DISTRIBUTED CONTROL OF COMPUTER COMMUNICATION SYSTEMS

BY

GOPALAKRISHNAN RAMAMURTHY

A thesis submitted for the degree of

Doctor of Philosophy

of the

University of Aston in Birmingham

Department of Electrical and Electronic Engineering

September 1983

THE UNIVERSITY OF ASTON IN BIRMINGHAM

DISTRIBUTED CONTROL OF COMPUTER COMMUNICATION SYSTEMS

GOPALAKRISHNAN RAMAMURTHY

A thesis submitted to
The University of Aston in Birmingham
for the degree of
Doctor of Philosophy 1983

SUMMARY

Flow control in Computer Communication systems is generally a multi-layered structure, consisting of several mechanisms operating independently at different levels. Evaluation of the performance of networks in which different flow control mechanisms act simultaneously is an important area of research, and is examined in depth in this thesis.

This thesis presents the modelling of a finite resource computer communication network equipped with three levels of flow control, based on closed queueing network theory. The flow control mechanisms considered are: end-to-end control of virtual circuits, network access control of external messages at the entry nodes and the hop level control between nodes. The model is solved by a heuristic technique, based on an equivalent reduced network and the heuristic extensions to the mean value analysis algorithm. The method has significant computational advantages, and overcomes the limitations of the exact methods. It can be used to solve large network models with finite buffers and many virtual circuits. The model and its heuristic solution are validated by simulation. The interaction between the three levels of flow control are investigated.

A queueing model is developed for the admission delay on virtual circuits with end-to-end control, in which messages arrive from independent Poisson sources. The selection of optimum window limit is considered. Several advanced network access schemes are postulated to improve the network performance as well as that of selected traffic streams, and numerical results are presented.

A model for the dynamic control of input traffic is developed. Based on Markov decision theory, an optimal control policy is formulated. Numerical results are given and throughput-delay performance is shown to be better with dynamic control than with static control.

KEY WORDS

COMPUTER COMMUNICATION, FLOW CONTROL, HEURISTIC SOLUTIONS, PERFORMANCE EVALUATION, QUEUEING NETWORKS

ACKNOWLEDGEMENTS

The author wishes to express his sincere thanks to his two supervisors, Dr. D.J.Holding and Dr.M.H.Ackroyd, for their guidance, many useful suggestions, constructive criticisms and encouragement.

Thanks are also due to Mr.D.Bear for his useful comments on priority scheduling, and Dr. G.F.Carpenter for his help with the Unix word processor.

The author is also grateful to the Association of Commonwealth Universities for the financial support during this research.

Finally, the author owes a great debt of gratitude to his wife Hema and daughter Meera, for their love, patience and support during the period of this research.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER ONE

INTRODUCTION

1. INTRODUCTION

1.1. DISTRIBUTED COMPUTER COMMUNICATION SYSTEMS

The field of distributed computer communication systems has evolved rapidly and has involved many technological innovations in recent years. The term distributed computer communication applies to a variety of user to computer and computer to computer communication systems, where geographically seperated computing resources are connected by a network of data links. The computing resources include the processing power of (host) computers and the programme libraries and data bases attached to these computers. The network of data links and switches that interconnect these resources, forms the communication sub-network or the computer network. Messages in the form of commands, files, inquiries etc, travel through the communication sub-network, over these data links. Key applications of distributed computer communication systems are in distributed computing, distributed data base systems, distributed process communication and in teleprocessing applications.

Transmission and switch capacity are a major cost component in computer communication systems. Data traffic is generally bursty in nature. The time between arrival of messages and the message lengths are random variables. Consequently, the amount of resources demanded by the messages can be viewed as being stochastic in nature. Fixed resource allocation schemes that assign enough network resources to

meet the peak demand of each subscriber, result in low resource utilisation. Message and packet switching systems have been developed to overcome this problem, where network resources are statistically multiplexed between subscribers [18,27,42,81,88]. Packet switching is basically the same as message switching except that the messages are decomposed into smaller pieces called packets, each of which has a maximum length. The packets are numbered and addressed, and make their way through the network in a store-and-forward fashion (as with message switching). Many packets of the same message can be in transmission simultaneously. This pipelining effect results in considerable reduction of transmission delay (compared to message switching).

A computer communication network can be viewed as a collection of limited resources, shared dynamically by a population of competing users or processes. The user population includes the collection of host computers and terminals that use the sub-network to communicate with each other. The network resources includes the switching processor time, buffers, transmission bandwidth, logical channels, etc. All message and packet switched systems are designed to share this limited pool of resources and at the same time, meet the simultaneous peak demands of all users. Though such simultaneous demands are rare, they can cause conflicts to occur among the users of the system, resulting in the degradation of system performance and eventual deadlock [33]. This behaviour is typical of many contention

systems, where partial allocation of resources results in a circular wait condition, and none of the users have their resource needs satisfied.

1.2. CONGESTION CONTROL IN COMPUTER COMMUNICATION NETWORKS

A network cannot accept all the traffic offered to it. Throughput degradation and deadlock occur if the traffic admitted into the network (i.e., traffic that has already been allocated network resources) exceeds the nominal capacity of the network. Rules must be established to govern the acceptance of traffic from outside the network, and coordinate their flow inside the network. These rules known as congestion control, determine the sharing of network resources to meet user demands, and restructure the user demands to a level which can be effectively handled by the network. Congestion control can be divided into two broad areas, namely flow control and routing. Flow control regulates the admittance of new traffic such that the network is not overloaded [21,63]. More precisely, flow control is a set of mechanisms that limit the traffic within the network to a level that is compatible with the amount of available resources. Routing determines the best way of directing admitted traffic to its destination, such that the network resources are optimally utilised [84]. Internal congestion may be relieved by re-routing some of the traffic from heavily loaded paths to under utilised paths. However, if the admitted traffic is greater than the nominal carrying

capacity of the network, routing can only delay the onset of congestion but cannot prevent it. Unless proper flow control is enacted, the network can become overloaded leading to throughput degradation and eventual deadlock. Flow control protocols are designed to protect the network from problems related to overloads, transit bottlenecks and mismatched sinks.

The main functions of flow control mechanisms are [21]:

1] Prevention of throughput degradation and loss of efficiency due to overload;

2] Deadlock avoidance;

3] Fair allocation of resources among competing users;

4] Speed matching between the network and its users.

In a properly flow controlled network, the transit delay of admitted traffic is minimised at the expense of admittance delay to the network.

Flow control procedures are generaly equipped with throttling mechanisms to control the traffic [63]. The most commonly used protocols are based on tokens and are known as window flow control. New traffic is rejected when the number of unacknowledged messages in a region of the network exceeds a certain limit. The upper bound of unacknowledged messages is known as the window limit. Depending on the

region where these window limits are applied, flow control protocols are classified as:

1] Global control applied to the entire network (Isarthmic control [17,46,64]);

2] Local control applied to individual nodes (Input buffer limits [23,24,30,34,35,36,55,79,82];

3] End-to-end control applied to virtual circuits [21,62,63].

1.3. MODELLING AND PERFORMANCE EVALUATION

Considerable advances have been made in the modelling of computer communication systems for performance prediction and evaluation [42,48,49,69,74,80,81,89]. These models help to capture the main factors determining the system performance. They offer insight into the behaviour of the system and aid in system design. Models are often used when measurements on the actual system is infeasible, as in the design stages. Typically, such models are used to determine important performance measures like mean throughput, transit delay, packet loss probability, and line and buffer utilisation. Computer communication systems are most naturally represented by a network of queues. Among a number of mathematical tools pertinent to analytical modelling of computer communication systems, queueing theory plays an important role [41,42,80,81]. Queueing models and network of queues have been widely used, and they have proven to be

effective and inexpensive [8,74,89]. However, most queueing models make assumptions to simplify the model and render them tractable for numerical solution. When, in the limit, such assumptions become questionable and effect the applicability of the model, simulation is used to solve the model inspite of its relatively high cost [49,80,89]. Even when simulation is used, an analytical model can serve as a guideline in narrowing down the range of system configuration and parameters.

Queuing networks require the message arrival process and their service demands to be characterised. The Poisson processes are a good characterisation of message arrivals, when they originate from a large number of sources [2]. If the service demands made by messages at each queue (i.e., their service time) is selected from a negative exponential distribution, the resulting queueing network falls under the category of queueing networks with separable queues and have a product form solution [4,31,41,54,68,69,88]. Exponential networks with fixed population, known as closed networks also have a product form solution [4,25]. Product form networks belong to a restricted class of queueing networks that have a closed form solution. In computer network application, messages preserve their length as they are transmitted from node to node. Since the service time of a message depends on its length, the service times of the same message at different nodes are also statistically dependent. Product form network however assume they are independent (for

the queues to be separable). Kleinrock [40] recognised this defect and introduced the message independence assumption to remove this dependency. The assumption states that each time a message enters a node, a new service time is selected from a negative exponential distribution.

Closed queueing networks have algorithmic solutions only [5,9,57,67,68,69,70,71,72]. The computational effort involved in the solution of closed networks with many chains is large. The mean value analysis leads to heuristic methods which are computationally very efficient [10,70,71,87]. Littles's formula [58], which relates the mean throughput and the mean number of messages in the system to the mean transit time through the system also finds many application in modelling computer systems. In the case of computer networks, many phenomena of interest like blocking, non-exponential distribution, simultaneous resource possession etc., violate product form condition. Such networks are solved by approximate methods, generally based on the heuristic extensions of the properties of product form networks [7,8,47,51,59]. Closed queueing networks are generally approximated by the method of aggregation [6,8,80], and the heuristic extensions to the mean value analysis algorithm [10,70,71,87]. Review of queueing models for computer communication networks can be found in [74,89,92].

1.4. MODELLING AND ANALYSIS OF FLOW CONTROL TECHNIQUES

A major difficulty in the analysis of computer network

models with flow control, is the complexity involved in modelling a relatively large network. Modelling of store-and-forward computer networks involve queueing network models, which have general solutions only in special cases. The addition of flow control mechanisms lead to queueing models with large number of states and complex state dependencies. For this reason most analysis have tended to focus on three classes of simple models, namely:

1] A single node in a large homogeneous network;

2] A virtual circuit or tandem link model representing the path followed by a message constrained to a fixed route;

3] A small network of nodes connected in an arbitrary topology.

Most analysis assume Poisson arrivals and exponential packet lengths, along with the message independence assumption.

In any real network, the number of buffers at any node is limited. This can lead to blocking and dropping of packets. The retransmission of dropped packets can lead to congestion and deadlocks. Efficient buffer management techniques in the form of hop level and network access control are essential to protect the network. Kaumon and Kleinrock [34], Rich and Schwartz [75] and Ireland [30] studied buffer management techniques at an isolated node, where messages are classified according to the output link to which they

are routed. Exact analysis of a network of nodes with finite buffers, is infeasible for all but the smallest models [50]. This is because blocking introduces complex state dependencies between the queues making them inseparable: thus violating the assumptions which led to product form sclution. Pennotti and Schwartz [62], and Rudin [77] proposed an approximate model for tandem queueing networks with finite buffers. Their approximations were found by simulation to give satisfactory results. Scheweitzer and Lam [86] studied buffer overflow in a store-and-forward node using a similar approximation. Lam [53] extended this model to a multinode network and developed an approximate queueing model for an open network with finite buffers, based on flow balancing arguments. This model, which included retransmission between nodes, was solved by an iterative procedure.

Raubold and Haenle [66] proposed the structured buffer pool strategy for preventing direct and indirect store-and-forward deadlocks. This protocol which is implemented in the GMDNET was verified by extensive simulation [23,24]. Merlin [61] proved its deadlock avoidiance property using buffer graphs. A simple version of this scheme is the Input Buffer Limit protocol (IBL for short), where transit messages are given preference over external messages by limiting the number of buffers available for external messages. The improvement in network performance was first noticed in the simulation studies by Price [64]. A similar idea was also suggested by Chou and Gerla [12]. Lam and Reiser [55]

analysed this network access control based on input buffer limits for a symmetrical network model. They proposed a heuristic choice for selecting the number of buffers available for external traffic at a node. Lam and Lein [56] verified this by simulation for nonhomogeneous networks. Saad and Schwartz [79,82] extended this model further by imposing additional constraints. Kaumon [36] proposed another version of the IBL control called the Drop and Throttle Flow control. He extended this model further by imposing a maximum limit on the number of buffers allocated to an output queue [35]. These two models were applied to a symmetric network, where blocked messages were dropped from the network.

A choke packet scheme, where a queue sends a choke packet to the source node, when its server utilisation exceed a preset value has been implemented in the Cigale network [26]. Chu et al. [14] studied a two node model where the input traffic at the first node is controlled, based on the queue length or channel intensity at the second node. Jaffee [32] studied a bottleneck flow control mechanism that has many features in common with the choke packet scheme. Matsumoto and Mori [60] studied a flow control method based on gradual restriction of packet flow in virtual calls. These models ignore deadlocks and hence quantitatively accurate results can be expected only for small blocking probabilities. Kaufman et al. [37], studied a two node model with finite buffers, which also considered the

strong coupling between the nodes and the possibility of store-and-forward deadlocks. They showed the utility of applying sparse matrix methods in flow control problems.

The main objective of end-to-end flow control of virtual circuits is to prevent congestion at the exit node. An important byproduct of this protocol is the prevention of global congestion. Examples of the end-to-end flow control are the RFNM in the ARPANET [42], and the Pacing control in the SNA network [3]. Virtually all end-to-end flow controls are based on a window mechanism, that limits the total number of messages in transit on any virtual circuit. Pennotti and Schwartz [62] studied the end-to-end control of a single virtual circuit in a tandem queueing network. Chatterjee et al. [11] extended the results for the case of a logical link with random routing. Kleinrock and Kermani [45] modelled a single source-destination stream, where the round trip acknowledgement delay was assumed to have an Erlang-2 distribution. Schwartz [83,85] investigated Pacing control in the SNA network and compared it with two other window control mechanisms. Kleinrock [43] found the optimum window limit by optimising Power [44], defined as the ratio of virtual circuit throughput to delay. For optimum power, the window limit is equal to the number of hops covered. The power measure was also used by Kumar [52], Schwartz [85] and Reiser [73] to optimise network performance. Reiser [73] also modelled the admission queueing delay of virtual circuits, when messages originate from a finite set of

terminals.

Davies [17] and Price [64] introduced the global network control called the Isarthmic control, based on the circulation of a fixed number of permits. Schwartz and Saad [82] modelled isarthmic control in a three node network. Kleinrock and Tseng [46] proposed flow control based on limiting the permit generation rate. Here, permits are generated at each entry node and subsequently destroyed when messages are delivered to their destination hosts. The total number of messages admitted to the network is limited by controlling the permit generation rate. The end-to-end flow control models mentioned so far offer some insight into multiuser flow control, but suffer from the limitation that only one virtual circuit can be flow controlled at a time, while the remaining traffic components are constant. To overcome this limitation, several multiple source multiple destination models with selectively controlled user pairs, have been developed. Wong and Unsoy, [90] studied a five node model with isarthmic control and end-to-end control. Pujolle [65] developed a queueing model for a network with three levels of flow control namely, user to user, host to host and node to node. Because of the complexity, he used simulation in conjunction with analysis. Georganas [20] studied the performance of two virtual circuits in a four node network with finite buffers, equipped with global (isarthmic) and end-to-end control. These closed queueing network models were solved using the convolution algorithm.

However, they have only a limited application, because of the computational complexity involved in using the convolution algorithm. The exact analysis of multinode network with several individually controlled virtual circuits, becomes impractical for networks with four or more virtual circuits. To circumvent this problem, Reiser and Lavenberg proposed an approximate solution based on the mean value analysis algorithm and its heuristic extensions [70,71]. The heuristic extensions to the mean value analysis algorithm does not give the marginal queue length probabilities and hence the model is only applicable to networks without nodal blocking.

Traditionally, routing and flow control have been developed independently, under the assumption that, while flow control keeps excess traffic out of the network, routing tries to transport the accepted traffic efficiently across the network. Gerla and Nilsson [22], studied the interaction between routing and flow control, and developed an approximate iterative solution based on the mean value analysis algorithm and the flow deviation optimisation. Chu and Shen [13] used a simulation model to study a hierarchical routing and flow control policy. Based on a channel threshold, alternate routes were used. When all alternate routes became unavailable, input traffic was temporarily rejected. Rudin and Muller [78] investigated the interaction between dynamic routing and end-to-end flow control using a simulation model

Lastly Markov decision theory has been used to model dynamic flow control mechanisms. Kermani and Kleinrock [38], proposed an adaptive policy for dynamic adjustment of the window size to time-varying traffic. Kermani [39] also analysed a local control scheme called the feed back control scheme using Markov decision theory. Here, each node in the network controls the rate of traffic it receives from its neighbouring nodes.

1.5. OUTLINE OF PRESENT RESEARCH

Flow control in computer communication network is generally a multilayered structure, consisting of several mechanisms operating independently at different levels. The existence of such multiple levels of control and the possible integration of some of these mechanisms into hybrid arrangements brings an important issue that requires further study, namely the interaction between the different levels. While the performance of individual flow control mechanisms are known, a study of their combined effect when they are operating simultaneously in the network has to be made. These issues can only be investigated by developing models which include multiple levels of flow control. Such models are also essential for the performance evaluation of networks equipped with multiple levels of flow control. Further, computationally efficient methods are required to solve these models, so that realistic topologies with a large number of nodes and virtual circuits can be modelled.

In chapter 2, a queueing network model for a store-and-forward computer communication network with finite buffers and equipped with three levels of flow control is developed. The flow control mechanisms are of the type used in most networks. The model is solved by an iterative procedure using the convolution algorithm and the mean value analysis algorithm. A recursive procedure is developed to determine the marginal queue length probabilities when there are two classes of traffic, namely external traffic and transit traffic. Important performance measures of virtual circuit throughput, mean end-to-end delay and nodal blocking probabilities are derived. The computational complexity of the exact algorithms is outlined.

Computer network models generally involve many virtual circuits. However, the computational complexity and the storage requirements of the standard algorithms grow exponentially with the number of chains. This limitation excludes their use for networks with four or more virtual circuits. A computationally efficient technique, that overcomes the severe limitations of the exact method, is developed in chapter 3. The method is a heuristic one, based on an equivalent reduced network and the heuristic extensions to the mean value analysis algorithm. The method can be used to solve large networks with finite buffers and many virtual circuits. The computational effort is reduced further by the use of the Norton's theorem for queueing networks. A study is made of a five node network and the

analytical results obtained by the heuristic methods are compared with the results obtained by the exact method.

In chapter 4, the interaction between the three levels of flow control is investigated, by modelling a five node network with finite buffers and with four virtual circuits. The model is solved analytically as well as by computer simulation. The effect of the virtual circuit window limit, the network access control parameter and the size of the buffer pool, on the performance of the network (as a function of the virtual circuit load) is investigated. Simulation of the five node network also validates the closed queueing network model and its heuristic solutions developed in chapter 2 and 3.

Computer communication networks equipped with end-to-end control of virtual circuits have been modelled as loss systems, where messages that are blocked by the flow control mechanism are lost [11,20,57,62,70,90]. In real computer networks, such loss of messages do not occur and messages wait outside the network until they are admitted. The loss models do not consider the effect of buffering messages outside the network, on the stochastic process by which messages arrive at the network. A queueing model for the admission queueing process, when messages arrive from independent Poisson sources is developed in chapter 5. The model provides a method for estimating the admission delay and the mean queue length of the admission queues associated

with the virtual circuits. The chapter also discusses the selection of the optimum virtual circuit window limit to optimise the network performance (including the admission delay).

Several advanced network access schemes are postulated in chapter 6. These access schemes are divided into two categories. The first category improves the performance of the network by more efficient use of the virtual circuit tokens. The first method under this category, buffers any message that has secured a free token but is blocked by the network access control. The second method buffers any message, that arrives at the network but fails to secure a token. The two methods ensure that the tokens do not remain idle and hence improve their utilisation. The second category considers the selective improvement in throughput of certain virtual circuits, at the expense of other virtual circuits. In the first method under this category, priority messages that fail to secure tokens are allowed to seize those allocated to nonpriority messages. A second method uses a local access scheme to control the sharing of virtual route tokens between two sources. A third method considers priority virtual circuits that are exempted from the network access control. All these schemes are modelled and solved using the heuristic method developed in chapters 3 and 5. Numerical examples for a five node network equipped with these control mechanisms are presented.

Using the theory of Markov decision process, an optimal policy for the dynamic control of external messages is formulated in chapter 7. The admittance of external messages are dynamically determined, based on the state of the node and the transit traffic load. The dynamic control throttles the external traffic when the transit traffic is high, thereby ensuring that the nodal resources are made available to transit traffic alone. At the same time, the buffer utilisation is maximised (and hence the nodal throughput) by admitting more external messages when the transit traffic is low. Priority scheduling of transit messages is also incorporated in the dynamic model as a means of reducing their queueing delay. Numerical results are presented and the throughput-delay performance is shown to be better with dynamic control, than with static control. The formulation of the model is general. Application of the model to other flow control problems are outlined.

Chapter 8 provides a summary of the present work and considers some directions for further research.

CHAPTER TWO

QUEUEING NETWORK ANALYSIS OF COMPUTER COMMUNICATION NETWORKS WITH MULTI-LEVEL FLOW CONTROL

2. QUEUEING NETWORK ANALYSIS OF COMPUTER COMMUNICATION NETWORKS WITH MULTI-LEVEL FLOW CONTROL

2.1. Introduction

Computer communication networks are generally equipped with several flow control mechanisms, operating independently at different levels. Multiple levels of flow control are implemented since no single flow control mechanism can cope with all the problems in the network. The presence of multiple levels of flow control and the possible integration of some of them into hybrid arrangements raises the problem of interaction between the different levels, when they operate simultaneously in a network. These issues can only be investigated by developing models which include multiple levels of flow control. Such models can also be used in the performance evaluation of networks with multiple levels of flow control.

In this chapter a queueing model is developed for store-and-forward computer communication networks with finite buffer resources, and which operate with three levels of flow control. The flow control mechanisms considered are:

1] End-to-end control of individual virtual circuits;

2] Network access control based on the input buffer limit scheme (IBL control), that favours transit traffic over external or input traffic;

3] Hop level control based on complete sharing of buffers, and retransmission between nodes to prevent message loss due to nodal blocking.

The chapter begins with the description of the network and the flow control mechanism associated with it. The network is modelled as a closed queueing network, and is solved using the convolution algorithm [67,68,69] and the mean value analysis algorithm [70,71]. A recursive procedure is developed to determine the marginal queue length probabilities, when there are two classes of traffic at a node (namely transit traffic and external traffic). The finite pool of buffers at each node leads to nodal blocking. Since nodal blocking introduces dependencies between the queues, the network of queues are no longer separable and hence product form solution cannot be obtained. An exact queueing analysis of systems with nodal blocking is unfeasible for all but the smallest models [50]. In order to render the model mathematically tractable, the finite length queues are approximated by infinite length queues with suitably modified service rates [52,55,62,77], and the nodal blocking probabilities are replaced by overflow probabilities. Important performance measures of virtual circuit throughput, end-to-end delay and nodal blocking probabilities are determined. The complexity and limitation of the convolution algorithm and the mean value analysis algorithm are discussed.

2.2. THE NETWORK AND ITS QUEUEING MODEL

This section describes the structure of a store-and-forward computer communication network with finite buffers at each node, and equipped with three levels of flow control. A queueing network model based on the theory of closed queueing networks with multiple closed chains is developed, and the assumptions involved in the analysis are elaborated.

2.2.1. THE NETWORK AND ITS FLOW CONTROL MECHANISMS

The store-and-forward computer communication network has M switching nodes, that are interconnected by L links or channels [20,53,57,70]. In addition to the L interconnecting links, there are M exit links (one at each node) that connect the nodes and the destination hosts to which messages are finally delivered. The links are unidirectional in nature and the transmission capacity of link i(i=1,2,...,L,L+1,...,L+M) is Ci bits per second. If link i connects node n and node m, or node n and host m, then the transmission capacity of link i is also represented as Cnm bits per second. Link level over heads like acknowledgements etc, are assumed to be factored into the link capacity. Full duplex lines are represented by two links in opposite directions.

All messages are single packet messages, and they are classified according to the virtual circuit to which they

belong. There are R unidirectional virtual circuits between designated pairs of nodes. Each virtual circuit has an entry node and an exit node. A message is said to belong to virtual circuit r (or class r, or chain r), r=(s,d), if its entry node is s and its exit node is d. Messages belonging to chain r, arrive at the source node from a Poisson source host. Their mean arrival rate is Ar messages per second. Message lengths for all classes are selected from a common exponential distribution, with a mean length of 1/a bits. Each virtual circuit or chain has a routing matrix P = {P(ij,r)}, which could be fixed or probabilistic. In the case of probabilistic routing, P(ij,r) is the probability of a chain r message currently at node i, being routed to node j next. In the case of fixed routing, the elements of the routing matrix indicate the next node to which a message has to be routed from a given node, for a given class or chain. The elements of the fixed routing matrix are either one or zero, such that,

$$P(ij,r) = \begin{cases} 1 & \text{if chain r messages visit node j next, from node i;} \\ 0 & \text{otherwise.} \end{cases}$$

The virtual circuits are individually flow controlled by an end-to-end flow control mechanism based on a token scheme. Each virtual circuit r(r=1,2,...,R), has a token or window limit Kr. The end-to-end flow control mechanism, lim-

its the total number of class r messages that can be in transit across the network to Kr messages. The messages on a virtual circuit are individually acknowledged by the destination hosts, if they are received without errors. Acknowledgements which are returned to the source host can be piggy-backed on normal messages travelling in the opposite direction, or they can be stand alone messages. If they are stand alone messages, they can have equal or higher priority than data messages. Messages are admitted to the network only if they secure a free token. The number of free virtual circuit r(r=1,2,...,R) tokens is decremented by one, whenever the network accepts a message from the chain r source host. The number of free tokens is incremented by one, when an end-to-end acknowledgement is received from the destination host. The number of unacknowledged message is also known as the instantaneous window size of the virtual circuit. The virtual circuit has a maximum window size equal to the token limit. If a class r window is not full, new messages are generated by the source host r at the rate of Ar messages per second. Messages which are generated when the window limit is full are lost, or the arrival process is halted.

Fig. 2.1 shows the typical structure of a node. Each node m(m=1,2,...,M) in the network has a nodal processor and a pool of limited buffers. The buffers at a node are completely shared by all the outgoing links at the node, including the exit link connecting the node and its hosts.

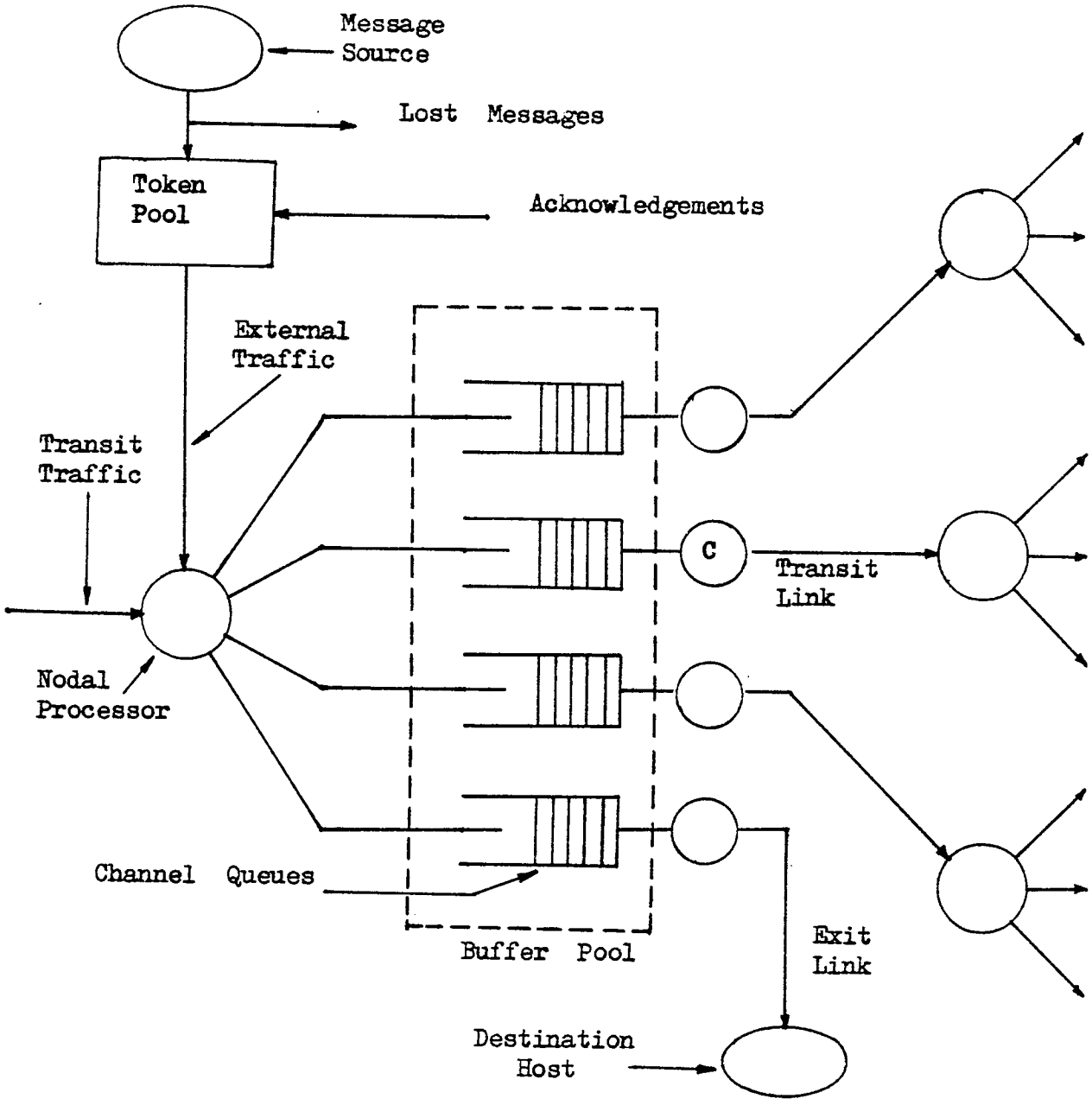


Fig. 2.1 Model of a Node.

A network access control gives preference to traffic arriving from neighbouring nodes, over external traffic entering the network at that node. The transit traffic has access to all the NTm buffers at node m, where NTm is the total number of buffers at node m. External traffic entering the network at node m, has access to only NEm buffers, where NEm is less than NTm. This prevents the external traffic from occupying all the buffers at a node and gives preference to transit traffic, to which resources have already been committed. Transit traffic arriving at node m are blocked, when all NTm buffers are occupied. This event occurs with a probability BTm. External messages arriving at node m are blocked when there are NEm external messages, already waiting for service or transmission, or when all NTm buffers are occupied. This event occurs with a probability BEm. This network access control is the second level of flow control in the network.

The nodal processor accepts any message arriving at the node, iff it is error free and a free buffer is available as laid down by the network access control mechanism. The processor directs the message to the appropriate output link queue, as dictated by the routing table. Messages leaving the network at the node, are directed to their hosts via the exit link connecting the node and the hosts. A first-in-first-out (FIFO) scheduling is used for messages waiting for transmission or service. For simplicity, the nodal processing delays which are small compared to the queueing and transmission delays, are neglected. That is, the cause of

congestion in the network is due to limited channel capacity.

External messages blocked at the entry node due to shortage of buffers are assumed to be lost. Hence, an external message is admitted to the network only when it secures a free token as well as a free buffer at the entry node. In the case of transit messages, the upstream node retransmits the messages until they are accepted by the downstream node (i.e, the next node). The successful delivery and acceptance of a message is made known to the upstream node by the receipt of an acknowledgement. This third level of flow control, which transports messages successfully across individual physical links, or between two neighbouring nodes, is known as the hop level flow control. The function of the hop level control is to retransmit transit messages that are dropped because of congestion at the receiving node. It maintains a smooth flow of traffic between the neighbouring nodes, avoiding local buffer congestion. It is assumed that the destination hosts have sufficient number of buffers.

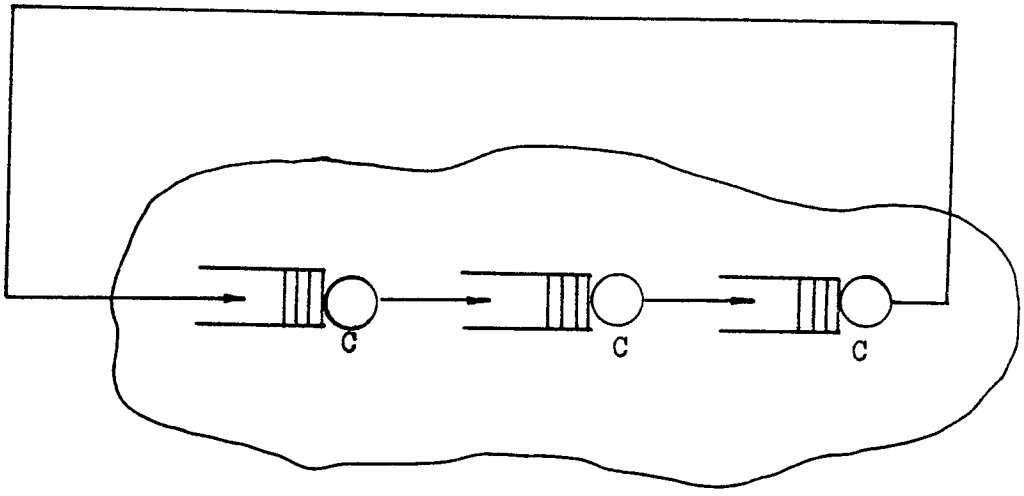
2.2.2. CLOSED QUEUEING NETWORKS

A queueing network is said to be closed if all its routing chains are closed [4,25,80]. The rth routing chain (or class r virtual circuit) is closed, if the total number of chain r messages that are in transit within the network is equal to the window limit of chain r. Since the chain population must be equal to its window limit at all times,

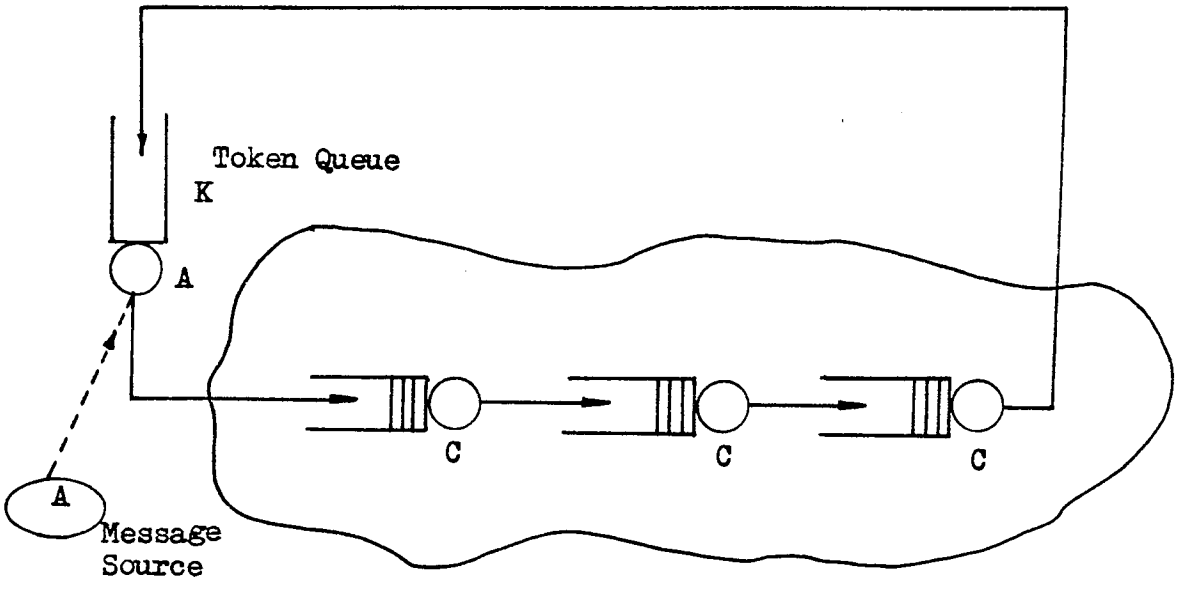
the arrival rate of chain r messages must be sufficiently high to saturate the chain. Hence, a closed network will be of interest only when saturation loads are considered.

2.2.3. THE MODIFIED CLOSED QUEUEING NETWORK

The closed queueing network can still be used to model the network at all traffic levels by suitable modification [46,57,85]. The modified network is shown in fig. 2.2, where, the R Poisson sources are replaced by R token queues. That is, the Poisson source r associated with virtual circuit r(r=1,2,...,R), is replaced by a token queue Qtr. The token queue Qtr is assumed to have a service time with a negative exponential distribution. The mean service rate of the token queue Qtr is Ar messages per second, which is also the mean arrival rate of messages from the rth Poisson source associated with virtual circuit r. The token queue is sufficiently large to hold all the Kr tokens. Tokens exist as free tokens in the token queue. If a message arrives and finds a free token in the token queue, the token is removed and the message is admitted to the network. If a message on its arrival finds the token queue empty, it is assumed to be lost. Once a message is successfully delivered to its destination host, the tokens return as acknowledgements and join the token queue again. The total number of free tokens in the token queue, and messages in transit within the network (including acknowledgements) in any virtual circuit, is constant and equal to the window



(a)



(b)

Fig. 2.2 (a) Tandem queue model of a virtual circuit - saturation model. (b) Modified tandem queue model of a virtual circuit with token queue.

limit of the virtual circuit at all times. Hence, the modified network along with its R token queues, forms a closed queueing network. Closed queueing network theory can now be applied to this modified network for all arrival rates of messages. At low arrival rates, the tokens exist as free tokens in the token queue. At high arrival rates, the token queues are empty. The tokens exist as busy tokens in the form of messages and their end-to-end acknowledgements, in transit within the network.

2.2.4. QUEUEING MODEL OF THE CLOSED QUEUEING NETWORK

The packet switched network is assumed to have M switching nodes, interconnected by L links or transmission channels. There are M outgoing links connecting the M nodes and their hosts. There are R classes of messages. A class r(r=1,2,...,R) message completing service at queue i, is next routed to queue j with a probability P(ij,r). The routing matrix P={P(ij,r)} can be considered as defining a Markov chain. The Markov chain is assumed to be decomposable into R ergodic chains (or virtual circuits in this case) [4,54]. Corresponding to the R chains or virtual circuits, there are R Poisson arrival streams. The rth Poisson stream is represented by the token queue Qtr, with an exponentially distributed service time. The mean service rate of the rth token queue is Ar. The probability that a token at the head of the token queue Qtr, has to wait for a time $t \leq ta$ before it is allocated to a message, is [1-exp(-Ar.ta)]. Each chain

is controlled by an end-to-end flow control mechanism, that limits the maximum number of chain r messages in transit within the network to Kr messages. That is, the total number of free chain r tokens and chain r messages in transit is always constant and equal to Kr. The end-to-end control vector (also known as the population vector) is represented by the vector K=K1,K2,...,KR.

The number of buffers at each node is finite and is shared completely by the outgoing links at the node. A network access control gives preference to traffic arriving from neighbouring nodes (i.e, transit traffic), over traffic entering the network at the node (i.e, external traffic). The transit traffic at node m has access to NTm buffers, while the external messages entering the network at node m have access to NEm ($<$ NTm) buffers. Transit messages arriving at node m are blocked with a probability BTm, while external messages at node m are blocked with a probability BEm ($>$ BTm). The nodal blocking probability vectors are represented by BE(BE=BE1,BE2,...,BEM) and BT(BT=BT1,BT2,...,BTM). The token queues are sufficiently large to hold all the tokens and hence, they are never blocked.

Nodal blocking due to finite number of buffers at each node introduces complex dependencies between the queues. This renders the queues in the network inseparable and makes the model mathematically intractable. To make the analysis

tractable so that the theory of closed queueing networks may be applied, the number of buffers at each node is assumed to be infinite and the transmission capacities of the links are modified to account for blocking. The transmission links are represented by FIFO unbounded queues and the blocking probabilities are replaced by overflow probabilities (i.e., the probability of queue occupancy exceeding a finite measure). However, the overflow probabilities will still be referred to as blocking probabilities [20].

The infinite or unbounded queue model makes the network nonblocking. The effect of blocking due to the finite number of buffers at each node in the actual network, is reflected on the unbounded queue model as follows. A message after completing service at the upstream service centre, makes a visit to the downstream node. At the downstream node, a decision mechanism decides whether the message should join the queue at this node (with a probability (1-p)), or should return to the upstream service centre (with a probability p). If the number of such visits are assumed to be geometrically distributed, then from renewal theory [16], the message makes 1/(1-p) visits to the downstream node before joining its queue. This increases the mean service demand placed by the message on the upstream service center by a factor 1/(1-p). The load on the downstream node is not increased by the 1/(1-p) visits, since, on p/(1-p) visits the message returns to the upstream service centre without joining the queue at the down stream node. The mechanism

slows down the progress of messages through the network, and at the same time it brings about a natural limiting of the queue length at the downstream node.

In the actual network, the decision mechanism makes its decision by observing the number of messages queueing at the downstream node. It decides that a transit message should join the queue at the downstream node with a probability (1-BT). The decision mechanism also decides that a token associated with an external message shall enter the entry node with a probability (1-BE). The token returns to the token queue with a probability BE, and waits for the arrival of a new message from the external source. However, an exact analysis in which the decision mechanism makes its decision by observing the state of the downstream node, introduces dependencies between the queues in the network. To overcome this problem, the steady state nodal blocking probabilities are assumed to be known before hand. This means that the probability p(p=BE or BT) associated with the decision mechanism is a constant and independent of the state of the down stream node. Closed queueing network theory can now be applied to this model. The model can be solved by an iterative process, where an initial approximation BT(0) and BE(0) is used to generate a sequence of approximations BT(1),BE(1);BT(2),BE(2);...;BT(k),BE(k), which converge to a solution [20,53,62,79].

In the actual network the upstream service centre is

blocked until the downstream node has a free buffer available. The Data Link Control protocol closest to this situation is the SEND AND WAIT protocol. However if the upstream server is idle (blocked) while there are messages waiting for service, the queueing system will not be work conserving [42,48,49]. Hence it is assumed that messages are continuously retransmitted 1/(1-p) times (where p is assumed to be a constant and equal to BE or BT). This increases the service time of a message by a factor 1/(1-p) or the service rate of the server is reduced by a factor (1-p).

Consider two nodes j and m interconnected by link jm, designated as link i (fig. 2.3). The channel capacity of link i is Ci or Cjm bits per second. If node m is assumed to be blocked with a probability BTm for transit messages the transmission capacity of link i is modified as,

$$C'i = Ci\ (1-BTm) \qquad i=1,2,\ldots,L,\quad m=1,2,\ldots,M.$$

The mean time a chain r token has to wait at the head of its token queue before it is removed, is 1/Ar seconds. However, because of blocking at the entry node m, the tokens are returned to the head of the token queue with a probability BEm. The mean number of attempts a token has to make, to enter the entry node m is 1/(1-BEm). Hence, the mean time spent by a token at the head of the token queue is increased by a factor 1/(1-BEm). The effective service rate of the token queue Qtr associated with chain r is,

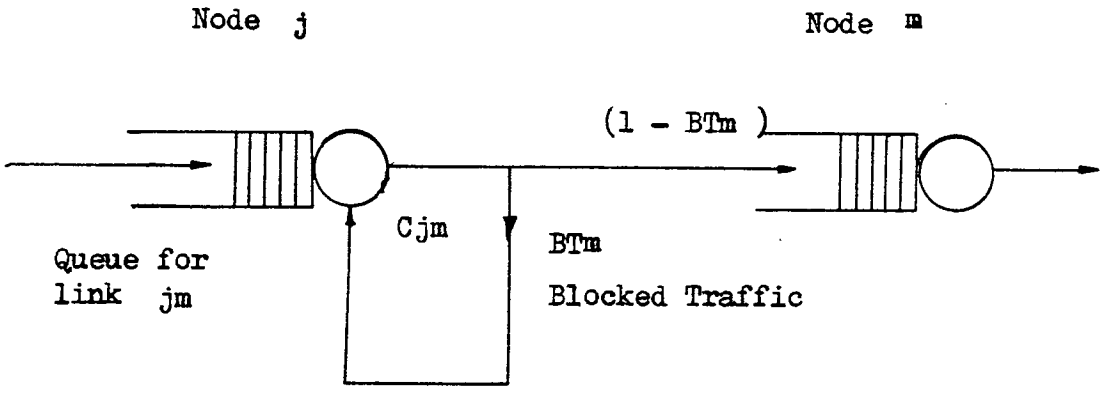


Fig. 2.3 Blocking Model.

$$C'r = Ar\ (1-BEm) \qquad r=1,2,\ldots,R.$$

Where,

Ci	Actual service rate of link i;
C'i	Modified service rate of link i;
C'r	Modified service rate of the rth token queue;
Ar	Arrival rate of messages from chain r source;
BTm	Probability of node m being blocked to transit messages;
BEm	Probability of source node m being blocked to external messages.

In the discussions to follow, the overflow probabilities will be referred to as blocking probabilities.

2.2.5. ASSUMPTIONS

The assumptions required to make the analysis possible are summarised below.

1] There are R unidirectional virtual circuits between designated pairs of nodes. Each virtual circuit has a source host and a destination host associated with it. The time between consecutive arrival of virtual circuit r(r=1,2,...,R) messages from its source, to the network, is drawn from a negative exponential distribu-

tion. The mean arrival rate of class r messages is Ar messages per second.

2] All messages are single packet messages. A message occupies only one buffer at every node it visits.

3] Messages retain their "identity" as they are transmitted via a node, where each time a message is transmitted, its length is selected from a common negative exponential distribution (common to all classes). The mean length of a message is 1/a bits. This is the independence assumption [40].

4] The nodes are connected by L unidirectional communication channels. The communication channels are modelled by FIFO unbounded queues.

5] The links are perfectly reliable and there are no transmission errors.

6] Processing delays within the network are ignored since they are smaller than channel delays.

7] The virtual circuit r, has a window limit Kr (r=1,2,...,R.). When the window limit is full, new message arrivals are lost.

8] The number of buffers at each node is limited. Messages that do not find free buffers are retransmitted by the upstream node. The nodal blocking probabilities are replaced by overflow probabilities, and the overflow

probabilities are assumed to be known.

9] The effect of nodal blocking is reflected on the infinite queue length model, by suitably modifying the transmission capacity of the links.

10] The end-to-end acknowledgement delays are ignored. This would be the case if acknowledgements are returned through separate high speed channels. However, the acknowledgement delays can be accounted for, by considering the queues traversed by the acknowledgement messages in the opposite direction. A method for accounting the acknowledgement delays is discussed in chapter 3.

2.3. ANALYTICAL SOLUTION OF THE CLOSED QUEUEING NETWORK MODEL

This section presents the analysis of the modified closed queueing network model. A recursive procedure is developed to determine the queue length probabilities when there are two classes of messages, namely external messages entering the network and transit messages. Important network performances of individual virtual circuit throughput, virtual circuit delay and the nodal blocking probabilities are determined.

2.3.1. STATES OF THE CLOSED QUEUEING NETWORK AND ITS EQUILIBRIUM STATE PROBABILITIES

Consider the general multichain closed queueing network model described in section 2.1 and 2.2, for which the following notations were defined.

M Number of nodes in the network. This is also the number of exit links in the network;

L Number of internode links in the network;

R Number of virtual circuits or chains. This is also the number of token queues in the network;

l	Total number of queues in the network;
Kr	Population of chain r, r=1,2,...,R;
K	Population Vector, K=K1,K2,...,KR;
Ar	Arrival rate of chain r messages to the network, r=1,2,...,R;
Qtr	Token queue associated with chain r;
1/a	Mean length of a message in bits;
Ci	Transmission capacity of link i in bits/sec;
C'i	Modified service rate of queue i;
NTm	Total number of buffers at node m;
NEm	Number of buffers available to external messages at node m;
BTm	Probability of node m being blocked to transit messages;
BEm	Probability of node m being blocked to external messages;
eir	Relative number of visits made by a chain r message to the service centre at queue i. This factor is determined by the routing probability matrix.

In addition, the following notations are also defined.

S	Vector representing the state of the network;
Fs	Set of feasible states of the network;
Ni	Vector representing the conditions prevailing at queue i;
Nir	Number of chain r messages at queue i;
P(S|K)	Equilibrium state probability of the network being in the state S, when the population vector is K;
C	Normalisation constant of the network;
Q(r)	Set of queues visited by chain r messages.

Let the states of the network be represented by the vector S=N1,N2,...,Nl, where Ni=(Ni1,Ni2,...,NiR) is the condition at queue i, Nir is the number of chain r messages at queue i and l(l=L+M+R) is the total number of queues in the closed queueing network model. The network state space for a given population vector K=K1,K2,...,KR is defined by the set of feasible states

$$Fs = \{ S \mid Nir \geq 0, \sum_{i \in Q(r)} Nir = Kr \quad \text{for } r=1,2,...,R; \quad K=K1,K2,...,KR; \quad i=1,2,...,l \quad \text{and} \quad l=L+M+R \ \}. \tag{2.1}$$

If Nir is the number of chain r messages at queue i

when the network is in the state S, the total number of chain r messages in the network is,

$$Nr(S) = \sum_{i=1}^{L+M} Nir(S),$$

and

Kr = Nr(S) + (number of free tokens in the rth token queue, when the network is in state S),

for r=1,2,...,R.

The total number of messages in the network is,

$$N(S) = \sum_{r=1}^{R} Nr(S) \ .$$

The network falls under the category of networks with separable queues [4,68]. The equilibrium network state probability has a product form solution given by [4],

$$P(S|K) = C\ d(S) \prod_{i=1}^{1} fi(Ni) \qquad S \in Fs, \tag{2.2}$$

where, S is an element of the state space Fs, and

P(S|K) Equilibrium state probability of the network being in the state S, given the population vector K;

C Normalisation constant chosen to make the equilibrium state probabilities sum to one;

Ni Condition prevailing at queue i;

d(S) A function of the arrival process and the number of messages in the network. This is equal to one for a closed queueing network since there are no arrivals or departures;

fi(Ni) A function that depends on the type of service centre at queue i.

For FIFO scheduling at queue i, fi(Ni) is given by [4,54],

$$fi(Ni) = \{ \sum_{r=1}^{R} Nir \}! \{ \prod_{r=1}^{R} [1/Nir!] [eir/a\ C'i]^{Nir} \}. \qquad (2.3)$$

The factors eir must satisfy the following set of linear equations for each routing chain:

$$eir = \sum_{j=1}^{l} ejr\ P(ji,r) + qir$$

$$\text{for } r=1,2,\ldots,R \text{ and } i=1,2,\ldots,l. \qquad (2.4)$$

Where, qir is the probability of a chain r message entering

the network (or departing from the network) at queue i. Since the population of a closed chain is constant, there can be no arrivals or departures and hence qir=0 for all i=1,2,...,1 and all r=1,2,...,R. In this case the factors eir are determined within a multiplicative constant for each chain. The factor eir can be interpreted as the relative number of visits made by a chain r message to the service centre at queue i (in the unbounded queue length model), during its life time in the network. It does not include retransmissions resulting from nodal blocking. In practice the factors eir are derived from the network description or from the routing probabilities. In the case of fixed routing where a message makes only one visit to each service centre along its route,

$$eir = \begin{cases} 1 & \text{if chain r messages visit queue i} \\ 0 & \text{otherwise,} \end{cases}$$

for r=1,2,...,R and i=1,2,...,1.

In the case of probabilistic routing,

$$eir = \sum_{j=1}^{1} ejr \; P(ji,r),$$

where, P(ji,r) is the probability of a chain r message at queue j, being routed to queue i next.

C'i is the effective service rate of queue i and is given by,

$$C'i = Ci\ (1-BTm), \tag{2.5}$$

where BTm is the probability of node m being blocked, and messages in queue i are routed to node m next. Similarly, the effective service rate of token queue Qtr associated with chain r is,

$$C'r = Ar\ (1-BEm). \tag{2.6}$$

Where, chain r messages enter the network at node m.

The normalisation constant C in (2.2) can be found by summing all state probabilities and equating the sum to unity. That is,

$$C^{-1} = \sum_{S \in Fs} \prod_{i=1}^{1} \{ \sum_{r=1}^{R} Nir \}! \ \{ \prod_{r=1}^{R} [1/Nir!]\ [eir/a\ C'i]^{Nir} \}. \tag{2.7}$$

Once the normalisation constant has been found, the nodal blocking probabilities can be determined. The blocking (or overflow) probability for external messages at node m is given by,

$$BEm = 1 - \sum_{S \in Fem} P(S|K) \qquad m=1,2,\ldots,M;$$

where,

$$Fem = \{\ S \in Fs \mid \sum_{r \in RE(m)} \sum_{i \in QN(m)} Nir < NEm$$

$$\text{and} \quad \sum_{r=1}^{R} \sum_{i \in QN(m)} Nir < NTm \quad \},$$

and

$$RE(m) = \{ \text{ set of chains entering the network at node m } \},$$

$$QN(m) = \{ \text{ set of queues at node m } \}. \tag{2.8}$$

The blocking (or overflow) probability for transit messages at node m is given by,

$$BTm = 1 - \sum_{S \in Ftm} P(S|K) \qquad m=1,2,\ldots,M;$$

where

$$Ftm = \{\ S \in Fs \mid \sum_{r=1}^{R} \sum_{i \in QN(m)} Nir < NT \ \}$$

and

$$QN(m) = \{\text{set of queues at node m}\}. \tag{2.9}$$

Equations (2.1) to (2.9) define a system of non-linear algebraic equations from which, the normalisation constant C and the blocking probability vectors BE and BT can be found. The system of equations can be solved by an iterative pro-

cedure. To begin with, an initial value for the blocking probability vectors BT and BE is assumed. The link capacities are suitably modified (equations 2.5 and 2.6), and the normalisation constant C is evaluated (2.7). Using this value of C, a better approximation for the blocking probability vectors are found (2.8 and 2.9). The process is continued until the blocking probability vectors converge to a solution. The process is terminated on the kth iteration if the convergence criterion

$$\{ \sum_{i=1}^{M} [Bi(k) - Bi(k-1)]^2 \}^{1/2} \leq \xi \,, \quad \text{for } \xi << 1 \tag{2.10}$$

is satisfied. Once the normalisation constant and the blocking probabilities have been found, all the performance measures can be evaluated.

2.3.2. THE NORMALISATION CONSTANT C

The normalisation constant C=C(K) of the closed queueing network with R chains and population vector K=K1,K2,...,KR can be determined by the convolution algorithm [57,67,68,69]. Consider the closed queueing network model with 1(1=L+M+R) queues and R closed chains, which are conditioned to contain Kr messages (r=1,2,...,R).The following notations are defined in addition to those defined in section 2.3.1.

G(K)	Normalisation constant of the network when the population vector is K;
Pi(Ni|K)	Probability of queue i being in the state Ni, when the population vector is K;
Gi(Ni)	Normalisation constant of the network when only queue i is present;
R(i)	The set of chains visiting queue i;
Re(i)	The set of chains entering the network at queue i;
Rt(i)	The set of transit chains at queue i;
yr	R dimensional unit vector in the direction r. That is, a vector with a one in the rth position and zero elsewhere.

The equilibrium network state probability distribution, when the network is in the state S is,

$$P(S|K) = P(\text{State} = S \mid \text{chain r contains Kr jobs for r=1,2,...,R }). \quad (2.11)$$

From the laws of conditional probability,

$$P(S|K) = P(S,K)/G(K) \quad \text{if feasible,}$$
$$= 0 \quad \text{if not feasible.} \quad (2.12)$$

Where,

$$G(K) = P(\text{ Chain r contains Kr jobs, } r=1,2,\ldots,R).$$

G(K) is given by the sum,

$$G(K) = \sum_{S \in Fs} P(S,K),$$

with the set of states defined by,

$$Fs = \{S \mid Nir \geqslant 0 \text{ and } \sum_{i \in Q(r)} Nir = Kr$$

$$\text{for } r=1,2,\ldots,R; \quad K=K1,K2,\ldots,KR;$$

$$i=1,2,\ldots,1; \quad \text{and} \quad 1=L+M+R \}.$$

Comparing (2.2) and (2.12),

$$G(K) = 1/C. \tag{2.13}$$

The constant G(K) is the reciprocal of the normalisation constant C used in section 2.3.1. G(K) can be evaluated through the Reiser Kobayashi convolution algorithm [67,68,69]. An algorithm for determining the constant G(K) is given in appendix A. During the computation of G(K), one also obtains the constants G(Y) for all population vector Y, such that $0 \leqslant Y \leqslant K$.

2.3.3. THE MARGINAL QUEUE LENGTH PROBABILITIES

The summations in (2.8) and (2.9) to determine the blocking probabilities are rather involved. However, the

blocking probabilities can be determined with less effort from the marginal queue length probabilities. The marginal queue length probabilities of each queue in the network can be found using the constants G(Y), $0 \leq Y \leq K$.

The marginal queue length probability Pi(Nex,Nts|K) of queue i is defined as,

Pi(Nex,Nts|K) = P(the number of external messages at queue i is Nex and the number of transit messages at queue i is Nts, when the population vector is K).

The improper or un-normalised marginal queue length probability of queue i is given by the recursive relation,

$$\overset{*}{Pi}(Nex,Nts|\ K) = \sum_{r \in Re(i)} Uir\ \overset{*}{Pi}(Nex-1,Nts|\ K-yr) + \sum_{r \in Rt(i)} Uir\ \overset{*}{Pi}(Nex,Nts-1|\ K-yr)$$

$$\text{for } Nex + Nts \neq 0. \qquad (2.14)$$

$$\overset{*}{Pi}(0,0|\ K) = G(K) - \sum_{r=1}^{R} Uir\ G(K-yr)$$

$$\text{for } Nex + Nts = 0. \qquad (2.15)$$

Where,

$$Uir = eir/(a\ C'i)\ ;$$

$$K\text{-}yr = K1, K2, \ldots, Kr - 1, \ldots KR.$$

Proof

Let the state of queue i be represented by the vector Ni(Ni=Ni1,Ni2,...,NiR), where Nir is the number of chain r messages at queue i. The probability Pi(Ni| K), that queue i is in state Ni is,

$$Pi(Ni=Ni1,Ni2,\ldots,NiR|\ K) = \sum_{S \in Fi} P(S|\ K)$$

and

$$Fi = \{\ S \in Fs|\quad Ni=Ni1,Ni2,\ldots,NiR \quad \text{and} \quad \sum_{j \in Q(r)} Njr = Kr\ \}. \tag{2.16}$$

Where, the vector S=N1,N2,...,Nl and l=L+M+R. From equation (2.2), this may be expanded by separating the ith queue as follows:

$$Pi(Ni|\ K) = C(K)\ \{\sum_{r=1}^{R} Nir\}!\ \{\prod_{r=1}^{R} (1/Nir!)\ (eir/a\ C'i)^{Nir}\ \}$$

$$X \quad \{\sum_{S \in Fs-i} \prod_{\substack{j=1 \\ j \neq i}}^{1} (\sum_{r=1}^{R} Njr)!\ \prod_{r=1}^{R} (1/Njr!)\ (ejr/a\ C'j)^{Njr}\ \},$$

$$= C(K)\ Gi(Ni)\ G\text{-}i(K\text{-}Ni),$$

$$= Gi(Ni)\ G\text{-}i(K\text{-}Ni)\ /G(K)\ , \tag{2.17}$$

where,

$$C(K) = 1/G(K),$$

$$Gi(Ni) = \{ \sum_{r=1}^{R} Nir)! \ \{ \prod_{r=1}^{R} (1/Nir!) \ (eir/a \ C'i)^{Nir} \ \},$$

G-i(K-Ni) = Normalisation constant of the network with queue i removed,

Fs-i = The set of feasible states with queue i removed.

Gi(Ni) is the normalisation constant of a network consisting of queue i alone, when the population of queue i is Ni. It can be shown that [68],

$$Gi(Ni) = \sum_{r=1}^{R} Uir \ Gi(Ni-yr).$$

Equation (2.17) can now be written as,

$$Pi(Ni|\ K) = \{ \sum_{r=1}^{R} Uir \ Gi(Ni-yr)\} \ G-i(K-Ni) \ /G(K).$$

The above equation can be written as,

$$Pi(Ni|\ K) = \{ \sum_{r=1}^{R} Uir \ Gi(Ni-yr)\} \ \{G-i(K-yr-Ni+yr) \ /G(K-yr)\} \ X \ \{G(K-yr) \ /G(K)\}. \qquad (2.18)$$

Comparing equations (2.18) and (2.17), equation (2.18) can be written as,

$$Pi(Ni|\ K) = \sum_{r=1}^{R} Uir\ Pi(Ni-yr|\ K-yr)\ \ G(K-yr)\ /G(K). \tag{2.19}$$

Using the identity $G_i(0)=1$ [68], for Ni=0, equation (2.17) can be written as,

$$\begin{aligned} Pi(0|\ K) &= C(K)\ Gi(0)\ G\text{-}i(K), \\ &= G\text{-}i(K)\ /G(k). \end{aligned} \tag{2.20}$$

Using the identity [68],

$$G\text{-}i(K) = G(K) - \sum_{r=1}^{R} Uir\ G(K-yr),$$

equation (2.20) can be written as,

$$Pi(0|\ K) = 1 - \sum_{r=1}^{R} Uir\ G(K-yr)\ /G(k). \tag{2.21}$$

Equations (2.19) and (2.21) for the marginal queue length probabilities of queue i, conventionally form the basis for the derivation of the improper marginal queue length probabilities. In a development of this previous work, consider the separation of transit and external messages such that, Nex and Nts are the number of messages in the set Re(i) and Rt(i) respectively. The marginal queue length probability Pi(Nex,Nts| K) is given by,

$$Pi(Nex,Nts|\ K) = \sum_{\substack{\sum Nie = Nex \\ e \in Re(i) \\ \sum Nit = Nts \\ t \in Rt(i)}} Pi(Ni|\ K). \qquad (2.22)$$

From (2.19),

$$Pi(Nex,Nts|\ K)$$

$$= \sum_{\substack{\sum Nie = Nex \\ e \in Re(i) \\ \sum Nit = Nts \\ t \in Rt(i)}} \{[\sum_{r \in Re(i)} Uir\ Pi(Ni-yr|\ K-yr)\ G(K-yr)\ G(K)]$$

$$+ \quad [\sum_{r \in Rt(i)} Uir\ Pi(Ni-yr|\ K-yr)\ G(K-yr)\ /G(K)]\},$$

This can be written as,

$$= \sum_{r \in Re(i)} Uir \sum_{\substack{\sum Nie = Nex \\ e \in Re(i) \\ \sum Nit = Nts \\ t \in Rt(i)}} Pi(Ni-yr|\ K-yr)\ G(K-yr)\ /G(K)$$

$$+ \sum_{r \in Rt(i)} Uir \sum_{\substack{\sum Nie = Nex \\ e \in Re(i) \\ \sum Nit = Nts \\ t \in Rt(i)}} Pi(Ni-yr|\ K-yr)\ G(K-yr)\ /G(K)\ . \qquad (2.23)$$

Comparing equations (2.22) and (2.23), equation (2.23) can be written as

$$Pi(Nex,Nts|\ K) = \sum_{r \in Re(i)} Uir\ Pi(Nex-1,Nts|\ K-yr)\ G(K-yr)\ /G(K)$$
$$+ \sum_{r \in Rt(i)} Uir\ Pi(Nex,Nts-1|\ K-yr)\ G(K-yr)\ /G(K)$$
$$\text{for}\quad Nes + Nts \neq 0. \tag{2.24}$$

and from (2.21),

$$Pi(0,0|\ K) = 1 - \sum_{r=1}^{R} Uir\ G(K-yr)\ /G(K)$$
$$\text{for } Nex + Nts = 0. \tag{2.25}$$

The improper or un-normalised marginal queue length probabilities of queue i are given by multiplying equation (2.24) and (2.25) by G(K), resulting in the simple expression:

$$\overset{*}{Pi}(Nex,Nts|\ K) = \sum_{r \in Re(i)} Uir\ \overset{*}{Pi}(Nex-1,Nts|\ K-yr)$$
$$+ \sum_{r \in Rt(i)} Uir\ \overset{*}{Pi}(Nex,Nts-1|\ K-yr)$$
$$\text{for } Nex + Nts \neq 0. \tag{2.14}$$

and

$$\overset{*}{Pi}(0,0|\ K) = G(K) - \sum_{r=1}^{R} Uir\ G(K-yr)$$
$$\text{for}\quad Nex + Nts = 0. \tag{2.15}$$

Once the constants G(Y) ($0 \leq Y \leq K$) have been found,

the improper marginal queue length probabilities $\overset{*}{P}i(Nex,Nts|\ K)$ can be determined using the recursive relation given by (2.14) and (2.15). An algorithm for determining the improper marginal queue length distribution $\overset{*}{P}i(Nex,Nts|K)$ is given in appendix A. The proper marginal queue length probabilities are then given by,

$$Pi(Nex,Nts|\ K) = \overset{*}{P}i(Nex,Nts|\ K)\ /G(K)$$

$$\text{for} \quad Nex=0,1,\ldots,\sum_{r\in Re(i)}Kr,$$

$$\text{and} \quad Nts=0,1,\ldots,\sum_{r\in Rt(i)}Kr. \qquad (2.26)$$

2.3.4. MARGINAL POPULATION PROBABILITIES AND NODAL BLOCKING PROBABILITIES

The marginal population probability of node m is defined as,

PNm(Nex,Nts| K) = P(number of external messages at node m is Nex, number of transit messages at node m is Nts, given the population vector K).

