NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

CONGESTION CONTROL USING DYNAMIC ROUTING AND FLOW CONTROL IN STORE-AND-FORWARD COMPUTER NETWORKS

by

Wahida Chouikha

A thesis submitted to
the school of Graduate Studies in the fulfillment of
the thesis requirement for the degree of Master of Science

in

Systems Science

ISBN 0-315-68033-4

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Abstract

In this thesis, the problem of congestion control in computer communication networks is studied. The method employed consists of a combination of a dynamic routing policy that minimizes the delay, and buffer management schemes to control the flow of data. These schemes include (1) Complete Sharing, (2) Square Root Sharing, and (3) Sharing with Minimum Allocation.

A dynamic model for a computer queueing network with finite buffer size, stochastic input and random channel errors, is used. Simulations using linear programming have been carried out on a simple 3-node network, under balanced and unbalanced traffic conditions. The important performance measures such as average delay, average throughput, the number of packets lost due to limitation in the buffer size, the total queues and the total external arrivals, are obtained from the simulation and compared. The relative merits and drawbacks of each approach are discussed.

Acknowledgement

I wish to express my gratitude to Dr. N. U. Ahmed for his supervision and encouragement. Special appreciation is expressed to Dr. J. M. Thizy for his help in the optimization process, to Dr. D. Lane and Dr. J. Sidney for helping me review this thesis.

I would like also to express my sincere thanks to the Tunisian Government for its scholarship program

Contents

List of Tables

List of Figures

Chapter 1

Introduction

1.1 Congestion Control

A great deal of attention has been paid in the communication literature to congestion control, since it is one of the most important objectives in the design of message-switched or packet-switched computer communication networks. Because of the random nature of traffic flow and the limitation of resources in a store-and-forward computer communication network, congested spots can appear within the network, which may eventually lead to performance degradation.

Networks cannot afford to accept all the traffic that is offered without control, hence there must be rules which govern the acceptance of traffic flow from outside and coordinate the flow inside the network.

The major procedures of congestion control fall into two main categories . The first is flow control, which is the set of mechanisms that regulate the traffic entering the network at the various nodes, to maintain the data flow within

limits compatible with the amount of available resources, and prevent build-up of congestion at one or more parts of the network. The second category is routing, which involves the selection of the best paths for routing the traffic that is already accepted into the network from source to destination, so that some performance criterion is optimized.

Based on how routing procedures vary with the traffic conditions, they may be classified as static routing, quasi-static routing, and dynamic routing procedures. In static routing procedures, fractions of the traffic at every link are kept constant over time. These fractions have been decided upon before the network starts operating and will be independent of normal traffic variations. In quasi-static routing procedures, changes of routes will be allowed only at given intervals of time, or whenever extreme situations occur. On the other hand, with dynamic routing strategies, messages or packets are routed according to the instantaneous states of the queues at the nodes of the network. The path on which a packet or a message is routed from source to destination varies according to the network traffic conditions.

In store-and-forward communication networks, packets in transit use resources, such as buffers, bandwidth and processor time. The two resources that are most commonly wasted are communication capacity (channels) and storage capacity (buffers).

An important component of flow control in computer communication networks is the management of buffers at the nodes, since the number of buffers

available at each node is limited. An efficient buffer management scheme cannot only control congestion effectively, and hence ensure satisfactory throughput, but can also provide additional performance features, such as fairness in sharing the resources among network users, deadlock prevention, and minimization of retransmission of lost packets. Furthermore, buffer management prevents underutilization and waste of storage resources.

This thesis focuses on a particular means of congestion control in communication networks, namely dynamic routing in conjunction with flow control.

1.2 Literature Review

1.2.1 Buffer Management

The buffer management problem has received considerable attention and has been extensively studied, giving rise to a variety of schemes available in the literature [12],[14],[16],[22]. At one extreme, there is the complete partitioning (CP) scheme, which permanently allocates a fixed amount of the buffer for each queue. In the opposite extreme, the complete sharing (CS) scheme permits an unrestricted sharing of the total buffer among all types of packets at the node. In between these two extremes, other schemes have been developed. Some notable ones are sharing with maximum queue lengths(SMXQ), sharing with minimum allocation(SMA), and sharing with a maximum queue and minimum allocation(SMQMA) schemes. A very detailed comparative performance analysis of these schemes has been con-

ducted by Kamoun and Kleinrock [14], under steady state conditions, where they showed that sharing with some restrictions on the contention for space is more advantageous than non-sharing(complete partitioning), especially when little storage is available.

In their comparative survey, Kleinrock and Gerla [16] stated that any of the restricted buffer management policies mentioned above will provid at least a minimum protection to avoid throughput degradation, unfairness and deadlocks.

D. Tipper and M. K. Sundareshan [37] addressed the problem of developing an adaptive buffer management policy. Their approach consists of presenting a precise formulation of the optimal buffer allocation problem in a mathematical optimization framework, and developing an adaptive policy for buffer management when the incoming load at the node is dynamically varying. They also demonstrated the superiority of their policy over other existing ones such as the CP scheme.

In the optimal restricted buffer sharing scheme proposed by Irland [12] a limit (M) is imposed on the queues in the buffer such that the number of packets for the jth output process can be at most equal to M. The problem addressed in this scheme is to select optimally a new buffer limit for each level of traffic (dynamic SMXQ). An approximation of the optimal scheme is offered by the "Square-Root Sharing" scheme(SQRT), a load invariant scheme a with fixed buffer limit $M = B/\sqrt{N}$, where B=the buffer size and N= number of output channels. The SQRT scheme is simpler to implement since it does not depend on the traffic

load. Moreover it has been shown [12] to be practically as efficient as the optimal policy for many cases.

1.2.2 Routing

There is little question that a routing strategy, which can adapt to traffic flow changes in the network, should increase the traffic carrying capacity of the network, thereby reducing the chance of congestion. This issue was considered in the following studies.

Chou, Bragg, and Nilson [6] used simulation specifically to address the question of the worth of adaptive routing. They concluded that some well-chosen adaptive strategies perform better than deterministic or static strategies, for a network faced with unbalanced and chaotic traffic patterns.

In 1977, an analytic continous time model for problems of dynamic routing in data communication networks was developped by Segall [33]. Assuming deterministic input and infinite buffers that can accomodate all external and internal traffic flow, Segall and Moss [34] applied optimal control theory to find a centralized routing policy.

While the research reported in [12,14,22,37] has concentrated on the study of flow control,by assuming fixed routing strategies, the work reported in [6,33,34] was restricted to finding dynamic routing strategies with no attention paid to flow control. On the other hand, none of these studies considered dynamic routing and flow control together. However, it is well recognized that traffic loads vary

significantly in computer communication networks, and hence a certain degree of adaptivity to changes in the incoming traffic is desired of routing and flow control to improve the network performance under such dynamically varying loads.

The above statement makes it important to find models that combine dynamic routing together with flow control, and hence handle transient and dynamic situations as well as steady state situations.

The first attempt in this direction is due to Gallager and Golestaani [9], who developed a combined objective function that could be optimized by simultaneously selecting the link flows(routing) and the admitted traffic inputs(flow control).

The new approach presented by Muralidhar and Sundareshan [26] yields a two-level adaptive strategy for combined routing and flow control in packet switched networks of the datagram type. The overall control effort consists of two parts: (1) a distributed computation of the routing and flow control parameters at the lower level of the network nodes which are adaptively updated with an objective of minimizing the end-to-end delay, and (2) a computation on a slower time-scale by a supervisor (network control center) at a higher level of a set of combined routing and flow control parameters, which are transmitted to the nodes at time intervals larger than the nodal updating period. The computation and updating of these parameters by the supervisor are guided by the objectives of throughput improvement and queue management in the nodal buffers.

An interesting attempt has been made by N.U. Ahmed, T.E. Dabbous and Y.W. Lee [1] to find a dynamic routing policy that minimizes the delay and max-

imizes the throughput, in networks with finite buffer size, stochastic input and random channel errors.

The dynamic routing procedure considered is as follows. At every short interval of time [t,t+Δ] the states of the queues and the expected number of external arrivals are transferred from every node to the control center where the control parameters (the number of packet to be transmitted from each type at every node) will be computed based on this information, with an objective of minimizing the total size of the queue in the network with respect to the control variables. Results are sent back to each node so that the transmission rates at every channel can be updated.

Among the previous models explained in the literature review we are more interested by the staochastic dynamic approach presented in [1]. A more detailed description of Ahmed's model [1] is given below.

1.2.3 Ahmed's model [1]

This is a discrete time dynamic model, developed based on Segall's model [33]. However, in order to make it more realistic, Ahmed et al. have considered stochastic input, finite buffers and channel errors.

Consider a data communication network consisting of the node set N= $\{1, 2, \ldots . N\}$. Let us denote $L = \{(i,k)/i,k \in N\}$ as the collection of all channels in the network such that channel (i,k) has a direction from node i to node k and a capacity in units of packets/unit-time denoted by C_{ik}. Every node $i \in$ N has a set of receiving

nodes denoted by $E(i) = \{k \in N/(i,k) \in L\}$ and a set of sending nodes denoted by $I(i) = \{l \in N/(l,i) \in L\}$. A buffer of size B_i^j is provided for every type (i.e. destination) of packet at each node in the network, where B_i^j denotes the capacity of the buffer of type-j packets at node i. Let the number of waiting packets of type-j at node i be denoted by Q_i^j(the queues are associated with nodes rather than with channels), and the number of external packets of type-j arriving at node i during the time interval Γ_t be denoted by $A_i^j(\Gamma_t)$, which is assumed to obey a generalized Poisson distribution. The number of packets of type-j at node i to be sent through the channel (i,k) during the time interval Γ_t is denoted by $U_{ik}^j(\Gamma_t)$, which is considered to be the control variable over Γ_t.

Queue and state dynamics

The basic equation which describes the variation of the queue length during the interval of time [t,t+dt] is given as the difference between the incoming and the outgoing flow rates, i.e.,

$$Q_i^j(dt) = R_i^j(dt) - O_i^j(dt)$$

At every node i, the incoming flow rate consists of the internal input coming from the neighboring nodes $l \in I(i)$, and the external input coming from outside the network. Since buffers are considered to be of finite size, the available storage space cannot accomodate all the incoming packets. An arriving packet is lost if no buffer space is available. Further, due to channel errors some of the internal input

packets may get lost (these packets have to be retransmitted again).We denote by ξ^j_{ik} the number of type-j packets at node i that failed to be transmitted through channel (i,k) due to channel errors. After removing the defective packets ($\xi^j_{i.}$) from the internal input, the rate of incoming flow of type-j at node i is given by

$$R^j_i(dt) = [A^j_i(dt) + \sum_{l \in I(i)} \{U^j_{li}(dt) - \xi^j_{li}(dt)\}] \wedge [B^j_i - Q^j_i(t)], \qquad (1.1)$$

where $a \wedge b \equiv min\{a,b\}$, $B^j_i - Q^j_i(t)$ denotes available buffer space.

The outgoing flow consists of all the packets that are transmitted from node i to all the destination nodes $k \in E(i)$, after removing the defective packets due to channel errors (ξ^j_{ik}). Thus the rate of outgoing flow of type-j packets at node i is given by

$$O^j_i(dt) = \sum_{k \in E(i)} \{U^j_{ik}(dt) - \xi^j_{ik}(dt)\}$$

Hence it follows from the above that the variation of the queue length during the interval [t,t+dt] is given by

$$Q^j_i(dt) \equiv Q^j_i(t+dt) - Q^j_i(t) = [A^j_i(dt) + \sum_{l \in I(i)} \{U^j_{li}(dt) - \xi^j_{li}(dt)\}] \wedge [B^j_i - Q^j_i(t)]$$
$$- \sum_{k \in E(i)} \{U^j_{ik}(dt) - \xi^j_{ik}(dt)\}. \qquad (1.2)$$

Over a short interval of time $\Gamma_t = [t, t+\Delta]$, the dynamic equation can be approximated as the sum of the queue state at time t ($Q^j_i(t)$) and the queue variation during this interval, and is expressed as

$$Q_i^j(t+\Delta) = Q_i^j(t) + [A_i^j(\Gamma_t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\}] \wedge [B_i^j - Q_i^j(t)]$$

$$- \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\}. \qquad (1.3)$$

where $t > 0$, and Δ is a suitable positive number.

Objective Function

The objective is to minimize the delay in the network. Assuming that the processing time and the propagation delay are negligible, the delay is approximated by the waiting time which is given by the weighted summation of the expected value of the queue states at time t+Δ, given the queue states at time t, and the expected external arrivals. Therefore, the objective function is given by

$$J(Q) = \sum_{ij} \alpha_i^j E\{Q_i^j(t+\Delta) \mid Q_i^j(t)\} \qquad (1.4)$$

Where α_i^j are the weighting factors. In order to reduce the losses and increase the throughput under congested conditions, the weighting factors α_i^j are defined as follows

$$\alpha_i^j(\Gamma_t) = \begin{cases} 1 + Z_i^j(\Gamma_t) & \text{if } Z_i^j(\Gamma_t) > 0 \\ 1 & \text{otherwise} \end{cases}$$

where

$$Z_i^j(\Gamma_t) = \frac{Q_i^j(t) + E\{A_i^j(\Gamma_t)\} - B_i^j}{B_i^j}$$

Constraints

The objective function is to be minimized given the following constraints:

$$Q_i^j(t) \geq 0 \qquad (1.5)$$

$$U_{ik}^{j}(\Gamma_t) \geq 0 \tag{1.6}$$

$$\sum_{j} U_{ik}^{j}(\Gamma_t) \leq C_{ik}. \qquad (i,k) \in L, i \neq j \tag{1.7}$$

The queues and the control variables have to be non-negative quantities. also the total amount of traffic cannot exceed the channel capacity. Besides, a buffer constraint is added to avoid the situation in which the size of the incoming load exceeds the difference between the available buffer space and the estimated number of external arrivals, thus reducing the amount of internal retransmission. This constraint is defined as follows :

$$\begin{cases} \sum_{l \in I(i)} U_{il}^{j}(\Gamma_t) \leq B_i^j - Q_i^j(t) - E\{A_i^j(\Gamma_t)\} & \text{if } B_i^j - Q_i^j(t) \geq E\{A_i^j(\Gamma_t)\} \\ \sum_{l \in I(i)} U_{il}^{j}(\Gamma_t) = 0 & \text{otherwise} \end{cases} \tag{1.8}$$

This model is interesting because of its requirement for the use of mathematical programming which provides a simple optimization scheme that can easily be implemented, and most importantly the more realistic assumption of finite buffers. However, in addition to an efficient dynamic routing policy, an efficient buffer management scheme that controls the flow entering the network and hence minimizing the losses is necessary. In this thesis, we propose to incorporate three buffer management schemes, namely (1) Complete Sharing scheme, (2) Square Root Sharing scheme and (3) Sharing with minimum allocation scheme , into the dynamic routing model [1] presented above. In model [1], at every node a buffer is associated with each type of packet, this buffer cannot be used by other types so a non-sharing (complete partitioning) scheme is assumed. However the schemes

mentioned above insure the sharing of a single buffer among all types of packets in order to reduce the wastage of storage space and hence make a more efficient use of the buffers.

