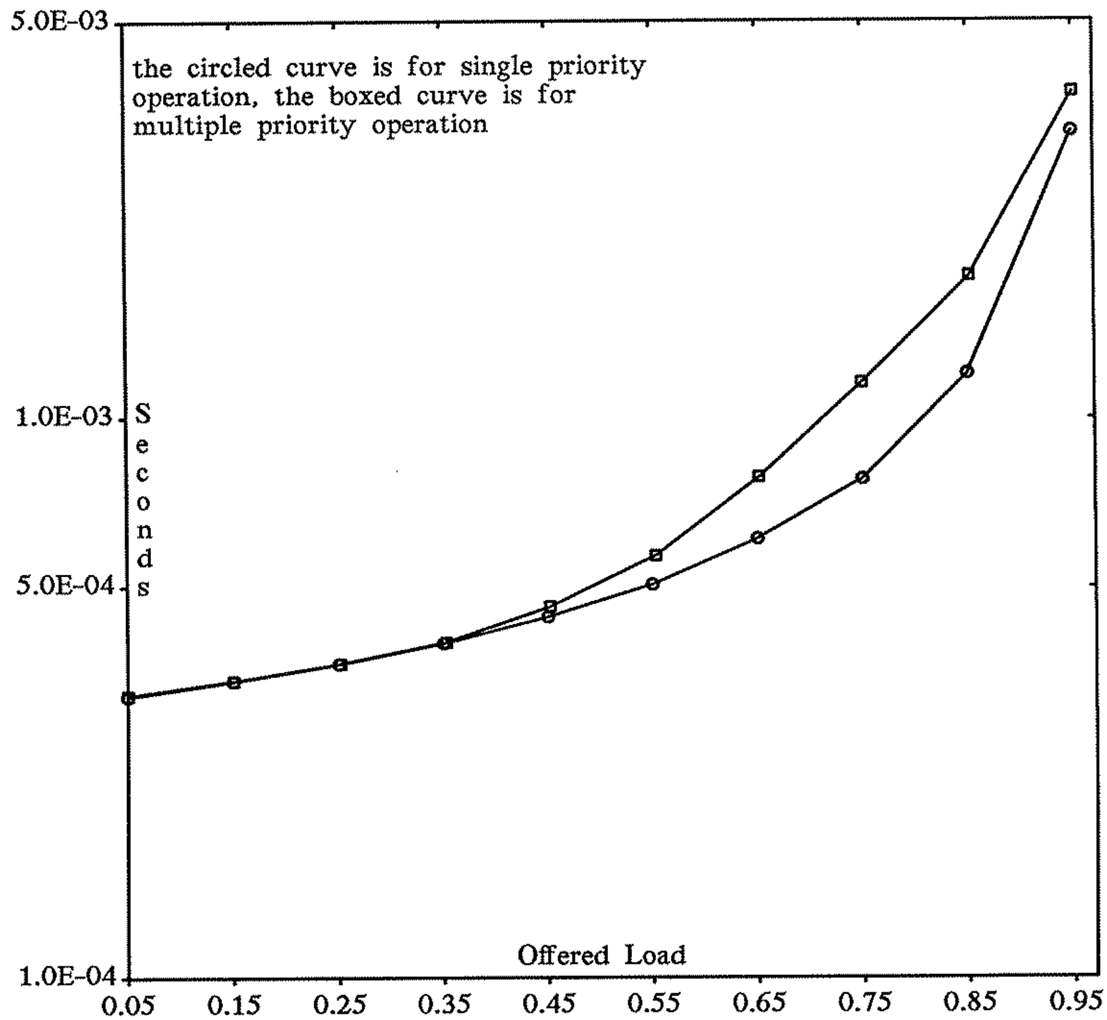


Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.23 — Station Delay

Multiple Priority Operation



Configuration:

40 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.24 — Service Delay

Multiple Priority Operation

Figure 4.25 shows the service delays for the eight priorities for 10 stations. Notice that the delay for priority 7 packets is only slightly less than that for priority 0 packets. Figure 4.26 shows an increasing effect on the relative service given to the different priorities, and Figure 4.27 shows a difference of about a factor of two in service delay between the highest and lowest priorities.

In order to obtain a more pronounced difference in the relative service received by each priority, simulations were run for loads ranging from 0.05 to 2.00 of ring capacity. The results can be seen in Figure 4.28. It is obvious that in order to gain any real benefit from the use of multiple priorities, offered load must be greater than ring capacity.