That is,

PNm(Nex,Nts| K) = P(Nex1+Nex2+...+Nexp = Nex;
Nts1+Nts2+...+Ntsp = Nts;
given the population vector K).

Where, (1,2,...,p) are the set of queues at node m. Nexi and Ntsi are the number of external and transit messages at queue i. The probability distribution PNm(Nex,Nts|K) is given by the p-fold convolution sum [49]:

$$PNm(Nex,Nts|K) = P1(Nex,Nts|K) \circledast P2(Nex,Nts|K) \circledast .. \ldots \circledast Pp(Nex,Nts|K), \tag{2.27}$$

The convolution of Pi(Nex,Nts| K) and Pj(Nex,Nts| K) is given by the sum,

$$Pi(Nex,Nts|K) \circledast Pj(Nex,Nts|K) = \sum_{b=0}^{Nts} \sum_{a=0}^{Nex} Pi(a,b|K)\, Pj(Nex-a,Nts-b|K) \tag{2.28}$$

Once the marginal population probabilities of a node have been found, the blocking (or more precisely) the overflow probabilities can be evaluated. The blocking (or overflow) probability for external messages at node m is given by,

BEm = 1 - P(number of messages at node m is less than NTm and the number of external messages is less than NEm),

$$BEm = 1 - \sum_{\substack{Nex + Nts < NTm \\ Nex < NEm}} PNm(Nex,Nts|K). \tag{2.29}$$

The blocking (or overflow) probability for transit messages at node m is given by,

$$BTm = 1 - P(\text{number of messages at node m is less than NTm}),$$

$$BTm = 1 - \sum_{Nex + Nts < NTm} PNm(Nex,Nts \mid K) \qquad (2.30)$$

These probabilities are overflow probabilities rather than exact blocking probabilities, since the finite length queues in the network are approximated by infinite length queues.

2.3.5. PERFORMANCE MEASURES

Network performance measures on a virtual circuit basis can be calculated once the blocking probability vectors BT and BE (and the normalisation constant G(K)) have converged to a solution (section 2.3.1).

The throughput Vri of chain r messages at queue is given by [68,69],

$$Vir = eir\ G(K-yr)\ /G(K), \quad i \in Q(r). \qquad (2.31)$$

Here Vir=Vir(K) is the throughput of chain r at queue i, when the population vector is K, and Q(r) is the set of queues visited by chain r. In the case of fixed routing eir=1 and Vir=Vr, for all queues visited by chain r.

The mean number of class r messages in the network (excluding the rth token queue) is,

$$\bar{N}r = \sum_{i \in Q'(r)} \bar{N}ir, \tag{2.32}$$

where, $Q'(r)$ is the set of queues in the network visited by class r messages, except the rth token queue. The mean number of class r messages at queue i is given by the relation [68,69],

$$\bar{N}ir = Uir\ Gi_{+}(K-yr)\ /G(K), \tag{2.33}$$

where The factor Gi_{+} is given by the recursive relation [68,69],

$$Gi_{+}(K) = G(K) + \sum_{j \in R(i)} Uij\ Gi_{+}(K-yj).$$

An algorithm to determine the factor Gi_{+} is given in appendix A. $\bar{N}r$ is also given by,

$$\begin{aligned} \bar{N}r &= Kr - (\text{mean number of free tokens in} \\ &\qquad \text{the token queue Qtr associated with} \\ &\qquad \text{virtual circuit r}), \\ &= Kr - \{[1/(a\ Ar\ (1-BEm))]\ G_{Qtr_{+}}(K-yr)\ /G(K)\}. \end{aligned} \tag{2.34}$$

Where, it is assumed that chain r messages enter the network

at node m.

Applying Little's theorem [58], the transit or end-to-end delay incurred by chain r messages is given by,

$$Dr = \bar{N}r/Vr \qquad r=1,2,\ldots,R. \tag{2.35}$$

The total network throughput is

$$TN = \sum_{r=1}^{R} Vr \quad . \tag{2.36}$$

The mean network delay is,

$$DN = \sum_{r=1}^{R} \bar{N}r/TN \quad . \tag{2.37}$$

The transit delay is the mean time that elapses from the moment a message is accepted by the network, to the time it is successfully delivered to its destination host. The delay does not include the time a message has to wait outside the network, before it is admitted. The model assumes that such messages are lost.

2.4. COMPUTATIONAL COMPLEXITY OF THE CONVOLUTION ALGORITHM

The closed queueing network model is solved by an iterative procedure. During each iterative cycle, the normalisation constant G(K) of the network and the marginal queue length probabilities of all the queues in the network

are determined. The solution of the normalisation constant G(K) by the convolution algorithm (section 2.3.2 and appendix A) involves a recursive solution over all chain population from K=0,0,...,0 to K=K1,K2,...,KR. If L is the total number of queues in the network, R the number of chains and K=K1,K2,...,KR the population vector, the number of computations to determine the normalisation constant G(K) is of the order R.L.K1...KR, and the storage requirement is of the order K1.K2...KR. The determination of the queue length probabilities requires an operation count of the order N.N.R.K1...KR for each queue in the network, and the corresponding storage requirement is of the order 2.N.K1.K2...KR. Here, N is the sum of the population vectors of all chains visiting a queue. However, for determining the nodal overflow probabilities, N is equal to NT, the number of buffers at a node (equation 2.29 and 2.30).

2.5. PERFORMANCE EVALUATION OF THE CLOSED QUEUING NETWORK THROUGH THE MEAN VALUE ANALYSIS ALGORITHM

The mean value analysis algorithm (MVA for short) [71] can be used to evaluate the performance of the closed queueing network: this provides an alternative to the use of the convolution algorithm. The MVA algorithm evaluates the mean throughput of each closed chain, and the mean queue length of each queue in the network. A byproduct of this algorithm is the throughput of the closed chains for all chain population. This can be used in the evaluation of the queue length probabilities, from which the nodal blocking probabilities can be determined.

2.5.1. SOLUTION OF THE MULTI CHAIN CLOSED QUEUEING NETWORK WITH PRODUCT FORM SOLUTION

Consider the general multichain closed queueing network model described in the earlier sections, for which the following notations were defined:

M — Number of nodes in the network. This is also the number of exit links in the network.

R — Number of closed chains or virtual circuits. This is also the number of token queues in the network.

L Number of internode links in the network.

l Total number of queues in the network, l=L+M+R.

Kr Population of chain r, r=1,2,...,R.

K Population Vector, K=K1,K2,...,KR;

R(i) Set of chains visiting queue i, i=1,2,...,l.

Re(i) Set of chains entering the network at queue i, i=1,2,...,L+M.

Rt(i) Set of transit chains visiting queue i, i=1,2,...,L+M.

Ar Mean arrival rate of chain r messages to the network, r=1,2,...,R.

1/a Mean length of a message in bits;

Q(r) Set of queues visited by chain r messages, r=1,2,...,R.

eir Relative number of visits made by a chain r message to the service centre at queue i. This factor is determined by the routing probability matrix.

Ci Transmission capacity of link i in bits/sec.

BTm Probability of node m being blocked to transit messages;

BEm Probability of node m being blocked to external messages.

The following notations are defined in addition to those defined above.

Hi Effective service time of queue i;

Htr Effective service time of token queue Qtr, associated with chain r;

Hd Effective service time of exit queue d;

$\bar{N}ir$ Mean number of chain r messages at queue i;

Wir Mean queueing time for chain r messages at queue i, i=1,2,...,1 and $r \in R(i)$;

Vr Mean throughput of chain r;

The effective service time of the token queue Qtr, transit queue i, and the exit queue d are given by,

$$\begin{aligned} Htr &= 1/[Ar\ (1-BEm)] && r=1,2,\ldots,R; \\ Hi &= 1/(a\ C'i) && i=1,2,\ldots,L; \\ Hd &= 1/(a\ Cd) && d=1,2,\ldots,M; \\ C'i &= Ci\ (1-BTm) \end{aligned} \qquad (2.38)$$

Where, it is assumed that chain r messages enter the network at node m, and messages at queue i are routed to node m next.

The mean value analysis for closed queueing networks with product form solution states that, the state seen upon arrival instants has the same probability distribution as the steady state solution of the same closed system with one message removed in the arriving chain [70,71]. In [70,71] it is shown that the equilibrium performance statistics for a closed queueing exponential network with population Y, can

be expressed in terms of statistics for population (Y-yr), where yr is a R-dimensional unit vector in the direction r. The mean queueing time of a message at a queue is equal to its service time plus the mean backlog upon arrival. For FIFO scheduling, the mean queueing time for chain r messages at queue i when the population vector is Y ($0 \leqslant Y \leqslant K$), is given by [70,71],

$$Wir(Y) = Hi \left[1 + \sum_{j \in R(i)} \bar{N}ij(Y-yr) \right] . \tag{2.39}$$

The throughput of chain r message stream is,

$$Vr(Y) = Yr \Big/ \left[\sum_{i \in Q(r)} eir \; Wir(Y) \right] . \tag{2.40}$$

The mean number of chain r messages at queue i, i=1,2,...,l and $r \in R(i)$ is,

$$\bar{N}ir(Y) = Wir(Y) \; Vir(Y)$$

for r=1,2,...,R; i=1,2,...,l,

and Y=Y1,Y2,...,YR. (2.41)

The three equations are known as the single iteration MVA algorithm. The equations are solved by an R-dimensional recursion, starting at population Y=0,0,...,0 and working up to population Y=K1,K2,...,KR, with Nir(0)=0 for i=1,2,...,l and $r \in R(i)$.

2.5.2. QUEUE LENGTH PROBABILITIES AND NODAL BLOCKING PROBABILITIES

In the process of evaluating the throughput of the R chains with population vector K, one also evaluates the chain throughputs for all population Y, $(0 \leq Y \leq K)$. The marginal queue length probabilities can be found using the chain throughputs for all population vectors. From (2.24) and (2.25), the proper queue length probabilities of queue i are given by,

$$Pi(Nex,Nts|\ K) = \sum_{r \in Re(i)} Uir\ Pi(Nex-1,Nts|\ K-yr)\ G(K-yr)\ /G(K) + \sum_{r \in Rt(i)} Uir\ Pi(Nex,Nts-1|\ K-yr)\ G(K-yr)\ /G(K)\ , \quad \text{for } Nex + Nts \neq 0 \tag{2.24}$$

and

$$Pi(0,0|\ K) = 1 - \sum_{r=1}^{R} Uir\ G(K-yr)\ /G(K) \quad \text{for } Nex + Ntx = 0 \tag{2.25}$$

But $Uir = eir/(a\ C'i)$ and from (2.31) ,

$$Vr(K) = eir\ G(K-yr)\ /G(K) \quad \text{where } i \in Q(r).$$

Therefore,

$$Uir\ G(K-yr)\ /G(K) = (eir/a\ C'i)\ G(K-yr)\ /G(K), = Vr(K)\ /(a\ C'i)\ .$$

Hence the marginal queue length probabilities can be written as,

$$Pi(Nex,Nts|\ K) = \sum_{r \in Re(i)} [1/(a\ C'i)]\ Vr(K)\ Pi(Nex-1,Nts,|\ K-yr) + \sum_{r \in Rt(i)} [1/(a\ C'i)]\ Vr(K)\ Pi(Nex,Nts-1|\ K-yr), \quad \text{for } Nex + Nts \neq 0. \tag{2.42}$$

$$Pi(0,0|\ K) = 1 - \sum_{r=1}^{R} [1/(a\ C'i)]\ Vr(K), \quad \text{for } Nex + Nts = 0. \tag{2.43}$$

Once the queue length probabilities of all queues have been found, the marginal population probabilities can be evaluated, from which the nodal blocking probabilities can be found as in section 2.3.4.

2.5.3. SUMMARY OF THE SOLUTION BY THE MVA METHOD

The following procedural steps are required to solve the closed queueing network model by the MVA method. The procedure consists of initialisation, the recursive solution of the mean queueing time, throughput and population of each chain, and the determination of the nodal blocking probabilities.

1] Set the mean service rate of token queue Qtr (r=1,2,...,R) to the arrival rate Ar, of chain r message stream.

2] Initialise the blocking vectors as BT=0 and BE=0.

3] Using equation (2.39 - 2.41), evaluate the throughput for all chains and for all population vector Y, such that $0 \leqslant Y \leqslant K$.

4] Evaluate the queue length probabilities using (2.42 - 2.43), and then the nodal population probabilities (section 2.3.4).

5] Evaluate the blocking probability vectors BE and BT, using (2.29) and (2.30).

6] Modify the service time of all queues as in (2.38).

7] Repeat steps 2 to 6 until the blocking probabilities converge.

The virtual circuit throughputs and their end-to-end delays, the mean queueing delay at each queue, the mean queue length of all queues and the nodal blocking probabilities are available after the last iteration.

2.5.4. COMPUTATIONAL COMPLEXITY

The MVA algorithm also requires a complete solution, i.e., solutions for all population from K=0,...,0 to K=K1,K2,...,KR. If L is the number of queues in the network, R the number of chains and K=K1,K2,...,KR the population vector, the number of computations per iterative step to determine the chain throughputs is of the

order R.L.K1.K2...KR. This is of the same order as for the computation of the normalisation constant through the convolution algorithm. The storage requirements are of the order L.K1.K2...KR. In addition, the throughput of all chains for all population must be stored to determine the marginal queue length probabilities. This requires an additional storage of the order R.K1.K2...KR. The determination of the queue length probabilities require an operation count of the order N.N.R.K1.K2...KR for each queue in the network and the storage requirements are of the order 2.N.K1...KR, where N is the sum of the population vectors of all chains visiting a queue. Hence the MVA algorithm requires the same order of computation as the convolution algorithm. It also requires more storage space. However the algorithm is conceptually simpler and easier to implement than the convolution algorithm. Further it avoids a genuine problem of the convolution algorithm, namely, the floating point range of many computers may be easily exceeded leading to overflow and underflow [9,68,69].

CHAPTER THREE

A HEURISTIC METHOD OF SOLVING COMPUTER COMMUNICATION NETWORK MODELS WITH MULTIPLE LEVELS OF FLOW CONTROL

3. A HEURISTIC METHOD OF SOLVING COMPUTER COMMUNICATION NETWORK MODELS WITH MULTIPLE LEVELS OF FLOW CONTROL

3.1. INTRODUCTION

A closed queueing network model for computer communication networks with finite buffers, and equipped with multiple levels of flow control was developed in chapter 2. The network was solved numerically using the convolution algorithm and the mean value analysis algorithm. The problem with these two algorithms is that they require a complete solution for all population from 0 to K. That is, their computational complexity and storage requirements are proportional to K1.K2...KR. When the number of routing chains and/or their population are large, these algorithms require prohibitively large computation time and memory requirements. This limitation excludes their use for networks with four or more chains. Unfortunately, computer network modelling applications require many virtual circuits, which are mapped into closed chains. There may well be hundreds of chains and service centres, which makes the solution intractable by recursive techniques such as the convolution algorithm and the MVA algorithm.

Approximate methods provide alternative approachs to the solution of closed queueing networks with large number of chains and/or population: these solutions are computationally efficient and fast. The most important approach to approximation of closed queueing networks are aggregation

[6,8,80] and heuristic extensions to the MVA algorithm [10,70,71,80,87]. In aggregation, a portion of the network is replaced by a single composite queue (i.e, a queue with a queue length dependent service rate). In its interaction with the remaining queues in the network, the composite queue is intended to behave in the same way as the sub-network which it replaces. That is, the composite queue is flow equivalent to the section of the network it replaces. The resulting network usually has fewer states, and may have few enough states to be solved numerically. If not, the process of aggregation can be repeated until the network has a tractable solution. The aggregation process can be performed exactly for product form networks [6] and in the limiting case, for other networks (weekly coupled sub-networks which are independent of product form [15]). The aggregation technique can be extended to multichain networks. Consideration of multiple chain networks greatly increases the complexity of the problem since the complexity grows with the number of chains [8,80]. The complexity is also increased because of problems in characterisation of the composite queue. Since, the determination of the composite queue requires the solution of the chain throughputs for all population, the solution is of the same order of complexity as the direct use of the convolution or the MVA algorithm.

A second approach to the fast solution of closed queueing networks by approximate methods is provided by the heuristic extensions to the MVA algorithm as proposed by

Reiser and Lavenberg [70,71], and Schweitzer [87]. These algorithms are fast and have low memory requirements compared to the MVA algorithm and the convolution algorithm. The heuristic extensions to the MVA algorithm have improved in accuracy, speed, memory requirements and ease of implementation [10].

Approximate methods based on heuristic principles cannot be defended formally. By basing the heuristic approximations on methods which are exact for product form networks, the results might be expected to be close to the actual solution. The heuristic extensions to the MVA algorithm is asymptotically valid for large population [70]. The accuracy of the heuristic methods may be verified by simulation if an exact solution is not possible [70].

The heuristic extensions to the MVA algorithm as proposed by Reiser and Lavenberg [70,71], replaces the R dimensional recursive equation (2.39-2.41) by an efficient iterative procedure. The complexity per iterative step is of the order R.L.(2+K1+K2+...+KR). This is certainly affordable for large networks. However, the heuristic extensions to the MVA algorithm compute only the mean chain throughputs and the mean chain delays. They do not give the marginal queue length probabilities, which are essential for the determination of the nodal blocking probabilities. Hence, they can only be applied to closed multichain networks without nodal blocking. That is, networks for which the number of buffers

at each node, is greater than the sum of the population of all chains visiting the node.

In this chapter, a method for evaluating the approximate queue length probabilities based on the heuristic extensions to the MVA algorithm and a flow equivalent reduced network, is developed. The method can be used to solve networks with limited buffers and large number of routing chains, but with sparsely populated queues. A queue is said to be sparsely populated if the number of chains and/or their population visiting this queue is sufficiently low to allow numerical solution with reasonable resource and time constraints. By applying the method of aggregation to the equivalent reduced network, the number of states can be reduced further, making it possible to remove this limitation. The closed queueing network model and its heuristic solutions are validated by a simulation experiment in chapter 4. The method is extended further to account for the end-to-end acknowledgement delays on virtual circuits. This is incorporated in the model in section 3.6.

The use of these techniques are demonstrated in section 3.7. A study is made of a five node network and the analytic results obtained by the heuristic methods are compared with the results obtained by the exact method (based on the convolution algorithm). The results obtained by the heuristic methods and the exact method differ by less than 10% for throughputs and by less than 15% for delays, even at high

loads. The heuristic methods based on the equivalent reduced network are computationally efficient, fast and easy to implement with low storage requirements. These techniques can be used to evaluate performance measures like virtual circuit throughput, end-to-end delay and nodal blocking probabilities, in large networks with finite buffers and many routing chains.

3.2. NETWORKS WITH SPARSELY POPULATED QUEUES

Consider a network with 1 queues and R chains with a population vector K=K1,...,KR. Let the number of closed chains in the network and their population be such that any computation of the order K1.K2...KR is prohibitively large or infeasible. This rules out the use of exact algorithms, such as the convolution and the MVA algorithm, for the solution of the networks. Let Fi be a function such that,

$$Fi = \prod_{r \in R(i)} Kr \qquad i=1,2,...,1;$$

where, R(i) is the set of chains visiting queue i. The function Fi is the product of the population of the chains visiting queue i. A network is said to have sparsely populated queues if, computations of the order Fi are feasible and not prohibitively large. That is, for a network with sparsely populated queues,

$$\text{Max}\left[\prod_{j \in R(i)} Kj\right] << \prod_{r=1}^{R} Kr, \qquad \text{for } i=1,2,\ldots,l.$$

3.3. THE EQUIVALENT REDUCED NETWORK

The evaluation of the queue length probabilities require the use of the convolution or the MVA algorithm. The exact algorithms assigns a state to each possible ordering of the messages in the network. Thus the number of states can be enormous if all chains in the network are considered. The equivalent reduced network [73,74,85] reduces the number of states by discarding chains that do not use the queue of interest (i.e, the queue whose queue length probabilities are to be determined). The equivalent reduced network is based on the property that, in any exponential network with product form solution, background traffic streams (i.e., data flow outside the virtual circuits of interest) can be taken into account by simply subtracting from the channel capacity of the links, the portion used by these background traffic [74]. This property holds only if the background traffic streams are independent of the flow in the virtual circuits of interest. In the closed queueing network under consideration, the throughput of those chains which are not of interest at a queue forms the background traffic at that queue. Since the network is closed, the background traffic is dependent on the flow in the virtual circuits of interest and the method of simplification using adjusted service

rates is not strictly valid.

If the assumption is made that the method of adjusted service rates can be used to simplify the network, then some degree of approximation will be introduced. However, the resulting equivalent reduced network still has a product form solution since any change in the mean service rate of a queue does not alter the exponential nature of the service centre. This means that precisely equivalent results are obtained in so far as the remaining closed chains (which are of interest) are concerned, by subtracting the throughput of the background traffic from the capacity of the links used by them.

Let i(i=1,2,...,1) be any queue whose queue length probabilities are to be found. Let R(i) be the set of chains visiting queue i, and let F(i) be the set of queues visited by the set of chains in R(i). Let the throughput of chain r(r=1,2,...,R) be Vr messages per second. It is assumed that the chain throughputs have already been found. The equivalent reduced network for queue i is formed by replacing the rest of the network (i.e, the network without queue i) by an equivalent sub-network with fewer states. The first step in deriving this reduced network, is to delete all queues in the network that are not visited by the set of chains in R(i). That is, all queues in the network that do not belong to the set F(i) are deleted. Next, consider queue j which is an element of the set F(i). Let R(j) be the

set of chains visiting queue j, and R'(j) be the set of chains that visit queue j but do not belong to the set R(i). That is, chains in R'(j) visit queue j but not queue i. Viz.,

$$R'(j) \in R(j),$$
$$R'(j) \notin R(i),$$
$$j \in F(i).$$

The set of chains in R'(j) forms the background traffic at queue j, in so far as queue i is concerned. The throughput of the background traffic at queue j, in the equivalent reduced network of queue i is,

$$VBji = \sum_{\substack{r \in R'(j) \\ j \in F(i) \\ j \neq i}} Vr \quad .$$

This background traffic which is now assumed to be independent of the flow in the virtual circuits of intrest is subtracted from the transmission capacity of link j. The modified service rate of queue j, after removing the background traffic is,

$$C'ji = Cj - (VBji / a).$$

Where, the channel capacities are expressed in bits per second and 1/a is the mean length of a message in bits. This is repeated for all queues visited by the set of chains

in R(i), except queue i. The resulting network which consists of queue i, the chains visiting queue i and the remaining queues visited by these chains (with adjusted link capacities), forms the equivalent reduced network in so far as queue i is concerned. This reduced network is flow equivalent to the original network, in so for as the chains visiting queue i are concerned. That is, the flow of messages belonging to the chains visiting queue i in the original network, is the same as the flow of messages in the chains of the reduced network. Since the reduced network has discarded chains that do not visit queue i, the number of states involved in the determination of the queue length probabilities of queue i by the exact algorithm is reduced.

3.4. SOLUTION OF THE FINITE BUFFER CLOSED QUEUEING NETWORK MODEL BY THE METHOD OF THE EQUIVALENT REDUCED NETWORK

The closed queueing network model is solved by an iterative procedure. Starting with an initial value for the blocking probability vectors BE and BT, the closed queueing network is solved and the marginal queue length probabilities and the nodal blocking probabilities are determined. The process of iteration is continued until the blocking probabilities converge. The method of equivalent reduced network can be used to find the approximate marginal queue length probabilities. First, the chain throughputs are found using the heuristic extensions to the MVA algorithm (section 3.4.1). With the throughput of the chains known,

the marginal queue length probabilities of each queue in turn can be found by constructing the equivalent reduced network of the queue in question, as discussed in section (3.4.2). The marginal queue length probabilities of the queue can be found by solving the equivalent reduced network through the convolution or the MVA algorithm as discussed in chapter 2. Once the queue length probabilities of all queues have been found, the nodal blocking probabilities can be evaluated as in section (2.3.4). The process of iteration is continued until, the blocking probabilities converge. The following procedural steps are required to solve the closed queueing network model by the method of the equivalent reduced network.

1] Assume an initial value for the blocking probability vectors BE and BT;

2] Determine the chain throughputs using the heuristic extensions to the MVA algorithm (section 3.4.1);

3] Form the equivalent reduced network of each queue in the network (section 3.4.2), using the chain throughputs from step 2;

4] Using the convolution algorithm or the MVA algorithm, determine the marginal queue length probabilities of each queue (section 2.3 and 2.5), from their associated equivalent reduced network;

5] Determine the new values for the nodal blocking probability vectors BE and BT (equation 2.29 and 2.30);

6] Modify the service times of all queues (equation 2.38), using the values of the new blocking probability vector from step 5.

7] Repeat steps 2 to 6 until the blocking probability vectors converge to a solution.

3.4.1. THE HEURISTIC EXTENSIONS TO THE MEAN VALUE ANALYSIS ALGORITHM

Consider the general closed queueing network model with M nodes, L links and R virtual circuits between designated pairs of nodes, as described in section 2.1 and 2.2. The chain throughputs of the closed queueing network are found by heuristically extending the MVA algorithm. The notations used in this section are defined in section 2.3.

The following service time parameters were also defined for the l(l=L+M+R) queues in the network.

$$\begin{aligned} Htr &= 1/[Ar\ (1-BEm)] && \text{for } r=1,2,..,R, \\ Hi &= 1/(a\ C'i) && \text{for } i=1,2,...,L, \\ Hd &= 1/(a\ Cd) && \text{for } d=1,2,...,M, \\ C'i &= Ci\ (1-BTm), \end{aligned} \qquad (2.38)$$

Where, it is assumed that chain r messages enter the network at node m, and messages at queue i are routed to node m

next.

A method was derived for the solution of the closed queueing network model using the single iteration MVA algorithm in section 2.5. Consider the equations for the mean queueing time of chain r messages at queue i,

$$Wir(Y) = Hi \left[1 + \sum_{j \in R(i)} \bar{N}ij(Y-yr) \right]; \tag{2.39}$$

the mean throughput of chain r,

$$Vr(Y) = Yr \Big/ \sum_{i \in Q(r)} [eir \; Wir(Y)]; \tag{2.40}$$

and the mean number of chain r messages at queue i,

$$\bar{N}ir(Y) = Vr(Y) \; Wir(Y)$$
$$\text{for } r=1,2,\ldots,R; \quad i=1,2,\ldots,1; \tag{2.41}$$

when the population vector is Y (Y=Y1,Y2,...,YR). The three equations are solved by an R-dimensional recursion starting at population (0,0,...,0) and working up to population K=K1,K2,...,KR.

In the heuristic extensions to the MVA algorithm suggested by Reiser and Lavenberg [70,71], this recursion is replaced by an efficient iteration. A correction factor dij(r) is introduced, where,

$$dij(r) = \bar{N}ij(K) - \bar{N}ij(K-yr).$$

$$i=1,2,\ldots,l \text{ and } j \in R(i) \qquad (3.1)$$

The factor dij(r) is the difference in the mean queue size of chain j at queue i, when the population vector is K and K-yr (i.e., one message less in chain r). In its exact form, the evaluation of dij(r) is computationally as complex as the original recursion. In order to estimate the correction factor dij(r) (and hence reduce the computational effort), the following heuristic approximations are made.

1] The chain with one message less is affected most. That is,

$$dir(r) >> dij(r) \quad \text{for} \quad i=1,2,\ldots,l \quad \text{and} \quad j \in R(i),\ r \in R(i).$$

Hence the component dij(r), $j \neq r$ is neglected.

2] For the affected chain (j=r), the correction factor can be estimated by a single chain problem with suitably redefined parameters for the service times of the queues visited by chain r. The correction factor dir=dir(r) is given by,

$$dir = \bar{N}'ir(Kr) - \bar{N}'ir(Kr-1). \qquad (3.2)$$

Where, $\bar{N}'ir(Kr)$ is the mean number of chain r messages at queue i in the single chain problem, when the population of chain r is Kr. From equation (3.1), substituting for $\bar{N}ij(K-yr)$ in equation (2.39) with Y=K, the following set of non-linear equations result.

The mean queueing time of chain r messages at queue i is,

$$Wir = Hi \left[1 + \sum_{j \in R(i)} (\bar{N}ij - dij) \right], \tag{3.3}$$

The throughput of chain r is,

$$Vr = Kr \Big/ \sum_{i \in Q(r)} [eir\ Wir], \tag{3.4}$$

The mean number of chain r messages at queue is,

$$\bar{N}ir = Vr\ Wir, \quad \text{for } r=1,2,\ldots,R; \text{ and } i=1,2,\ldots,l. \tag{3.5}$$

The correction factor dir for each i=1,2,...,l and r=1,2,...,R is found by solving a single chain problem involving only chain r and the queues visited by chain r, with suitably redefined service time parameters. In this single chain problem, the service time parameters are defined as :

$$H'tr = 1/[Ar\ (1-BEm)]$$

for the token queue of chain r

r=1,2,...,R;

$$H'ir = 1/[a\ (Ci - \sum_{\substack{j \in R(i) \\ j \neq r}} Vj)\ (1-BTm)]$$

for transit queue i(i=1,2,...,L+M)

visited by chain r. (3.6)

With the service time parameters of the single chain network defined, the correction factors dir, $i \in Q(r)$, are found by solving the following set of recursive equations (starting at Yr=0 and working up to Yr=Kr):

$$W'ir(Yr) = H'ir\ [\ 1 + \bar{N}'ir(Yr-1)], \tag{3.7}$$

$$V'r(Yr\ ) = Yr/\sum_{i \in Q(r)} [eir\ W'ir(Yr)], \tag{3.8}$$

$$\bar{N}'ir(Yr) = V'ir(Yr)\ \ W'ir(Yr),$$

for Yr=1,2,...,Kr. (3.9)

The correction factor dir is given by,

$$dir = \bar{N}'ir(Kr) - \bar{N}'ir(Kr-1)$$

for i=1,2,...,1; and $i \in Q(r)$. (3.2)

The three non-linear equations (3.3-3.5) can be solved

by an iterative procedure. The algorithm for determining the throughput of the closed chains, through the heuristic extension to the MVA algorithm is summarised below.

1] Initialise $\bar{N}ir$ and Vr for i=1,2,...,1 and r=1,2,...,R. The initial value of $\bar{N}ir$ can be found from step 2 by setting $\bar{N}ir=\bar{N}'ir(Kr)$ and Vr=0.

2] Evaluate the correction factor dir (equation 3.2) for i=1,2,...,1 and $i \in Q(r)$, using a single chain problem (3.7-3.9) with suitably redefined parameters (3.6).

3] Determine the mean queueing time Wir, for i=1,2,...,1, $i \in Q(r)$ and r=1,2,...,R, using (3.2).

4] Find the mean chain throughput Vr for r=1,2,...,R, using (3.3).

5] Find the mean chain population $\bar{N}ir$, i=1,2,...,1; where $i \in Q(r)$, and r=1,2,...,R, using (3.4).

6] Repeat steps 2 to 5 until the chain throughputs converge.

3.4.2. THE EQUIVALENT REDUCED NETWORK AND THE MARGINAL QUEUE LENGTH PROBABILITIES

Having found the throughput of all the closed chains, the approximate marginal queue length probabilities of all the queues can be found, using the equivalent reduced network. Let V1,V2,...VR be the throughput of the R closed

chains that have been evaluated using the heuristic extensions to the MVA algorithm. The marginal queue length probabilities of queue i(i=1,2,...,L+M) are found from the equivalent reduced network of queue i, for which the following sets are defined.

R(i) Set of chains visiting queue i,i=1,2,...,L+M.

F(i) Set of queues visited by the chains in R(i), and $i \notin F(i)$.

R'(j) Set of chains visiting queue j, but not queue i, and $j \in F(i)$.

The equivalent reduced network of queue i is formed, by first removing the queues that do not belong to the set F(i). That is, queues that are not visited by chains visiting queue i, are removed. The remaining queues (other than queue i) belong to the set F(i).

Consider queue j which is a queue in the set F(i). A chain r message at queue j ($r \in R(j)$), sees only a fraction of the server capacity queue j. The rest of the server capacity is devoted to other chains visiting queue j. The capacity devoted to chain p at queue j ($p \in R(j)$), is (Vp/a), where Vp is the throughput of chain p in messages per second, and 1/a is the mean length of a message in bits. Hence the server capacity of queue j devoted to chains that do not belong to the set R(i) is $\sum(Vp/a)$; $p \in R'(j)$. The server capacity available for chains belonging to the set R(i) at queue j is,

$$C'(i,j) = Cj - \sum_{p \in R'(j)} (Vp\ /a).$$

The mean service time for a message belonging to the set R(i) at queue j, after removing the background traffic is,

$$H(i,j) = 1\ /\ [\ a\ C'(i,j)]$$
$$= 1/\{a\ Cj\ [1 - \sum_{p \in R'(j)} (Vp\ /(a\ Cj))]\}.$$

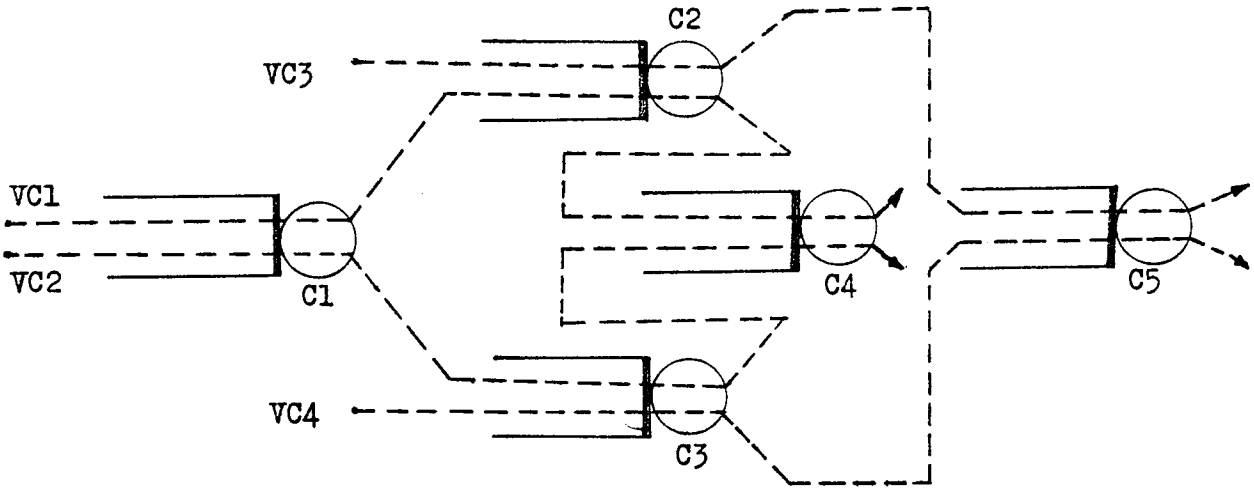
Since Hj = (1/ a Cj), then H(i,j) can be written as,

$$H(i,j) = Hj\ /(1 - \sum_{p \in R'(j)} Vp\ Hj). \qquad (3.10)$$

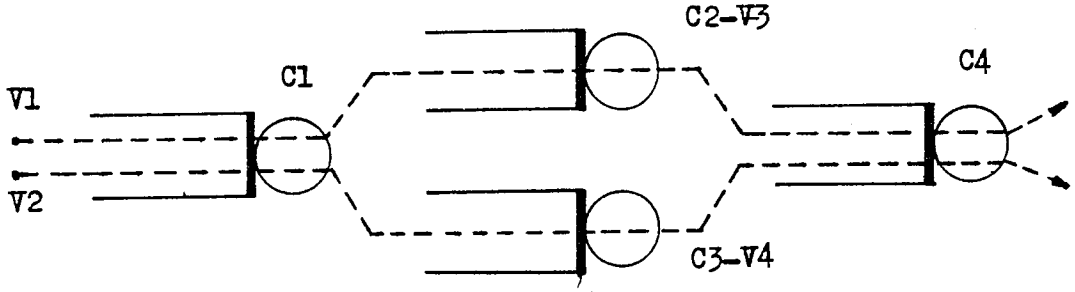
Queue j with adjusted service rate in the equivalent reduced network is equivalent to queue j in the original network, in so far as the flow of messages in the chains belonging to the set R(i) are concerned. The actual service time of queue j in the equivalent reduced network, taking into account the nodal blocking probabilities, is given by,

$$H'(i,j) = H(i,j)\ /(1-BTm). \qquad (3.11)$$

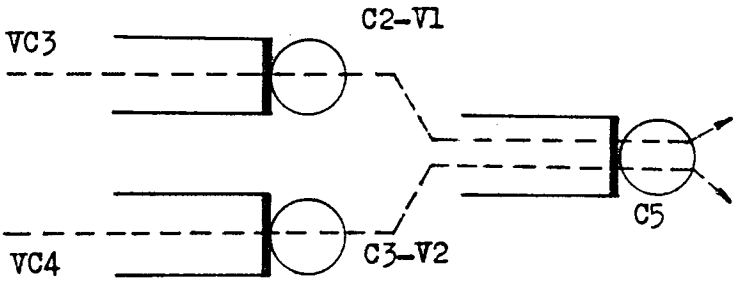
Where, it is assumed that messages queueing at queue j are routed to node m next. Fig. 3.1a, shows a network with four chains and five queues. Fig. 3.1b, shows the equivalent reduced network for queues 1,4 and 5.



(a)



(b)



(c)

Fig. 3.1 (a) Queueing model of a network. (b) Equivalent reduced network of queue 1 and queue 4, and (c) queue 5.

Once the equivalent reduced network of queue i(i=1,2,...,L+M) has been found, the marginal queue length probabilities of queue i can be found using the convolution algorithm or the MVA algorithm (chapter 2). Having found the marginal queue length probabilities of all queues, the nodal blocking probabilities of all nodes can be determined using (2.29) and (2.30).

The steps to be followed, to determine the marginal queue length probabilities of queue i(i=1,2,...,L+M) by the method of the equivalent reduced network is summarised below.

1] Identify the set of chains R(i), that visit queue i.

2] Identify the set of queues F(i) visited by the chains in the set R(i), with $i \notin F(i)$.

3] Delete all queues j(j=1,2,...,1), not in the set F(i), and $j \neq i$.

4] For queue j, $j \neq i$ and $j \in F(i)$, identify all chains that visit queue j but do not belong to the set R(i), that is, the chains in the set R'(j).

5] Determine the mean service time H(i,j) of queue j, after removing the background traffic contributed by the chains in the set R'(j) (3.10).

6] Determine the modified service time H'(i,j) taking into account the effect of blocking at the next node (3.11).

7] Repeat steps 4 to 6 for all queues in the set F(i) and hence, form the equivalent reduced network of queue i.

8] Using the equivalent reduced network of queue i, determine the marginal queue length probabilities of queue i, by the convolution or the MVA algorithm developed in chapter 2.

3.4.3. COMPUTATIONAL COMPLEXITY

Let R be the number of chains and L be the number of queues in the network. The number of computations per iterative step to determine the chain throughputs using the heuristic extensions to the MVA algorithm is of the order R.L.(2+K1+K2+...+KR). The storage requirements are of the order L.R. The operation count to determine the marginal queue length probabilities of queue i, is of the order $(N^2+Q)P\prod Kj$, $j \in R(i)$, and the storage requirement is of the order $2.N\prod Kj$, $j \in R(i)$. Where,

P — the number of chains visiting queue i, i=1,2,...,L+M;

Q — the number of queues visited by chains in R(i);

N — the sum of the chain population of chains visiting queue i;

Table 3.1 compares the computational effort and the storage requirements for the three methods using the convolution algorithm, the MVA algorithm and the method based on

Table 3.1

Computational effort and storage requirements.

Method	Chain throughput		Marginal queue length probabilities	
	Computational effort	Storage requirement	Computational effort	Storage requirement
Convolution algorithm	$RL\prod_{r=1}^{R} Kr$	$\prod_{r=1}^{R} Kr$	$N^2 R\prod_{r=1}^{R} Kr$	$2N\prod_{r=1}^{R} Kr$
M.V.A algorithm	$RL\prod_{r=1}^{R} Kr$	$(L+R)\prod_{r=1}^{R} Kr$	$N^2 R\prod_{r=1}^{R} Kr$	$2N\prod_{r=1}^{R} Kr$
Equivalent reduced network	$RL(2+\sum_{r=1}^{R} Kr)$	LR	$(N^2+Q)P\prod_{j \in R(i)} Kj$	$2N\prod_{j \in R(i)} Kj$

the equivalent reduced network and the heuristic extensions to the MVA algorithm. In a large network with sparsely populated queues, the number of chains and/or chain population at any queue (worst case) will be less than the total number of chains and/or chain population in the whole network. That is,

$$\text{Max}\{\prod_{j \in R(i)} Kj\} \quad << \quad \prod_{r=1}^{R} Kr.$$

Hence the total operation count and storage requirements using the heuristic extensions to the MVA algorithm, and the equivalent reduced network can be reduced considerably. This makes the method suitable for the solution cf closed finite buffer networks with large number of chains and/or chain population.

3.5. FURTHER AGGREGATION OF THE EQUIVALENT REDUCED NETWORK

The equivalent reduced network will now be aggregated further, leading to lower computational effort. The aggregation is applied to the portion of the equivalent reduced network, that does not contain the queue whose marginal queue length probabilities are to be found. The aggregation is carried out by replacing the portion of the equivalent reduced network without the queue of interest, by a set of composite queues with queue length dependent service rates [6,8,80].

Consider a network where the queue length probabilities of queue i have to be found. Let R(i) be the set of chains visiting queue i. To begin with, the equivalent reduced network of queue i is formed first (sec. 3.4.2). The portion of the equivalent reduced network without queue i can be reduced further by applying the Norton's theorem of queueing networks [6]. The method consists in decomposing a network (in this case the equivalent reduced network) into two closed systems. One of the closed system is the queue that is under study (in this case, queue i) and the other closed system being the rest of the network. The second closed system forming the rest of the network without queue i, is called the complement of queue i.

The method consists in replacing the complement network by a set of composite queues that is flow equivalent to the original network it replaces. The composite queues can be found by considering a short circuit to be placed in each chain between the input and output of queue i (thus removing queue i) and determining the throughput of each shorted chain for all population. The aggregated equivalent reduced network consists of queue i and a set of composite queues (one for each chain visiting queue i). The service rate of the jth composite queue, $j \in R(i)$, is determined by the throughput of chain j, found without queue i. The composite queue service rates are queue length dependent [6,8,80]. The service rate of the jth composite queue when the population vector is Y, is represented by $Ccj(Y) = Vcj(Y)$.

Where, Vcj(Y) is the throughput of chain j in the complement network (i.e, without queue i), when the chain population vector is Y.

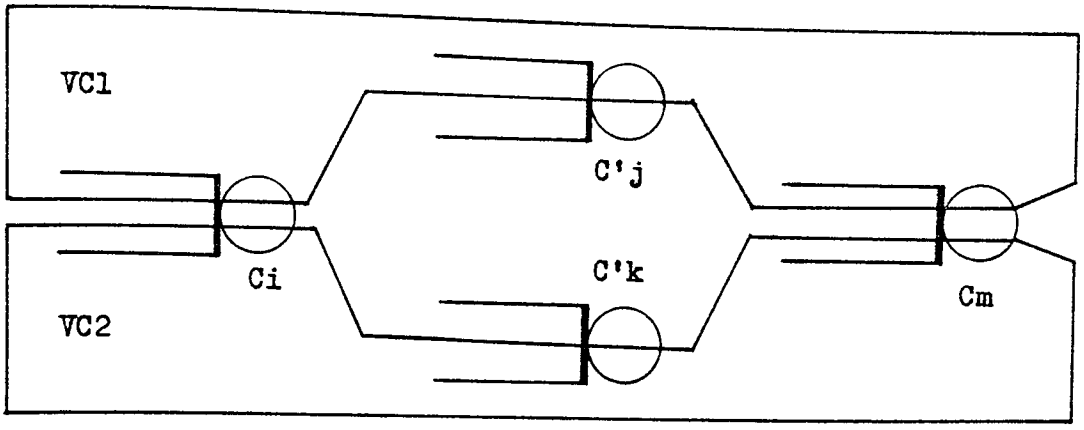
The process of aggregation is applied only to the equivalent reduced network and not to the original network. Characterisation of the composite queues requires the evaluation of the throughput of all shorted chains, for all chain population. Since the equivalent reduced network contains fewer chains and queues than the original network, the equivalent composite queues can be determined with less computational effort for the equivalent reduced network. However, the application of the Norton's theorem to the equivalent reduced network will reduce the computational effort, only if the chains in the equivalent reduced network have few queues in common. That is, the chains are dispersed in the equivalent reduced network.

The process of aggregation using the Norton's theorem is exact for product form networks (such as the original network). Since the equivalent reduced network considered here is also a product form network, the application of the Norton's theorem to this network gives results which are identical to those obtained directly from the equivalent reduced network.

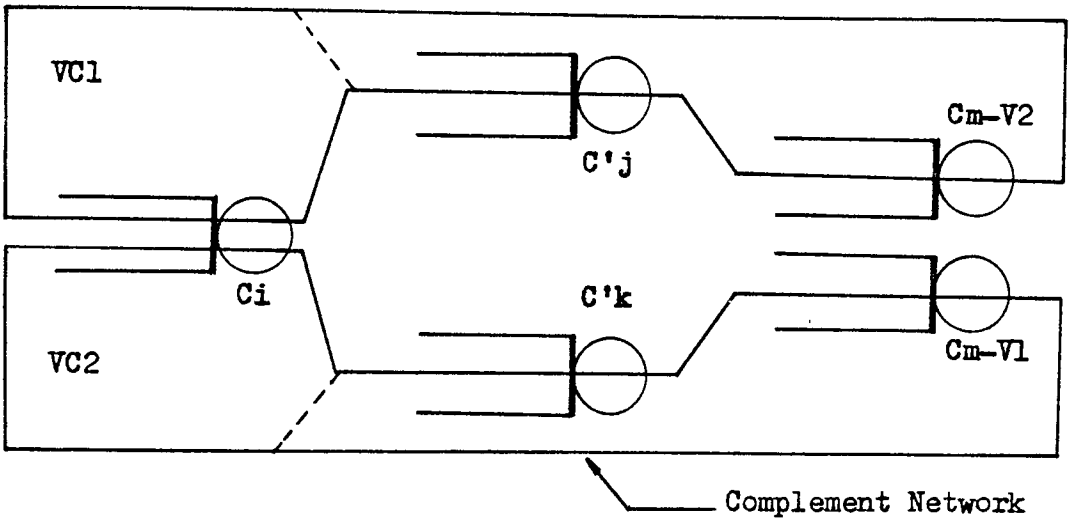
In the case where the chains are not dispersed, the following approximations can be used to reduce the computational effort. Consider again the evaluation of the queue

length probabilities of queue i in the equivalent reduced network. Since the chains are not dispersed, there will exist a number of instances in which two or more chains that visit queue i will also share visits to other queues. Consider such a queue, m, visited by chain j and k (that is, $j,k \in R(i)$). Queue m is replaced by two queues mj and mk with adjusted service rates. Queue mj is visited only by chain j and queue mk is visited only by chain k. The service capacity of queue mj is the server capacity experienced by chain j messages at queue m in the equivalent reduced network, and is found by subtracting the throughput of chain k from the server capacity of queue m. That is, chain k is treated as a background traffic at queue m, in so far as chain j is concerned (section 3.3.1). Similarly, the service rate of queue mk is found by subtracting the throughput of chain j from the service rate of queue m in the equivalent reduced network. This process is repeated for all queues in the equivalent reduced network, except queue i. The resulting network consists of queue i and a set of single chain queues. Fig. 3.2a shows the equivalent reduced network of queue i and fig. 3.2b shows the modified network.

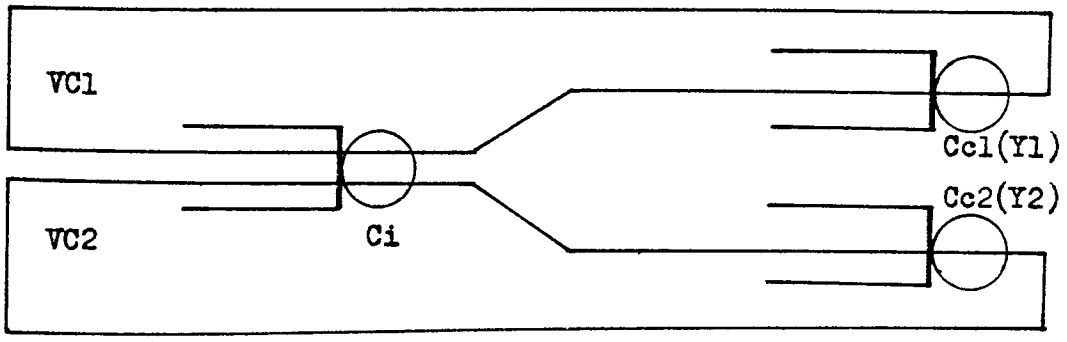
If the Norton's theorem is now applied to this modified network by removing queue i, the resulting network is made up of P independent closed single chain networks. Where, P is the number of chains visiting queue i. Each of these P closed single chain network can be replaced by a composite queue. The service rate Ccj(Yj) of the jth composite queue,



(a)



(b)



(c)

Fig. 3.2 (a) Equivalent reduced network of queue i.
(b) Modified single chain complement network.
(c) Final aggregated network with composite queues.

$j \in R(i)$, is found by solving a single chain problem and finding the throughput Vcj(Yj), for all chain j population from 0 to Kj. The final aggregated network now consists of queue i and P composite queues, as shown in fig. 3.2c.

The solution of each composite queue involves only a single chain and thus the order of computation can be reduced considerably, especially if the chain population is large and the chains are concentrated. Assume that there are Q queues in the equivalent reduced network excluding queue i, and there are P chains with population K for each chain. Further, let all the P chains visit all the Q queues. The order of computation involved in determining the composite queues, without reducing the complement network to a set of single chain network is, P.Q.K1.K2...KR. If the complement network is reduced to P independent single chain networks, the order of computation to determine the P composite queues is P.Q.(2+K1+...+KR). Hence, for networks with large number of chains and/or chain population, this method can result in considerable reduction of computational effort.

The final aggregated network consisting of queue i and the P composite queues, can be solved using the convolution algorithm or the MVA algorithm for queues with queue length service rates [72]. The convolution algorithm is preferred, since it is numerically more stable than the MVA algorithm when solving network of queues with queue length dependent service rates [9,80].

3.6. END-TO-END ACKNOWLEDGEMENT DELAYS OF VIRTUAL CIRCUITS

The model analysed so far assumes that the end-to-end acknowledgement (ack for short) delays are small, and hence are neglected. Such will be the case if ack messages are returned to the source through seperate high speed channels. However, if the ack messages use normal data channels, the ack delays can be significant. Free tokens will take a longer time to return to their respective token queues, resulting in reduced virtual circuit throughput. End-to-end acknowledgement of virtual circuit messages can be piggy backed on data messages travelling in the opposite direction. They can also be stand-alone messages, with or without priority. In this analysis they are assumed to be short stand-alone messages, with priority over normal messages. Further, it is assumed that there are additional buffers at each node that can accommodate all the ack traffic. Hence ack messages are not blocked at any node.

The ack traffic of chain r is Vr, where Vr is the throughput of chain r(r=1,2,...,R). The ack messages are short in length. The mean length of an ack message is 1/ak bits, and 1/ak << 1/a, where 1/a is the mean length of a data message. The set of queues traversed by a chain r ack message is denoted by Qa(r). The total delay suffered by chain r ack messages is,

$$Dar = \sum_{i \in Qa(r)} Wi,ar \ ,$$

(3.12)

where Wi,ar is the delay suffered by a chain r ack message at queue i. Since the ack messages are short and have priority, their queueing delays are also small compared to the queueing delays of data messages. The capacity used by the ack traffic can be accounted, by subtracting the ack traffic from the link capacity. Hence the actual capacity available for data traffic at queue i is

$$Cdi = Ci - \sum_{r \in Ra(i)} Vr/ak \ ,$$

(3.13)

where Ra(i) is the set of ack chains visiting queue i.

Since the traffic intensity of ack messages at any queue is low (because of their small service time), the queueing delay for ack messages can be approximated. The queueing delay of ack messages is modelled as an open M/G/1 queue with the ack messages having non-preemptive priority. The mean waiting time for a higher priority message (in this case, an ack message) in an M/G/1 queue with two priority classes is given by [42,81]

$$W'i,ar = \frac{\sum_{r \in Ra(i)} Ui,ar \ Hi,ar \ + \ \sum_{j \in R(i)} Uij \ Hi}{1 - \sum_{r \in Ra(i)} Ui,ar} \ ,$$

(3.14)

where,

$Ui,ar = Vr/(ak\ Ci)$,

$Ui,j = Vj/(a\ C'i)$,

$Hi,ar = 1/(ak\ Ci)$,

$C'i = Ci\ (1-BTm)$

$Hi = 1/(a\ C'i)$,

BTm probability of node m being blocked. It is assumed that messages at queue i are routed to node m next.

The transmission time of an ack message at queue i is $1/(ak\ Ci)$. Hence the total queueing delay for ack messages at queue i is

$$Wi,ar = W'i,ar + 1/(ak\ Ci) .$$

The total ack delay of chain r is,

$$Dar = \sum_{i \in Qa(r)} Wi,ar.$$

Once a message is delivered to its host, the token returns as an ack message to its token queue. The number of tokens is equal to the sum of the free tokens in the token queue, the messages in transit and the returning ack messages. The throughput of chain r in equation (3.4) and (3.8) is modified as,

$$Vr = Kr/[\sum_{i \in Q(r)} Wir + \sum_{i \in Qa(r)} Wi,ar] \; . \qquad (3.15)$$

Since the solution of the chain throughputs depend on the ack delays, the ack delays are evaluated for each iterative step of the MVA heuristic algorithm, using the values computed in the previous iterative step.

3.7. NUMERICAL RESULTS

This section presents the numerical results for a five node network with finite buffers, solved analytically by the exact method using the convolution algorithm (chapter 2), and the heuristic methods developed in this chapter (section 3.4 and 3.5). The results obtained by the heuristic methods are compared with those of the exact method to verify the accuracy of the heuristic solution.

The five node network has four virtual circuits which are controlled by an end-to-end flow control mechanism. In addition, there is a network access control at each node and a hop level control between the nodes. The network and its queueing model is described in detail in the next chapter. In the example considered here, the network has five buffers at each node. The network access control limits the number of buffers available for external traffic at each node to three buffers. The window limit for each of the four virtual circuits is set at four. The analytical methods used

to solve this network are:

1] Complete solution of the network using the convolution algorithm (chapter 2);

2] The heuristic method based on the equivalent reduced network and the heuristic extensions to the MVA algorithm (section 3.4);

3] Further aggregation of the equivalent reduced network (section 3.5).

The results obtained by the heuristic methods are compared with those obtained by the exact method, in table 3.2 and 3.3.

Table 3.2 shows the network throughput and the virtual circuit throughput for representative values of offered load on each virtual circuit, when the window limit is four. Table 3.3 shows the mean network delay and the individual virtual circuit end-to-end delay. The first line for each offered load represents the throughput (in table 3.2) and delay (in table 3.3) obtained by the exact method using the convolution algorithm. The second line gives the corresponding performance when the network is solved using the equivalent reduced network and the heuristic extensions to the MVA algorithm. The third line gives the throughput and delay obtained after further aggregation of the equivalent reduced network. The last column in table 3.2 and 3.3 gives the percentage difference between the results

Table 3.1

Network and Virtual Circuit Throughput with K=4, NT=5 and NE=3

Offered load on each VC	Network throughput	Virtual Circuit Throughput				Percentage difference VC2	
		VC1	VC2	VC3	VC4		
2	7.742	1.913	1.962	1.921	1.946		a
	7.769	1.917	1.966	1.931	1.955	-0.52%	b
	7.747	1.914	1.962	1.923	1.948	-0.10%	c
6	14.674	3.087	4.264	3.188	4.136		
	15.048	3.274	4.285	3.346	4.143	-4.95%	
	15.094	3.284	4.291	3.367	4.152	-5.60%	
10	15.829	2.991	4.741	3.289	4.809		
	16.348	3.224	4.759	3.551	4.815	-7.96%	
	16.318	3.216	4.770	3.535	4.797	-7.47%	
20	16.124	2.804	4.838	3.331	5.150		
	16.664	3.033	4.850	3.623	5.159	-8.76%	
	16.227	2.942	4.738	3.497	5.050	-4.98%	

For each value of offered load:

a) Convolution algorithm method;

b) First heuristic method;

c) Second heuristic method.

Table 3.2

Network and Virtual Circuit Delay with K=4, NT=5 and NE=3

Offered load on each VC	Network delay	Virtual Circuit Delay				Percentage difference for VC2	
		VC1	VC2	VC3	VC4		
2	0.452	0.597	0.370	0.477	0.369		a
	0.446	0.586	0.369	0.464	0.369	2.70%	b
	0.447	0.587	0.370	0.464	0.369	2.70%	c
6	0.657	1.012	0.517	0.840	0.502		
	0.647	0.914	0.516	0.737	0.501	12.26%	
	0.644	0.909	0.513	0.729	0.500	13.26%	
10	0.788	1.198	0.610	0.975	0.582		
	0.752	1.093	0.610	0.866	0.582	11.17%	
	0.753	1.096	0.606	0.872	0.583	9.75%	
20	0.889	1.366	0.713	1.085	0.669		
	0.857	1.256	0.713	0.984	0.668	9.30%	
	0.878	1.296	0.729	1.016	0.681	6.35%	

For each value of offered load:

a) Convolution algorithm method;

b) First heuristic method;

c) Second heuristic methos.

obtained by the exact method and the heuristic methods, for virtual circuit VC3 (worst case virtual circuit). The tables show that, for this network, the results obtained by the exact method and the heuristic methods differ by less than 10% for throughputs and by less than 15% for delays, even at high loads. Further, the approximations introduced by aggregation of the equivalent reduced network do not significantly degrade the results; the results differ by less than 10% for throughputs and by less than 15% for delays.

The number of iterations required before the blocking probabilities converge, increases as the offered load increases and the nodal blocking probabilities tend towards unity. A convergence factor of 0.001 was chosen in all cases. The time required to evaluate the network performance (on the Harris 800 computer) when the offered load is 10 messages/second, is about 3600 seconds by the convolution algorithm method and about 70 seconds by the first heuristic method (i.e, the equivalent reduced network and the heuristic extensions to the MVA algorithm, developed in section 3.4). Assuming both programmes were equally efficient (or inefficient), the approximate method is nearly fifty times faster than the method that requires a complete solution. For this simple network with four virtual circuits, the second heuristic method based on the aggregation of the equivalent reduced network (section 3.5) also required about 70 seconds. This is because, the chains in the network are not concentrated. The approximate methods give accuracies

that are more than adequate, especially at the design phase of a network, and are fast enough to be suitable for interactive computing.

Networks with mesh topologies and multiple routing can deadlock. The model developed here ignores deadlocks in the network. Hence, it is strictly valid only when the network is deadlock free or when the mean time to deadlock is very large. It follows that the quantitative results generated by this method can only be interpreted correctly if the nodal blocking probabilities are small ($0 < 0.5$). For the window limits chosen in this example, the blocking probability for transit messages at the bottle neck node M2 is less than 0.5, even at saturation loads.

3.8. CONCLUSIONS

A queueing model for a store-and-forward computer communication network with finite buffers and equipped with multiple levels of flow control is formulated. The model is solved using the convolution algorithm (exact method), and by computationally efficient heuristic methods. The results obtained by the heuristic methods are in close agreement with those obtained by the exact method. The heuristic methods, based on the equivalent reduced network and the heuristic extensions to the MVA algorithm are fast, easy to implement and require low storage and computational effort. These techniques can be used to evaluate the performance of large networks with finite resources and many routing

chains. The closed queueing network model and its solution based on the heuristic methods give accuracies that are more than adequate for the potential application of the model, like the selection of window limits, selection of the number of buffers at each node, the choice of the network access control parameter and location of bottlenecks for a given traffic profile, etc.

CHAPTER FOUR

PERFORMANCE EVALUATION OF A FIVE NODE COMPUTER COMMUNICATION NETWORK

4. PERFORMANCE EVALUATION OF A FIVE NODE COMPUTER COMMUNICATION NETWORK

4.1. INTRODUCTION

Queueing network models play an important role in estimating the performance of computer communication systems. Queueing models can be very simple compared to the actual system, or they can be a very detailed representation of the actual system. Generally, the more detailed a model, the less manageable it becomes. Hence, the choice of an appropriate model involves a compromise between model complexity and computational efficiency.

Analytic models of computer networks can be evaluated numerically and the results compared with those obtained by simulation. Wong and Unsoy [90] studied a five node network with two groups of virtual circuits, and two levels of flow control. Georganas [20] studied a four node network with two virtual circuits, and three levels of flow control. The two network models were solved using the convolution algorithm. Reiser [70], used the heuristic extensions to the MVA algorithm to evaluate the performance of a seven node non-blocking network with twenty one virtual circuits. In this chapter, the performance of a five node network is evaluated numerically using the method based on the equivalent reduced network and the heuristic extensions to the MVA algorithm (section 3.4). The results of the analytical model are supported by a computer simulation of the

network.

The five node network under study has finite number of buffers at each node. Communication between the hosts is on a virtual circuit basis. Three levels of flow control are used in the network. Each virtual circuit is flow controlled by an end-to-end flow control mechanism that limits the number of messages that can be in transit within the network. Secondly, a network access control gives preference to transit traffic over external traffic entering the network, by limiting the number of buffers available for external traffic at the entry nodes. Finally, a hop level control ensures that transit messages are not lost due to nodal blocking. The hop level control retransmits transit messages until they are accepted by the downstream node. An investigation of the network performance as a function of network load, resource capacities and flow control parameters is made. The analytical and simulation models of the network are used to study the interaction between the three levels of flow control.

Fig. 4.1 shows the topology of the network and its closed queueing model. The network topology is similar to the one used by Wong and Unsoy [90]. The network has five nodes (M1 to M5). The number of buffers at each node is limited to five. The network access control parameter NE is varied from 1 to 5. There are four virtual circuits in the network (VC1 to VC4). The virtual circuits are controlled by

an end-to-end flow control mechanism that limits the number of messages in transit within the network. The virtual circuit window or token limit is varied from 1 to 6. The internode links, as well as the exit links are assumed to have a transmission capacity of 10 messages per second. A fixed routing is assumed for the virtual circuit messages. The route followed by the virtual circuits is also illustrated in fig. 4.1.

4.2. NUMERICAL RESULTS OF THE FIVE NODE NETWORK

The numerical evaluation of the model is carried out in two distinct stages. First, it is assumed that the network has local flow control only, consisting of hop level control, and hop level control in conjunction with network access control. The network is solved by an iterative procedure using the convolution algorithm [53,55]. Each node in the network is modelled as an open queueing network with different classes of messages, and the nodal blocking probability for transit and external messages are determined. These single node results are interfaced by invoking the principle of conservation of flows: for any service facility, total flow in must be equal to total flow out. In the second stage of evaluation, global flow control mechanisms are introduced through the end-to-end control of virtual circuits. Though the main objective of end-to-end control is to protect the exit nodes from congestion (because of slow sinks), an important byproduct is the prevention of

global (or internal) congestion. The closed network models are solved by the method of the equivalent reduced network and the heuristic extensions to the MVA algorithm. A Fortran programme for the performance evaluation of this network is given in Appendix B.

4.2.1. OPEN NETWORK EVALUATION

In the absence of the end-to-end control on virtual circuits, the network reduces to an open network having only the network access control and the hop level control. The network access control used in this model is based on local congestion and is similar to the input buffer limit control (IBL for short) studied by Lam and Reiser [55]. The model was evaluated by the open network method outlined above to give the network throughput and delay performance when the load on the virtual circuits are increased uniformly. Fig. 4.2 shows the network performance for various network access control buffer limits, and for the limiting case (NE=5), when no buffers are reserved for transit traffic messages. For the case without the network access control (i.e, NE=5), the network throughput initially increases with the offered load. However, when the offered load on each virtual circuit exceeds 6 messages per second, the network throughput collapses and falls to zero. The sudden fall in throughput is due to store-and-forward deadlock occurring in the network. Without the network access control, external traffic entering the network are allowed to occupy all the buffers

at the entry node, resulting in the blocking of transit traffic messages (as well as exit traffic) at these nodes. This results in a back-pressure effect and causes the deadlock to spread over the entire network, bringing the movement of all messages within the network to a stand still.

The introduction of a network access control that limits the number of buffers available for external traffic at the entry nodes (i.e NE < 5), prevents the network throughput from falling to zero even at high offered loads. The network access control prevents the external traffic from occupying all the buffers at the entry nodes, thereby preventing store-and-forward deadlock at the entry-exit nodes. As the network access control becomes tight, the virtual circuit throughputs at moderate loads are reduced because of the throttling effect of the control. But at higher loads, the network throughput improves and the transit delays are reduced.