1.3 Organization of the Thesis

In Chapter 2, we present the modified models resulting from the incorporation of the three buffer management schemes mentioned above into the dynamic routing model [1]. For each of these models we obtain the corresponding queue's dynamic equation, the constraints and the weighting factors, since those quantities vary with the buffer management scheme. The objective function presented in [1] will be maintained in all the models.

In Chapter 3, we describe the simulation of the modified models, model [1] and a fixed routing scheme based on minimum hop [32], to be described later, in a 3-node network. The methodology followed and the simulation flow chart are presented at the end of this chapter.

In Chapter 4, a numerical comparison is made between the three different buffer management models considered in this thesis, model [1] and the fixed routing scheme mentioned above, based on the simulation results obtained under balanced and unbalanced traffic conditions.

Finally, in Chapter 5, the conclusions of this work and some recommendations for further research are presented.

Chapter 2

The Three Modified Buffer Management Models

The model in [1] considered buffers with finite sizes. Non-sharing buffer management was assumed since at every node in the network a buffer is provided for each type of packet such that an arriving type-j packet may enter the system only if the buffer dedicated to the packets of type-j is not filled. A major difficulty with this scheme is the underutilization and the waste of buffer space. In fact, the storage space allocated to an almost empty queue is wasted since it is not used by that queue and cannot be used by the others.

An alternative to this scheme is to consider a complete sharing scheme (CS) in which a single buffer of size B_i (i means node i) is paired with each node i where all the packets are stored at the same location and such that an arriving packet is accepted if any stotrage space is available, independent of its destination.

Such an approach could succeed in maximizing the utilization of the storage

space, however it may lead to undesirable behavior. Since the buffer is shared with no restrictions, one or more of the queues that have higher input rates, especially under a highly unbalanced load, may monopolize the buffer and prevent the others from getting in.

The above considerations suggest that, in order to reduce the impact of such circumstances, some restrictions on the contention for space must be imposed. Two solutions are suggested:

- Imposing some limits on the queues. These limits can be fixed and set to the same value for all the queues, or can change dynamically in time and from queue to queue based on the traffic fluctuation, as in optimal policies [12],[37]. Although the optimal choice of the limits approach is in theory superior to the fixed one, it may be difficult to implement in practice. Instead one would set the limits to a fixed value which yields reasonable results. This approach is incorporated in the third scheme, Square-Root Sharing (SQRT) which imposes a limit on the maximum amount of storage space to be allocated to any queue, and this limit is equal to $B_i/\sqrt{N-1}$, where B_i is the size of the single buffer at node i and $N-1$ represents the number of destinations. The SQRT scheme has been shown to be attractive as it provides a good approximation to the optimal policy, but is easier to implement in reality [12].

- Sharing with minimum allocation (SMA), where a minimum portion of the

buffer is always reserved for each queue and the remaining space is shared among all the queues.

In what follows, we present the modified models resulting from the incorporation of the three buffer management schemes into the model introduced in [1].

Notations

Below is a summery of the notations used, they are similar to those in [1]:

- N: set of all the nodes $\{1, 2, ... N\}$ in the network.
- (i,k): direct channel connecting node i to node k.
- L: collection of all channels (i,k), such that i,k $\in$ N and there is a direct link between i and k.
- $E(i)$: set of all receiving nodes from i.
- $I(i)$: set of all sending nodes to i.
- C_{ik} : capacity of channel (i, k) in units of packets/unit-time.
- B_i^j : buffer size at node i for packets of type-j
- B_i : size of the single buffer at node i.
- Q_i^j : number of waiting packets of type-j at node i.
- Q_i : number of all waiting packets at node i.
- $U_{ik}^j(\Gamma_t)$:number of packets of type-j to be sent through channel (i, k), during the interval of time Γ_t.
- $A_i^j(\Gamma_t)$: number of external arrivals of type-j at node i, during the time interval Γ_t

- $\xi_{ik}^{j}(\Gamma_t)$: number of packets of type-j at node i failed to be transmitted through channel (i,k) during the time interval Γ_t

- ρ_{ik} : error rate of channel (i,k).

- α_i^j: portion of the buffer reserved for packets of type-j at node i (in SMA model)

2.1 Complete Sharing Model (CS)

The dynamic routing procedure used in this case is the same as [1]. However, the buffer spaces B_i^j are combined into a common pool, whose size is B_i, where all the packets destined to go to other nodes are stored together with no restrictions. Similarly to [1], the global queue is given by the summation of the number of all the packets waiting in the queue at time t and that of all the accepted incoming packets from which we subtract the number of all the transmitted packets during the interval [t,t+Δ], which is written as :

$$Q_i(t+\Delta) =$$

$$Q_i(t) + \left[\sum_j \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + \sum_j A_i^j(\Gamma_t)\right] \wedge [B_i - Q_i(t)]$$

$$- \sum_j \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\}, \qquad (2.1)$$

where

$$Q_i(t) = \sum_j Q_i^j(t)$$

and

$$a \wedge b \equiv min\{a, b\}$$

Note that with CS, an arriving packet can be accepted, if any storage space is available in the buffer. Thus the number of type-j packets at node i (Q_i^j), at time t+Δ is given by

$$Q_i^j(t+\Delta) =$$

$$Q_i^j(t) + \left[\sum_{l\in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t)\right] \wedge \left[B_i - \sum_j Q_i^j(t)\right]$$

$$- \sum_{k\in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\}. \tag{2.2}$$

The external arrivals are assumed to obey a generalized Poisson distribution with mean

$$E\{A_i^j(\Gamma_t)\} = \int_{\Gamma_t} \lambda_i^j(t)dt$$

, where $\lambda_i^j(t)$ denotes the instantaneous mean arrival rate at node i with destination j, and ξ_{ik}^j is the number of lost packets due to channel errors, which is considered to be a doubly stochastic Poisson process satisfying

$$P\{\xi_{ik}^j(\Gamma_t) = m \mid U_{ik}^j(\Gamma_t)\} = \frac{\exp(-\rho_{ik}^j(\Gamma_t))(\rho_{ik}^j(\Gamma_t))^m}{m!}$$

, where $\rho_{ik}^j(\Gamma_t) \equiv E\{\xi_{ik}^j(\Gamma_t) \mid U_{ik}^j(\Gamma_t)\}$ is the expected number of lost packets in the time interval Γ_t.

Objective Function

The goal is to minimize the delay approximated by the weighted summation of the expected value of the queue lengths. Similarly to [1], the objective function is given by equation (1.4), and is repeated below:

$$J(Q) = \sum_{ij} \alpha_i^j E\{Q_i^j(t+\Delta) \mid Q_i^j(t)\}$$

. Taking the conditional expectation of (2.2) and using the following expression :

$$\{a \wedge b\} = (1/2)\{a + b\} - (1/2)|a - b|$$

we have :

$$E\{Q_i^j(t+\Delta) \mid Q_i^j(t)\} = Q_i^j(t) +$$

$$1/2\left[E\{A_i^j(\Gamma_t)\} + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - E\{\xi_{li}^j(\Gamma_t)|U_{li}^j(\Gamma_t)\} + B_i - \sum_j Q_i^j(t)\right]$$

$$-1/2E\left[\left|A_i^j(\Gamma_t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} - B_i + \sum_j Q_i^j(t)\right| |Q_i^j(t)\right]$$

$$- \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - E\{\xi_{ik}^j(\Gamma_t)|U_{ik}^j(\Gamma_t)\}\} \qquad (2.3)$$

with $|a-b|$=absolute value of (a-b)

The expected number of packets of type-j that are discarded due to channel errors during the time interval Γ_t is assumed to be proportional to the number of packets transmitted through channel (i,k) during that time interval and given by:

$$E\{\xi_{ik}^j(\Gamma_t) \mid U_{ik}^j(\Gamma_t)\} = \tilde{\rho_{ik}} U_{ik}^j(\Gamma_t)$$

where $\tilde{\rho_{ik}}$ is the error rate of the channel (i,k), which is assumed to be constant. with $0 < \tilde{\rho_{ik}} < 1$.

Thus we can write equation (2.3) as

$$E\{Q_i^j(t+\Delta) \mid Q_i^j(t)\} =$$

$$Q_i^j(t) + 1/2\left[\sum_{l\in I(i)}\{U_{li}^j(\Gamma_t)(1-\tilde{\rho}_{li})\} + E\{A_i^j(\Gamma_t)\} + B_i - \sum_j Q_i^j(t)\right]$$

$$-1/2E\left[\left|\sum_{l\in I(i)}\{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - B_i + \sum_j Q_i^j(t)\right| \mid Q_i^j(t)\right]$$

$$-\sum_{k\in E(i)}\{U_{ik}^j(\Gamma_t)(1-\tilde{\rho}_{ik})\} \qquad (2.4)$$

To prevent congestion, the weighting factors are modified to give more weight to the queues which are longer or expecting a higher external load:

$$\alpha_i^j(\Gamma_t) = 1 + \frac{Q_i^j(t) + E\{A_i^j(\Gamma_t)\}}{B_i}$$

The objective function (1.4) is to be minimized such that the control variables (U_{ik}^j) and the queue states are positive, and the maximum capacity of the links is not exceeded, i.e.

$$U_{ik}^j(\Gamma_t) \geq 0 \qquad (2.5)$$

$$Q_i^j(t) \geq 0 \quad or \quad Q_i^j(t+\Delta) \geq 0 \qquad (2.6)$$

$$\sum_j U_{ik}^j(\Gamma_t) \leq C_{ik}, \qquad (i,k) \in L, i \neq j \qquad (2.7)$$

Furthermore, the number of transmitted packets from node i to the neighboring nodes, during the time interval Γ_t, should not exceed the number of packets existing in the queue during Γ_t. This is implied by the constraint (2.6).

In agreement with the CS scheme, the buffer constraint is modified such that the total incoming flow to node i, in addition to the total number of packets

remaining in the buffer, at time t in the same node , does not exceed the buffer size B_i. This constraint is given by

$$\sum_j \left(Q_i^j(t) + E\{A_i^j(\Gamma_t)\} + \sum_{l \in I(i)} U_{li}^j(\Gamma_t) \right) \leq B_i$$

which is equivalent to

$$\sum_j \left(\sum_{l \in I(i)} U_{li}^j(\Gamma_t) \right) \begin{cases} \leq S_i(\Gamma_t) & \text{if } S_i(\Gamma_t) > 0 \\ = 0 & \text{otherwise} \end{cases} \tag{2.8}$$

with

$$S_i(\Gamma_t) = B_i - \sum_j \left(Q_i^j(t) + E\{A_i^j(\Gamma_t)\} \right)$$

Let us now derive the expression for the number of lost packets. Recall that with CS scheme, a packet is lost if, upon arrival, the entire space in the buffer is full. Thus, the number of packets lost at node i, at time t+Δ is given by

$$Q_{lost_i}(t+\Delta) =$$

$$\max \left[0, \sum_j \left(Q_i^j(t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\} \right) - B_i \right]$$

In order to find the number of packets of each type that are discarded, we have to determine the destination of every packet lost, but the destinations of the lost packets are hard to discern, since they arrive in groups. To overcome this problem, those lost packets are assumed to have a multivariate hypergeometric distribution [21], i.e.

$$P(x_i^1, ..., x_i^{i-1}, x_i^{i+1}, ..., x_i^N | Q_{lost_i}(t+\Delta)) =$$

$$\frac{\binom{A_i^1(\Gamma_t)}{x_i^1} \cdots \binom{A_i^{i-1}(\Gamma_t)}{x_i^{i-1}} \binom{A_i^{i+1}(\Gamma_t)}{x_i^{i+1}} \cdots \binom{A_i^N(\Gamma_t)}{x_i^N}}{\binom{\sum_{i \neq j} A_i^j(\Gamma_t)}{Q_{lost_i}(t+\Delta)}} \tag{2.9}$$

where x_i^j, represents the number of packets discarded at node i destined to node j.

$$Q_{lost_i}(t+\Delta) = \sum_j x_i^j \quad \text{and} \quad x_i^j \leq A_i^j(\Gamma_t)$$

Using the buffer constraint (2.8) the control parameter ($U_{ik}^j(\Gamma_t)$) will be determined in a way that makes the left side of the "$\wedge$" in equation (2.2) always valid. Consequently the objective function (1.4) becomes linear in control. Thus the minimization of this objective function subject to the constraints (2.5), (2.6), (2.7) and (2.8), with respect to the control variables (U_{ik}^j), can be carried out using linear programming techniques.

2.2 Square Root Sharing Model (SQRT)

Like CS, the SQRT allows the sharing of the buffer, with a further constraint imposed on the amount of the buffer to be allocated to each type of packets, in fact the maximum storage space that type-j packets can occupy is equal to the entire buffer space (B_i) divided by the square root of the number of types of packets in the buffer. Consequently, the queue's dynamic equation (2.1) remains the same. However, the length of Q_i^j at time t+Δ changes accordingly. Further, in order to derive the latter quantity we should note that an arriving type-j packet to node i is accepted in the buffer only if there is enough buffer space and the number of packets of type-j at node i is less than the limit given by $\frac{B_i}{\sqrt{N-1}}$. Therefore, the length of Q_i^j at time t+Δ is given by

$$Q_i^j(t+\Delta) =$$

$$Q_i^j(t) + \left[\sum_{l\in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t)\right] \wedge \left[B_i - \sum_j Q_i^j(t)\right] \wedge \left[\frac{B_i}{\sqrt{N-1}} - Q_i^j(t)\right] - \sum_{k\in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\} \quad (2.10)$$

Similarly to CS, the SQRT uses the objective function (1.4) and the constraints (2.5), (2.6) and (2.7). The constraint (2.8) is also considered in this case, because of the sharing rule. The constraint imposed on the maximum amount of the buffer to be used by type-j packets, is given by

$$Q_i^j(t) + E\{A_i^j(\Gamma_t)\} + \sum_{l\in I(i)} U_{li}^j(\Gamma_t) \le \frac{B_i}{\sqrt{N-1}}$$

which is equivalent to

$$\sum_{l \in I(i)} U_{li}^j(\Gamma_t) \begin{cases} \leq T_i^j(\Gamma_t) & \text{if } T_i^j(\Gamma_t) > 0 \\ = 0 & \text{otherwise} \end{cases} \tag{2.11}$$

with

$$T_i^j(\Gamma_t) = \frac{B_i}{\sqrt{N-1}} - Q_i^j(t) - E\{A_i^j(\Gamma_t)\}$$

Similarly to the CS, the weighting factors are given by:

$$\alpha_i^j(\Gamma_t) = 1 + \frac{Q_i^j(t) + E\{A_i^j(\Gamma_t)\}}{\frac{B_i}{\sqrt{N-1}}}$$

The evaluation of the number of lost packets is much more complicated here, because of the added constraint on Q_i^j. Recall that an arriving packet is dropped when the entire buffer is full, or when the number of type-j packets in the buffer is equal to $\frac{B_i}{\sqrt{N-1}}$. Therefore the number of lost packets at node i is given by

$$Q_{lost_i}(t+\Delta) = \sum_j QL_i^j(t+\Delta) + QL_i(t+\Delta) \tag{2.12}$$

where

$$QL_i^j(t+\Delta) =$$

$$\max\left[0, Q_i^j(t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\} - \frac{B_i}{\sqrt{N-1}}\right]$$

and

$$QL_i(t+\Delta) = \max[0, \eta_i(t+\Delta) - B_i]$$

with

$$\eta_i(t+\Delta) =$$

$$\sum_j \left(Q_i^j(t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \varsigma_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \varsigma_{ik}^j(\Gamma_t)\} - QL_i^j(t+\Delta) \right)$$

A simple interpretation of equation (2.12) is that, QL_i corresponds to the losses when the entire buffer is used, and QL_i^j represents the losses due to Q_i^j being equal to $(B_i/\sqrt{N-1})$.