4.2.3. Variations in Packet Size, Arrival and Service

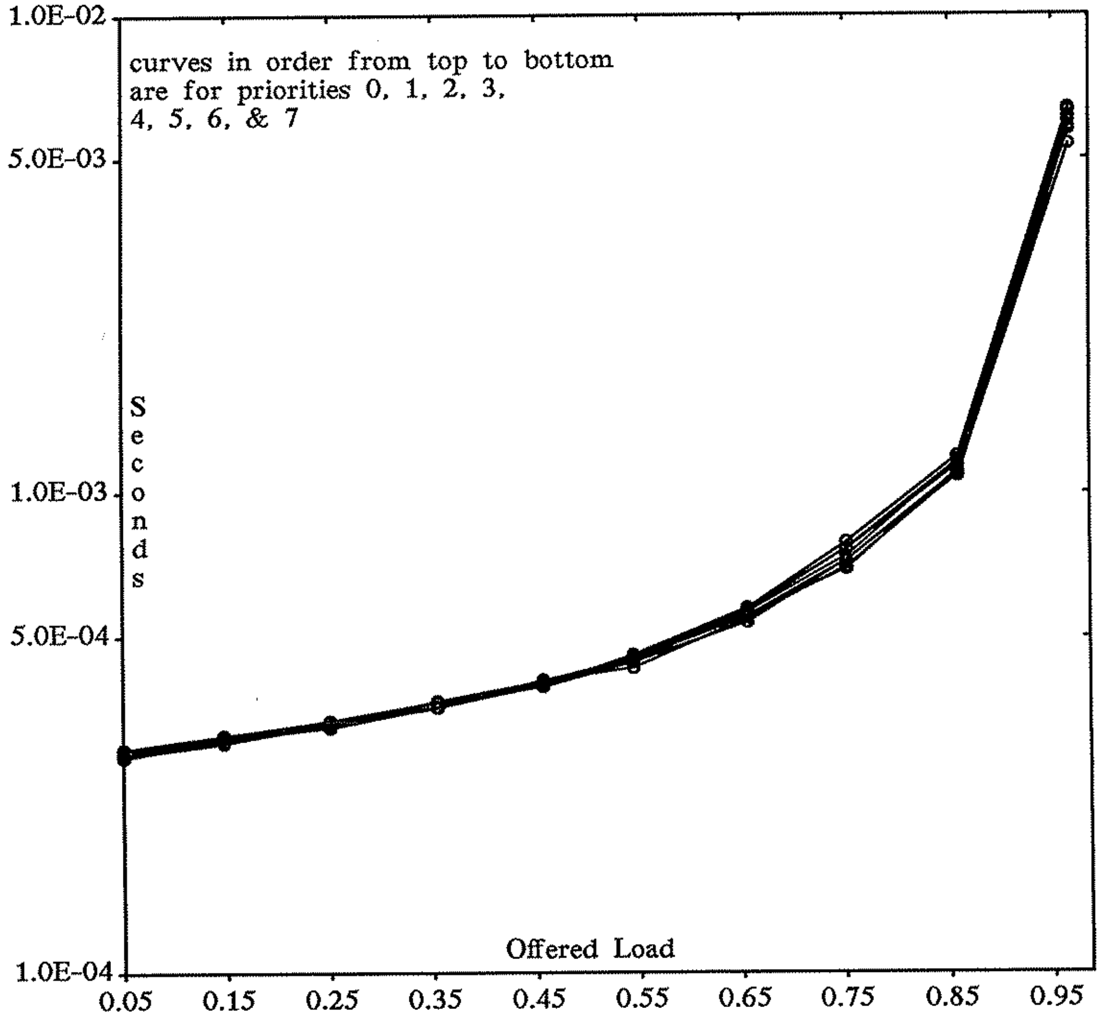
No noticeable difference in performance was seen as the result of varying packet size (other than the small increase in delay mentioned above). This result was not unexpected, and is due to the small percentage of the load given to each station. Varying the service and arrival distributions also had no noticeable effect on performance.

4.2.4. Two Priority Operation

Simulations were run for the case in which only two priorities were used. The amount of load for each priority was varied from 0.1 and 0.9 of total offered load given to the low and high priorities respectively, to 0.9 and 0.1 of total load given to the low and high priorities. The results were very close to those for the multiple priority case, when the highest and lowest priorities are considered.

