Spring 5-31-1994

Performance analysis of the "Fiber distributed data interface (FDDI) network" using petri nets and SPNP software package

Savvas Eteoclis Christodoulou
New Jersey Institute of Technology

Copyright Warning & Restrictions

The copyright law of the United States (Title 17, United States Code) governs the making of photocopies or other reproductions of copyrighted material.

Under certain conditions specified in the law, libraries and archives are authorized to furnish a photocopy or other reproduction. One of these specified conditions is that the photocopy or reproduction is not to be "used for any purpose other than private study, scholarship, or research." If a, user makes a request for, or later uses, a photocopy or reproduction for purposes in excess of "fair use" that user may be liable for copyright infringement,

This institution reserves the right to refuse to accept a copying order if, in its judgment, fulfillment of the order would involve violation of copyright law.

Please Note: The author retains the copyright while the New Jersey Institute of Technology reserves the right to distribute this thesis or dissertation

Printing note: If you do not wish to print this page, then select "Pages from: first page # to: last page #" on the print dialog screen

ABSTRACT

PERFORMANCE ANALYSIS OF THE FIBER DISTRIBUTED DATA INTERFACE (FDDI) NETWORK USING PETRI NETS AND SPNP SOFTWARE PACKAGE

by

Savvas Eteoclis Christodoulou

The main purpose of this thesis is to model a "Fiber Distributed Data Interface" (FDDI) Network using Petri Nets, and to analyze its performance with the help of the SPNP software package. The verification of a communication protocol, by modeling it as a discrete-event system using Petri Nets is a new approach.

The correlation between the throughput rate, voice and data throughput, and the parameters of the system, such as the network load and the network speed are investigated. An overview of the "Fiber Distributed Data Interface" is provided, along with its network protocol and the limitations of its operating parameters. A proposed Petri Net approach is then introduced. Finally, the effect of the network latency and load on the network's overall performance is derived. A method for minimum delay is also proposed, and demonstrated with examples and computation results.

PERFORMANCE ANALYSIS OF THE "FIBER DISTRIBUTED DATA INTERFACE (FDDI) NETWORK" USING PETRI NETS AND SPNP SOFTWARE PACKAGE

by
Savvas Eteoclis Christodoulou

A Thesis
Submitted to the Faculty of
New Jersey Institute of Technology
in Partial Fulfillment of the Requirements for the Degree
of Master of Science in Electrical Engineering

Department of Electrical and Computer Engineering

May 1994

APPROVAL PAGE

PERFORMANCE ANALYSIS OF THE "FIBER DISTRIBUTED DATA INTERFACE (FDDI) NETWORK" USING PETRI NETS AND SPNP SOFTWARE PACKAGE"

Savvas Eteoclis Christodoulou

Dr. MengChu Zhou, Thesis Advisor — Date
Assistant Professor of Electrical and Computer Engineering, NJIT

Dr. Anthony Robbi, Committee Member — Date
Associate Professor of Electrical and Computer Engineering, NJIT

Dr. Daniel Chao, Committee Member — Date
Associate Professor of Computer and Information Science, NJIT

BIOGRAPHICAL SKETCH

Author: Savvas E. Christodoulou

Degree: Master of Science in Electrical Engineering

Date: May 1994

Undergraduate and Graduate Education:

- Master of Science in Electrical Engineering,
  New Jersey Institute of Technology, Newark, NJ, 1994

- Bachelor of Engineering in Electrical Engineering,
  City College of City University of New York, New York, NY, 1992

Major: Electrical Engineering

This thesis is dedicated to
my family and especially to
the memory of my mother.

ACKNOWLEDGMENT

The author wishes to express his sincere gratitude to his advisor, Professor MengChu Zhou, for his guidance, friendship, and moral support throughout this research.

Special thanks to Professors Anthony Robbi and Daniel Chao for serving as members of the committee.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION TO FDDI NETWORK

1.1 Characteristics

FDDI is a high performance fiber optic token ring Local Area Network (LAN), running at 100 Mbps over distances up to 200km with up to 1000 stations connected [Burr,1986; Ross, 1986,1987].

It is designed to provide both synchronous (voice) and asynchronous (data) service. It also provides higher priority value to synchronous service. With synchronous services a user receives a preallocated maximum bandwidth (i.e., time to transmit specified frames) and a guarantee of a maximum delay per frame. Packet voice is one example of a service with these requirements. A timed token rotation (TTR) protocol, described in Chapter 2, is used to enforce these guarantees.

Using a maximum latency 0.6 μsec per physical connection, an FDDI system produces a 600 μsec maximum total station latency around the ring. Maximum propagation delay, end-to-end, using the figure of 5 μsec/km as the delay per km, is 1 msec. The maximum ring latency, therefore, turns out to be L_{max}= 1.6 msec. The maximum frame length is specified as 9000 symbols. At the 100 Mbps symbol rate, the maximum frame transmission time is 0.36 msec.

1.2 Network Physical Description

In an FDDI system, N stations transmitting both synchronous (Voice) and asynchronous (Data) traffic are connected as a Local Area Network (LAN). Each station in the system can have more than one Voice or Data source. Stations can also have either Voice or Data sources only. A general FDDI network connecting N stations with a total of Vs voice sources and Ds data sources is shown in Fig. 1-1. Each station in the network shown, is connected to only two stations, forming a ring network. The formed network is unidirectional. Each station can receive the token ring from the station on its left, and pass the network only to the station to its right. For example station 2 is connected to station 1 and to station 3. It can receive the token ring only from station 1 and can pass the token only to station 3. This sequence should be followed throughout the ring, regardless that, some stations in the network do not have any messages to transmit. Voice and data messages are arriving to the stations exponentially with a rate λ forming different queues of voice and data respectively. Each station has one queue of data and one queue of voice. Therefore, in an N station network, N queues of voice messages and N queues of data messages are formed. Each queue has one server, and the service provided is based on first in, first out algorithm.

The communications rules of the network are controlled by the Timed Token Rotation (TTR) protocol described later in this chapter.

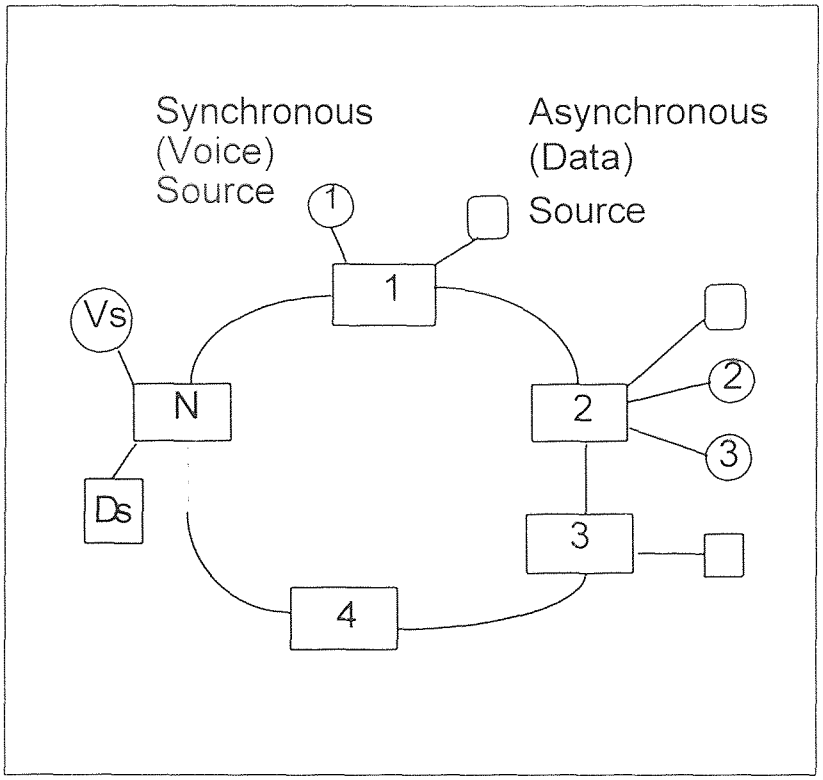

Figure 1-1. General FDDI network consisting of N stations.
Vs voice sources and Ds data sources are available in the network

1.3 Timing Requirements

On ring initialization, all stations on the ring negotiate a key parameter, the **Target Token Rotation Time (TTRT)** [ANSI 1987]. TTRT is actually the average token rotation time on the ring. Each station requests a value **T_REQ** for the TTRT. The minimum value of T_REQ is chosen as the operational value, **T_OPR**, of TTRT.

The maximum token rotation time around the ring is proved to be, however, 2T_Opr [Johnson 1987]. Because of that each station should request a T_REQ value which is one-half of its absolute maximum token rotation time.

The minimum value of T_Opr is acquired by all stations, and each station receives a fractional allotment ST of this time to be used in transmitting synchronous traffic. If, for

example, there are Vs synchronous sources on the ring, the total synchronous allotment is then, Vs*ST sec. T_Opr should then be chosen large enough to accommodate this synchronous allotment and to allow at least one data (asynchronous) frame per token rotation, plus the time to transmit the token.

It is clear then, that T_Opr must satisfy the following inequality:

$$T\_Opr > (Vs * ST) + (Token\_Time) + (L) + (F\_Max) \quad (1.1)$$

where:

L is the ring latency

F_Max is the maximum frame transmission time and equals to 0.36 msec

Token_Time is the token transmission time and equals to 0.88 μsec

The operational value T_Opr of the target token rotation time TTRT is bounded by a default minimum value of 4 msec and a default maximum value of 165 msec. The later value is chosen to guarantee stable ring recovery [ANSI 1987].

1.4 Timed Token Rotation (TTR) Protocol

An FDDI station receiving the token may capture it and transmit any waiting frames up to a specified time limit to be discussed below.

Each station has two timers involved in this protocol. A third timer, not discussed in this thesis, is used to provide recovery from transient ring error conditions [ANSI 1987]. A timer called the **Token Rotation Timer (TRT)** controls access to the ring. This timer is initialized to T_Opr.

If TRT>0 when the token arrives to the station, the token is said to be on time. This thesis, however, considers this case only, since we will not study the performance analysis of the system for abnormal cases such as errors, corrections, etc.. If this is the case, on the arrival of the token, the value of the TRT is transferred to the second timer, the **Token Holding Timer (THT)**, and TRT is reset to T_Opr.

Any waiting synchronous (voice) traffic is then transmitted up to the maximum allotment ST. On expiration of ST, or completion of synchronous transmission, whichever comes first, THT is enabled and any waiting asynchronous (data) frames are transmitted, until THT expires, or there are no further frames to transmit, whichever comes first.

Asynchronous frames may have up to eight levels of priority, if desired, but in this thesis we are focusing on one priority level only. If THT expires while an asynchronous frame is being transmitted, completion of that frame is allowed.

The following flow diagram, Fig. 1-2, shows the FDDI Timer Token Rotation Protocol as it was presented in [Sanker 1989]. The flow diagram, however, does not take into consideration the possibility TRT<0 when the station receives the token.

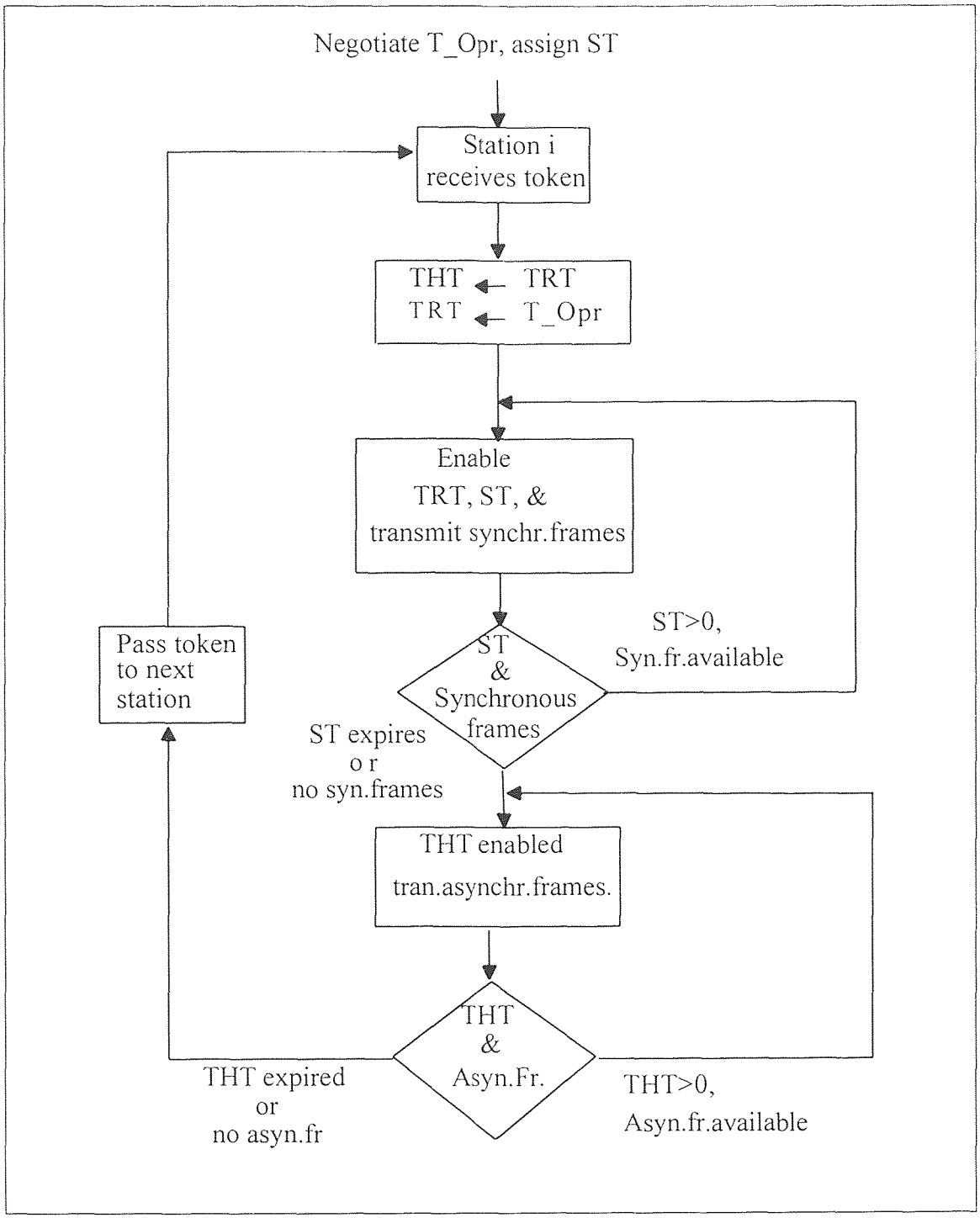

Figure 1-2. FDDI Timer Token Rotation Protocol.

1.5 Review on Modeling and Analysis of a Token Ring

1.5.1 Previous Studies on Token Ring System

Token rings constitute one of the most widely known candidates for a local area network. In this case the nodes of the network are linked in a circular fashion. The connected stations gain access to the transmission channel by means of a "right to transmit" which is represented by a special configuration of bits, namely the token.

In the literature we can find a lot of studies concerning cyclic server systems. Most of these are not mathematically tractable. Also, some of them are using approximation methods based on simplifying hypotheses. Some studies do not consider the walking time between the stations [Nair 85], and some other assume symmetric systems where an equal load is offered to each station [Cooper 69], [Kaye 72].

The discipline of the service at one queue was also a simplification method used for modeling a token ring. Many results exist for the case of an exhaustive service discipline either for a symmetrical, [Kaye 72], or an asymmetrical network, [Ferguson 85]. Limited results were also obtained using the gated service case, [Ferguson 85]. There is no paper at the moment, to our knowledge, analyzing the general case of a token ring system, with queues having non-exhaustive service.

1.5.2 Queueing Theory Approach for Analyzing Token Ring

The analysis of the token ring using queueing theory is mainly based on the assumptions of the exponential distribution, and this utilizes the properties of the Markov chains for solution.

Each terminal is assumed to spend an exponentially distributed amount of time with mean $1/\alpha$ preparing service request. If the state of the system is $N(t) = k$, then the number of idle terminals is K-k. Therefore the rate at which service requests are generated is $(K-k)\alpha$. We also assume that the time required to service each request is an exponentially distributed amount of time with mean $1/\mu$. N(t) is then the continuous-time Markov chain with the transition rate as shown in Fig. 1-3.