Similarly to the previous model, since the destinations of the lost packets represented by QL_i are not known, we assume that they have a multivariate hypergeometric distribution written as

$$P(x_i^1, ..., x_i^{i-1}, x_i^{i+1}, ..., x_i^N | QL_i(t+\Delta)) = \frac{\binom{A_i^1(\Gamma_t)}{x_i^1} ... \binom{A_i^{i-1}(\Gamma_t)}{x_i^{i-1}} \binom{A_i^{i+1}(\Gamma_t)}{x_i^{i+1}} ... \binom{A_i^N(\Gamma_t)}{x_i^N}}{\binom{\sum_{i \neq j} A_i^j(\Gamma_t)}{QL_i(t+\Delta)}} \tag{2.13}$$

where

$$\sum_j x_i^j = QL_i(t+\Delta)$$

and

$$x_i^j \leq A_i^j(\Gamma_t)$$

x_i^j, represents the number of packets discarded at node i destined to node j.

In this case also, linear optimization is used to minimize the objective function (1.4), which becomes linear in control, by using the constraints (2.8) and (2.11).

2.3 Sharing with minimum allocation Model (SMA)

With the SMA, at every node i, a minimum amount α_i^j of the buffer is always reserved for type-j packets. In addition, another portion of the buffer (B_0) is to be shared among all types of packets. Consequently the maximum amount of the buffer that can be occupied by type-j packets is equal to $\alpha_i^j + B_0$. As a result a type-j packet is accepted in the buffer if upon arrival the queue length (Q_i^j) is less than α_i^j or if the shared section is not full and also (Q_i^j) did not reach the upper limit $\alpha_i^j + B_0$. Therefore, the number of packets of type-j at node i at time t+Δ is give by:

$$Q_i^j(t+\Delta) =$$

$$Q_i^j(t)+\left[\sum_{l\in I(i)}\{U_{li}^j(\Gamma_t)-\xi_{li}^j(\Gamma_t)\}+A_i^j(\Gamma_t)\right]\wedge\left[B_0+\alpha_i^j-Q_i^j(t)-\sum_{c\neq j}Max(0,Q_i^c(t)-\alpha_i^c)\right]$$
$$-\sum_{k\in E(i)}\{U_{ik}^j(\Gamma_t)-\xi_{ik}^j(\Gamma_t)\} \qquad (2.14)$$

The objective function (1.4) and the constraints (2.5), (2.6) and (2.7) will be used in this case as well. The weighting factors are as follows:

$$\alpha_i^j(\Gamma_t) = 1+\frac{Q_i^j(t)+E\{A_i^j(\Gamma_t)\}}{\alpha_i^j+B_0}$$

In order to avoid excess loss of packets, two buffer constraints are considered, the first one is imposed on the packets in the shared area, while the second one puts an upper bound on each queue (Q_i^j). These constraints are respectively

$$\sum_j Max\left(0,Q_i^j(t)+\sum_{l\in I(i)}U_{li}^j(\Gamma_t)+E\{A_i^j(\Gamma_t)\}-\alpha_i^j\right)\leq B_0 \qquad (2.15)$$

and

$$Q_i^j(t) + \sum_{l \in I(i)} U_{li}^j(\Gamma_t) + E\{A_i^j(\Gamma_t)\} \leq B_0 + a_i^j \tag{2.16}$$

In the same manner as explained in the previous sections, the constraint (2.15) and (2.16), make the objective function (1.4) linear in control, which can be minimized using linear programming techniques.

We now proceed with the derivation of the number of lost packets, in this case a type-j packet is discarded, if upon arrival, the queue (Q_i^j) has already reached the upper limit ($a_i^j + B_0$), or when Q_i^j exceeds a_i^j but also the shared area (B_0) is full. Therefore the number of packets lost at node i at time t+Δ, is given by.

$$Q_{lost_i}(t+\Delta) = \sum_j QL_i^j(t+\Delta) + QL_i(t+\Delta) \tag{2.17}$$

where

$$QL_i^j(t+\Delta) =$$

$$\max\left[0, Q_i^j(t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\} - B_0 - a_i^j\right]$$

and

$$QL_i(t+\Delta) = Max\left(0, \sum_j Min(B_0, \psi_i^j(t+\Delta) - a_i^j) - B_0\right)$$

with

$$\psi_i^j(t+\Delta) =$$

$$Q_i^j(t) + \sum_{l \in I(i)} \{U_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\} + A_i^j(\Gamma_t) - \sum_{k \in E(i)} \{U_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\}$$

In equation (2.17), the first term represents the lost packets of type-j when Q_i^j has already reached $B_0 + a_i^j$, while the second one represents the losses from the different types of packets, when the shared area is full.

To determine the number of lost packets of each type, we assume that the packets represented by QL_i have a multivariate hypergeometric distribution as explained in section (2.2).

Chapter 3

Simulation

In order to study the impact on network performance of the various modifications incorporated in the models presented in Chapter 2, namely, the complete sharing model (CS), the square root sharing model (SQRT) and the sharing with minimum allocation model (SMA), simulation of these models and model [1] was conducted on the network plotted in Figure 3.1. This network is composed of three nodes interconnected by six unidirectional channels each having a capacity of 50 packets/unit-time. A buffer of size B_i is provided at each node i for the modified models. For model [1] two buffers of size B_i^j are provided at every node i.

Five important performance measures are observed and compared, namely, (1) the network average throughput, (2) the average delay, (3) the total queue length, (4) the number of lost packets and (5) the total number of external arrivals. The throughput is the number of packets that are successfully delivered to their destinations per unit-time. The delay represents the average time that a packet

spends in the network before reaching its destination. The total queue length is equal to the sum of all the queues present at every buffer of every node over the entire simulation. The packets lost represents the number of packets discarded at the buffer gate due to the shortage of the available buffer space, over the entire simulation as well. Finally the total external arrival represents the total number of external packets arriving at the network during the simulation.

Based on the simulation results, four comparisons have been carried out in Chapter 4 to illustrate the impact of the modifications mentioned earlier. We characterize the first three comparisons as follows:

- comparison of model [1] and the CS model to observe the impact of using a single buffer shared by all the packets instead of having a buffer for each type of packet.

- comparison of model [1], the CS and the SQRT models to investigate the effect of restricting the buffer sharing by imposing some limits on the queue lengths.

- comparison of model [1], the CS, and the SMA models to observe the impact of another type of restricted buffer sharing incorporated by the SMA scheme.

Also, to investigate the impact of coupling dynamic routing with buffer management (flow control), a fourth comparison is made between the three modified models, model [1] and a fixed routing scheme. Similarly to the CS model, in the fixed routing scheme at every node a single buffer is shared among all packets.

However, the route of each packet is previously chosen based on minimum hop [32], and this is always fixed. i.e., every packet arriving at a node will be sent through a pre-fixed channel. The specifics of this fixed routing scheme are detailed later. The simulation of the fixed routing scheme is also conducted on the network of Figure 3.1.

Two kinds of traffic patterns are considered. The first is a balanced load pattern, in which the mean external arrival rates at all the buffers are equal. These rates are kept constant throughout the simulation. In the second traffic pattern, an unbalanced external load is applied to the system to observe the behavior of the models in a more realistic scenario. In this case the mean external arrival rates are selected according to a discrete uniform distribution.

During the simulation, the random events, such as the number of external arrivals and the transmission errors are generated by the IMSL subroutine GGPOS (generates random numbers from a Poisson distribution). The external arrival rates in the unbalanced case are generated by a second IMSL subroutine RNUND which generates random numbers from a uniform distribution. While the third IMSL subroutine called RNHYP generates the number of lost packets of each type from a hypergeometric distribution.

With the balanced traffic pattern as mentioned earlier the mean external arrival rate is kept constant at every buffer throughout the simulation. This rate increases over the range [1,70] packets/unit-time with an increment of 1 packet/unit-time. We expect congestion to occur when the mean external arrival rate reaches

the channel capacity which is equal to 50. Thus we believe that with values of the external arrival rate in the range [1.70] we will be able to observe the behavior of the network before congestion (low arrival rate) and under congestion (arrival rate larger than 50). For each value of the mean external arrival rate the simulation is repeated for 60 time intervals (steady-state conditions are reached with 60 simulations) after which the simulation stops. A total of 10 independent replications are made to have more accurate results. For each replication the initial conditions are the same however the random number seeds differ. The results in the following chapter are within 95 % confidence intervals, and have a maximum standard deviation of 5 %.

In the unbalanced traffic case the mean external arrival rates are uniformly distributed over the range [0,k]. These are the average arrival rates which are used at each time interval to randomly generate the number of external arrivals from a Poisson distribution. The simulation is repeated for 60 intervals of time. For each interval of time a different seed is used.

In order to test model [1], the three modified models described in chapter 2 and the fixed routing scheme mentioned earlier, under light, moderate and heavy loads, the values of k are chosen as follows:

- 50, 90 and 125 in the comparison of model [1] and the CS model.
- 50, 90 and 105 in the comparison of model [1], the CS and the SQRT models.
- 50, 85 and 105 in the comparison of model [1], the CS and the SMA models.
- 50, 85, and 105 in the comparison of the four models with the fixed routing

scheme.

As mentioned earlier the simulation is conducted on a 3-node network. Its small size permits the analysis of the models within a decent computing time. For instance, in the CS, the SQRT and the SMA models each optimization takes an average of 0.01 seconds while each optimization in model [1] takes an average of 0.02 seconds.

The FORTRAN programs for model [1], the three modified models and the fixed routing scheme are given in the appendix. The methodology used in this simulation is presented in the following section and the flow chart of the computer programs is given in Figure 3.2.

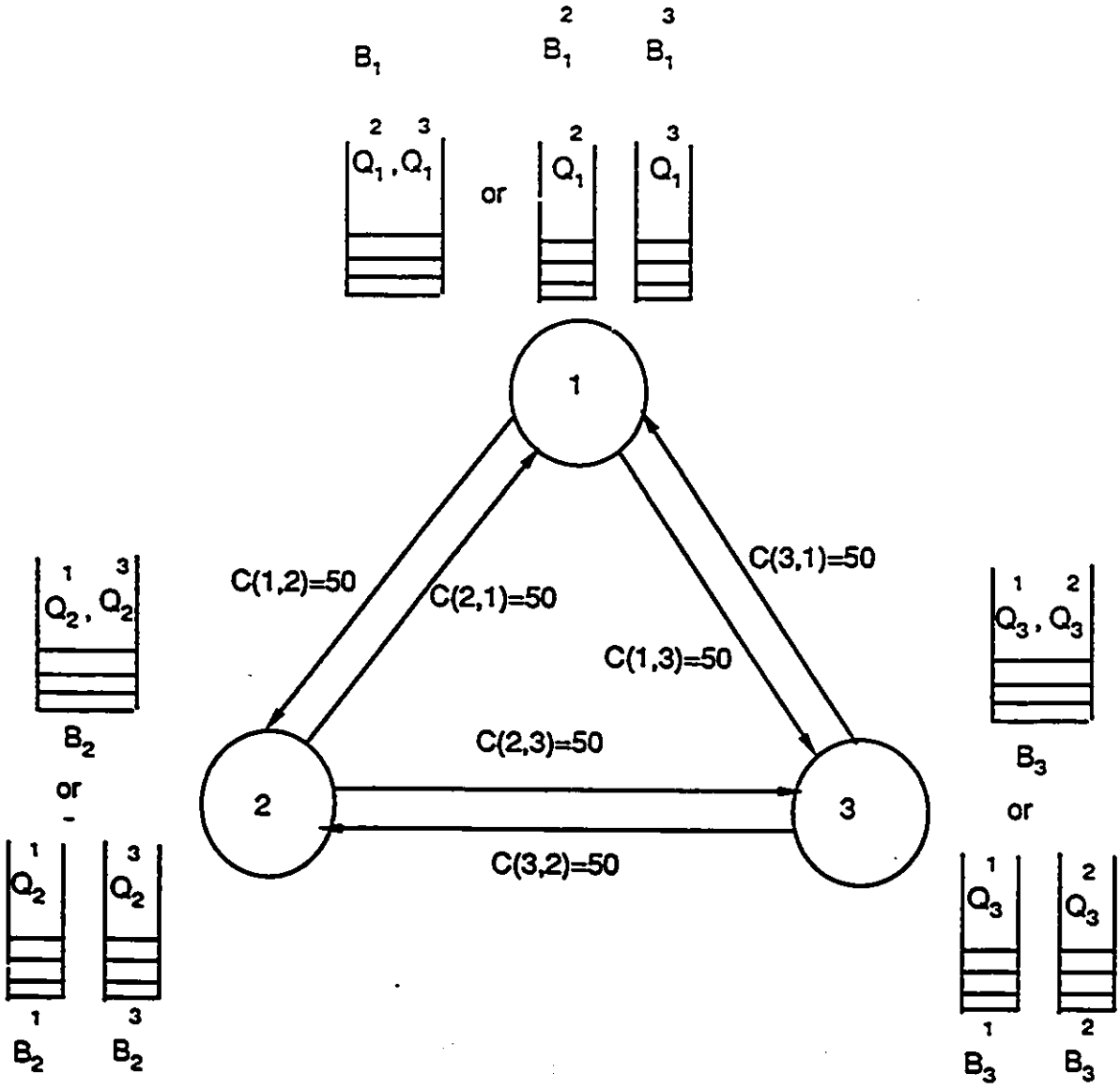

Figure 3.1: The network (used in the simulation)

3.1 Methodology

Figure 3.2 represents the simulation flow chart. The major steps required to perform this simulation are : initialization of the algorithm, generation of the external arrivals, formulation of the constraints, optimization of the objective function and execution of the routing policy.

Initialization

In this first step, the initial time t is set to zero, the final time T is set to 60, and the network constant parameters, namely the size of the buffers, the capacity of the channels C_{ik} and the channel error rates $\tilde{\rho_{ik}}$ are provided (as in the numerical example given below). The initial state of the queues is then arbitrarily chosen (in this case it is equal to 50 packets).