The routing of the virtual circuits in this five node network is such that, the network access control can guarantee deadlock free operation. However, if virtual circuit VC4 were routed through node M2 (i.e, M5-M4-M2-M3), then, the network access control alone cannot prevent the deadlock that would occur between node M2 and M4. In this case all the buffers at node M2 and M4 can be occupied by transit messages belonging to virtual circuit VC1 and VC4 respec-

tively. Although the probability of deadlock can be significantly reduced with well designed input buffer limits, network access control based on input buffer limits cannot guarantee deadlock free operation in all networks [55]. Even if the network were free from deadlock, the network performance can deteriorate at saturation loads. This is because the control mechanism is based on local congestion at the entry node, rather than, on the global congestion of the network. While improving the local network performance, a local scheme cannot optimise the global network performance [21].

4.2.2. CLOSED NETWORK EVALUATION

Global flow control mechanisms control the input to the network well before the network is loaded to saturation. The control is achieved by limiting the total number of messages simultaneously existing in the network, to a value below the network capacity. End-to-end control of virtual circuits limits the number of messages in each virtual circuit. This in turn, limits the total number of messages in transit within the network. The network has been analysed using the heuristic extensions to the MVA algorithm. Fig. 4.3 shows the network throughput and delay for various window limits (K=1 to 6), when the network has ample buffers (i.e., there is no nodal blocking) and the virtual circuits are subjected to end-to-end control. The throughput of the network increases with the offered load initially, and then levels

off at higher levels of offered load, when the chains are saturated. Further, the throughput is higher, for larger window limits. The network throughput however does not collapse at high offered loads since the nodes are nonblocking, and there is no retransmission traffic in the network (i.e, the network resources are not wasted by frequent retransmissions).

Limiting the number of buffers at each node changes the network behaviour. Fig. 4.4 shows the network throughput and delay when the network is equipped with end-to-end control of virtual circuits, and a hop level control between nodes. The hop level control ensures that transit messages are not lost because of nodal blocking. Each node is assumed to have five buffers, and the external traffic entering the network has access to all the buffers at the entry node (i.e., no access control). For low window limits, K=1,2 and 3, the network throughput initially increases with the offered load. The throughput is always lower than the offered load, and it levels off as the offered load becomes high. The mean network delay is small and increases slowly with the offered load. In other words, the network is stable even at high levels of offered load. The end-to-end control drastically reduces the number of messages admitted to the network. The number of messages in transit within the network is small and do not cause the depletion of buffers. Messages once admitted to the network, are quickly carried across the network, because of the small queueing times at all nodes.

As the window limit is raised to four messages per virtual circuit, the network throughput initially increases with the offered load. The network has a good throughput for low and moderate loads. However as the offered load increases further, the network throughput begins to fall, with the maximum throughput being obtained at an offered load of 10 messages per second. At high offered loads the throughput is further reduced and is significantly less than that obtained with lower window limits (when K=3). The network delay also increases rapidly with the offered load. For larger window limits (greater than four), the network throughput falls rapidly and the network delays become very large. Simulation of the network shows that the network deadlocks when the window limit is five or more. All the buffers at the entry node are filled by external messages (since the number of buffers at each node is now less than or equal to the window limit), and no buffers are left for transit and exit messages. This brings the flow of messages within the network to a stand still.

The offered load at which the network performance collapses, decreases as the window limit is increased. As the window limit is raised, more messages are admitted to the network. This results in the depletion of buffers leading to nodal blocking and retransmissions, and fewer messages succeed in getting across the network. In fact, the network behaves like an open network when the window limit is large. The introduction of the end-to-end control of virtual

circuits in a network with finite buffer resources can prevent network collapse, if the window limits are chosen correctly. If the window limits are large, the control is ineffective and the network performance degrades at high offered loads. If the window limits are too small, the throughputs are unnecessarily throttled because of the tight control. For the network under study (i.e, without the network access control), a window limit of three tokens per virtual circuit is optimum.

Introduction of a network access control in addition to the end-to-end control and hop level control can improve the network performance further. Fig. 4.5 shows the network throughput and delay with the introduction of the network access control, that limits the number of buffers available to external traffic at each node to three buffers (NE=3). The new control has hardly any effect on the network performance when the window limit is less than four. However for larger window limits, the network is able to withstand higher loads, and there is a considerable reduction in the network delay. The throughput curves have a broader peak and the delay curves are flatter. Hence, the network access control can improve the performance of a network equipped with imperfect end-to-end control.

Fig. 4.6 shows the effect of network access control on the network performance, when the window limit is five (K=5). In the absence of the network access control (NE=5),

the virtual circuit throughput and delay deteriorate at high offered loads. Simulation of the network shows that the network deadlocks when the network access control is absent. With the introduction of the network access control that limits the number of buffers available to external traffic to three (NE=3), the virtual circuit performance improves. Simulation shows that the network does not deadlock with the network access control.

The effect of the network access control is more pronounced when the virtual circuit window limit is six (K=6). In the absence of the network access control (i.e, NT=5 and NE=5), the throughput of all four virtual circuits deteriorate at high offered loads (fig. 4.7). The delay on the virtual circuits increase rapidly with the offered load. Once again, simulation shows that the network deadlocks. With the introduction of the network access control, there is a marked improvement in the throughput and delay performance of all the virtual circuits (Fig. 4.8 - 4.11). As the input buffer limit is reduced (from NE=4 to NE=1), the virtual circuit throughputs at low and moderate loads reduce. This is because of the the throttling effect of the network access control mechanism. However, at high offered loads, higher virtual circuit throughputs and lower transit delays are obtained. The virtual circuit throughputs have a broader peak and they fall off less sharply, for lower buffer limits. The network can withstand a larger offered load. The delay suffered by the virtual circuits are

dramatically lowered. The best network throughput and delay performance are obtained when the input buffer limit NE is one. Node M2 (fig. 4.1) is a bottleneck node. Without the network access control, the bottleneck first originates at node M2 and spreads to the other four nodes in the network. The network access control contains the bottleneck to node M2.

Fig. 4.12 shows the blocking probabilities of node M2 as a function of offered load when the virtual circuit window limit is five and the network access control parameter is varied from NE=1 to NE=5. The transit blocking probability at moderate loads (10 messages/second) is reduced by as much as 50%, when the input buffer limit is reduced from NE=5 to NE=1.

The network performance shows a marked improvement, when the number of buffers at each node is increased from five to six (fig. 4.13 and 4.14). The window limit is set at five (K=5) and the input buffer limit is varied from NE=1 to NE=5 (NE=6). With the window limit set at five and the number of buffers at each node equal to six (NT=6), the network becomes deadlock free even with out the network access control. Since, the number of external messages at any entry node is less than the total number of buffers at the node, direct store-and-forward deadlock is avoided. However, the network performance at high offered loads is better with the network access control, than without it.

While the network throughput improves marginally, the network delay shows some reduction with the network access control. The best network throughput and delay performance at high loads are obtained when the input buffer limit NE=1 for NT=5, and NE=1 or 2 for NT=6.

Fig. 4.15 shows the network performance, when end-to-end acknowledgements are included. The window limit is set at five (K=5), the network has five buffers at each node, and the input buffer limit is set at three (NE=3). The mean length of an ack message is assumed to be 0.05 times that of a data message. The effect of the ack delay on the virtual circuit performance, is to reduce its effective window limit. The virtual circuit throughput and delay are lower, when the ack delays are included.

For a given set of virtual circuit window limits and offered load, the introduction of the network access control results in fewer messages in transit within the network. This is because, the transit blocking probabilities are reduced, and messages once admitted to the network, are quickly carried across it. On the other hand, the end-to-end control of virtual circuits limits the total number of messages simultaneously existing in the network. This reduces the contention for resources within the network. A tight network access control reduces the network throughput at low loads. However at high loads the network performance improves as the number of buffers available for external

messages is reduced. The network performance at high loads is best when the input buffer limit NE=1 for NT=5, and NE=1 or 2 for NT=6. Lam and Reiser [55] have shown that in a symmetrical homogeneous network, a good heuristic choice for the optimum input buffer limit (to maximise the network throughput at high loads) is NE=NT/(1+H), where H is the mean hop length. In this five node network the mean hop length is two, and the optimum value of the input buffer limit NE, when NT=5, is 1.6 buffers (and NE=2 when NT=6). The analytical results show that the network performance at high loads is optimum when NE=1 for NT=5 (and NE=1 or 2 for NT=6).

4.3. SIMULATION OF THE FIVE NODE NETWORK

The five node network was simulated to validate the analytical model developed in chapter 2 and 3. This section presents the simulation results of the five node network and compares them with the results obtained by the analytical methods.

4.3.1. SIMULATION FOR SYSTEM MODELING

The most popular approach to the solution of a computer communication system model is to simulate it on a computer. The principal advantage of simulation is its generality. Simulation has two main purposes:

1] To provide a mechanism for the systematic study of models that do not have an analytical solution and

where experiments on the actual system is impractical.

2] To provide a reference for the validation of analytical models, especially when the model is based on a number of simplifying assumptions and approximations.

The models proposed in this thesis have been built using a number of assumptions, and their solution involves further approximations. The first set of assumptions were those involved in deriving for the system, a closed queueing network model with product form solution (chapter 2). The main assumptions were that messages arrived from Poisson sources (i.e., the time between arrival of messages is exponentially distributed), message lengths were drawn from a negative exponential distribution each time they were transmitted, the scheduling at each queue was of the FIFO type, and the effect of nodal blocking due to finite number of buffers was taken into account by modifying the service rate of the queues. These assumptions are necessary for a product form solution so that an analytical solution can in principle be obtained by the convolution or the MVA algorithm.

The second set of approximations were introduced to find an efficient method for solving the closed queueing network model (chapter 3). They were introduced in the heuristic extensions to the MVA algorithm and in the aggregation approximation of the closed queueing network model (i.e., the equivalent reduced network). Though approximate

techniques have a low computational cost, they introduce an unknown amount of error.

The analytical model of the computer communication network developed in chapter 2 and 3 was used to generate the numerical results of section 4.2. In this network model, nodal blocking resulting from the finite number of buffers at each node makes the network model intractable since the global network of queues does not have a product form solution. The problem was solved by making two assumptions. First, an equilibrium blocking probability was postulated for each node for a given set of arrival rates and chain population. The equilibrium nodal blocking probability is therefore static and state independent. Secondly, all the finite length queues were replaced by unbounded queues, and the blocking probabilities were replaced by overflow probabilities. The effect of blocking at the downstream node was reflected on the upstream service centres by modifying their service rates. This slows down the queues in the network, but is only an approximate representation of the actual blocking process in the network. In order to reduce the computational effort, the model was solved through the equivalent reduced network and the heuristic extensions to the MVA algorithm. The simulation experiment is used to verify these approximations and validate the analytical model.

4.3.2. FORMULATION OF THE SIMULATION MODEL

TIMING CONTROL

The five node network is simulated by a discrete event self-driven simulator [49,80]. The simulation programme is written in Fortran 77 and run on the Harris 800 computer. The simulation is event-driven in simulated time. The driving source for the model is a set of simulated independent Poisson sources which generate the external messages and constitute the load on the network, and a set of simulated independent service centres which service messages queued at these centres. The simulation model provides a timing mechanism such that the simulator clock is driven forward from the time of occurrence of an event to the time of occurrence of the next chronological event. The simulator clock therefore takes up the value of the time attributes of a set of discrete events.

The simulation model identifies significant events in the system. Two types of events are identified. The first type of event consists of the arrival of messages from their respective Poisson sources. The second type of event occurs on the completion of the service of a message at a service centre. Each event is characterised by its type and its time of occurrence. Both types of events are associated with a list of actions, which the event invokes. The events are ordered by their time of occurrence and assembled into an event list.

Conceptually, the timing mechanism keeps the system running until an event occurs, at which point the simulator momentarily pauses to record the change in the system. To implement this concept, the simulator proceeds by keeping track of the event list. The simulator programme has a single scheduler which scans through the event list, selects the event with the the earliest event time, updates the simulated clock time, and performs the actions invoked by the event.

RANDOM NUMBER GENERATION

The simulation model assumes that the time between arrival of messages from their respective sources is exponentially distributed, and the service time of the messages at each queue is also exponentially distributed. Each Poisson message source and each of the queues have an independent pseudo random number generator. In this simulation experiment the random number generators available on the Harris 800 were used. The random number generators which are initialised with different seed values, produce uniform random variables in the interval (0,1). Consider for example the probability of a message arriving from its source in a time $t \leqslant T$:

$$P(t \leqslant T) = 1 - \exp(-A\ T).$$

If the random number generator produces a number Rn in the interval (0,1), and if

$$Rn = 1 - \exp(-A\ T),$$

then

$$T = - (1/A)\ \ln(1-Rn).$$

Since (1-Rn) is itself a random number in the interval (0,1), the simpler transformation

$$T = -(1/A)\ \ln(Rn)$$

is used. Applying the above transformation to the random sequences of each generator, a sequence of random observation points are generated.

EVENT HANDLING ROUTINE

The event handling routine performs all the necessary actions when an event occurs. In the case of events arising from messages arriving from external Poisson sources, the messages are admitted to the network if a free token and a free buffer (as dictated by the network access control) are available at the entry node. Otherwise the event is treated as a lost arrival. In either case, a new time is selected for the next arrival and entered in the event list. In the case of events arising from the completion of a message service at a service centre, the message is accepted by the

next node on its route if a free buffer is available at this node. The link level protocol between nodes is similar to the Send and Wait protocol [18,88], where a NACK message is assumed to arrive instantaneously if the downstream node is blocked. If a message is blocked, it is retransmitted and a new time for the completion of its service is selected and entered in the event list. The message is retransmitted until it is accepted by the down stream node.

Each queue is organised as a FIFO scheduling discipline. The message at the head of the queue is serviced and is removed from the queue only when it is admitted to the down stream node. The service times are drawn from an independent negative exponential distribution that is common to all messages passing through the service centre. Once a message completes service at the exit queue, the token associated with the message is returned instantaneously to its token queue. The simulation model ignores acknowledgement traffic and delays.

BASIC PERFORMANCE STATISTICS

System related performance statistics like message arrival rate, virtual circuit throughput, end-to-end delay, network throughput and delay, mean number of messages in transit and the nodal blocking probabilities are collected by invoking a statistics gathering routine each time the system changes state (i.e., when an event occurs). The simulator proceeds by keeping track of the events in the

event list. System changes occur when an event takes place. The system state is represented by a collection of related objects described by a set of data structures like the event list, and the list for each queue, node and virtual circuit in the model. The data structure for each queue consists of a list, the elements of the list representing the messages in the queue. The elements contain information about the messages like their virtual circuit number and their current position in the queue. The data structure for the nodes consist of counters to keep track of the number of transit and external messages at the node, and counters to count the number of arrivals and the number of messages rejected by the node. The data structure for the virtual circuits consist of counters to count the number of arrivals from their respective Poisson sources, number of messages accepted by the virtual circuits and the number of free token currently available.

In this experiment, only the mean values are considered (rather than probability distribution of queue size and response time). The simulator simply counts cumulative sums of observed quantities like message arrivals on each virtual circuit, number of messages accepted by each virtual circuit, etc. After the simulation run, the sample means are obtained from these cumulative sums. The number of message arrivals from each Poisson source and the number of messages accepted on each virtual circuit, are counted during the simulation run. The mean arrival rate and the mean

throughput of each virtual circuit is estimated by dividing these cumulative sums by the simulation time. The mean queue length and the mean number of messages belonging to each virtual circuit that are in transit, are estimated as follows. Each time the queue length changes, the time is recorded as a reference. The product of the previous queue length and the period for which this queue length was held, is added to form a cumulative queue length/time product. The cumulative sum divided by the simulation time, gives an estimate of the mean queue length. The mean number of virtual circuit messages in transit is found by subtracting the mean queue length of the token queue, from its corresponding window limit. The mean queueing time and the mean end-to-end delay for each virtual circuit are estimated by applying Little's theorem [58]. The mean queueing time of a queue (and thus, the mean end-to-end delay on a virtual circuit) is found by dividing the mean queue length (mean number of virtual circuit messages in transit), by the mean queue throughput (virtual circuit throughput). Nodal blocking probabilities are given by the ratio of the number of messages rejected by the node to the number of messages arriving at the node.

The objective of this simulation study is to investigate the behaviour of the simulated network in its steady state operating condition. Since the behaviour of the simulation model will represent the typical behaviour of the simulated system only when the model reaches a steady state

condition, the simulation experiment is preceded by an initial stabilisation run. Further, the results of a single experiment may or may not be close to the desired measure, and hence, the experiments are replicated. That is, the simulation experiment is run many times using different starting conditions. The average of the experimental results is taken as the final estimate.

In this simulation experiment, the simulation runs were made consecutively, using the final condition of one run as the steady state starting condition for the next run. The first run was preceded by an initial stabilisation run. The number of runs and the length of each run were chosen such that, there is 95% confidence that the confidence interval is less than 5% of the mean estimates. In this experiment, ten runs were used and the length of each run was 20000 seconds in simulated time.

4.4. SIMULATION RESULTS

The network was simulated for window limits of 4,5 and 6, over a wide range of offered loads. The number of buffers at each node was limited to five, and the effect of the network access control on the performance of the network was investigated. The simulation results are compared with those of the analytical model.

When each virtual circuit has a window limit of four (K=4), the network is found to be stable at high offered

loads (50 messages per second), even without the network access control (fig. 4.16). At high offered loads the virtual circuit throughput and delay remain fairly constant. The introduction of the network access control with an input buffer limit of three (NE=3), has only a marginal effect on the network performance (fig. 4.17). At low loads there is a slight reduction in throughput, but at higher levels of offered load there is a small improvement in the network throughput and delay performance. The simulation results and the results of the analytical model differ by less than 10% over the entire range of offered load.

The network deadlocks without the network access control, when the window limit is five (K=5). The deadlock occurred when the mean arrival rate on each virtual circuit exceeds two messages per second. The deadlock originates as a direct store-and-forward dead lock between node M4 and M5. In a typical simulation run when deadlock occurred, all the five buffers at node M5 were occupied by virtual circuit VC4 messages. The five buffers at node M4 were occupied by two virtual circuit VC3 messages and three virtual circuit VC1 messages, leading to a store-and-forward deadlock between node M4 and M5. This brought to a stand still the movement of all virtual circuit VC1, VC3 and VC4 messages. Node M2 had three virtual circuit VC3 messages and two virtual circuit VC1 messages and hence, the node was blocked to virtual circuit VC2 messages. The mean time to deadlock depends on the offered load, and it decreases as the offered load

increases.

The introduction of a network access control (NE < 5) partitions the buffers at the entry node. A direct store-and-forward deadlock between node M4 and M5 is avoided, and virtual circuit VC1 and VC3 messages can successfully leave the network at node M5. Simulation of the network with an input buffer limit of three (fig. 4.18), shows that the network does not deadlock even at high loads, though the performance of the network degrades. At high offered loads (above 10 messages per second) the throughput begins to fall and the end-to-end delay begins to rise. The simulation results and the results of the analytical model differ by less than 12% for loads less than 10 messages per second.The transit blocking probability at the bottle neck node M2 for this load is less than 0.5. However, at high loads (40 messages/second), the throughput and delay (of virtual circuits VC3 and VC4 in particular) differ by as much as 20% and 35% respectively.

When the window limit is set at six (K=6), the network without the network access control deadlocks even when the arrival rate is as low as one message per second. Once again the deadlock originates as a direct store-and-forward deadlock between node M4 and M5. A network access control with an input buffer limit less than five, eliminates the deadlock completely. With an input buffer limit of four (fig. 4.19), the blocking probabilities for transit messages

are high, because the buffers at the entry nodes are monopolised by the traffic entering the network at these nodes. In particular, node M2 has a high probability of blocking leading to reduced throughout on virtual circuits VC1 and VC2. Hence, even though the the network does not deadlock, its performance degrades at high loads. When the input buffer limit is lowered further (NE=2), the transit blocking probabilities are low even at high loads, leading to higher throughput and lower delays on all four virtual circuits (fig. 4.20). As the network access control is made tight, it lowers the throughput at low and moderate loads, but gives a higher throughput at high loads without deadlock setting in. The transit delay also shows a considerable reduction. The simulation results and the analytical results are in some agreement (difference less than 15%) only when the offered load is such that the virtual circuit throughputs have not reached their peak.

For a given set of virtual circuit window limit, both the analytical and simulation results show that the introduction of the network access control results in fewer messages in transit within the network. The results also show that the window limit affects the maximum throughput. For example, with an input buffer limit of three, the network throughput is highest and the transit delay is lowest with a window limit of four. These measures deteriorate as the window limit is increased to five, and then to six. Both the simulation and the analytical models show that the mean

number of messages in transit is about fourteen, when the network throughput reaches a maximum. The simulation also confirms that node M2 is a bottleneck node, and a tight network access control helps to contain it to node M2.

It follows that the network access control prevents deadlock in this network and reduces the blocking probability for transit messages. Lower transit blocking probabilities lead to fewer retransmissions between the nodes, and messages once admitted to the network, spend less time in it. The end-to-end control helps to limit the total number of messages simultaneously existing in the network, thereby reducing the contention for network resources. Further, as the network becomes congested, the token pool becomes depleted. This in turn, limits the number of messages admitted to the network.

4.5. COMPARISON OF ANALYTIC AND SIMULATION RESULTS

In this section the results of the simulation model are compared with those obtained by the analytical methods. The analytical methods used to solve the network are:

1] The heuristic method based on the equivalent reduced network and the heuristic extensions to the MVA algorithm (chapter 3, section 3.4);

2] Further aggregation of the equivalent reduced network (chapter 3, section 3.5);

3] Complete solution of the network using the convolution algorithm method (chapter 2).

The results obtained by the three analytical methods are compared with those obtained by the simulation of the network, in table 4.1 and 4.2.

The window limit of each virtual circuit is set at four, and the input buffer limit is set at three. Table 4.1 shows the network throughput and the virtual circuit throughput for representative values of offered load on each virtual circuit. Table 4.2 shows the mean network delay and the individual virtual circuit end-to-end delay. The last column in table 4.1 and 4.2 gives the percentage difference between the results obtained by the simulation model and the analytic methods, for virtual circuit VC2. The two tables show that the simulation results and the results of the three analytical methods differ by less than 10% even at high loads. Hence, the results of the simulation model and the analytical models are in good agreement. For the flow control parameters chosen in this example, the blocking probability at the bottle neck node M2 is less than 0.5 at saturation loads.

Table 4.3 and 4.4 show the throughput and delay performance of the network, when the window limit is raised to five. The table compares the results obtained by the method based on the equivalent reduced network (chapter 3, section 3.4), with those obtained by simulation. The results show

Table 4.1

Network and Virtual Circuit Throughput with K=4, NT=5 and NE=3

Offered load on each VC	Network throughput	Virtual Circuit Throughput				Percentage difference VC2	
		VC1	VC2	VC3	VC4		
2	7.778	1.932	1.969	1.930	1.947		a
	7.769	1.917	1.966	1.931	1.955	0.15%	b
	7.747	1.914	1.962	1.923	1.948	0.36%	c
	7.742	1.913	1.962	1.921	1.946	0.36%	d
6	15.629	3.309	4.496	3.559	4.265		
	15.048	3.274	4.285	3.346	4.143	4.69%	
	15.094	3.284	4.291	3.367	4.152	4.56%	
	14.674	3.087	4.264	3.188	4.136	5.16%	
10	17.113	3.336	5.117	3.687	4.973		
	16.348	3.224	4.759	3.551	4.815	7.00%	
	16.318	3.216	4.770	3.535	4.797	6.78%	
	15.829	2.991	4.741	3.289	4.809	7.35%	
20	17.454	3.194	5.196	3.747	5.317		
	16.664	3.033	4.850	3.623	5.159	6.66%	
	16.227	2.942	4.738	3.497	5.050	8.81%	
	16.124	2.804	4.838	3.331	5.150	6.88%	

For each value of offered load:

a) Simulation method;

b) First heuristic method;

c) Second heuristic method;

d) Convolution algorithm method.

Table 4.2

Network and Virtual Circuit Delay with
K=4, NT=5 and NE=3

Offered load on each VC	Network delay	Virtual Circuit Delay				Percentage difference for VC2	
		VC1	VC2	VC3	VC4		
2	0.450	0.593	0.369	0.473	0.369		a
	0.446	0.586	0.369	0.464	0.369	0.00%	b
	0.447	0.587	0.370	0.464	0.369	-0.27%	c
	0.452	0.597	0.370	0.477	0.369	-0.27%	d
6	0.650	0.930	0.505	0.757	0.499		
	0.647	0.914	0.516	0.737	0.501	-2.17%	
	0.644	0.909	0.513	0.729	0.500	-1.58%	
	0.657	1.012	0.517	0.840	0.502	-2.38%	
10	0.733	1.064	0.581	0.866	0.568		
	0.752	1.093	0.610	0.866	0.582	-5.00%	
	0.753	1.096	0.606	0.872	0.583	-4.30%	
	0.788	1.198	0.610	0.975	0.582	-5.00%	
20	0.812	1.193	0.664	0.944	0.636		
	0.857	1.256	0.713	0.984	0.668	-7.37%	
	0.878	1.296	0.729	1.016	0.681	-9.79%	
	0.889	1.366	0.713	1.085	0.669	-7.37%	

For each value of offered load:

a) Simulation method;

b) First heuristic method;

c) Second heuristic method;

d) Convolution algorithm method.

that at moderate levels of offered load (10 messages per second), when the blocking probabilities are below 0.5, the difference between the results of the simulation model and the analytical method is around 10% for throughput and delay estimates. For higher arrival rates beyond the point of maximum throughput, the maximum difference in throughput is around 20% and the difference in delay estimates are around 35%. In this region, the transit blocking probabilities are high (> 0.5). For transit blocking probabilities less than 0.1, the results differ by less than 3%. The analytical model ignores deadlocks in the network. These results show that for small blocking probabilities (< 0.5), the closed queueing network model and its heuristic solution based on the equivalent reduced network are in good agreement with those obtained by simulation, if the network is deadlock free.

4.6. CONCLUSIONS

The results generated from the analytical and simulation models show that the performance of the network with finite nodal buffers deteriorates if proper control is not exercised on the traffic entering the network. Networks having a mesh topology, with multiple routing chains run the risk of store-and-forward deadlock. The risk of deadlock can be decreased, if not eliminated, by careful partitioning of the buffer pool at each node. Even when the network is free from deadlock, its performance can deteriorate at

Table 4.3

Virtual Circuit Throughput with K=5, NT=5 and NE=3

Offered load on each VC	Virtual Circuit Throughput				Blocking probability of M2
	VC1	VC2	VC3	VC4	
2	1.973 1.957 0.82%	1.984 1.977 0.35%	1.951 1.941 0.51%	1.958 1.955 0.15%	a 0.0132 b
4	3.255 3.132 0.37%	3.662 3.546 3.16%	3.203 2.950 7.81%	3.446 3.369 2.23%	0.3112
10	3.357 3.049 9.17%	4.574 4.258 6.90%	3.529 3.264 7.51%	5.022 4.744 5.53%	0.4601
20	3.006 2.690 10.51%	3.792 3.855 1.66%	3.594 3.241 9.82%	5.457 4.776 12.48%	0.5767
40	2.753 2.475 10.09%	3.260 3.424 5.03%	3.645 3.213 11.85%	5.549 4.498 18.94%	0.6428

For each value of offered load:

a) Simulation method;

b) First heuristic method.

Table 4.4

Virtual Circuit Delay with K=5, NT=5 and NE=3

Offered load on each VC	Virtual Circuit Delay				
	VC1	VC2	VC3	VC4	
2	0.606	0.373	0.482	0.372	a
	0.611	0.374	0.486	0.373	b
	-0.82%	-0.27%	-0.83%	-0.21%	
4	0.889	0.472	0.726	0.464	
	0.911	0.488	0.737	0.472	
	-2.47%	-3.38%	-1.51%	-1.72%	
10	1.338	0.748	1.023	0.664	
	1.476	0.800	1.138	0.721	
	-10.31%	-6.95%	-11.24%	-8.58%	
20	1.587	0.973	1.094	0.746	
	1.780	1.062	1.342	0.914	
	-12.16%	-9.16%	-22.67%	-22.50%	
40	1.763	1.136	1.136	0.781	
	1.979	1.305	1.451	1.059	
	-12.25%	14.88%	-21.7%	-35.6%	

For each value of offered load:

a) Simulation method;

b) First heuristic method.

saturation loads. That is, the throughput at very high input loads will decrease and may only stabilise at a very low level. Hence, flow control mechanisms are needed to prevent such throughput degradation, whether or not the network is proved to be deadlock free.

If a network has only a hop level control, the introduction of a network access control that gives preference to transit traffic over external traffic, can improve its performance to some extent. Though the network can withstand higher loads with the network access control, its performance can still deteriorate at high loads, even if it is free from deadlock. This is due to the fact that the network access control mechanism is based on local congestion at the entry node, and not on the global state of the network. The network access control based on input buffer limits can improve the network performance marginally, but cannot offer protection against global congestion and deadlocks.

Messages in transit within any network, can be in one of two states. They are either not blocked (i.e. their resource requirements are met), or they are blocked (their resource requirements are not met). The number of non-blocked messages existing in the network at any time depends on the nodal buffer capacities, the channel capacities, the number of messages in transit within the network and the distribution of these messages over the network. The last

two factors depend on the network load, the routing of messages and the flow control mechanisms in the network. The network throughput is proportional to the mean number of non-blocked messages in the network. At high offered loads, if the number of messages admitted to the network is large, many messages compete for the limited number of resources available in the network. The distribution of the messages and the resources required by them, over the network of queues is such that few messages have their resource requirements met. This leads to fewer non-blocked messages, resulting in lower throughput, and if deadlock occurs the throughput becomes zero.

Such degradation of network performance, can be prevented by limiting the total number of messages in transit within the network, below the network saturation limit. Introduction of an end-to-end control that limits the number of messages in each virtual circuit, can prevent global congestion if the window limits are chosen correctly. With a small window limit (tight control) congestion can be prevented, but the throughput also reduces because of the throttling effect of the control. With large window limits, the control is less effective and the network performance degrades at high loads.

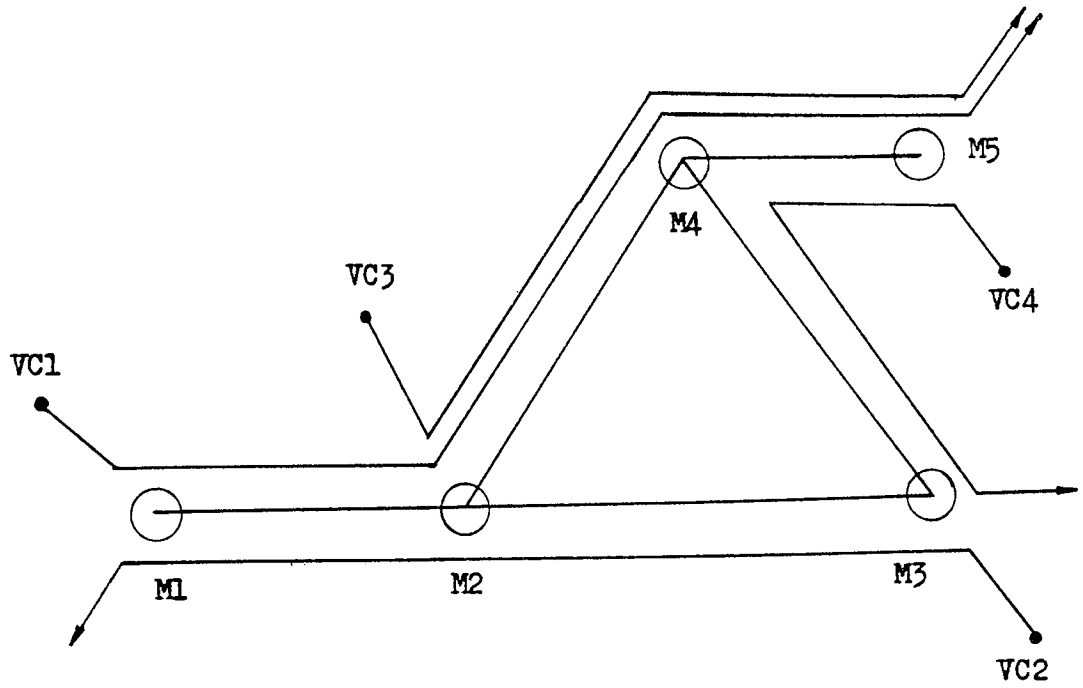
Hence, end-to-end control can provide protection against congestion, and at the same time, maintain high throughput efficiency if the window limits are chosen

correctly. However, the optimum window limit depends on the number of active virtual circuits, their routing and the number of buffers at each node. End-to-end control of virtual circuits can protect the network when the load on individual virtual circuits become large. However, they provide little protection to the network when the overloading is from an increase in the number of active virtual circuits. The number of active virtual circuits and their distribution are not easy to control, since they are established and terminated by individual node pairs (i.e., they are user dependent). End-to-end control under such circumstances can only be effective, if the window limits are adaptively controlled, based on the global state of the network. Such an implementation requires a central control or a distributed control that will coordinate the actions of all the nodes.

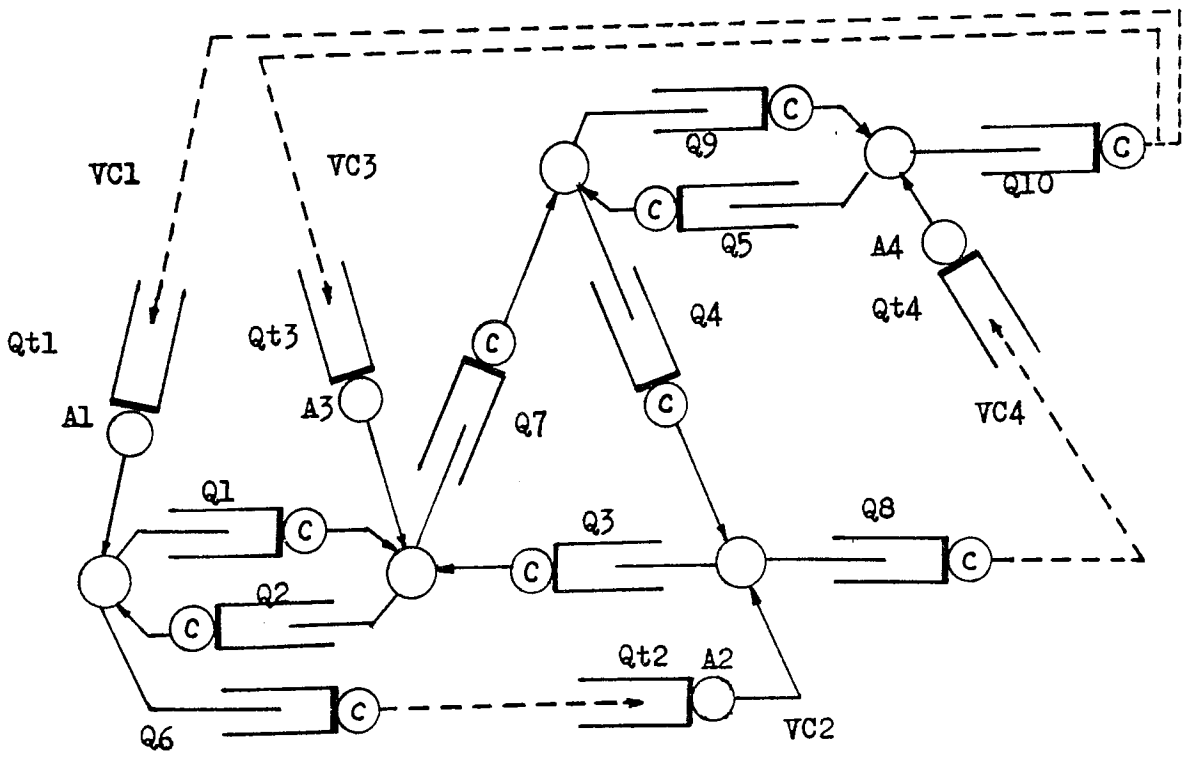
Even if the optimal window limits are chosen, the network can still deadlock if the buffers are not partitioned carefully. The addition of a network access control with correctly chosen buffer limits can help to reduce the risk of deadlock. It can also improve the performance of the network at high offered loads, even if the window limits are large. This is because, the network access control reduces the blocking probability for transit messages at nodes where there is a large influx of external messages. This in turn reduces the retransmission traffic in the network and ensures that the network resources are not wasted. Hence, in the event of overloads due to improper choice of window

limits, or due to too many virtual circuits being active in the network, the network access control can help to prevent performance degradation of the network. A tight network access control reduces the network throughput at low loads. However, at high loads the network performance improves (even when the window limits are large), as the number of buffers made available for external messages is reduced. For the five node network investigated in this chapter, the optimum input buffer limit (that maximises the network throughput at high loads) is in agreement with the heuristic choice suggested by Lam and Reiser [55] for symmetrical homogeneous networks.

The simulation of the five node network also validates the closed queueing network model (chapter 2), and its heuristic solutions developed in chapter 3. The closed queueing network model and its solution based on the equivalent reduced network and the heuristic extensions to the MVA algorithm give accuracies that are more than adequate, when the network is deadlock free and the blocking probabilities are small.



(a)



(b)

Fig. 4.1 (a) Topology of the five node network, (b) and its queueing model with token queues.

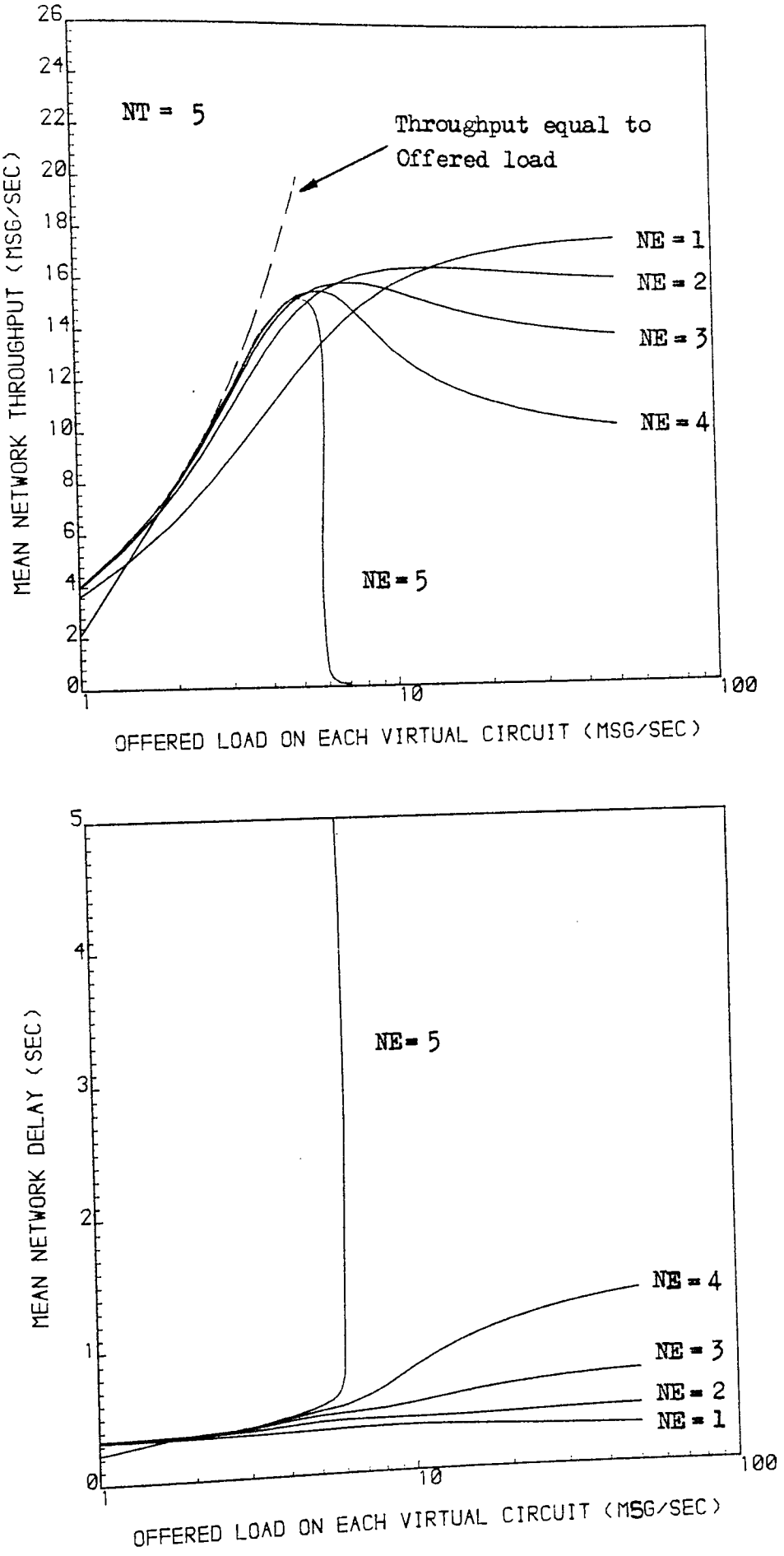
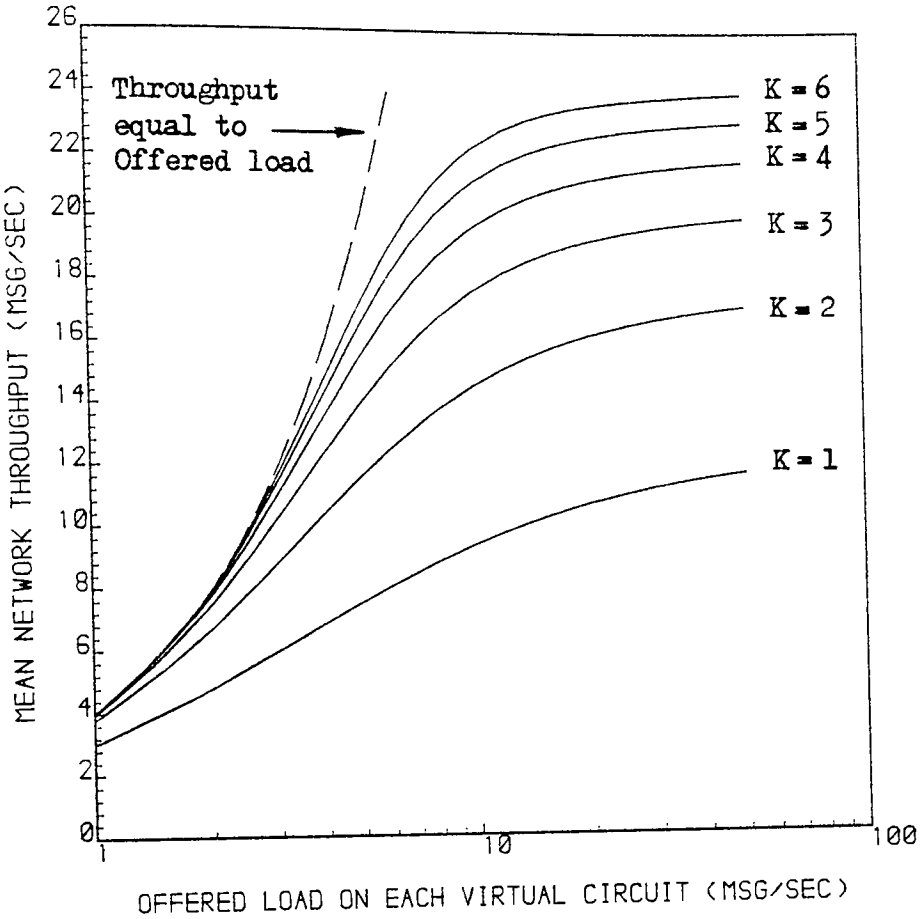


Fig. 4.2 Network throughput and delay performance without end-to-end control (NT=5).



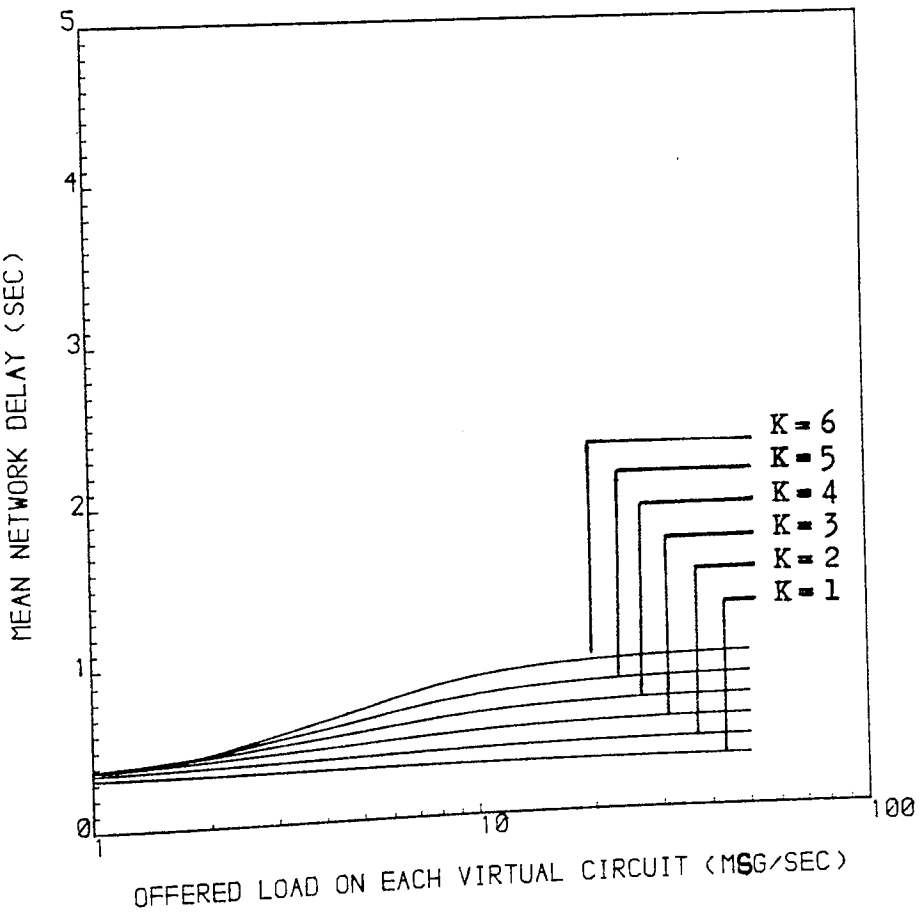
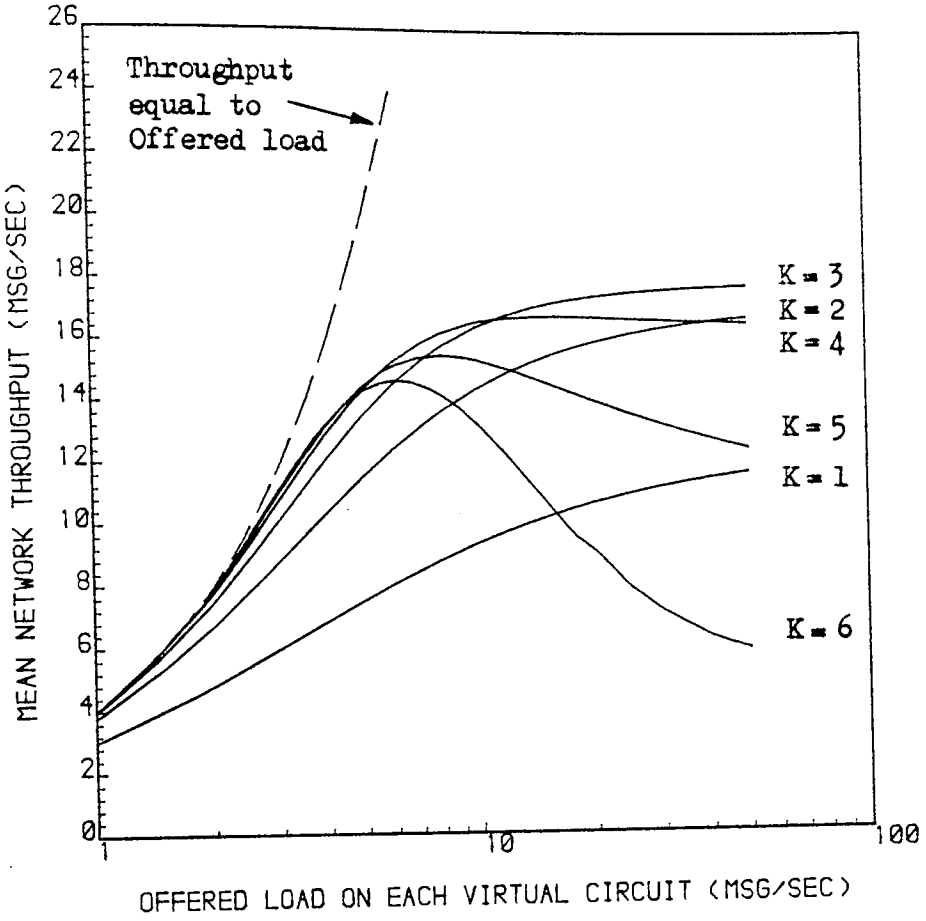


Fig. 4.3 Network performance with end-to-end control, but without nodal blocking.



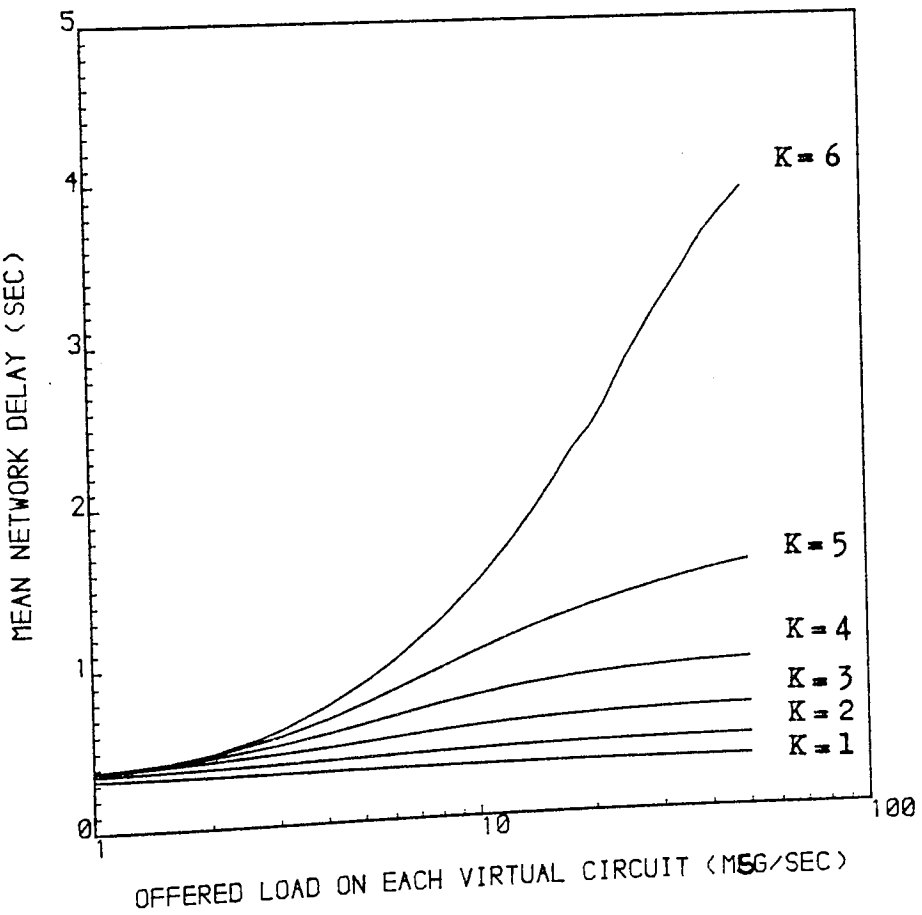
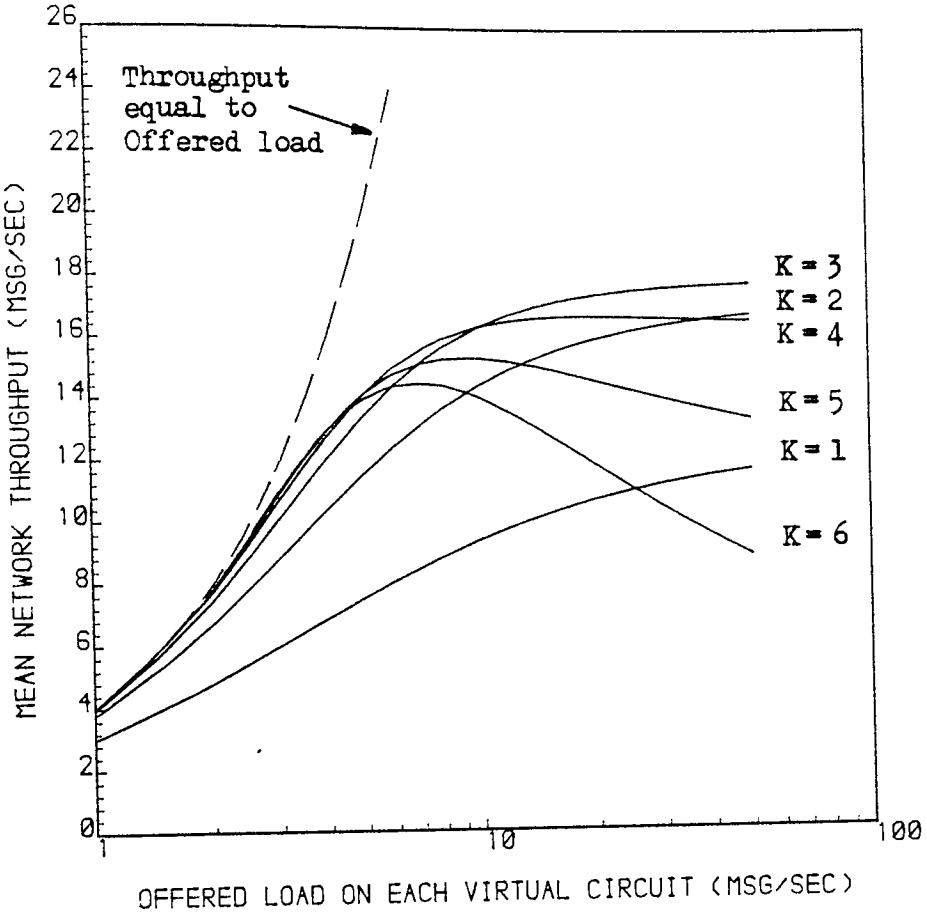


Fig. 4.4 Network performance with nodal blocking and end-to-end control, but without network access control (NT = 5, NE = 5).



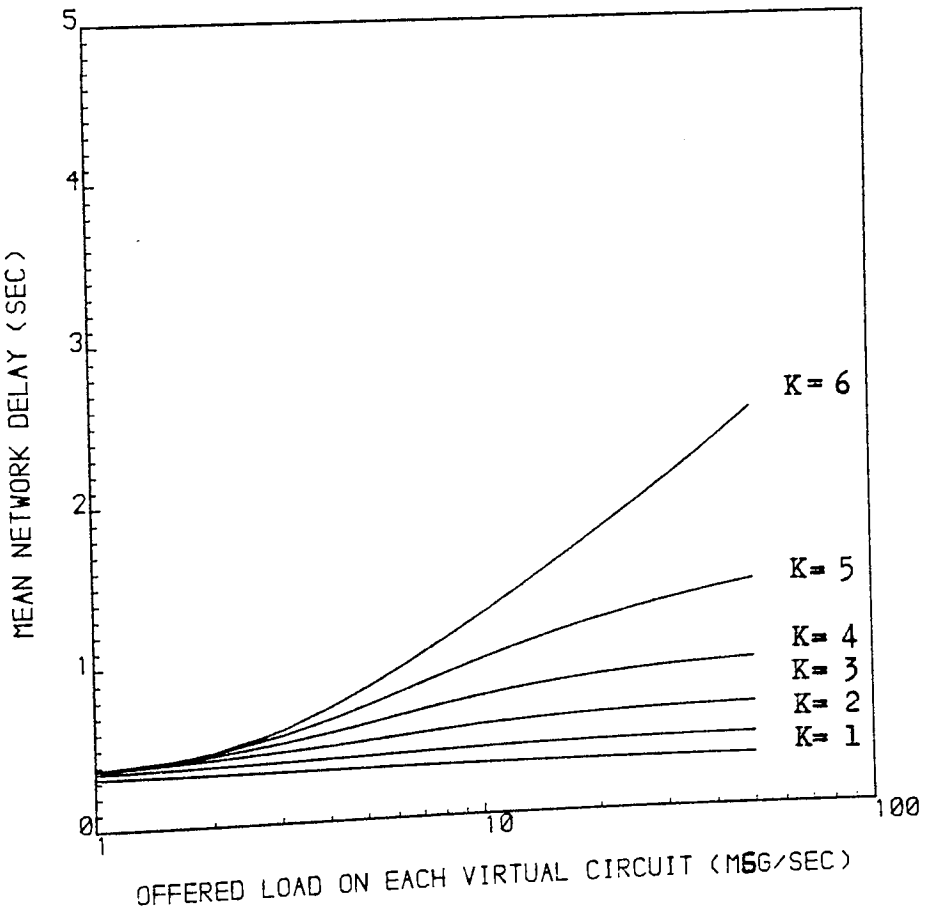
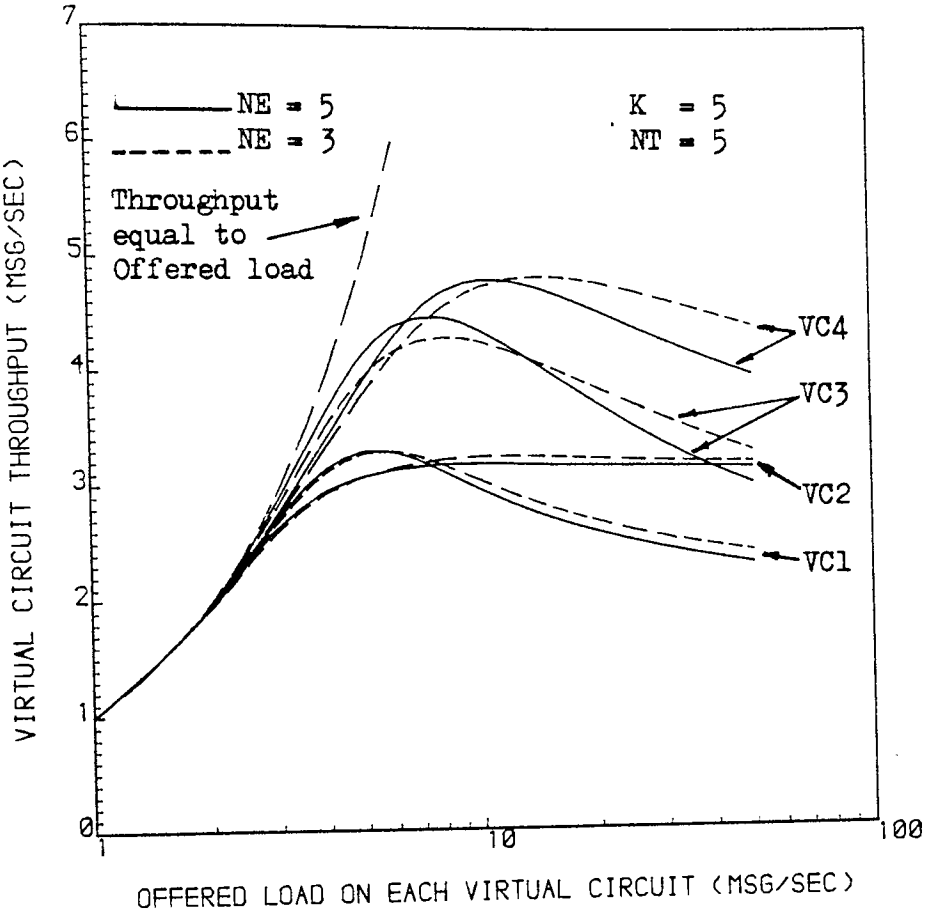


Fig. 4.5 Network performance with end-to-end control and network access control (NT = 5 and NE = 3).



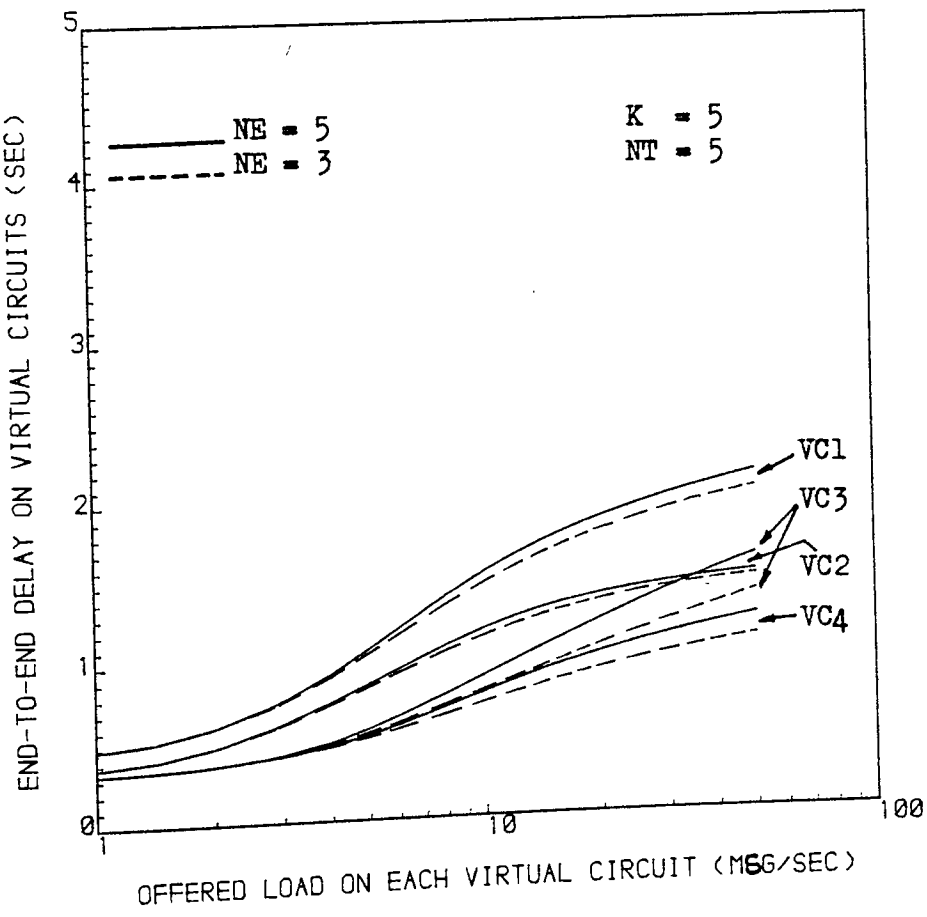
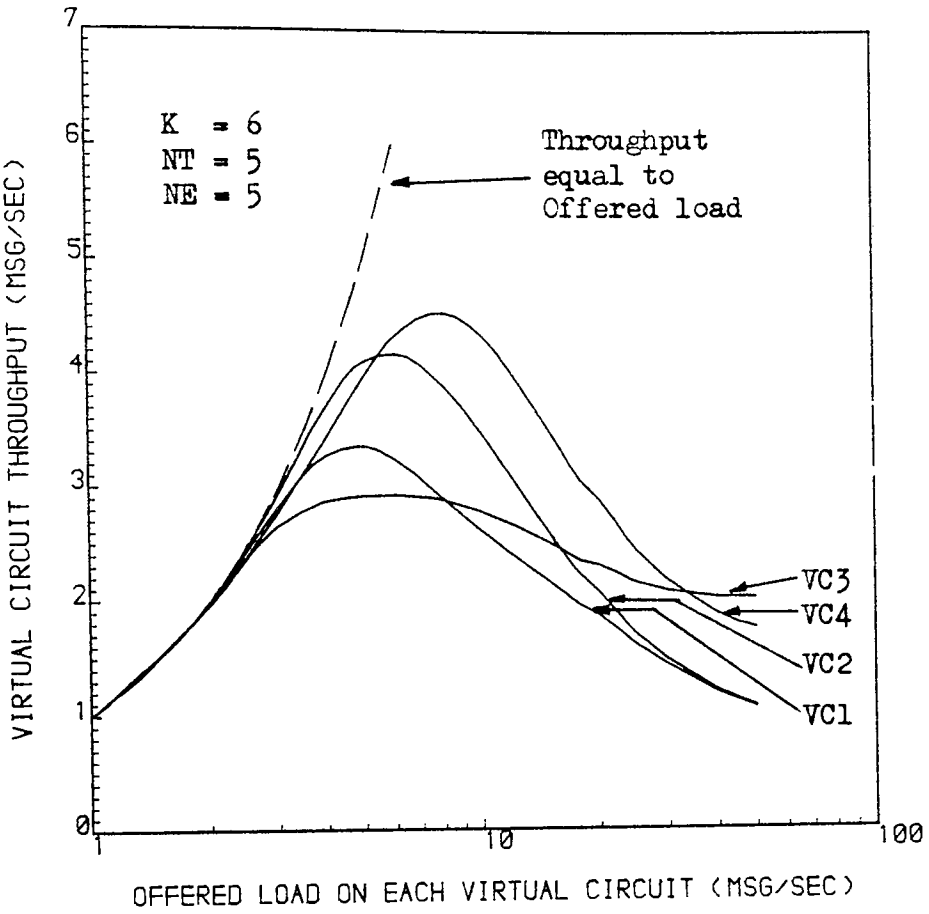


Fig. 4.6 Virtual circuit performance with end-to-end control and nodal blocking (K=5, NT=5, NE = 3 and 5).