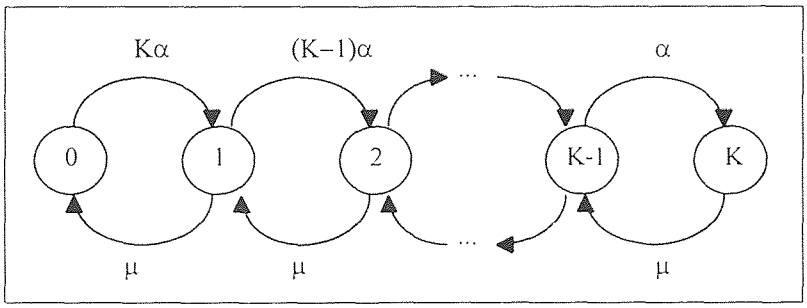

Figure 1-3. Transition rate diagram for a finite source single-server computer system.

The throughput of the computer system shown above, is defined as the rate at which it completes transactions.

It is proved [Garcia 89] that the mean delay in the system, E[T], for each request is,

$$E[T] = (K/\lambda) - (1/\alpha)$$

whereas the throughput, λ, is,

$$\lambda = K / [\, (1/\alpha) + E[T] \,]$$

From the above equations, it can be seen that λ grows linearly with K. But as K increases, the computer will eventually become fully utilized, and then outputs transactions at its maximum rate, namely μ transactions per second. Thus $\lambda \cong \mu$, for large

K. Figures 1-4 and 1-5 show respectively, the delay and throughput for finite source system as a function of number of sources as they were presented in [Garcia 89].

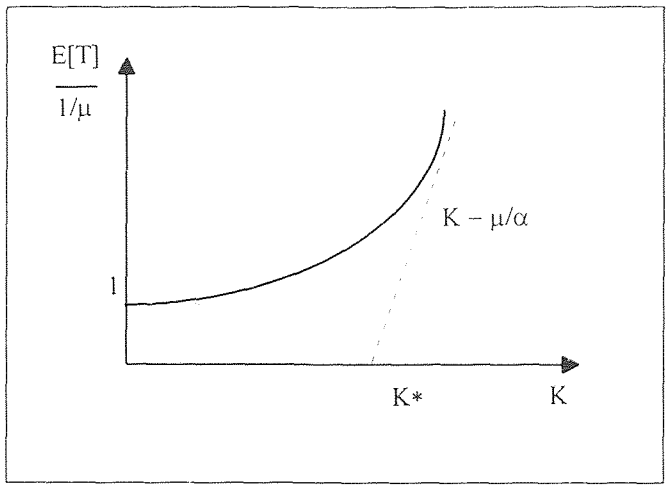

Figure 1-4. Delay for finite source system as a function of number of sources.

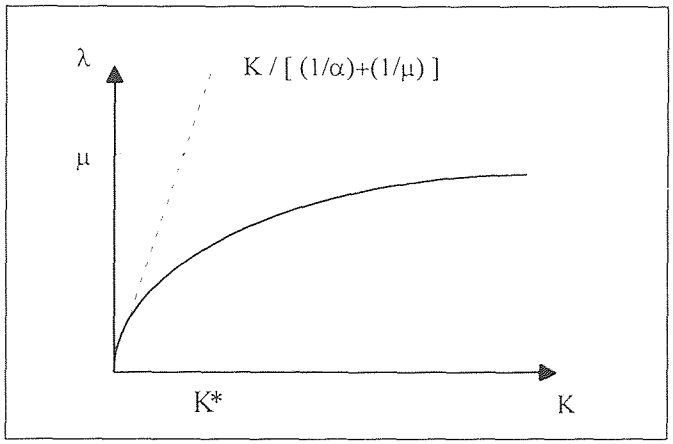

Figure 1-5. Throughput for finite source system as a function of number of sources.

The dashed lines in the Figures shown above, indicate asymptotes for small and large values of K. The value of K where the two asymptotes for E[T] intersect is called the system saturation point, K*. When K becomes larger then K*, the requests from the

terminal are certain to interfere with each other and the response time increases accordingly.

1.5.3 Simulation Approach for Analyzing Token Ring

The simulation analysis, is the most common analysis for examining the behavior of a network, since it provides comparisons with theoretical approaches. Because of this reason, many simulation software packages were developed in the last decade to simulate the performance of the token ring networks.

One of the most complete simulation analysis presented so far, was the one given by [Ghani 91]. The later paper analyzes theoretically, most of the parameters involved in the network, and also simulates the network presenting the comparisons between the two approaches. Figures 1-6, 1-7 and 1-8 show the Delay vs the Network Utilization for varying number of stations, T_Opr and network length respectively.

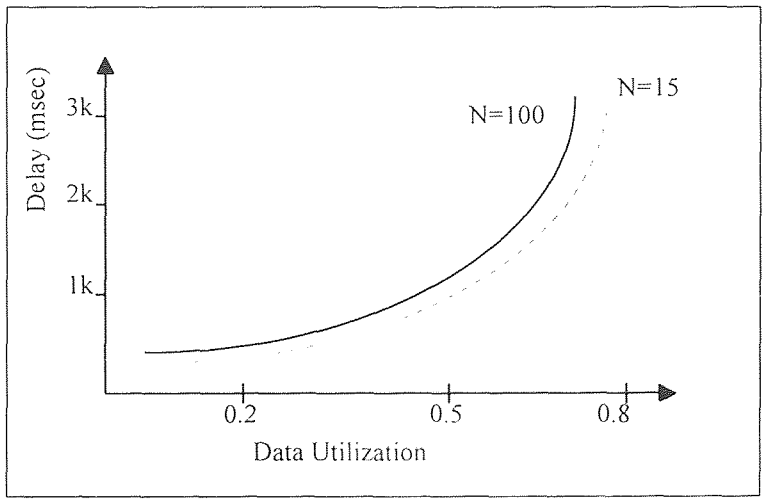

Figure 1-6. The effect on delay as a function of the number of stations in the network [Ghani 91].

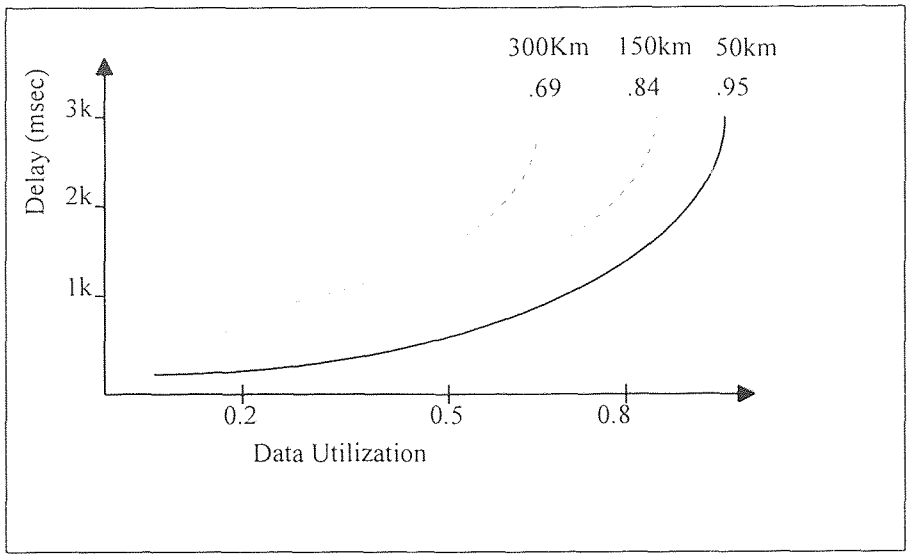

Figure 1-7. The effect on delay as a function of network's Geographical size [Ghani 91].

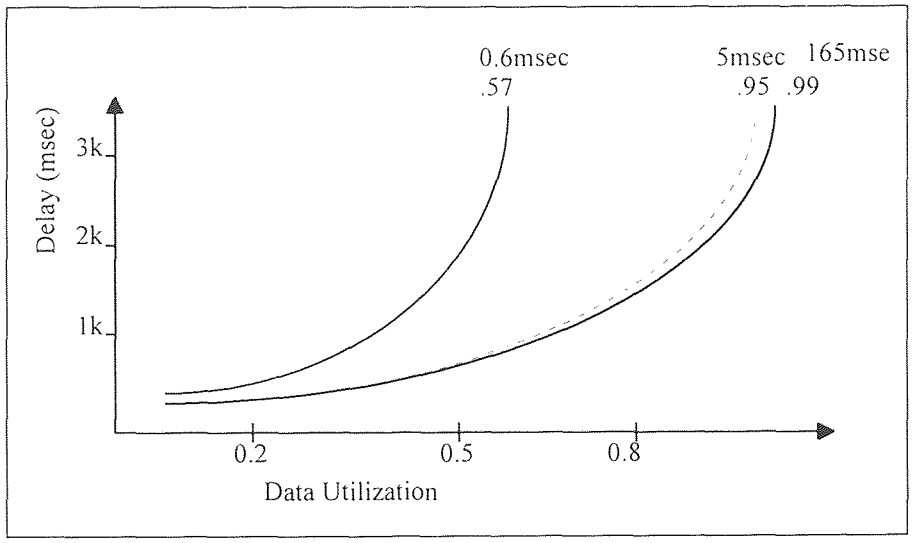

Figure 1-8. The effect on the delay as a function of the value of T_Opr [Ghani 91].

1.5.4 Petri Net Approach for Analyzing Token Ring

The power and flexibility of Petri nets, have been a popular tool for token bus Local Area Networks modeling and performance [Zhou 92]. That paper examined the single, exhaustive, gated and limited service types in both symmetric and asymmetric token bus LAN. The correlation between the throughput and delay for the system with respect to the offered load, was provided among the GSPN model. The following figure illustrate the advantage of asymmetric over symmetric system.

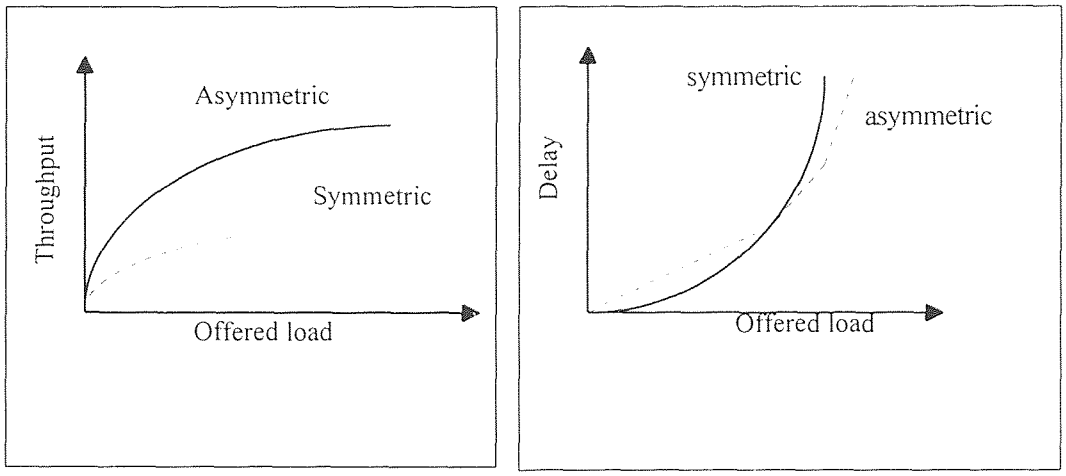

Figure 1-9. Performance of symmetric and asymmetric LAN [Zhou 92].

1.6 Objectives

The goal of this thesis is to model and analyze the "Fiber Distributed Data Interface" Network using Petri Net theory. The objectives are:

1. To propose a model describing the protocol used in FDDI systems, using Stochastic Petri Nets.
2. To propose techniques to maximize the output throughput of the network.

3. To analyze the performance of the network using SPNP Software package.
4. To compare the results of the present approach, with the results obtained from previous approaches.
5. To point out the advantages of the Petri net model over the existing Timed Token Rotation protocol.

CHAPTER 2

PETRI NET MODEL

2.1 Introduction to Petri Nets

Petri nets are a promising tool for describing and studying information processing systems that are characterized as being concurrent, asynchronous, distributed, parallel, non-deterministic, and/or stochastic.

Petri Nets are a graphical and mathematical modeling tool applicable to many systems. As a graphical tool Petri nets can be used as a visual-communication aid similar to flow charts, block diagrams, and networks. As a mathematical tool, it is possible to set up state equations, algebraic equations, and other mathematical model governing the behavior of systems [Murata 89].

Petri nets consist of places (circles) which represent conditions, and transitions (bars) which represent events. Most of the approaches for modeling *Discrete Event Dynamic Systems* (DEDS) with Petri nets, the following interpretations for places, transitions, and tokens are employed [Zhou 93]:

1. A place represents a resource status or an operation; when it represents the later, one or more tokens in the place indicate that the resource is available and no token indicates that it is not available. If a place represent an operation, a token in it shows that an operation is being executed and no tokens shows that it is not.

2. A transition represents either start or completion of an event or operation process.

Places and transitions are connected by directed arcs from places to transitions or from transitions to places. If an arc is directed from node *i* to node *j* (either from a transition to a place or from a place to a transition), then *i* is an input to *j* and *j* is an output to *i*. An arc is directed from place p_i to transition t_j if the place is an input to the transition. Similarly, an arc is directed from a transition t_j to a place p_i if the place is an output of the transition. A PN is a directed graph.

2.2 Advantages of Petri Net Modeling

Petri nets as a graphical tool provide a unified method for design of discrete event systems from hierarchical system description to physical realizations. Compared with other model approaches, they have the following advantages [DiCesare 91; Ma 92; Martinez 86; Zhou 89a]:

1. Ease of modeling DES characteristics: concurrency, asynchronous and synchronous features, conflicts, mutual exclusion, precedence relations and system deadlocks.
2. Ability to generate supervisory control code directly from graphical PN representation.
3. Ability to check the system for undesirable properties such as deadlock and instability.

4. Performance analysis without simulation is possible for many systems. Production rates, resource utilization, reliability and performability can be evaluated.

2.3 Transition Enabling and Firing Rules

A Petri net Z is a 5-tuple, $Z = (P, T, I, O, m_o)$ where:

$P = \{p_1, p_2, p_3,...p_n\}$ is a finite set of places;

$T = \{t_1, t_2, t_3,...t_s\}$ is a finite set of transitions;

$I : P \times T \rightarrow N = \{0, 1,2 ...n\}$ is the input function that specifies the arcs directed from places to transitions;

$O : P \times T \rightarrow F = \{0, 1,2 ...n\}$ is the output function that specifies the arcs directed from transitions to places;

$m_o : P \rightarrow \{0, 1, 2, 3,...\}$ is the initial marking

$P \cap T = \varnothing$ and $P \cup T \neq \varnothing$

A Petri net is a particular kind of directed graph. It is presented with an initial state called the *initial marking* m_o. The graph basically consists of two kinds of nodes, namely *places* and *transitions*. In graphical representation, places are drawn as circles and transitions as bars or boxes. *Arcs* are drawn either from places to transitions (input arcs), or from transitions to places (output arcs). Arcs are labeled with their weights (positive integers), where a *K*-weighted arc can be interpreted as *K* parallel arcs. Labels for the unity weight are usually omitted. A marking assigns to each place a nonnegative integer k. Graphically, k black dots (tokens) are placed in place p. Marking m_i actually represents the number of tokens in all places at state i (m_o is the initial state). In other words, marking m is an n-vector, where n is the total number of places. The ith component is

the number of tokens in place i. K tokens are put in a place to indicate that K data items or resources are available.

The behavior of many systems can be described in terms of systems states and their changes. A state or marking in a Petri Net is changed according to the following transition firing rule:

1) A transition $t \in T$ is enabled iff $m(p_j) \geq I(p_j, t)$ for $1 \leq j \leq n$

A transition t is said to be enabled if and only if, the number of tokens in the input places of transition t is equal or greater than the number of input arcs from each of the places to t.

2) An enabled transition t may fire at marking m', yielding new marking

$m(p_i) = m'(p_i) + O(p_i, t) - I(p_i, t)$ for $i = 1,2,3...,|P|$.

A firing of an enabled transition removes I(p,t) tokens from each input place p of t, and adds O(p,t) tokens to each output place p of t.