Generation of the external arrivals

In this simulation as mentioned earlier two traffic patterns are used. So the external arrivals are determined depending on the traffic pattern. For the balanced traffic case, the same value is attributed to all the mean external arrival rates of every buffer, which will be kept the same throughout the simulation (this means that all $E\{A_i^j(\Gamma_t)\}$'s are equal for all $i, j \in \{1, 2, 3\}$ and all Γ_t). For the unbalanced load case, the mean external arrival rates are selected from the discrete uniform distribution over [0,k], by the IMSL subroutine RNUND, where k takes the values specified earlier. The IMSL subroutine GGPOS is then used to generate the external arrivals from the Poisson distribution with the appropriate mean.

Formulation of The Constraints

In this third step, new constraints are formulated, based on the queue states and the mean external arrival rates determined above. At this step, new weighting factors are determined as well based on the same information.

Optimization

Now that all the necessary data for optimization is available, and the constraints are formulated, the IMSL subroutine DLPRS (solves linear programming problems via the revised simplex algorithm) is used to minimize the objective function (1.4), subject to the given constraints.

Execution of the Routing Policy

After a decision has been taken, and flow variables are obtained, the routing procedure is executed and the queues are updated. The above steps are repeated until the final time T is reached, at which time the performance measures are computed and listed for analysis.

In the fixed routing scheme, almost the same steps are followed. The only difference is that in model [1] and the three modified models the control variables $(U_{ik}^j(\Gamma_t))$ are calculated through optimization, while in the fixed routing scheme they are calculated as follows: if $Q_i^j(t) + A_i^j(\Gamma_t)$ is greater than C_{ik} then $U_{ik}^j(\Gamma_t) = C_{ik}$, otherwise $U_{ik}^j(\Gamma_t) = Q_i^j(t) + A_i^j(\Gamma_t)$ for all $i, j \in \mathrm{N}$ and $(i, k) \in \mathrm{L}$.

With balanced traffic the above process is replicated 10 times with the same initial conditions and different seeds, after which the mean values, the standard deviations and the confidential intervals are calculated.

3.2 Numerical Example

For simplicity we assume that all the channels have the same capacity and the same error rate, the buffers have the same size (B_i for the three modified models and the fixed routing scheme, and B_i^j for model [1]), and the queues have the same initial state. Thus we have the following data:

- the size of the single buffer at node i: B_i= 200 packets (in the CS, SQRT and SMA models as well as the fixed routing scheme).
- the size of the buffer of packets of type j at node i: B_i^j= 100 packets (in model [1]).
- channel capacity: C_{ik} = 50 packets/unit-time.
- initial queue state: $Q_i^j(0)$= 50 packets.
- error rate : $\tilde{p}_{ik}$ = .001.

In the simulation of the SMA model the buffer space reserved for each type of packet a_i^j is set to a constant value a_0=50 packets.

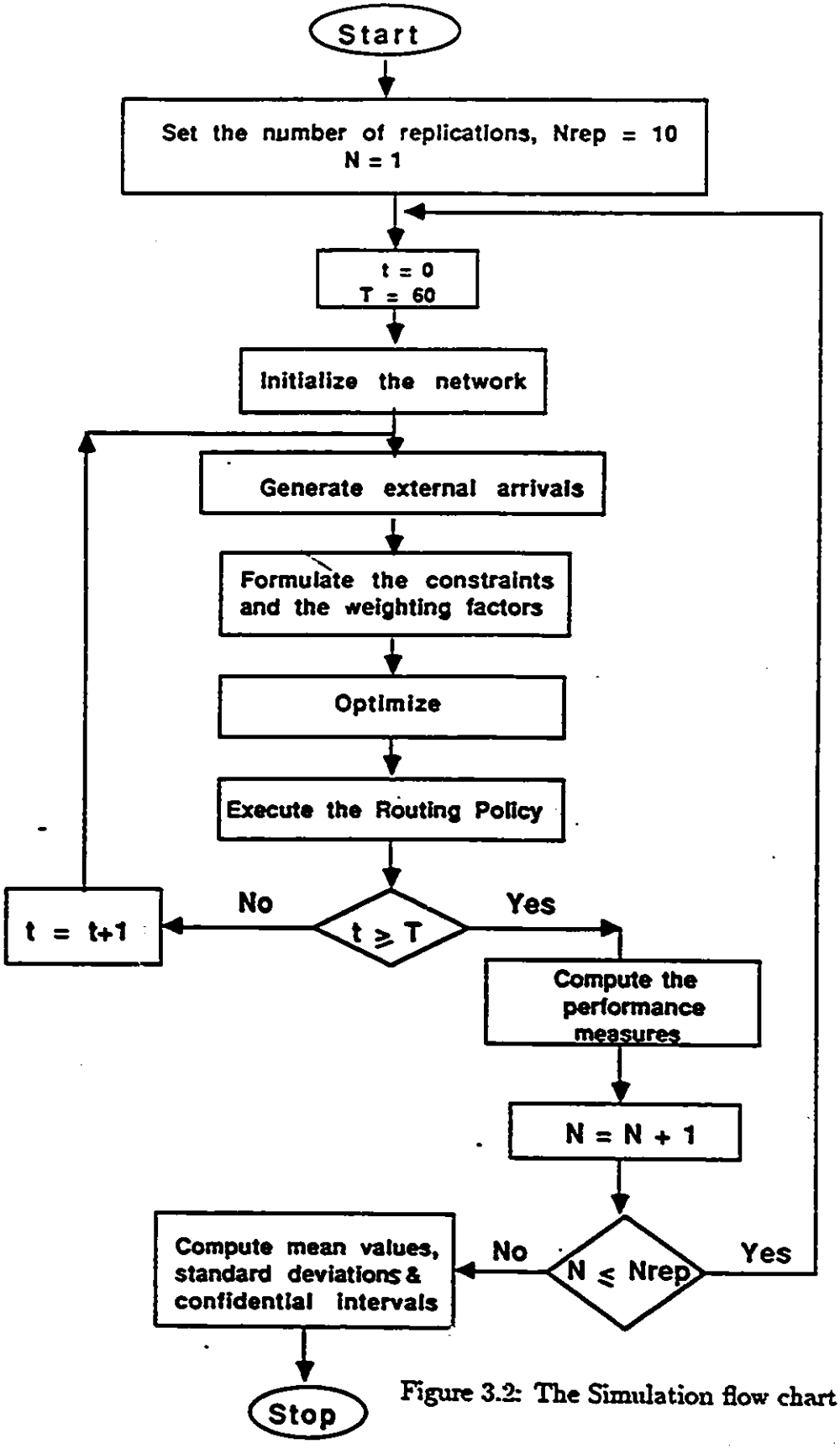

Figure 3.2: The Simulation flow chart

Chapter 4

Simulation Results and Comparisons

In this chapter the four comparisons mentioned in chapter 3, namely, (1) comparison of model [1] and the CS model. (2) comparison of model [1], the CS and the SQRT models. (3) comparison of model [1], the CS and the SMA models. (4) comparison of the four models with the fixed routing scheme described in Chapter 3 are carried out. In each of the three first comparisons two cases are examined. The first is a comparison under a balanced load. In the second one the comparison is made under an unbalanced load. The fourth comparison is conducted under an unbalanced traffic load only.

It is important to note that in all the figures and tables in this chapter the mean external arrival rates and the average throughput are measured in terms of packets/unit-time. The losses, the total queues and the total arrivals are measured in number of packets. Finally the average delay is measured in terms of unit-

time/packet.

4.1 Comparison of model [1] and the CS model

Balanced Load

Consider first the case where all the mean external arrival rates are equal. Figures 4.1, 4.3 and 4.4 illustrate respectively the behavior of the number of packets discarded due to shortage in the available buffer space, the average throughput, and the average delay with respect to the mean external arrival rate. Figure 4.2 magnifies a portion of the Figure 4.1 graphs. In Figures 4.1, 4.3 and 4.4, the horizontal axis represents the mean external arrival rate, with values in the range [1,70], while that of Figure 4.2 covers the range [40,55].

As seen from Figure 4.2, before congestion, the CS model loses fewer packets than model [1], since with the CS no space is wasted, while with model [1] the portion of the buffer allocated to a short queue is wasted and cannot be used by any other queue. On the other hand the average throughput and the average delay are the same for both models.

Under congestion, the performance of the CS model becomes worse than model [1]; in fact the CS leads to more packets lost, a lower average throughput and a higher delay. This behavior is attributed to the fact that even under a balanced traffic where the mean external arrival rate is the same at every buffer of every node, some queues may have more arrivals than others due to the Pois-

son distribution. Moreover with the CS model, packets share the buffer with no restrictions thus the queues having more arrivals tend to occupy a larger amount of the buffer leaving less space for the others. As a consequence more packets are lost, and some channels are not fully utilized which means a lower throughput and a higher delay. The impact of such circumstances is more significant under an unbalanced load; thus more details are given in the following section.

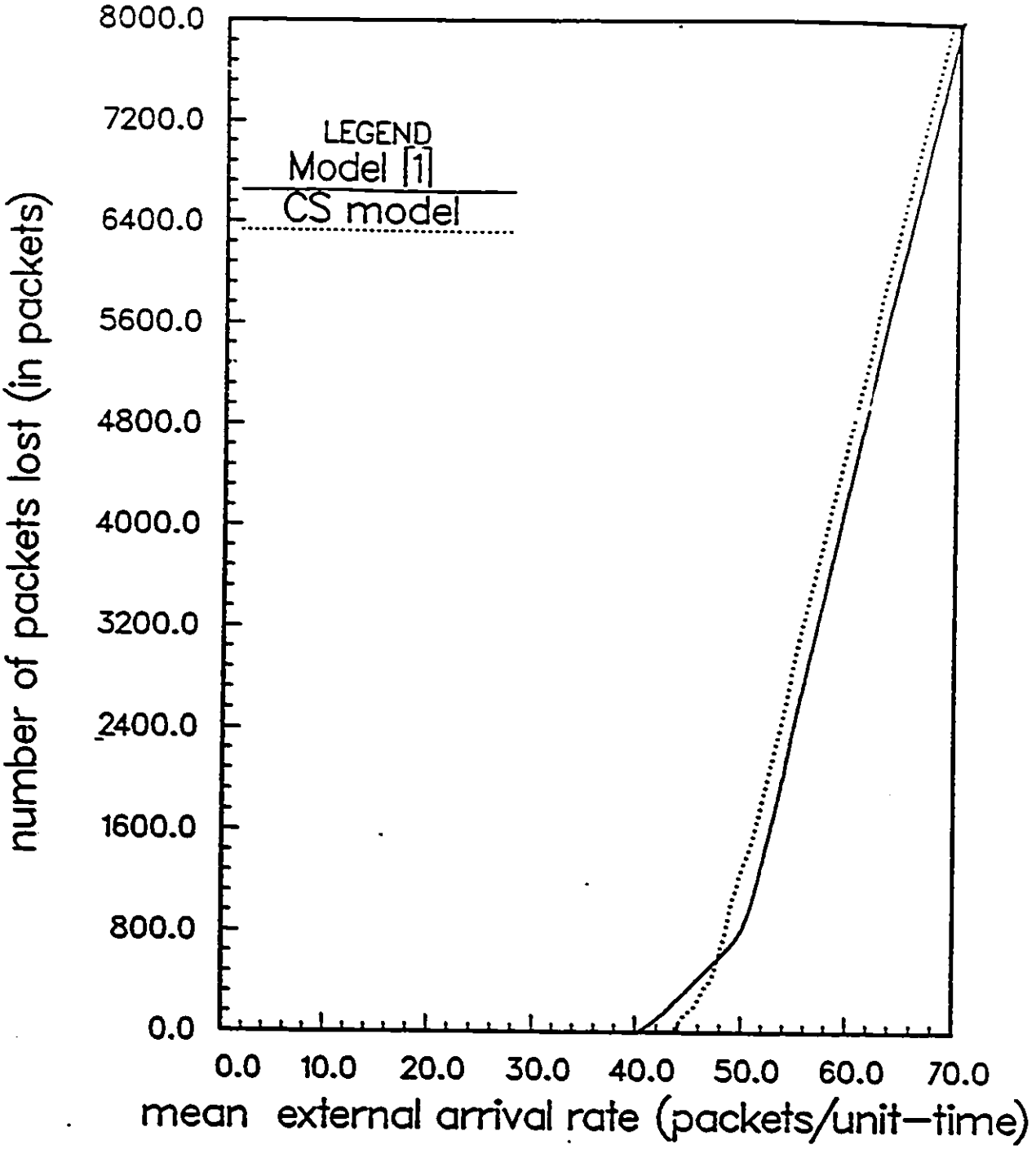

Figure 4.1: Lost packets for model [1] and the CS model under a balanced load (mean external arrival rate in the range [1,70])

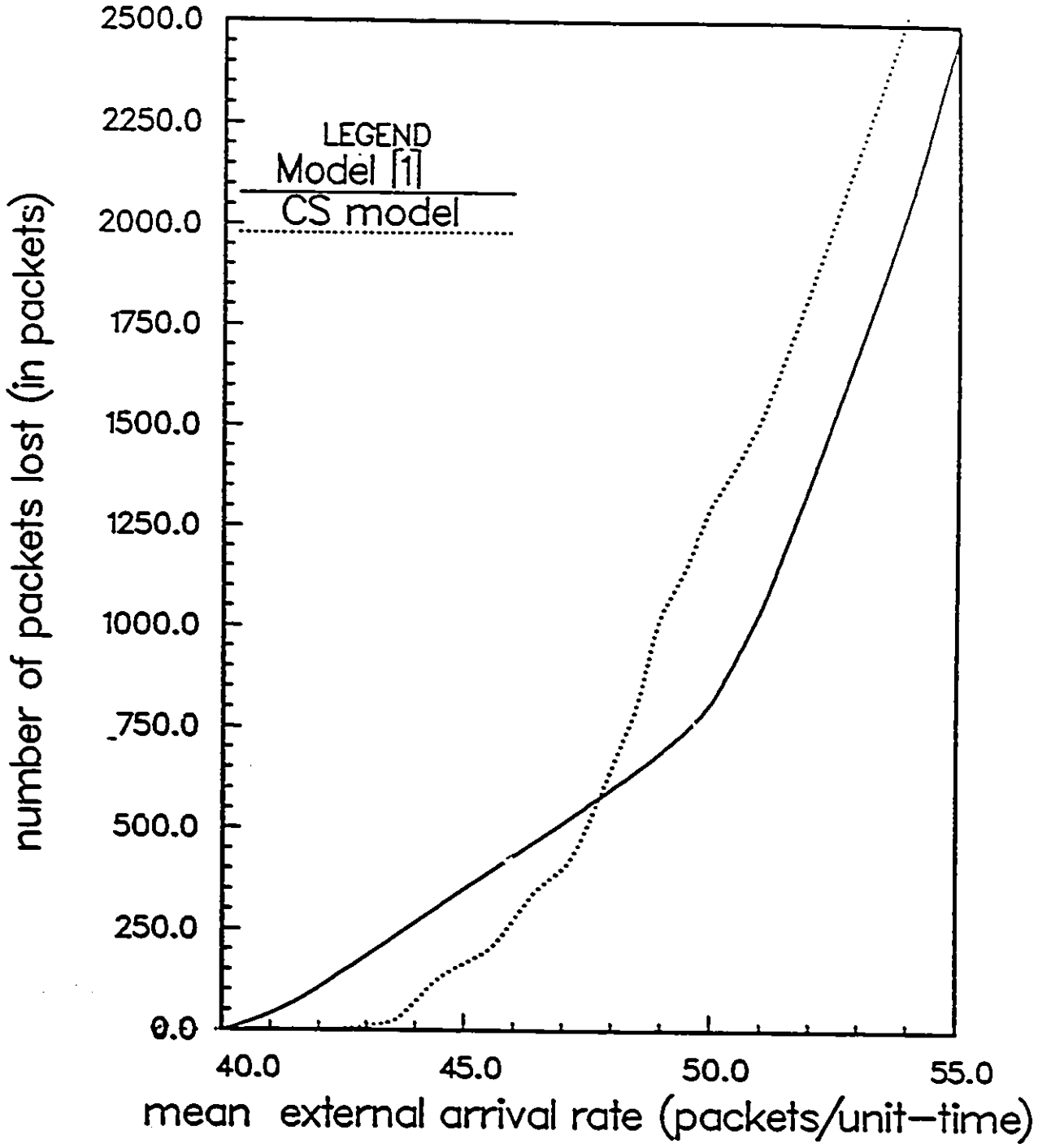

Figure 4.2: Lost packets for model [1] and the CS model under a balanced load (mean external arrival rate in the range [40,55])