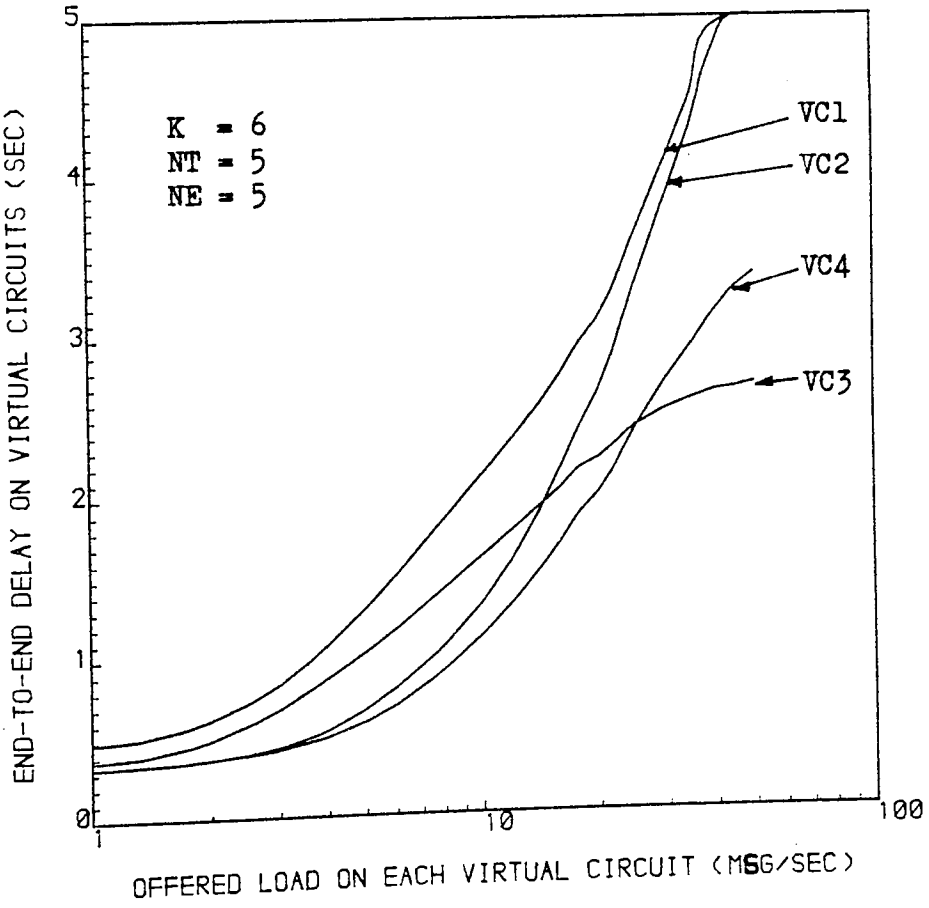
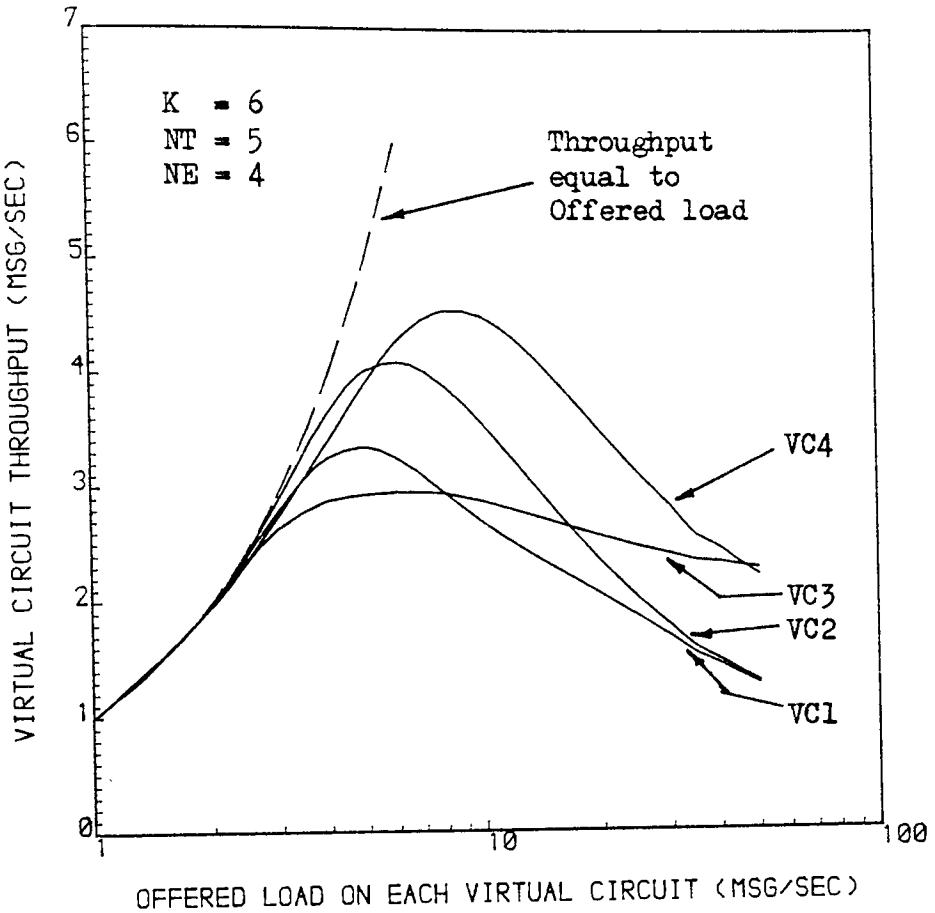


Fig. 4.7 Virtual circuit performance with end-to-end control and nodal blocking (K=6, NT=5 and NE=5).



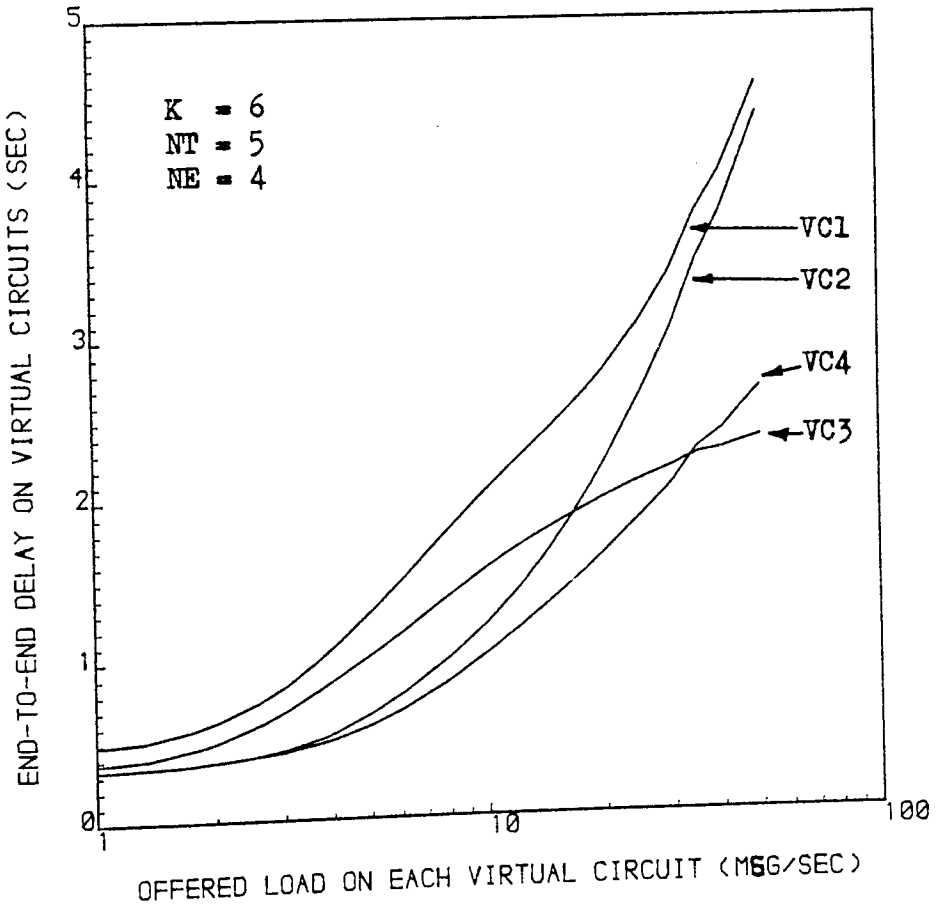
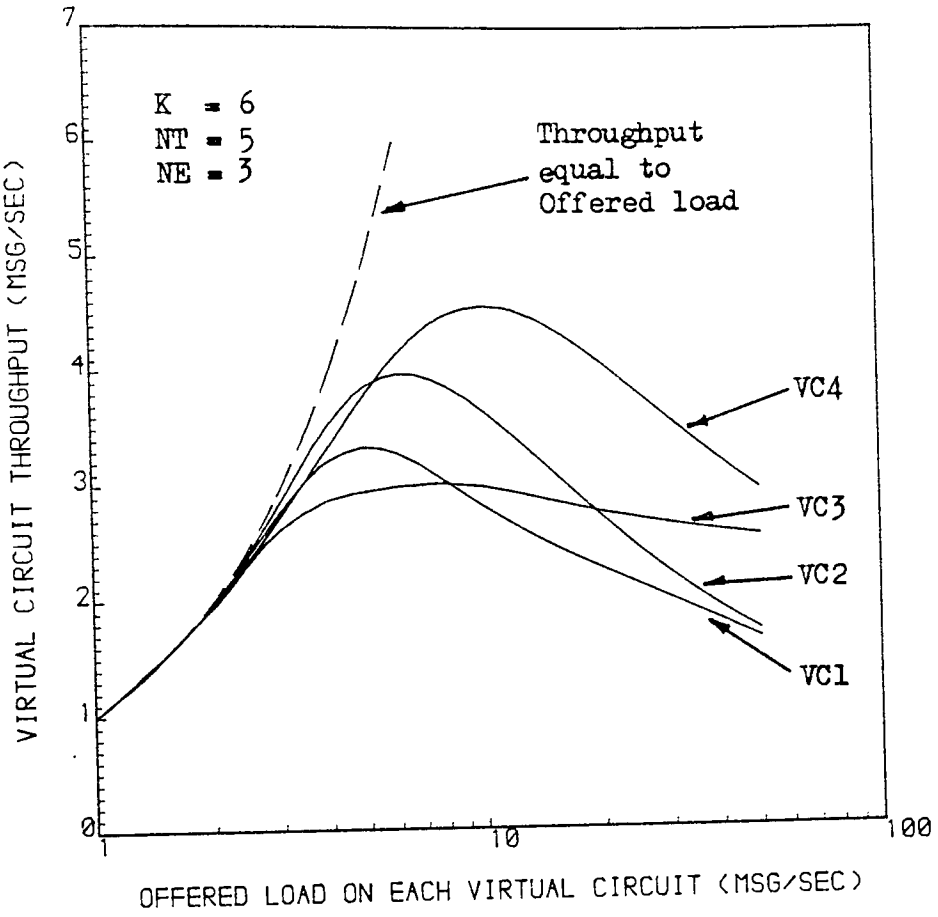


Fig. 4.8 Virtual circuit performance with end-to-end control and nodal blocking (K=6, NT=5 and NE=4).



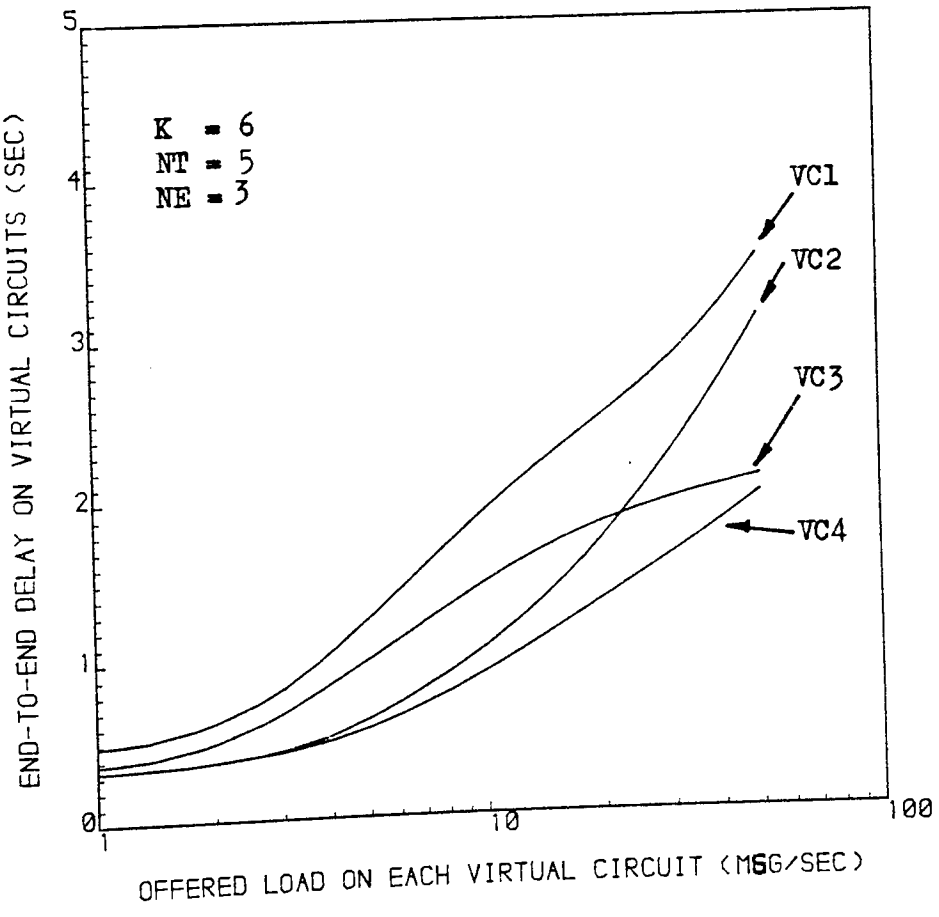
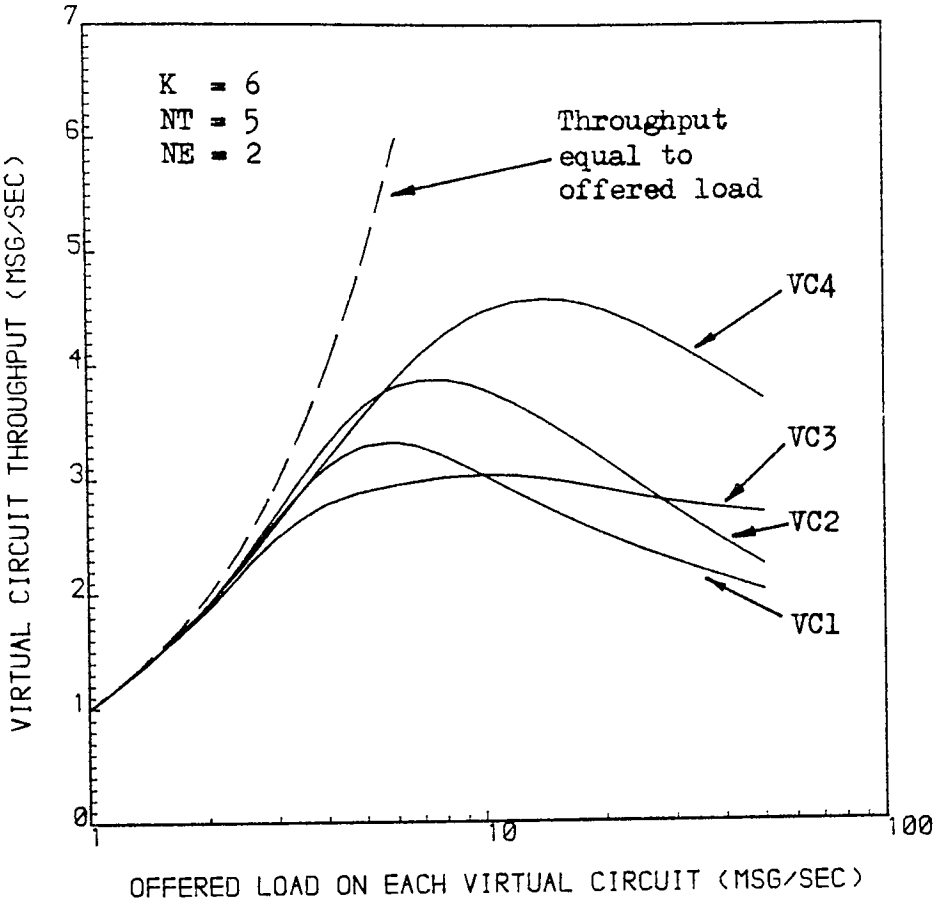


Fig. 4.9 Virtual circuit performance with end-to-end control and nodal blocking (K=6, NT=5 and NE=3).



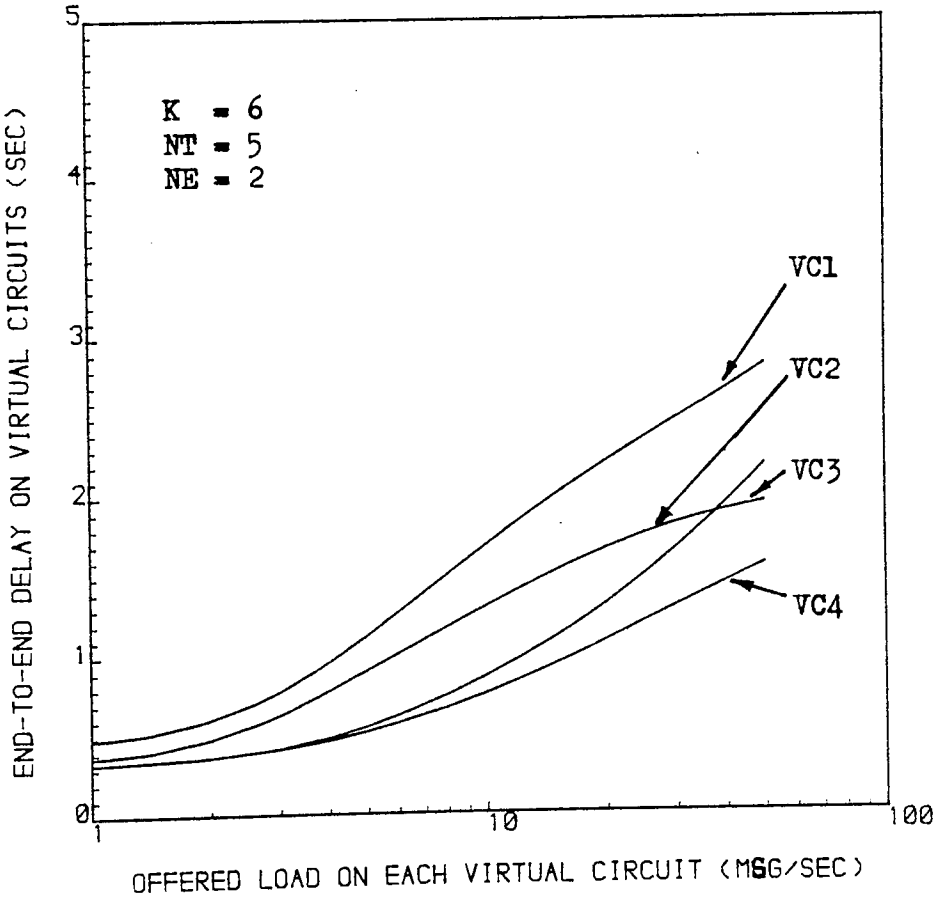
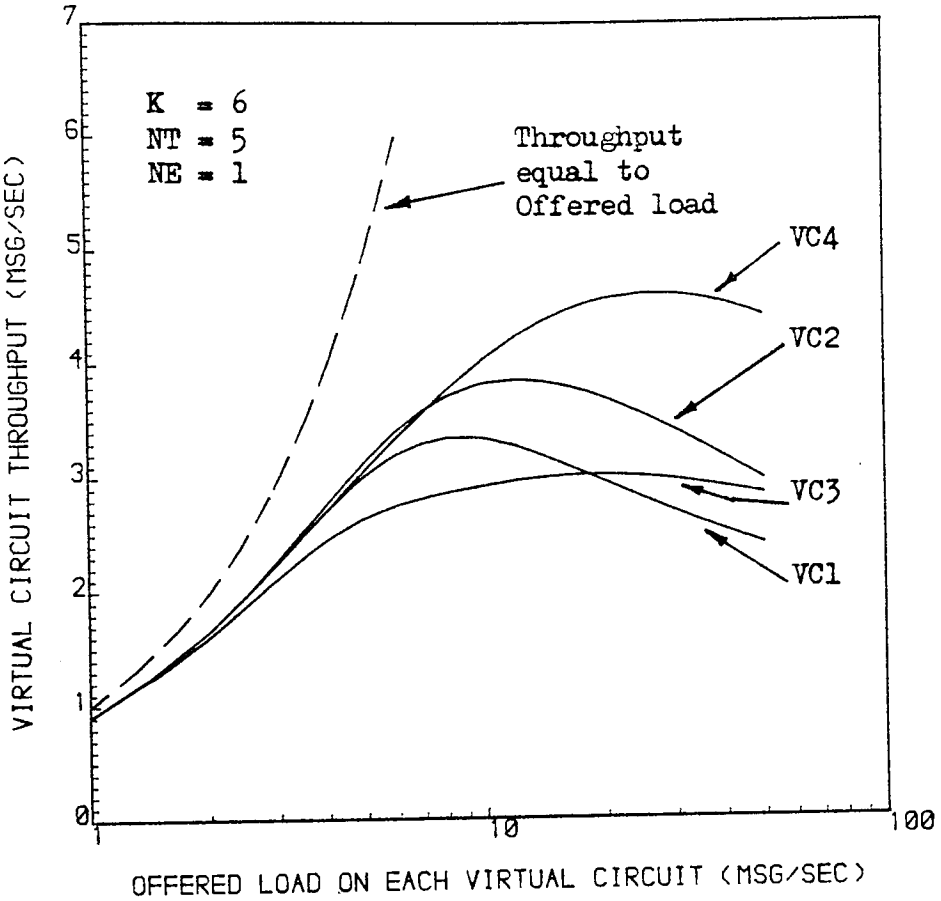


Fig. 4.10 Virtual circuit performance with end-to-end control with nodal blocking (K=6, NT=5 and NE=2).



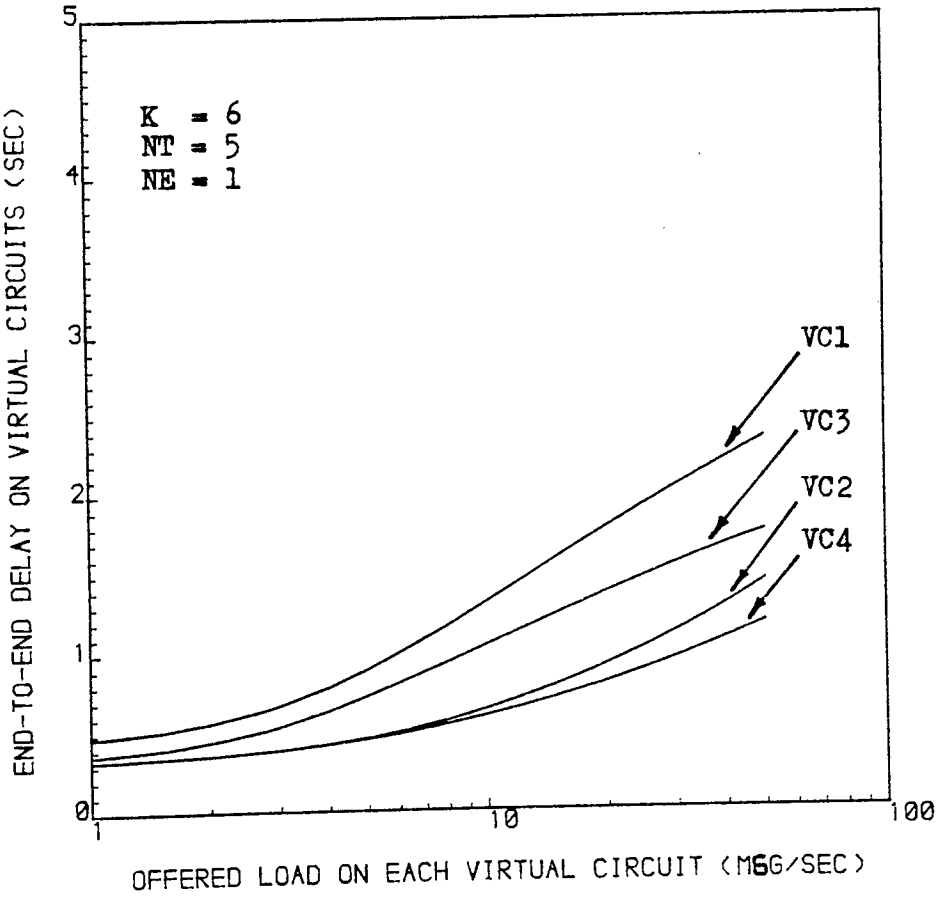
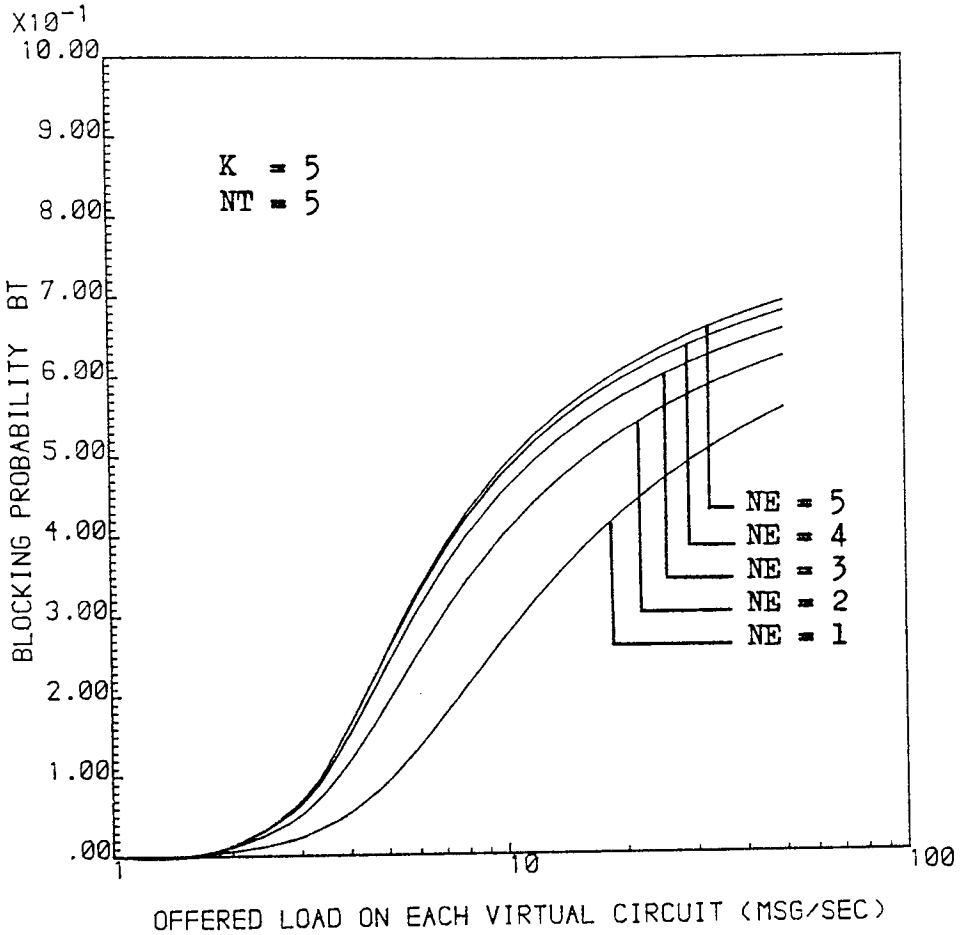


Fig. 4.11 Virtual circuit performance with end-to-end control and nodal blocking (K=6, NT=5 and NE=1).



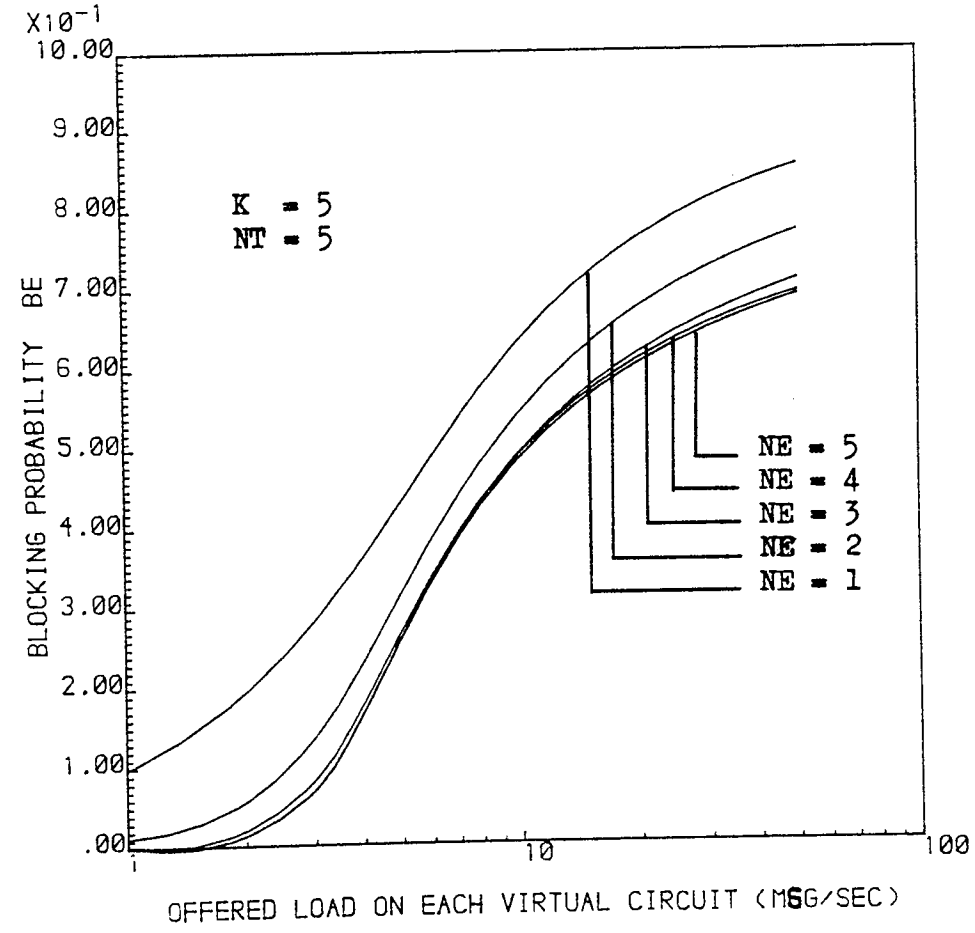
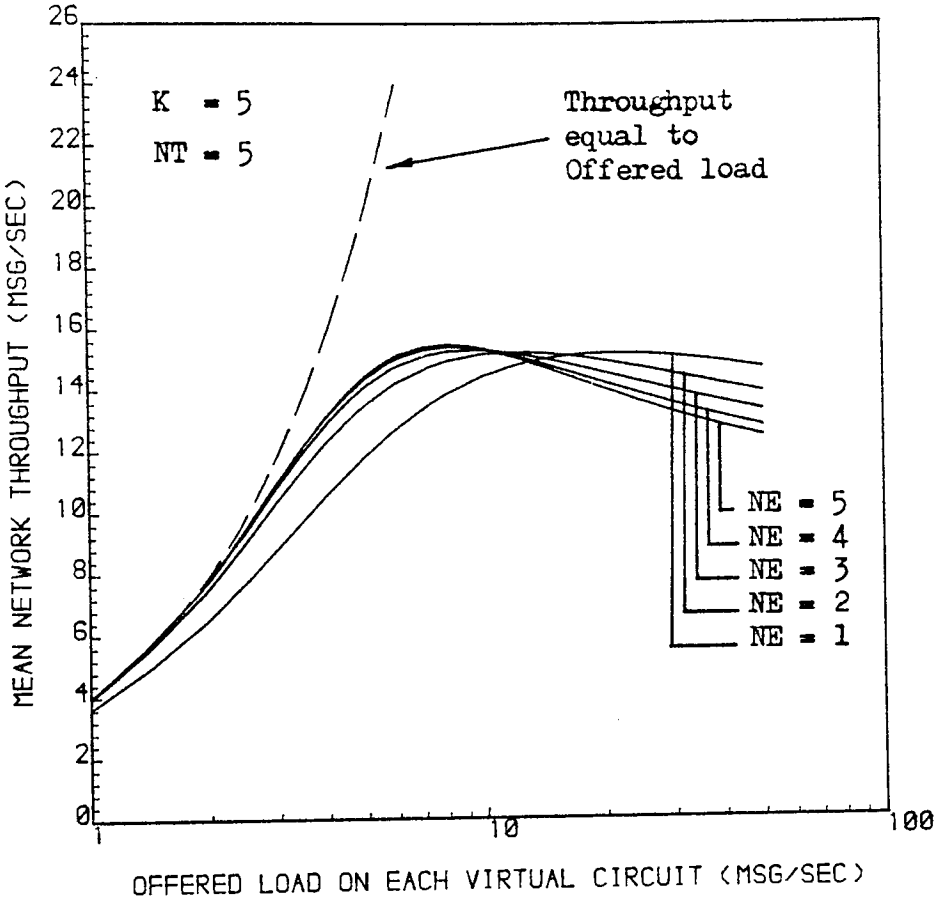


Fig. 4.12 Blocking probability for transit and external messages at node M2, with end-to-end and network access control (K=5 and NT=5).



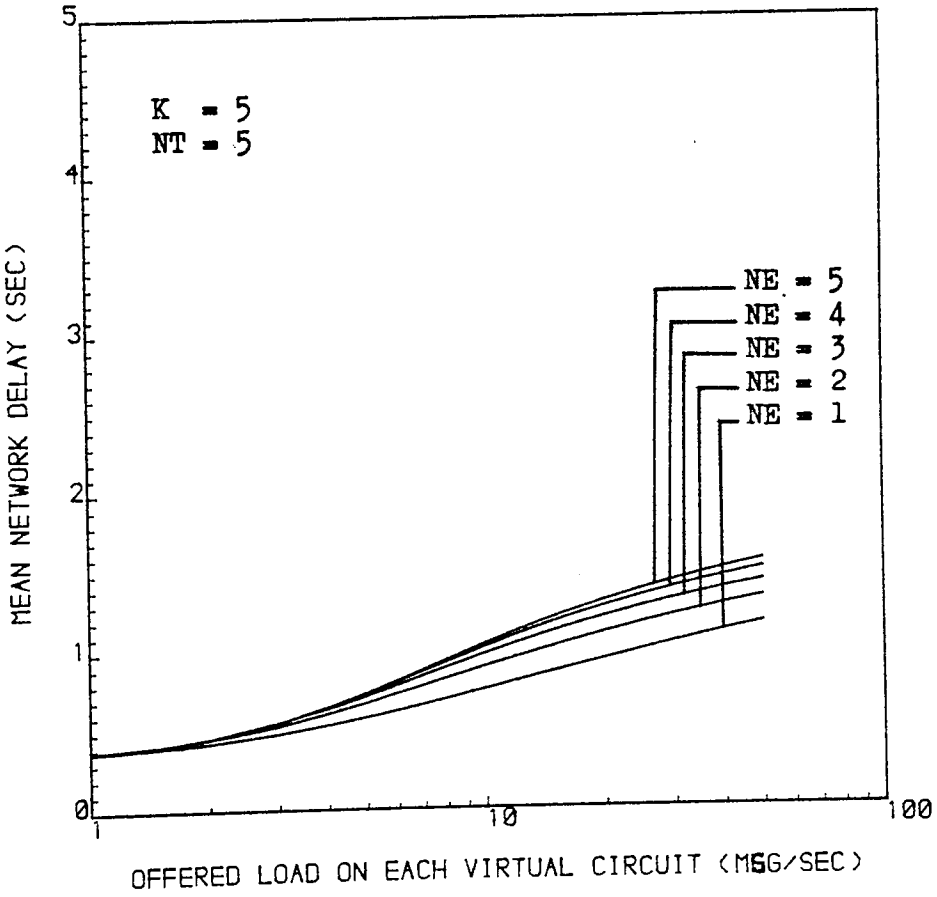
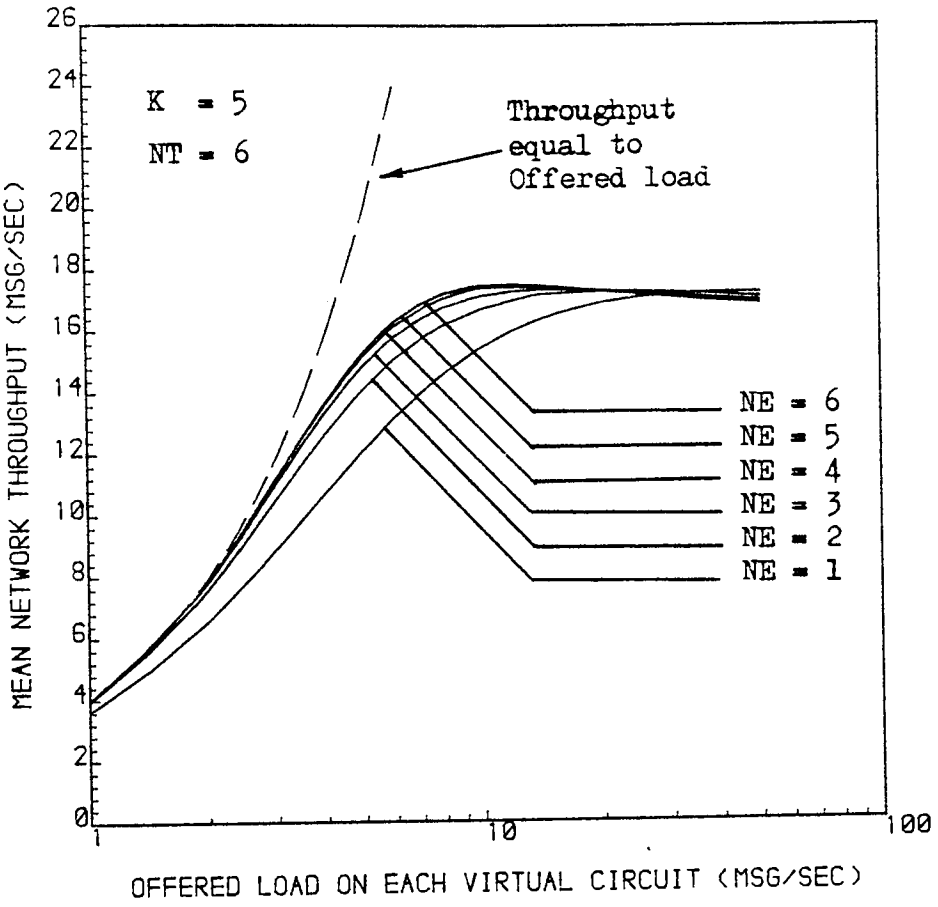


Fig. 4.13 Network performance with end-to-end and network access control when NT = 5.



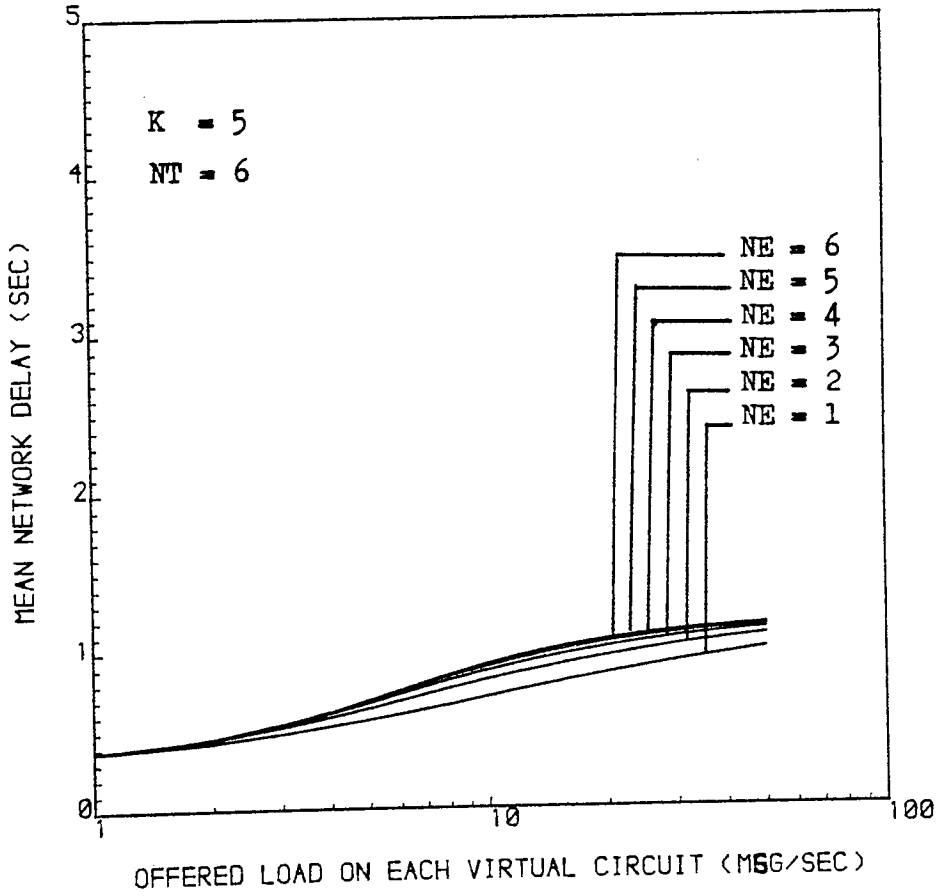
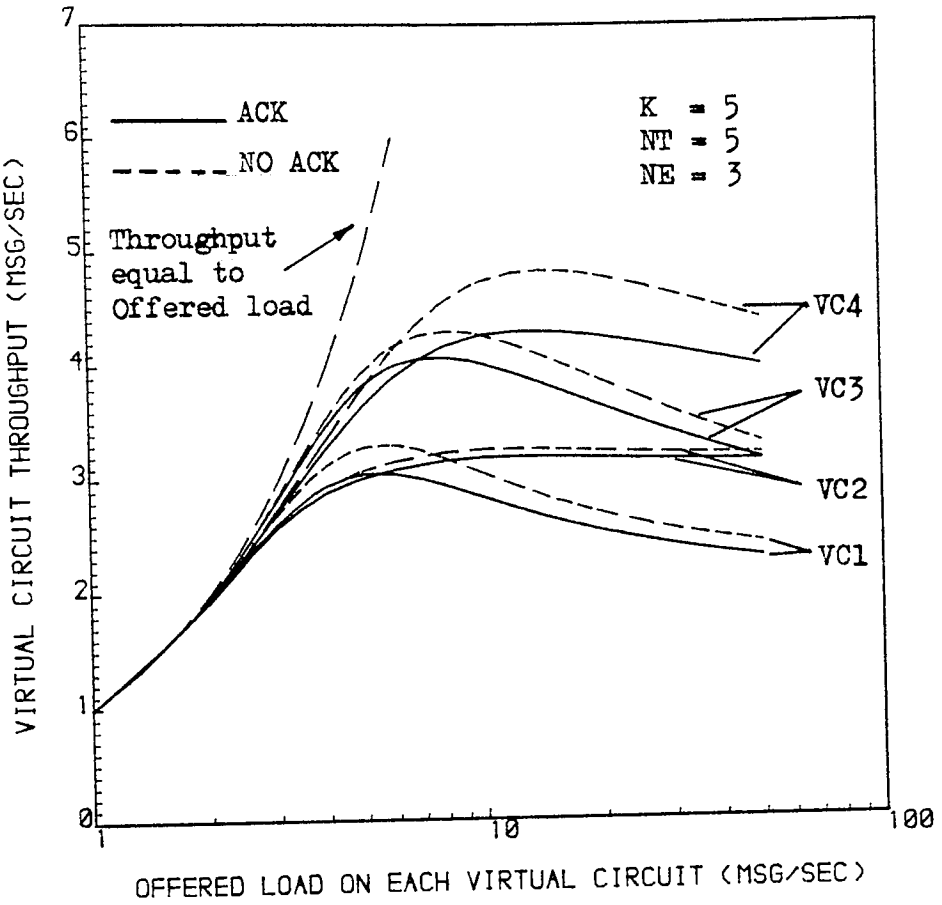


Fig. 4.14 Network performance with end-to-end and network access control when NT = 6.



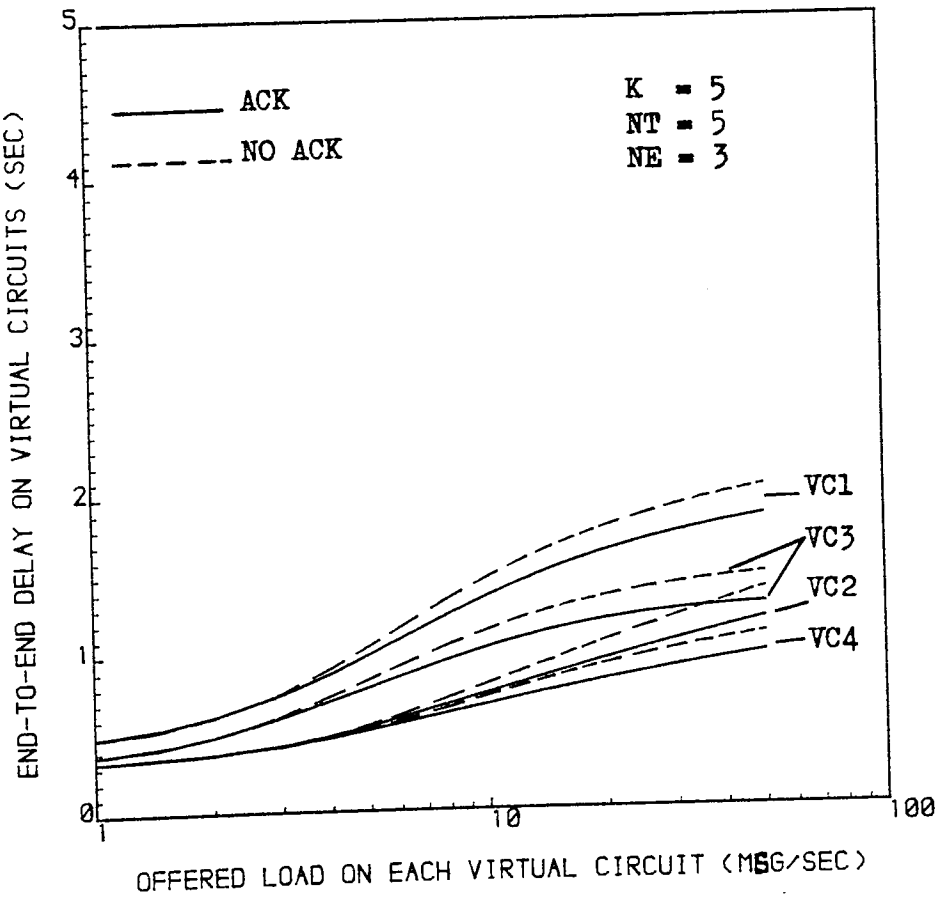
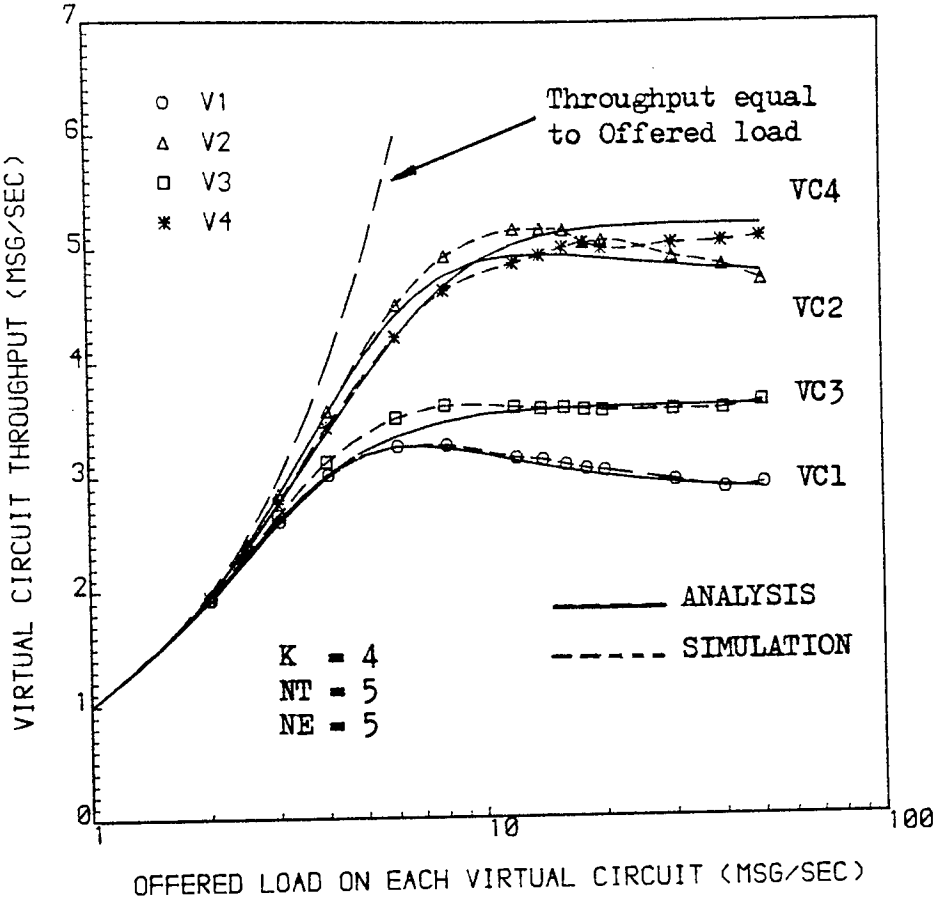


Fig. 4.15 Virtual circuit performance with end-to-end acknowledgements (K = 5, NT = 5 and NE = 3).



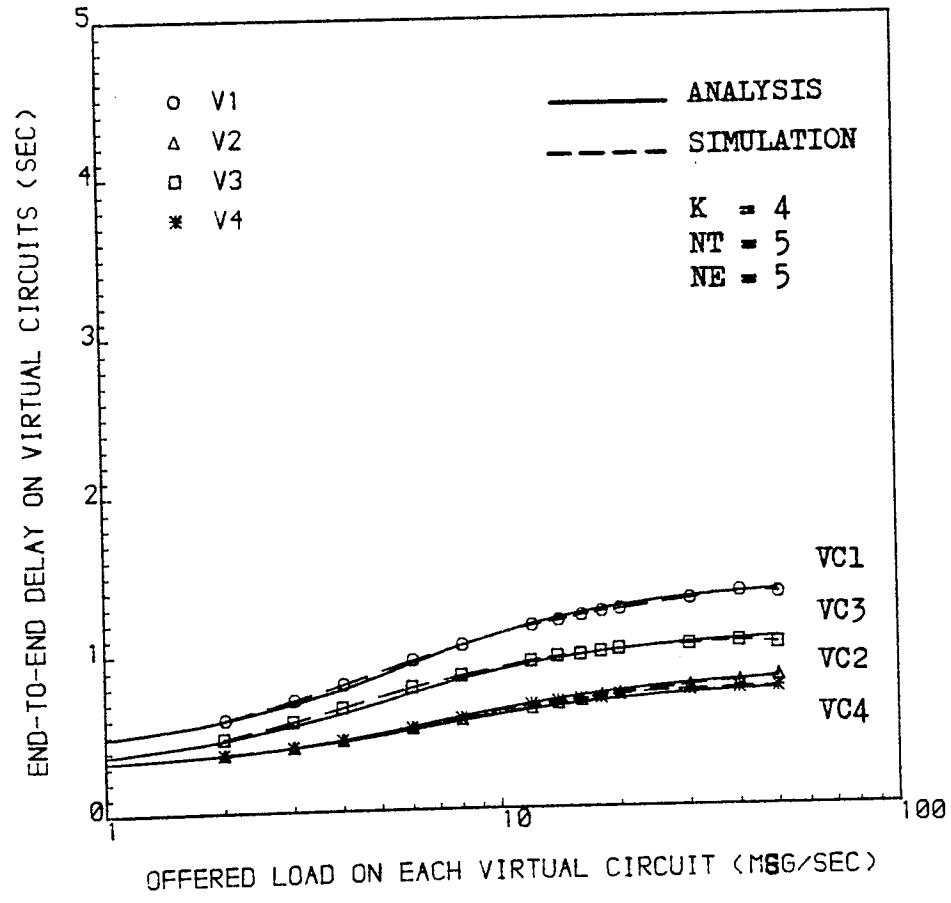
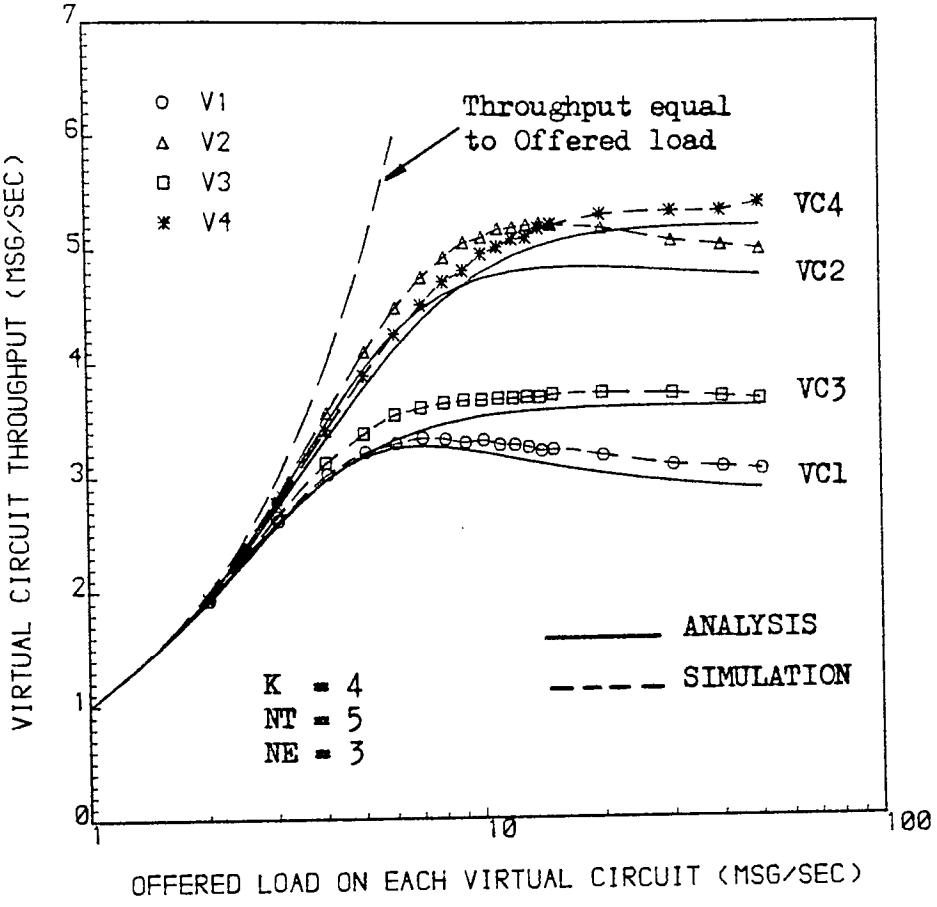


Fig. 4.16 Simulation results of virtual circuit performance with end-to-end control and nodal blocking (K = 4, NT = 5 and NE = 5).



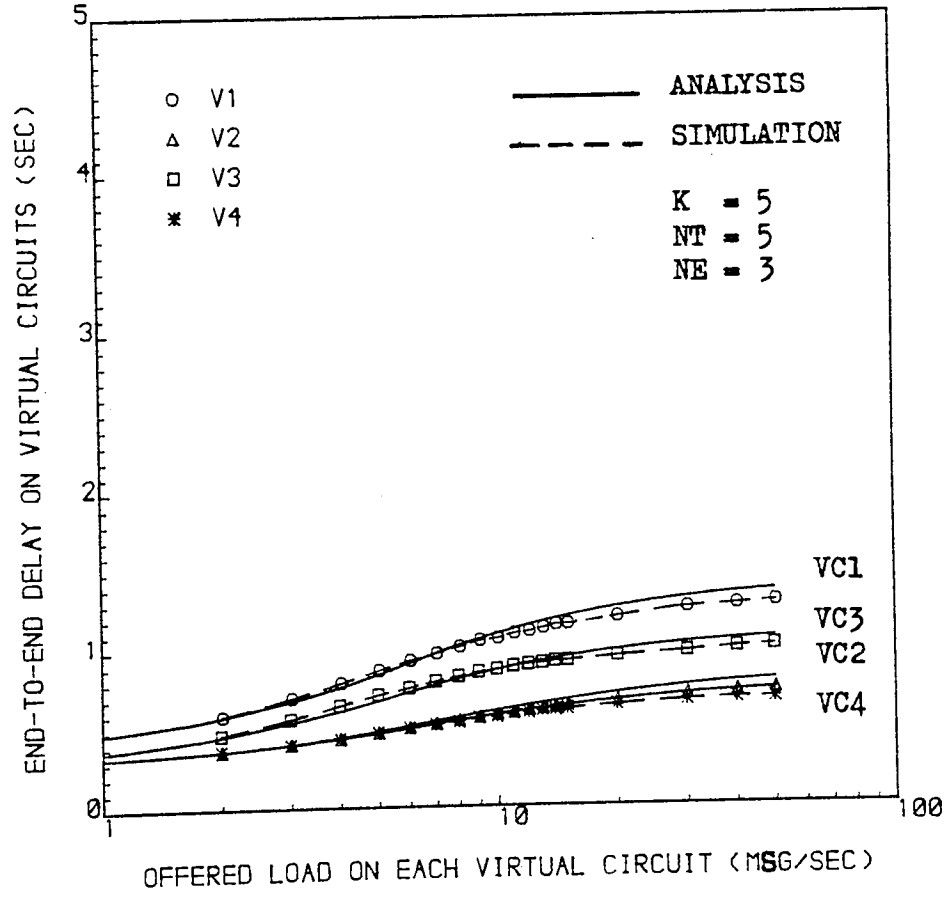
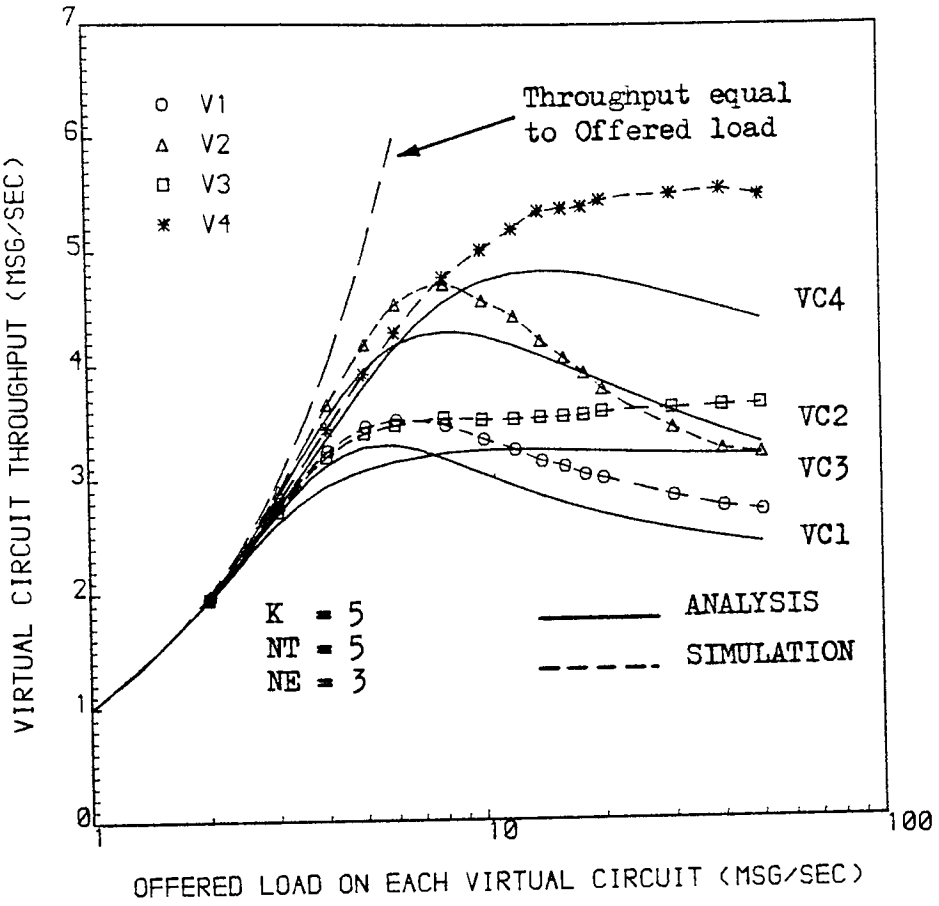


Fig. 4.17 Simulation results of virtual circuit performance with end-to-end control and network access control (K = 4, NT = 5 and NE = 3).



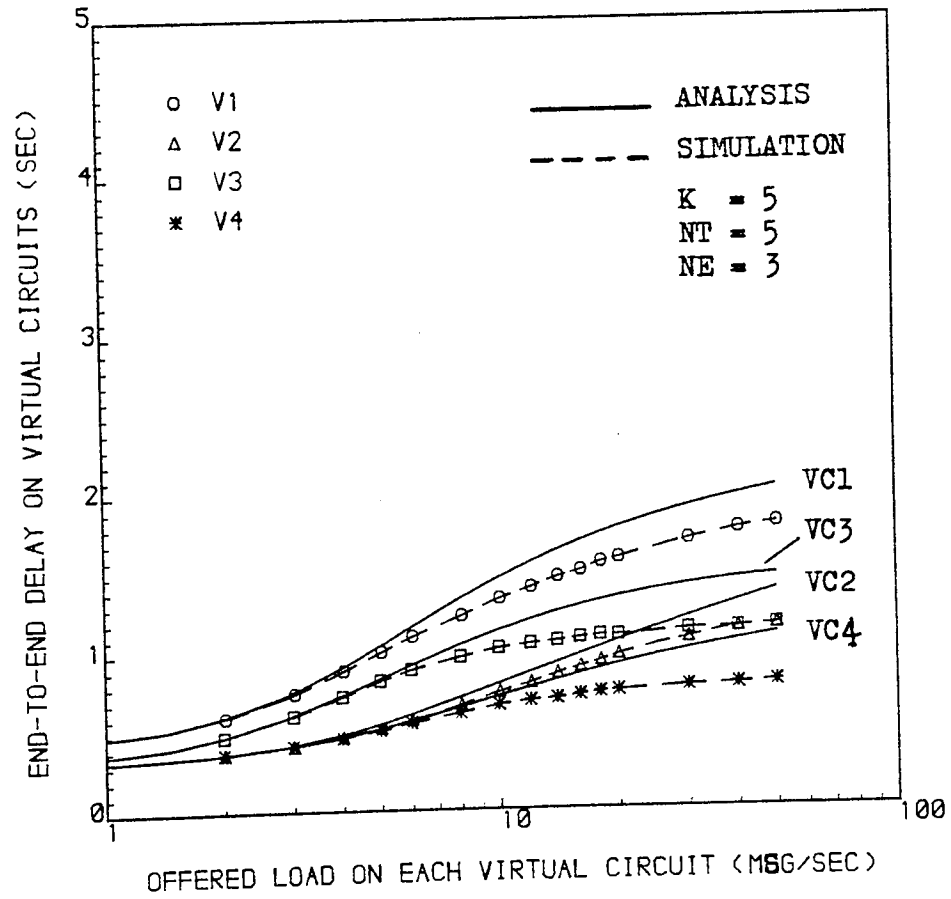
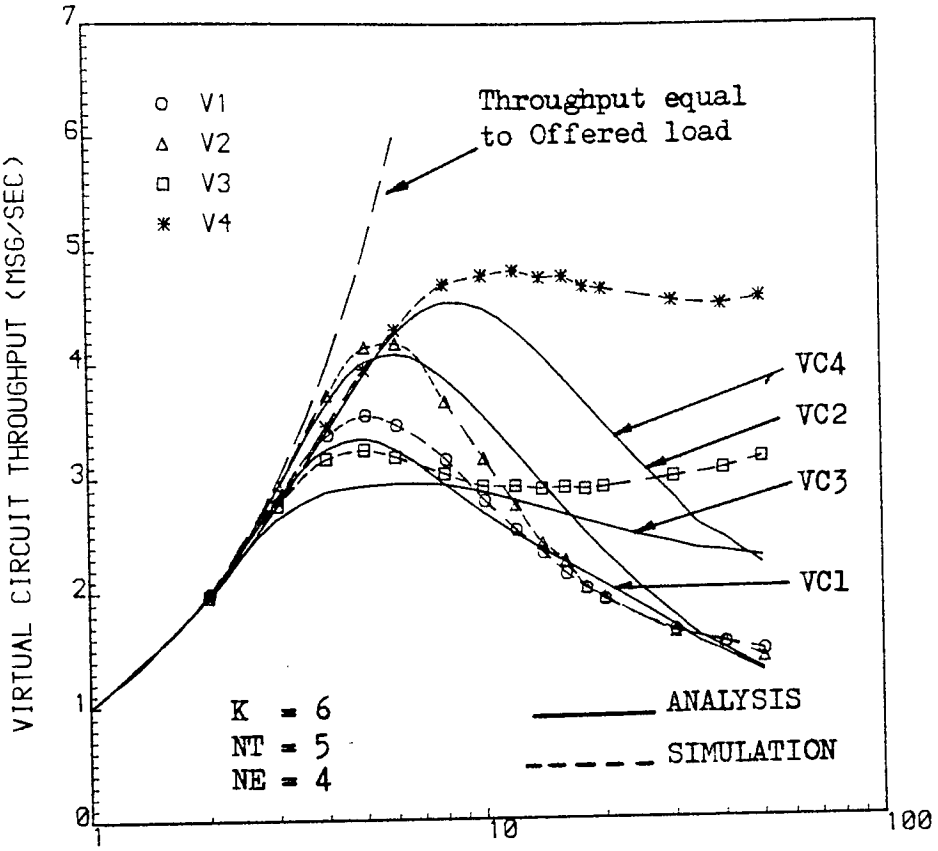


Fig. 4.18 Simulation results of virtual circuit performance with end-to-end control and network access control (K = 5, NT = 5 and NE = 3).