Systems with priority, however, cannot be modeled with the common arcs discussed above. A new kind of arc called *inhibitor arc* is introduced to overcome priority problems [Agerwala 74]. An inhibitor arc is represented by a dashed line terminating with a small circle instead of an arrowhead at the transition, like the common arc. The inhibitor arc disables the transition when the input place has a token, and enables the transition when the input place has no token. No token is removed through an inhibitor arc when the transition fires.

The transition enabling and firing rules are shown in Fig. 2-1. Transition t is enabled iff

$$m(p_1) \geq k_1,\ m(p_2) \geq k_2,\ m(p_3) \leq k_3,\ m(p_4) \leq k_4$$

Firing transition t will remove k_1 and k_2 tokens from p_1 and p_2 respectively, and deposit one token to its output place p_5. No tokens will be removed from place p_3 and p_4.

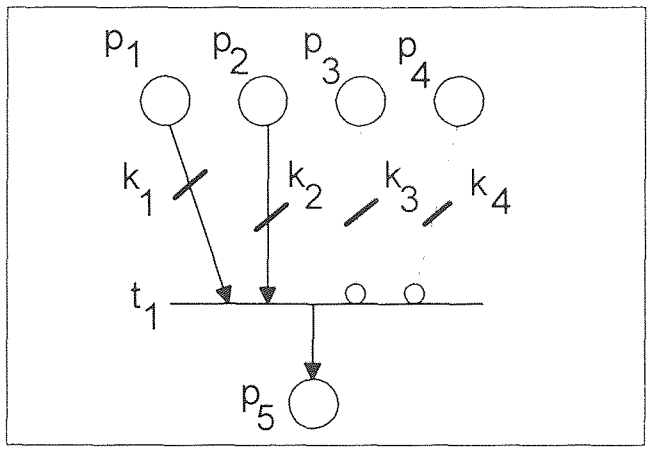

Figure 2-1. Petri net example using inhibitor arcs.

2.4 Petri Net Approach to Analysis of FDDI

The key to modeling FDDI with a Petri net was the conversion of the time-driven event system (Time Token Rotation protocol) to a discrete-driven one.

The first step was the conversion from *bits to messages (tokens).* The service rate was then converted from *bits per second (bps) to messages (tokens) per transition firing.* Finally, the timers used in the Timer Token Rotation protocol (Fig.1-2) for controlling THT, TRT, T_Opr are now replaced by a "Number of Tokens Presented Control Method". The maximum time available for transmitting voice (ST), or data (THT) used in the real-life time-driven system, are now replaced to a discrete-event system (DES) by

using the maximum number of times a transition can fire. Each firing implies transmission of a voice or data message (token).

Figure 1-2 which was suggested by R. Sanker in 1989, was used for so many years as a standard tool to study the performance analysis of FDDI network. It can now be replaced by a discrete event dynamic system (DEDS), suggested in this thesis and shown briefly in Fig. 2-2.

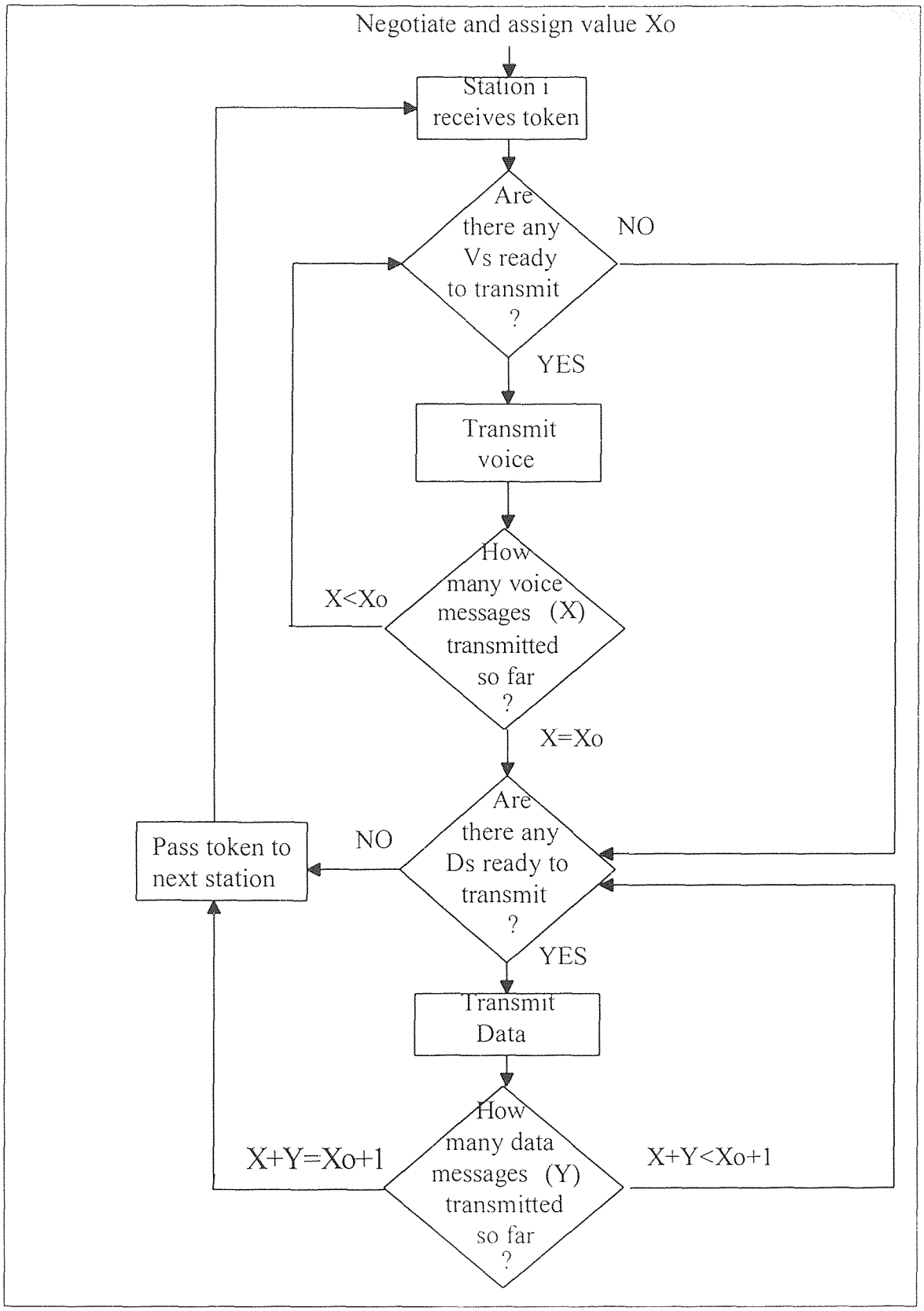

Figure 2-2. FDDI Timer Token Rotation protocol for Petri Net approach.

This diagram in Fig. 2-2 is in fact a DEDS which has the same algorithm as the one used in Fig. 1-2, and it will be the basis for this thesis' approach to studying the performance of FDDI network from the discrete event point of view.

2.5 Petri Net Model

2.5.1 Overview of the Petri Net Model

Figure 4-3 is a simple PN model with four (4) places and five (5) transitions which only support a small portion of the real life TTR protocol's features. It is very important though, to mention this model and its features, in order to follow the overall model approach, and the final Petri net model to be suggested later. This figure shows a station holding the token, able to transmit voice and data messages, with voice messages having priority. The three (3) transitions on the upper-right side present the three (3) conditions, based on which the station holding the token, should pass the token to the next station. The transition on the top represents the case when the station holding the token ring does not have any messages to transmit. The next transition represents the case when the station transmits the maximum number of voice permitted, but it does not have any data messages to transmit. Finally, the third transition shows the case when the station transmits all voice and data messages permitted. The place on the right represents the next station on the ring. The two places on the lower part of the figure represent the voice and data messages presented in the station. The remaining transitions describe the voice and data service rate. The last two transitions are exponentially distributed since they

represent rates, whereas the first three are immediate. The inhibitor arc in the network supports the priority of voice over data.

Figure 2-3 however, is unable to capture all the features described in Fig. 2-2. First, it does not support the complete description of the priority function. The protocol states that, maximum Xo number of voice sources can be transmitted prior to data transmittion. Figure 2-3 does however, support exhaustive priority of voice over data. Also, the later figure does not show the maximum number Y of data a station can transmit, as a function of the number of voice messages already have been transmitted. Finally, Fig. 2-3 does not cover the issue of voice and data message arrival rate λ and the formation of different queues.

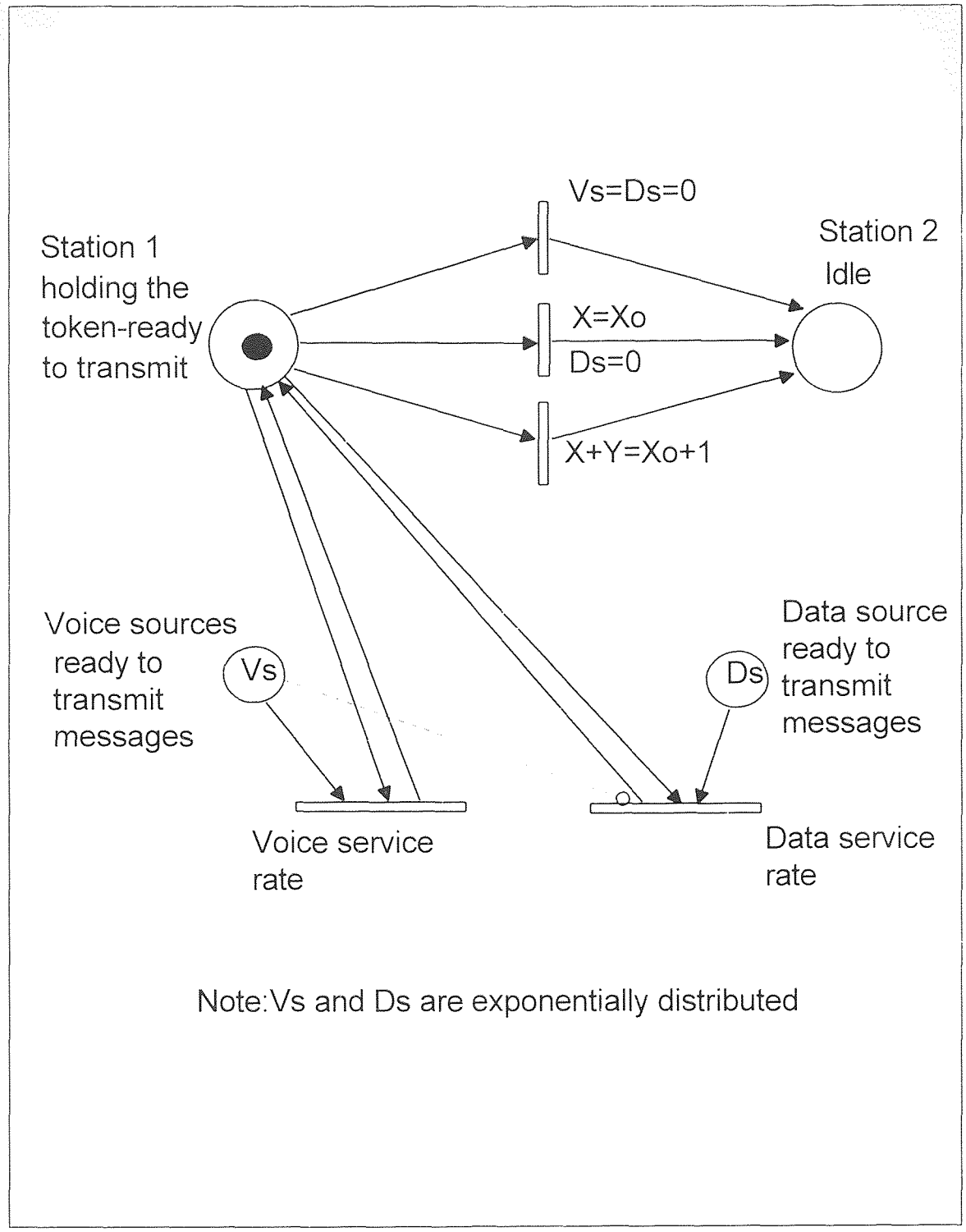

Figure 2-3. Petri net model for a single station

2.5.2 Description of the Petri Net Model Operation

The complete Petri net supporting all features of Timer Token Rotation protocol first presented in Fig. 1-2, and then in Fig. 2-2 as a DEDS, is shown in Fig. 2-4. Each Petri Net station consists of thirteen (13) places and nineteen (19) transitions. Table 2-1 tabulates all places and transitions presented in this final Petri net model.

Place p_1 is part of the token ring and indicates whether the station is idle or not. The available token in p_1 indicates that the station is able to transmit any waiting voice or data messages represented by $m(p_2)$ and $m(p_3)$ respectively. Voice and data messages are arriving with a rate λ to place p_2 and p_3 through transitions t_1-t_{16} and t_2-t_{17}, respectively. As long voice has priority over data, the station starts transmitting voice messages with a rate μ by firing transition t_3. Every time a voice message is transmitted, a token is stored in place p_6. The number of token available in place p_6 controls the muximum number of voice messages (Xo) the station can transmit per token ring holding time. As soon as the place p_6 is filled with Xo tokens, the station is prohibited from transmitting any more voice messages. The Xo-weighed inhibitor arc from place p_6 to transition t_3, basically controls the protocol's feature concerning the TRT time.

Data messages are allowed to be transmitted under two conditions.

- The first condition is when no more voice messages are available in the station. This condition is represented by the inhibitor arc from place p_2 to transition t_4. In that case, transition t_4 is enabled, transmitting any available data message in place p3, with a rate μ.

- The second condition, is when Xo voice messages have already been transmitted. In that case, transition t_3 is disabled as mentioned before, while at the same time, transition t_5 is enabled.

In both cases, every time a data message is transmitted, either through transition t_4 or transition t_5, a token is stored in place p_7. Place p_7 is contolling the feature of the THT protocol. The myval-weighed inhibitor arcs from p_7 to transition t_4 and t_5, indicate that the maximum number of data messages that can be transmitted depends on the remaining time available after transmitting X voice messages. It should be noted here that as long as a data message is transmitted no voice message is able to be transmitted. To reinforce this feature, transition t_3 which is responsible for voice service is disabled as soon as a token is stored in place p_7. To avoid deadlock, transition t_1, which represents voice message arrival rate, is also disabled with the appearance of a token in place p_7.

Transitions t_8-t_9 and t_{10} indicate the three conditions, the station should transmit the ring token to the next station, represented by place p_8, as discussed briefly in the previous section. It should be noted here, that as soon as the station passes the token to the following one, the control counters p_6 and p_7 are reset. This is achieved by the immediate transitions t_6 and t_7.

The remaining transitions and places shown in the final Petri net are mainly inserted for control purposes. Basically, they model the interarrival message rate in a way to avoid deadlock. The part of the Petri net model including t_{12}-p_9-p_{10}-t_{13}, blocks out any arriving voice messages while $m(p_1)=1$ and $m(p_2)=0$. Any messages that arrives at the station wlile $m(p_1)=1$ and $m(p_2)=0$ is stored in a temporary memory buffer, modeled by

t_{16}-p_{13}-t_{18} combination, and transfered to the voice queue as soon as the station goes idle, i.e. when $m(p_1)=0$. Similarly, any data message that arrives while $m(p_1)=1$ and $m(p_3)=0$ is stored in a temporary buffer, and transfered to the data queue as soon as the station goes idle. The temporary data buffer and the control system for preventing deadlock is modeled by t_{14}-p_{11}-t_{15}-p_{12}-t_{17}-p_{14}-t_{19} combination.