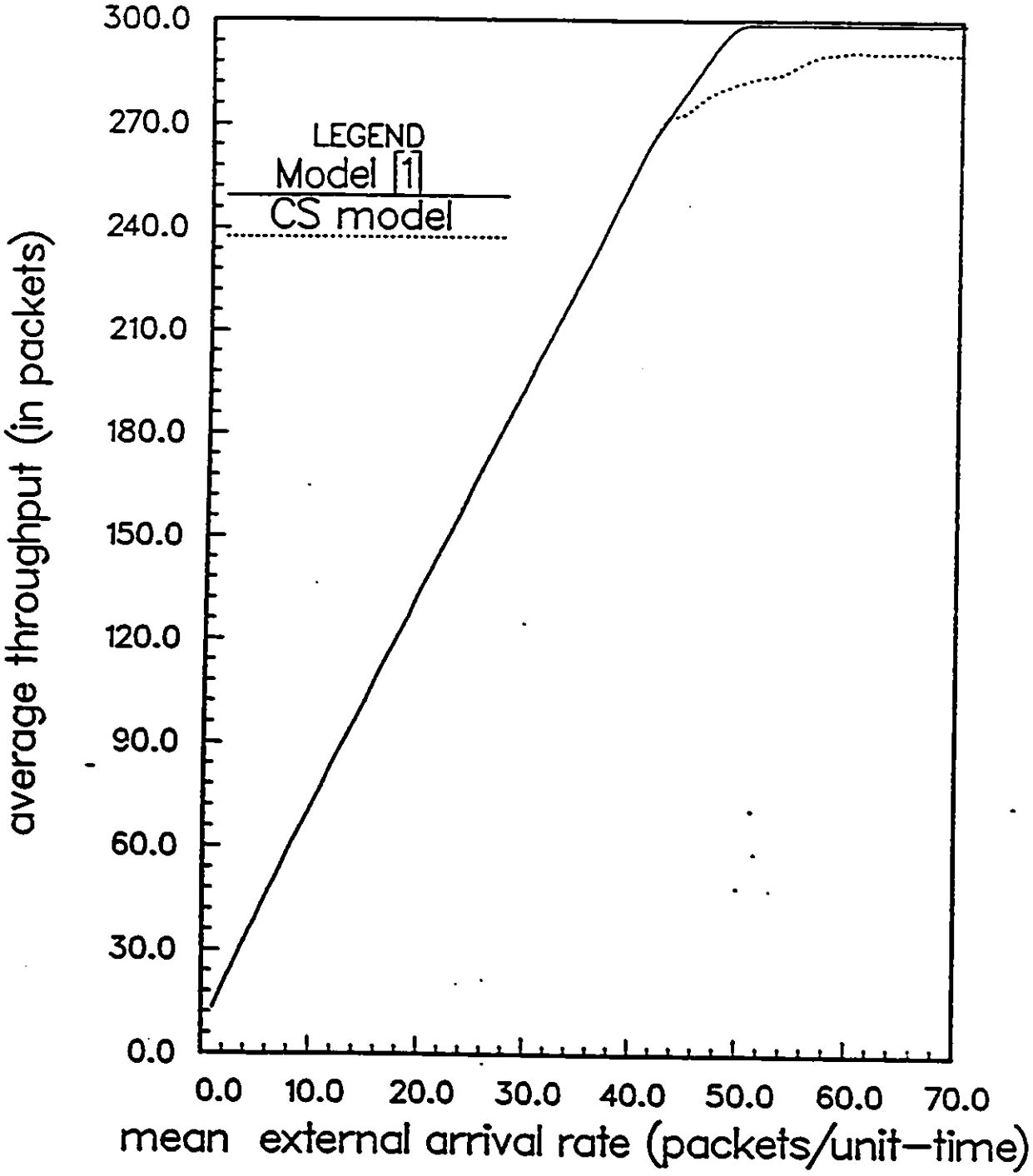

Figure 4.3: Average throughput for model [1] and the CS model: Balanced load

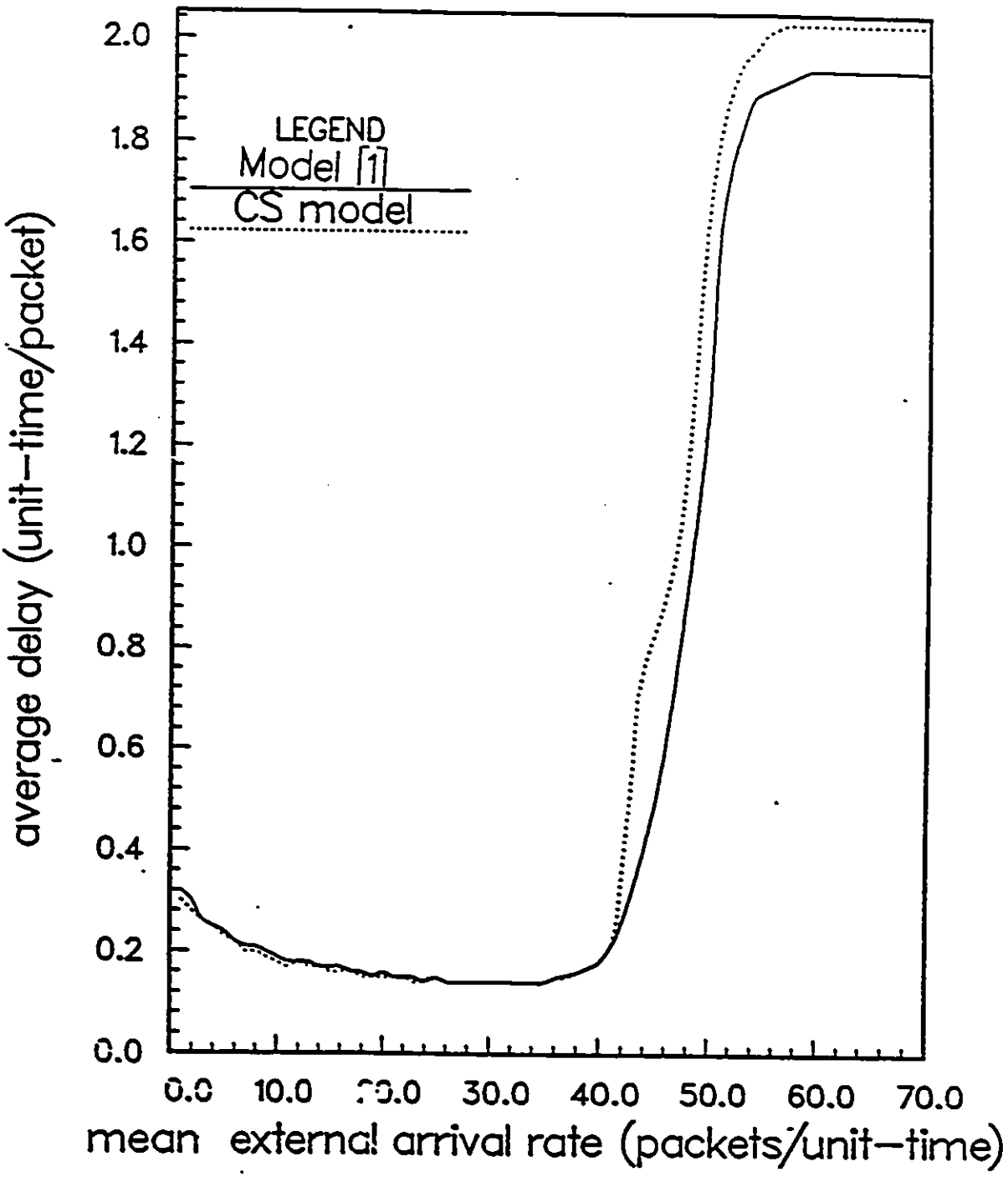

Figure 4.4: Average delay for model [1] and the CS model: balanced load

Unbalanced Load

In this case, an unbalanced external input is applied to the system, to compare the two models in a realistic situation. Three cases are investigated, where the mean external arrival rates are uniformly distributed over the ranges [0,50], [0,90], and [0,125] respectively, in order to test these models under increasing congestion

The obtained results are summarized in Tables 4.1, 4.2 and 4.3, showing the network average delay, average throughput, the number of packets lost, the total queue lengths and the total number of external arrivals.

As expected, for light loads both models produce the same results as shown in Table 4.1. Note that no packets are lost for mean external arrival rates below 50, no matter which model is used.

Table 4.1: Network performance for model [1] and the CS model: unbalanced load (mean external arrival rate uniformly distributed over [0,50])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.13	165	0	1334	9618
CS	0.14	165	0	1341	9618

As the traffic increases (Table 4.2), where the range of the mean external arrival rates is extended to [0,90], congestion builds up, and model [1] shows some shortage dealing with this problem due to the no sharing rule which causes wastage in the buffer space. This explains why the total queues are shorter, while the number of lost packets is higher than with the CS model. However the CS model works very well with this increase in the offered load, especially in accepting more packets in the buffers due to the sharing rule which allows longer queues to be accepted and use the space left by the shorter ones, and hence reducing the number of packets discarded and increasing the throughput as well. As a consequence of these long queues, the delay is higher with the CS model than with model [1].

Table 4.2: Network performance for model [1] and the CS model: unbalanced load (mean external arrival rate uniformly distributed over [0,90])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.84	273	554	13841	16888
CS	1.05	274	431	17236	16888

In the third case, the range is extended to [0,125], to observe the impact of a very heavy load on the performance of the network.

Table 4.3, shows a considerable degradation in the performance of the CS model compared to model [1]. The CS has more packets lost, lower throughput, and a higher delay. It also shows that the sharing rule which led the CS model to react very well under a moderate load, fails when the traffic becomes heavier.

This behavior is explained by the following: under a heavy unbalanced load, sharing the buffer with no restriction allows some queues to get longer and longer even after saturation of their corresponding channels which leads to the monopolization of the buffer by one or more of these long queues to the detriment of others. Moreover, although alternative paths can be provided, due to the dynamic routing strategy used, the problem is not solved, because transit packets cannot be sent to the neighboring nodes unless these nodes have enough storage space, but this may not be always true since a heavy load is applied to the system. As a consequence to this some transmission capacity is wasted which means a degradation in the throughput, the delay increases and more packets are lost.

We conclude that neither scheme is consistently better over the entire range of offered load. For light and moderate loads, the CS model reacts very well in preventing an excessive loss of packets without degradation in the throughput or increase in the delay, whereas model [1] is more efficient under heavy loads.

Table 4.3: Network performance for model [1] and the CS model: unbalanced load (mean external arrival rate uniformly distributed over [0.125])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	1.56	298	5208	27970	23220
CS	1.89	291	5451	33073	23220

4.2 Comparison of model [1], the CS and the SQRT models

Our main objective in this section, is to compare the three models: [1], the CS and the SQRT, and show how the SQRT model can improve the network performance by introducing some limits on the queue lengths.

Balanced Load

Following the same steps as earlier, we first consider the case where all the mean external arrival rates are equal. Figures 4.5, 4.6 and 4.7, illustrate the performance comparison.

Results in Figure 4.5 clearly show that under a balanced load, model [1] performs the poorest for light loads, while the CS model performs the worst for moderate and heavy loads, as explained earlier. On the other hand the SQRT seems to behave fairly well over all loads. Under congestion the SQRT model improves the performance degradation caused by the CS, while before congestion the SQRT model reduces the packet losses compared to model [1]. However it loses more packets than the CS model. The reason for this is that although the SQRT model provides flexibility of sharing the buffers among all the types of packets, some storage space is wasted when the load is light, because of the limits imposed on the queues.

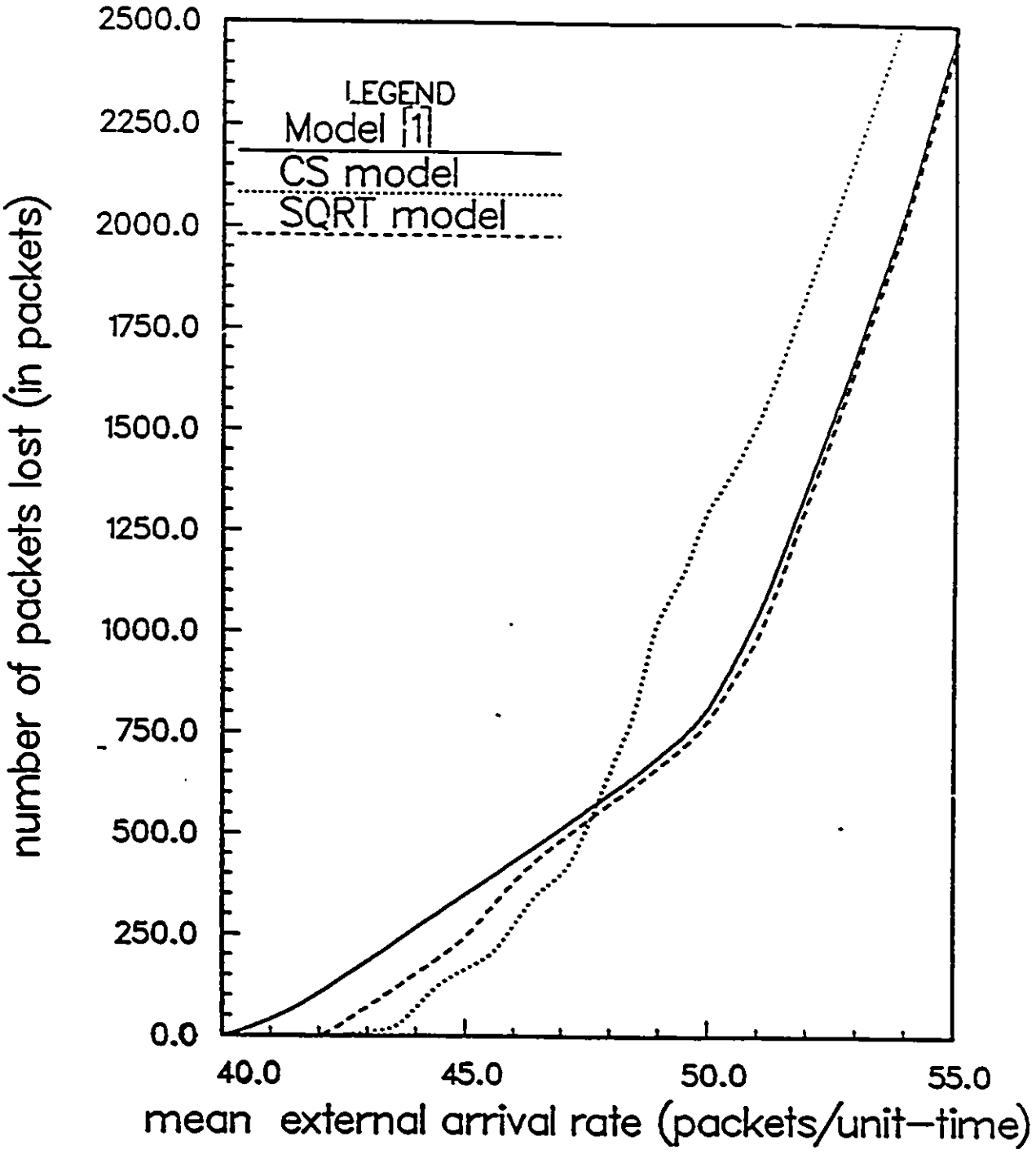

Figure 4.5: Lost packets for model [1], the CS and the SQRT models under a balanced load (mean external arrival rate in the range [40.55])