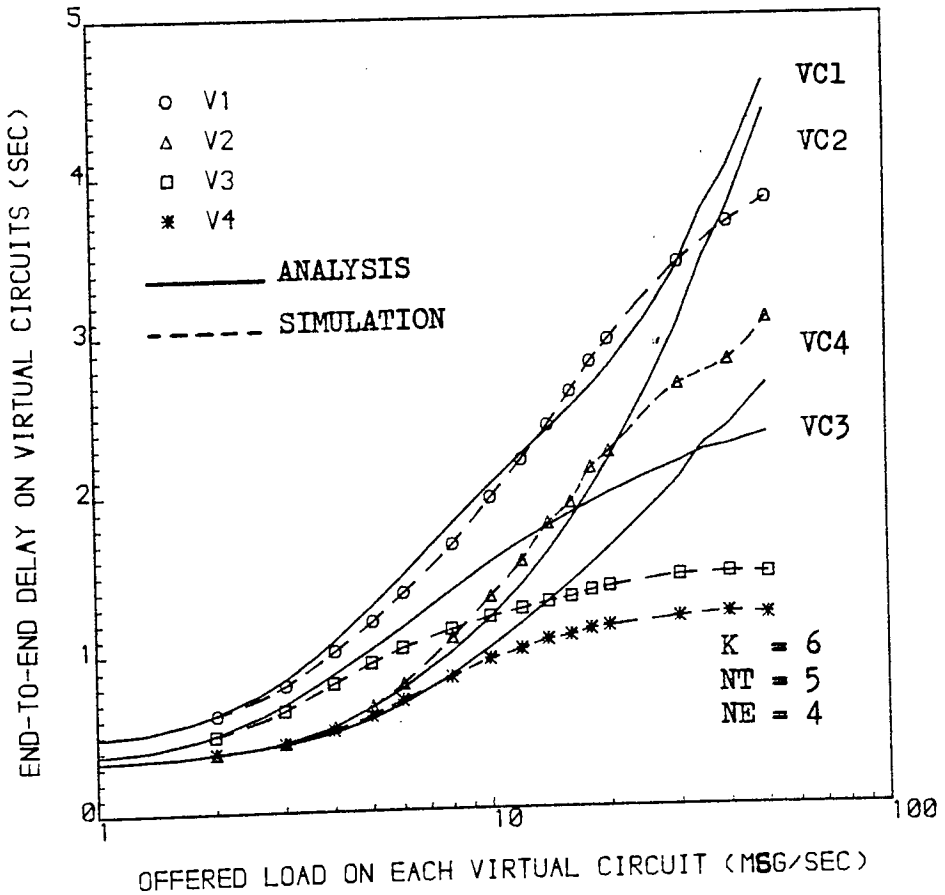
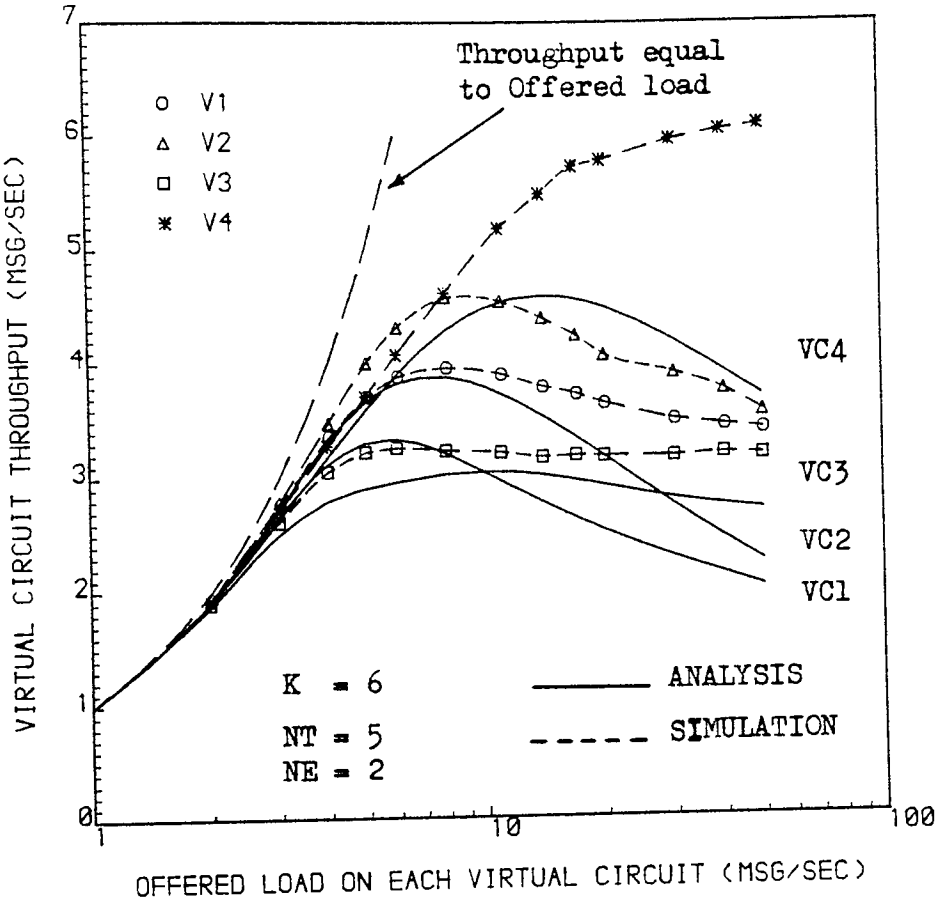


Fig. 4.19 Simulation results of virtual circuit performance with end-to-end control and network access control (K = 6, NT = 5 and NE = 4).



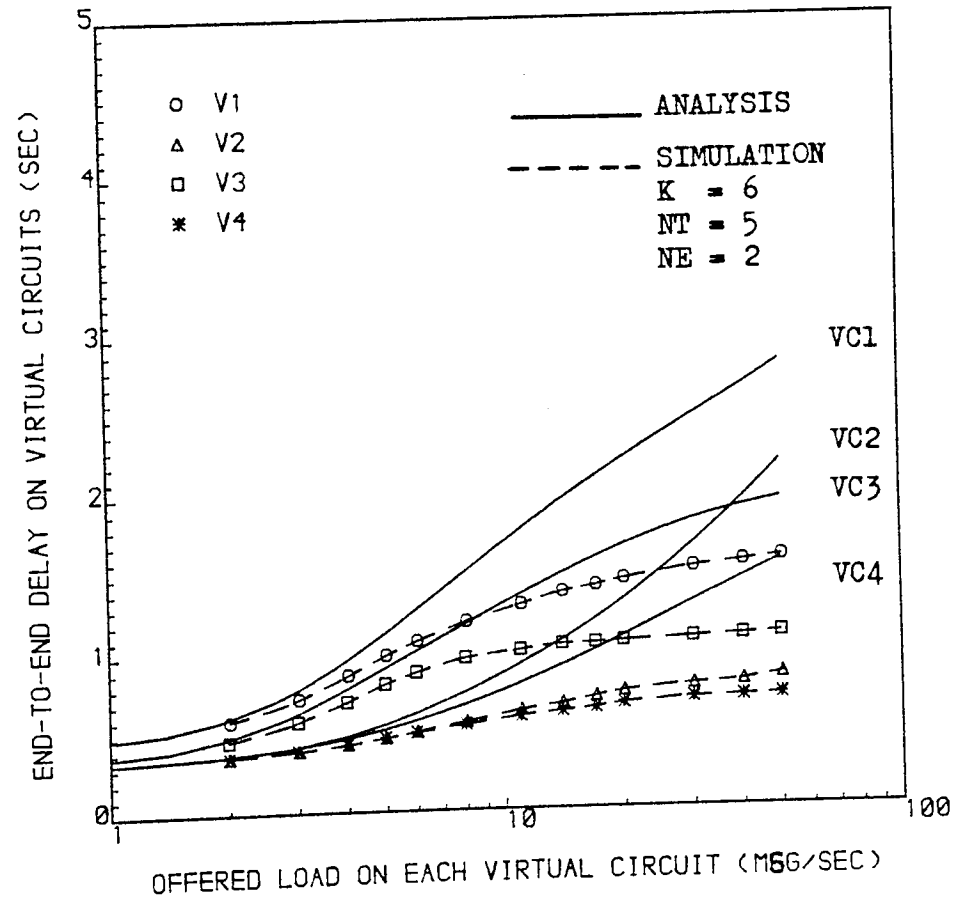


Fig. 4.20 Simulation results of virtual circuit performance with end-to-end control and nodal blocking (K = 6, NT = 5 and NE = 2).

CHAPTER FIVE

ADMISSION DELAY ON VIRTUAL CIRCUITS WITH END-TO-END FLOW CONTROL

5. ADMISSION DELAY ON VIRTUAL CIRCUITS WITH END-TO-END FLOW CONTROL

5.1. INTRODUCTION

Computer communication networks equipped with end-to-end flow control of virtual circuits, can be modelled as queueing networks with multiple closed chains (chapter 2 and 3). However, such closed chain models require the assumption of a loss system. In these models, messages that arrive from external sources are blocked and lost if the virtual circuit is fully occupied (i.e., when the window limit is full and there are no free tokens). Real data networks on the other hand are wait systems, where such a loss of messages does not occur. Messages wait in a queue outside the transit network until they can be admitted to the network. This buffering of external messages affects the stochastic nature of the arrival process. The effect of such buffering, which was not taken into account in the closed network model, is considered in this chapter.

The external queue in which messages wait until they are admitted to the transit network is known as the admission queue, and the queueing delay associated with it is called the admission delay of the virtual circuit. Reiser gave a queueing analysis for the admission delay on a single virtual route, whose traffic originate from a finite set of terminals [73]. The virtual route is controlled by an end-to-end flow control mechanism based on a token scheme, and

the finite set of terminals which use this virtual route share a common pool of tokens.

This chapter presents a new queueing model for the admission queueing process in networks with many virtual circuits. Each virtual circuit is controlled by an end-to-end flow control mechanism, and messages on each virtual circuit arrive from independent Poisson sources. The effect of buffering messages outside the network, is to increase the effective arrival rate of messages to the transit network. With the addition of the admission queue, the time spent by a token at the head of its token queue is no longer exponential. However, to make the analysis mathematically tractable, the time spent by a token at the head of its token queue is assumed to be drawn from a negative exponential distribution. Although the virtual circuits in the network are closed and the queues are exponential, the dependence between the admission queue and the token queue prevents a product form solution. In this chapter, an iterative procedure is developed for the solution of the model.

The model provides a method for evaluating the admission delay and the mean queue length of the admission queue associated with the virtual circuits. Optimum selection of window limits are also discussed. It is shown that by optimising power (defined as the ratio of virtual circuit throughput to total delay [43,44], including the admission

delay), the optimum token limit is equal to twice the hop length traversed by the virtual circuit.

5.2. THE NETWORK AND ITS QUEUEING MODEL

This section describes the structure of a network equipped with only end-to-end flow control of virtual circuits. A queueing model is developed, for the admission delay on virtual circuits, when messages arrive from independent Poisson sources.

5.2.1. STRUCTURE OF THE NETWORK

Consider a network model in which the traffic originates from independent Poisson sources. The source hosts are assigned to virtual circuits established between the source host and the destination host, through the transit network. In order to keep the network traffic within desirable limits, the virtual circuits are controlled by an end-to-end flow control mechanism. The flow control procedure is based on a token or window mechanism that limits the total number of messages in any virtual circuit, that are in transit within the network. Fig. 5.1 shows such a network model with six nodes, two independent source-destination pairs and two virtual circuits.

Each virtual circuit has a pool of K tokens, that wait in a token queue at the entry node associated with the source host. If a free token is available, a message arriving from its source host is admitted to the network, and the

token is removed from the token queue. Once the messages are delivered to their destination hosts, the tokens are returned to their token pool in the form of acknowledgements. If a message finds the token queue empty on its arrival, it waits in the admission queue for the arrival of a (free) token. The mean time spent by a message in the admission queue, is the admission delay of the virtual circuit.

5.2.2. ASSUMPTIONS

The following assumptions are made to make the model amenable to analysis.

1] There are R virtual circuits connecting R source destination host pairs.

2] Traffic on each virtual circuit originates from an independent Poisson source host. The time between consecutive arrival of messages from their respective source host, is drawn from a negative exponential distribution. The mean arrival rate of messages from the rth source host associated with virtual circuit r(r=1,2,...,R), is Ar messages per second.

3] All messages are single packet messages.

4] Messages retain their identity as they are transmitted across the network [40]. Each time a message is transmitted, its length is drawn randomly from a nega-

tive exponential distribution, that is common to all messages in the network. The mean length of a message is 1/a bits.

5] Each virtual circuit is controlled by an end-to-end flow control mechanism. The total number of messages associated with virtual circuit r, that can be in transit within the network is limited to Kr messages (r=1,2,...,R).

6] Messages are individually acknowledged and the acknowledgements are returned to their respective entry nodes with priority. The acknowledgement messages are assumed to be short in length compared to the data messages and hence the acknowledgement delays are neglected.

7] Nodes are connected by links which are modelled by simple FIFO queues.

8] There are no transmission errors.

9] There are ample number of buffers at each node and hence, there is no nodal blocking.

10] Messages blocked by the flow control mechanism queue outside the network in the admission queue, until they secure a free token. That is, no messages are lost from the system.

11] The time spent by a free token at the head of a token queue, is drawn from a negative exponential distribution. The mean time spent by a token at the head of the rth token queue is Htr (r=1,2,...,R).

5.2.3. QUEUEING MODEL AND ITS ANALYSIS

With the above assumptions, the network can be modelled as a queueing network with multiple closed chains, which has a product form solution [4,67,68,69]. Fig. 5.2 shows the queueing model of the network illustrated in fig. 5.1. Each virtual circuit is controlled independently by an end-to-end flow control mechanism, based on token or window limits. The token at the head of a token queue has to wait for the arrival of a message from its respective source host, if the associated admission queue is empty. Messages from the rth Poisson source have a mean arrival rate of Ar messages per second. A message arriving from the rth source host can enter the network immediately, if it finds a free token in the rth token queue. Otherwise it waits in the rth admission queue for the arrival of a free token.

The probability density function of the time spent by a message at the head of the rth admission queue is,

$$\begin{aligned} Far(t) = {} & [(1-Ptr(0))\ a\ Car \exp(-a\ Car\ t)] \\ & + \quad Ptr(0)\ \{[a\ Car \exp(-a\ Car\ t)] \circledast Fvr(t)\} \\ & \qquad \text{for } r=1,2,\ldots,R. \end{aligned} \tag{5.1}$$

Where,

Ptr(0) The probability of the rth token queue associated with virtual circuit r being empty;

1/a The mean length of a data message in bits;

Car Transmission capacity of the link connecting the rth admission queue and its entry node, in bits per second;

Fvr(t) Probability density function of the time between consecutive arrival of free tokens to the rth token queue.

The token at the head of the rth token queue is immediately picked up by a message in the rth admission queue, if the admission queue is not empty. Otherwise the token has to wait for a message to arrive from the rth source host. The probability density function of the time spent by a token at the head of the rth token queue (r=1,2,...,R), is given by,

$$Ftr(t) = [(1-Par(0)) \text{ at Ctr } \exp(-\text{at Ctr } t)] + Par(0) \{[\text{at Ctr } \exp(-\text{at Ctr } t)] \textcircled{*} \text{Ar } \exp(-\text{Ar } t)\}. \quad (5.2)$$

Where,

Par(0)	The probability of the rth admission queue associated with virtual circuit r being empty;
1/at	The mean length of a free token in bits;
Ctr	Transmission capacity of the link connecting the rth token queue and the entry node associated with virtual circuit r, in bits per second;
Ar	Mean arrival rate of messages from the rth source host.

The mean time spent by a token at the head of the rth token queue is,

$$Htr = \int_0^\infty t\ Ftr(t)\ dt,$$

$$= 1/(at\ Ctr) + Par(0)/Ar\ . \qquad (5.3)$$

The probability density function of the time spent by a token at the head of the rth token queue (equation 5.2) is no longer exponential. To make the analysis mathematically tractable, it is assumed that the time spent by a token at head of the rth token queue, is drawn from a negative exponential distribution having the same mean as that of equation 5.2. That is, the time spent by a token at the head of the rth token queue is exponentially distributed with a

mean given by,

$$Htr = 1/(at\ Ctr) + Par(0)/Ar. \tag{5.3}$$

With the above assumption, the R token queues can be modelled as FIFO queues. The service time of the rth token queue, is drawn from a negative exponential distribution with a mean service time of Htr, given by (5.3). Hence the network with R virtual circuits can be modelled as a closed exponential queueing network, where the rth chain has a population of Kr messages. If the probabilities Par(0), r=1,2,...,R, of the admission queues being empty are known, then the throughput of the closed chains can be solved by the convolution algorithm [68,69], or the mean value analysis algorithm [9,71]. If the network has finite number of buffers at each node (which leads to nodal blocking), the heuristic methods developed in chapter 3 can be used.

If the throughput of the rth virtual circuit is Vr (r=1,2,...,R), the probability of the rth token queue being empty is,

$$Ptr(0) = 1 - Vr\ Htr. \tag{5.4}$$

Since the closed queueing network model is an exponential system, the time between consecutive arrival of tokens to the rth token queue is also exponential, with a mean arrival

rate of Vr tokens per second. The probability density function of the time between consecutive arrival of free tokens to the rth token queue is,

$$Fvr(t) = Vr \exp(-Vr\ t) \qquad (5.5)$$

However, before the closed queueing network can be solved, the probability Par(0) of the admission queue being empty, for all r(r=1,2,...,R) must be found. The solution of the closed network requires a knowledge of the arrival rate of messages from their sources, as well as the probability of their respective admission queues being empty.

In general Ptr(0), the probability of the rth token queue being empty, can be written as,

$$Ptr(0) = Fl(\ Ar, Vr, Par(0)\ ) \quad \text{for } r=1,2,\ldots,R.$$

From equation 5.5, substituting for Fvr(t) in equation 5.1, the time spent by a message at the head of the rth admission queue has a probability density function given by,

$$Far(t) = [(1-Ptr(0))\ a\ Car \exp(-a\ Car\ t)] + Ptr(0)\ \{[\ a\ Car \exp(-a\ Car\ t)] \circledast Vr \exp(-Vr\ t)\}. \quad \text{for } r=1,2,\ldots,R. \qquad (5.6)$$

This has a mean equal to,

$$Har = 1/(a\ Car) + Ptr(0)/Vr. \tag{5.7}$$

The admission queue is modelled as an M/M/1 queue with a mean service time equal to Har. The probability Par(0) of the the admission queue associated with virtual circuit r being empty is given by,

$$Par(0) = 1 - Ar\ Har\ , \tag{5.8}$$

Hence the probability of the rth admission queue being empty can be written in general as,

$$Par(0) = F2(\ Ar,Vr,Ptr(0)\ ).$$

For a given set of arrival rates Ar (r=1,2,...,R), the above system of equations can be solved by the following iterative procedure.

1] Initialise Par(0) for r=1,2,...,R. A convenient choice is Par(0)=1.

2] Find the mean service time Htr, r=1,2,...,R, using (5.3).

3] Solve for the virtual circuit throughput Vr (r=1,2,...,R), using the convolution algorithm or the mean value analysis algorithm. If the network has

nodal blocking, the heuristic method developed in chapter 3 can be used.

4] Find Ptr(0) from equation (5.4), using the values for Vr and Htr from step 2 and 3, for all r=1,2,...,R.

5] Find Har from equation (5.7), using the values for Vr and Ptr(0) from step 2 and 4, for all r=1,2,...,R.

6] Find Par(0) from equation (5.8), using the value for Har found in step 5, for r=1,2,...,R.

7] Repeat steps 2 to 6 until probabilities Par(0) and Ptr(0) converge, for all r=1,2,...,R.

Once the iterations have converged, the admission delay of virtual circuit r is given by,

$$Dar = Har /(1 - Ar\ Har), \quad \text{for } r=1,2,\ldots,R, \tag{5.9}$$

which is the queueing delay for an M/M/1 queue [41,81]. The throughput and transit delay of the virtual circuits are the closed chain throughputs and transit delays after the last iteration.

5.3. NUMERICAL RESULTS

This section presents the numerical results for a queueing network with a single virtual circuit, that is controlled by an end-to-end flow control mechanism. Messages

arriving from the external Poisson source wait in an admission queue until they are admitted to the network. The number of hops traversed by the virtual circuit is H, and the virtual circuit window limit is K. Each transmission link in the network is assumed to have a transmission capacity of 10 messages per second. The links are represented by FIFO unbounded queues (i.e., there is no blocking within the network due to shortage of buffers).

Fig. 5.3 compares the throughput of the loss model and the wait model, for a network consisting of three store-and-forward queues between the entry node and the exit node (H=3). The virtual circuit is assumed to have window (token) limits of three and six (K=3 and 6). The throughput of the wait model follows the offered load until the virtual circuit saturates (6 messages/second when K=3, and 7.5 messages/second when K=6). For higher loads, the throughput of the virtual circuit quickly levels off and remains constant thereafter. The throughput performance of the wait model is close to the ideal performance. The throughput of the loss model is lower than the throughput of the corresponding wait model, but asymptotically approaches the throughput of the wait model for large offered loads (when the virtual circuit in the loss model also saturates).

In the loss model the free tokens in the token queue have to wait for the arrival of messages from the source. While in the wait model, since messages are buffered rather

than being lost, free tokens are alloted immediately to messages waiting in the admission queue. Since the tokens are circulated at a higher rate in the wait model, the throughput of the virtual circuit is also high. That is, the buffering of messages outside the network does not allow the free tokens to remain idle in the token queue. Hence, the admission queue improves the network throughput at low and moderate loads. The effect of buffering messages that would have been lost otherwise, is to increase the effective load on the virtual circuit. This causes the wait model to saturate at a lower source arrival rate than the loss model. Though the wait model saturates earlier, its throughput is higher than that of the loss model.

Fig. 5.4 shows the plot of network transit delay for the wait model and the loss model, as a function of offered load for window limits of three and six (K=3 and 6). The virtual circuit is assumed to traverse three hops (H=3). For a given load, the transit delay of the wait model is about 25% larger than that of the loss model. This is because, more number of messages are in transit in the wait model than in the loss model. In both cases the network delay increases with the offered load, and also with the window limit.

Fig. 5.5 shows the admission delay and the network transit delay as a function of offered load, for various window limits, when the hop length is three (H=3). For a

given window limit, when the offered load is small, the admission delay is small compared to the network transit delay. As the load increases, the admission delay also increases and becomes very large when the offered load exceeds the saturation load. The network transit delay also increases with the offered load, but levels off and remains constant for large loads. For a given offered load, larger window limits lead to lower admission delays, although the network transit delay does increase. For the admission delay to be small, the window limit must be chosen such that the saturation load of the window limit is larger than the offered load.

Fig. 5.6 shows the plot of the admission delay and network transit delay as a function of window limit, for various hop lengths (H=2 to 6). The message arrival rate is assumed to be constant at three messages per second (A=3 messages/second). For a given window limit, the admission delay increases as the hop length increases. This is because, as the hop length increases, tokens take more time to return to the token queue, and messages in the admission queue have to wait longer. However, if the window limit is increased, more tokens become available and the waiting time is reduced. Hence, if the admission delay is not exceed a given measure, the window limit must also increase with the hop length.

The admission delay of a virtual circuit can be kept

small by increasing the window limit. However, an increase in the window limit leads to larger network transit delay and can also reduce the effectiveness of the end-to-end flow control mechanism. Hence, a proper choice of the window limit is necessary to balance the admission delay and the network delay. A criteria for choosing the window limit is power, defined as the ratio of virtual circuit throughput to total delay [44]. In this case, the delay is the sum of the network transit delay and the admission delay . That is,

$$\text{Power} = \frac{\text{Virtual circuit throughput}}{\text{admission delay + network transit delay}}$$

Fig. 5.7 shows the plot of maximum power versus window limit, for various hop lengths. Since power is also dependent on the offered load (the throughput and delay depend on the offered load), for a given window limit the optimum offered load is one that gives maximum power. The plot shows that for a given hop length, as the token limit increases, power also increases, reaching a maximum and then begins to decrease very slowly. The peak value of power is obtained when the token limit is around twice the hop length. This result agrees with those of Reiser's model [73]. Further, the offered load at which power is maximised is approximately the same for all hops, when the window limit is chosen to be twice the hop length. From fig. 5.8, the optimum offered load is approximately 6 messages per second.

Since the channel transmission capacities of all the links are assumed to be equal to 10 messages per second, the optimum offered load is approximately 60% of the transmission capacity of the transit links.

In the case of the loss model, by maximising the ratio of virtual circuit throughput to network transit delay, the optimum number of tokens can be shown to be equal to the hop length [43,52]. However, a token limit equal to the hop length gives rise to higher admission delays, especially at high loads. This is because, with fewer tokens assigned to the virtual circuit, messages have to wait longer for the arrival of a free token. If the token limit is made equal to twice the hop length, the saturation limit is raised and lower admission delays can be obtained if the offered load is less than the saturation load. From fig. 5.5, the admission delay at a load of 6 message per second when the token limit is equal to the hop length (K=3 and H=3), is about 1.75 seconds. With a token limit equal to twice the hop length (K=6 and H=3), the admission delay is about 0.04 seconds. The corresponding transit delays are 0.5 and 0.6 seconds. Hence, choosing a token limit equal to twice the hop length reduces the admission delay by 95%, while the network delay increases by 20%. The improvement in total delay (admission and transit delay) is about 70%. However, for smaller loads, though the percentage improvement in admission delay is large, the improvement in total delay reduces. This is because, at small loads, the transit delay

is larger than the admission delay.

Table 5.1 gives the percentage change in admission delay and network transit delay, when the token limit is chosen to be equal to twice the hop length and the offered load is equal to the saturation load. The comparisons are made with respect to the admission delay and transit delay obtained when the token limit is equal to the hop length. In the SNA network, a window limit between H and 3H is chosen, where H is the path hop length [1]. The GMDNET has a window limit equal to 2H+1 [21].

5.4. CONCLUSIONS

A model is developed for the admission queueing process in network with many virtual circuits. The virtual circuits are controlled by an end-to-end flow control mechanism, and traffic originate from independent Poisson sources. This model represents real data networks, where messages that fail to secure tokens wait in a queue until they are admitted to the network. That is, the model represents a wait system rather than a loss system. The effect of buffering messages that would have otherwise been lost due to blocking by the flow control mechanism, is to increase the effective offered load on the virtual circuits. The virtual circuit throughput in the wait model follows the offered load until saturation, when the throughput levels off. Though the virtual circuits saturate at lower levels of offered load, than in the case of the loss model, higher throughputs are

Table 5.1

Percentage change in admission delay and network delay when the window limit is made equal to twice the hop length

Number of hops	Reduction in admission delay	Increase in network delay
2	86%	26%
3	91%	21%
4	94%	17%
5	96%	15%
6	97%	13%

obtained with the wait model. The admission queue improves the virtual circuit throughput at low and moderate loads.

For a given window limit, when the offered load is small, the admission delay is smaller than the network transit delay. For loads larger than the saturation loads (which occurs at moderate values of offered load for small window limits), the admission delay becomes very large. A large window limit raises the saturation limit leading to smaller admission delays (if the offered load is less than the saturation load). But this also increases the transit delay. An optimum window limit can be found by maximising power, defined as the ratio of virtual circuit throughput to total delay. For maximum power, the optimum window limit is found to be equal to twice the hop length. With the window limit equal to twice the hop length, the optimum arrival rate at which power is maximised, is found to be the same for all hop lengths, and is approximately equal to 60% of the transmission capacity of transit links.

The model gives the admission delay on virtual circuits, the mean number of messages queueing in the admission queue, the virtual circuit throughputs and end-to-end delays, for a given set of arrival rates and window limits. The analysis is based on certain simplifying assumptions. In view of this, the results are only an approximation to the actual behaviour of real systems. However, the model can be used to decide the admission queue size and the window lim-

its in the initial design phase and thus, adds to the set of tools for the design of networks.

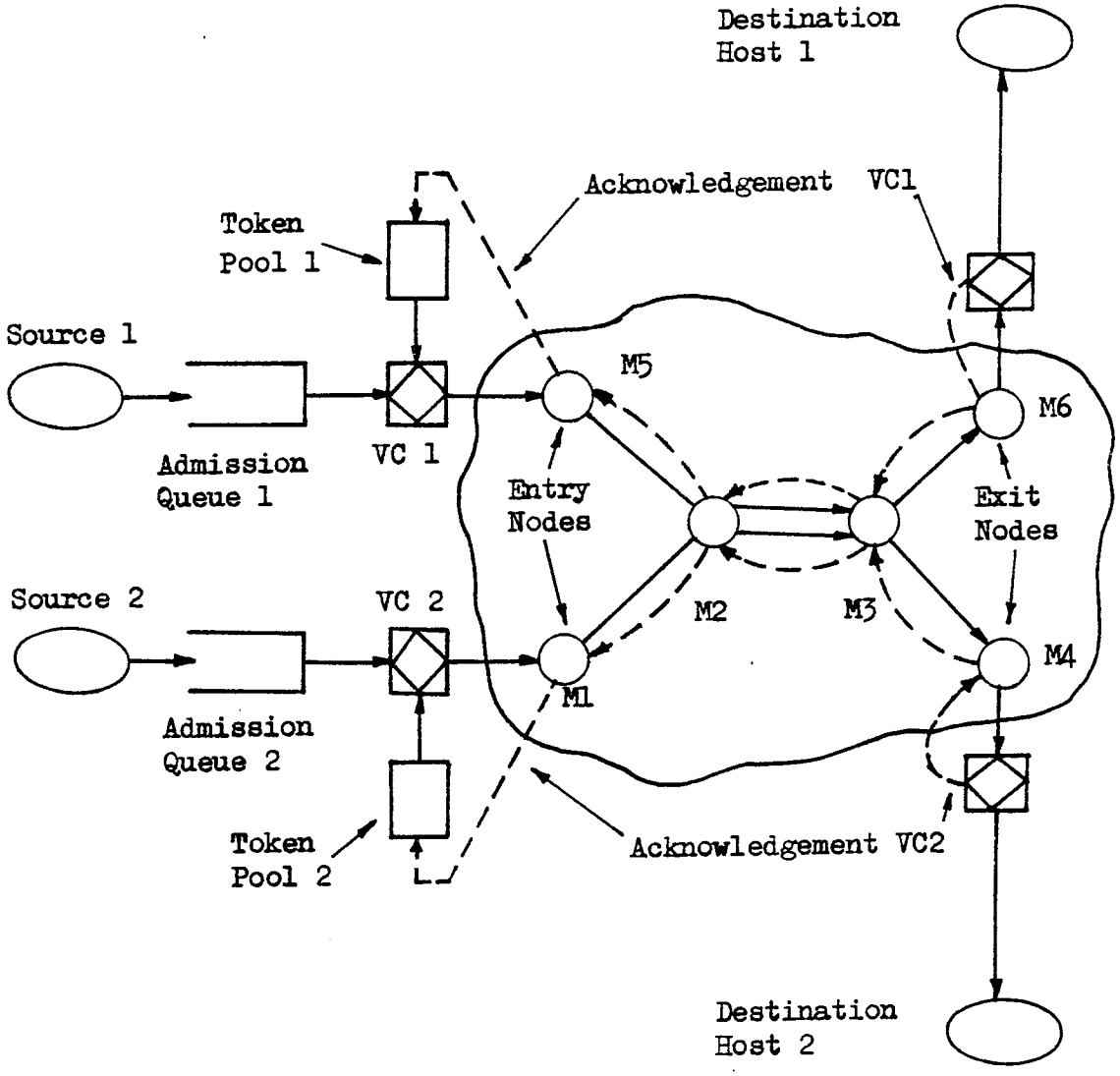


Fig. 5.1 Model of a network with admission queues.

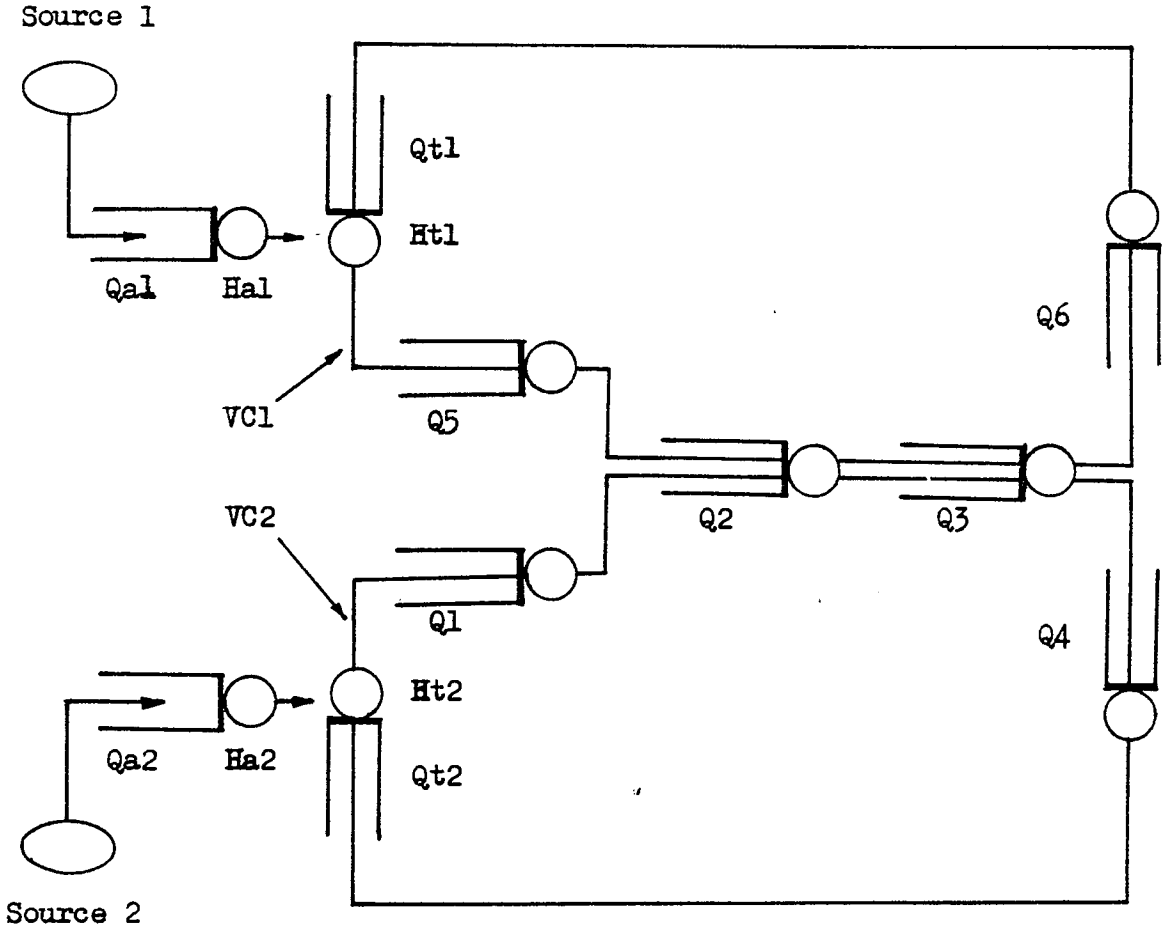


Fig. 5.2 Queueing model of the network in figure 5.1.

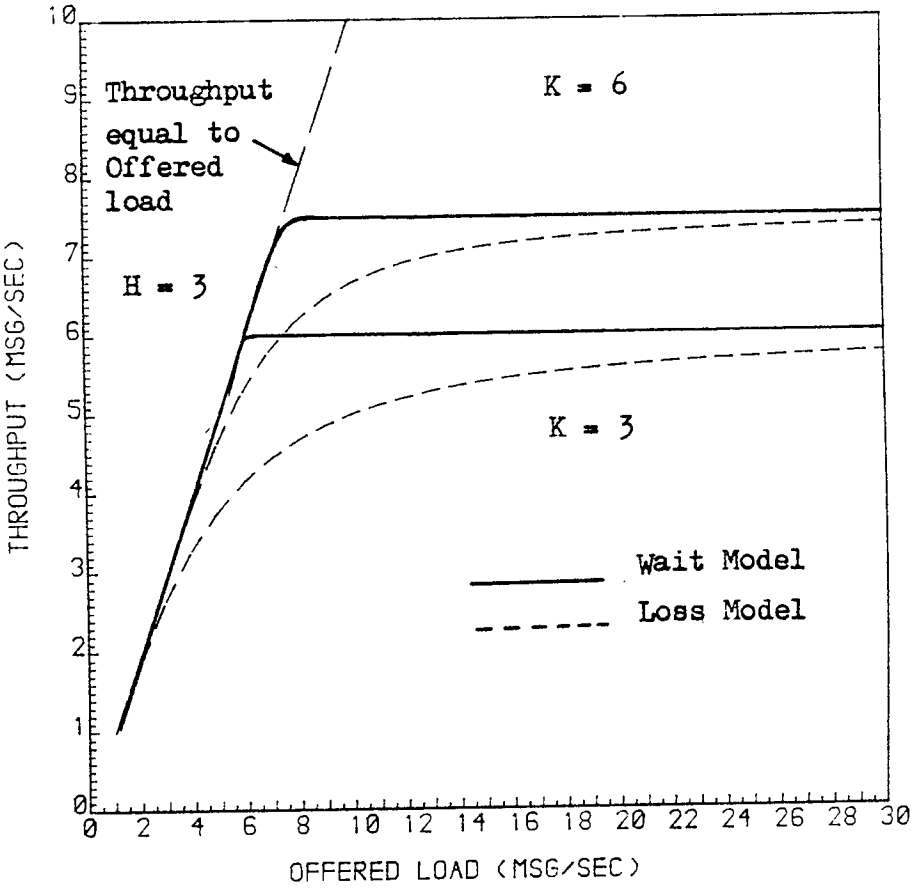


Fig. 5.3 Virtual circuit throughput for the loss and wait model (K = 3 and 6, H = 3).

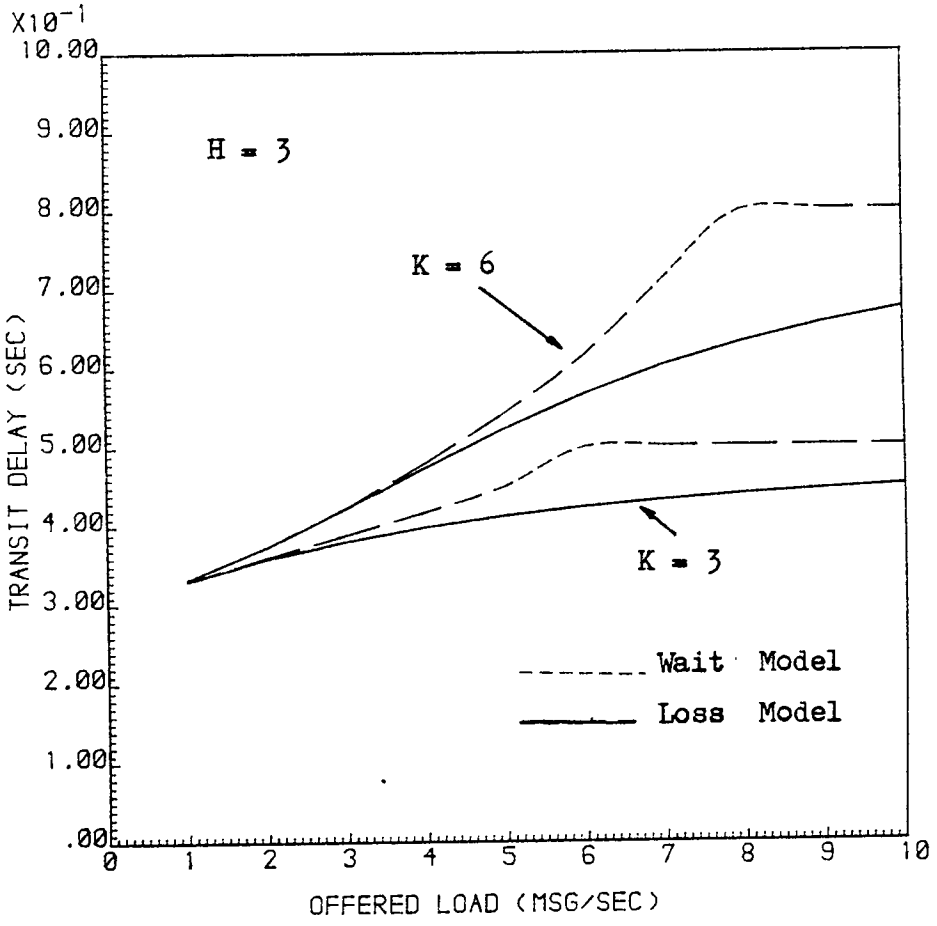


Fig. 5.4 Transit delay for the wait and loss model.

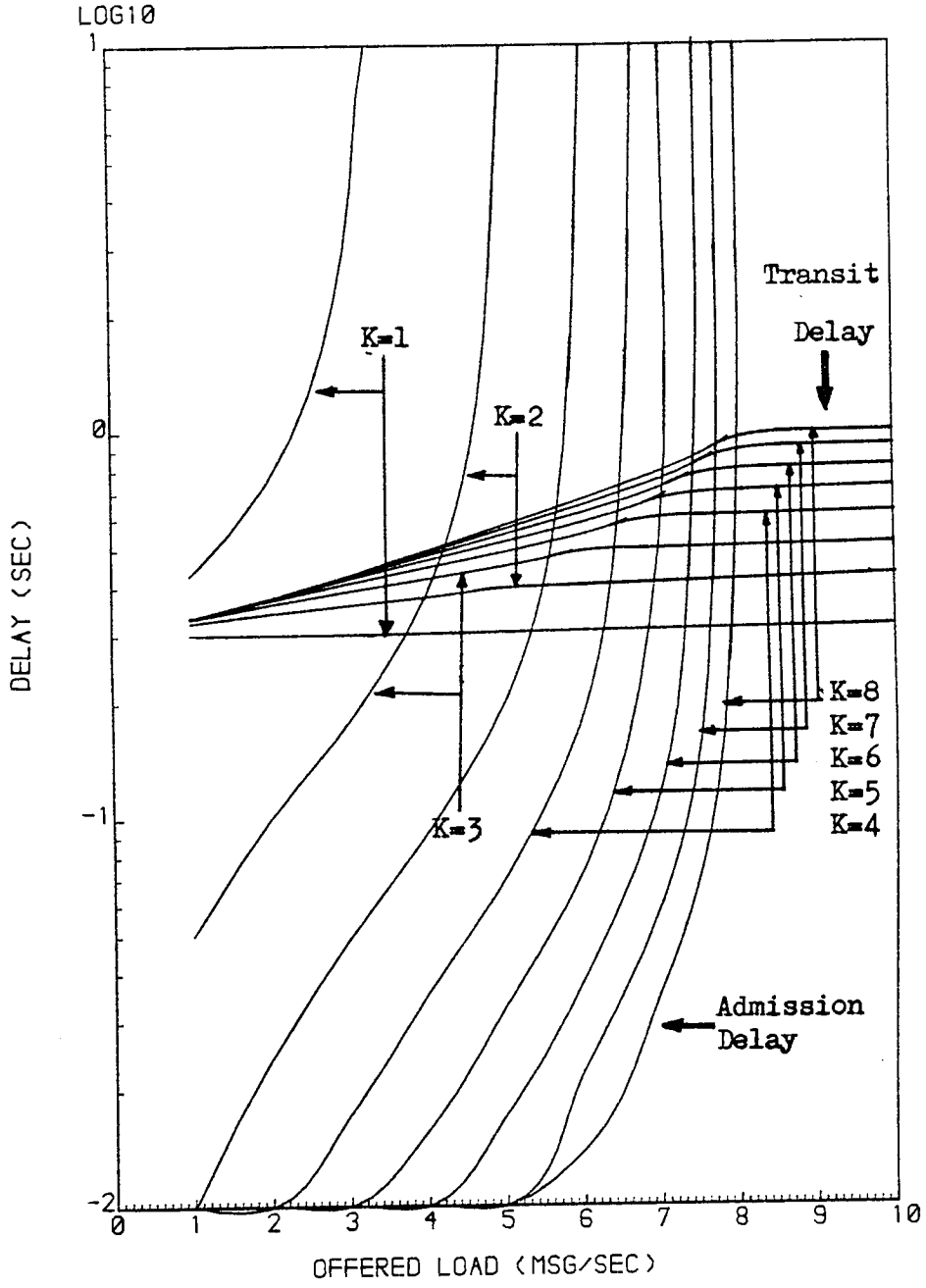


Fig. 5.5 Admission delay and transit delay as a function of offered load, for various window limits (number of hops, H = 3).

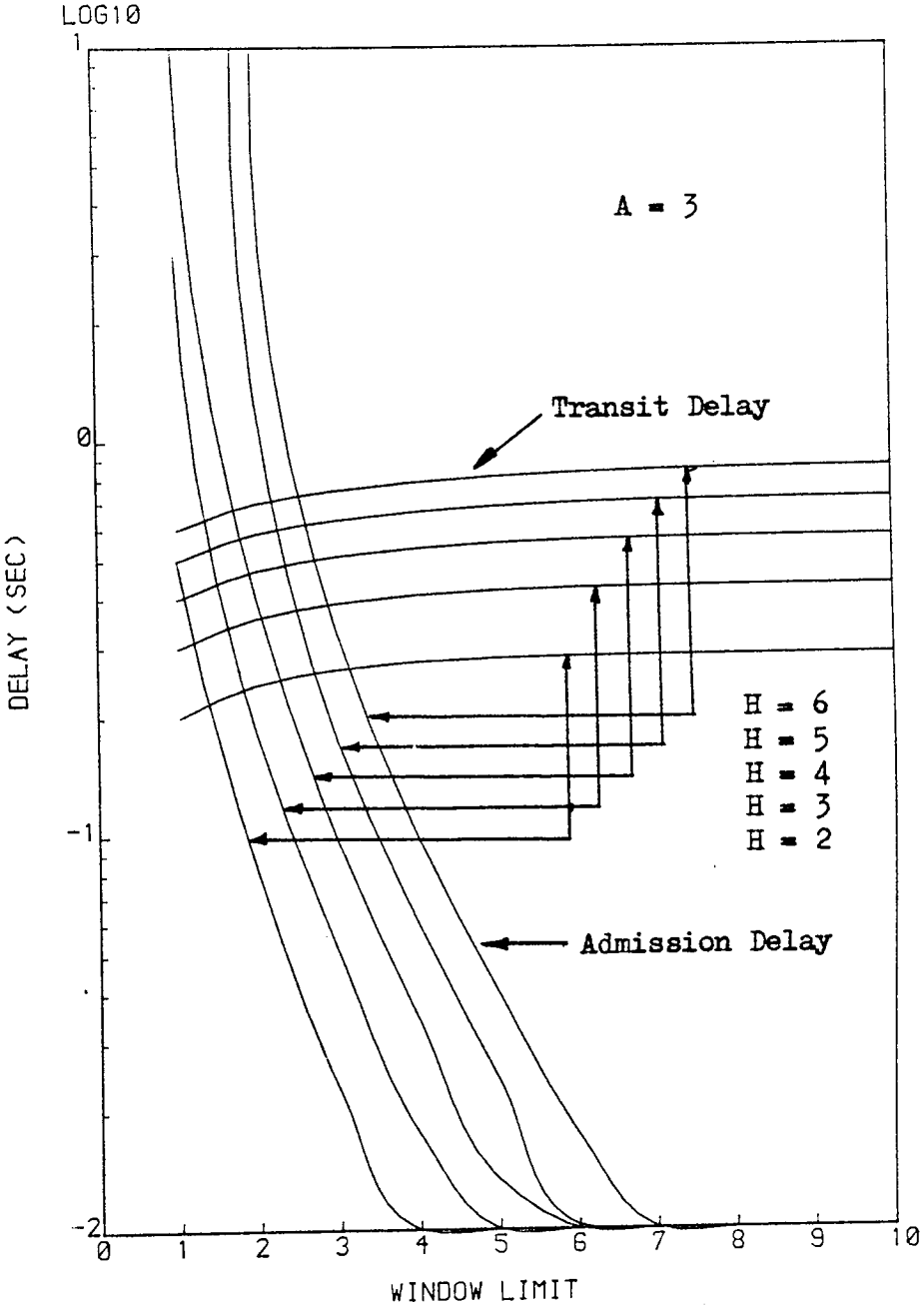


Fig. 5.6 Admission delay and transit delay as a function of window limit, for various hop lengths when the offered load is A = 3 messages/second.

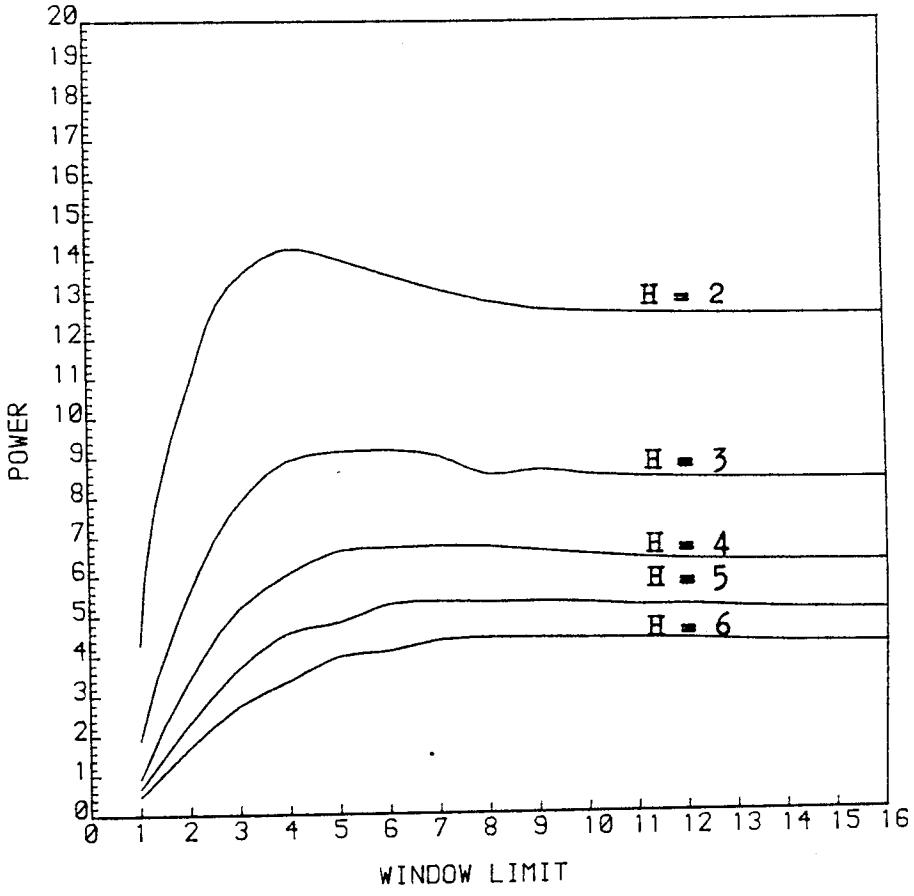


Fig. 5.7 Maximum power versus window limit for various hop lengths.

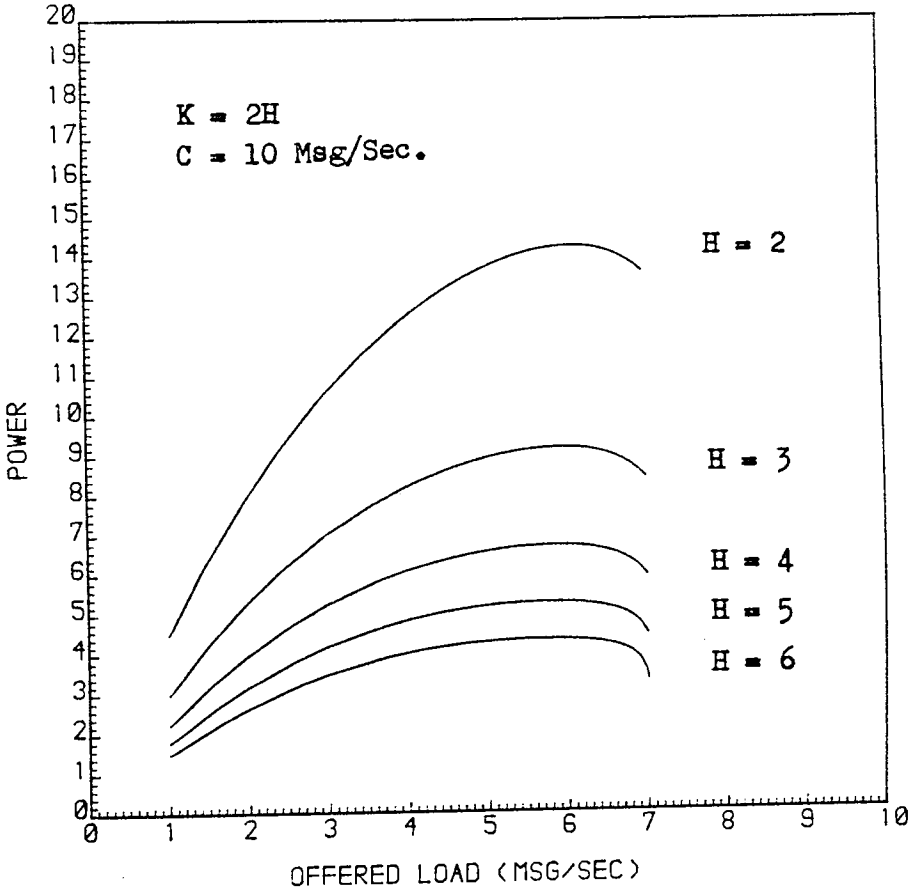


Fig. 5.8 Maximum power versus offered load for various hop lengths, when the window limit is equal to twice the hop length.

CHAPTER SIX

ADVANCED NETWORK ACCESS CONTROL TECHNIQUES

6. ADVANCED NETWORK ACCESS CONTROL TECHNIQUES

6.1. INTRODUCTION

In this chapter, certain advanced network access techniques are considered. These new access procedures are built on the experience obtained with the loss model networks of chapter 2,3 and 4, and the admission queue model of chapter 5. The new methods aim to realise the advantages of using admission queues at low or moderate load levels, while retaining the hierarchy of network controls needed to manage high levels of impressed load. The new access methods proposed in this chapter are assumed to augment the three levels of flow control already considered.

The access methods proposed in this chapter fall into two categories. The first improves the network performance at high loads, while the second deals with the sharing of virtual route resources between a number of users. The network models built using the new access methods are solved using the equivalent reduced network and the heuristic extensions to the MVA algorithm for multi chain networks (chapter 3).

6.2. THE BUFFER ACCESS QUEUE OR THE ARRIVAL RATE LIMITER

Consider the model of the network of fig 4.1, in which, messages arriving at the network are admitted only if they can secure a free token as well as a free buffer at the entry node. Otherwise they are assumed to be lost. Even if a

free token is available, an arriving message can be lost if free buffers are not available at the entry node. When a buffer does become available, a token at the head of the token queue has to wait for the arrival of the next message from the source. Hence, the tokens which are resources by themselves are inefficiently used. The network access control which restricts the number of buffers available at the entry node for external messages, increases the idle time of the tokens further. The net result is that, at low and moderate levels of offered load, fewer messages are accepted by the network than it can actually carry. The network throughput is lower than the offered load, even though the network has sufficient capacity to carry all the offered traffic, without any degradation in the network performance.

In the admission access queue method described in chapter 5, all messages denied access to the network are queued outside the network. The messages held in this queue on a FIFO basis are serviced as soon as a free token becomes available. This scheme was shown to operate most satisfactorily at low levels of offered load. However, at moderate and high loads, the access or admission queue brought about an early onset of saturation. Following saturation, the infinite access queue simply absorbed the excess demand and added unacceptable delay to the network. It follows that considerable development is required before the access queue can be incorporated in the network having hierarchical controls, or before it can be used in networks subject to large

impressed loads.

A method can be evolved which ensures the efficient use of tokens and which limits the loss of messages at light and moderate levels of offered load. This is not achieved by using an admission access queue, but ensuring that those messages which secure free tokens are not turned away because of blocking at the entry node. The method is to introduce a queue called the buffer access queue between the token queue and the entry node. Messages which have secured free tokens from the token queue wait in the buffer access queue until they are admitted to the network by the network access control. Only those messages which fail to secure free tokens are lost. The buffer access queue is large enough to hold all the messages that have secured tokens. The idea is illustrated in fig 6.1.

The addition of the buffer access queue increases the load on the virtual circuit (though the increase is less than that of the admission access queue). The buffer access queue can be thought of as a FIFO scheduler, where the time between transmission of messages waiting in this queue is drawn from a negative exponential distribution. The mean service rate Cb of the buffer access queue in the analytical model, is equal to this transmission rate. Since the link connecting the buffer access queue and the entry node is a high speed local link, the actual time to transfer a message is small and can be neglected. The service rate Cb of the

buffer access queue (i.e., the transmission rate of the scheduler), is an important parameter and must be chosen carefully. The model with the buffer access queue included, can be solved using the equivalent reduced network and the heuristic extensions to the MVA algorithm (chapter 3).

The performance of the network of fig. 4.1 when all entry nodes are equipped with buffer access queues is shown in fig. 6.2, for the specific instance when the buffer access queue service rate Cb is 100 messages per second. At low offered loads the virtual circuits have a throughput higher than that of the corresponding network without the buffer access queues. However, the throughput at low loads is not as high as might be expected using an external access queue. This is because, some messages still arrive when the token queue is empty and are lost. At moderate to high levels of offered load the network performance is similar to, but lower than, that of the corresponding network without the buffer access queues. This is because the buffering of messages in the high server rate buffer access queue increases the effective arrival rate of external messages at the entry node, thus overloading the virtual circuit even for moderate arrival rates. At very high levels of offered load, the network performance is similar to that of a network without the buffer access queues. In both cases, the grossly overloaded network operates in a degraded mode with reduced throughput under the influence of the hop level, end-to-end and network access controls.

The network performance can be can be held at its peak value if the service rate of the buffer access queue is kept small. Fig. 6.3 and 6.4 show the virtual circuit throughputs when the buffer access queue service rate is 10 and 5 messages per second. At low levels of offered load, the virtual circuit throughput is nearly equal to the source arrival rate when the service rate is 10 messages per second. However, when the service rate is 5 messages per second, the low load performance degrades. At high arrival rates, the virtual circuit throughputs saturate and remain constant. At a service rate of 10 messages per second, the network throughput does not deteriorate as the offered load increases. The saturation throughput is comparable with the peak throughput achieved in the corresponding network without the buffer access queues. If the service rate is reduced to 5 messages per second, the throughput is again held constant for high impressed loads, although the resulting performance of the network is some what reduced.

At high arrival rates, the token queue is always empty, since any token returning to the token queue is quickly picked up by an arriving message. The tokens are now held by messages in the buffer access queue, waiting for buffers to become free at the entry node. In effect, the token queue disappears, leaving only the buffer access queue whose service rate may be chosen as a control parameter. Hence, for large arrival rates the network becomes independent of the source. Instead, the network sees a source with an arrival

rate equal to the service rate of the buffer access queue. If the service rate of the access queue is chosen carefully, the network can be made to deliver all the offered traffic at low loads. As the offered load increases, the throughput also increases reaching a maximum. Thereafter, it remains constant and does not fall even at high loads. In effect, the buffer access queue with correctly chosen service rate, acts as a source arrival rate limiter giving optimum control performance. In the case of the network with a window limit of five messages per virtual circuit (K=5), and the input buffer limit set at three (NT=5, NE=3), a service rate of 10 messages per second is found to be optimum. This adds a fourth level of control to the network, namely the arrival rate limiter. The control aids the end-to-end and network access flow control mechanisms, in achieving near optimum control.

6.3. THE TOKEN ACCESS QUEUE

In any real system, messages wait outside the network until they secure a free token. In the model under study such messages were assumed to be lost. The effect of buffering these messages outside the network is to increase the effective arrival rate of messages to the network. In the model with the buffer access queue (sec. 6.2), messages that find the token queue empty on their arrival are lost. Tokens that return to an empty token queue, have to wait for messages to arrive from the source before they can be picked

up. However, if those messages that do not secure a free token wait outside the network (in a queue called the token access queue), then free tokens returning to the token queue can be picked up immediately by the messages waiting in the token access queue. This reduces the idle time of the free tokens in the token queue further. With the addition of the infinite token access queue, no message is lost for want of network resources.

A queueing model for the admission queueing process of virtual circuits controlled by end-to-end flow control mechanism, when messages arrive from independent Poisson sources was developed in chapter 5 . The effect of buffering messages outside the network was to increase the effective arrival rate of messages to the transit network. The effective arrival rates were solved numerically using an iterative procedure.

Consider the closed queueing network model with R virtual circuits with the three levels of flow control (chapter 2 and 3). The network is assumed to have a buffer access queue at each entry node. Messages arriving from Poison source r(r=1,2,...,R) wait in the token access queue until they secure a free token. The time spent by a token at the head of the token queue is assumed to be exponentially distributed, and the token access queue is modelled as an M/M/1 queue (chapter 5). With the addition of the token access queue, the mean time spent by a token at the head of the rth

token queue, r=1,2,...,R, is given by,

$$Htr = Tar + Par(0)/Ar, \tag{5.3}$$

where,

Ar — The mean rate at which chain r messages are generated by chain r sources, r=1,2,...,R;

Tar — The mean time required to remove a token from the rth token queue and attribute it to a message in the token access queue;

Par(0) — The probability of the rth token access queue being empty.

Par(0) is given by,

$$Par(0) = 1 - Ar\ Har, \tag{5.8}$$

where, Har is the mean time spent by a message at the head of the rth token access queue. Har is given by,

$$Har = Ttr + Ptr(0)/Vr, \tag{5.7}$$

where,

Ttr The mean time required to remove a message holding a token from the token access queue to the network;

Vr The throughput of chain r;

Ptr(0) The probability of the rth token queue being empty.

Ptr(0) is given by,

$$Ptr(0) = 1 - Vr\ Htr. \qquad (5.4)$$

These four equations are interdependent. For a given set of arrival rates Ar (r=1,2,...,R), they can be solved iteratively by assuming Par(0)=1 initially, and evaluating the values Vr and Ptr(0). From these approximate values, an approximation for Par(0) can be found. The process can be repeated until convergence occurs.

The admission queue process can now be considered for the finite buffer multi-chain network model, by modifying the mean service rate of token queues r(r=1,2,...R). The mean service rate Htr of the rth token queue is now given by (5.3). The iterative process required to find Htr (r=1,2,...,R), can be incorporated in the heuristic extension to the MVA algorithm (sec. 3.4.1). The algorithm to find Htr is summarised below.

1] Assume Par(0)=1, and find the initial value of Htr (r=1,2,...,R), using (5.3).

2] Find Vr (r=1,2,...,R), using the heuristic extensions to the MVA algorithm, as discussed in section section 3.4.

3] Find Ptr(0) from (5.4), using the computed values for Htr and Vr from step 1 and 2, for r=1,2,...,R.

4] Find Har (r=1,2,...,R), from (5.7), using the values of Ptr(0) and Vr from step 3 and 2.

5] Find Par(0) (r=1,2,...,R), from (5.8), using the values of Har computed from step 4.

6] Find the new values for Htr (r=1,2,...,R), from (5.3), using the values for Par(0) computed from step 5.

7] Repeat steps 2 to 6 until the probabilities Par(0) and Ptr(0) converge for all r=1,2,...,R.

The model with the token access queue can be solved as out-lined above, the determination of the effective arrival rates being incorporated in the algorithm for determining the mean throughput of the virtual circuits. The resulting network performance would be that of a wait model, which has taken into account, the admission queueing process.

The performance of the network in fig. 5.1 are shown in fig. 6.5 for the specific case when the token access queues

have a mean service rate of 100 messages per second and the buffer access queues have a mean service rate of 10 messages per second. For low arrival rates, the virtual circuit throughputs are nearly equal to the offered load. But they reach their maximum quickly and thereafter remain constant for all offered loads. The saturation throughputs are reached for loads very much lower than the case where, only the buffer access queue is present. The figure also shows the virtual circuit performance of the network with buffer access queues, but without the token access queues. The saturation throughputs of the network are the asymptotic throughputs of the network with the buffer access queues but without the token access queues.