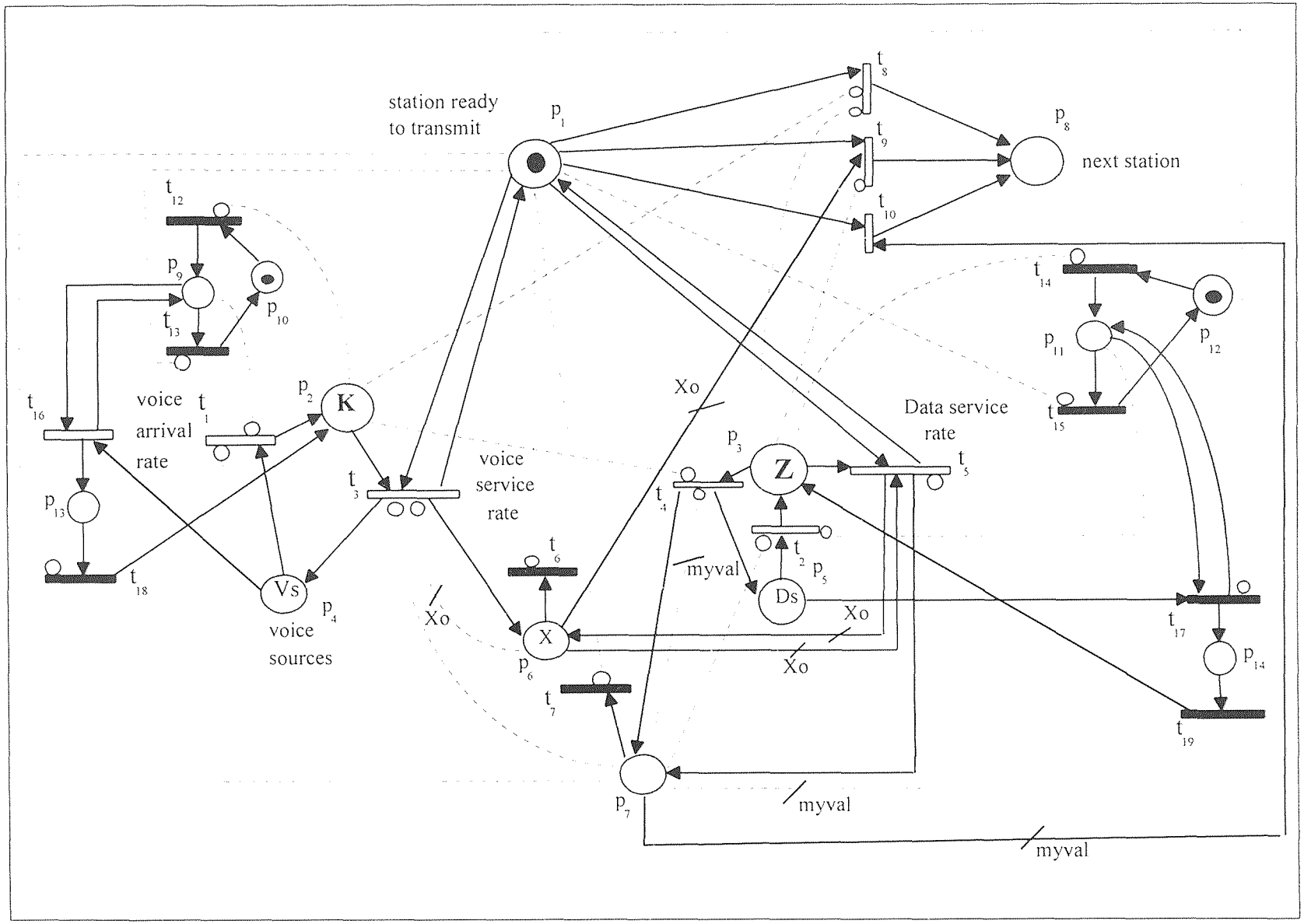

Figure 2-4. Petri Net model for a single station.

Table 2-1. Labels of places and transitions in final Petri Net Model.

NOTATION	DESCRIPTION
p_1	Token ring is available and the station is able to transmit
p_2	Synchronous (voice) messages ready to transmit
p_3	Asynchronous (data) messages ready to transmit
p_4-p_5	Voice-Data sources
p_6-p_7	# of voice-data messages transmitted
p_8	Next station on ring (idle)
p_9-p_{10}-p_{11}-p_{12}	Places for control purposes. They block out any arriving messages while $m(p_2)$ or $m(p_3)$=0, and $m(p_1)$=1
p_{13}-p_{14}	Voice-Data message temporary buffers
t_1-t_2	Voice-Data message arrival rate
t_3	Voice service rate
t_4-t_5	Data service rate
t_6-t_7	Immediate transitions for control purposes
t_8-t_9-t_{10}	Token ring transitions between successive stations
t_{11}	Connects rest of the net with first station
t_{12}-t_{13}-t_{14}-t_{15}	Transitions for control purposes
t_{16}-t_{17}	Voice-Data message arrival rate
t_{18}-t_{19}	Immediate transitions to transfer any messages from the temporary buffers to the queues
X [= $m(p_6)$]	# of voice messages transmitted so far
Y [= $m(p_7)$]	# of data messages transmitted so far
Xo	maximum number of voice messages a station can transmit per token ring holding time
myval [= (Xo+1) - X]	Variable to support FDDI protocol

CHAPTER 3

THE EFFECT OF THE NETWORK'S LOAD ON OVERALL PERFORMANCE

3.1 Performance Analysis of FDDI in other Approaches

As stated earlier in this thesis, FDDI uses a token-based protocol. Since it is a token ring LAN, its access delay clearly depends on the overall latency L.

The use of the timed token rotation protocol provides service to synchronous traffic, and limits the amount of asynchronous data traffic on the network. The data access delay and the maximum data throughput turn out to be dependent critically on the initial value for the Token Rotation Timer, T_Opr. The performance thus depends, in a rather complex way, on both the latency L and T_Opr.

This complexity of the protocol urges many engineers to study the performance analysis of the FDDI. The work of Karvelas and Leon-Garcia [Karvelas 88,90] developed simple worst-case bounds on the synchronous traffic delay (2T_Opr, as noted in the earlier discussion of the FDDI), the worst-case maximum data traffic throughput and a relatively simple data access delay approximation. The effect of L and T_Opr on the network's performance was a main focus of the study in these papers. The results of these papers, mainly based on simulation, help choosing the value of T_Opr.

3.2 Introduction to SPNP Software Package

The SPNP (Stochastic Petri Net Package) Version 3.0 used in this thesis was developed at Duke University [Ciardo 89]. The Stochastic Petri Net (SPN) to be analyzed must be described in a CSPL (C-based Stochastic Petri Net Language) file, which specifies the structure of the SPN and the desired outputs, by means of predefined functions.

The SPN model is obtained from the PN model by associating a *probability distribution function* to the firing time of each transition. Transitions with an associated exponential distribution are said to be *timed*; transitions with a constant 0 distribution are said to be *immediate*.

3.3 Network's Modified Model

To study the effect of the network's load on any individual station throughput, a simplified network is proposed. The general N station network shown in Fig. 1-1, is replaced by the network shown in Fig. 3-1, by simply replacing the part of the network from station 2 through station N, by a single station, called "station N-1".

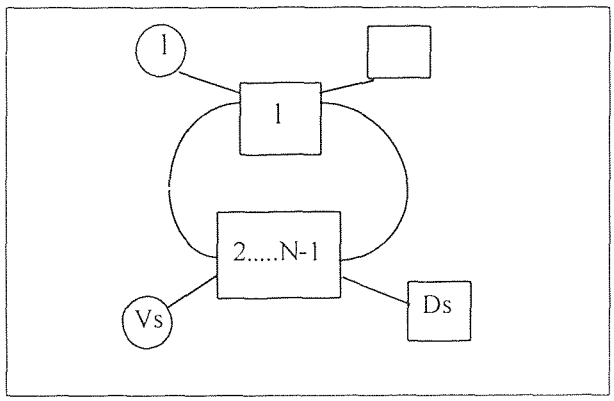

Figure 3-1. Modified N station network.

The Petri net version of the newly revised network, can be constructed by simply modeling the N-1 stations by a single place. The walking time throughout the N-1 stations, plus the service time of the N-1 stations is represented by the transition t_{11}. *The higher the transition's t_{11} rate is, the lighter the network load is.* Figure 3-2 shows the complete revised ring network using Petri nets.

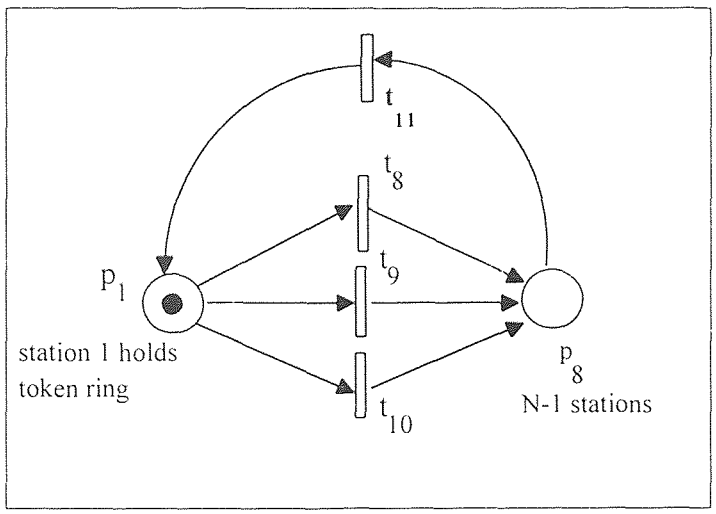

Figure 3-2. Modified N station Petri network.

It should be noted that the notations of the places and the transitions shown in Fig. 3-2 correspond to the same places and transitions shown in the original Petri net model of Fig. 2-4.

3.4 FDDI Performance using SPNP

The final PN model of an individual station proposed in Fig. 2-4, is connected as a part of an N station network using the method shown in Fig. 3-2. The modified network is described in CSPL file and executed using the SPNP software package. The C-code for the network is listed in Appendix A.

The effect of the network's parameters on the individual station's throughput are examined in this part of the thesis. The main focus of the study is on the relation between:

1. The number of voice sources presented at the station vs the average voice throughput of the station;
2. The speed/length of the net (latency) vs the average voice throughput of the station;
3. The effect of data sources on average voice and data throughputs of the station.

The results of the SPNP analysis are listed in Appendix D. The following figures are sketched using the results obtained by the SPNP analysis..

Figure 3-3 presents the relation between the number of voice sources in an individual station, and its average voice throughput. Since the effect of data sources is not covered in this figure, the individual station to be examined, is assumed to have no data sources.

Figure 3-4 presents the effect of the (N-1) stations walking and service rate on the voice throughput of the individual station. This case does in fact show the effect of the network's speed and latency on the individual station's throughput rate.

Figure 3-5 shows the effect of data sources presented in a station, on its average voice/data throughput.

Finally, Fig. 3-6 shows the effect of the (N-1) stations on the individual's voice/data throughput.

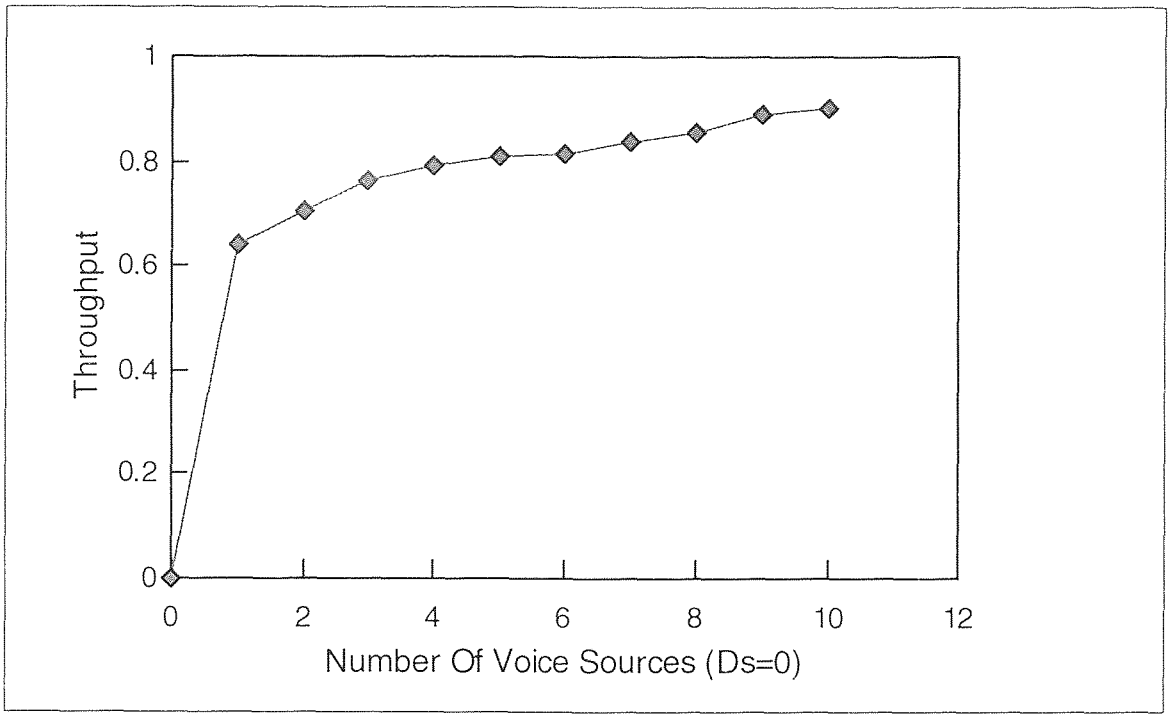

Figure 3-3. Voice Throughput (100K Messages per Second) vs. number of Voice Sources (Ds=0).

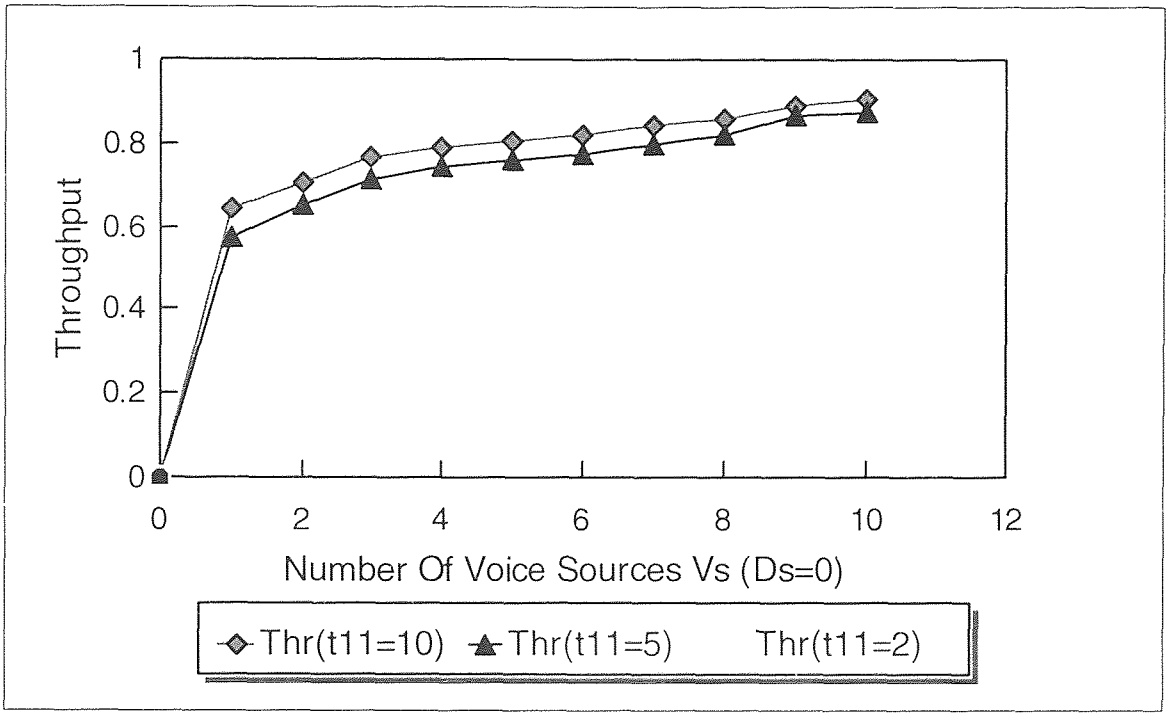

Figure 3-4. Voice Throughput (100K Messages per Second) vs. number of Voice Sources (Ds=0) for variable latency.