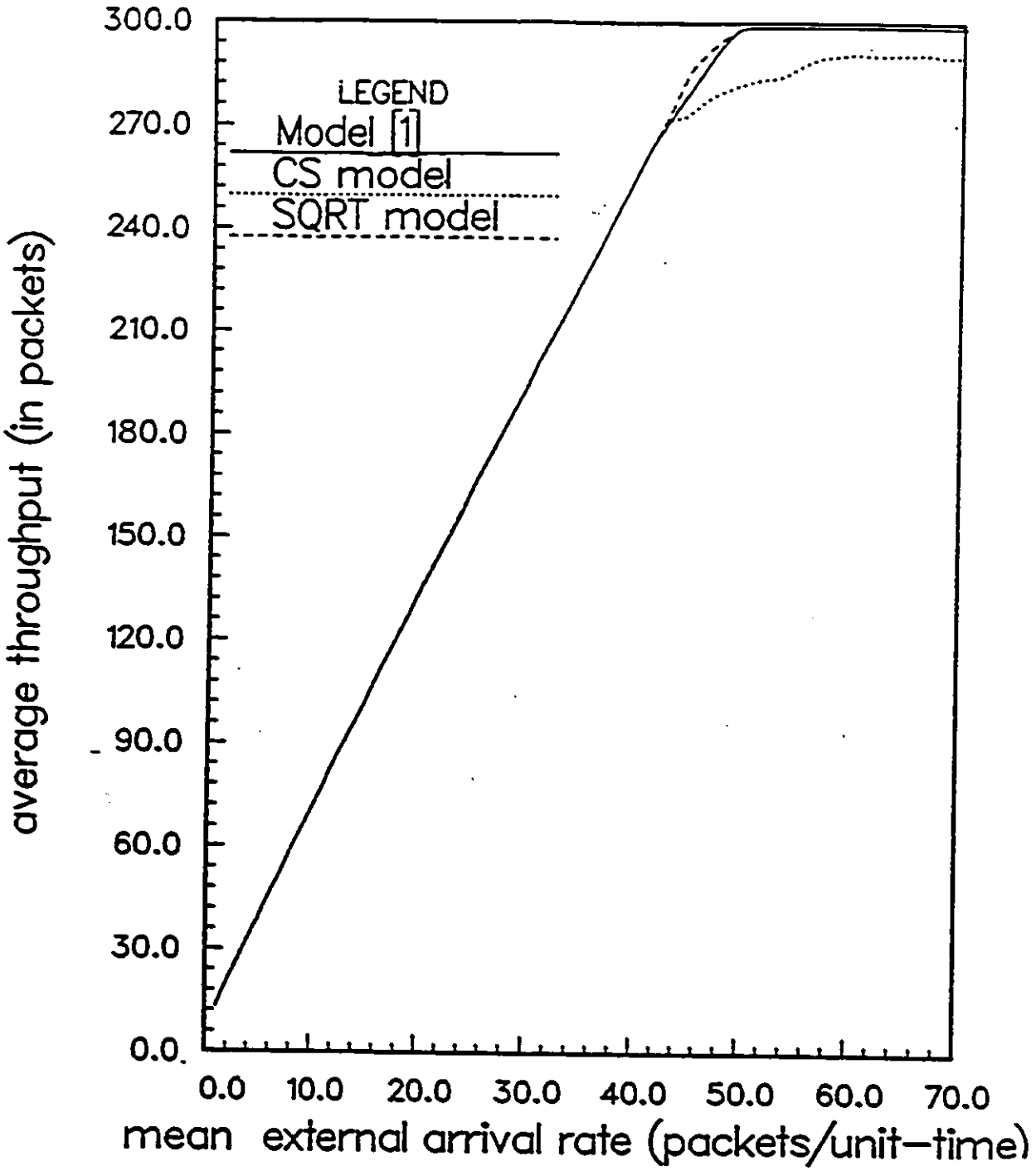

Figure 4.6: Average throughput for model [1], the CS and the SQRT models: balanced load

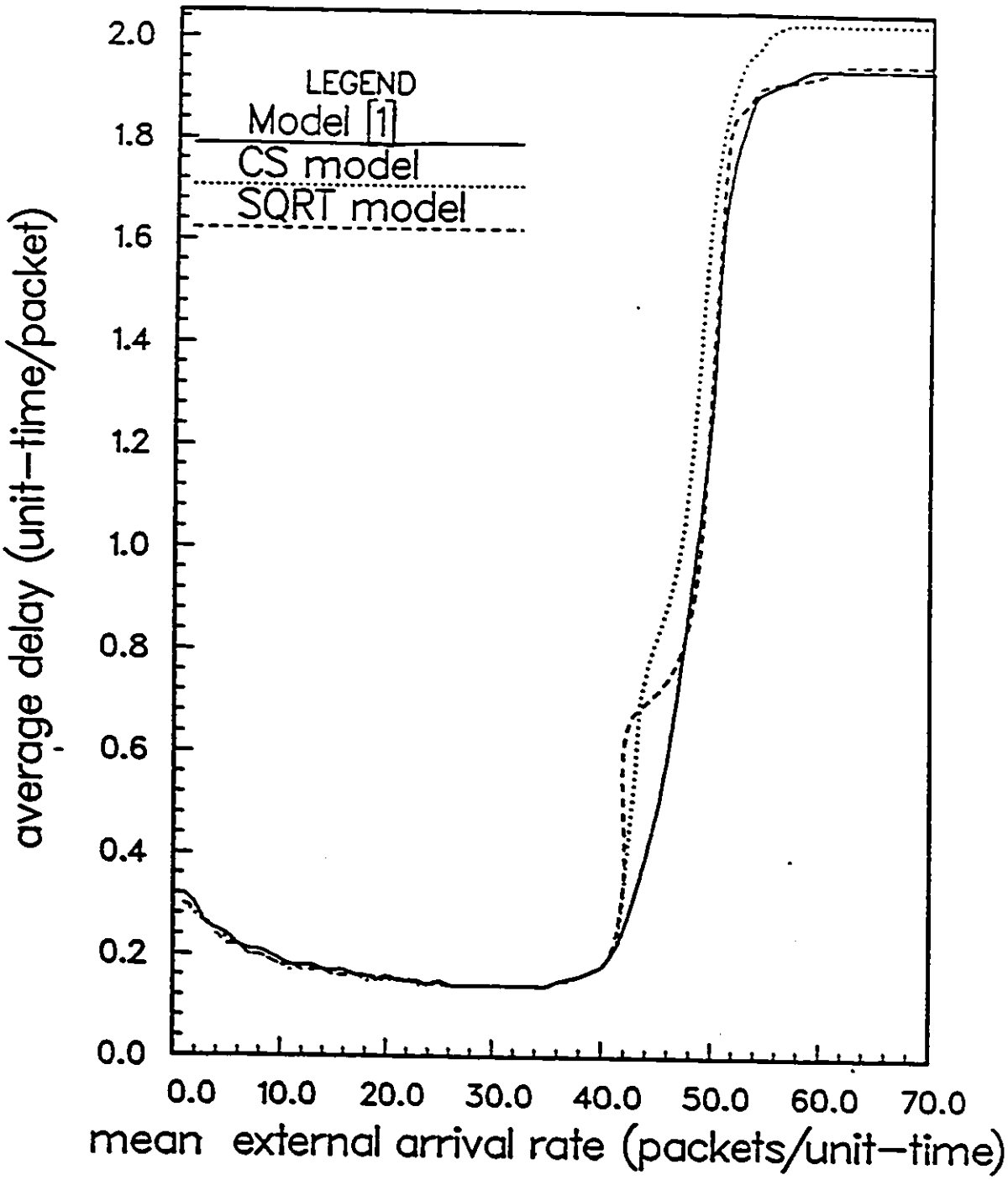

Figure 4.7: Average delay for model [1], the CS and the SQRT models: balanced load

Unbalanced Load

In this part, the mean external arrival rates are uniformly distributed over the ranges [0,50], [0,90] and [0,105]. The behavior of the three models over these ranges, under an unbalanced load are respectively illustrated by tables 4.4, 4.5 and 4.6.

Table 4.4: Network performance for model [1], the CS and the SQRT models: unbalanced load (mean external arrival rate distributed over [0,50])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.13	165	0	1334	9618
CS	0.14	165	0	1341	9618
SQRT	0.14	165	0	1337	9618

It may be observed from Table 4.4, that the results are the same for the three models before congestion. However significant differences appear between the performance of these three models, as the traffic increases and becomes more important. As shown in Table 4.5, for moderate traffic model [1] always gives the worst result with respect to the number of packets lost, since it provides the least flexible type of sharing. On the other hand it yields the lowest average delay, due

to the excessive loss of packets, which results in shorter queues.

An improvement in the number of packets lost, is observed with the SQRT compared to model [1] when the mean external arrival rates are distributed over the range [0,90]. However it is interesting to note that the CS model performs the best in such conditions.

Table 4.5: Network performance for model [1] the CS and the SQRT models: unbalanced load (mean external arrival rate distributed over [0,90])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.84	273	554	13841	16888
CS	1.05	274	431	17236	16888
SQRT	0.94	273	506	15505	16888

The superiority of the SQRT model over the other two, becomes much more significant under a heavy load. Note from Table 4.6 that when the range is extended over [0,105], the number of lost packets is significantly less and the average throughput is higher with the SQRT than the other two models although model [1] has always the lowest average delay explained by the shorter queues.

This behavior is attributed to the fact that, the SQRT allows all types of

packets to share the buffer to reduce the wastage of storage space under light and moderate traffic loads, but there comes a point where the sharing with no restrictions leads to a degradation in the network performance, this takes place when the traffic load becomes heavier and queues with higher arrivals tend to fill in the buffer. The SQRT model seeks to avoid the above degradation by limiting the queue lengths and preventing the monopolization of the buffer by one or more queues to the detriment of the others. As a consequence more storage space is given to the shorter queues, the utilization of the channels increases, the delay decreases and the number of packets lost is reduced .

The results in Table 4.6 demonstrate the fact that we can improve the network performance by using the SQRT model, which seems to have the ability to avoid the deficiencies of model [1] in the absence of congestion, by using the sharing rule, and also the deficiencies of the CS model under congestion conditions, by introducing limits on the queue lengths, which prevent one or more types of packets from claiming all the buffer and blocking the others.

Another interesting observation that can be made from the above, is that the necessity for implementing the SQRT model depends on the load. In fact, when the load is heavy, it would be wise to use this model rather than model [1] or the CS model since it yields the best performance particularly in terms of average throughput and packet losses. Otherwise, under light loads using the CS model would be better since it shows the best results.

Table 4.6: Network performance for model [1] the CS and the SQRT models: unbalanced load (mean external arrival rate distributed over [0.105])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	1.24	292	2053	21827	19611
CS	1.59	288	2073	27596	19611
SQRT	1.51	294	1815	26648	19611

4.3 Comparison of model [1], the CS and the SMA models

In the SMA model, as defined in Chapter 2, a_i^j is the portion of the buffer permanently allocated to packets of type-j at node i. In our simulation we assumed that $a_i^j = 50$ for all $i, j \in \{1, 2, 3\}$ and $i \neq j$.

Balance Load

Figures 4.8, 4.9 and 4.10, respectively show the number of packets lost, the average throughput, and the average delay obtained with model [1], the CS and the SMA models, under balanced traffic conditions.

Similarly to the SQRT model in the previous section, the SMA shows some improvement in packet losses before congestion compared to model [1], while it provides a full utilization of the channels under congestion.