Table 6.1 shows the percentage improvement in the network throughput for different offered loads, when the buffer access queue has a service rate of 5, 10 and 100 messages per second. The comparisons are made with respect to the throughput of the network without the token access queues, and without the buffer access queues. With a service rate of 10 messages per second for the buffer access queues, the network with the token access queues shows an improvement in throughput compared to the network without the token access queues and the buffer access queues. At low loads, the throughput performance is enhanced by over 5%, and at high loads by over 10%. The performance of the network is only equaled by that of the network without the buffer access queues, at the latters peak performance (when the load is 10

Table 6.1

Percentage change in network throughput

with the addition of the token access queue

Offered load per VC	Percentage change in network throughput		
	Buffer access queue service rate		
	100 mps	10mps	5mps
2	2.12%	2.12%	2.17%
5	-10.92%	7.72%	0.10%
10	-19.90%	0.23%	-7.98%
20	-14.00%	5.12%	-2.70%
30	-9.54%	8.88%	1.35%
40	-6.57%	11.35%	4.02%
50	-4.43%	13.13%	5.95%

Percentage changes are with respect to the corresponding network without the token access queue and the buffer access queue.

messages per second). While the token access queue improves the network performance at low and moderate loads, the buffer access queue ensures that the network is not overloaded by the token access queue, as well as by large impressed loads.

6.4. PRIORITY ACCESS OF VIRTUAL CIRCUIT TOKENS

The models so far assumed that each virtual route had only one virtual circuit, with a single source destination pair. However, several sources or users could be sharing the same virtual route and its tokens. This raises the problem of sharing the virtual route resources between two sources, and becomes of particular interest if one source has priority. This could equally well be interpreted as having one source with two classes of messages, a priority class and a non-priority class. Various schemes for sharing of the virtual route capacity between a number of input sources are developed and discussed in the following subsections.

6.4.1. PREEMPTIVE PRIORITY ACCESS PROTOCOL

The first scheme is based on a priority access scheme. Consider the general queueing network model of chapter 2 and 3. Each virtual circuit is assumed to have two input sources, sharing the virtual route capacity as well as the tokens. One of the sources has preemptive priority over the other, in so far as securing tokens are concerned. Fig. 6.6 shows the network, and the two sources belonging to virtual

route r(r=1,2,...,R), along with its virtual route token queue and the buffer access queue. The sources are designated as Sr1 and Sr2. Messages from source Sr1 have priority over messages from source Sr2 in securing tokens. The mean arrival rate of messages from the two sources are Ar1 and Ar2 respectively.

Once the priority message enters the actual network, it is treated as any other message. That is, it is not given preferential treatment at the entry node or at any other transit node within the network. If a non-priority message (i.e., a message belonging to source Sr2) finds the token queue empty on its arrival, it is assumed to be lost. If it secures a free token on its arrival, it waits in the FIFO buffer access queue until it is admitted to the entry node. If a priority message (i.e, a message from source Sr1) on its arrival secures a free token, it enters the FIFO buffer access queue and waits just as the non-priority message does. However, if a priority message arrives to find the the token queue empty, one of two things happen. If the buffer access queue has non-priority messages waiting for service, the priority message takes the place of the first non-priority message in the order of service (and its token). The non-priority message is lost. However if the buffer access queue does not contain any non-priority message, the priority message is lost. Hence a priority message that finds the token queue empty on its arrival can still be admitted to the network if, there is a non-priority message

(with a token) in the buffer access queue waiting for service.

Let Vr(K) be the throughput of the virtual route r(r=1,2,...R), when both sources are non-priority sources and their arrival rates are Ar1 and Ar2. Then the throughput of each individual source is given by,

$$Vr1(K) = Ar1\ Vr(K)/(Ar1 + Ar2),$$
$$Vr2(K) = Ar2\ Vr(K)/(Ar1 + Ar2). \qquad (6.1)$$

Where, K=K1,K2,...,KR is the population vector, and Vr1(K) and Vr2(K) are the throughput of source Sr1 and Sr2 respectively (when no priority is exercised). If messages from source Sr1 have preemptive priority over messages from source Sr2 in securing tokens, then a fraction "g" of the non-priority source throughput is actually made up of priority messages. The throughput of the two sources are now given by,

$$Vrp(K) = Vr1(K) + g\ Vr2(K)\ ,$$
$$= (\ Ar1 + g\ Ar2)\ Vr(K)/(Ar1 + Ar2),$$

and

$$Vrn(K) = Vr2(K) - g\ Vr2(K)$$
$$= Ar2\ (1 - g)\ Vr(K)/(Ar1 + Ar2). \qquad (6.2)$$

Where,

Vrp Throughput of messages from source Sr1, when their arrival rate is Ar1;

Vrn Throughput of messages from source Sr2, when their arrival rate is Ar2;

g Probability that a priority message on its arrival finds the token queue empty, and at the same time, finds a non-priority message waiting for service in the buffer access queue.

That is,

$$g = P(\text{ a priority message on its arrival finds the token queue empty }) \times P(\text{ there is at least one non-priority message in the buffer access queue waiting for service.})$$

$$= [Ar1/(Ar1 + Ar2)]\ (Pt(0))\ (1 - Pan(0)), \qquad (6.3)$$

where,

Pt(0) Probability of the token queue being empty.

Pan(0) Probability that the buffer access queue does not contain non-priority messages waiting for service.

Now,

$$Pt(0) = 1 - [Vr(K)/(\ Ar1 + Ar2)]\ , \tag{6.4}$$

and is independent of the priority scheme. Further,

$$\begin{aligned} Pan(0) = {} & P(\text{there are no non-priority messages} \\ & \text{in the buffer access queue}) \\ & + P(\text{there is one non-priority message} \\ & \text{in the buffer access queue undergoing service}). \end{aligned}$$

That is,

$$Pan(0) = \sum_{Np=0}^{Kr} Par(0,Np,K) + \sum_{Np=0}^{Kr-1} (1/(1+Np))\ Par(1,Np,K). \tag{6.5}$$

Where, Par(Nn,Np,K) is the probability of finding Nn non-priority messages and Np priority messages in the buffer access queue associated with virtual route r, when the population vector is K. But Par(Nn,Np,K) itself depends on the relative throughputs of the priority and non-priority message streams, and hence on g. The queue length probabilities of the buffer access queue associated with the rth virtual route can be found from its equivalent reduced network. The queue length probability Par(Nn,Np,K=Kr) is given by the recursive relation (equation 2.42 and 2.43):

$$Par(Nn,Np,Kr) = \{ Vrn(Kr)/(a\ C'r) \}\ Par(Nn-1,Np,Kr-1) + \{ Vrp(Kr)/(a\ C'r) \}\ Par(Nn,Np-1,Kr-1),$$

Using (6.2), Par(Nn,Np,Kr) can be written as,

$$\begin{aligned} Par(Nn,Np,Kr) = & \left[\{(1-g)\ Ar2/(Ar1 + Ar2)\}\ \{Var(Kr)/(a\ C'r)\} \times Par(Nn-1,Np,Kr-1)\right] \\ & + \left[\{(Ar1 + g\ Ar2)/(Ar1 + Ar2)\}\ \{Var(Kr)/(a\ C'r)\} \times Par(Nn,Np-1,Kr-1)\right]. \end{aligned} \tag{6.6}$$

This equation depends on g, and can be solved iteratively by starting with g=0. The iterative procedure is summarised below.

1] Form the equivalent reduced network of buffer access queue r(r=1,2,...,R), (section 3.4).

2] Solve the network and determine the throughput of virtual circuit r for all population $0 \leq Yr \leq Kr$, using the MVA algorithm (section 2.5).

3] Set g=0.

4] Determine the marginal queue length probabilities of the buffer access queue, using (6.6) and the throughputs from step 2

5] Determine g, using the marginal queue length probabilities found in step 4, and equations (6.3-6.6).

6] Repeat step 4 and 5 until the factor g converges.

The numerical results for the five node network of fig. 4.1 now equipped with preemptive priority access of virtual route tokens, is discussed. Each virtual route is assumed to have a priority source and a non-priority source. The improvement in the priority traffic throughput for a given virtual route token limit, depends on the arrival rates of the priority and non-priority message streams and on the service rate Cb, of the buffer access queue. Fig. 6.7 shows the change in throughput of the priority and non-priority traffic streams, as a function of total offered load, for two different arrival ratios and two different buffer access queue service rates. For a given buffer access queue service rate, the improvement in the throughput of the priority traffic (compared to its throughput when preemptive access for tokens is not employed) is higher, when the ratio of priority traffic (P1) to total traffic is low. Further, for a given traffic ratio, the percentage improvement in the throughput of the priority message stream is higher, for lower buffer access queue service rates. From fig. 6.7, the maximum improvement in the throughput of the priority stream is nearly 70 % when the priority traffic constitutes 10% of the total arrival traffic and the buffer access queue service rate Cb is 5 messages per second. This improvement

falls to 50% when the service rate of the buffer access queue is increased to 10 messages per second. When the priority traffic constitutes 50% of the total arrival traffic, the corresponding improvement in its throughput are 38% and 28% respectively. Table 6.2 shows the percentage change in the throughput of the two streams when the buffer access queue has a service rate of 5 and 10 messages per second, when the priority traffic constitutes 10% of the offered load. All comparisons are made for a network with the buffer access queues but without the priority access scheme.

Fig. 6.8 shows the percentage of priority messages that are normally blocked, but succeed in geting through the network with the introduction of the priority access scheme. This percentage is a maximum when both the arrival rate of priority traffic and the service rate of the buffer access queue are low. The percentage increases as the total offered load increases, reaches a maximum (when the total offered load on each virtual route is 3 messages per second), and then begins to decrease. The percentage reaches a maximum before the virtual circuits attain their peak throughput. Fig. 6.9 shows the performance for two fixed arrival rates of priority traffic, as the non-priority traffic is varied. Once again it can be seen that the percentage improvement in the throughput of the priority stream is higher, for lower arrival rates of priority traffic and lower service rate of the buffer access queue.

Table 6.2

Change in virtual circuit throughput with priority access of virtual route tokens. Percentage changes are with respect to the network without priority access but with buffer access queue

Offered load per VC	Buffer access queue service rate			
	10 msg/sec		5 msg/sec	
	Change in priority & non priority stream throughput			
	Pr1	Pr2	Pr1	Pr2
2	0.24%	-0.03%	1.05%	-0.12%
5	9.95%	-1.10%	21.72%	-2.41%
10	29.99%	-3.33%	46.75%	-5.19%
20	43.26%	-4.8%	61.47%	-6.83%
30	47.74%	-5.30%	66.44%	-7.38%
50	51.30%	-5.70%	70.42%	-7.82%

Table 6.3 shows the percentage improvement in the throughput of the priority stream, when it constitutes 10% of the total offered load, and, the service rate of the buffer access queue is 10 and 5 messages per second. The percentage improvement is with respect to the throughput of the network without the buffer access queue and without the priority access scheme. This table shows that, with the introduction of the buffer access queue and the priority access of virtual route tokens, the throughput of the priority stream can be improved by 30% at moderate loads (total load is 10 messages per second) and by 75% when for higher loads. Even with the introduction of priority access, the throughput of the non-priority stream is better, compared to its throughput when no buffer access queue is present. At moderate loads they show an improvement of 1%, while at higher loads the improvement is nearly 10%. However, when the service rate of the buffer access queue is 5 messages per second, the improvement at low and moderate loads is reduced. This is because, with a service rate of 5 messages per second for the buffer access queue, the network throughput is lower than that of the corresponding network without the buffer access queue (fig. 6.3). However, at higher loads the throughput of the priority traffic stream improves by as much as 82%, but the non-priority traffic stream throughput is reduced by about 2%. It must be noted that the buffer access queue is an essential part of the priority access scheme.

Table 6.3

Change in virtual circuit throughput with priority access of virtual route tokens. Percentages changes are with respect to the network without priority access and buffer access queue

Offered load per VC	Buffer access queue service rate			
	10 msg/sec		5 msg/sec	
	Change in priority & non priority stream throughput			
	Pr1	Pr2	Pr1	Pr2
2	1.05%	0.77%	0.15%	-1.00%
5	13.25%	1.86%	12.75%	-9.60%
10	32.24%	-1.66%	34.72%	-12.96%
20	52.76%	1.50%	57.31%	-9.23%
30	63.55%	4.82%	69.02%	-5.94%
50	75.23%	9.21%	82.00%	-1.77%

Hence, the priority access scheme is a simple method for giving preferential treatment to one class traffic over the other, as far as the virtual route resources are concerned. The improvement in the throughput of the priority stream can be considerable, when the ratio of priority traffic to non-priority traffic is low. The method is also easy to implement.

6.4.2. LOCAL ACCESS CONTROL

In this scheme the virtual route tokens are accessed by the sources, through a local network. Consider the general queueing network model with finite buffers and three levels of flow control (chapter 2 and 3). Each virtual route r(r=1,2,...,R) is assumed to have two sources Sr1 and Sr2. Each of these two sources have a local pool of Kri (i=1,2) tokens. A message generated at source Sri, must first pick up a local token from its local token queue. The message then waits in the virtual route token access queue, common to both the sources. A message at the head of the token access queue has to wait until it can pick up a free token from the token queue of the virtual route. Once the message enters the network, it releases the local token, which is then returned to its local token queue. It is assumed that both sources compete for the tokens belonging to a common virtual route. The scheme is illustrated in fig. 6.10. By varying the local token ratio Kr1/Kr2, the throughput of each source can be controlled relative to the

other, when the virtual route token limit is fixed. The network model with the local access control can be solved using the technique developed for the token access queues developed in section 6.3.

The numerical results for the five node network of fig. 4.1 now equipped with the local access control is presented. Fig. 6.11 shows the network throughput as a function of offered load, for various local token limits. There are two sources associated with each of the four virtual routes, and each virtual route has five tokens. The arrival rates from the two sources are equal and the sources compete for the virtual route tokens. When the local window limits are equal, the throughput of the two sources are also equal. As the local token limit of source Sr1 is increased, while that of the other is kept constant, the throughput of source Sr1 can be improved relative to that of source Sr2. Hence, by varying the ratio Kr1/Kr2 the relative throughputs of the two sources can be controlled. In fact the throughput ratio Vr1/Vr2 (r=1,2,3,4), is found to be equal to the local token limit ratio Kr1/Kr2 (for equal source arrival rates). Here, Vr1 and Vr2 are the throughput of source Sr1 and Sr2 respectively.

This method can also ensure greater access of virtual route tokens for low volume priority traffic. This is achieved by first chosing the local token limit for the non-priority source, which has to be large enough to utilise

the virtual route capacity. The local token limit for the priority source is then made sufficiently large, compared to the local token limit of the non-priority source. For a given level of priority traffic, its throughput should reduce as the non-priority traffic increases. However, by chosing a large local token limit for the priority source, this fall in throughput can be reduced (fig. 6.11).

Table 6.4 shows the percentage improvement in the throughput of priority traffic when its window limit is 4 and 8. The local token limit for the non-priority source is 2 and the priority traffic constitutes 10% of the total offered load. The comparisons are made with respect to the throughput obtained when the two local token limits are 2. It can be seen that the throughput of one of the stream can be improved at the expense of the other, by varying the local token limit ratio.

Table 6.5 shows the percentage improvement in the throughput, with respect to the throughput of a network without the local access network and without the buffer access queue. The priority traffic constitutes 50% of the total offered traffic. Even when the local window limits are equal, there is an improvement in throughput. This is due to the presence of the token access queue and the buffer access queue, which utilises the virtual route tokens more efficiently. By using different local token ratios, large improvement in the throughput of one source relative to the

Table 6.4

Change in priority source throughput with local access control. Percentage changes are with respect to throughputs when local token ratio is 2:2

Offered load on VR	Change in priority traffic throughput	
	Local network token ratio	
	4:2	8:2
2	0.00%	0.00%
5	1.94%	1.99%
10	14.84%	17.55%
20	32.26%	53.53%
30	39.62%	72.93%
50	40.94%	77.25%

Table 6.5

Change in priority source throughput with local access network. Percentage changes are with respect to network without local access control and buffer access queue

Offered load per VR	Percentage change in priority source throughput		
	local network token ratio		
	2:2	4:2	8:2
2	2.15%	2.78%	2.78%
5	8.69%	23.02%	33.51%
10	0.69%	29.25%	55.70%
20	5.7%	39.75%	68.17%
30	9.88%	45.98%	75.34%
50	15.18%	53.37%	84.12%

other source can be obtained.

6.4.3. PRIORITY VIRTUAL CIRCUITS

In the last network access control scheme discussed here, priority sources have independent virtual circuits. These virtual circuits do not have any buffer limits imposed by the network access control mechanism at the entry node. The priority messages are treated like transit messages at the entry node. The priority virtual circuits have their own pool of tokens to gain access to the network. However, they compete for other virtual route resources like the buffers and the channel capacities, which they have to share with other traffic streams on the virtual route. The priority traffic is given preferential treatment only at the entry node, by exempting it from the network access control.

The numerical results for the five node network of fig. 4.1 are presented. In the example discussed here, each virtual route is assumed to have a priority and a non-priority virtual circuit, with a token limit of 3 for each virtual circuit. The network access control limit is set at two (NE=2) to avoid store-and-forward deadlock. The offered load on the priority virtual circuit constitutes 10% of the total virtual route load. The model is solved using the heuristic method developed in chapter 3. Fig. 6.12 shows the virtual circuit throughput as a function of offered load. With the removal of the network access control on the priority virtual circuit, its throughput can be improved

Table 6.6

Change in priority virtual circuit throughput. Comparisons are with respect to network with network access control for priority source

Total load per VR	Change in virtual circuit throughput	
	Pr1	Pr2
2	2.86%	-0.03%
6	13.5%	-1.31%
10	20.80%	-3.64%
20	29.18%	-11.16%
30	27.38%	-13.11%
50	14.64%	-24.30%

relative to that of the non-priority stream. Table 6.6 shows the percentage change in the priority and non-priority virtual circuit throughput, when the priority virtual circuit is exempted from network access control. All comparisons are made with respect to the network where the priority virtual circuit is also subjected to network access control. However, the maximum load on the priority virtual circuit must be limited, and its token limit must be carefully chosen, to prevent the network from being over-loaded.

6.5. CONCLUSIONS

A number of advanced access techniques to improve the network performance as well as that of selected traffic streams were developed. These access methods augment the three flow control methods already considered (in chapter 2,3 and 4). The network equipped with these new access methods were solved using the heuristic method developed in chapter 3. The introduction of the buffer access queue with correctly chosen service time parameter, stabilises the network performance and holds the network throughput at its peak value, even at high loads. The buffer access queue acts as a source arrival rate limiter, when the source arrival rate exceeds the buffer access queue service rate. The addition of the token access queue transforms the model from a loss model to a wait model. The token access queue improves the network performance at low loads, while the buffer access queue ensures that the network is not over

loaded at high loads.

The schemes based on the priority access of virtual circuit tokens, improve the throughput of selected traffic streams at the expense of other traffic streams. The scheme based on the preemptive access of virtual circuit tokens is simple and easy to implement. However, the throughput of the priority stream is high only when the ratio of priority traffic to non-priority traffic is low. The scheme based on the local token access control, gives better control over the throughput of the two sources without affecting the total virtual circuit throughput. By simply varying the local token ratio, the throughput of the two streams can be controlled over a wide range. The last scheme that exempts priority virtual circuits from network access control provides a maximum improvement in throughput of about 30%. However, this method can be implemented only when the volume of priority traffic is very low, so that the absence of the network access control on the priority virtual circuits does not overload the network and lead to network performance degradation and deadlocks.

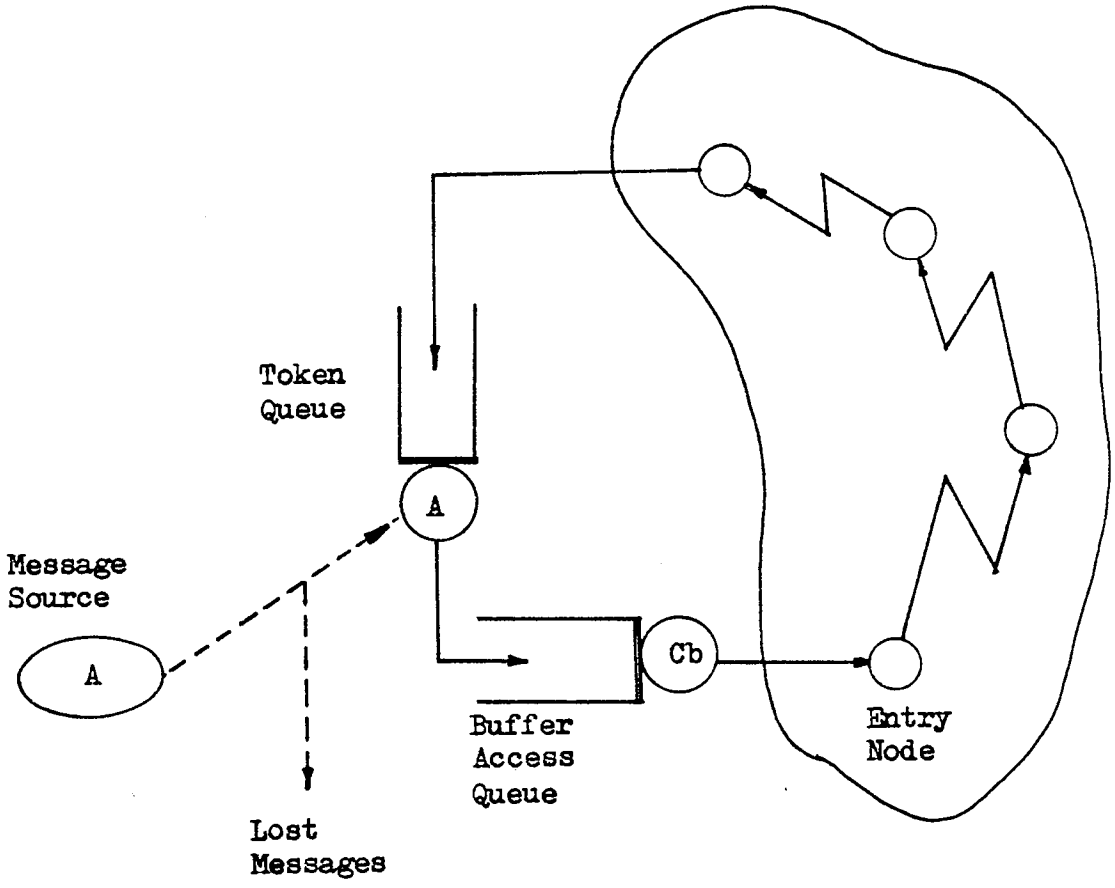
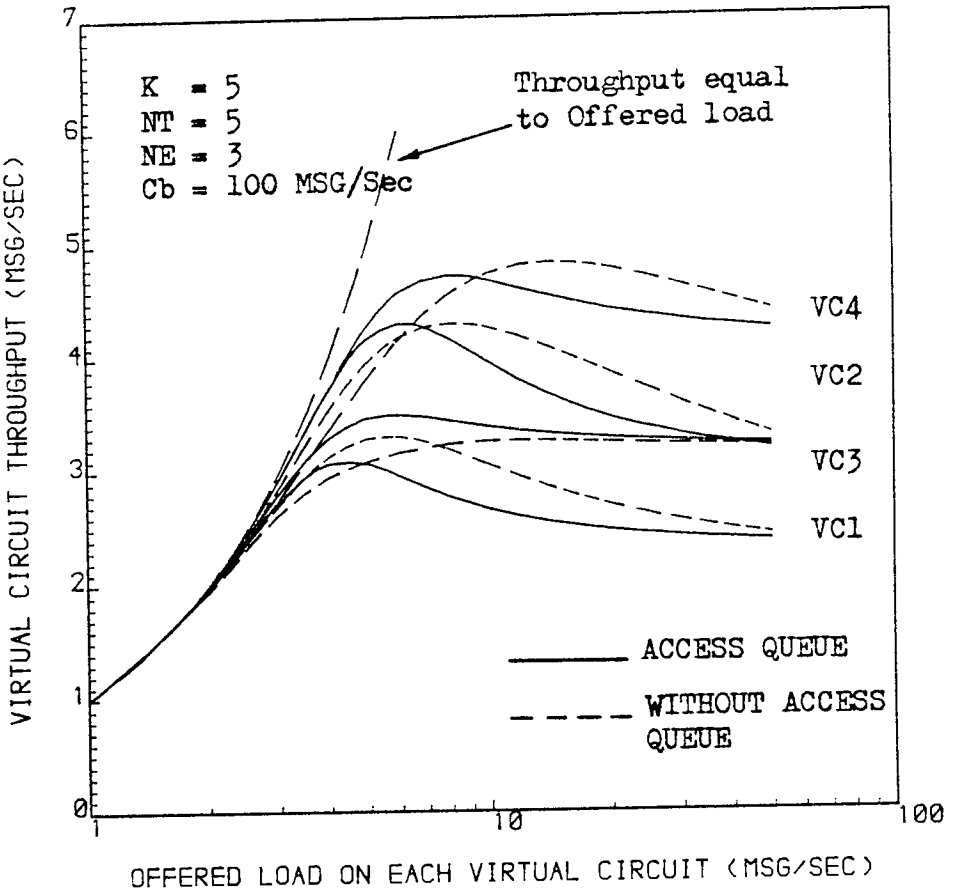


Fig. 6.1 Queueing model of a network with buffer access queue.



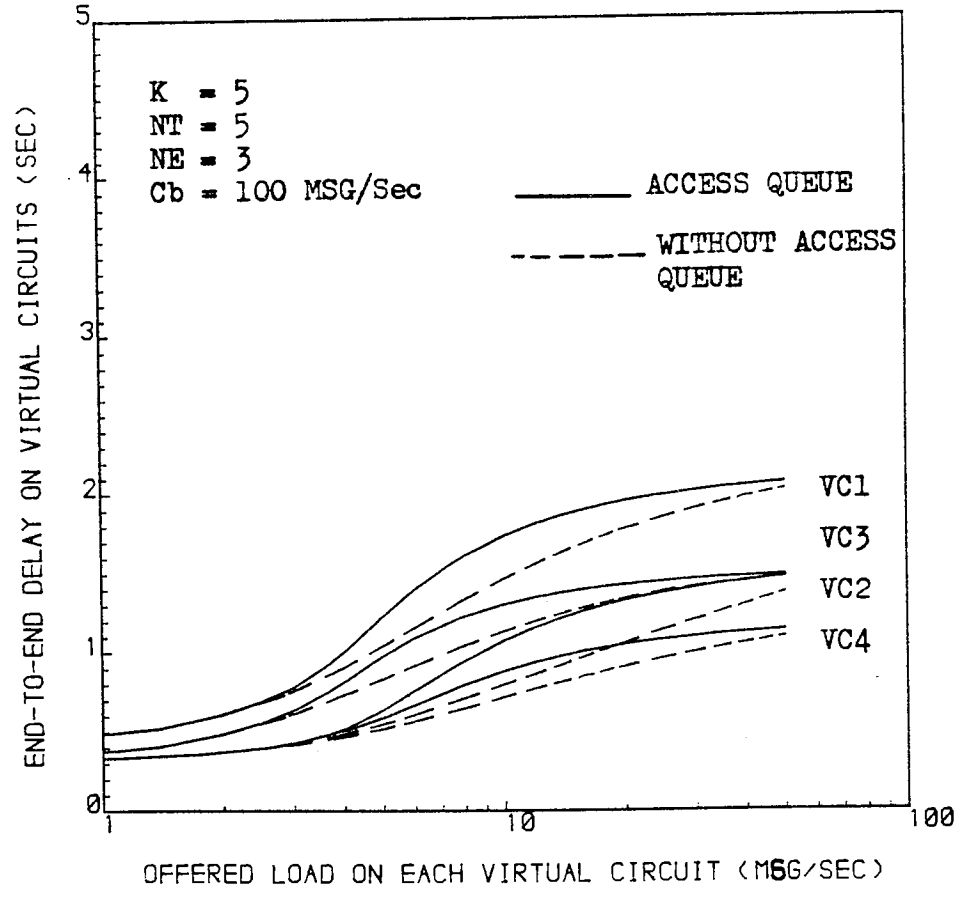
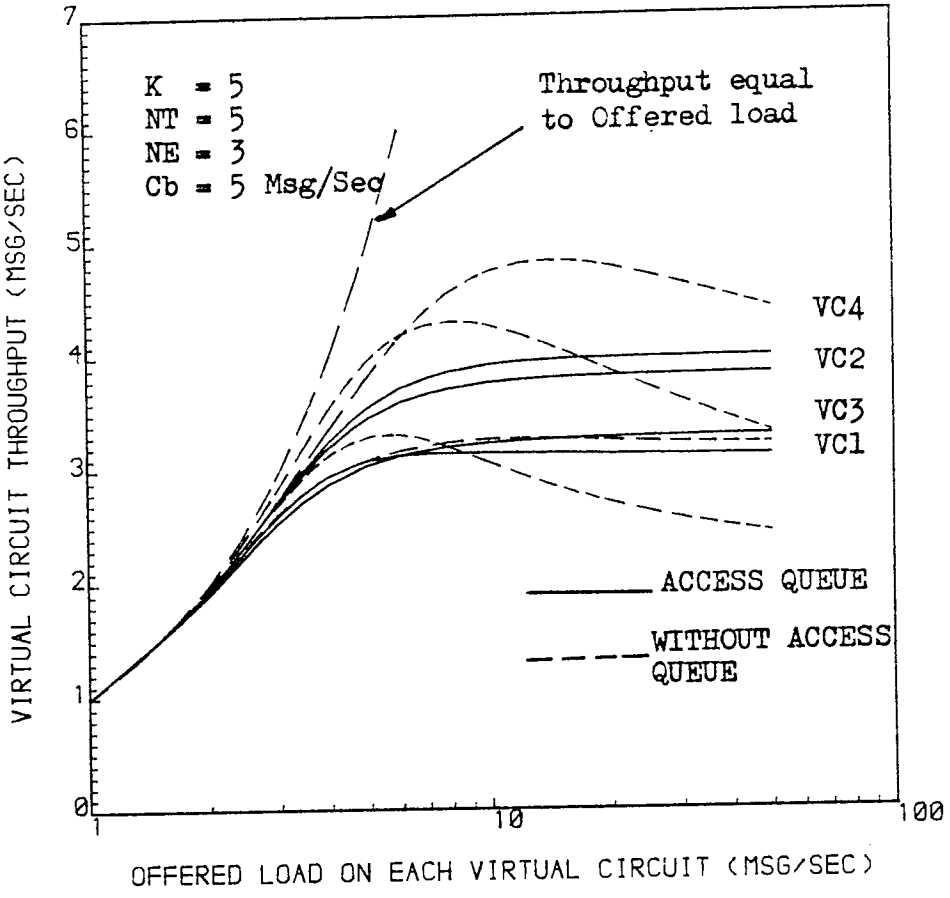


Fig. 6.2 Virtual circuit performance with buffer access queue with a service rate of 100 messages per second (K = 5, NT = 5 and NE = 3).



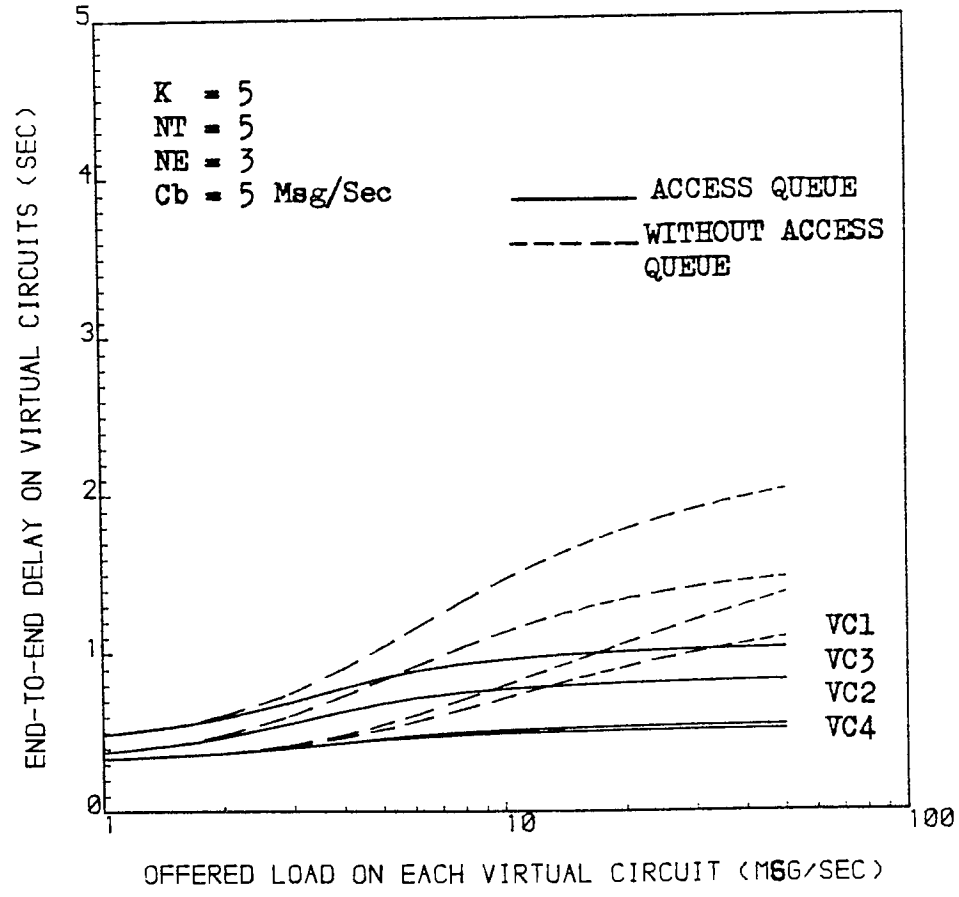
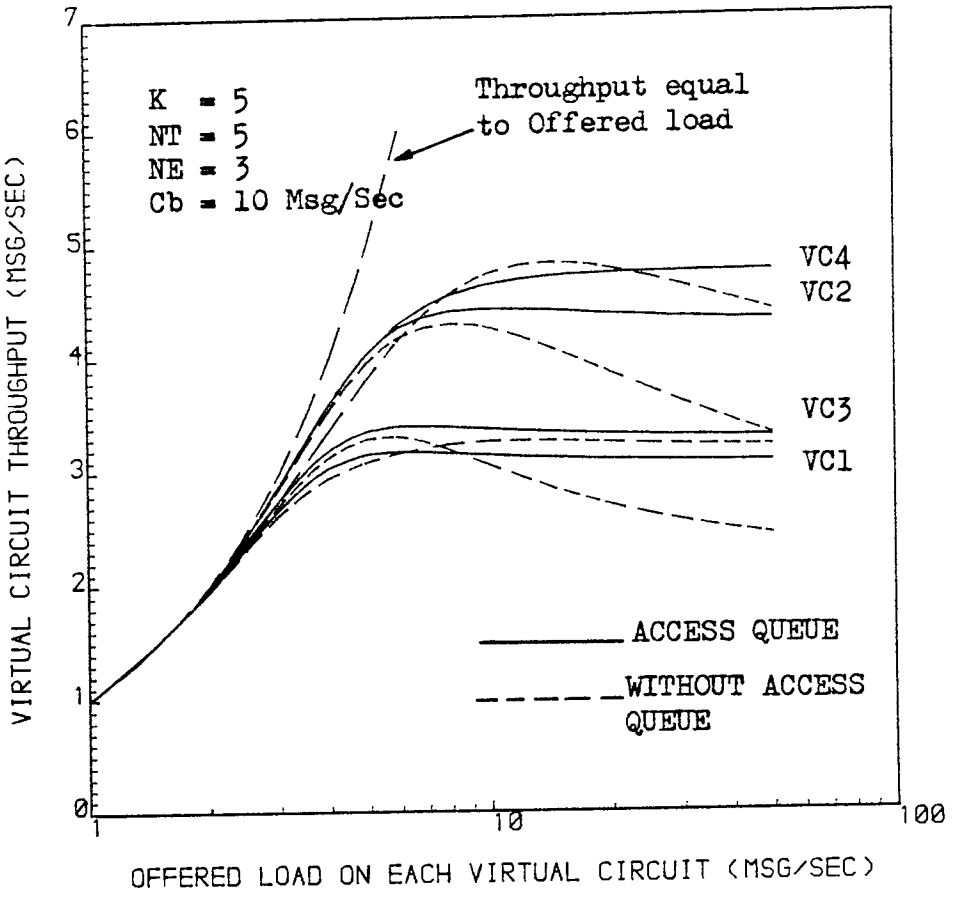


Fig. 6.3 Virtual circuit performance with buffer access queue with a service rate of 5 messages per second (K = 5, NT = 5 and NE = 3).



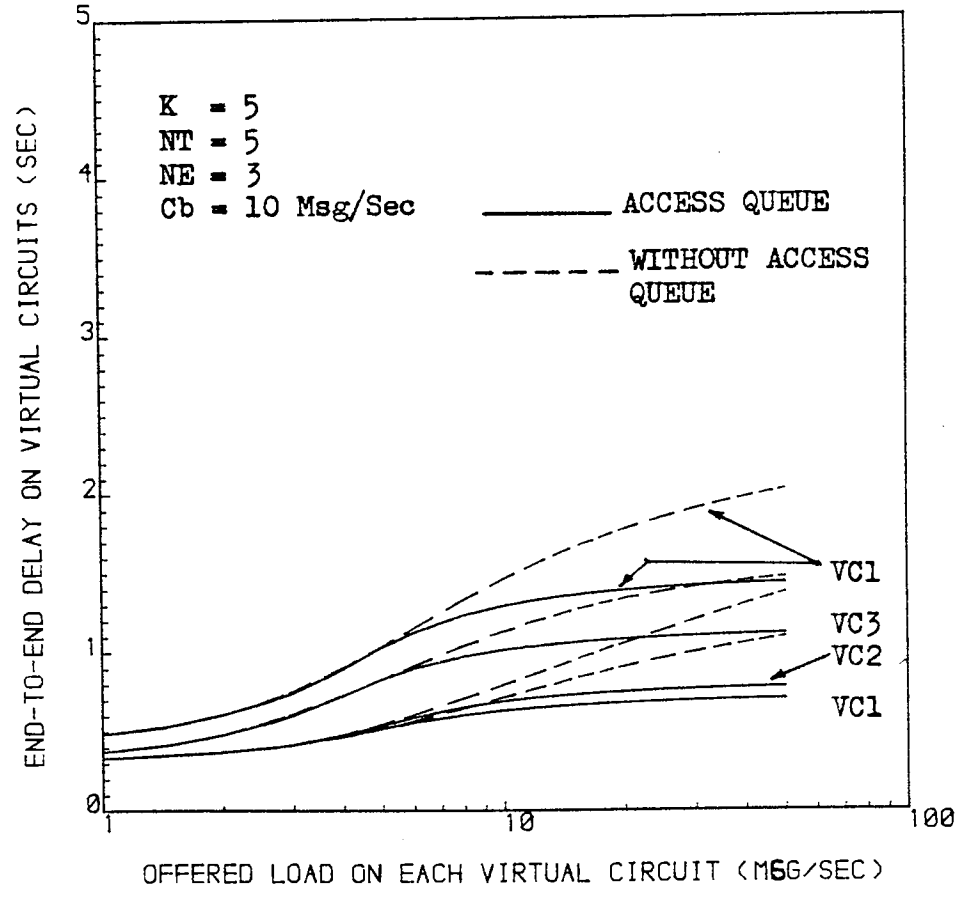
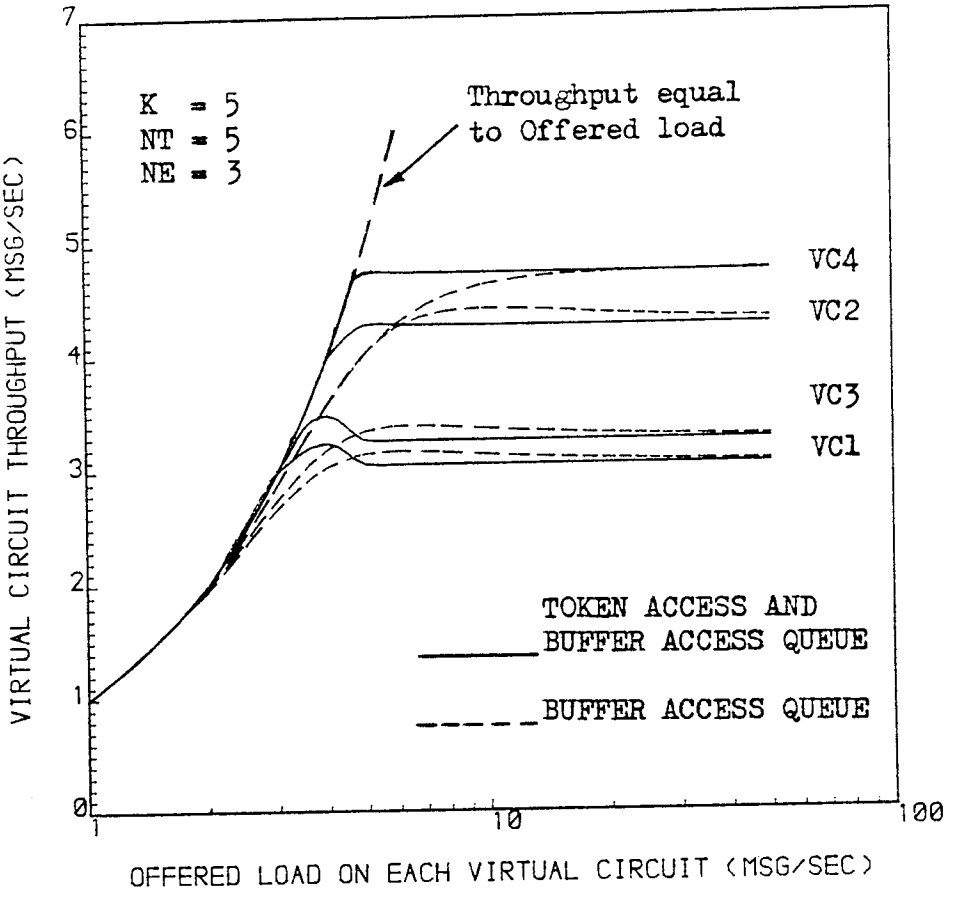


Fig. 6.4 Virtual circuit performance with buffer access queue with a service rate of 10 messages per second (K = 5, NT = 5 and NE = 3).



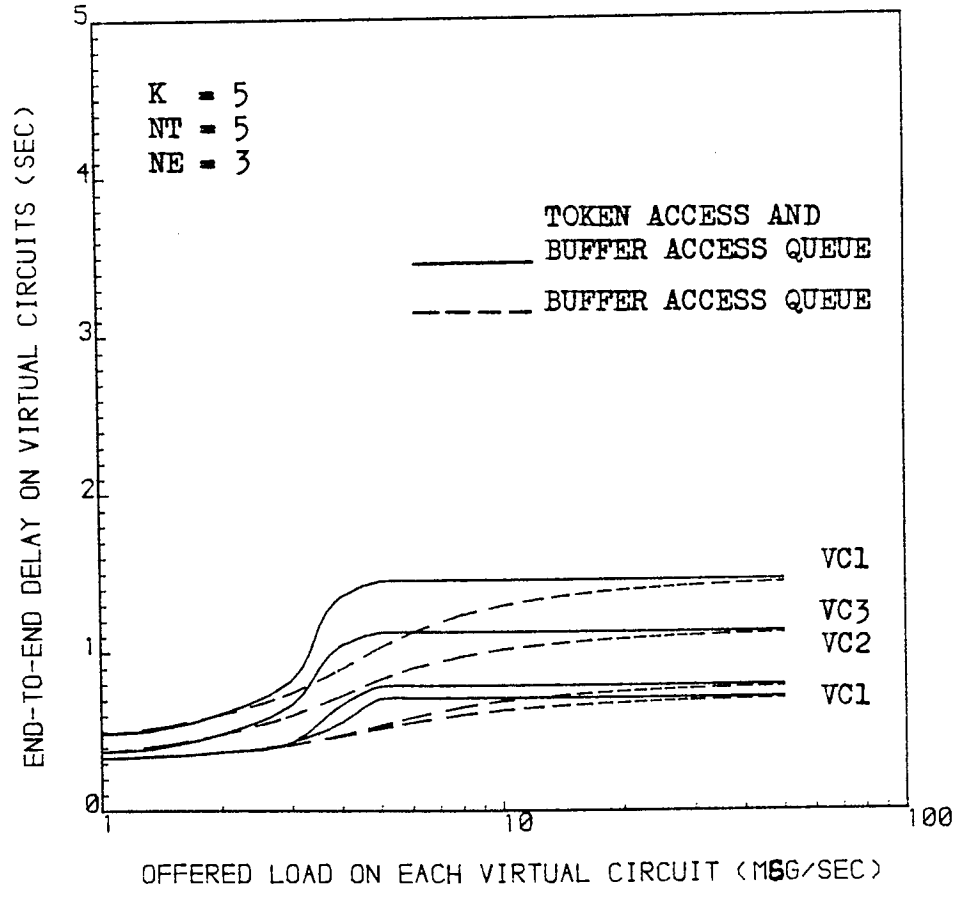


Fig. 6.5 Virtual circuit performance with Token access queue and Buffer access queue.

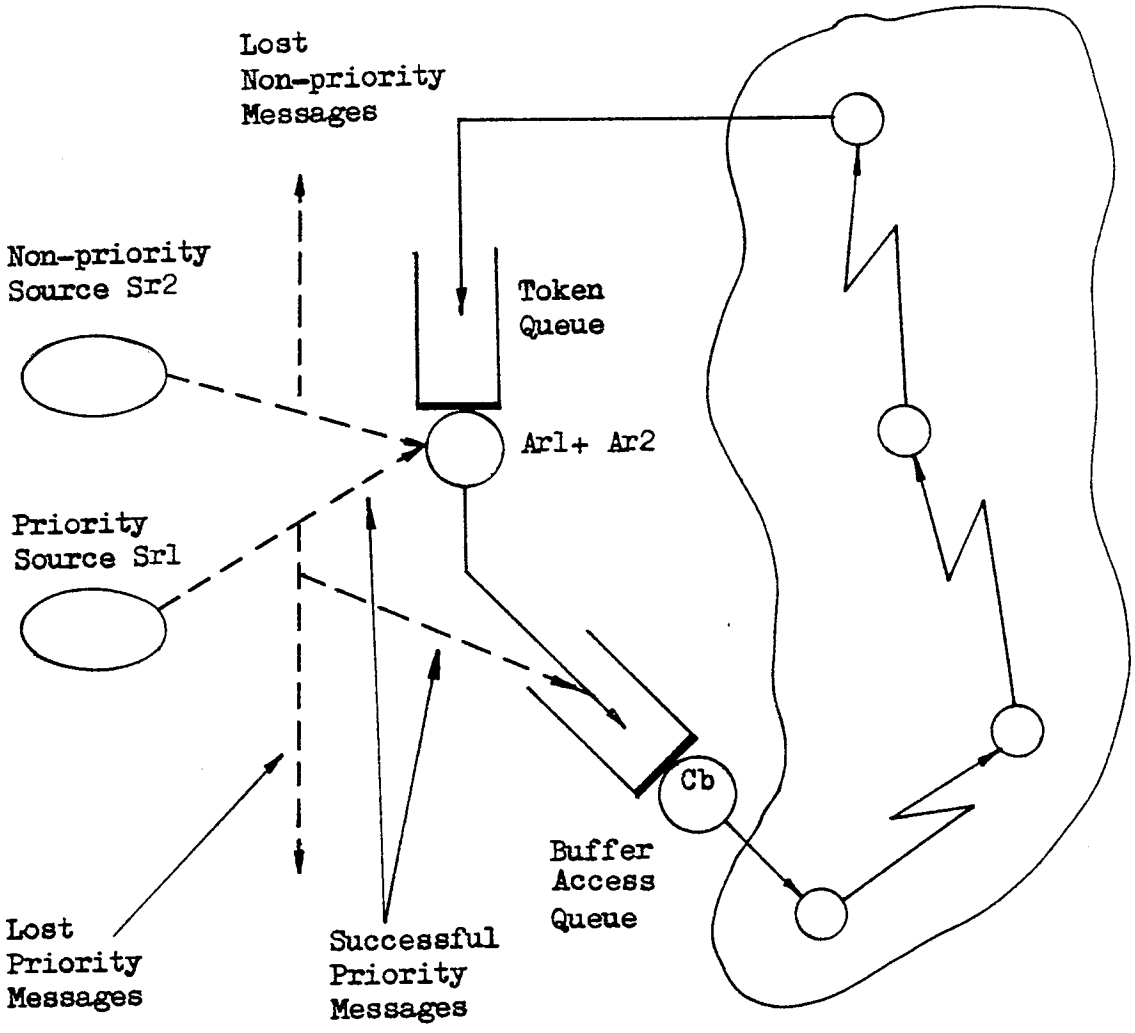
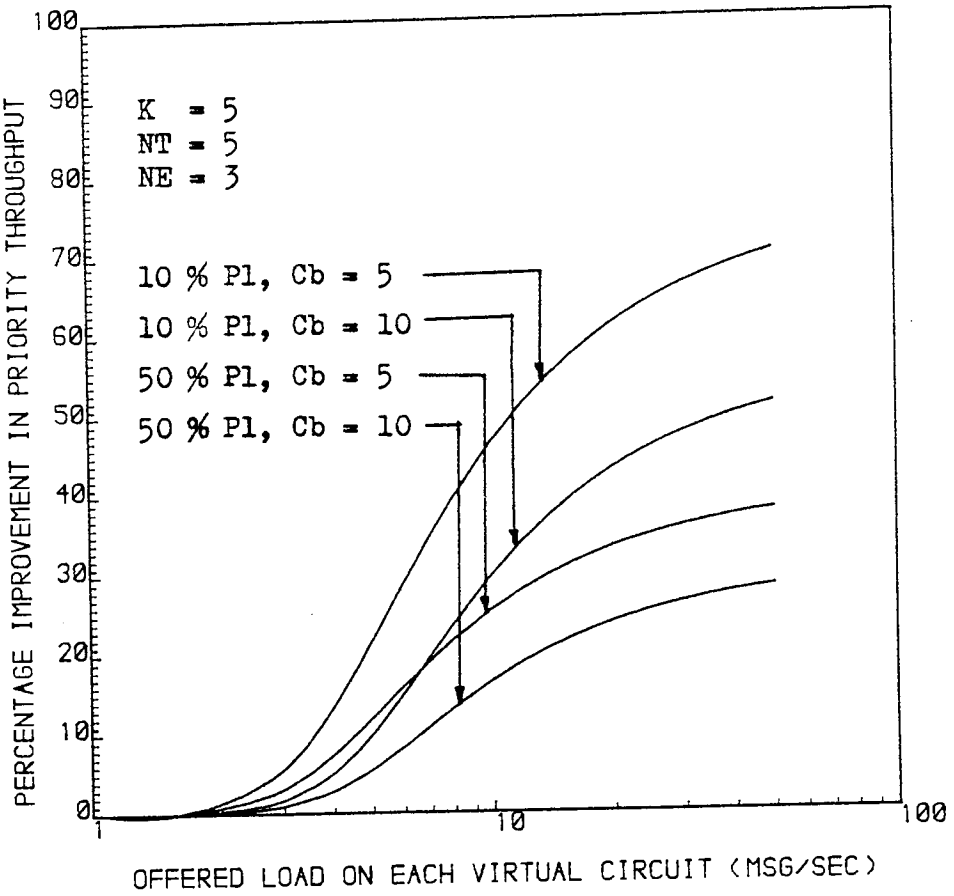


Fig. 6.6 Model of a network with priority access of virtual circuit tokens.



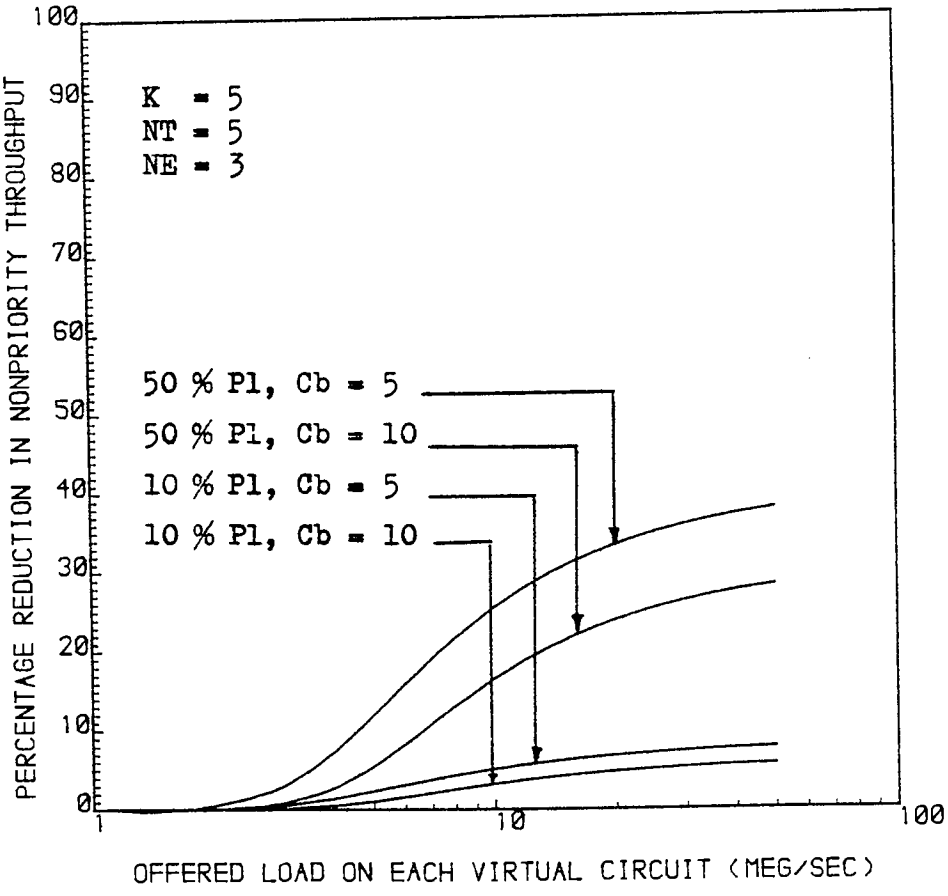


Fig. 6.7 Percentage change in priority and non-priority message throughput, when the priority message constitutes 10 % and 50 % of the load.

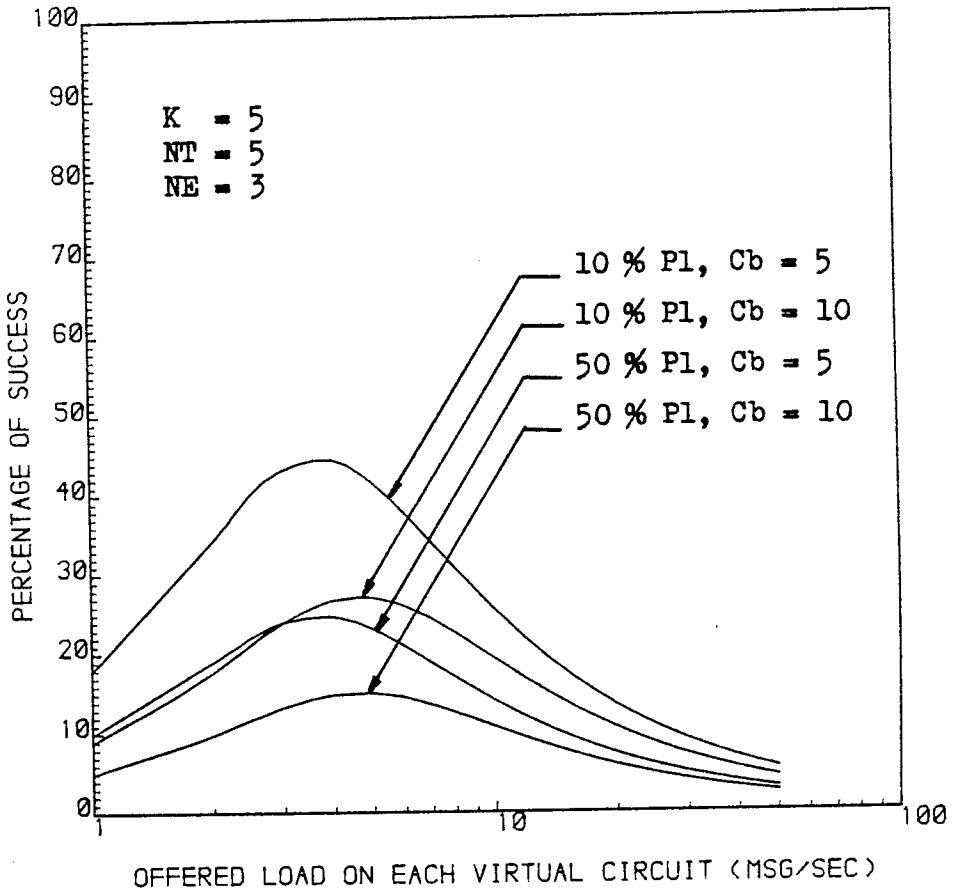
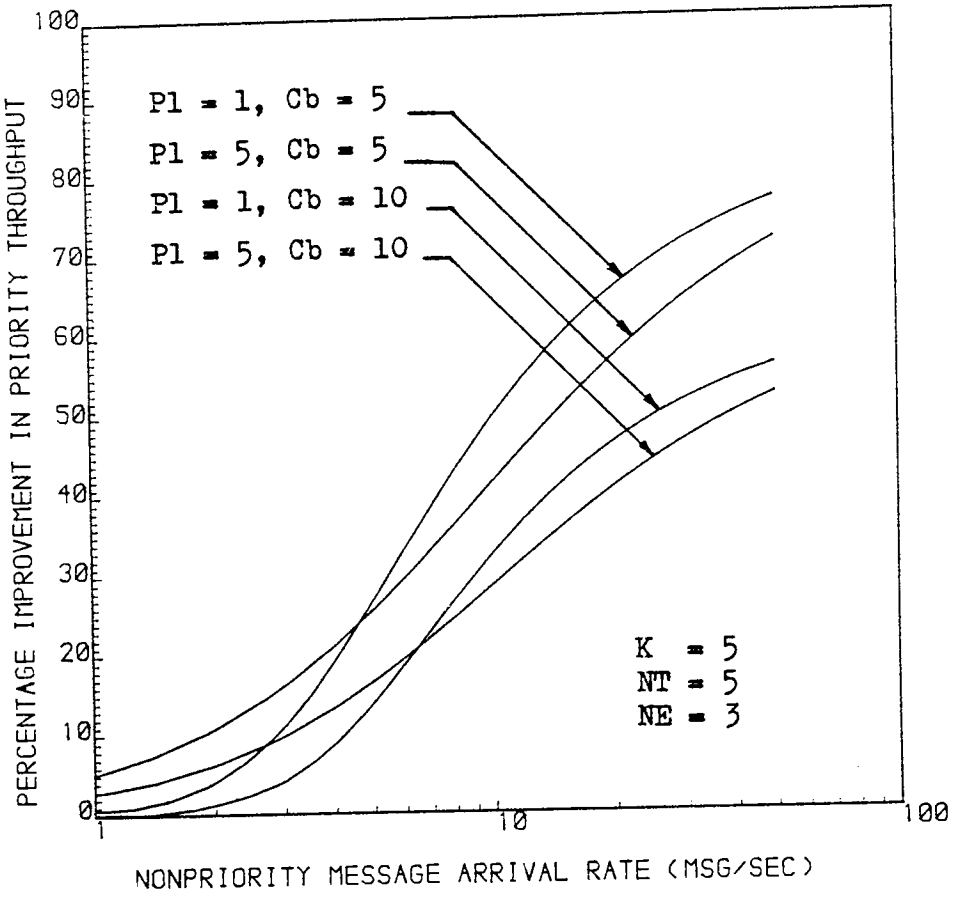


Fig. 6.8 Percentage of priority messages that are normally blocked, but succeed in geting through, with the introduction of the priority scheme.



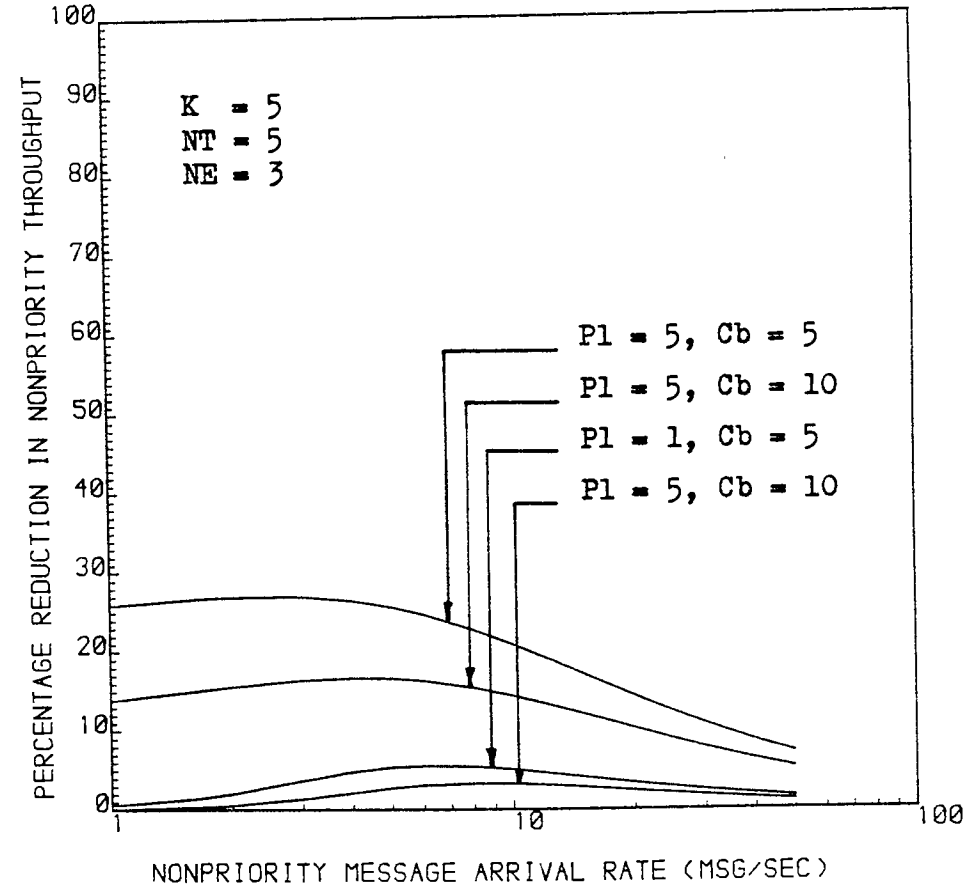


Fig. 6.9 Percentage change in priority and non-priority message throughput, for priority message arrival rates of 1 and 5 messages per second.

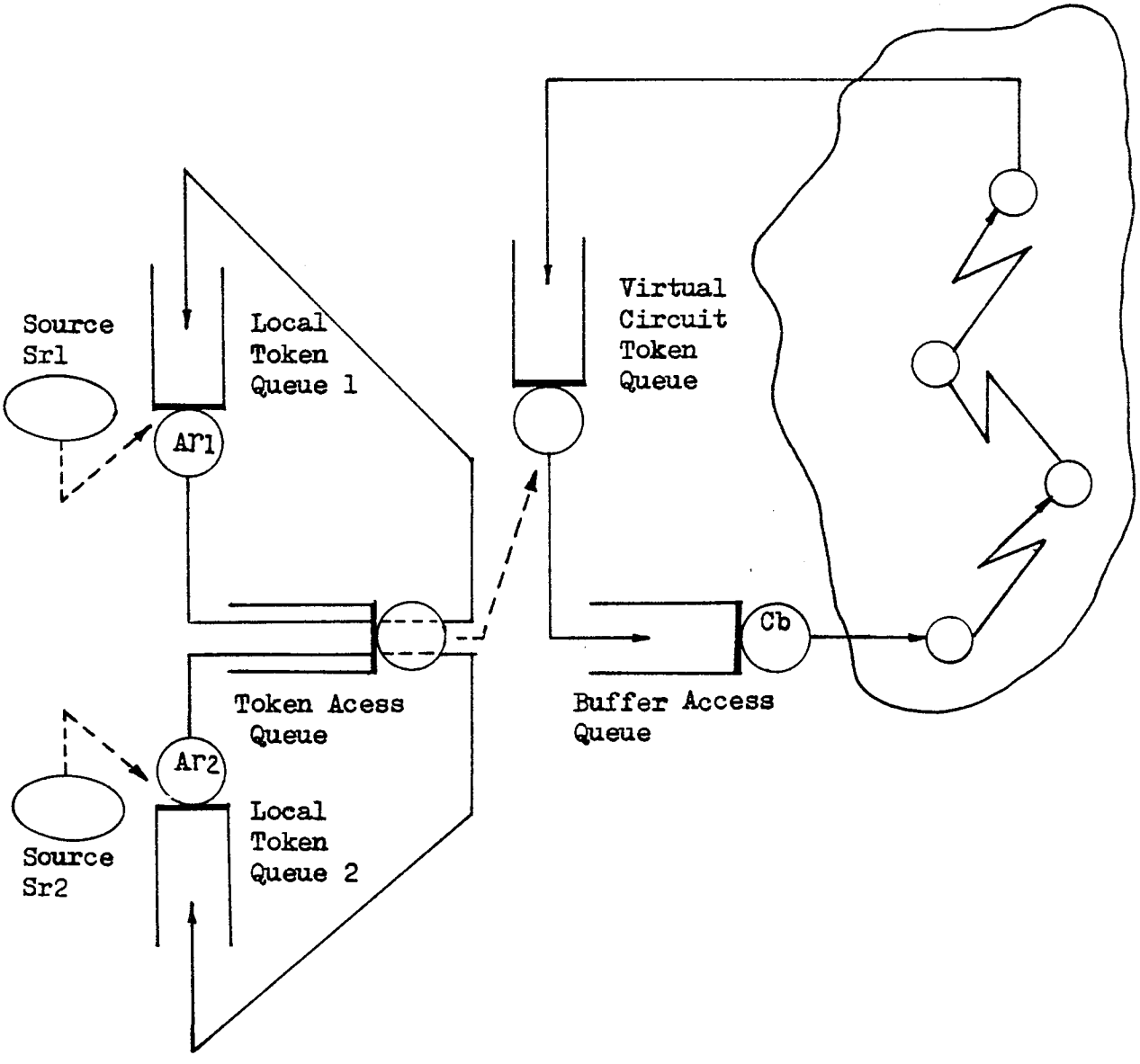


Fig. 6.10 Model of a network with local access control.