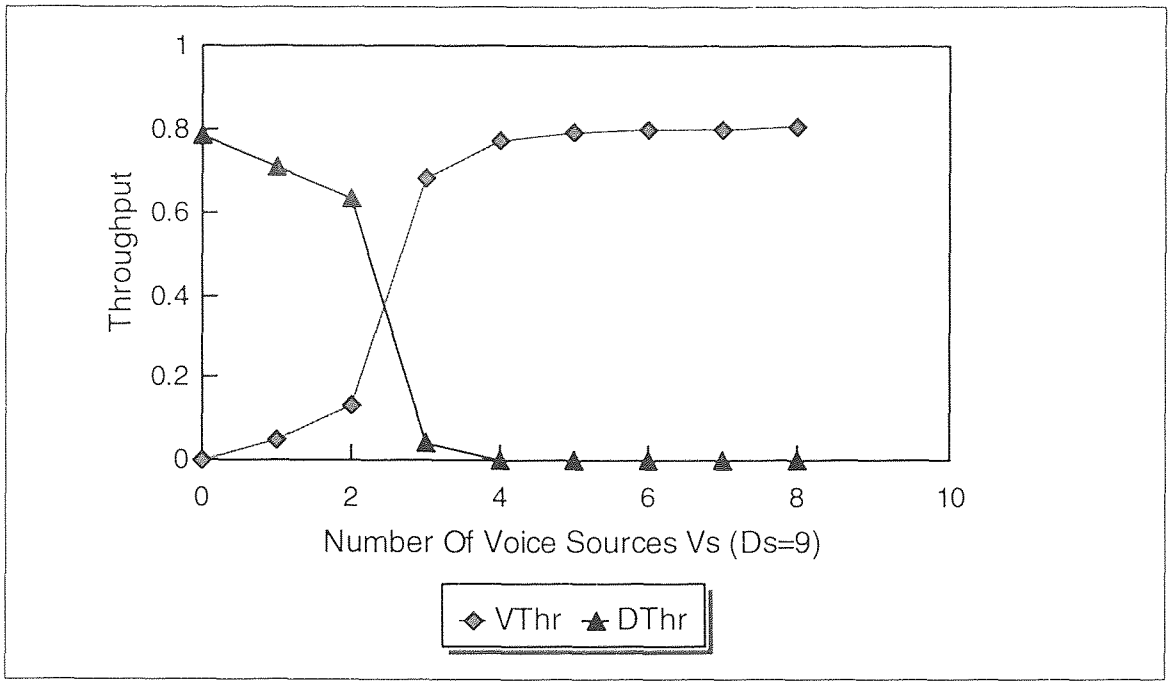

Figure 3-5. Voice and Data Throughput (100K Messages per Second) vs. number of Voice Sources (Ds=9)

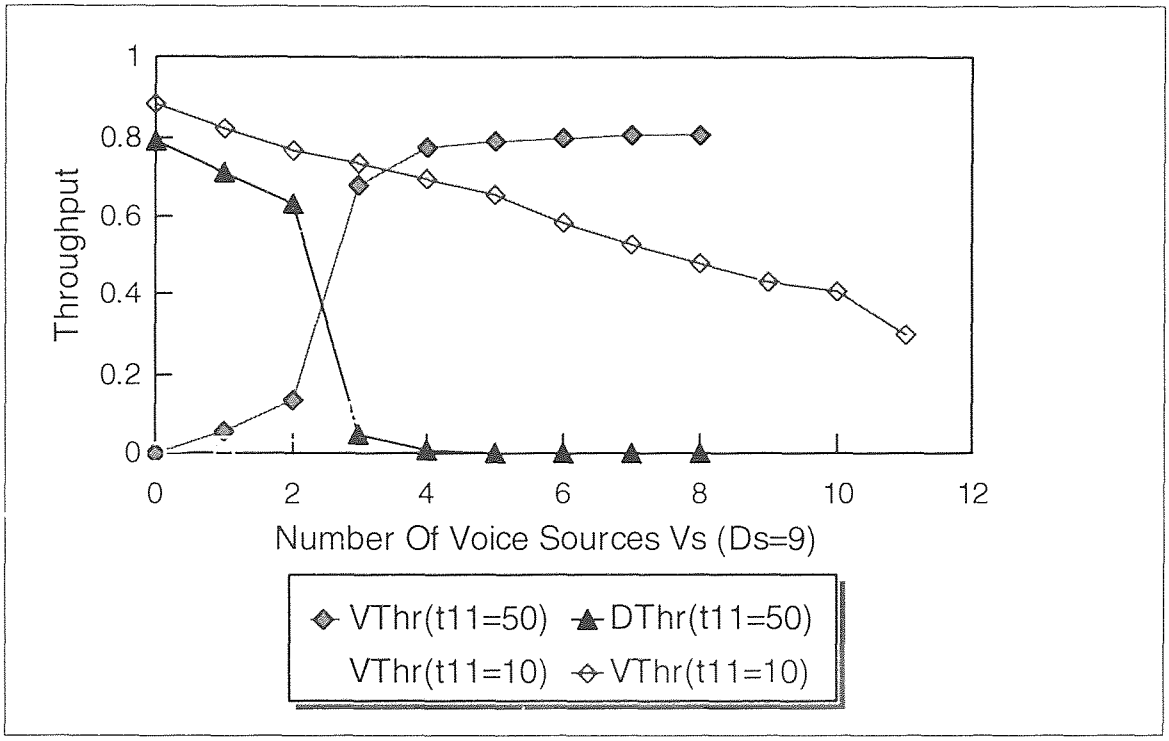

Figure 3-6. Voice and Data Throughput (100K Messages per Second) vs. number of Voice Sources (Ds=9) for Variable Latency.

3.5 Analysis of Results

- The upper-bound limit on the average voice throughput is shown in Fig. 3-3. Even with no data sources available in the station, voice throughput cannot increase unlimitedly. The upper bound limit is controlled by ST in the time-driven protocol and by the "number of tokens in place p_6" in the discrete-driven system proposed in this thesis. In this case, the number of tokens in place p_6, cannot exceed Xo, i.e., after the station transmits a maximum of Xo messages, it should transmit data or pass the token to the next station.

- The effect of the latency on the network performance is presented in Fig. 3-4. The faster the token goes around the ring, the higher the individual station's throughput is. By gaining speed we are gaining efficiency. Short queues of voice messages ready to transmit are formed causing short transmission delays.

- Figure 3-5 demonstrates the priority of voice over data. It is clear that, by increasing the number of voice sources, while keeping the number of data sources constant, the voice throughput is increasing at the expense of data throughput. Voice throughput is increased up to a certain upper-bound limit, while data throughput approaches to zero. In other words, by gaining voice efficiency, long data queue delay and low-level data efficiency, are created.

- The results of the three previous figures are summarized in Fig. 3-6. Voice priority over data, upper-bound limit in voice efficiency, long data queue delays for large number of voice sources in the network, as well as, latency effect on station's voice/data efficiency can be seen in the later graph.

As it is shown so far, there is no such an ideal combination of parameters for maximum efficiency. Network's speed (latency), throughput and queue delays, are related in a very complicated way. Depending on the network we are planning to develop, we should consider different combinations of parameters.

CHAPTER 4

THE EFFECT OF THE T-OPR ON THE NETWORK'S PERFORMANCE

4.1 Introduction

This chapter focuses on the effect of T_OPR on the network's performance. Previous papers on this subject, such as [Karvelas 88] and [Karvelas 90], proved that the value of T_Opr is bounded by a default minimum value of 4 msec and a default maximum value of 165 msec. The latter value is chosen to ensure stable ring recovery [ANSI 87] . For asynchronous traffic higher throughput is possible with a larger T_Opr, but the maximum asynchronous frame delay increases correspondingly.

Since this thesis, however, deals with DEDS, the optimum value of T_Opr (and allotment time ST), will be as a function of "how many messages per token ring holding time, a station can transmit - Ω ". The theoretical approach, as well as the analytical approach using SPNP, will be initially done, using a two-station-network case. It will, then be extended to the N-station network case, for general conclusions. The simplest two-station network is shown in Fig. 4-1.

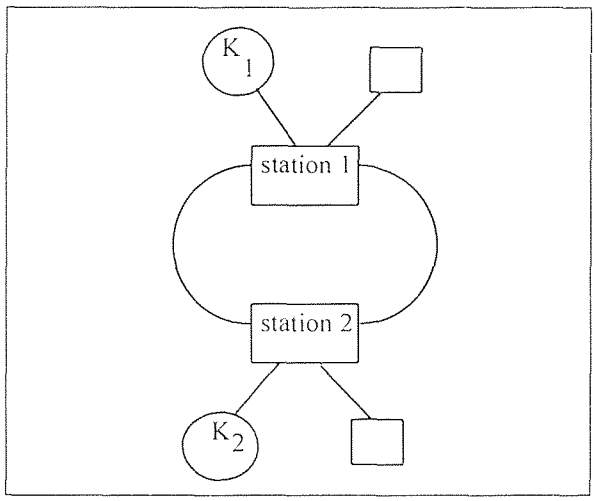

Figure 4-1. Two-station network.

4.2 Two-Identical-Station Network

A simple network consisting of two identical stations having initially K voice messages to transmit, will be the basis for this chapter's approach to studying the effect of Ω on network's time delays.

This section aims to estimate the number of Transmitted Messages per Token Holding Time, denoted by Ω, that produces the minimum time (propagation and service time), in transmitting the K messages from each station.

The optimum value for the two-identical-station network can be simply evaluated through the following two examples.

Example 1 (K is even)

Assume that two station's PNs are joined together forming an FDDI LAN. Each station has initially K=8 voice messages to transmit.

Find how many messages each station should transmit every time it holds the ring token, in order to have minimum delays.

Solution:

A simple way for determining the optimal value of Ω is to enumerate all of its possible values, i.e. Ω =1,2,3...8, and obtain the minimum time to transmit all messages from each station.

A detailed analysis of these cases follows:

Case 1: $\Omega = 1$

In order to transmit all initial K=8 messages, each station should receive the ring token eight times. Note that the messages arriving to the station with rate λ are not taken into consideration at this point.

Case 2: $\Omega = 2$

In this case, each station has to transmit the messages two at a token holding time, resulting to a total of four token holding times.

Case 3: $\Omega = 3$

For this case, each station transmits three messages per token holding time. The following scheme can then be obtained:

Number of messages sent from Station 1	Number of messages sent from Station 2
3	
	3
3	
	3
2	

It can be clearly seen that each station should receive the ring token three times to transmit all initial messages.

A similar pattern is obtained for the rest of the cases, and is summarized in Table 4-1. Note that the guaranteed transmission of one data message per cycle, provided in the Time Token Rotation Protocol described in Section 1.4, is considered in the table.

Table 4-1. The effect of Ω value on two-identical-station network initially having K=8 messages to transmit.

	Station 1 K_1=8	**Station 2 K_2=8**	**Total time to transmit K_1=8 messages (station 1 receives the token at t=0)**
Ω=3	3/3/2	3/3	$t = 2L + 14V/\mu + 2D/\mu$
Ω=4	4/4	4	$t = L + 12V/\mu + D/\mu$
Ω=5	5/3	5	$t = L + 13V/\mu + D/\mu$
Ω=6	6/2	6	$t = L + 14V/\mu + D/\mu$
Ω=7	7/1	7	$t = L + 15V/\mu + D/\mu$

where:

L is the propagation delay for a complete token ring cycle (latency)

μ is time required to transmit a single message (service rate)

V is the Voice message length

D is the Data message length

Note that the Ω=8 case (exhaustive case) is ignored. For real life N station network where N>>2, such a case will produce long delays in all stations.

It is concluded then, that for the case K_1=K_2=8, Ω=4 is the optimal value for minimum delays.

Example 2 (K is odd)

Assume that two station's PNs are connected forming an FDDI LAN, as in Fig. 4-1. Each station has initially K=9 voice messages to transmit. Find out how many messages each station should transmit every time it holds the token, for minimum delays.

Solution:

The approach is similar to the one used in Example 1.

The total time to transmit K_1=9 messages from station 1, for all Ω cases is shown below in Table 4-2:

Table 4.2 The effect of Ω value on two-identical-station network initially having K=9 messages to transmit.

	Station 1 K_1=9	**Station 2 K_2=9**	**Total time to transmit K_1=9 messages (station 1 receives the token at t=0)**
Ω=2	2/2/2/2/1	2/2/2/2	$t = 4L + 17V/\mu + 4D/\mu$
Ω=3	3/3/3	3/3	$t = 2L + 15V/\mu + 2D/\mu$
Ω=4	4/4/1	4/4	$t = 2L + 17V/\mu + 2D/\mu$
Ω=5	5/4	5	$t = L + 14V/\mu + D/\mu$
Ω=6	6/3	6	$t = L + 15V/\mu + D/\mu$
Ω=7	7/2	7	$t = L + 16V/\mu + D/\mu$
Ω=8	8/1	8	$t = L + 17V/\mu + D/\mu$

Now, it is obvious that for the case of $K_1=K_2=9$, Ω=5 produces the minimum delay.

Conclusion 1

For a two-identical-station FDDI network, initially having K messages ready to send, the optimal value for Ω for minimum delay, is:

$$\Omega = \lceil K/2 \rceil \qquad (4.1)$$

The minimum time to transmit all messages presented at any station starting the clock time as soon as the station receives the ring token is:

$$t_{min} = L + (K + \lceil K/2 \rceil) * V / \mu + D / \mu \qquad (4.2)$$

By induction the results obtained can be extended to the N-station network case:

- For N-identical station network, the optimum value of Ω for minimum delay is:

$$\Omega = \lceil K/2 \rceil \tag{4.3}$$

- The minimum delay to transmit K messages from station 1, in N-identical station network is:

$$t_{min} = L + (K + (N-1) * \lceil K/2 \rceil) * V / \mu + D / \mu \tag{4.4}$$

4.3 Two-Non-Identical-Station Network

In this section a more realistic non-identical station network will be studied. The approach, however, is the same used in section 4.1, and illustrated in the following example.

Example 3 (Non-identical stations)

Assume that two PN stations are connected forming an FDDI network. Initially, station 1 and station 2, have respectively K_1=8 and K_2=6 messages ready to transmit.

Find the value of the Ω which produces the minimum delay. Calculate the time required to transmit all initial K_1 messages.

Solution

A similar approach, applies to this example as well. Table 4-3 shows all possible values of Ω and their performance results on this specific example.

Table 4-3. The effect of Ω value on two-non-identical-station network initially having K_1=8 and K_2=6 messages to transmit. Station 1 receives the token at t=0.

	Station 1 K_1=8	**Station 2 K_2=6**	**Total time to transmit K_1=8 messages (station 1 receives the token at t=0)**
Ω =2	2/2/2/2	2/2/2	t = 3L + 14V/μ + 3D/μ
Ω=3	3/3/2	3/3	t = 2L + 14V/μ + 2D/μ
Ω=4	4/4	4	t = L + 12V/μ + D/μ
Ω=5	5/3	5	t = L + 13V/μ + D/μ
Ω=6	6/2	6	t = L + 14V/μ + D/μ
Ω=7	7/1	7	t = L + 15V/μ + D/μ

The above table covers the case when station 1 with K_1=8 holds the token at t=0. It seems that Ω=4 is the optimal value for this case.

To show however, that the result is independent of which station holds the token at t=0, a similar table should be formed in which station 2, (K_2=6), holds the token at t=0. Table 4-4 shows the results of the later case:

Table 4-4. The effect of Ω value on two-non-identical-station network initially having K_1=8 and K_2=6 messages to transmit. Station 2 receives the token at t=0.

	Station 2 K_2=6	**Station 1 K_1=8**	**Total time to transmit K_2=6 messages (station 2 receives the token at t=0)**
Ω=2	2/2/2	2/2/2	t = 2L + 12V/μ + 2D/μ
Ω=3	3/3	3	t = L + 9V/μ + D/μ
Ω=4	4/2	4	t = L + 10V/μ + D/μ
Ω=5	5/1	5	t = L + 11V/μ + D/μ

Table 4-4 does not cover Ω=6 and Ω=7 cases due to exhaustive-service-case reasons. It does however, suggests a different value of Ω, namely Ω=3. Between the two values suggested by the two different tables, the value to be used is Ω=4. By using this choice, less delays are encountered, mainly dependent on the overall propagation delay rather than the transmission delay.