4.2.5. Discussion

The performance of the token ring is generally unaffected by the use of the different priorities (at least in the case of the reference configuration). When all

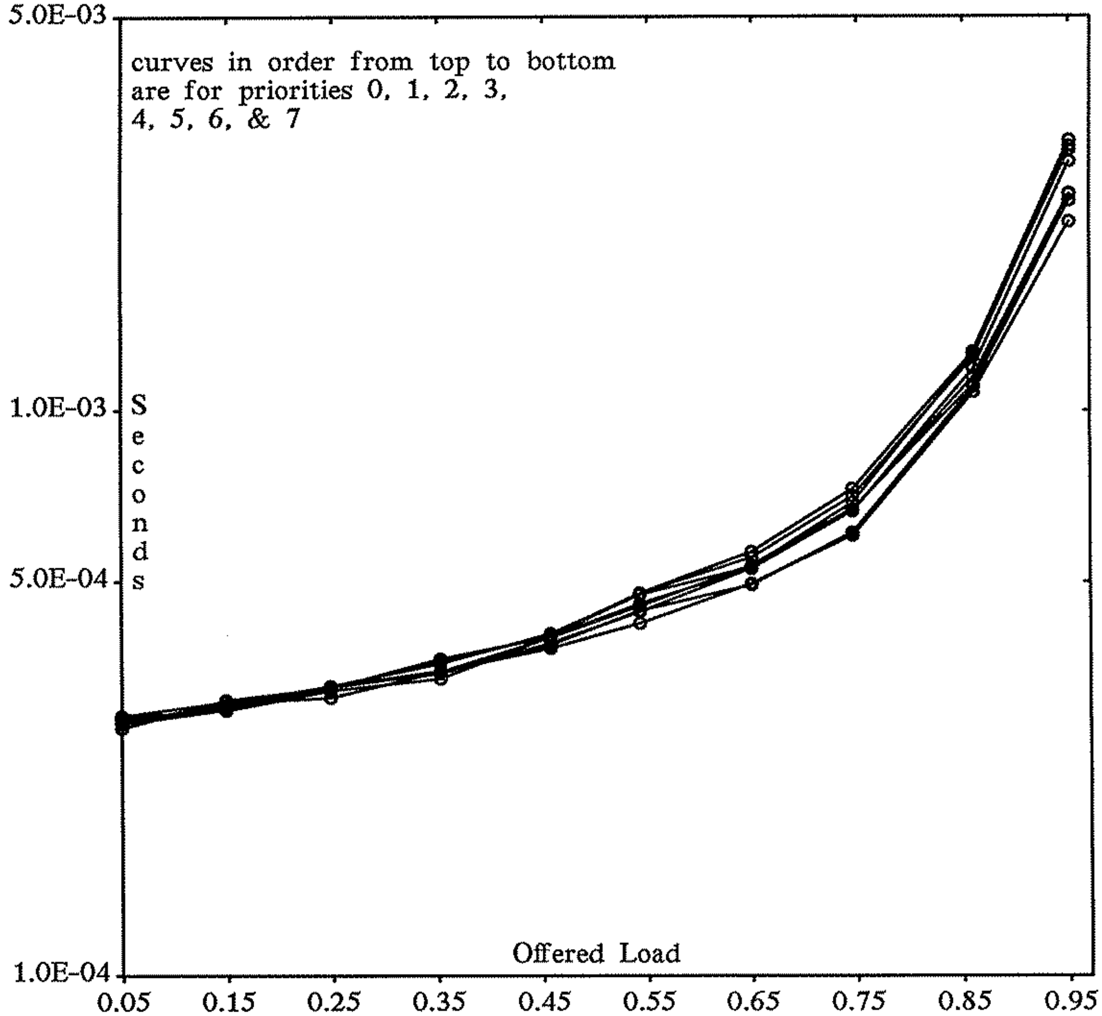


Configuration:

10 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.25 — Service Delay

Priority Delay for 10 Stations

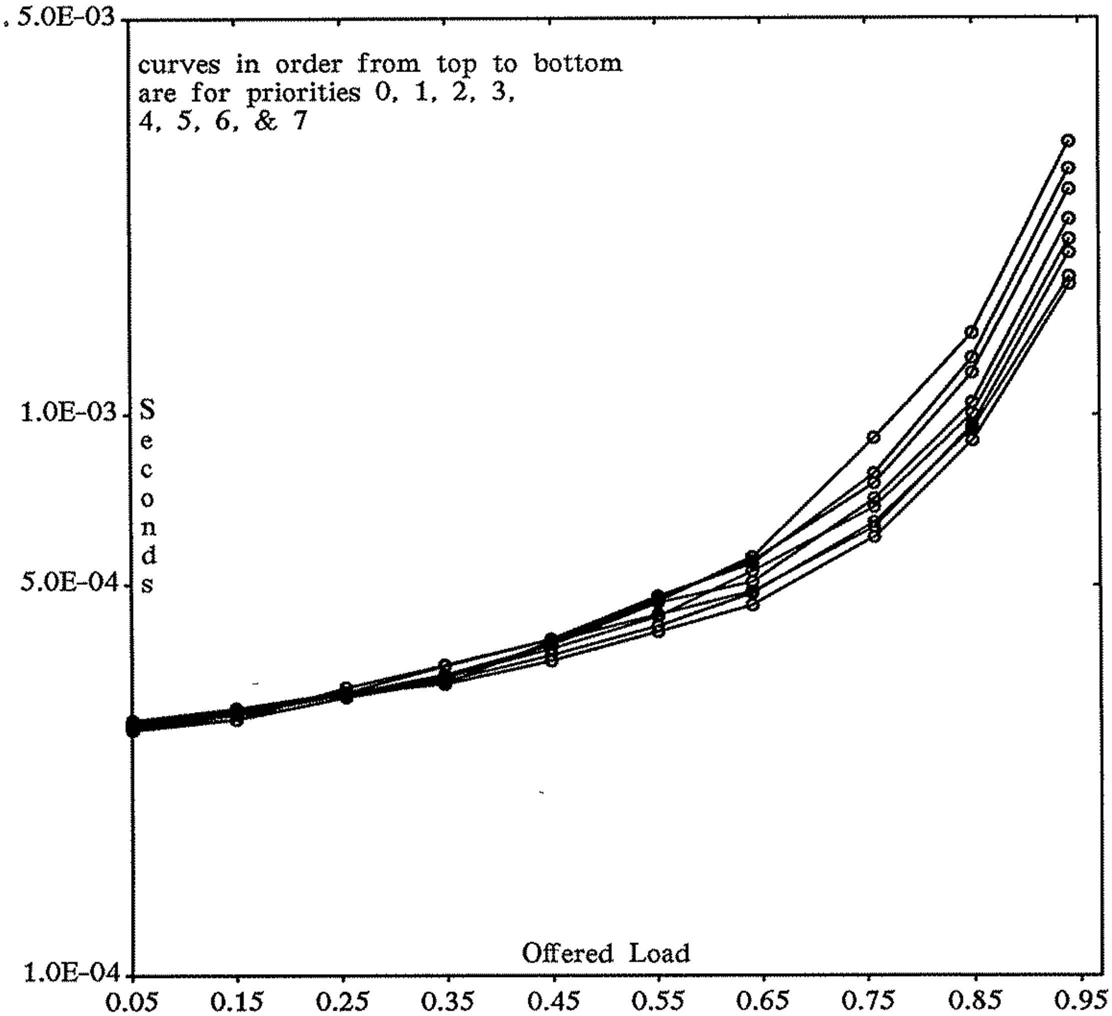


Configuration:

5 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.26 — Service Delay

Priority Delay for 5 Stations

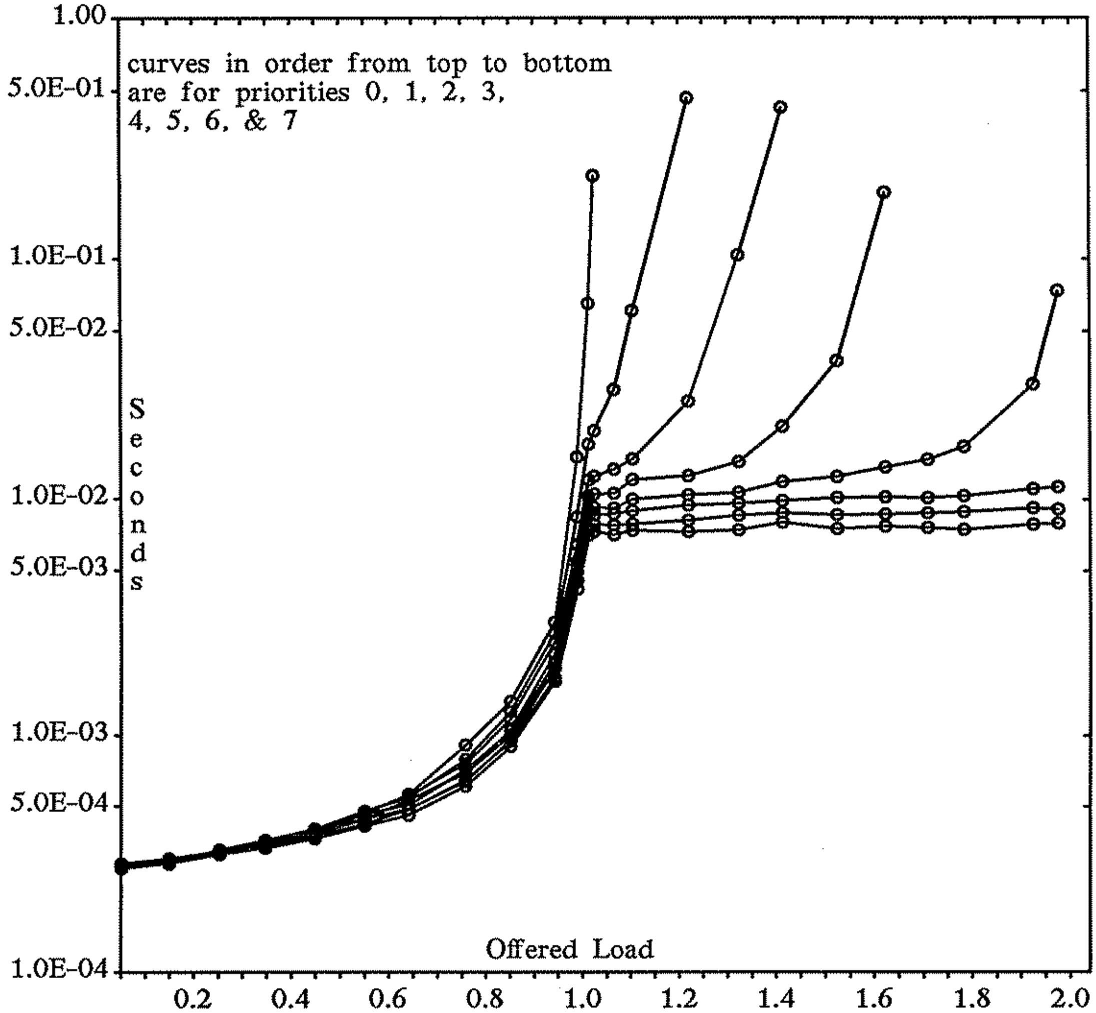


Configuration:

3 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.27 — Service Delay

Priority Delay for 3 Stations



Configuration:

3 Stations
256 Bit Packets including framing
1 Bit Time Station Latency
Exponential Arrivals
1Mbps Medium

Figure 4.28 — Service Delay

Priority Class Service Cutoff

ring stations are contributing equally to the load, packet arrivals are such that there are several token cycles before any one station has a packet (of any priority) to be delivered. Only in the case when packet arrivals are quite close together is there an effect on packet delivery delay. Note that maximum throughput for the reference configuration (using Equation 8 in Chapter 3) is 0.9917. Simulation results not shown also show maximum throughput above 0.99.

Chapter 5

Conclusions

The goal of the research summarized in this thesis was to gain a basic understanding of the operation of the 802.5 token ring. To that end many different configurations of the network were examined, but the extremely large number of possible configurations prevents exhaustive testing.

5.1. Effects on Performance

5.1.1. Protocol Overhead

The overhead of the token ring from token traffic is minimal. This is due to the small token size (24 bits), and also to the fact that a station may access a token while its upstream neighbor is still repeating the token's latter bits. This pipelining effect adds greatly to the efficiency of the network.

The amount of packet framing relative to data size has a direct impact on the throughput of the ring. As was seen in the previous chapter, throughput increases as packet size increases.

5.1.2. Number of Ring Stations

The most significant effect of protocol overhead was seen when the number of stations was increased to the point that the Return_to_Repeat_Timer (TRR) caused the station to wait before token transmission after completing packet transmission. This "invocation" of the TRR causes a rather sharp increase in token cycle time, delay, and queue lengths, although there is very little effect on throughput. This suggests that for a very large number of stations, performance may be improved by having a "token ring of token rings", thereby eliminating the impact of the TRR.