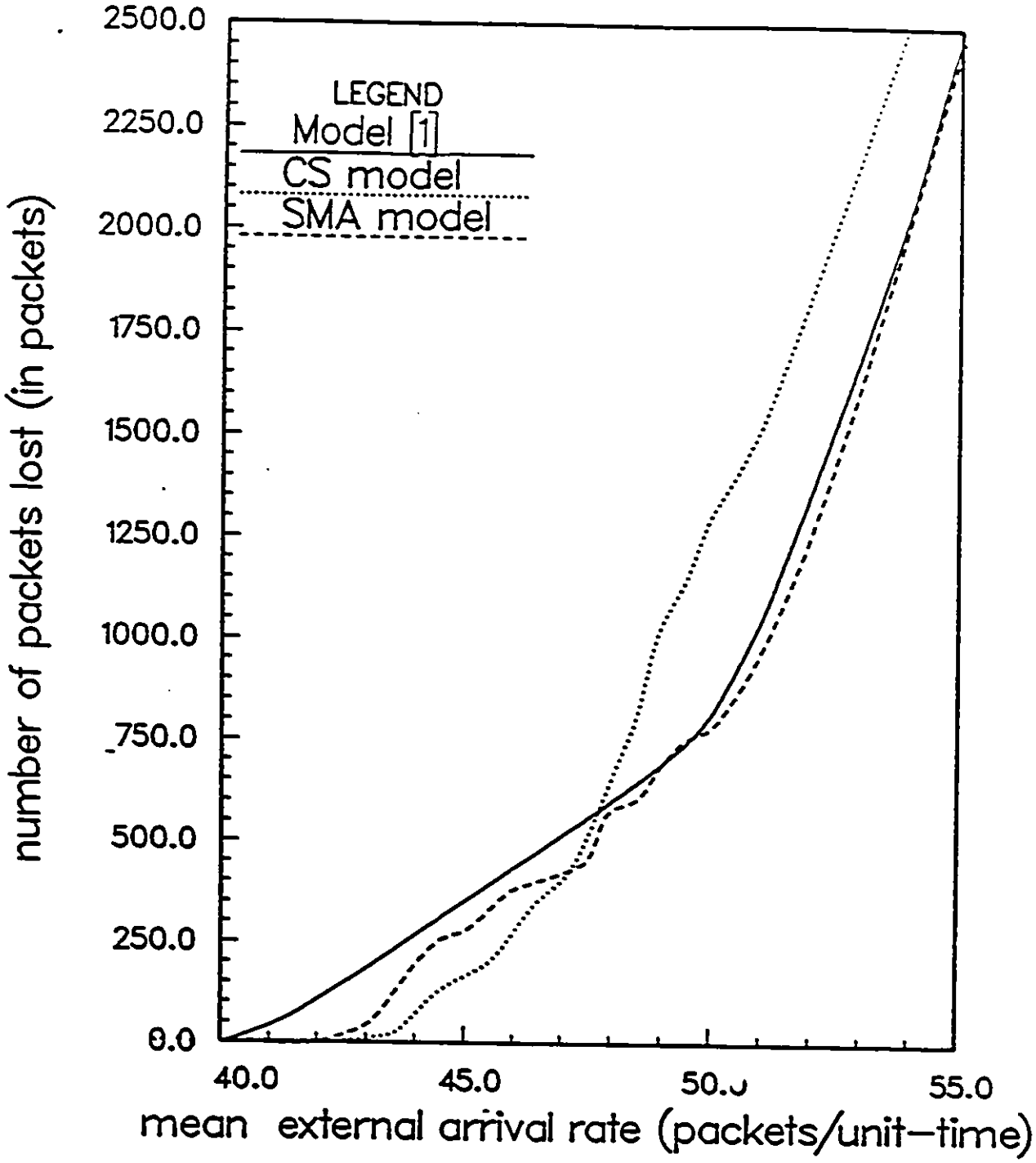

Figure 4.8: Lost packets for model [1], the CS and the SMA models under a balance load (mean external arrival rate in the range [40.55])

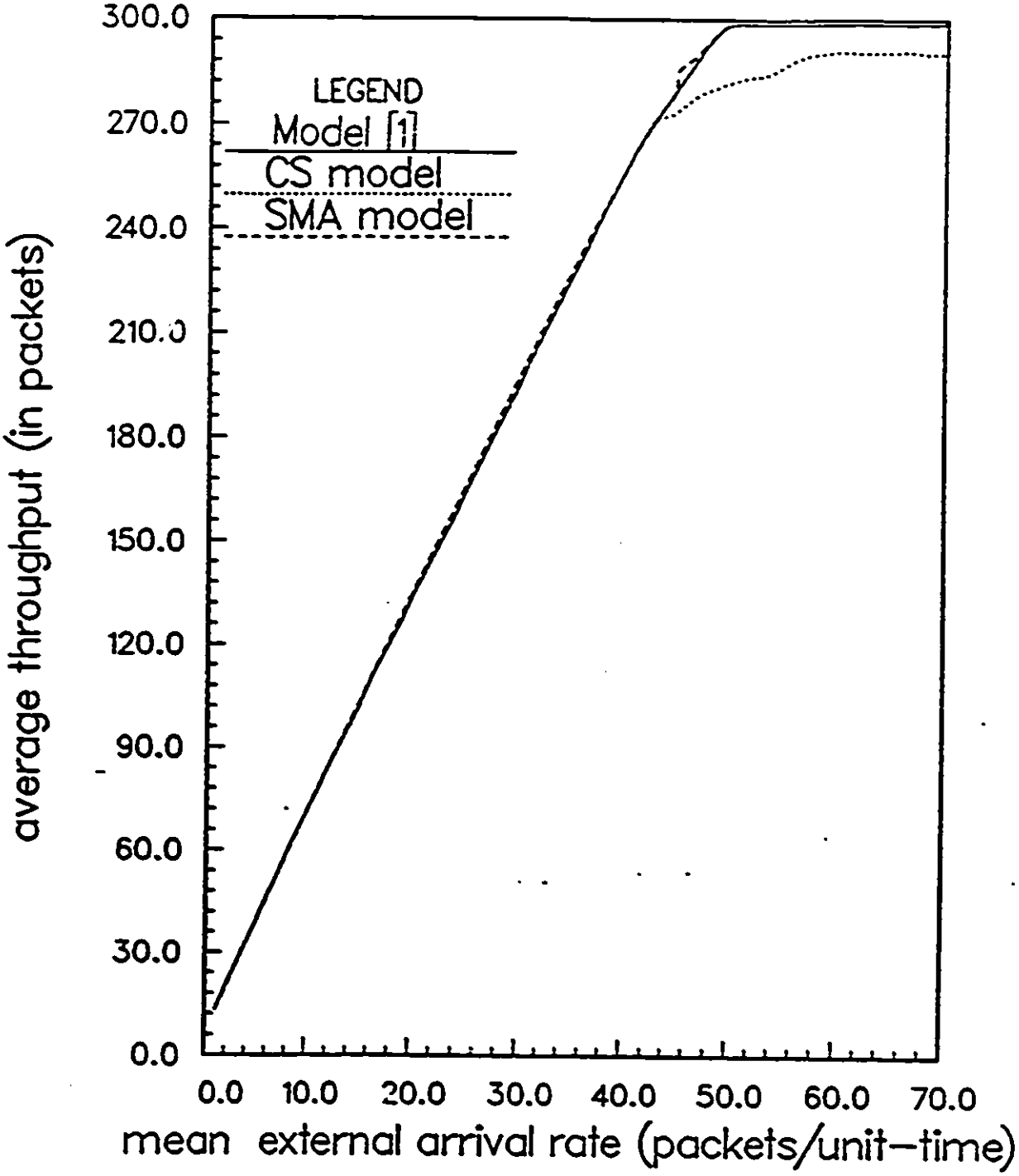

Figure 4.9: Average throughput for model [1], the CS and the SMA models: balanced load

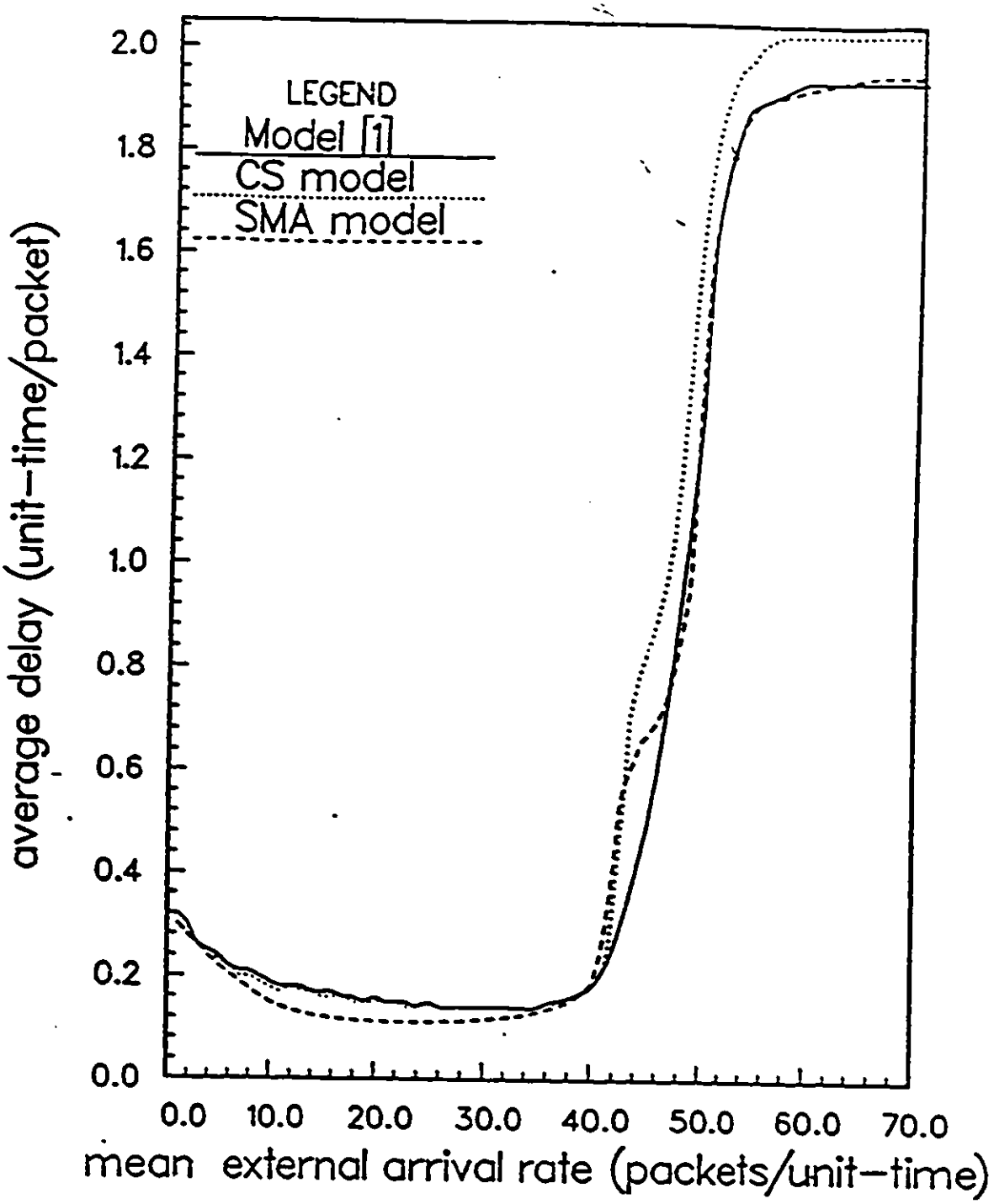

Figure 4.10: Average delay for model [1], the CS and the SMA models: balanced load

Unbalanced Load

The results of model [1], the CS and the SMA models under unbalanced traffic conditions, over the ranges [0,50], [0,85], [0,105] are listed in Tables 4.7, 4.8 and 4.9 respectively.

Table 4.7: Network performance for model [1] the CS and the SMA models: unbalanced load (mean external arrival rate distributed over [0,50])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.13	165	0	1334	961S
SMA	0.13	165	0	1254	961S
CS	0.14	165	0	1341	961S

Table 4.7 shows that the SMA model has the same results as model [1] and the CS model before congestion. As congestion starts to build up the SMA tends to minimize the wastage of storage space by providing a portion of the buffer to share among all types of packets. As a consequence it has less packets lost than model [1]. Note that the SMA has a few more packets lost than the CS where the entire buffer space is shared (Table 4.8)

Table 4.8: Network performance for model [1] the CS and the SMA models: unbalanced load (mean external arrival rate distributed over [0.85])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.71	264	256	11270	15970
SMA	0.83	264	165	13233	15970
CS	0.83	264	154	13205	15970

Besides the shared portion, the SMA permanently allocates some storage space to each type of packet. Thus under a heavy unbalanced load the shorter queues are given more chance to be accepted in the buffer. This allows an efficient throughput while it diminishes the packet losses as shown in Table 4.9.

As a consequence of the large number of packets lost model, [1] always has the lowest delay (see Tables 4.8 and 4.9)

Table 4.9: Network performance for model [1] the CS and the SMA models: unbalanced load (mean external arrival rate distributed over [0.105])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	1.24	292	2053	21827	19611
SMA	1.53	294	1806	27030	19611
CS	1.59	288	2073	27596	19611

4.4 Comparison of the four models with the fixed routing scheme

In order to investigate the impact of coupling dynamic routing with buffer management, we compare model [1] and the three modified models with a fixed routing scheme based on minimum hop. The comparison is made under an unbalanced traffic load where fluctuations between the external arrivals are expected to be more significant which makes the dynamic routing more useful.

Tables 4.10, 4.11 and 4.12 show the results of the four models and the fixed routing scheme under an unbalanced traffic load where the mean external arrival rates are uniformly distributed over the ranges [0,50], [0,85] and [0,105] respectively.

As expected there are no losses when the mean external arrival rates are lower than 50 packets/unit-time. We notice from Table 4.10 that the total queue is shorter with the fixed routing scheme than the other four models consequently the delay is lower. This leads us to say that using dynamic routing before congestion tends to favor the transmission of the transit packets to the neighboring nodes which increases the queue lengths and hence the delay with no improvement in the throughput.

Table 4.10: Network performance for the four models and the fixed routing scheme: unbalanced load (mean external arrival rate uniformly distributed over [0.50])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.13	165	0	1334	9615
SMA	0.13	165	0	1254	9615
SQRT	0.14	165	0	1337	9615
CS	0.14	165	0	1341	9615
Fixed routing	0.041	165	0	405	9615

In Table 4.11 the range is extended to [0,55] so that all methods encounter congestion at some point in the simulation. The CS, the SMA and the SQRT work well in such conditions due to dynamic routing which provides alternative paths for the packets at congested buffers, and the sharing of the buffer which minimizes the waste of the storage space. Although model [1] uses dynamic routing which yields an efficient throughput, it loses more packets than the latter models. The fixed routing shows its limitations to deal with such conditions. In fact under increasing congestion the sharing of the buffer with no restrictions tend to accumulate more packets in the queues, however using a fixed routing scheme leads to underutilization of the channels. As a consequence the throughput is low, the

delay increases as well as the number of packets discarded. This explains why the CS model is the best (in terms of packet losses and throughput) while the fixed routing is the worst although they have the same buffer management scheme.

Table 4.11: Network performance for the four models and the fixed routing scheme: unbalanced load (mean external arrival rate uniformly distributed over [0,85])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	0.71	264	256	11270	15970
SMA	0.83	264	165	13233	15970
SQRT	0.81	264	177	12834	15970
CS	0.83	264	154	13205	15970
Fixed routing	0.86	261	379	13442	15970

Finally in Table 4.12 the range of the mean external arrival rates is extended to [0,105] to observe the impact of a more intensive congestion .

The first thing that we may observe from Table 4.12 is that the CS model yields similar results to the fixed routing scheme. This is attributed to the fact that under a heavy load alternative paths may not be always available so the

routing is almost done in a fixed way. This situation is rectified by using the SMA or the SQRT models where more efficient buffer management schemes are used and the performance results are improved.

Table 4.12: Network performance for the four models and the fixed routing: unbalanced load (mean external arrival rate uniformly distributed over [0,105])

Model	Delay	Throughput	Lost packets	Total queues	Total arrivals
Model [1]	1.24	292	2053	21827	19611
SMA	1.53	294	1806	27030	19611
SQRT	1.51	293	1815	26648	19611
CS	1.59	288	2073	27596	19611
Fixed routing	1.63	288	2075	28185	19611

Despite their efficiency, the SMA and the SQRT models pay a price for having less packet losses and more throughput. This price is the high delay. Note that model [1] yields the lowest delay as explained earlier (see Tables 4.11 and 4.12).

Finally, an interesting observation is that the differences between the results in Tables 4.10, 4.11 and 4.12 are not very significant. This is explained by the

small size of the network. In fact as the number of nodes in the network increases the waste of the buffer space in model [1] is expected to increase as well as the underutilization of channel capacities in the fixed routing scheme, and hence the impact of the modifications introduced in this thesis will be more significant.