Conclusion 2

For a two-non-identical-station FDDI network, initially having K_1 and K_2 messages ready to send respectively, the optimal value for Ω, for minimum delay, is:

$$\Omega = \lceil K_{max}/2 \rceil \tag{4.5}$$

The minimum time to transmit all messages presented at any station i starting the clock time as soon as the station i receives the ring token is:

$$t_{min} = L + (K_i + \lceil K_{max}/2 \rceil) * V / \mu + D / \mu \tag{4.6}$$

The two-non-identical-station network approach, can be extended to an N-non-identical station network.

- For N-identical station network, the optimum value of Ω for minimum delay is:

$$\Omega = \lceil K_{max}/2 \rceil \tag{4.7}$$

- The minimum time to transmit all K_i messages presented at station i (t=0 when station i receives the token) is given by:

$$t_{min} = L + (K_i + (N-1) * \lceil K_{max}/2 \rceil) * V / \mu + D / \mu \tag{4.8}$$

4.4 Real Life N-Non-Identical-Station Network

The previous two sections cover very specific, and actually, non realistic network cases. They suggest however, very important results, which are useful to complete the performance analysis of real life N station network.

As mentioned in earlier chapters, during ring initialization, T_Opr is acquired by all stations. Each station receives a fractional allotment ST of this time to be used in transmitting synchronous traffic. If, for example, there are Vs synchronous sources on the ring, the total synchronous allotment is then Vs*ST sec. This mainly implies that not necessarily all stations have the same allotment time to transmit messages.

In the DEDS framework, all stations should not necessarily transmit the same number of messages per holding token time. Different values of Ω can then be applied to individual stations, according to the number of sources (messages) presented at the initialization of the network. Basically, each station acquires its own value of Ω and according to Eqn. 4.1 this value of Ω is equal to the ceiling function of the initial messages presented at the individual station, divided by two.

Based on this, station 1 (K_1=8), and station 2 (K_2=6) will not have the same value of Ω as suggested in example 3. Station 1 will now transmit four messages per token holding time (Ω_1=4), whereas station 2 will transmit three messages per token holding time (Ω_2=3). A new transmitting message sequence can then be obtained, as shown below:

Number of messages sent from Station 1 (K_1=8, Ω_1=4)	Number of messages sent from Station 2 (K_2=6, Ω_2=3)
4	
	3
4	

The above scheme indicates that the minimum time to transmit all messages initially presented at station 1 (t=0 when station 1 receives the token) is:

$$t_{min} = L + (K_1 + \lceil K_2/2 \rceil) * V / \mu + D / \mu \qquad (4.8)$$

Conclusion 3

For a real N (non identical) station FDDI network,

- the optimum number of messages, Ω_i, transmitted per ring token holding time by station i, (station i having initially K_i messages ready to transmit), is:

$$\Omega_i = \lceil K_i/2 \rceil \qquad (4.9)$$

- the minimum time to transmit all messages presented at station 1, starting the clock time as soon as station 1 receives the token, is:

$$t_{min} = L + (K_1 + \sum_{i=2}^{N} \lceil K_i/2 \rceil) * V / \mu + D / \mu \qquad (4.10)$$

CHAPTER 5

THE EFFECT OF T-OPR USING SPNP ANALYSIS

5.1 Introduction

The previous chapter covers the theoretical approach to the effect of the T_Opr on the overall network's performance. Important results regarding the optimal value of the number of messages transmitted per token ring holding time, denoted by Ω, as well as the minimum time needed to transmit all messages presented at the stations, were drawn. In addition to the results presented in the previous chapter, a computer software package (SPNP) is utilized to reinforce the conclusions drawn. The reason this software is chosen is because of its capability to solve problems involving stochastic Petri Nets and Markov chains.

5.2 N-Identical-Station Network

The first case to be examined, is the simplest N identical-station network introduced in Fig. 1-1. It consists of N stations connected back to back forming a token ring. For simplicity reasons, all N stations in the network are identical and they all have K messages initially ready to transmit, i.e. $K_1=K_2=...=K_n=K$. The value of K is chosen randomly to be equal to eight.

Due to software limitations, however, the N station network was designed and simulated, using the approach illustrated by Fig. 3-1. The interrelation between Ω and

voice throughput, can be examined by altering the value of Ω in the network described by the CSPL file, listed in Appendix B. Figure 5-1 illustrates this interrelation.

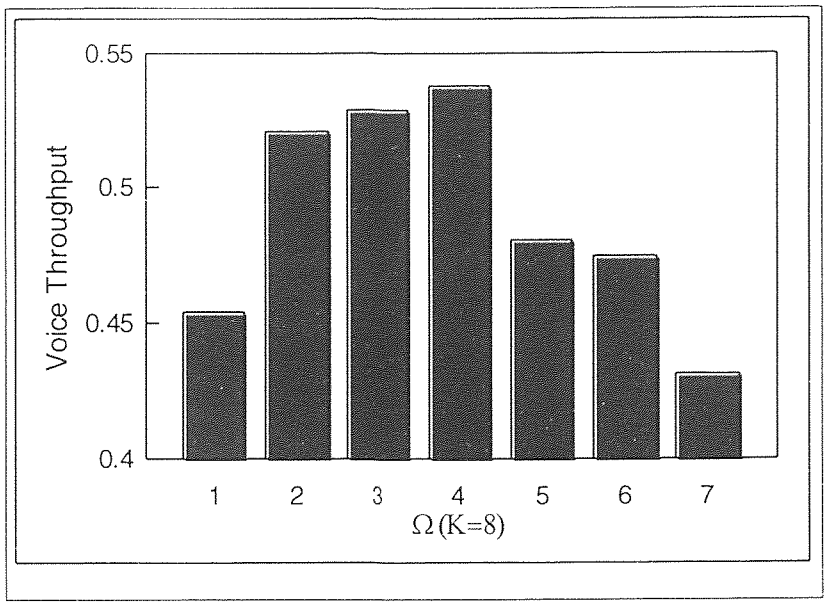

Figure 5-1. The effect of Ω on the Voice Throughput, in an N-identical-station network (K is even and equal to eight).

The stochastic Petri Net analysis (Fig. 5-1) suggests the same value of Ω for maximum throughput, as the one suggested by the theoretical approach discussed in Chapter 4.

To be more consistent with Chapter 4's results, however, an N-identical-station network is examined where K's value is even. The value of K is chosen to be equal to nine. The output voice throughput as a function of the value of Ω is obtained using the SPNP analysis. The outcomes of the computer's analysis, are plotted in the Fig. 5-2.

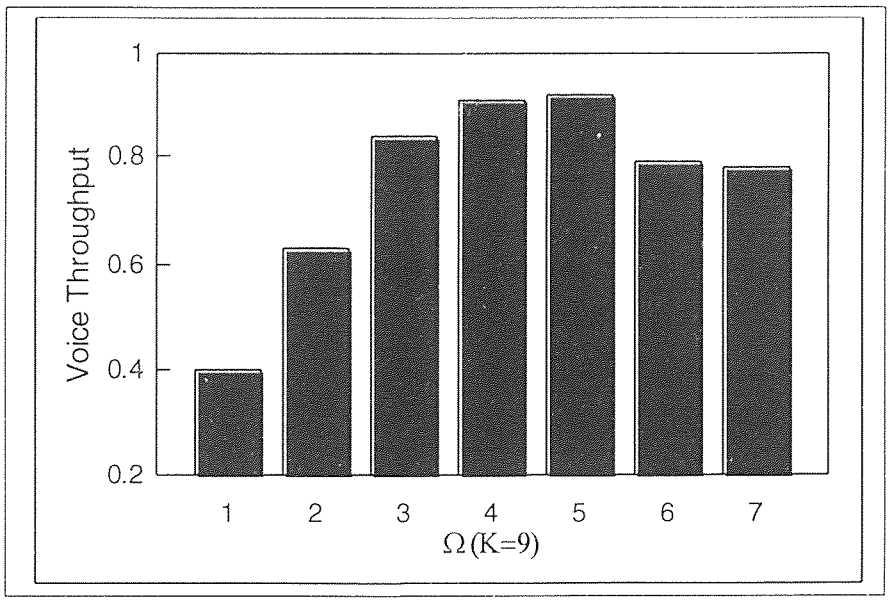

Figure 5-2. The effect of Ω on the Voice Throughput, in an N-identical-station network (K is odd and equals to nine).

Conclusion

SPNP analysis does in fact proposes, a value of Ω for maximum Voice Throughput. The value of Ω depends on the initial messages presented at the stations. Its value is equal to the ceiling function of the number of messages presented, divided by two, i.e.,

$$\Omega = \lceil K/2 \rceil$$

5.3 N-Non-Identical-Station Network

A more general case token ring network is to be examined in this section of the thesis. It consists of N non-identical stations connected as shown in Fig. 1-1. Since this network consists of non-identical stations, the approach suggested by Fig. 3-1 in Chapter 3 cannot be applied. To overcome the software's limitations as well as the non-identical stations

presence, a modified network is proposed. The newly modified network consists of two non-identical stations plus a station representing the remaining N-2 stations. Figure 5-3 pictures the newly modified ring network, whereas Fig. 5-4 shows the part of the Petri Net version, which is responsible for the ring connection.

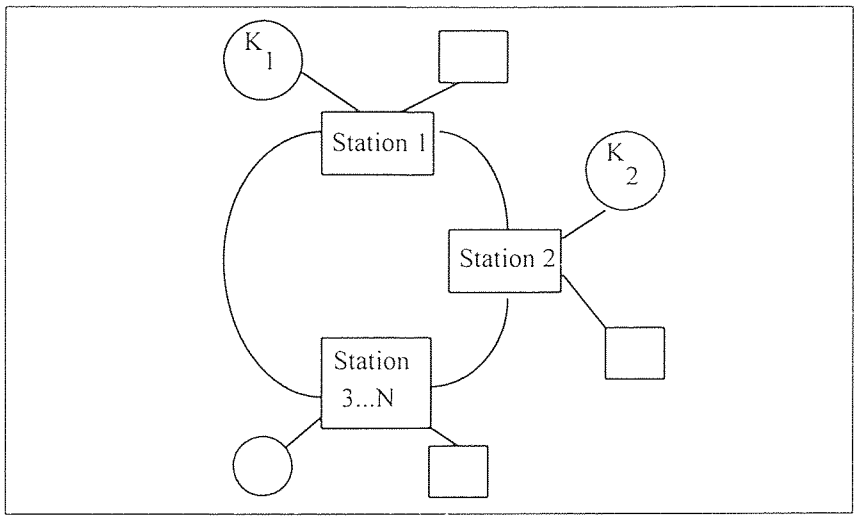

Figure 5-3. Modified N Non-identical-station token ring network.

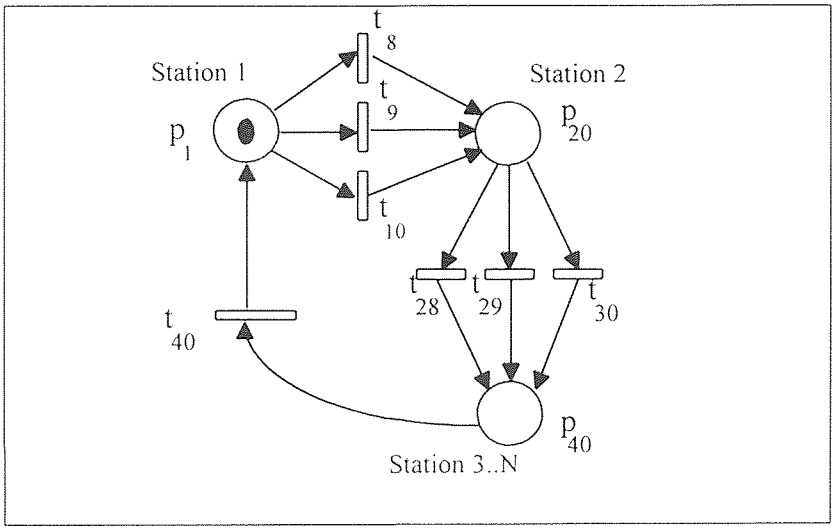

Figure 5-4. PN model representing the N Non-identical-station ring network.

It should be noted here, that the place p_{40} represents the N-2 stations, and the transition t_{40} the propagation delay through the distance between the last N-2 stations. It should be noted also, that the labels used in the Fig. 5-4, are the same used in the code listed in Appendix C, describing the complete network.

By assigning different values of K to stations 1 and 2, an N-non-identical-station is designed. The analysis of the network is performed using the code of Appendix C. Figure 5-5 pictures the effect of Ω on Station 1's Voice Throughput, for the case K_1=8 and K_2=6. Note that, Fig. 5-5 demonstrates the case, where the same value of Ω is assigned to both station 1 and station 2.

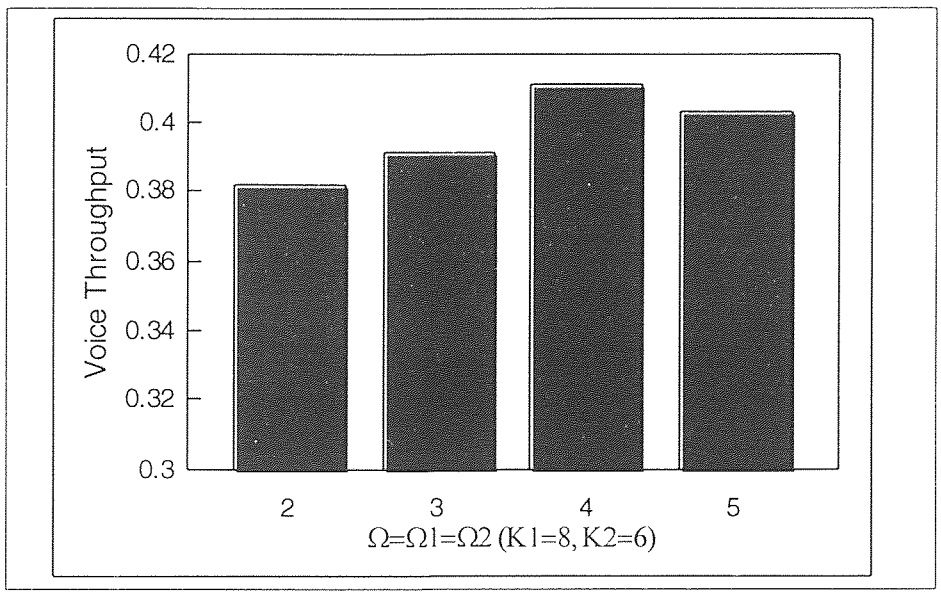

Figure 5-5. The effect of Ω on the Voice Throughput, in an N-non-identical-station network. The same value of Ω is assigned to all stations.

Conclusion

Both SPNP analysis and theoretical approach developed in Chapter 4, reinforce the same value of Ω for maximum voice throughput. For unique Ω value throughout the network, W should follow Eqn. (4.4), i.e.,

$$\Omega = \lceil K_{max}/2 \rceil$$

5.4 Real N-Station Network

The use of the SPNP analysis, ends with the study of an actual network. It consists of N, not necessarily identical, stations. As mentioned in previous chapters, the value of Ω varies from station to station throughout the network. Since, every station has a unique optimal value of Ω for its maximum throughput, N different values of Ω can be assigned to the network. The value of $\Omega=(\Omega_1, \Omega_2,...,\Omega_n)$ assigned to station i=(1, 2,..n) depends on the number of voice messages K_i initially available at station i.

The aim of this section is to analyze the network performance using the real network's parameters and compare the results with the results drawn by simulating the non-realistic network examined in Section 7.3. By assigning Ω=(4,3) to stations 1 and 2 of the network of Section 5.3, a realistic network is formed. Figure 5-6 presents the difference on the network's performance, between the realistic network with Ω=(4,3) and the non-realistic one with identical Ω throughout all stations.