5.1.3. Packet Length

The effects of varying the packet length were seen to be quite linear. As packet length increases, so does delay. Queue length, when measured as the number of packets enqueued, varies in inverse proportion to increases in packet lengths. As was stated above, throughput also undergoes a slight increase as packet length increases.

5.1.4. Arrival and Service Distributions

Simulations were run for the reference configuration with modifications to arrival and service distributions. Three distributions were tested (uniform random, exponential, and constant) and the random distribution was given a range of twice the mean. As reported in the previous chapter, varying these distributions has no observable effect on performance.

5.1.5. Multiple Priorities

The use of multiple priority packets on the ring had only a very minor effect on performance, in that the mean delay experienced by a packet rose by less than a factor of two. This is due to the added overhead involved in handling multiple priority packets. However, higher priority packets received no better service than lower priority packets until very high network loads were reached, and shutoff of service to low priority packets only occurred at ring loads of over 100%.

Since the existence of multiple priorities appears to be a passive feature at normal loading (the only effect being to add slightly to the delay), it is suggested that only single priority messages be used. However, if it is expected that fairly long periods of extremely high transient loads will be experienced, the use of multiple priorities will assure that higher priority packets receive virtually uninterrupted service. We believe that periods of high transient load on the network would need to occur fairly frequently in order to justify the added overhead due to multiple priority traffic.

5.2. Suggestions for Future Research

One obvious subject for future research is the study of non-homogeneous ring loads, where the percent of load given to each station is varied. Also suggested is non-homogeneous priority loadings, such that both load and packet size are varied between priorities (it has already been shown in Chapter 4 that, for the case of two priorities, varying the percentage of load between them has almost no effect on performance, but no study was made of packet size variations between priorities).

We feel that research into transient loading would be beneficial, especially for the case where multiple priorities are implemented. This could be done for cases where the entire network is given an unusually high temporary load, and also when a single station is given a high transient load. Another possibility is to study the effects of varying relative station priorities by only offering certain priority packets at particular stations.

Another subject for research is variations in transmission speed and station latency. It is felt, however, that varying the station latency will not have a significant effect, and increasing the transmission rate can only have the effect of decreasing delay. Variations in timer settings should also be studied.

5.3. Summary

Even though much remains to be done in a thorough investigation of the token ring, we believe that this thesis gives a good indication of the operation of the 802.5 token ring protocol. One important discovery made is that the operation of multiple priorities has no significant effect on performance.

It was found that the mathematical model presented in Chapter 3 is a very good predictor of ring performance, and can be used to give initial indications of performance for new configurations. However, as the model predicts only mean behavior, simulations should also be run.

The protocol studied in this thesis gives every indication of being quite deterministic, and therefore quite predictable. Performance of the token ring has been shown to be quite good, which augurs well for actual implementations in industry. The simplicity of the protocol is also an advantageous feature.

Bibliography

[CARL80] D. Carlson, Bit-Oriented Data Link Control Procedures, *IEEE Transactions on Communications 28*, (Apr. 1980), .

[CLAR78] D. D. Clark, K. T. Pogran and D. P. Reed, An Introduction to Local Area Networks, *Proceedings of the IEEE 66*, 11 (Nov. 1978), 1497-1517.

[HEYM83] D. P. Heyman, Data-transport Performance Analysis of Fasnet, *Bell System Technical Journal 62*, 8 (Oct. 1983), 2547-2560.

[IEEE84] IEEE, *Logical Link Control*, The Institute of Electrical and Electronics Engineers, Inc., 1984.

[IEEE85a] IEEE, *Token-Passing Bus Access Method and Physical Layer Specifications*, The Institute of Electrical and Electronics Engineers, Inc., 1985.

[IEEE85b] IEEE, *Token Ring Access Method and Physical Layer Specifications*, The Institute of Electrical and Electronics Engineers, Inc., 1985.

[IEEE85c] IEEE, *Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications*, The Institute of Electrical and Electronics Engineers, Inc., 1985.

[KONH74] A. G. Konheim and B. Meister, Waiting Lines and Times in a System with Polling, *J. ACM 21*, 3 (July 1974), 470-490.

[KUEH79] P. J. Kuehn, Multiqueue Systems with Nonexhaustive Cyclic Service, *Bell System Technical Journal 58*, 3 (Mar. 1979), 671-698.

[LEVY75] Y. Levy and U. Yechiali, Utilization of Idle Time in an M/G/1 Queuing System, *Manage. Sci. 22*, 2 (Oct. 1975), 202-211.