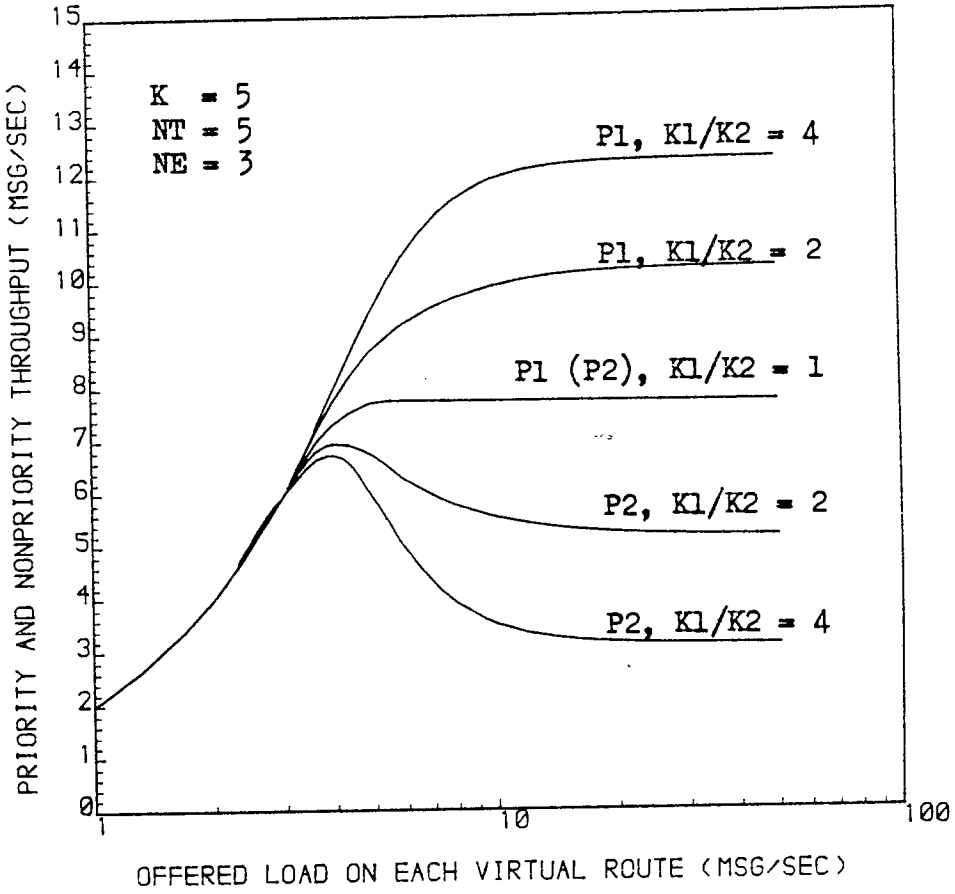


Fig. 6.11 Throughput of priority and non-priority sources, with local access control for K1/K2 = 1, 2 and 4.

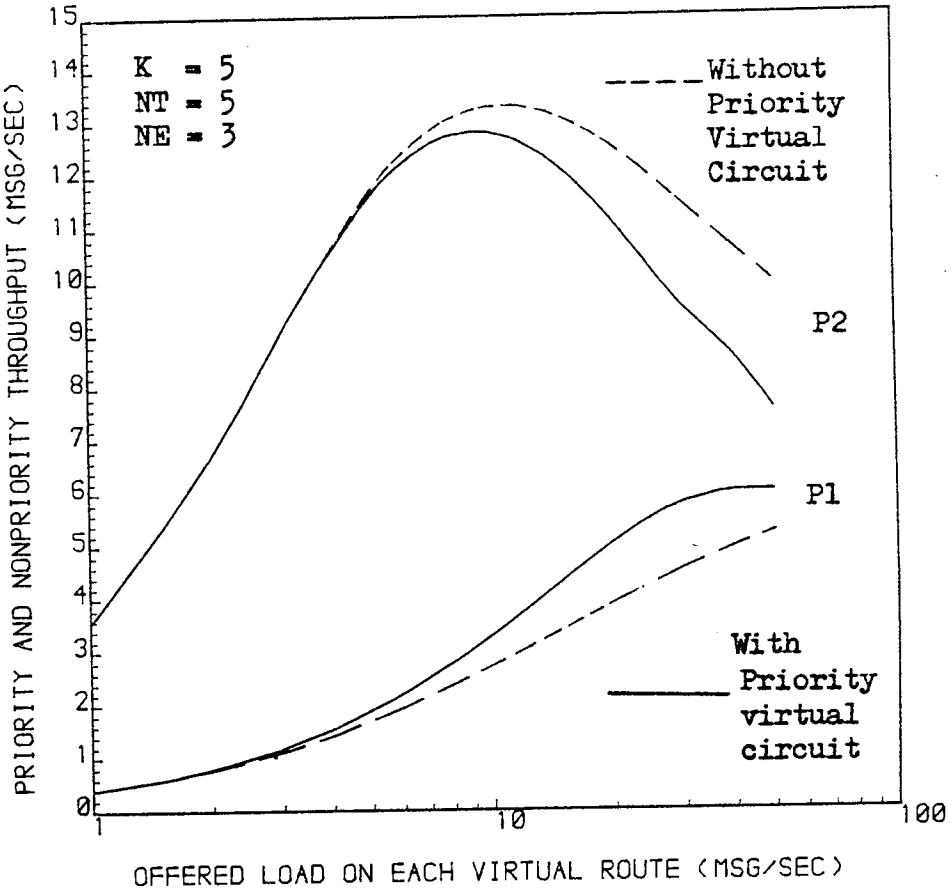


Fig. 6.12 Throughput of virtual circuits with no network access control for priority virtual circuit.

CHAPTER SEVEN

DYNAMIC FLOW CONTROL IN A STORE-AND-FORWARD COMPUTER COMMUNICATION NETWORK

7. DYNAMIC FLOW CONTROL IN A STORE-AND-FORWARD COMPUTER COMMUNICATION NETWORK

7.1. INTRODUCTION

Store-and-forward computer communication networks generally have a limited number of buffers at each node. Efficient buffer management techniques are required to avoid contention and deadlocks. Various flow control methods based on buffer allocation algorithms, have been proposed for the prevention of congestion and deadlocks [23,24,55,66]. These are essentially static methods of flow control, where the system parameters are assumed to be fixed, time invariant and state independent. It is often advantageous to design systems that dynamically adapt to varying loads and system changes, to provide improved system performance.

The network access control based on input buffer limits [55], is a static flow control mechanism. By imposing a limit on the fraction of buffers available for external messages at the entry node, the flow control mechanism favours transit messages over external messages. If NT is the total number of buffers at a node and NE is the number of buffers available for external traffic, there is an optimum ratio NE/NT that maximises the network throughput at high loads. A good heuristic choice for this ratio is K/(H+1), where H is the mean hop length and K is a scaling factor less than one, that accounts for traffic imbalance in nonhomogeneous

networks [55,56]. The factor K is determined individually for each node. The results of the five node network investigated in chapter 4, also show that at high loads, the network throughput is a maximum and the delays are a minimum if the ratio NE/NT is chosen to be equal to 1/(1+H).

With a static input buffer limit, the number of buffers available for external messages at the entry node is less than half the number at the node (since the mean hop length must be greater than one). The small buffer limit ensures that the network performance does not degrade when the transit traffic is high. The low buffer limit can however unnecessarily throttle the external traffic when the transit traffic is low, leading to low buffer utilisation. In a general nonhomogeneous network, at any given time the distribution of the load over the network can be such that, there can be regions where the transit traffic is low. In the presence of a low buffer limit, the buffer resources and the channel capacities in these regions will remain underutilised, even if external traffic is trying to gain access to these resources. Dynamic buffer allocation based on the demand from transit and external traffic, and the state of the system, can improve the buffer utilisation and the network throughput.

In this chapter, a model for the dynamic allocation of buffers for transit and external traffic messages at a node, based on Markov decision theory [19,76] is developed. Mar-

kov decision theory has been used before in the modelling of dynamic end-to-end flow control [38]. The dynamic control depends on the external and transit traffic rates, as well as the state of the system.

The main objectives of this dynamic control are:

1] Minimise blocking of transit messages and maximise their throughput.

2] Improve the buffer utilisation and maximise the network throughput, when the transit traffic is low.

Dynamic control is achieved by controlling the time and state dependent parameters of the system. A control mechanism at each node monitors the nodal buffer occupancy, as well as the mean arrival rates of external and transit messages. The mean arrival rates can be determined by measurements, or they can be estimated from the throughputs declared by the virtual circuits at the time they were established. The number of buffers available for external messages is lowered as the transit traffic increases, thereby ensuring that all the nodal resources are made available to transit messages. When the transit traffic is low, the number of buffers available for external messages is increased, improving the nodal buffer utilisation and throughput.

7.2. DYNAMIC NETWORK ACCESS CONTROL IN A TANDEM QUEUEING NETWORK

In this section a model for the dynamic control of external messages, in a tandem queueing network is developed using Markov decision theory. The tandem queueing network consists of two nodes M1 and M2 with finite number of buffers, which are inter-connected by an unidirectional link. The tandem queueing network is assumed to be a part of a larger network. Consider two streams of traffic that converge on node M1. One traffic stream arrives from an external source, while the second stream consists of transit messages from neighbouring nodes. The dynamic control problem consists of determining an optimum control policy for the admission of external messages at node M1. This optimum policy is administered by a decision mechanism located at node M1. The idea is illustrated in fig. 7.1. The decision mechanism monitors the actual arrival rate of the two traffic streams and the state of the node (or system), in order to determine the appropriate action, which either accepts or rejects the newly arrived external message. The transit traffic is, of course, always accepted by the node if a free buffer is available. Messages accepted by node M1 are routed to node M2 next. The only traffic arriving at node M2 are those from node M1. If all the buffer at node M2 are occupied, node M1 retransmits the message until it is successfully accepted by node M2.

In the following sections an optimum policy is found for the decision mechanism which accepts or rejects external messages. This optimum policy maximises the long-run throughput of transit messages, as well as the buffer utilisation of node M1. It also minimises the delay incurred by transit messages due to the presence of external messages at node M1. A reward criterion is defined and a maximal reward rate stationary operating policy for the tandem queueing network model is derived. A policy is said to be stationary if it is non-randomised and the action it chooses at time t depends only on the state of the system at time t. The problem reduces to the optimal control of external messages at a finite queue with variable transit traffic load.

7.2.1. DESCRIPTION OF THE TANDEM QUEUEING NETWORK

Consider a tandem queueing network consisting of two nodes M1 and M2, where each node is modelled as a finite queue with NT buffers. The nodes are interconnected by an unidirectional transmission link with transmission capacity of C bits per second. External messages arrive at node M1 from a Poisson source, and have a mean arrival rate of Ae messages per second. Transit messages from neighbouring nodes also arrive at node M1 with a mean arrival rate of At messages per second. It is assumed that the external message traffic rate Ae is time invariant and is fixed at the maximum arrival rate of external messages. The transit traffic on the other hand, can have one of K possible

arrival rates, designated as At(k), k=1,2,...,K. Transit messages are always accepted if one of the NT buffers at node M1 is free. External messages are admitted subject to the control mechanism located at node M1. Messages once accepted at node M1 are routed to node M2, and are accepted at M2 if a buffer is free. Otherwise, node M1 retransmits the message until it is successfully accepted by node M2. The exit link at node M2 is assumed to have a transmission capacity of C bits per second.

A mathematical model for the decision process which admits external messages to this network is developed in section 7.3, subject to the following assumptions which are made to make the analysis possible.

1] The arrival of external and transit messages at node M1 is a poisson process, with mean arrival rates Ae and At(k), k=1,2,...,K, respectively.

2] The message lengths are drawn from a negative exponential distribution, each time a message is transmitted. The mean length of a message is denoted as 1/a bits.

3] The transmission capacity of the link between nodes M1 and M2, and the exit link at node M2 is C bits per second.

4] There are NT buffers at each node. Each message occupies only one buffer.

5] The number of states are finite and the time between transitions is random. The process forms a continuous time Markov process.

7.3. ANALYSIS

7.3.1. THE STATE SPACE [19,76]

The state of the system, when there are Ne external messages and Nt transit messages at node M1, and Nd messages at node M2, is represented by (Ne,Nt,Nd,Ae,At(k)). It is assumed that the external message arrival rate is constant and time invariant. The external message arrival rate is chosen to be equal to the maximum arrival rate Ae(max). The transit traffic is constant and time invariant, and has a mean arrival rate of At(k) messages per second. The state space can now be denoted by

$$S(k) = \{ Ne,Nt,Nd \mid \begin{array}{l} 0 \leqslant (Ne+Nt) \leqslant NT, \\ 0 \leqslant Nd \leqslant NT, \\ Ae = Ae(max), \\ At = At(k) \end{array} \},$$

$$\text{for } k=1,2,\ldots,K. \qquad (7.1)$$

7.3.2. THE ACTION SPACE [19,76]

Messages arriving at node M1 are accepted or rejected by the control mechanism, based on the state of the system. To reduce the state space, K independent decision policies are determined, corresponding to the the K transit message arrival rates. It is assumed that the kth decision table

(located at node M1) specifies the actions to be taken, when the external message arrival rate is Ae(max), the transit message arrival rate is At(k), and the system is in the state (Ne,Nt,Nd,Ae,At(k)). For any decision table k, k=1,2,...,K, the set of actions taken by the control mechanism forms the action space and is denoted by,

$$A = \{De, Dt\} \tag{7.2}$$

Each element in A can have one of two possible values 0 or 1. The action space defines the following actions at node M1.

De = 0 reject external message

De = 1 accept external message

Dt = 0 reject transit message

Dt = 1 accept transit message

In the discussions to follow, it is assumed that the transit traffic is At(k), k=1,2,...,K, and the kth decision table is being determined. Hence, the two variables Ae and At(k) are dropped from the system state representation. That is, the state of the system is designated as (Ne,Nt,Nd).

7.3.3. THE POLICY SPACE [19,76]

The policy space specifies the action to be taken in each state of the system. More precisely, a policy space F

is a decision rule that specifies, given the system is in state (Ne,Nt,Nd), the action to be taken is,

$$F(Ne,Nt,Nd) \in A. \qquad (7.3)$$

Where, it is assumed that the external traffic is Ae(max) and the transit traffic is At(k). The action taken at each node may be based on the global state of the system, or it may be based on the state of the individual nodes alone. If the actions are based on the global state of the system, then the global performance of the system can be optimised. However this may not be practical because of the large number of states and the complex dependencies between the nodes. Actions based on the state of individual nodes, forms a local control. In this case other global control mechanisms are necessary to protect the network. In this analysis, the actions are based on the combined state of node M1 and M2. Further if the policy determines the actions for both transit and external messages, a buffer allocation policy results. If the actions for transit messages are assumed, then the policy determines only the actions for external messages, giving a dynamic network access control policy for external messages. In this analysis the transit messages are assumed to be accepted at node M1, if free buffers are available at the node. That is,

$$F(Ne,Nt,Nd) = \{De,1\},$$

if $Ne + Nt < NT$ at node M1.

In addition, the following boundary condition is assumed in the analysis. Transit and external messages arriving at node M1 are rejected if all buffers at node M1 are full. That is, if Ne+Nt=NT at node M1, then

$$F(Ne,Nt,Nd) = \{0,0\}.$$

Here, policies which are functions of the present state of the system are implicitly chosen. Such policies are known as stationary policies [19,76], and the set of such policies is denoted by F. The number of external messages and transit messages at node M1, and the number of messages at node M2 are random variables. A stationary policy F maps the state space S into the action space A. Given a stationary policy F, F(Ne,Nt,Nd)=(De,Dt) means that, when the system is in the state (Ne,Nt,Nd), each external message is accepted with a probability De(De=0 or 1). and each transit message at node M1 is accepted with a probability Dt(Dt=0 or 1).

7.3.4. STATE TRANSITION RATES

Since the system forms a finite state continuous Markov process, state transition rates, rather than state transi-

tion probabilities are considered. A birth-death process is assumed, where transitions occur only between adjacent states in any short interval dt. Multiple arrivals, multiple departures or both arrival and departure are excluded in the sense that, the probability of such an event occurring is of the order 0(dt). Let the system be in the state (Ne,Nt,Nd), and the stationary policy used is F(Ne,Nt,Nd)={De,Dt}. The equilibrium transition rates from state (Ne,Nt,Nd) to (Ne′,Nt′,Nd′) under policy F is denoted by P[(Ne,Nt,Nd);(Ne′,Nt′,Nd′);F] and is defined as,

P[(Ne,Nt,Nd);(Ne′,Nt′,Nd′);F] =

P[Transition rate from state (Ne,Nt,Nd) to state (Ne′,Nt′,Nd′) when policy F is used.]

The following transition rates from state (Ne,Nt,Nd) to (Ne′,Nt′,Nd′) are defined.

P[(Ne,Nt,Nd);(Ne′,Nt′,Nd′);F) =

1) External arrival:	Ae De	if Ne′=Ne+1 Nt′=Nt Nd′=Nd and Ne+Nt < NT;
2) Transit arrival:	At Dt	if Nt′=Nt+1 Ne′=Ne Nd′=Nd and Ne+Nt < NT;

3) Departure of an external message at node M1:	[Ne/(Ne+Nt)] a C	if Ne′=Ne-1 Ne > 0 Nt′=Nt Nd′=Nd+1 Nd < NT;
4) Departure of a transit message at node M1:	[Nt/(Ne+Nt)] a C	if Nt′=Nt-1 Nt > 0 Ne′=Ne Nd′=Nd+1 Nd < NT;
5) Departure of a message at node M2:	a C	if Nd′=Nd-1 Nd > 0 Nt′=Nt Ne′=Ne
6) Otherwise	0.	

(7.4)

The diagonal elements of the state transition martix are given by [28,41],

$$P[(Ne,Nt,Nd);(Ne,Nt,Nd);F] = - \sum_{\substack{(Ne',Nt',Nd') \in S \\ (Ne',Nt',Nd') \neq (Ne,Nt,Nd)}} P[(Ne,Nt,Nd);(Ne',Nt',Nd');F].$$

The state transition rate matrix under the policy F is denoted by P(F):

$$P(F) = \{ P[(Ne,Nt,Nd);(Ne',Nt',Nd');F]\}. \tag{7.5}$$

7.3.5. THE PERFORMANCE CRITERIA, THE REWARD FUNCTION AND THE REWARD RATE

The basic performance measures in computer communication network are throughput and delay. There are two main objectives in using dynamic network access control. The first is to improve transit traffic throughput. The second, is to improve the buffer utilisation, especially when the transit traffic is low and the external traffic is high. The dynamic control can reject external traffic with a high probability when the transit traffic is high, thereby ensuring that the transit traffic is not denied any resources. It can also accept external traffic with a high probability, when the transit traffic is low, thereby increasing the buffer utilisation and the nodal throughput. However, the presence of a large number of external messages (when the transit traffic is low), can increase the nodal queueing delay for transit messages. Hence a third objective will be the reduction of nodal queueing delay for transit messages, incurred due to the presence of external messages queueing at the node. This will ensure that the dynamic control does not accept too many external messages.

In the performance measure used here, the long run average throughput of the node (both transit and external messages) is maximised and the probability of transit messages being blocked or delayed by the presence of external messages, is minimised.

Let

Vt throughput of transit traffic;

Ve throughput of external traffic;

a′ The reward gained by accepting a transit message;

b The reward gained by accepting an external message;

c The loss due to the increase in delay of transit messages, caused by accepting an external message;

d The loss due to a transit message being blocked at node M1, caused by the presence of external messages at node M1;

$\bar{N}e$ Mean number of external messages at node M1.

Then the total reward of the system is defined as,

$$R = a' Vt + (b-c) Ve - d At(k) \bar{N}e/NT, \qquad (7.6)$$

where ,

$$a' \geq b > c > 0, \quad \text{and} \quad d > 0.$$

Having defined the objective function R that is to be maximised, the expected immediate earning rate Q[(Ne,Nt,Nd);F] of the system, when operating under policy F is of interest. Let the system be in the

state (Ne,Nt,Nd). The system can make a transition to some other state (Ne′,Nt′,Nd′) during the interval dt. The probability of the system making a transition from state (Ne,Nt,Nd) to state (Ne′,Nt′,Nd′) in a time dt is,

$$P(\text{The system makes a transition from state } (Ne,Nt,Nd) \text{ to state } (Ne',Nt',Nd') \text{ during the interval dt under policy F}) = P[(Ne,Nt,Nd),(Ne',Nt',Nd');F]\,dt.$$

Let $F(Ne,Nt,Nd)=(De,Dt)$ be the action taken under policy F when the system is in the state (Ne,Nt,Nd). The system earns a reward of "a′" units if it accepts a transit message. The reward R1 the system earns in the state (Ne,Nt,Nd), by accepting a transit message is,

$$R1 = a' P[(Ne,Nt,Nd),(Ne,Nt+1,Nd);F]\,dt.$$

From equation (7.4), R1 can be written as

$$R1 = a' At\ Dt\,dt.$$

Hence, the expected immediate rewards the system earns by making transitions out of state (Ne,Nt,Nd), in a time dt are:

1) R1 = reward gained by accepting a transit message,

= a′ At Dt dt;

2) R2 = reward gained by accepting an external message,

= b Ae De dt;

3) R3 = Loss due to the increase in delay of transit messages, caused by accepting an external message,

= c Ae De dt.

Here, De and Dt are the actions taken on external and transit message arrivals, when the system is in the state (Ne,Nt,Nd). Hence the expected immediate earning rate when making a transition out of state (Ne,Nt,Nd) is,

$$Q1 = (R1 + R2 + R3)/dt,$$

$$= (a' At Dt) + [(b-c) Ae De],$$

for $Ne + Nt \neq NT$ at node M1.

The system also incurs a loss when it is in the state (Ne,Nt,Nd| Ne+Nt=NT), which is the state when all the buffers at node M1 are occupied. In this state, transit messages are blocked and the loss to the system during an interval dt is,

$$R4 = - d (Ne/NT) \text{ (Probability of a transit message arriving in the interval dt)},$$

$$= - d (Ne/NT) At dt.$$

The expected immediate earning rate when the system is in the state (Ne,Nt,Nd| Ne+Nt=NT) is given by,

$$Q2 = - d\ (Ne/NT)\ At.$$

Hence the expected immediate earning rate when the system is in the state (Ne,Nt,Nd), under policy F is,

$$Q[(Ne,Nt,Nd);F] = a'\ At\ Dt + (b-c)\ Ae\ De$$

for Ne+NT $\neq$ NT at node M1,

$$Q[(Ne,Nt,Nd);F] = - d\ (Ne/NT)\ At$$

for Ne+Nt = NT at node M1.

The number of variables can be reduced by setting,

$$a' = b = 1, \quad 0 \leqslant c < 1 \quad \text{and} \quad d > 0.$$

The vector of immediate earning rate under policy F is denoted by,

$$Q(F) = \{Q[Ne,Nt,Nd)+F]\}. \tag{7.7}$$

7.3.6. STATEMENT OF THE PROBLEM

The above formulation specifies a Markov decision process with states and transition rates defined by (7.1) and (7.4); action and policy space defined by (7.2) and (7.3); and the earning rate given by (7.7). Under a stationary policy F, the average reward per unit time, is determined by

the vector G(F). That is,

$$G(F) = \{ g[(Ne,Nt,Nd);F] \},$$

where , g[(Ne,Nt,Nd);F] is the long run expected reward per unit time, given that the process initially starts in the state (Ne,Nt,Nd) and policy F is used. G(F) is given by [19,76],

$$G(F) = \underset{N\to\infty}{\text{Limit}} \{ [1/(N+1)] \sum_{J=0}^{N} [P(F)]^{J} Q(F) \},$$

where , $[P(F)]^{J}$ is the Jth power of P(F) under the stationary policy F. If F is a policy such that the Markov chain defined by P(F) is completely ergodic (i.e., there is only one recurrent chain in the system), then all components of G(F) are equal. This is due to the fact that, in an ergodic chain, any state can be reached from any other state in a finite number of transitions. Hence the long run expected reward per unit time, is the same for all (initial) states. That is, all states have the same gain G(F). The objective is now to find a stationary policy F*, which maximises each component of the expected average reward per unit time. That is,

$$G(F^*) = \underset{F^* \in F}{\text{Max}} \{G(F)\}.$$

It can be shown that for a bounded reward function, when the

action space and state space are finite, a stationary policy always exists [19,76]. These conditions are met in this problem. For an ergodic chain it can be shown that [28],

$$g[(Ne,Nt,Nd);F] = \sum_{(Ne,Nt,Nd) \in S} E[(Ne,Nt,Nd);F] \quad Q[(Ne,Nt,Nd);F] \tag{7.8}$$

where , E[(Ne,Nt,Nd);F] is the limiting state probability of the system being in the state (Ne,Nt,Nd) when policy F is used. Substituting for Q[(Ne,Nt,Nd);F] from (7.4), in equation (7.8),

For $Ne + Nt \neq 0$,

$$g[(Ne,Nt,Nd);F] = \sum_{(Ne,Nt,Nd) \in S} a' \; At \; F(Ne,Nt,Nd) \; E[(Ne,Nt,Nd);F] + \sum_{(Ne,Nt,Nd) \in S} (b-c) \; Ae \; F(Ne,Nt,Nd) \; E[(Ne,Nt,Nd);F]. \tag{7.9}$$

Now,

$$\sum_{(Ne,Nt,Nd) \in S} At \; E[(Ne,Nt,Nd);F] \; F(Ne,Nt,Nd) = At \sum_{(Ne,Nt,Nd) \in S} E[(Ne,Nt,Nd);F] \; F(Ne,Nt,Nd)$$

$$= At \sum_{(Ne,Nt,Nd) \in S} P[\text{ the system being in the state } (Ne,Nt,Nd)] \times P[\text{ accepting a transit message in the the state } (Ne,Nt,Nd), \text{ under policy } F]$$

$$= At\ P[\text{ the system accepting a transit message}]$$

$$= \text{The mean transit traffic throughput,}$$

$$= Vt. \qquad (7.10)$$

Similarly,

$$\sum_{(Ne,Nt,Nd) \in S} Ae\ F(Ne,Nt,Nd)\ E[(Ne,Nt,Nd);F] = Ve,$$

$$= \text{External traffic throughput,} \qquad (7.11)$$

Therefor for $Ne + Nt \neq 0$,

$$g[(Ne,Nt,Nd);F] = a'\ Vt + (b-c)\ Ve.$$

For $Ne+Nt=NT$,

$$g[(Ne,Nt,Nd);F] = -\sum_{(Ne,Nt,Nd) \in S} d\ (Ne/NT)\ At\ E[(Ne,Nt,Nd);F],$$

$$= -\ d\ (At/NT) \sum_{(Ne,Nt,Nd) \in S} Ne\ E[(Ne,Nt,Nd);F],$$

$$= -\ d\ At\ (\bar{Ne}/NT).$$

Where, $\bar{Ne}$ is the mean number of external messages at node Ml. Hence,

$$G(F) = a' Vt + (b-c)\ Ve - d\ At\ (\bar{N}e/NT), \qquad (7.12)$$

and maximising G(F) is the same as maximising the reward function R in equation (7.6).

7.3.7. SOLUTION OF THE PROBLEM

The problem now consists in finding the optimum control policy for a given transit traffic arrival rate At(k), k=1,2,...,K, and a given set of gain constants "a'", "b", "c" and "d". The optimal stationary control policy is found using the policy iteration algorithm of Howard [28,29]. The decision model is expressed as a set of linear homogeneous equations. The computation starts with an arbitrary policy, and each iteration improves the policy. If such an improvement is possible, the new policy replaces the old policy. Otherwise, the last policy is taken as the optimum policy.

The initial policy is formulated by accepting all external message arrivals, if free buffers are available at node M1. For each new policy F (including the initial policy), the gain G(F) (i.e., the average reward per unit time under policy F), is evaluated. The policy that gives the maximum gain is chosen as the optimum control policy (for the given set of gain constants and transit and external traffic arrival rates). In [28], it is shown that the algo-

rithm converges in a finite number of iterations. The policy iteration algorithm is given in appendix C.

7.3.8. PERFORMANCE MEASURES

Once the optimal policy has been evaluated, the performance measures of the system can be evaluated. From [28],

$$\sum_{(Ne,Nt,Nd) \in S} E[(Ne,Nt,Nd);F] = 1 \qquad (7.13)$$

and

$$E\{F\}\ P\{F\} = 0. \qquad (7.14)$$

Where, E{F} and P{F} are the limiting probability state matrix and the transition rate matrix. The limiting state probabilities can be solved using (7.4), (7.13) and (7.14).

The mean transit traffic throughput is given by,

$$Vt = At \sum_{(Ne,Nt,Nd) \in S} E[(Ne,Nt,Nd);F]\ \ F(Ne,Nt,Nd). \qquad (7.8)$$

The mean external traffic throughput is given by,

$$Ve = Ae \sum_{(Ne,Nt,Nd) \in S} E[(Ne,Nt,Nd);F]\ \ F(Ne,Nt,Nd). \qquad (7.9)$$

The mean number of transit messages at node M1 is,

$$\bar{N}t = \sum_{N=1}^{NT} N \sum_{\substack{Ne=0 \\ Nt=N}}^{Ne=NT-Nt} E[(Ne,Nt,Nd);F]. \qquad (7.15)$$

The mean number of external messages at M1 is,

$$\bar{N}e = \sum_{N=1}^{NT} N \sum_{\substack{Nt=0 \\ Ne=N}}^{Nt=NT-Ne} E[(Ne,Nt,Nd);F]. \qquad (7.16)$$

The mean queueing delay for transit messages at node M1 is,

$$Dt = \bar{N}t/Vt \qquad (7.17)$$

and the mean queueing delay for external messages at node M1 is,

$$De = \bar{N}e/Ve. \qquad (7.18)$$

7.4. DYNAMIC NETWORK ACCESS CONTROL WITH PRIORITY SCHEDULING FOR TRANSIT MESSAGES

The dynamic control reduces the throttling effect on the external messages when the transit traffic is low. The resulting increase in queueing delay incurred by transit messages (because of the presence of large number of

external messages), is controlled by including the loss factor "c" in the reward criterion. An alternative method is to use a scheduling discipline which gives preemptive priority to transit messages. In this case an arriving transit message is queued ahead of all external messages. If an external message is in service, the service is interrupted and the transit message is served immediately. The only modification necessary in the theory is with the transition rate probability matrix. In particular, the transition rate for the departure of an external message from node M1 is,

$$P[(Ne,Nt,Nd);(Ne-1,Nt,Nd+1);F] = \begin{cases} a\,C & \text{if } Ne > 0,\ Nt = 0,\ Nd < NT, \\ 0 & \text{otherwise.} \end{cases}$$

The transition rate for the departure of a transit message from node M1 is,

$$P[(Ne,Nt,Nd);(Ne,Nt-1,Nd+1);F] = \begin{cases} a\,C & \text{if } Nt > 0,\ Nd < NT, \\ 0 & \text{otherwise.} \end{cases} \qquad (7.19)$$

7.5. NUMERICAL RESULTS

This section presents the numerical results for a tandem queueing network equipped with dynamic network access control. The transmission links have a capacity of 10 messages per second, and the external message arrival rate is

fixed at 10 messages per second. Two values for the number of buffers at each node, namely NT=5 and NT=10 are considered. In each case, the optimum control policy is found and the mean throughput for transit and external traffic streams are determined (equation 7.8 and 7.9). The effect of the "d" parameter is investigated first. Fig. 7.2 shows the percentage change in transit and external traffic throughputs under the optimum policy, as parameter "d" is varied from 0.1 to 10. The results are obtained for two values of transit message arrival rates, namely At=2 and At=10 messages per second. These two values correspond to the situation when the transit traffic is low and high. The change in throughputs are with respect to the transit traffic throughput when d=0 and the optimum policy is used. Here, the parameters a'=1, b=1 and c=0. When d=0, external and transit messages are treated alike. For low transit traffic rates (At=2) and NT=10, the improvement in transit traffic throughput when d=0.1 is negligible. But as d is made larger, the transit traffic throughput improves. When d=10, the improvement is nearly 22%. At the same time the external traffic throughput decreases by only 6%, when d=10.

When the transit traffic load is high (At=10), the improvement in transit traffic throughput for d=0.1 is 63%. When d=10, the improvement is nearly 78%. The corresponding reduction in external message throughput is 82%. This shows that when the number of buffers is ten (NT=10), any positive non-zero value for d, improves the transit traffic

throughput. When the number of buffers at each node is reduced to five (NT=5), similar changes are observed for d > 0.2.

Fig. 7.3 shows the percentage decrease in mean queueing delay, as the parameter d is varied from 0.1 to 10. The mean queueing delay for transit and external messages are different for d > 0 (except when d=0, when they are equal). As d is varied from 0.1 to 10, the queueing delay for both transit and external messages decreases. For high transit traffic arrival rates (At=10), the improvement in queueing delay for transit and external messages are 30% and 65% respectively (when NT=10). External messages are admitted to the network only when the number of transit messages at node M1 is small (the probability of this happening is low when the transit traffic is high). Since fewer external messages are admitted to the network (and hence, the total number of messages at node M1 is reduced), the queueing delays are also small. These curves show that under the optimum policy for d > 0, the transit traffic throughput improves at the expense of external traffic throughput. At the same time, when the transit traffic is low, the nodal buffers are made available to external messages, thereby improving the nodal buffer utilisation and throughput.

Fig. 7.4 shows the transit transit traffic and external traffic throughput under the optimum policy, as a function of transit traffic arrival rate for several values of d

(with NT=10). As d is made large, the decrease in external traffic throughput for low values of transit traffic is small (5%). For high transit traffic arrival rates, the improvement in transit traffic throughput as d is increased is considerable (80%).

Fig. 7.5 shows the nodal throughput as a function of transit traffic arrival rate, when the tandem queueing network is equipped with only a static network access control (i.e., without the dynamic network access control). The input buffer limit for the static network access control is varied from NE=1 to NE=5 (with NT=5). Here, NE is the number of buffers available for external messages at node M1. For the network to be stable, the value of NE has to be limited to one or two. The limit NE=2, corresponds to a mean hop length of one, and NE=1 corresponds to a mean hop length of two or more (when NT=5). The figure also shows the throughput of the tandem queueing network, when it is equipped with the dynamic network access control. The dynamic network access control uses an optimum control policy with d=5. When the tandem queueing network is equipped with the dynamic network access control, the transit message throughput is higher for large transit message arrival rates (compared to the transit traffic throughput when the network has only a static network access control). At the same time, the external message throughput is also higher, when the transit traffic is low. The buffer limit in the static network access control improves the transit traffic

throughput, only if the number of buffers available for external messages is small. But this also reduces the external message throughput, when the transit traffic is low.

The dynamic control admits more external messages into the the network when the transit traffic is low. This can increase the queueing delay of transit messages. The parameter c ($0 \leq c < 1$) is introduced to reduce the queueing delay for transit messages. Fig. 7.6 shows the queueing delay for transit and external messages at node M1 under the optimum policy, as a function of transit message arrival rate, for five different values of c. The parameter d=5, NT=10 and the mean arrival rate of external messages is 10 messages per second. The queueing delay for transit messages improves by about 10%, when the parameter c is varied from 0 to 0.8. But, the delay incurred by transit messages when the transit message arrival rate is low, is nearly the same as the delay when the the arrival rate is high, even when c is 0.8. Hence the c control is not very effective in reducing the queueing delay for transit messages when their arrival rate is low.

Fig. 7.6 also shows the queueing delay at node M1, when preemptive priority scheduling is used for transit messages under the optimum policy (for d=5, c=0 and NT=10). Priority scheduling for transit messages, reduces their queueing delay, especially when the transit traffic is low. The

queueing delay for transit messages is reduced by as much as 80%, when their arrival rate is small (At=1). For higher arrival rates (At=10), the improvement with preemptive priority scheduling is 17%. The priority scheduling however increases the queueing delay for external messages. The increase is around 10% at low transit traffic arrival rates. For higher transit message arrival rates, the increase in queueing delay is about 170%. The priority scheduling causes no significant reduction of external message throughput, when the transit message arrival rate is low. However, at higher transit message arrival rates, the external message throughput is reduced by about 50%. This is because, the probability of an external message being in service is low and the probability of its service being interrupted by a transit message is high, at high transit message arrival rates.

7.6. SUMMARY OF RESULTS

The numerical results show that the dynamic control of external messages using an optimum policy (for a given set gain constants and arrival rates), can improve the network performance at all transit traffic loads. The control ensures that the buffer utilisation and nodal throughput are high at low transit traffic loads, by admitting more external messages into the network. At the same time, the external messages are throttled at high transit traffic loads, thereby ensuring that all nodal resources are made available

to transit messages. Priority scheduling for transit messages under the optimum policy, reduces the queueing delay incurred by them, especially when their arrival rate is low.

7.7. APPLICATION OF THE MODEL

A model for the dynamic allocation of buffers using Markov decision theory was developed. The formulation of the model is general, and can be extended to more than two classes of traffic. The method can be used to model the Structured Buffer Pool flow control for avoiding store-and-forward deadlocks [23,24,66]. In this strategy, messages at each node are classified according to the number of hops they have covered. The buffer pool at each node is divided into H classes, where H is the maximum number of hops covered by a message in the network. A message that arrives at a node after covering h hops, can have access to buffers belonging to class h (or all buffer belonging to class 0,1,...,h) at the node. Though this strategy eliminates deadlocks, the partitioning of buffers results in poor utilisation of the buffers. The network throughput degrades when the load exceeds a critical limit. Simulation studies have shown that such degradation in network performance can be avoided, if the size of each buffer class is dynamically determined [24]. The model developed in this chapter can be used to determine an optimum control policy for accepting or rejecting messages of each class , based on the state of the node and the arrival rates of messages from

each class.

Dynamic buffer allocation can be used in virtual circuit hop level control [21,63], based on virtual circuit class or priority. Preferential treatment can be given to selected virtual circuits, by dynamically controlling the nodal resources available to each virtual circuit. Lastly, the model can also be used for dynamic gateway control in inter-networking. Consider two computer networks connected through a gateway as shown in fig. 7.7. Traffic handled by gateway A consists of through traffic from network B, and traffic generated in network A that is destined for other networks. Priority can be given to through traffic from gateway B, by dynamically controlling the acceptance of messages generated in network A.

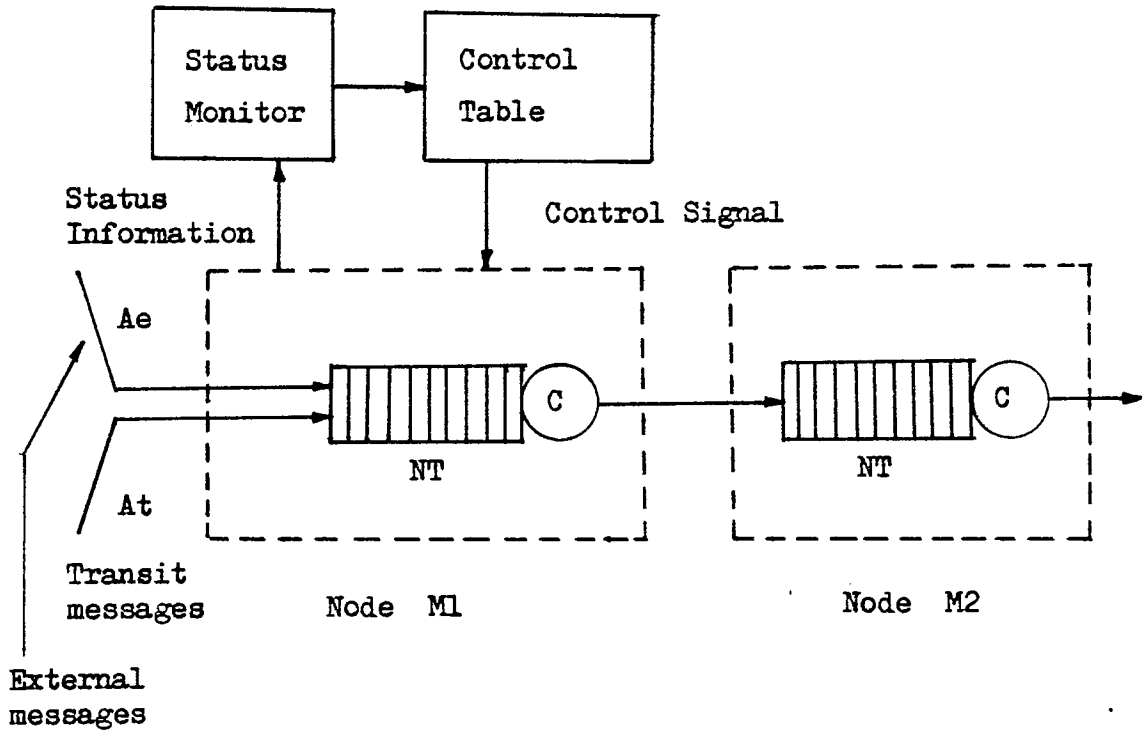
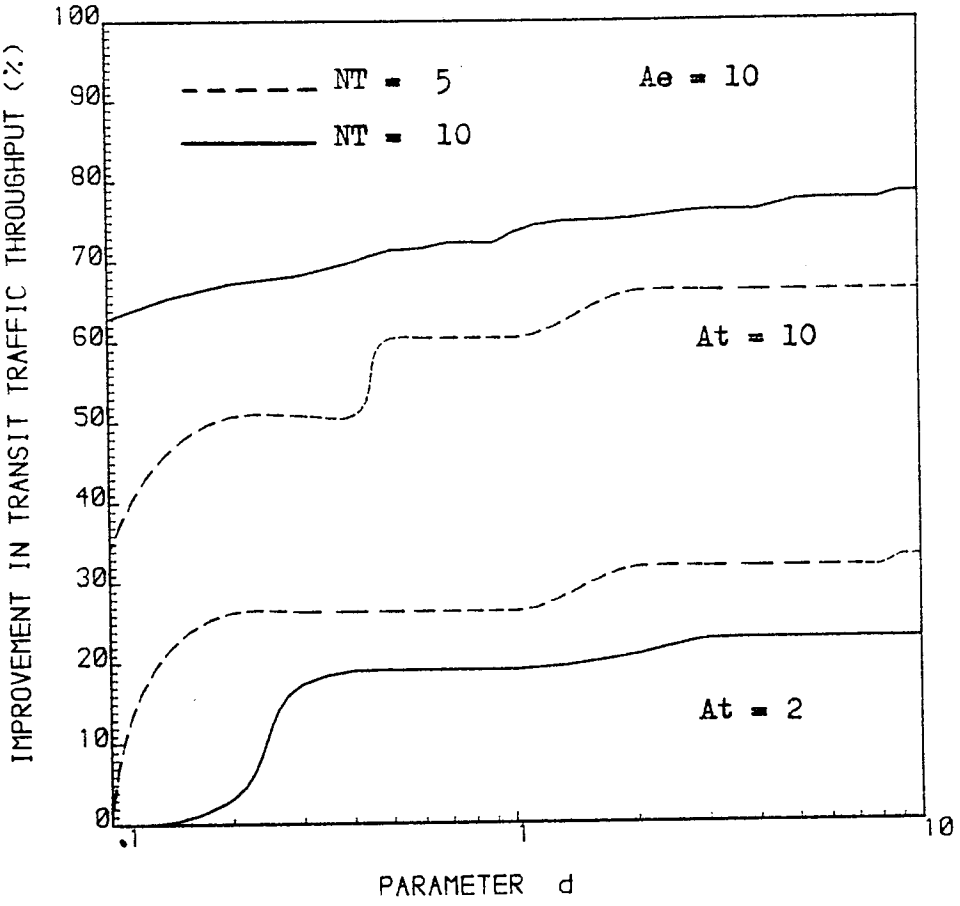


Fig. 7.1 Model of a tandem queueing network with dynamic network access control.



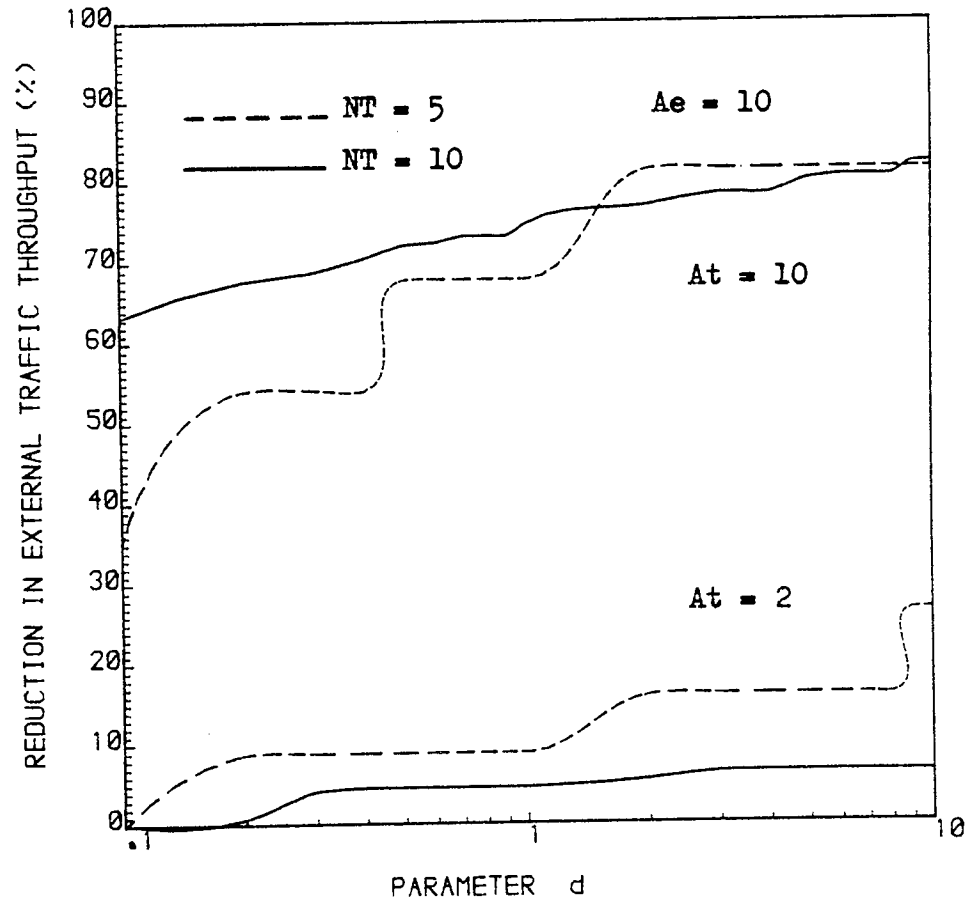
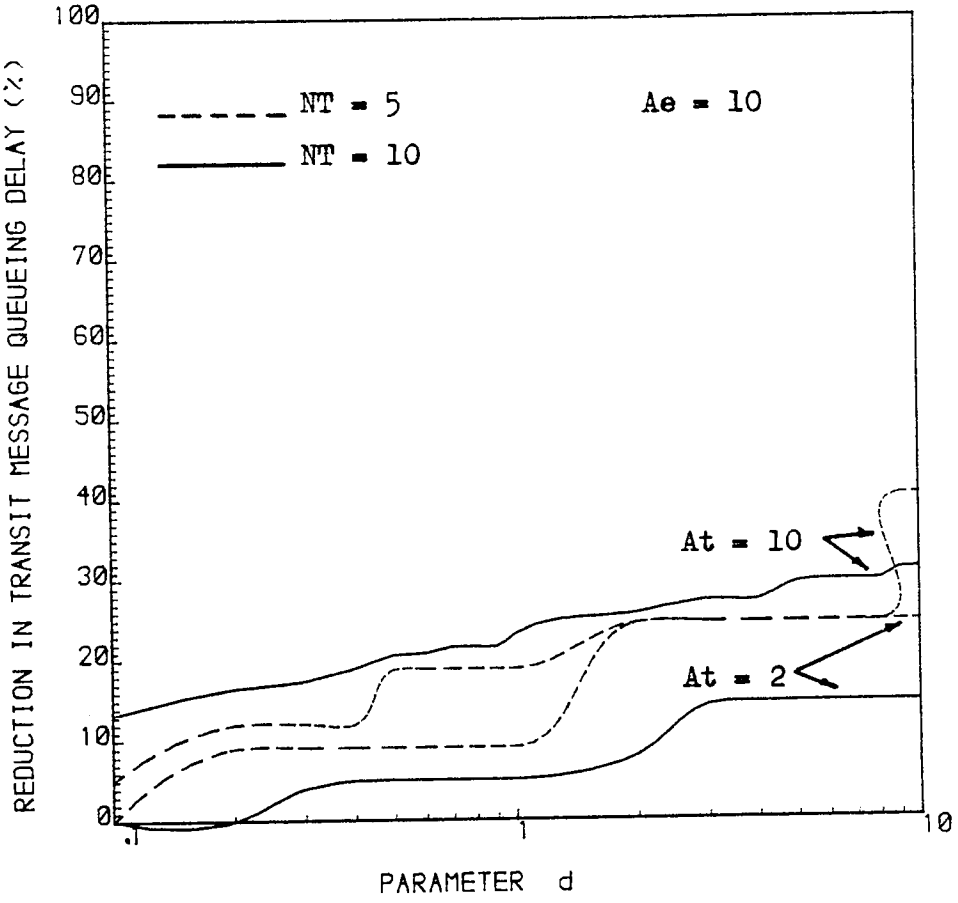


Fig. 7.2 Percentage change in transit and external traffic throughput under the optimum policy as a function of the parameter " d " (NT = 5 and 10).



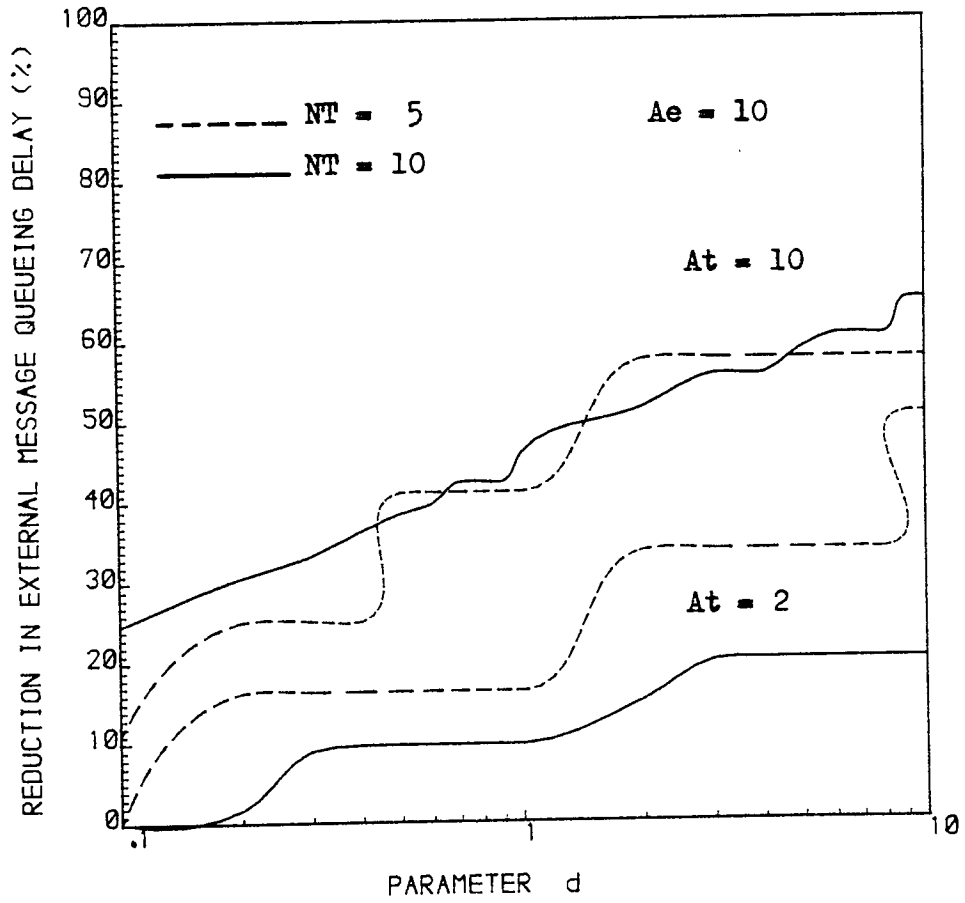
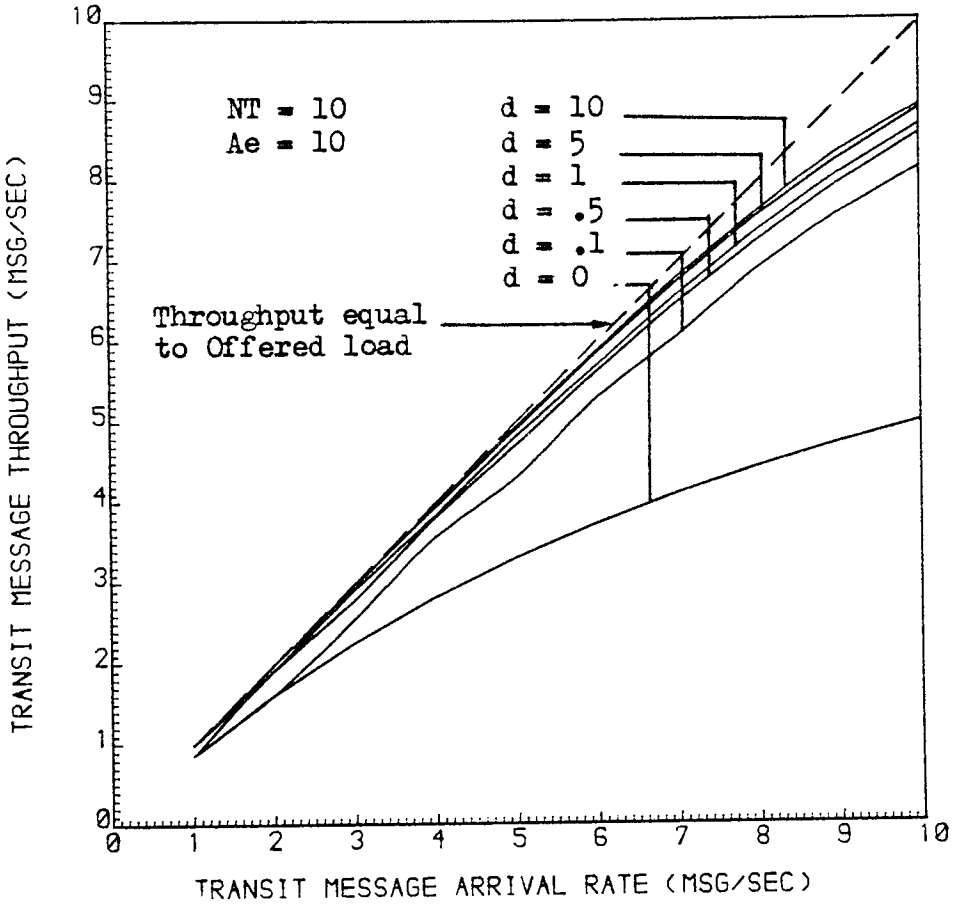


Fig. 7.3 Percentage change in transit and external message queueing delay under the optimum policy, as a function of the parameter " d ".



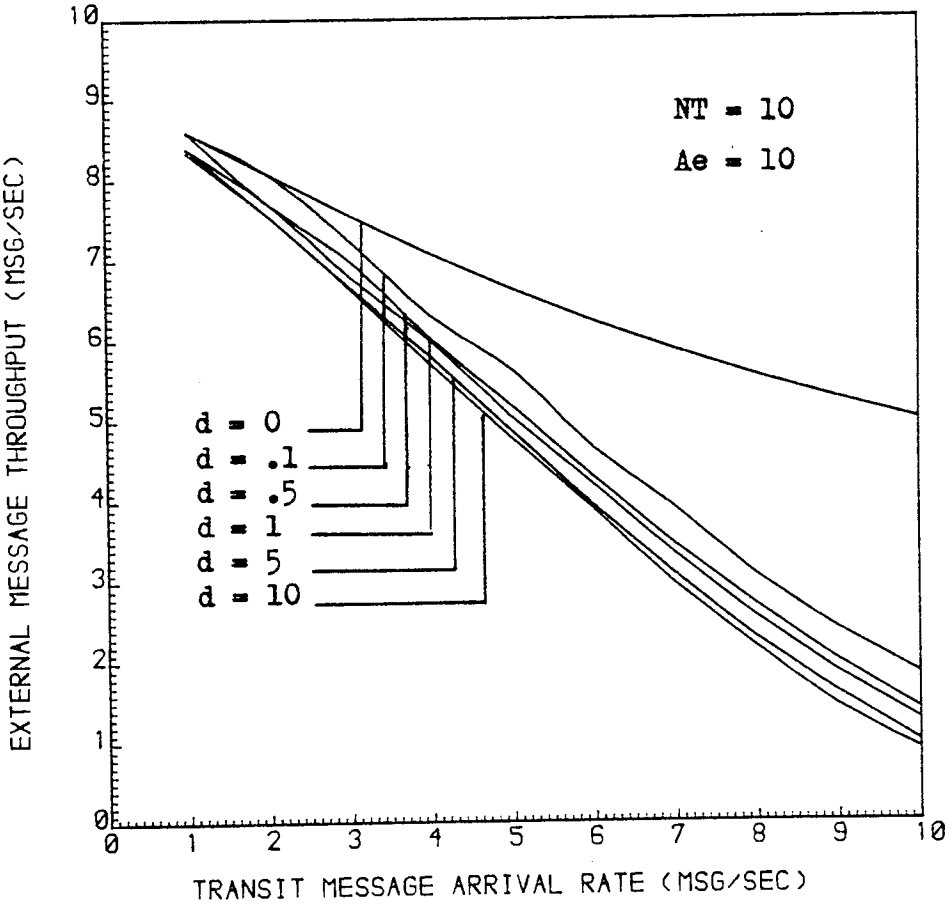
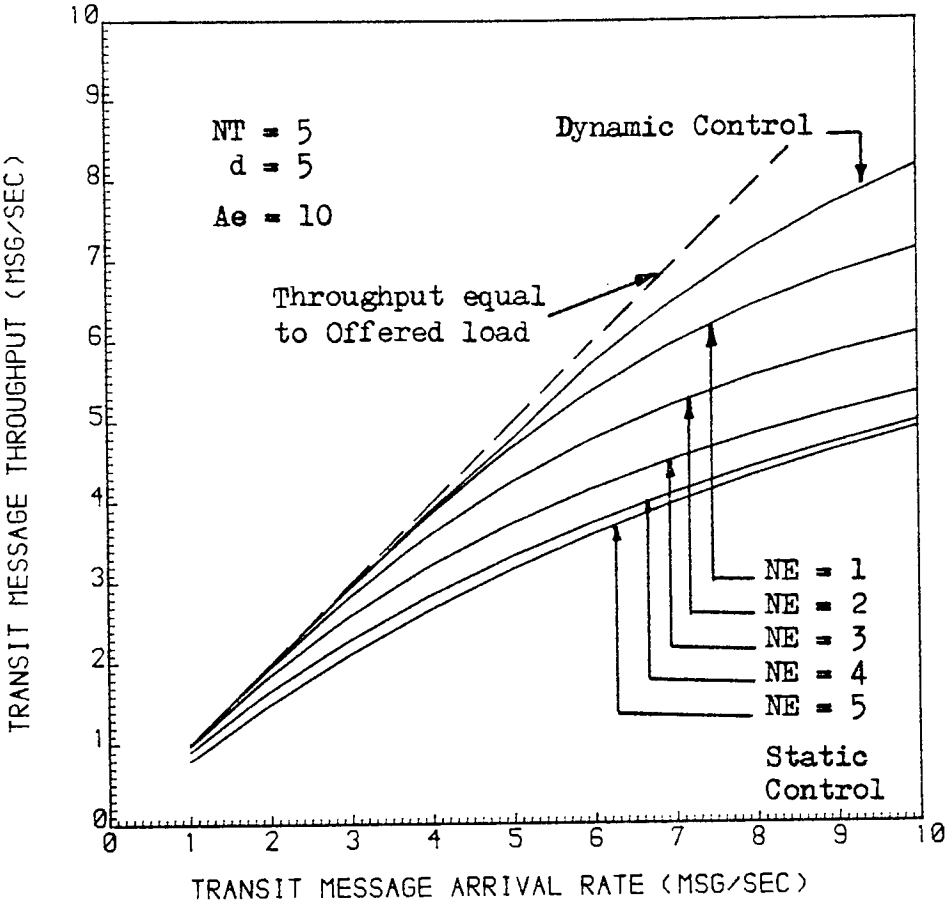


Fig. 7.4 Transit traffic and external traffic throughput as a function of transit message arrival rate, for various values of " d ".