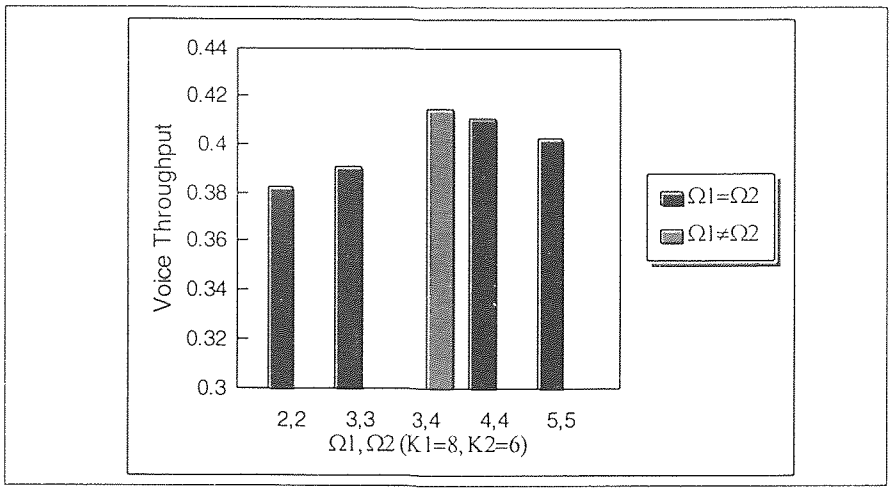

Figure 5-6. Comparison between realistic and non-realistic networks.

It is obvious that by assigning different values of Ω to individual stations, better throughput can be achieved. And in fact, this is what exactly happening in real FDDI networks.

Conclusion

The assignment of correct value of ST or Ω at the initialization of the network can cause tremendous difference on the networks performance.

CHAPTER 6

CONCLUSION

6.1 Contribution of Petri Net FDDI Model

The Petri Net model proposed in this thesis, can provide better results on the network's performance, than the ones currently supported by the TTR protocol's timers. The advantages of using such a model are summarized below.

1. Petri Nets have the ability to check the system for undesirable properties, such as deadlock and instability.
2. Petri Nets can achieve better throughput, and create fewer delays. This is due to the fact that in case of a message error/correction, the PN model is able to locate the specific message causing the error, and retransmit only that specific message. On the other hand, in the case of an error in the TTR protocol, the station should retransmit its entire previous transmission, even though the error is spotted immediately. Since the transmission of the entire process takes longer, such a TTR process is much more time-consuming than the corresponding PN model process, which is the transmission of a single PN message.
3. The Petri Net model guarantees message completion in all cases, since it transmits messages, and not bits.

4. The PN model proposed in this thesis, supports non-exhaustive service for the token ring system, which is very rarely seen in other published papers.
5. Finally, this model can provide results for asymmetric networks, which are difficult to be examined by queueing methods.

6.2 Limitations of Petri Net FDDI Model

The Petri Net model proposed in this thesis, however, does not support all the features that the TTR protocol supports. The PN model's limitations are listed below.

1. The PN model does not support the error correction mechanism.
2. The PN model does not support the eight (8) priority levels supported by the TTR protocol.
3. The PN model does not consider the walking time between the stations.

It should also be noted that the PN model used was developed and implemented using the SPNP software, and therefore affected by the limitations of the SPNP software itself. The introduction of these limitations in the modeling approach could not be avoided, since the PN FDDI model was developed around the SPNP software. The most important limitations of the SPNP software package that were encountered during the development and implementation of the PN model are,

1. The upper bound limit on the length of Markov chain that can be solved by the software. This limits the number of stations that can be connected in the Token Ring to be examined.

2. The software output, limits the relationships that can be examined between the network's parameters. Namely, the software output that could be utilized in the deduction of conclusions, was the throughput. This throughput was correlated with the latency, and T_Opr, as discussed in this thesis. All other relationships between the operational parameters of the model, had to be deduced indirectly, based on the throughput's behavior.

6.3 Future Research

In this thesis we have investigated the performance of the FDDI network under Voice/Data integration. We have derived the constraints that the operational time parameter Ω must obey in order for the maximum voice throughput to be met. Finally, we have found, through analysis, that the system behavior is related in a very complicated way to the network's latency and T-Opr. Unfortunately, SPNP's limitations did not allow us to connect N individual stations and analyze the performance of an authentic N station network, in order to have a broader overview of the model.

Since, however, this model has the potential of being a strong protocol itself, it should be extended in order to support all FDDI's functions. High level Petri Nets, such as color PNs, should be involved to support the eight levels of priority. Also, the network should be modified to support the error-correction capability. Only after completion of all the above modifications, we will be able to thoroughly analyze the performance of the network.

Finally, a new software, capable to support color PNs and longer Markov chains, should be developed. Only with the help of such a powerful tool we will be able to demonstrate, what we have talked about, throughout this thesis.

APPENDIX A

C-Code Representing N-Station Network Using Fig. 3-1's Modeling Approach, to Study the Effect of Latency on Network's Performance

```
/* This program is the PETRI NET description for the Fiber Distributed Data Interface
(FDDI) network used in Chapter 3 to study the effect of the latency on overall network's
performance.
The output contains the steady-state transition rates, the token probabilities, the voice
throughput, the data throughput and the individual station's utilization */
#include "user.h"
        int X;
        int L;
        int myval() {return (9 - mark("p6"));}

/* This function declares the SPNP options */
        parameters() {
        iopt(IOP_PR_FULL_MARK, VAL_YES) ;
        iopt(IOP_PR_MC, VAL_YES) ;
        iopt(IOP_PR_RGRAPH, VAL_YES);
        iopt(IOP_PR_PROB, VAL_YES) ;

/* The user must input the voice message load ready to be transmitted by the station*/
        X = input("Number of voice messages already at the station service queue (value
from 0 to 10):");
/* The user must also input the rate value of transition t11 which presents the latency
through the N-1 stations*/
        L = input("Rate value of transition t11 (value from 1 to 20) : ");
        }

/*      There are 13 places and 19 transitions in each station PN model */
        net () {

/* place p1 represents the token ring. If there is a token in this place, the station is able to
transmit */
```

```
        place ("p1");   init("p1",1);

/* place p2 represents the queue of voice sources which are ready to transmit */
        place ("p2");   init("p2",X);

/* place p3 represents the queue of data sources which are ready to transmit */
         place ("p3");   init("p3",9);

/* place p4 and place p5 represent voice and data sources */
        place ("p4");    /*  init("p4",8); */
        place ("p5");    /*  init("p5",8); */

/* place p6 and place p7 represent how many voice and data messages are transmitted */
        place ("p6");
        place ("p7");

/* place p8 represents the token ring status of the next station.(Similar to p1 place) */
        place ("p8");

/* place p9, p10, p11 and p12 are for control purposes */
        place ("p9");
        place ("p10");     init("p10",1);
        place ("p11");
        place ("p12");     init("p12",1);
/*place p13 and p14 are used as memory buffers for the arriving voice/data messages.
The tokens in these places are transfered into the queues as soon as the station passes the
token to the next station.*/
        place ("p13"); place("p14");
```

```
/* Both exponential and immediate transitions are used. Most of the immediate
transitions are for net interconnections, and have a firing probability of 1.0. The rates for
the exponential transitions are declared below.*/
/* Transitions t1 and t2 represent voice and data message arrival rate respectively */
        trans ("t1");    rateval("t1",0.064);
        trans ("t2");    rateval("t2",0.064);
/* Transitions t3 and t4-t5 represent voice and data message service rate respectively */
        trans ("t3");    rateval("t3",1.0);
        trans ("t4");    rateval("t4",1.0);
        trans ("t5");    rateval("t5",1.0);
/* Transitions t6 and t7 are immediate transitions with a probability of 1.0. They are used
for control purposes. They are enabled after the token is passed to the next station */
        trans ("t6");     probval("t6",1.0); priority("t6",3);
        trans ("t7");     probval("t7",1.0); priority("t7",2);
/* Transitions t8, t9 and t10 are also immediate transitions. They are enabled under
different conditions and they actually pass the token to the next station after firing. */
        trans ("t8");    rateval("t8",5000.0);
        trans ("t9");    rateval("t9",5000.0);
        trans ("t10");   rateval("t10",5000.0);
/* Transition t11 represents the rest of the stations in the network */
        trans ("t11"); rateval("t11",L);
/* Transition t12, t13, t14 and t15 are immediate transitions for control purpose */
        trans("t12");  rateval("t12",5000.0);        trans("t13");  rateval("t13",5000.0);
        trans("t14");  rateval("t14",5000.0);        trans("t15");  rateval("t15",5000.0);
/*Transitions t16 and t17 represent voice and data message arrival rate. They are similar
to t1 and t2 respectively. In fact they are their complement in order to have interarriving
messages continually without deadlock. */
        trans("t16");   rateval("t16",0.064);  trans("t17");   rateval("t17",0.064);
/*Transitions t18 and t19 are immediate. As soon as the station passes the token to the
next station they transfer the voice/data message to the voice/data queues.*/
```

```
trans ("t18");     probval("t18",1.0);  priority("t18",3);
trans ("t19");     probval("t19",1.0);  priority("t19",2);

/* The rest of the net description consists of defining the arcs. */

iarc ("t1","p4");        harc("t1","p7");        harc("t1","p9");
oarc("t1","p2");

iarc ("t2","p5");        harc ("t2","p11");      vharc ("t2","p7",myval);
oarc("t2","p3");

iarc ("t3","p1");        iarc ("t3","p2");       oarc("t3","p6");
oarc("t3","p4");         mharc("t3","p6",8);     harc ("t3","p7");
oarc("t3","p1");

iarc ("t4","p1");        harc ("t4","p2");       oarc("t4","p7");
oarc("t4","p5");

iarc("t4","p3");         vharc("t4","p7",myval);      oarc("t4","p1");

iarc("t5","p1");         iarc ("t5","p3");       oarc("t5","p1");
oarc("t5","p7");

miarc("t5","p6",8);      moarc("t5","p6",8);     vharc("t5","p7",myval);

iarc ("t6","p6");        harc ("t6","p1");

iarc ("t7","p7");        harc ("t7","p1");

iarc ("t8","p1");        oarc("t8","p8");        harc ("t8","p2");
```

```
harc ("t8","p3");

iarc ("t9","p1");        oarc("t9","p8");        harc ("t9","p3");
miarc ("t9","p6",8);

iarc("t10","p1");        oarc("t10","p8");       viarc ("t10","p7",myval);

iarc("t11","p8");        oarc("t11","p1");

iarc("t12","p10");       oarc("t12","p9");       harc("t12","p2");

iarc("t13","p9");        oarc("t13","p10");      harc("t13","p1");

iarc("t14","p12");       oarc("t14","p11");      harc("t14","p3");

iarc("t15","p11");       oarc("t15","p12");      harc("t15","p1");

iarc("t16","p9");        iarc("t16","p4");
oarc("t16","p9");        oarc("t16","p13");

iarc("t17","p11");       iarc("t17","p1");
oarc("t17","p11");       oarc("t17","p14");

iarc("t18","p13");       harc("t18","p1");       oarc("t18","p2");

iarc("t19","p14");       harc("t19","p1");       oarc("t19","p3");
```

/* This function allows for the checking of illegal markings. It is not needed in this model, so always return a no error condition */

```
assert() {return (RES_NOERR);}
```

```
/* This function is called before the net analysis starts. Since the structure of the net is
fixed in this model, this function is not needed */
        ac_init() {}

/* This function is called with reachability graph information. */
        ac_reach(){fprintf(stderr,"/nThe reachability graph has been generated/n/n");}

/* Define output functions*/
/* Voice throughput */          reward_type ef0() {return (rate("t3"));}
/* Data throughput */           reward_type ef1 () {return(rate("t4") + rate("t5"));}
/* Station Utilization */       reward_type ef2 () { return ( mark("p1"));}

/* Output */
        ac_final() {
        pr_expected("voice throughput = ",ef0);
        pr_expected("data throughput = ",ef1);
        pr_expected("station utilization = ",ef2);
        pr_std_average ();
                }
```

APPENDIX B

C-Code Representing N-Station Network Using Fig. 3-1's Modeling Approach, to Study the Effect of Ω on Network's Performance

```
/* This program is the PETRI NET description for the Fiber Distributed Data Interface
(FDDI) network consisting of N identical stations.It uses the "N-1 station place" method
and it is used to examine the optimal value of Ω for better performance */

#include "user.h"
        int Ω;
        int X;
        int myval() {return ((Ω+1) - mark("p6"));}

/* This function declares the SPNP options */
parameters() {
        iopt(IOP_PR_FULL_MARK, VAL_YES) ;
        iopt(IOP_PR_MC, VAL_YES) ;
        iopt(IOP_PR_RGRAPH, VAL_YES);
        iopt(IOP_PR_PROB, VAL_YES) ;

/* The user must input the voice message load ready to be transmitted by the station*/
        X = input("Number of voice messages already at the station service queue (value
from 0 to 10):");
        Ω = input("Max number of tokens(messages) transmitted per token holding time
(value from 1 to 8):");          }

        net () {
        place ("p1");   init("p1",1);
        place ("p2");   init("p2",X);
        place ("p3");   init("p3",5);
        place ("p4");   place ("p5");
        place ("p6");   place ("p7");
        place ("p8");
```

```
place ("p9");   place ("p10"); init("p10",1);
place ("p11"); place ("p12"); init("p12",1);
place ("p13"); place("p14");

trans ("t1");    rateval("t1",0.064);    trans ("t2");    rateval("t2",0.064);
trans ("t3");    rateval("t3",1.0);      trans ("t4");    rateval("t4",1.0);
trans ("t5");    rateval("t5",1.0);
trans ("t6");    probval("t6",1.0);      priority("t6",3);
trans ("t7");     probval("t7",1.0);     priority("t7",2);
trans ("t8");    rateval("t8",5000.0);   trans ("t9");    rateval("t9",5000.0);
trans ("t10");   rateval("t10",5000.0);  trans ("t11");   rateval("t11",1.0);
trans("t12");    rateval("t12",5000.0);  trans("t13");    rateval("t1 3",5000.0);
trans("t14");    rateval("t14",5000.0);  trans("t15");    rateval("t15 ",5000.0);
trans("t16");    rateval("t16",0.064);
trans("t17");    rateval("t17",0.064);
trans ("t18");     probval("t18",1.0);   priority("t18",3);
trans ("t19");     probval("t19",1.0);   priority("t19",2);
```

/* The rest of the net description consists of defining the arcs. */

```
iarc ("t1","p4");        oarc("t1","p2");        harc("t1","p7");
harc("t1","p9");
iarc ("t2","p5");        oarc("t2","p3");        harc ("t2","p11");
vharc ("t2","p7",myval);
iarc ("t3","p1");        oarc("t3","p4");        iarc ("t3","p2");
oarc("t3","p6");
mharc("t3","p6",Ω);    oarc("t3","p1");        harc ("t3","p7");
iarc ("t4","p1");        oarc("t4","p5");        harc ("t4","p2");
oarc("t4","p7");
oarc("t4","p1");         iarc("t4","p3");        vharc("t4","p7",myval);
```

```
iarc("t5","p1");        iarc ("t5","p3");        oarc("t5","p7");
oarc("t5","p1");
miarc("t5","p6",Ω);   vharc("t5","p7",myval);      moarc("t5","p6",Ω);
iarc ("t6","p6");       harc ("t6","p1");
iarc ("t7","p7");       harc ("t7","p1");
iarc ("t8","p1");       oarc("t8","p8");         harc ("t8","p2");
harc ("t8","p3");
iarc ("t9","p1");       oarc("t9","p8");
harc ("t9","p3");       miarc ("t9","p6",Ω);
iarc("t10","p1");       oarc("t10","p8");        viarc ("t10","p7",myval);
iarc("t11","p8");       oarc("t11","p1");
iarc("t12","p10");      oarc("t12","p9");        harc("t12","p2");
iarc("t13","p9");       oarc("t13","p10");       harc("t13","p1");
iarc("t14","p12");      oarc("t14","p11");       harc("t14","p3");
iarc("t15","p11");      oarc("t15","p12");       harc("t15","p1");
iarc("t16","p9");       iarc("t16","p4");
oarc("t16","p9");       oarc("t16","p13");
iarc("t17","p11");      iarc("t17","p1");
oarc("t17","p11");      oarc("t17","p14");
iarc("t18","p13");      harc("t18","p1");        oarc("t18","p2");
iarc("t19","p14");      harc("t19","p1");        oarc("t19","p3");
 }
assert() {return (RES_NOERR);}
ac_init() {}
ac_reach(){fprintf(stderr,"/nThe reachability graph has been generated/n/n");}

/* Define output functions*/
/* Voice throughput */        reward_type ef0() {return (rate("t3"));}
/* Data throughput */         reward_type ef1 () {return(rate("t4") + rate("t5"));}
```

```
/* Station Utilization */          reward_type ef2 () { return ( mark("p1"));}
 /* Output */
ac_final() {
        pr_expected("voice throughput = ",ef0);
        pr_expected("data throughput = ",ef1);
        pr_expected("station utilization = ",ef2);
        pr_std_average ();
                }
```

APPENDIX C

C-Code Representing N-Station Network Using Fig. 5-3's Modeling Approach, to Study the Effect of Ω on Network's Performance

```
/* This program is the PETRI NET description for an N station network. It covers both
identical and non-identical stations cases. It is based on Fig. 7-3  approach.*/

#include "user.h"
        int X;
        int Ω1;
        int Ω2;
        int myval() {return ((Ω1+1) - mark("p6"));}
        int myval2() {return((Ω2+1)- mark("p26"));}
parameters() {
        iopt(IOP_PR_FULL_MARK, VAL_YES) ;
        iopt(IOP_PR_MC, VAL_YES) ;
        iopt(IOP_PR_RGRAPH, VAL_YES);
        iopt(IOP_PR_PROB, VAL_YES) ;
/* The user must input the voice message load ready to be transmitted by the station*/
        X = input("Number of voice messages already at the station service queue (value
from 0 to 10):");
        Ω1 = input("Max number of voice  messages transmitted per token ring holding
time by station 1(value from 1 to 8):");
        Ω2 = input("Max number of voice  messages transmitted per token ring holding
time by station 2 (value from 1 to 8):");
                }
net () {
        place ("p1");   init("p1",1);    place ("p2");   init("p2",X);
        place ("p3");   init("p3",1);    place ("p4");   place ("p5");
        place ("p6");   place ("p7");    place ("p9");   place ("p10"); init("p10",1);
        place ("p11"); place ("p12"); init("p12",1);  place ("p13"); place ("p14");

        trans ("t1");     rateval("t1",0.064);
```

```
trans ("t2");    rateval("t2",0.064);
trans ("t3");    rateval("t3",1.0);       trans ("t4");    rateval("t4",1.0);
trans ("t5");    rateval("t5",1.0);
trans ("t6");     probval("t6",1.0);      priority("t6",3);
trans ("t7");     probval("t7",1.0);      priority("t7",2);
trans ("t8");    rateval("t8",5000.0); trans ("t9");    rateval("t9",5000.0);
trans ("t10"); rateval("t10",5000.0); trans("t12");   rateval("t12",5000.0);
trans("t13"); rateval("t13",5000.0);
trans("t14"); rateval("t14",5000.0); trans("t15"); rateval("t15",5000.0);
trans("t16");  rateval("t16",0.064); trans("t17");  rateval("t17",0.064);
trans ("t18");    probval("t18",1.0); priority("t18",3);
trans ("t19");    probval("t19",1.0); priority("t19",2);

iarc ("t1","p4");        oarc("t1","p2");      harc("t1","p7"); harc("t1","p9");
iarc ("t2","p5");        oarc("t2","p3");
harc ("t2","p11");       vharc ("t2","p7",myval);
iarc ("t3","p1");        oarc("t3","p4")       iarc ("t3","p2"); oarc("t3","p6");
mharc("t3","p6",Ω1); oarc("t3","p1");        harc ("t3","p7");
iarc ("t4","p1");        oarc("t4","p5");      harc ("t4","p2"); oarc("t4","p7");
oarc("t4","p1");         iarc("t4","p3");      vharc("t4","p7",myval);
iarc("t5","p1");         iarc ("t5","p3");     oarc("t5","p7"); oarc("t5","p1");
miarc("t5","p6",Ω1); moarc("t5","p6",Ω1); vharc("t5","p7",myval);
iarc ("t6","p6");        harc ("t6","p1");     iarc ("t7","p7"); harc ("t7","p1");
iarc ("t8","p1");        oarc("t8","p21");     harc ("t8","p2"); harc ("t8","p3");
iarc ("t9","p1");        oarc("t9","p21");     harc ("t9","p3");
miarc ("t9","p6",Ω1);
iarc("t10","p1");        oarc("t10","p21");    viarc ("t10","p7",myval);
iarc("t12","p10");       oarc("t12","p9");     harc("t12","p2");
iarc("t13","p9");        oarc("t13","p10");    harc("t13","p1");
```

```
iarc("t14","p12");    oarc("t14","p11");    harc("t14","p3");
iarc("t15","p11");    oarc("t15","p12");    harc("t15","p1");
iarc("t16","p9");     iarc("t16","p4");
oarc("t16","p9");     oarc("t16","p13");
iarc("t17","p11");    iarc("t17","p1");
oarc("t17","p11");    oarc("t17","p14");
iarc("t18","p13");    harc("t18","p1");     oarc("t18","p2");
iarc("t19","p14");    harc("t19","p1");     oarc("t19","p3");

place ("p21");
place ("p22"); init("p22",6);
place ("p23"); init("p23",1);
place ("p24"); place ("p25");
place ("p26"); place ("p27");
place ("p29"); place ("p30"); init("p30",1); place ("p31");
place ("p32"); init("p32",1); place ("p33"); place ("p34");

trans ("t21"); rateval("t21",0.064); trans ("t22"); rateval("t22",0.064);
trans ("t23"); rateval("t23",1.0);   trans ("t24"); rateval("t24",1.0);
trans ("t25"); rateval("t25",1.0);
trans ("t26"); probval("t26",1.0);   priority("t26",3);
trans ("t27"); probval("t27",1.0);   priority("t27",2);
trans ("t28"); rateval("t28",5000.0);
trans ("t29"); rateval("t29",5000.0);
trans ("t30"); rateval("t30",5000.0);
trans("t32");  rateval("t32",5000.0);
trans("t33");  rateval("t3 3",5000.0);
trans("t34");  rateval("t34",5000.0);
trans("t35");  rateval("t3 5",5000.0);
trans("t36");  rateval("t36",0.064); trans("t37");  rateval("t37",0.064);
```

```
trans ("t38");     probval("t38",1.0);  priority("t38",3);
trans ("t39");     probval("t39",1.0);  priority("t39",2);

/* Place p40 and transition t40 represent the remaining N-2 stations of the network */
place ("p40");
trans ("t40");  rateval("t40",0.5);

iarc ("t21","p24");     oarc("t21","p22");     harc("t21","p27");
harc("t21"," p29");
iarc ("t22","p25");     oarc("t22","p23");     harc ("t22","p31");
vharc ("t22","p27",myval2);
iarc ("t23","p21");     oarc("t23","p24");     iarc ("t23","p22");
oarc("t23","p26");
oarc("t23","p21");     harc ("t23","p27");     mharc("t23","p26",Ω2);
iarc ("t24","p21");     oarc("t24","p25");     harc ("t24","p22");
oarc("t24","p27");
oarc("t24","p21");     iarc("t24","p23");     vharc("t24","p27",myval2) ;
iarc("t25","p21");     iarc ("t25","p23");     oarc("t25","p27");
oarc("t25","p21");
vharc("t25","p27",myval2);   miarc("t25","p26",Ω2);
moarc("t25","p26",Ω2);
iarc ("t26","p26");     harc ("t26","p21");
iarc ("t27","p27");     harc ("t27","p21");
iarc ("t28","p21");     oarc("t28","p40");     harc ("t28","p22");
harc ("t28","p23");
iarc ("t29","p21");     oarc("t29","p40");
harc ("t29","p23");     miarc ("t29","p26",Ω2);
iarc("t30","p21");     oarc("t30","p40");     viarc ("t30","p27",myval2);
iarc("t32","p30");     oarc("t32","p29");     harc("t32","p22");
```

```
        iarc("t33","p29");      oarc("t33","p30");      harc("t33","p21");
        iarc("t34","p32");      oarc("t34","p31");      harc("t34","p23");
        iarc("t35","p31");      oarc("t35","p32");      harc("t35","p21");
        iarc("t36","p29");      iarc("t36","p24");
        oarc("t36","p29");      oarc("t36","p33");
        iarc("t37","p31");      iarc("t37","p21");
        oarc("t37","p31");      oarc("t37","p34");
        iarc("t38","p33");      harc("t38","p21");      oarc("t38","p22");
        iarc("t39","p34");      harc("t39","p21");      oarc("t39","p23");
        iarc("t40","p40");      oarc("t40","p1");       }

        assert() {return (RES_NOERR);}
        ac_init() {}
        ac_reach(){fprintf(stderr,"/nThe reachability graph has been generated/n/n");}
/* Define output functions*/
/* Voice throughput of station 1  */   reward_type ef0() {return (rate("t3"));}
/* Data throughput of station 1  */    reward_type ef1 () {return(rate("t4") + rate("t5"));}
/* Station 1  Utilization  */          reward_type ef2 () { return ( mark("p1"));}
/* Voice throughput of station 2 */    reward_type ef3 () {return(rate("t23"));}
/*Data throughput of station 2*/ reward_type ef4 () {return(rate("t24") + rate("t25"));}
/* Station 2 Utilization */      reward_type ef5 () { return ( mark("p21"));}
/* Output */
        ac_final() {
        pr_expected("voice throughput of station 1 = ",ef0);
        pr_expected("data throughput of station 1 = ",ef1);
        pr_expected("station 1  utilization = ",ef2);
        pr_expected("voice throughput of station 2 = ",ef3);
        pr_expected("data throughput of station 2 = ",ef4);
        pr_expected("station 2 utilization = ",ef5); pr_std_average ();
                }
```

APPENDIX D

SPNP Analysis Data

Table D-1. Voice Sources vs. Throughput (DS = 0).

Vs	Thr
0	0.000
1	0.642
2	0.707
3	0.765
4	0.791
5	0.807
6	0.818
7	0.840
8	0.858
9	0.893
10	0.904

Table D-2. Effect Of (N-1) Stations on Throughput.

# of Voice Sources	Rate "t11" = 10	Rate "t11" = 5	Rate "t11" = 2
Vs	Thr	Thr	Thr
0	0.000	0.000	0.000
1	0.642	0.576	0.412
2	0.707	0.647	0.485
3	0.765	0.712	0.559
4	0.791	0.742	0.597
5	0.807	0.761	0.620
6	0.818	0.774	0.673
7	0.840	0.800	0.693
8	0.858	0.821	0.702
9	0.893	0.864	0.766
10	0.904	0.878	0.786

Table D-3. Effect Of Ds and Vs on each other's throughput.

Vs	Ds	V Thr	D Thr
0	9	0.000	0.788
1	9	0.052	0.708
2	9	0.136	0.634
3	9	0.678	0.044
4	9	0.769	0.005
5	9	0.789	0.002
6	9	0.798	0.001
7	9	0.801	0.000
8	9	0.804	0.000

Table D-4. Effect of (N-1) stations on Ds and Vs throughput.

Vs	Ds	V Thr t11=50	D Thr t11=50	V Thr t11=10	D Thr t11=10
0	9	0.000	0.788	0.000	0.884
1	9	0.052	0.708	0.016	0.821
2	9	0.136	0.634	0.056	0.763
3	9	0.678	0.044	0.100	0.731
4	9	0.769	0.005	0.150	0.693
5	9	0.789	0.002	0.203	0.653
6	9	0.798	0.001	0.280	0.580
7	9	0.801	0.000	0.340	0.528
8	9	0.804	0.000	0.400	0.477

Table D-5. Effect of Ω on Voice Throughput for identical-station network.

Ω	Voice Throughput K=8	Voice Throughput K=9
1	0.454	0.400
2	0.521	0.630
3	0.529	0.840
4	0.538	0.910
5	0.480	0.920
6	0.475	0.790
7	0.431	0.780

Table D-6. Effect of Ω on Voice Throughput for N non-identical-station network (Ω is identical for all stations).

Ω	Voice Throughput
2	0.382
3	0.391
4	0.411
5	0.403

Table D-7. Effect of W on Voice Throughput for N non-identical-station network (W varies between stations).

Ω1, Ω2	Voice Throughput	Voice Throughput
2,2	0.382	
3,3	0.391	
3,4		0.415
4,4	0.411	
5,5	0.403	

REFERENCES

[Agerwala 74] Agerwala, T. (1974). "A complete model for representing the coordination of asynchronous processes," Baltimore, John Hopkins University, Computer Science program, Res. Rep. No. 32.

[ANSI 87] *American Nasional Standard for Information Systems Fiber Distributed Data Interface (FDDI) - Token Ring Media Access Control (MAC)*, ANSI x3.139-1987, American National Standards Inst., New York, 1987.

[Burr 86] Burr, W.E. (1986). "The FDDI Optical Data Link," *IEEE Commun. magazine*, Vol.25, pp.18-23.

[Ciardo 89] Ciardo, G. (1989). *Manual for the SPNP Package*, Duke University, February 1989.

[Cooper 69] Cooper, R. B. (1969). "Queues served in cyclic order," *IEEE Commun. magazine*, Vol. 48, pp. 675-689.

[DiCesare 91] Dicesare, F. and A. A. Desrochers (1991). "Modeling, control, and performance analysis of automated manufacturing systems using Petri nets," in *Control and Dynamic Systems*, C. T. Leondes (ed), Vol.47, Academic Press, pp.121-172.

[Ferguson 85] Ferguson, M. J. (1985). "Exact results for non-symmetric Token Ring systems," *IEEE Computer transactions*, Vol. 33, pp. 223-231.

[Garcia 89] Leon-Garcia, A. (1989). "Probability and Random Processes for Electrical Engineering," *Addison Wesley Publications*, 1st Edition, pp. 504-508.

[Ghan 91] Ghani, S. and Schwartz, M. (1991). "Comparison of DQDB and FDDI MAC Access Protocols," *Proc. 16th Conf. on Local Computer Networks*, Minneapolis, pp. 84-95.

[Johnson 87] Johnson, M. J. (1987). "Proof That Timing Requirements of the FDDI Token Ring Protocol are Satisfied," *IEEE Commun. magazine*, Vol.35, pp.620-625.

[Karvelas 88] Karvelas, D. and A. Leon-Garcia (1988). "Performance Analysis of the Medium Access Control Protocol of the FDDI Token Ring Network," *IEEE Globecom'88*, Hollywood, Florida.

[Karvelas 90] Karvelas, D. and A. Leon-Garcia (1990). "Delay Analysis of Timed Token Protocol and Its Application to Hybrid Switching System, *IEEE Globecom'90*, San Diego.

[Kaye 72] Kaye, A. R. (1972). "Analysis of a distributed control loop for data transmission," *Proc. Comp. Comm. Networks and Teletraffic*, Poly. Inst. Brooklyn, pp. 47-58.

[Ma 92] Ma, J. and M. C. Zhou (1992). "Performance evaluation of discrete event systems via stepwise reduction and approximation of stochastic Petri nets", To appear in *Proc.31st IEEE Int. Conf. on Decision and Control,* Tucson, AZ.

[Martinez 86] Martinez, J., H. Alla, and M. Silva (1996). "Petri nets for the specifications of FMSs," in *Modeling and Design of Flexible Manufacturing Systems*, A.Kusiak (ed.), Elsevier Science Publishers, Amsterdam, pp. 389-406.

[Murata 89] Murata, T. (1986). "Petri nets: properties, analysis and application," *Proc. of the IEEE*, Vol.77(4), pp.541-579.

[Nair 85] Nair, S. S. (1985). "Altering priority queues with non-zero switch rule," *Computer and Operation Research*, Vol. 3, pp. 337-346.

[Ross 86] Ross, F. E. (1986). "FDDI - A Tutorial," *IEEE Commun. magazine*, col 24, pp.10-15.

[Ross 87] Ross, F. E. (1987). "Rings are 'Round for Good!," *IEEE Network magazine*, Vol.1, pp.31-38.

[Sanker 89] Sanker, R. and Y. Y. Yang (1989). "Performance Analysis of FDDI," *IEEE 14th Annual Conference on Local Computer Networks.*

[Zhou 89a] Zhou, M. C. and F. DiCesare (1989). "Adaptive design of Petri net controllers for error recovery in automated manufacturing systems," *IEEE Trans.on Systems, Man, and Cybernetics*, Vol.19(5), pp. 963-973.

[Zhou 92] Zhou, M. C. (1992). "A Petri net method for modeling and performance of token bus Local Area Networks," *Regional Control Conference*, Brooklyn, New York.

[Zhou 94] Zhou, M. C. and F. DiCesare (1994). "*Petri Net Synthesis for Discrete Event Control of Manufacturing Systems",* Kluwer Academic Publishers.