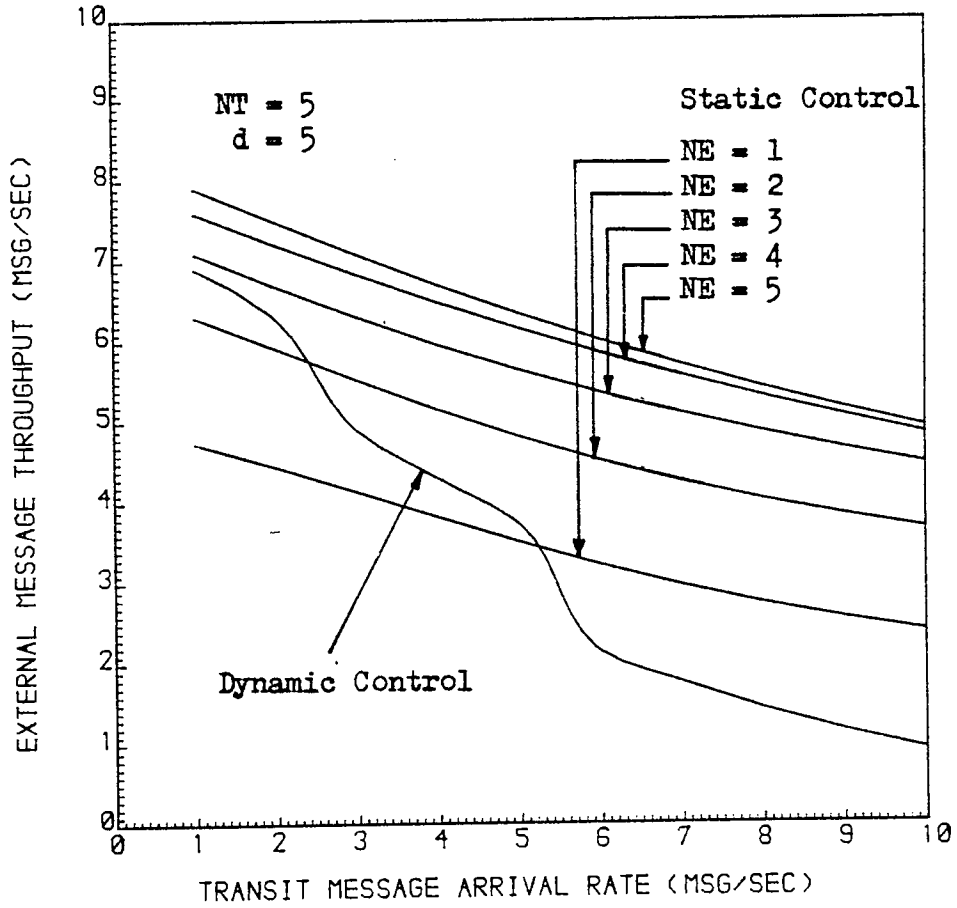
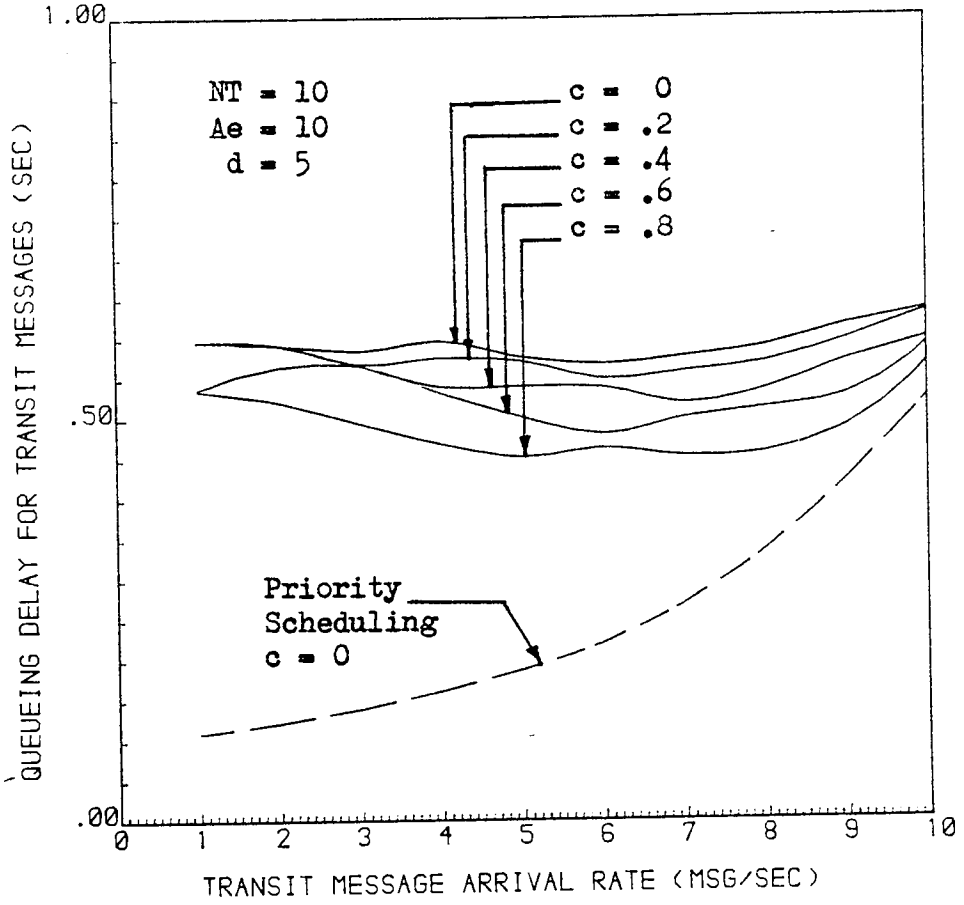


Fig. 7.5 Transit and external message throughput with static and dynamic network access control



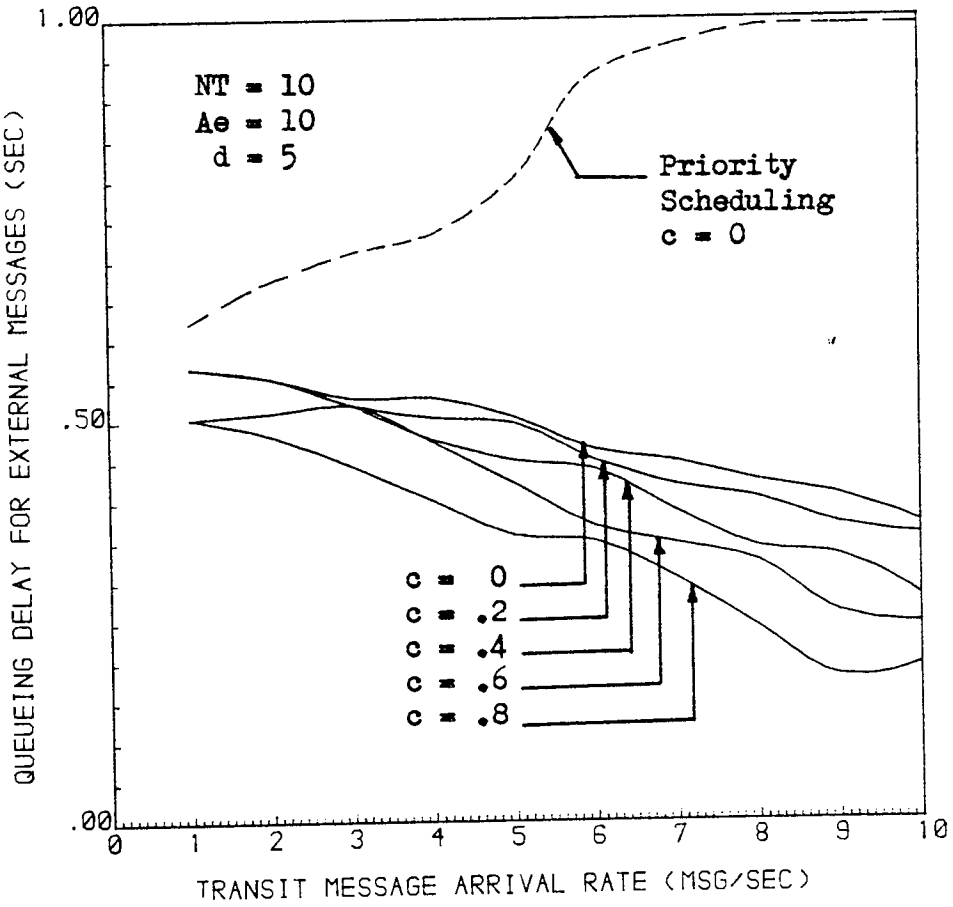


Fig. 7.6 Transit and external message queueing delay as a function of transit message arrival rate, with "c" control, and priority scheduling.

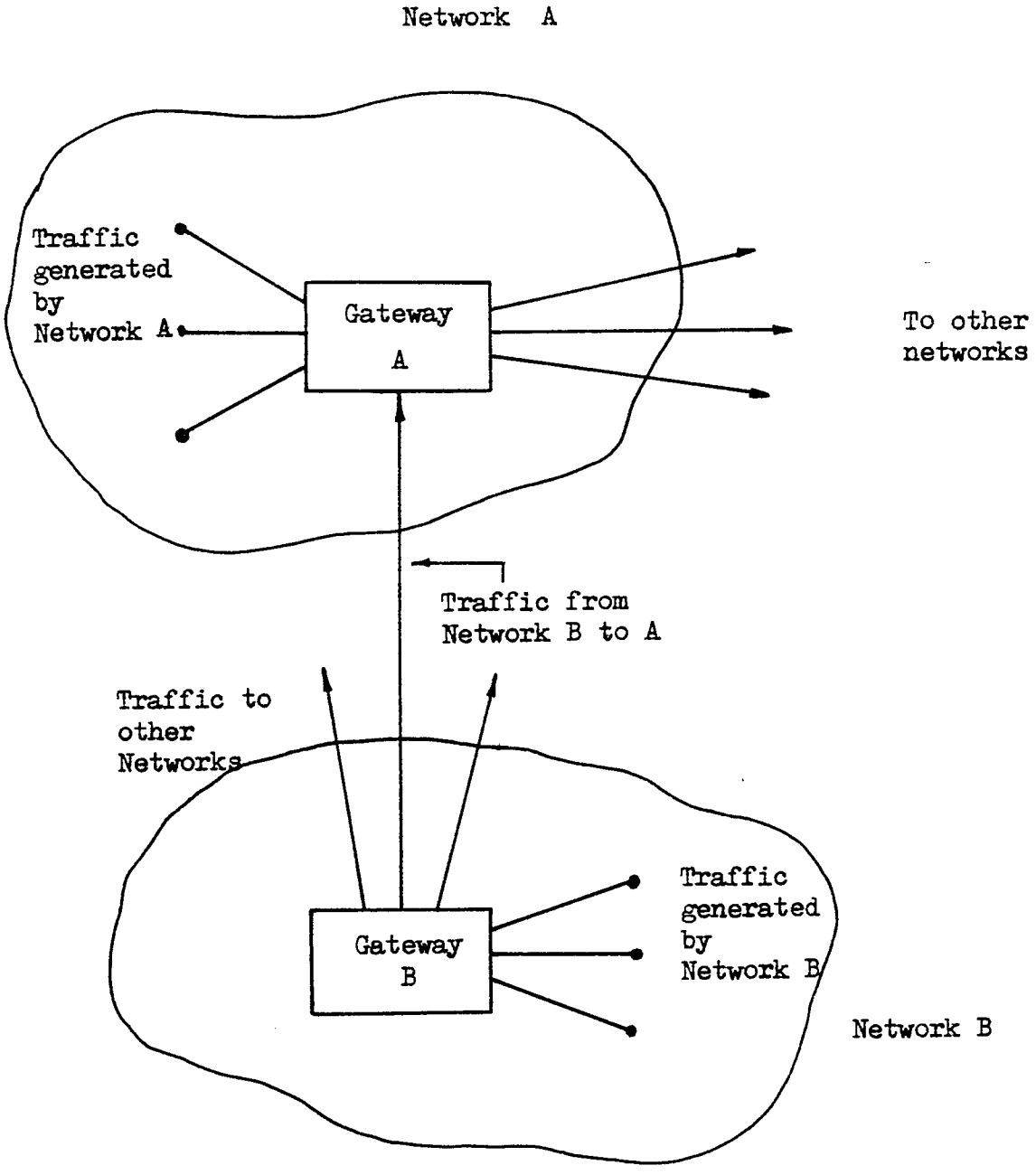


Fig. 7.7 Model of dynamic gateway control

CHAPTER EIGHT

CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH

8. CONCLUSIONS AND RECOMMENDATIONS FOR FURTHER RESEARCH

8.1. CONCLUSIONS

Flow control in computer communication networks is generally a multilayered structure, consisting of several mechanisms operating independently at different levels. The performance evaluation of networks in which different flow control mechanisms operate simultaneously is an important area of research, and formed the subject of this thesis.

In this thesis, closed queueing network theory was used to derive models of finite resource computer communication networks equipped with three levels of flow control. The flow control mechanisms considered were: end-to-end control of virtual circuits, network access control of external messages at the entry nodes, and a hop level control between nodes.

To explore this type of network it was necessary to develop new techniques to overcome the problem of finite resource models, which typically have resource allocation and blocking phenomena that prohibit solution by existing closed queueing theory methods. The model was solved by first extending modern solution techniques such as the convolution algorithm and the mean value analysis algorithm to multi-chain closed queueing networks with finite queues. This approach was further developed to allow the identification of two classes of messages, namely transit messages and

external messages, thus providing a mechanism for the implementation of the network access control. These new methods of analysis were shown to be feasible.

The direct application of the convolution algorithm and the mean value analysis algorithm are computationally unattractive for large networks, and methods such as the heuristic extensions to the mean value analysis algorithm have been evolved to reduce this computational requirement. In this thesis, an equivalent reduced network was derived for the multi-chain closed queueing network with finite queues, such that the reduced or aggregated network could be solved by either the convolution algorithm or the mean value analysis algorithm to give the approximate queue length probabilities. The equivalent reduced network was itself derived using the chain throughputs obtained by using the heuristic extensions to the mean value analysis algorithm. The method was shown to be feasible, and for the five node network investigated in this thesis, the results obtained by the heuristic methods and those obtained by the exact method differed by less than 10%, while the computational effort was reduced by a factor of 50. In general if there are R chains with population K, and L chains visit a queue (worst case), the computational effort is reduced by a factor of K^{R-L}.

The heuristic method based on the equivalent reduced network can be used to model large networks with many virtual circuits, especially when the queues are sparsely popu-

lated (i.e., when the number of chains, or the chain population at individual queues in the the network are small). For networks where this condition is not satisfied (i.e., the chains are not dispersed), a method for the further aggregation of the equivalent reduced network was developed using the Norton's theorem for closed queueing networks.

A five node network with finite buffers and three levels of flow control was modelled as a closed queueing network with finite buffers, and was solved using the heuristic method based on the equivalent reduced network and the heuristic extensions to the mean value analysis algorithm. The five node network was also simulated on a computer, and the results obtained by the simulation method were compared with those obtained by the analytical methods. For small blocking probabilities (less than 0.5), the virtual circuit throughputs and transit delays obtained by the heuristic methods and by simulation differed by less than 10%. The heuristic method is fast and requires lower storage, compared to the exact algorithms. The closed queueing model and its heuristic solution give accuracies that are more than adequate for potential application of the model such as the design phase selection of window limits, selection of the number of buffers at each node, choice of the network access control parameter and the location of the bottle-neck for a given traffic profile, etc.

The interaction between the three levels of flow con-

trol were investigated for the five node network. End-to-end control of virtual circuits was shown to prevent congestion in the network when the window limits were small. When the window limits were large, the network performance deteriorated at high offered loads because of the large number of messages admitted to the network. However, the degradation in the network performance was shown to be reduced by the introduction of a network access control, with correctly chosen control parameters. The optimum input buffer limit for external messages was found to be in agreement with the heuristic choice suggested in earlier work (viz., NE=NT/(1+H), where, NE is the input buffer limit, NT is the total number of buffers at a node, and H is the mean hop length).

A queueing model for the admission delay in finite resource networks with many virtual circuits was developed. The virtual circuits were assumed to be controlled by an end-to-end flow control mechanism. Messages which arrived from independent Poisson sources waited outside the network (in a queue called the admission queue or the token access queue), until they secured a free token. The effect of buffering messages outside the network was to increase the effective arrival rate of messages to the network. An optimum token limit was found by maximising power, defined as the ratio of throughput to delay, where the delay included the the network transit delay as well as the admission delay. For maximum power the optimum token limit was

found to be twice the hop length.

A number of advanced network access methods to improve the network performance, as well as that of selected traffic streams were explored. The new access methods were incorporated in the five node network, and the model solved using the heuristic method based on the equivalent reduced network. The introduction of a buffer access queue (with a suitably chosen service rate parameter) in which messages with tokens waited until they were admitted to the entry node, limited the maximum arrival rate of messages at the entry node. The buffer access queue helped to hold the virtual circuit throughputs at their peak value even for large imposed loads. The introduction of a token access queue in which messages waited until they secured free tokens, transformed the model from a loss model to a wait model. While the token access queue improved the network performance at low loads, the buffer access queue stabilised the network performance at high loads. With the introduction of the token access queue and the buffer access queue, the network performance was shown to improve further and near optimum control was achieved.

Three new access schemes based on priority access of virtual circuit tokens were developed, to give preferential treatment to selected traffic streams and improve their throughput, relative to the other traffic streams in the network. The method based on a preemptive priority access

protocol improved the throughput of priority traffic by as much as 80%, when the ratio of priority traffic to total traffic was low (the comparisons were made with respect to a network without the priority scheme). A second method used a local network and a local pool of tokens to gain access to the virtual route tokens. The relative throughput of the sources could be controlled over a wide range of offered loads, by simply varying the local token ratio. The third method removed the network access control for priority traffic, thereby giving them better access to the entry node buffers. The method based on preemptive priority access is simple and easy to implement, but is only effective when the ratio of priority traffic to non-priority traffic is low. The method based on local access control provides total control over the throughput of the priority and non-priority sources, over a wide range of offered loads. The third method improved the throughput of the priority stream by about 30%. However, it could be implemented only when the volume of priority traffic is very low.

Using the theory of Markov decision process, an optimal policy for the dynamic control of external messages at the entry node was developed. While maximising the transit traffic throughput the dynamic control improved the nodal buffer utilisation and throughput, at low transit traffic loads. The queueing delay for transit messages were reduced further, by using priority scheduling for transit messages under the optimum policy. The throughput-delay performance

was found to be better with dynamic control, than with static control. The formulation of the model was general, and the application of the model to other flow control problems were outlined.

8.2. RECOMMENDATIONS FOR FURTHER RESEARCH

The assumptions necessary for the closed queueing model to have a product form solution, impose severe limitations on the communication network model. The closed queueing model assumes that all messages have the same exponential length distribution. This is valid only if all messages in the network belong to the same class. By developing additional heuristics, the closed queueing models developed in this thesis could be extended to networks, where different chains have different message length distribution. Secondly, the hop level control assumes complete sharing of buffers at every node. Complete sharing can lead to heavily loaded traffic streams monopolising all the buffers. This can be remedied by imposing additional constraints on individual queue lengths. It seems feasible for these constraints to be included in the closed queueing network model using techniques similar to those developed in this thesis.

Hybrid packet and circuit switching networks are emerging to meet multimode user requirements (voice and data; interactive and batch). The design and analysis of such integrated switching systems are still in their infancy. These networks require new flow control techniques, since

existing flow control protocols only apply to packet switched messages. Flow control of the circuit switched component is essential to prevent unfairness to the packet switched components. If the circuit switched components are not controlled, while the packet switched components are, then the circuit switched components could monopolise all the network resources.

Traditionally, routing and flow control have been developed independently. However, the interaction between flow control and (adaptive) routing algorithms is a strong one. In fact, routing and flow control can be brought together into useful cooperation. Performance evaluation of flow control protocols without considering the effects of adaptive routing, can be misleading. This is because adaptive routing diverts traffic from congested paths to less congested paths. Thus, the congestion problem must be considered as a whole.

The scope of the dynamic flow control models based on Markov decision process is limited by the large state space, which grows with the number of variables. State space reduction techniques can help to model dynamic systems with large number of variables. Further, with a birth-death assumption which only allows transitions between adjacent states, the elements of the state transition probability matrix is largely made up of zeroes or is sparse. Application of sparse matrix techniques could result in consider-

able reduction in storage space.

APPENDICES

APPENDIX: A

COMPUTATION OF THE NORMALISATION CONSTANT G(K) AND THE MARGINAL QUEUE LENGTH PROBABILITIES THROUGH THE CONVOLUTION ALGORITHM

This section presents the computation of the normalisation constant G(K) and the improper marginal queue length probabilities $\overset{\star}{P}m(Nex,Nts|\ K)$, and the constant $G_{+i}(K)$ of the closed queueing network, using the Reiser Kobayashi convolution algorithm [67,68,69].

Let the network be defined by L first-in-first out queues and R closed chains. Let the population of the closed chains be represented by the vector $K=K1,K2,\ldots,KR$.

ALGORITHM 1 - DETERMINATION OF THE CONSTANT G(K)

1] Set up an R dimensional array G

2] Initialise all elements of array G := 0

3] Initialise i := 1

4] Initialise G(0,0,...,0) := 1

5] Loop on k

For $k=1,2,\ldots,\sum Kr$ (in this order) perform step 6

6] Loops on Y(Y=Y1,Y2,...,YR)

For Y1=0,1,...,K1; Y2=0,1,...,K2; YR=0,1,...KR

Such that $|Y|=k$, $Y \leqslant K$ and $Y \neq 0$; evaluate

$$G(Y) := G(Y) + \sum_{r=1} Uir\ G(Y-yr)$$

7] Loop on i

For i := i+1 if i < L go to step 5.

The normalisation constant G(K)=1/C is given by,

$$C = 1/G(K)$$
$$= 1/G(K1,K2,\ldots,KR).$$

During the evaluation of G(K), the constants G(Y) for all chain population $0 \leqslant Y \leqslant K$ is also obtained.

ALGORITHM 2 - DETERMINATION OF THE IMPROPER MARGINAL QUEUE LENGTH PROBABILITIES

1] Set up an R+2 dimensional array $P^{*}i$

2] Initialise all elements of the array, $P^{*}i := 0$

3] Initialise $P^{*}(0,0|\ 0) := 1$

4] Loop on k

For $k=1,2,\ldots,\sum Kr$ in this order perform step 5

5] Loop on $Y=Y1,Y2,\ldots,YR$

For $Y1=0,1,\ldots,K1$; $Y2=0,1,\ldots,K2$; $YR=0,1,\ldots,KR$

such that $|Y|=k$, $Y \leqslant K$ and $Y \neq 0$, evaluate for all Y

$$P^{*}(0,0|\ Y) = G(Y) - \sum_{r=1}^{R} Uir\ G(Y-yr)$$

and perform,

Loop on N For $N=1,2,\ldots,\text{Min}(N_{+}, \sum_{r \in R(i)} Yr\ )$

where $Ne_{+} = \sum_{r \in Re(i)} Kr$ and $Nt_{+} = \sum_{r \in Rt(i)} Kr$

and $N_{+} = Ne_{+} + Nt_{+}$

For $Nex=0,1,\ldots,\text{Min}(Ne_{+}, \sum_{r \in Re(i)} Yr\ )$

$Nts=0,1,\ldots,\text{Min}(Nt_{+}, \sum_{r \in Rt(i)} Yr\ )$

such that $Nex + Nts = N$ perform

$$P^{*}i(Nex,Nts|\ Y) = \sum_{r \in Re(i)} Uir\ P\ i(Nex-1,Nts|\ Y-yr) + \sum_{r \in Rt(i)} Uir\ P\ i(Nex,Nts-1|\ Y-yr)$$

The proper marginal queue length probabilities are given by,

$$Pi(Nex,Nts|\ K) = P^{*}i(Nex,Nts|\ K)\ /G(K)\ .$$

ALGORITHM 3 - DETERMINATION OF $G_+i(K)$

1] set up an R dimensional array G_+i

2] Initialise all elements $G_+i := 0$

3] Loop on k For $k=0,1,2,\ldots,\sum Kr$

in this order perform step 4

4] Loop on $Y=Y1,Y2,\ldots,YR$

For $Y1=0,1,\ldots,K1$; $Y2=0,1,\ldots,K2$; $YR=0,1,\ldots,KR$

evaluate

$$G_+i(Y) := G(Y) + \sum_{r \in R(i)} Uir\ G_+i(Y-yr)$$

such that

$|Y|=k$ and $Y \leqslant K$.

APPENDIX: B

```
C*************************************************************
C
C    A FORTRAN program for the performance evaluation of a
C    five node network with finite buffers and three levels
C    of flow control, using the method of the equivalent
C    reduced network and the heuristic extensions to the
C    mean value analysis algorithm
C
C    The flow control mechanisms considered are:
C    End-to-end control of virtual circuits
C    Network access control of external messages at entry
C    nodes
C    Hop level control between nodes
C
C*************************************************************
C
C
C         NUMBER OF VIRTUAL CIRCUITS.........................4
C         NUMBER OF BUFFERS AT EACH NODE.....................NT
C         NETWORK ACCESS CONTROL PARAMETER...................NE
C         VIRTUAL CIRCUIT WINDOW LIMIT.......................KW
C         CHANNEL CAPACIRTY..................................SR
C         MEAN ARRIVAL RATE OF MESSAGES......................AR
C         MEAN LENGTH OF A MESSAGE...........................1
C         NODAL BLOCKING PROBABILITY FOR TRANSIT MESSAGES...BT
C         NODAL BLOCKING PROBABILITY FOR EXTERNAL MESSAGES..BE
C         ROUTING MATRIX.....................................IM
C         VIRTUAL CIRCUIT THROUGHPUT.........................TP
C         VIRTUAL CIRCUIT END-TO-END DELAY...................D
C
C
C
          DIMENSION SR(14),BT(5),BE(5),BTN(5),BEN(5),DFA(5)
          DIMENSION DFB(5),AR(4),EB(14),AL(10),TP(4),ANA(4)
          DIMENSION D(4),KW(4),ESR(14),EX(14,4),AESR(14)
          DIMENSION ELZ(14,0:4),ETZ(14,0:4),EM(14,4),IM(14,4)
          DIMENSION B2E(5),B2T(5),TX(4)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C         Define all parameters
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
          DATA NE/3/,NT/5/,IMAX/50/
          DATA KW(1)/4/,KW(2)/4/,KW(3)/4/,KW(4)/4/
          DATA AR1/1./,AR2/1./,AR3/1./,AR4/1./
          DATA SR(1)/10./,SR(2)/10./,SR(3)/10./,SR(4)/10./
          DATA SR(5)/10./,SR(6)/10./,SR(7)/10./,SR(8)/10./
          DATA SR(9)/10./,SR(10)/10./
```

```
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C        Define the routing probability matrix
C        IM(IX,IY) = 0 or 1, where IM(IX,IY) is the
C        probability of virtual circuit IY visiting queue IX
C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
         INP=1
         DO 5=1,14
         DO 5 J=0,4
         IM(I,J)=0
       5 CONTINUE
         IM(1,1)=1
         IM(7,1)=1
         IM(9,1)=1
         IM(10,1)=1
         IM(11,1)=1
         IM(3,2)=1
         IM(2,2)=1
         IM(6,2)=1
         IM(12,2)=1
         IM(7,3)=1
         IM(9,3)=1
         IM(10,3)=1
         IM(13,3)=1
         IM(5,4)=1
         IM(4,4)=1
         IM(8,4)=1
         IM(14,4)=1
         DO 8 I=1,14
         DO 8 J=1,4
         EX(I,J)=FLOAT(IM(I,J))
       8 CONTINUE
 C
 C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
 C        Define chain IC as external or transit chain
 C        at queue IQ.
 C        IC is a transit chain at IQ if ETZ(IQ,IC)=1
 C        IC is an external chain at  IQ if ELZ(IQ,IC)=1
 C::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
 C
         DO 9 I=1,14
         DO 9 J=0,4
         ELZ(I,J)=0.
         ETZ(I,J)=0.
       9 CONTINUE
         ELZ(1,1)=1.
         ELZ(3,2)=1.
         ELZ(7,3)=1.
         ELZ(5,4)=1.
         ETZ(7,1)=1.
         ETZ(9,1)=1.
         ETZ(10,1)=1.
         ETZ(2,2)=1.
```

```
        ETZ(6,2)=1.
        ETZ(9,3)=1.
        ETZ(10,3)=1.
        ETZ(4,4)=1.
        ETZ(8,4)=1.
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Caption the RESULT file by calling the routine
C       CAPTION
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        CALL CAPTION (NE,NT,KW)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Clear all arrays by calling the routine CLARY
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        CALL CLARY (BT,BE,BTN.BEN,DFA,DFB,TP,B2E,B2T)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Assign a value for the arriival rate of each
C       virtual Circuit
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
     10 AM=FLOAT(INP)
        AR(1)=AM*AR1
        AR(2)=AM*AR2
        AR(3)=AM*AR3
        AR(4)=AM*AR4
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Determine the factors EB to modify the service rate
C       of all queues by calling the routine CETIR
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
     15 CALL CETIR (BE,BT,EB)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Determine the virtual circuit throughput for the
C       current iteration by calling the routine TRU
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        CALL TRU (TX,TP,KW,EX,SR,EB,ATL,ANA,DN,D,TN,AL)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Determine the blocking probabilities for the current
C       iteration by calling the routine  BKBP
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        CALL BKBP (KW,SR,EX,EB,NE,NT,IM,BTN,BEN,TP,ELZ,ETZ)
```

```
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Define the new blocking probability vector and find
C       the error in the blocking probabilities
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
     19 DO 20 I=1,5
        DFA(I)=BTN(I)-BT(I)
        DFB(I)=BEN(I)-BE(I)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       The new blocking vector is given by the mean of the
C       blocking probability vectors obtained from the last
C       three iterations
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        ENBK=(B2E(I)+BE(I)+BEN(I))/3.
        B2E(I)=BE(I)
        BE(I)=ENBK
        TNBK=(B2T(I)+BT(I)+BTN(I))/3.
        B2T(I)=BT(I)
        BT(I)=TNBK
     20 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Find the difference ER1 and ER2 in the blocking
C       probabilities
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        DASUM=0.
        DBSUM=0.
        DO 30 I=1,5
        DASUM=DASUM+(DFA(I)**2)
        DBSUM=DBSUM+(DFB(I)**2)
     30 CONTINUE
        ER1=SQRT(DASUM)
        ER2=SQERT(DBSUM)
        IF(ER1.GT.0.001)GO TO 15
        IF(ER2.GT.0.001)GO TO 15
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       If the blocking probabilities have converged
C       store all results by calling the routine STORE
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        CALL STORE (INP,TN,TP,ATL,ANA,DN,D,BE,BT,AL)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Increment the arrival rate INP
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
        INP=INP+1
        IF(INP.LE.IMAX)GO TO 10
        STOP
        END
```

```
C*************************************************************
C        Subroutine CLARY
C        This routine clears all the arrays
C*************************************************************
C
         SUBROUTINE CLARY (BT,BE,BTN,BEN,DFA,DFB,TP,B2E,B2T)
C
         DIMENSION BT(5),BE(5),BTN(5),BEN(5),DFA(5),DFB(5)
         DO 109 I=1,5
         BT(I)=0.
         BE(I)=0.
         BTN(I)=0.
         BEN(I)=0.
         B2T(I)=0.
         B2E(I)=0.
         DFA(I)=0.
         DFB(I)=0.
     109 CONTINUE
         DO 110 I=1,4
         TP(I)=0.
     110 CONTINUE
         RETURN
         END
C
C*************************************************************
C        Subroutine CETIR
C        This subrouitne determines the parameter EB, to modify
C        the service rate of all queues in the network
C*************************************************************
C
         SUBROUITNE CETIR (BE,BT,EB)
C
         DIMENSION BE(5),BT(5),EB(14)
         EB(1)=1.-BT(2)
         EB(2)=1.-BT(1)
         EB(3)=1.-BT(2)
         EB(4)=1.-BT(3)
         EB(5)=1.-BT(4)
         EB(6)=1.
         EB(7)=1.-BT(4)
         EB(8)=1.
         EB(9)=1.-BT(5)
         EB(10)=1.
         EB(11)=1.-BE(1)
         EB(12)=1.-BE(3)
         EB(13)=1.-BE(2)
         EB(14)=1.-BE(5)
         RETURN
         END
```

```
C*************************************************************
C       Subroutine TRU
C       This subroutine determines the virtual circuit
C       throughput, and the mean end-to-end dealy, using the
C       heuristic extensions to the MVA algorithm
C*************************************************************
C
        SUBROUTINE TRU (TX,TP,KW,EX,SR,EB,ATL,ANA,DN,D,TN,AL)
C
        DIMENSION ANA(4),D(4),SR(14),TX(4),TP(4),SST(14)
        DIMENSION EB(14),SCN(14,4),EX(14,4),CTP(4),ERP(14,4)
        DIMENSION ADL(14),AQL(14,4),TV(4),AL(10),DIL(4)
        DIMENSION KW(4),SCD(14)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Find the correction factor ERS using the single
C       chain problem
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        KRM=0
    202 DO 210 IC=1,4
        DO 203 IQ=1,14
        SCN(IQ,IC)=0.
    203 CONTINUE
        DO 206 IQ=1,14
        ZAM=0.
        DO 205 IC1=1,4
        ZAM=ZAM+(TP(IC1)*EX(IQ,IC1))
    205 CONTINUE
        ZAM=ZAM-TP(IC)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Find the service rate SST(IQ) of queue IQ in the
C       single chain problem of virtual circuit IC by
C       removing the background traffic ZAM
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        ESRS=SR(IQ)-ZAM
        SST(IQ)=EX(IQ,IC)/(EB(IQ)*ESRS)
    206 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Find the correction factor ERP(IQ,IC) for virtual
C       circuit IC, for all queues IQ
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        DO 209 IW=1,KW(IC)
        CF=0.
        DO 207 IQ=1,14
        SCD(IQ)=SST(IQ)*(1.+SCN(IQ,IC))*EX(IQ,IC)
        CF=CF+SCD(IQ)
    207 CONTINUE
```

```
          CTP(IC)=FLOAT(IW)/CF
          CX=CTP(IC)*SCD(IQ)
          ERP(IQ,IC)=CX-SCN(IQ,IC)
          IF(ERP(IQ,IC).GE.1.)ERP(IQ,IC)=1.
          IF(ERP(IQ,IC).LE.0.)ERP(IQ,IC)=0.
          SCN(IQ,IC)=CX
      208 CONTINUE
      209 CONTINUE
      210 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C         Initialise the queue length for the first iteration
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
          IF(KRM.EQ.1)GO TO 221
          DO 220 IQ=1,14
          DO 220 IC1=1,4
          AQL(IQ,IC)=SCN(IQ,IC1)
      220 CONTINUE
      221 DO 223 IQ=1,14
          RM=0.
          DO 222 IC1=1,4
          RL=AQL(IQ,IC1)-ERP(IQ,IC1)
          IF(RL.LT.0.)RL=0.
          RM=RM+RL
      222 CONTINUE
          ADL(IQ)=(1.+RM)/(EB(IQ)*SR(IQ))
      223 CONTINUE
          DO 226 IC=1,4
          SUM=0.
          DO 224 IQ=1,14
          SUM=SUM+(ADL(IQ)*EX(IQ,IC))
      224 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C         Determine the mean queue length AL of each queue
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
          AL(1)=AQL(1,1)
          AL(2)=AQL(2,2)
          AL(3)=AQL(3,2)
          AL(4)=AQL(4,4)
          AL(5)=AQL(5,4)
          AL(6)=AQL(6,2)
          AL(7)=AQL(7,1)+AQL(7,3)
          AL(8)=AQL(8,4)
          AL(9)=AQL(9,1)+AQL(9,3)
          AL(10)=AQL(10,1)+AQL(10,3)
```

```
      DO 229 IC=1,4
      DIL(IC)=ATP(IC)-TP(IC)
      IF(KRM.NE.0)GO TO 228
      TP(IC)=ATP(IC)
      TX(IC)=TP(IC)
      TV(IC)=TX(IC)
      GO TO 229
  228 TP(IC)=(TV(IC)+TX(IC)+ATP(IC))/3.
      TV(IC)=TX(IC)
      TX(IC)=TP(IC)
  229 CONTINUE
      KRM=1
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C     Check for convergence of virtual circuit throughput
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
      KCK=0
      DO 230 IC=1,4
      IF(ABS(DIL(IC)).GT.0.0001)KCK=1
  230 CONTINUE
      IF(KCK.NE.0)GO TO 202
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C     Determine the virtual circuit end-to-end delay D(IC)
C     and the mean number messages in transit in each
C     virtual circuit  ANA(IC)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
      SU=0.
      DO 232 IC=1,4
      SL=0.
      SO 231 IQ=1,10
      SL=SL+(ADL(IQ)*EX(IQ,IC))
  231 CONTINUE
      D(IC)=SL
      ANA(IC)=TP(IC)*D(IC)
      SU=SU+ANA(IC)
  232 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C     Determine the mean network throughput TN, the mean
C     network delay DN, and the mean number of messages
C     in the network ATL
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
      ATL=SL
      TN=TP(1)+TP(2)+TP(3)+TP(4)
      DN=ATL/TN
      RETURN
      END
```

```
C*************************************************************
C        Subroutine BKBP
C        This subroutine determines the nodal blocking
C        probabilities for transit and external messages
C        using the method of the equivalent reduced network
C        and the virtual circuit throughput obtained from
C        the routine TRU
C*************************************************************
C
         SUBROUTINE BKBP(KW,SR,EX,EB,NE,NT,BTN,BEN,TP,ELZ,ETZ)
C
         DIMENSION KW(4),EB(14),TP(4),SR(14),AESR(14),EX(14,4)
         DIMENSION IM(14,0:4),BTN(5),BEN(5),MI(2),MS(2)
         DIMENSION ADX(0:6,0:6,0:6,0:6),GW(0:6,0:6)
         DIMENSION ETZ(14,0:4),PMV(0:6,0:6,2),PM(0:6,0:6)
         DIMENSION ELZ(14,0:4),NQU(2),NCO(2)
         DO 530 NOD=1,5
         NQU(1)=NOD
         NQU(2)=NOD+5
         DO 510 NQE=1,2
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C        Form the equivalend reduced network of queue
C        NQU(NQE), at node NOD
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
         NCOUNT=1
         NCO(1)=0
         NCO(2)=0
         N1=0
         N2=0
         DO 417 I=1,14
         AESR(I)=SR(I)
     417 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C        Rename the virtual circuit IC visiting queue
C        NQU(NQE) as NCO(IC).  This eliminates all other
C        virtual circuits not visitine the queue NQU(NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
         DO 420 IC=1,4
         IF(IM(NQU(NQE),IC).EQ.0)GO TO 420
         NCO(NCOUNT)=1C
         NCOUNT=NCOUNT+1
     420 CONTINUE
         IF(NCO(1).EQ.0)GO TO 421
         N1=KW(NCO(1))
     421 IF(NCO(2).EQ.0)GO TO 422
         N2=KW(NCO(2))
```

```
    422 NES=N1+N2
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Modify the service rates of queues in the equivalent
C       reduced network, by subtracting the background
C       traffic SUM from the service rate SR of the queue
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        DO 419 IQ=1,10
        IF(IQ.EQ.NQU(NQE))GO TO 419
        SUM=0.
        DO 418 IC=1,4
        IF(IC.EQ.NCO(1))GO TO 418
        IF(IC.EQ.NCO(2))GO TO 418
        IF(IM(IQ,IC).EQ.0)GO TO 418
        SUM=SUM+TP(IC)
    418 CONTINUE
        AESR(IQ)=SR(IQ)-SUM
    419 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Determine the normalisation constant GW of the
C       equivalent reduced network of queue NQU(NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        DO 430 M2=0,N2
        DO 430 M1=0,N1
        GW(M1,M2)=0.
    430 CONTINUE
        GW(0,0)=1.
        DO 450 IQ=1,14
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Check if queue IQ is a queue in the complement
C       network of queue NQU(NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        IF(IQ.NE.NQU(NQE))GO TO 440
    431 DO 438 NB=1,NES
        DO 438 M2=0,N2
        DO 438 M1=0,N1
        IF((M1+M2).NE.MB)GO TO 438
        MS(1)=M1
        MS(2)=M2
        G1=GW(M1,M2)
        G2=0.
        DO 435 MC=1,2
        IF(NCO(MC).EQ.0)GO TO 435
        IF(MS(MC)-1).LT.0)GO TO 435
        DO 433 MZ=1,2
        IF(MZ.NE.MC)GO TO 432
        MI(MZ)=MS(MZ)-1
        GO TO 433
    432 MI(MZ)=MS(MZ)
```

```
  433 CONTINUE
      G2=G2+(EX(IQ,NCO(MC))*GW(MI(1),MI(2))/EB(IQ)*SR(IQ)))
  435 CONTINUE
      GW(M1,M2)=G1+G2
  438 CONTINUE
      GO TO 450
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Queue IQ is a queue in the complement network of
C       queue NQU(NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
  440 DO 445 NB=1,NES
      DO 445 M2=0,N2
      DO 445 M1=0,N1
      IF((M1+M2).NE.NB)GO TO 445
      MS(1)=M1
      MS(2)=M2
      G2=0.
      DO 444 NCL=1,2
      IF(NCO(NCL).EQ.0)GO TO 444
      IF(IM(IQ,NCO(NCL)).EQ.0)GO TO 444
      IF(MS(NCL)-1).LT.0)GO TO 444
      DO 443 MA=1,2
      IF(MA.NE.NCL)GO TO 441
      MI(MA)=MS(MA)-1
      GO TO 443
  441 MI(MA)=MS(MA)
  443 CONTINUE
      G2=G2+(GW(MI(1),MI(2))*EX(IQ,NCO(NCL))/
     *(EB(IQ)*AESR(IQ)))
  444 CONTINUE
      GW(M1,M2)=GW(M1,M2)+G2
  445 CONTINUE
  450 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Evaluate the marginal queue length probability
C       PMV(NEXP,NTX,NQE) of queue NQU(NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
      DO 458 NEX=0,NT
      DO 458 NTX=0,NT
      PMV(NEX,NTX)=0.
  458 CONTINUE
      DO 460 I1=0,NT
      DO 460 I2=0,NT
      DO 460 I3=0,N1
      DO 460 I4=0,N2
      ADX(I1,I2,I3,I4)=0.
  460 CONTINUE
```

```
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Find PMV(0,0,NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        ADX(0,0,0,0)=1.
        DO 470 MP=1,NES
        DO 470 M2=0,N2
        DO 470 M1=0,N1
        IF((M1+M2).NE.MP)GO TO 470
        MS(1)=M1
        MS(2)=M2
        AD1=GW(M1,M2)
        AD2=0.
        DO 486 MG=1,2
        IF(NCO(MG).EQ.0)GO TO 468
        IF((MS(MG)-1).LT.0)GO TO 468
        DO 466 MZ=1,2
        IF(MZ.NE.MG)GO TO 464
        MI(MZ)=MS(MZ)-1
        GO TO 466
    464 MI(MZ)=MS(MZ)
    466 CONTINUE
        AD3=EX(NQU(NQE),NCO(MG))*GW(MI(1),MI(2))
        AD3=AD3/(EB(NQU(NQE))*SR(NQU(NQE)))
        AD2=AD2+AD3
    468 CONTINUE
        ADX(0,0,M1,M2)=AD1-AD2
    470 CONTINUE
        PMV(0,0,NQE)=ADX(0,0,N1,N2)/GW(N1,N2)
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       For NTX=0 to NT, NEX=0 to NT, and NEX+NTX≠0
C       find PMV(NEX,NTX,NQE)
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
        DO 500 MP=1,NES
        DO 500 M2=0,N2
        DO 500 M1=0,N1
        IF((M1+M2).NE.MP)GO TO 500
        MS(2)=M2
        MS(1)=M1
        DO 495 NTX=0,MIN(NT,MP)
        DO 495 NEX=0,MIN(NT,MP)
        AQ1=0.
        AQ2=0.
        MY1=NEX-1
        MY2=NTX-1
        MO=0
        IF(ELZ(NQU(NQE),NCO(1)).EQ.0.)GO TO 474
        MO=MO+1
    474 IF(ELZ(NQU(NQE),NCO(2)).EQ.0.)GO TO 476
        MO=MO+M2
    476 IF(MO.LT.NEX)GO TO 495
```

```
      MO=0
      IF(ETZ(NQU(NQE),NCO(1)).EQ.0.)GO TO 482
      MO=MO+M1
  482 IF(ETZ(NQU(NQE),NCO(2)).EQ.0.)GO TO 484
      MO=MO+M2
  484 IF(MO.LT.NTX)GO TO 495
      IF(MY1.GE.0)GO TO 485
      IF(MY2.GE.0)GO TO 489
      GO TO 495
  485 DO 488 MQ=1,2
      IF(NCO(MQ).EQ.0)GO TO 488
      IF((MS(MQ)-1).LT.0)GO TO 488
      DO 487 MZ=1,2
      IF(MZ.NE.MQ)GO TO 486
      MI(MZ)=MS(MZ)-1
      GO TO 487
  486 MI(MZ)=MS(MZ)
  487 CONTINUE
      AQ3=ELZ(NQU(NQE),NCO(MQ))*ADX(MY1,NTX,MI(1),MI(2))
      AQ3=AQ3/(EB(NQU(NQE))*SR(NQU(NQE)))
      AQ1=AQ1+AQ3
  488 CONTINUE
      IF(MY2.LT.0)GO TO 494
  489 DO 492 MQ=1,2
      IF(NCO(MQ).EQ.0)GO TO 492
      IF(MS(MQ)-1).LT.0)GO TO 492
      DO 491 MZ=1,2
      IF(MZ.NE.MQ)GO TO 490
      MI(MZ)=MS(MZ)-1
      GO TO 491
  490 MI(MZ)=MS(MZ)
  491 CONTINUE
      AQ4=ETZ(NQU(NQE),NCO(MQ))*ADX(NEX,MY2,MI(1),MI(2))
      AQ4=AQ4/(EB(NQU(NQE))*SR(NQU(NQE)))
      AQ2=AQ2+AQ4
  492 CONTINUE
  494 ADX(NEX,NTX,M1,M2)=AQ1+AQ2
  495 CONTINUE
  500 CONTINUE
      DO 505 NB=1,NT
      DO 505 NTX=0,NB
      NEX=NB-NTX
      PMV(NEX,NTX,NQE)=ADX(NEX,NTX,N1,N2)/GW(N1,N2)
  505 CONTINUE
  510 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C       Evaluate the population probability PM(NEX,NTX) of
C       each node NOD, for NEX+NTX = 0 to NT
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
      DO 522 NB=0,NT
      DO 522 NEX=0,NB
```

```
          NTX=NB-NEX
          PUF=0.
          DO 520 NTA=0,NTX
          DO 520 NEA=0,NEX
          NTB=NTX-NTA
          NEB=NEX-NEA
          PUF=PUF+(PMV(NEA,NTA,1)*PMV(NEB,NTB,2))
      520 CONTINUE
          PM(NEX,NTX)=PUF
      522 CONTINUE
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C         Evaluate the overflow probability BTN(NOD) for
C         transit messages at node NOD
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
          PIX=0.
          DO 550 NB=0,NT-1
          DO 550 NEX=0,NB
          NTX=NB-NEX
          PIX=PIX+PM(NEX,NTX)
      550 CONTINUE
          BTN(NOD)=1.-PIX
C
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C         Evaluate the overflow probability BEN(NOD) for
C         external messages at entry node NOD
C:::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::::
C
          PIX=0.
          DO 555 NB=0,NT-1
          NAM=MIN(NB,NE-1)
          DO 554 NEX=0,NAM
          NTX=NB-NEX
          PIX=PIX+PM(NEX,NTX)
      554 CONTINUE
      555 CONTINUE
          BEN(NOD)=1.-PIX
      530 CONTINUE
          RETURN
          END
```

```
C*************************************************************
C       Subroutine CAPTION
C       This routine captions the files containing the results
C*************************************************************
        SUBROUTINE CAPTION (NE,NT,KW)
C
        DIMENSION KW(4)
        DO 620 I=16,19
        WRITE(I,610)
        WRITE(I,615)NE,NT,KW(1),KW(2),KW(3),KW(4)
    610 FORMAT(1X,' NE   NT   KW1   KW2   KW3   KW4')
    615 FORMAT(3X,6(I2,3X))
    620 CONTINUE
        WRITE(16,630)
    630 FORMAT(1X,'  INP   TP   TP1   TP2   TP3   TP4',
       *'   NN   N1   N2   N3   N4')
        WRITE(17,632)
    632 FORMAT(1X,'  INP   DN    D1    D2    D3    D4')
        WRITE(18,634)
    634 FORMAT(1X,'  INP   BE1   BE2   BE3   BE4   BE5',
       *'   BT1   BT2   BT3   BT4   BT5')
        WRITE(19,636)
    636 FORMAT(1X,'  INP   AL1   AL2   AL3   AL4   AL5',
       *'   AL6   AL7   AL8   AL9   AL10')
        RETURN
        END
C
C*************************************************************
C       Subroutine STORE
C       This routine stores all the results obtained for a
C       given set of arrival rates
C*************************************************************
C
        SUBROUTINE STORE (INP,TN,TP,ATL,ANA,DN,D,BE,BT,AL)
C
        DIMENSION TP(4),ANA(4),D(4),BE(5),BT(5),AL(10)
        WRITE(16,640)INP,TN,TP(1),TP(2),TP(3),TP(4),
       *ATL,ANA(1),ANA(2),ANA(3),ANA(4)
    640 FORMAT(1X,I3,2X,5(F7.3,2X),5(F6.3,2X))
        WRITE(17,641)INP,DN,D(1),D(2),D(3),D(4)
    641 FORMAT(1X,I3,2X,5(F7.3,2X))
        WRITE(18,642)INP,BE(1),BE(2),BE(3),BE(4),BE(5),
       *BT(1),BT(2),BT(3),BT(4),BT(5)
    642 FORMAT(1X.I3,2X,10(F10.8,2X))
        WRITE(19,643)INP,AL(1),AL(2),AL(3),AL(4),AL(5),
       *AL(6),AL(7),AL(8),AL(9),AL(10)
    643 FORMAT(1X,I3,2X,10(F6.3,2X))
        RETURN
        END
```

APPENDIX: C

POLICY ITERATION ALGORITHM

The policy iteration algorithm of Howard [28,29], involves two routines, namely, the value determination operation and the policy improvement routine. The value determination operation yields the gain G and the value Vi i=1,2,...,N. Where, N is the number of states, Vi=Vi(0), and Vi(t) are the total reward the system expects to earn in a time t, given that the initial state is i. The gain G and the Vi's are determined by solving the N linear homogeneous equations,

$$G = Qi + \sum_{j=1}^{N} Pij\ Vj$$

$$i=1,2,\ldots,N.$$

Where, Qi is the expected immediate earning rate of the system, in the state i, and Pij is the transition rate from state i to state j. Since there are (N+1) unknowns and only N equations, the relative values of Vi are determined by setting VN=0.

Qi and Vj are obtained from the policy improvement routine. The routine chooses the alternative K in the state i, that maximises the test quantity,

$$Qi(k) + \sum_{j=1}^{N} Pij(k)\ Vj$$

$$i=1,2,\ldots,N,$$

using the relative values of Vi of the previous policy. Then K becomes the new decision in the state i, and Qi(K) becomes Qi and Pij(K) becomes Pij.

The value determination routine yields G and Vi corresponding to a given choice of Qi and Pij. The policy improvement routine yields Pij and Qi that increase the gain for a given set of Vi. In other words, the value determination operation yields values as a function of policy, and the policy improvement routine yields the policy as a function of values. The iteration cycle can be started with either of the two routines. It is often convenient to start the process in the policy improvement routine, with all Vi=0. The iteration cycle is stopped when the policies on two successive iteration are identical. Each succeeding policy found has a higher gain than the previous one. The iteration terminates on the policy that has the largest attainable gain within the realms of the problem.

REFERENCES

REFERENCES

1] V.Ahuja, "Routing and Flow Control in System Architecture", IBM Syst. J., Vol. 18, No. 2, pp. 298-314, 1979.

2] H.A.Anderson and R.Sargent, "The Statistical Evaluation of the Performance of an Experimental APL/360 System", IBM Syst. J., Vol. 18, No. 2, pp. 73-98, 1979.

3] J.Atkins, "Path control, the Transport Network of SNA", IEEE Trans. on Commun., Vol.Com-28, No. 4, pp. 527-538, April 1980.

4] F.Baskett, K.M Chandy, R.R.Muntz and F.G.Palacios, "Open, Closed and Mixed Network of Queues with Different Classes of Costumers", Journal of the ACM., Vol. 22, No. 2, pp. 248-260, April 1975.

5] J.P Buzen, "Computational Algorithms for Closed Queueing Networks with Exponential Servers", Communications of the ACM., Vol. 16, No. 9, pp. 527-531, Sep. 1973.

6] K.M.Chandy, U.Herzog and L.Woo, "Parametric Analysis of Queueing Networks", IBM J. Res. Develop., Vol. 19, pp. 36-42, Jan. 1975.

7] K.M.Chandy, U.Herzog and L.Woo, "Approximate Analysis of General Queueing Networks", IBM. J. Res. Develop., Vol. 19, pp. 43-49, Jan. 1975.

8] K.M.Chandy and C.H.Sauer, "Approximate Methods for the Analysis of Queueing Network Models of Computer Systems", Computing Survey, Vol. 10, No. 3, pp. 263-280, Sep. 1978.

9] K.M.Chandy and C.H.Sauer, "Computational Algorithms for Product form Queueing Networks", Communications of the ACM., Vol. 23, No. 10, pp. 573-583, Oct. 1980.

10] K.M.Chandy and D.Neuse, "Linearizer: A Heuristic Algorithm for Queueing Network Models of Computing Systems", Communications of the ACM., Vol. 25, No. 2, pp. 126-134, Feb. 1982.

11] A.Chatterjee, N.D.Georganas and P.K Verma, "Analysis of a Packet Switched Network with End-to-End Control and Random Routing", Proc. of the 3rd Int. Conf. on Comput. Commun., pp. 488-494, Toronto, Cannada, Aug. 1976.

12] W.Chou and M.Gerla, "A Unified Flow Control and Congestion Model for Packet Networks", Proc. of the 3rd Int. Conf. on Comput. Commun., pp. 475-482, Toronto, Cannada, Aug. 1976.

13] W.W.Chu and M.Y.C.Shen, "A Hierarchical Routing and Flow Control Policy (HRFC) for Packet Switched Networks", IEEE Trans. on Comput., Vol.C-29, No. 11, pp. 971-977, Nov. 1980.

14] W.W.Chu, G.Fayolle and D.G.Hibbits, "An Analysis of a Tandem Queueing System for Flow Control in Computer Networks", IEEE Trans. on Comput., Vol.C-30, pp. 318-323, May 1981.

15] P.J.Courtois, Decomposability: Queueing and Computer System Applications, Academic Press, New York, 1967.

16] D.R.Cox, Renewal Theory, Meuthen, London, 1962.

17] D.W.Davies, "The Control of Congestion in Packet Switching Networks", IEEE Trans. on Commun., Vol.Com-20, pp. 546-550, June 1972.

18] D.W.Davies, D.L.A.Barber, W.L.Price and C.H.Solomonides, Computer Networks and their Protocols, Wiley-Interscience, Chichester, 1979.

19] C.Derman, Finite Markov Decision Processes, Academic Press, New york, 1970.

20] N.D.Georganas, "Modelling and Analysis of Message Switched Computer Communication Networks with Multilevel Flow Control", Computer Networks, Vol. 4, pp. 285-294, 1980.

21] M.Gerla and L.Kleinrock, "Flow Control, A Comparative Survey", IEEE Trans. on Commun., Vol.Com-28, No. 4, pp. 553-574, April 1980.

22] M.Gerla and P.O Nilisson, "Routing and Flow Control Interplay in Computer Networks", Proc. of the 5th Int. Conf. on Comput. Commun., pp. 84-89, Atlanta, USA, 1980.

23] A.Giessler, J.Hanle, A.Koenig and E.Pade, "Free Buffer Allocation - An Investigation by Simulation", Computer Networks, Vol. 2, pp. 191-208, 1978.

24] A.Giessler, A.Jagemann, E.Maser and J.Hanle, "Flow Control Based on Buffer Class", IEEE Trans. on Commun., Vol.Com-29, No. 4, pp. 436-442, April 1981.

25] W.J.Gordon and G.F.Newell, "Closed Queueing Systems with Exponential Servers", Operational Research, Vol. 15, No. 2, pp. 254-265, 1967.

26] J.L.Grange and J.C.Majithia, "Congestion Control for a Packet Switched Network", Computer Communications, Vol. 3, No. 3, pp. 106-116, June 1980.

27] R.L.Grimsdale and F.F.Kuo, Computer Communication Networks, NATO Advanced Study Institute Series, Leyden, The Netherlands, Noordhoff, 1975.

28] R.Howard, Dynamic Programming and Markov Processes, M.I.T Press, Cambridge, Mass, 1960.

29] R.Howard, Dynamic Probabilistic Systems Vol. 2: Semi Markov and Decision Processes, Wiley, New York, 1971.

30] M.I.Irland, "Buffer Management in a Packet Network", IEEE Trans. on Commun., Vol.Com-26, No. 3, pp. 328-337, March 1978.

31] J.R.Jackson, "Two Shop like Queueing System", Management Sci., Vol. 10, pp. 131-142, 1963.

32] J.M.Jaffe, "Bottleneck Flow Control", IEEE Trans. on Commun., Vol.Com-29, No. 7, pp. 954-962, July 1981.

33] R.E.Kahn and W.R.Crowther, "Flow Control in a Resource Sharing Computer Network", IEEE Trans. on Commun., Vol.Com-20, pp. 539-546, June 1972.

34] F.Kamoun and L.Kleinrock, "Analysis of Shared Finite Storage in a Computer Network Node Environment under General Traffic Conditions", IEEE Trans. on Commun., Vol.Com-28, No, 7, pp. 992-1003, July 1980.

35] F.Kamoun, A.Belguith and J.L.Grange, "Congestion Control with a Buffer Management Strategy based on Traffic Priorities", Proc. 5th Int. Conf. on Comput. Commun., pp. 845-850, Atlanta, USA, Oct. 1980.

36] F.Kamoun, "Drop and Throttle Flow Control Policy for Computer Networks", IEEE Trans. on Commun., Vol.Com-29, No.4, pp. 444-452, April 1981.

37] L.Kaufman, B.Gopinath and E.F.Wunderlich, "Analysis of a Packet Network Congestion Control using Sparse Matrix Algorithms", IEEE Trans. on Commun., Vol.Com-29, No. 4, pp. 453-465, April 1981.

38] P.Kermani and L.Kleinrock, "Dynamic Flow Control in A Store-and-Forward Computer Network", IEEE Trans. on Commun., Vol.Com-28, No. 2, pp. 262-271, Feb. 1980.

39] P.Kermani, "Analysis of a Feedback Scheme for Congestion Control in Computer Networks", Proc. of the Int. Conf. on Performance of Data Commun. Systems and their

Application, Paris, Sep. 1981.

40] L.Kleinrock, Communication Nets, McGraw Hill, New York, 1964.

41] L.Kleinrock, Queueing Systems - Volume 1: Theory, Wiley-Interscience, New York, 1975.

42] L.Kleinrock, Queueing Systems -Volume 2: Computer Applications, Wiley-Interscience, New York, 1976.

43] L.Kleinrock, "On Flow Control in Computer Networks", Proc. of the Int. Conf. on Commun., June 1978

44] L.Kleinrock, "Power and Deterministic Rules of Thumb for Probabilistic Problems in Computer Communication", Proc. of the Int. Conf. on Commun., June 1979.

45] L.Kleinrock and P.Kermani, "Static Flow Control in Store-and-Forward Computer Networks", IEEE Trans. on Commun., Vol.Com-28, No. 2, pp. 271-279, Feb. 1980.

46] L.Kleinrock and C.W.Tseng, "Flow Control Based on Limiting Permit Generation Rates", Proc. of the 5th Int. Conf. on Comput. Commun., pp. 785-790, Atlanta, Oct. 1980.

47] H.Kobayashi, "Application of Diffusion Approximation to Queueing Networks: Part 1 and 2", Journal of the ACM., Vol. 21, pp. 316-328 and 459-469, April 1974.

48] H.Kobayashi and A.G.Konheim, "Queueing Models for Computer Communications System Analysis", IEEE Trans. on Commun., Vol.Com-25, No. 1, pp. 2-29, Jan. 1977.

49] H.Kobayashi, Modelling and Analysis: An Introduction to System Performance Evaluation Methodology, Addison-Wesley, Reading, Mass., 1978.

50] A.G.Koneim and M.Reiser, "A Queueing Model with finite Waiting Room and Blocking", Journal of the ACM., Vol. 28, No. 2, pp. 328-341, April 1976.

51] P.J.Kuehn, "Approximate Analysis of General Queueing Networks by Decomposition", IEEE Trans. on Commun., Vol.Com-27, No. 1, pp. 113-118, Jan. 1979.

52] K.B.Kumar, "Optimum End-to-End Flow Control in Networks", IBM Research Report, RC 7949, Yorktown Heights, New York, 1979.

53] S.S.Lam, "Store-and Forward Buffer Requirements in a Packet Switching Network", IEEE Trans. on Commun., Vol.Com-24, No. 4, pp. 394-403, April 1976.

54] S.S.Lam, "Queueing Networks with Population Size Constraints", IBM J. Res. Develop., Vol. 21, No. 4, pp. 370-378, July 1977.

55] S.S.Lam and M.Reiser, "Congestion Control of Store-and-Forward Networks by Input Buffer Limits - An Analysis", IEEE Trans. on Commun., Vol.Com-27, No. 1, pp. 127-133, Jan. 1979.

56] S.S.Lam and Y.L.Lien, "An Experimental Study of the Congestion Control of Packet Communication Networks", Proc. of the 5th Int. Conf. on Comput. Commun., pp. 791-796, Atlanta, Oct. 1980.

57] S.S.Lam and Y.L.Lien, "Modelling and Analysis of Flow Controlled Packet Switching Networks", Proc. of the 7th Data Commun. Symp., pp. 98-106, Mexico City, Mexico, Oct. 1981.

58] J.D.C.Little, "A Proof of the Queueing Formuls: L = W, Operations Research, Vol. 9, pp. 383-387, 1961.

59] R.A.Marie, "An Approximate Analytical Method of General Queueing Networks", IEEE Trans. on Software Engineering, Vol.SE-5, No. 5, pp. 530-538, Sep. 1979.

60] J.Matsumoto and H.Mori, "Flow Control in Packet Switched Networks by Gradual Restriction of Virtual Calls", IEEE Trans. on Commun., Vol.Com-29, No.4, pp. 466-473, April 1981.

61] P.M.Merlin and P.J.Schweitzer, "Deadlock Avoidance in Store-and-Forward Networks - 1: Store-and-Forward Deadlocks", IEEE Trans. on Commun., Vol.Com-28, No. 3, pp. 345-354, March 1980.

62] M.C.Pennotti and Schwartz, "Congestion Control in Store-and-Forward Tandem Links", IEEE Trans. on Commun., Vol.Com-23, No. 12, pp. 1434-1443, Dec. 1975.

63] L.Pouzin, "Methods, Tools and Observation on Flow Control in Packet Switched Data Networks", IEEE Trans. on Commun., Vol.Com-29, No. 4, pp. 413-236, April 1981.

64] W.L.Price, "Data Networks Simulation Experiment at the National Physical Laboratory 1968-1976", Computer Networks, Vol. 1, pp. 199-210, 1977.

65] G.Pujolle, "Analysis of Flow Controls in Switched Data Networks by a Unified Model", Proc. of the 4th Int. Conf. on Comput Commun., pp. 123-128, Kyoto, Japan, Sept. 1978.

66] E.Raubold and J.Haenle, "A Method of Deadlock Free Resource Allocation and Flow Control in Packet Networks", Proc. of the 3rd Int. Conf. on Comput. Commun.,pp. 483-487, Toronto, Cannada, Aug. 1976.

67] M.Reiser and H.Kobayashi, "Queueing Networks with Multiple Closed Chains: Theory and Computational Algorithms", IBM J. Res. Develop., Vol. 19, pp. 285-294, 1975.

68] M.Resier, "Numrical Methods in Separable Queueing Networks", Studies in Management Sciences - Algorithmic Methods, Vol. 7, pp. 113-142, North Holland Publishing Company, 1977.

69] M.Reiser and C.H.Sauer, "Queueing Network Models: Methods of Solution and Their Program Implementation", Current Trends in Program Methodology - Volume III; Software Modelling and its Impact on Performance, pp. 115-167, Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

70] M.Reiser, "A Queueing Network Analysis of Computer Communication Networks with Window Flow Control", IEEE Trans. on Commun., Vol.Com-27, No. 8, pp. 1199-1209, Aug. 1979.

71] M.Reiser and S.S.Lavenberg, "Mean Value Analysis of Closed Multi-chain Queueing Networks", Journal of the ACM., Vol. 28, No. 2, pp. 313-322, April 1980.

72] M.Reiser, "Mean Value Analysis and Convolution Methods for Queue Dependent Servers in Closed Queueing Networks", Performance Evaluation, Vol. 1, pp. 7-18, North Holland Publishing Company, 1981.

73] M.Reiser,"Admission Delay on Virtual Routes with Window Flow Control", Proc. of the Int. Conf. on Performance of Data Commun. Systems and their Applications, Paris, France, Sep. 1981.

74] M.Reiser, "Performance Evaluation of Data Communication Systems", Proc. of the IEEE, Vol. 70, No. 2, pp. 171-196, Feb. 1982.

75] M.R.Rich and M.Schwartz, "Buffer Sharing in Computer Communication Network Nodes", IEEE Trans. on Commun., Vol.Com-25, No. 9, pp. 958-970, Sep. 1977.

76] S.M.Ross, Applied Probability Models with Optimisation Applications, Holden Day, San Francisco, 1970.

77] H.Rudin, "Flow Control: Session Chairman's Remarks", Proc. of the 3rd Int. Conf. on Comput. Commun., pp. 463-466, Toronto, Cannada, Aug. 1976.

78] H.Rudin and H.Mueller, "Dynamic Routing and Flow Control", IEEE Trans. on Commun., Vol.Com-28, No. 7, pp. 1030-1039, July 1980.

79] S.Saad and M.Schwartz, "Input Buffer Limiting Mechanism for Congestion Control", Proc. Int. Comput. Conf., Seattle, 1980.

80] C.H Sauer and K.M.Chandy, Computer System Performance Modelling, Prentice Hall, Englewood Cliffs, New Jersey, 1981.

81] M.Schwartz, Computer Communication Network Design and Analysis, Prentice-Hall, Englewood Cliffs, New Jersey, 1977.

82] M.Schwartz and S.Saad, "Analysis of Congestion Control Techniques in Computer Communication Networks", Proc. Int. Symp. Comput. Networks, Versailles, France, Feb. 1979.

83] M.Schwartz, "Routing and Flow Control in Data Networks", Proc. NATO Advanced Study Inst.; New Concepts in Multi-user Commun., pp. 273-301, Norwich, England, 1980.

84] M.Schwartz and T.E.Stern, "Routing Techniques used in Computer Communication Networks", IEEE Trans. on Commun., Vol.Com-28, No. 4, pp. 539-552, April 1980.

85] M.Schwartz, "Performance Analysis of the SNA Virtual Route Pacing Control", IEEE Trans. on Commun., Vol.Com-30, No. 1, pp. 172-184, Jan. 1982.

86] P.J.Schweitzer and S.S.Lam, "Buffer Overflow in a Store-and-Forward Network Node", IBM J. Res. Develop., Vol. 20, pp. 542, 1976.

87] P.Schweitzer, "Approximate Analysis of Multi-class Closed Network of Queues", Presented at the Int. Conf. Stochastic Control and Optimization, Amsterdam, 1979.

88] A.P.Tanenbaum, Computer Networks, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

89] F.A.Tobagi, M.Gerla, R.W.Peebles and E.G.Manning, "Modelling and Measurement Techniques in Packet Communication Networks", Proc. of the IEEE, Vol. 66, No. 11, pp. 1423-1447, Nov. 1978.

90] J.W.Wong and M.S.Unsoy, "Analysis of Flow Control in Switched Data Networks", Proc. Int. Fed. Inf. Processing Soc. Conf., pp. 315-320, Aug. 1977.

91] J.W.Wong, "Distribution of End-to-End Delay in Message Switched Networks", Computer Networks, Vol. 2, pp. 44-49, Feb. 1978.

92] J.W.Wong, "Queueing Network Modelling of computer Communication Networks", Computing Survey, Vol. 10, No. 3, pp. 343-381, Sep. 1978.