Chapter 5

Conclusions and Recommendations

5.1 Conclusions

In this thesis, we considered the incorporation of three buffer management schemes, namely (1) complete sharing scheme, (2) square root sharing scheme and (3) sharing with minimum allocation scheme, into the dynamic routing model introduced in [1], in order to control congestion in store and forward computer communication networks.

Simulation experiments have demonstrated that the large number of packets lost with model [1] due to the wastage of storage space makes it less efficient under a light load. On the other hand the examination of the results of the CS model revealed that the latter succeeds in achieving a better performance than model [1] under a balanced light load. However for an unbalanced heavy load, the CS tends to favor longer queues or the queues with higher input rates, which leads to the

monopolization of the storage space by one of the queues, and hence performance deterioration.

By introducing some limits on the queue lengths (in the SQRT model) we can partially remedy this situation. This reduces the packet losses and improves the throughput; however the delay is increased.

We also studied another solution to the buffer monopolization problem caused by the CS model. This solution consists of permanently allocating a portion of the buffer to each type of packet while providing another portion to share among all types of packets (the SMA model). This also leads to improvements in the packet losses and throughput, particularly under heavy unstable traffic loads.

Finally, we can conclude that the selection of a specific model must take into consideration the average delay, the average throughput and the packet losses, as well as the load on the system.

5.2 Recommendations

In this thesis, we have considered buffer management schemes where the limits on the queues are set to a fixed value. For further research it would be interesting to investigate dynamic routing together with adaptive buffer management schemes, where the limits are selected depending on the dynamically varying load conditions.

It would be also interesting to extend the work done in this thesis to include networks with more than three nodes, and observe the effect of the size of networks

on their performance. We expect the impact of the modifications incorporated in this thesis to be more significant as the number of nodes in the network increases.

Bibliography

[1] N. U. Ahmed, T. E. Dabbous and Y. W. Lee, "Dynamic routing for computer queueing networks," *Int. Journal of Systems Science*, vol. 19, pp. 967-977, August 1988.

[2] I. F. Akyildiz, "Performance Analysis of Computer Communication Networks with Local and Global flow Control ," *in Proceedings IEEE INFOCOM 88*,,pp. 401-410, 1988.

[3] K. Bharath-Kumar and J. M. Jeffe, "A new approach to performance oriented Flow control ," *IEEE Trans. on Commun.*, vol. COM-29, pp 427-435, April 1981.

[4] R. R. Boorstyn and A. Livn, "A Technique for daptive Routing in Networks," *IEEE Trans. on Commun.*, Vol. COM-29, pp 474-480, April 1981.

[5] A. Chatterjee N. D. Georganas and P. K. Verma, "Analysis of a Packet-Switched Network with End-to-End Congestion Control and Random routing," *IEEE Trans. on Commun.*, vol. COM-25, pp. 1485-1489, Dec. 1977.

[6] W. Chou, A. W. Bragg and A. A. Nilsson, "The need for Adaptive Routing in Chaotic and Unbalanced Traffic Environment," *IEEE Trans. on Commun.*, vol. COM-29, pp 481-490, April 1981.

[7] D. W. Davies, "The Control of Congestion in Packet-Switching Networks," *IEEE Trans. on Commun.*, vol. COM-20, pp 546-550, June 1972.

[8] G. Foschini and B. Gopinath. "Sharing Memory Optimally ," *IEEE Trans. on Commun.*, vol. COM-31. pp. 352-359. March 1983.

[9] R. G. Gallager and M. Golestaani."Flow Control and Routing algorithms for data Networks ," *in Proc. 5th Int. Conf. Comput. Commun. (ICCC 80)*. Atlanta, GA, Oct. 1980

[10] N. D. Georganas. "Modeling and Analysis of Message Switched Computer Communication Networks with Multilevel Flow Control." *Computer Networks*, vol. 4, pp. 285-294. 1980.

[11] A. Giessler et al., "Free Buffer Allocation An investigation by simulatiopn." *Computer Networks*, vol. 2. pp. 191-208. 1978.

[12] M. I. Irland, "Buffer Management in a packet switch," *IEEE Trans on Commun.*, vol. COM-26, pp. 328-337, March 1978.

[13] M. Jeffe, Jeffrey, "Bottleneck Flow control ," *IEEE Trans. on Commun.*, vol. COM-29, pp 954-962, July 1981.

[14] F. Kamoun and L. Kleinrock, "Analysis of Shared Finite Storage in a Computer Network Node Enviromnent Under General Conditions," *IEEE Trans. on Commun.*, vol. COM-28, pp. 992-1003, July 1980.

[15] R. E. Khan, W .R. Crowther, " Flow Control in a Resource Sharing computer Network," *IEEE Trans. on Commun.*, vol. COM-20, pp 539-546, June 1972.

[16] L. Kleinrok and M. Gerla. "Flow control: A Comparative Survey," *IEEE Trans. on Commun.*, vol. COM-28, pp. 553-574, April 1980.

[17] L. Kleinrock, and P. Kermani, "Static Flow control in Store-Forward Computer Networks," *IEEE Trans. on Commun.*, vol. COM-28, pp.271- 279, Feb. 1980.

[18] L. Kleinrock, and P. Kermani. "Dynamic Flow control in Store-Forward Computer Networks," *IEEE Trans. on Commun.*, vol. COM-28, pp. 263-271, Feb 1980.

[19] L. Kleinrock, "Queing Theory, Volume 2: Computer Applications ," Wiley, New York, 1976.

[20] S. S. Lam " Store-and-Forward Buffer Requirements in a Packet-Switching Network," *IEEE Trans. on Commun.*, vol. COM-24, pp. 394-403, April 1976.

[21] L. D. Lamontagne, "Investigation of packet loss reduction methods and decentralization for dynamic routing in packet-switching networks," M.S Thesis, Dep. Systems Science, Univ. of Ottawa, Canada , 1990.

[22] G. Latouche, "Exponential Servers Sharing a Finite Storage: Comparison of Space Allocation Policies,"*IEEE Trans. on Commun.*, vol. COM-28, pp. 910-915, June 1980.

[23] K. J. Lee and Y. Lim, "Performance Analysis of the Congestion Control Scheme in signaling System No.7 ,"*in Proceedings IEEE INFOCOM 89*,pp. 691-700, 1989.

[24] K. J. Lee D. Towsley and M. Choi, " Distributed Algotithms for Minimum Delay routing with Constraints in Communication Networks ,"*in Proceedings IEEE INFOCOM 87*,pp. 188-199, 1987.

[25] C. Lemieux, "Theory of Flow control in Shared Networks and Its Application to the Canadian Telephone Network ,"*IEEE Trans. on Commun.*, vol. COM-29, pp. 399-412, April 1981.

[26] K. H. Muralidhar and M. K. Sundareshan, "Combined Routing and Flow Control in Computer Communication Networks: A Tow-Level Adaptive scheme,"*IEEE Trans. on Automatic Control*, vol. AC-32, pp. 15-25 , January 1987.

[27] K. H. Muralidhar and M. K. Sundareshan, "Adaptive Routing and Flow Control in Communication Networks: a hierarchical scheme for Multiobjective Optimization ,"*in Proceedings IEEE INFOCOM 84*,pp. 299-308, 1984.

[28] M. C. Pennotti and M. Schwartz. "Congestion Control in Store and Forward Tandem Links." *IEEE Trans. on Commun.*, vol. COM-23, pp. 1434-1443, Dec. 1975.

L. Pouzin, "Methods, Tools and Observations on Flow Control on Packet -Swithed Data networks ." *IEEE Trans. on Commun.*, vol. COM-29, pp. 413-426, April 1981.

[29] P. E. Sarachic, "An Effective Local Dynamic Strategy to Clear Congested Multidestination Networks ." *IEEE Trans. on Automatic Control*, vol. AC-27, pp. 510-513 , April 1982.

[30] M. Schwartzs, "Computer-Communication Network Design and Analysis." Englewood Cliffs, N. J., Prentice-Hall, 1977.

[31] M. Schwartzs, "Telecommunication Networks: Protocols, Modeling and Analysis," Reading, MA: Addison-Wesley, 1987

[32] M. Schwartzs, "Routing and Flow Fontrol in Data Networks," New concepts in Mult-user Communications, J, K Skwirzinsky, Ed., NATO Advanced Study Inst. Series (Series E: Applied Science 43) 1981.

[33] A. Segall, "The Modeling of Adaptive Routing in Data-Communication Networks," *IEEE Trans. on Automatic Control*, vol. AC-25, pp. 859-95, January 1977.

[34] A. Segall and F. H. Moss, "An Optimal Control Approach to Dynamic Routing in Networks ," *IEEE Trans. on Automatic Control*, vol. AC-27, pp. 329-339 , April 1982.

[35] A. Thareja and A. Agrawala, "On the Design of Optimal Policy for Sharing Finite buffers ," *IEEE Trans. on Commun.*, vol. COM-32, pp. 737-740, June 1984.

[36] A. Thareja and A. Agrawala, "Impact of Buffer allocation Policies on delay in message switching Networks ." *in Proceedings IEEE INFOCOM 83*,pp.

436–442. 1983.

[37] D. Tipper and M. K. Sundareshan, "Adaptive Policies for Optimal Buffer Management in Dynamic Load Environments," *IEEE Trans. on Commun.*, vol. COM-23, pp. 535-544, 1988.

Appendix A

programs

A.1 The Program for model [1]

```
C      ***************************************************
C AVD : AVERAGE DELAY
C B   : BUFFER SIZE
C B1  : B-Q-EA (THE UPPER BOUND OF INTERNAL INPUT )
C C   : CHANNEL CAPACITY
C EA  : EXTERNAL ARRIVAL RATE
C RO  : CHENNAL ERROR RATE
C ER  : ERROR OF THE CHANNEL (ER = RO * X)
C MM  : NUMBER OF CONSTRAINTS (18)
C Q   : QUEUE LENGTH
C QL  : NUMBER OF PACKETS LOST AT EACH BUFFER
C QLOST :TOTAL NUMBER OF PACKETS LOST DURING THE ENTIRE SIMULATION
C QTT : TOTAL QUEUE LENGTH
C S   : TIME COUNTER
C THR : AVERAGE THROUGHPUT
C TARR: TOTAL EXTERNAL ARRIVALS
C      ***************************************************************
C
```

```
      INTEGER   K,II,NITER
      REAL      AVD,THR,QLOST,QTT,TARR,A1(10),A2(10),A3(10),A4(10)
     &          ,A5(10)
      DOUBLE PRECISION DSEED
      NITER=10
      DO 700 K=1,70
         DO 600 II=1,NITER
            CALL STSEED(II,DSEED)
            CALL SIM (K,DSSED,AVD,THR,QLOST,QTT,TARR)
            A1(II)=AVD
            A2(II)=THR
            A3(II)=QLOST
            A4(II)=QTT
            A5(II)=TARR
          WRITE(4,50) A1(II),A2(II),A3(II),A4(II),A5(II)
C         WRITE(4,50) K, AVD,THR,QLOST,QTT,TARR
50        FORMAT(2X,I3,2F9.3,3F12.1)
600       CONTINUE
          CALL COMP(K,NITER,A1,A2,A3,A4,A5)
700   CONTINUE
      STOP
      END
C-----------------------------------------
      SUBROUTINE STSEED (II,DSEED)
C-----------------------------------------
      INTEGER   II
      DOUBLE PRECISION NSEED,DSEED
C
      NSEED(1)=123457.0
      NSEED(2)=68374381.0
      NSEED(3)=964393174.0
```

```
      NSEED(4)=121742663.0
      NSEED(5)=61433579.0
      NSEED(6)=11572403.0
      NSEED(7)=15726055.0
      NSEED(8)=48108509.0
      NSEED(9)=999792099.0
      NSEED(10)=477424540.0
C
      DSEED=NSEED(II)
C
      RETURN
      END
C-------------------------------------------------------
      SUBROUTINE SIM (K,DSEED,AVD,THR,QLOST,QTT,TARR)
C-------------------------------------------------------
      INTEGER      N,NR,IR(100),IER,K,S,R,IX,U122,LDA,MM,NVAR,I,KK,
     &        U123,U132,U133,U211,U213,U231,U233,U311,U312,U321,U322
      PARAMETER    (MM = 18, NVAR = 12, LDA = MM)
      INTEGER      IRTYPE(MM), ISEED,DSEED,SSEED
      REAL        XLB(NVAR),XUB(NVAR),T(12),A(LDA,NVAR),BB(MM),
     &            CC(NVAR),OBJ,XSOL(NVAR),DSOL(NVAR)
     &           ,RS(3,3,900), EA(3,3,900),TARR,QL(3,3),QIL
     &           ,Q(3,3),QLOST,QTT,B(3,3),C(3,3),BS(3,3),B1(3,3)
     &           ,RLAM ,ER(NVAR),RO,ALF(3,3),QQT(3,3,900)
     &           ,Q1,Q2,Q3,Q4,Q5,Q6,AVD,THR,EER(3,3)
C **  IMSL SUBROUTINES **
      EXTERNAL     DLPRS, RNSET , RNUND, GGPOS,RNGET
      DATA XLB/12*0.0/
      DATA XUB/12*50./
      DATA IRTYPE/18*1/
C
```

```
C ***************************************************************************
C                          INITIALIZATION
C ***************************************************************************
      TARR  = 0.0
      QLOST = 0.0
      QTT   = 0.0
      QIL   = 0.0
      RO = .001
      ISEED = INT(DSEED)
C
      DO 1 I=1,3
      DO 1 J=1,3
      IF(I.EQ.J) GO TO 1
      B(I,J)  =100.
      C(I,J)  =50.0
      Q(I,J)  =50.0
      QQT(I,J,1) = 50.0
      QL(I,J) =0.0
      EA(I,J,1)=20.
C ----  EXTERNAL ARRIVAL AT T=0  ----
C
      RS(I,J,1) = EA(I,J,1)
1     CONTINUE
C
C---- THE LOOP STARTS FROM S=2 ------
      S=2
1000  CONTINUE
C
C------GENERATE EXTERNAL ARRIVAL RATE FOR THE UNBALANCED CASE-----
C     NR=7
C     CALL RNSET (ISEED)
```

```
C       CALL RNUND (NR, K, IR)
C       CALL RNGET (ISEED)
C       EA(1,2,S) = IR(2)
C       EA(1,3,S) = IR(3)
C       EA(2,1,S) = IR(4)
C       EA(2,3,S) = IR(5)
C       EA(3,1,S) = IR(6)
C       EA(3,2,S) = IR(7)
C ------------- EXTERNAL ARRIVAL RATE IN THE BALANCED CASE = K ----
        EA(1,2,S) = K
        EA(1,3,S) = K
        EA(2,1,S) = K
        EA(2,3,S) = K
        EA(3,1,S) = K
        EA(3,2,S) = K
        NR=1
        DSEED=DSEED + S
        DO 3 I=1,3
        DO 3 J=1,3
        IF(I.EQ.J) GO TO 3
        RLAM = EA(I,J,S)
        IF(RLAM.LE.0.0) GO TO 2
        CALL GGPOS(RLAM,DSEED,NR,IR,IER)
        RS(I,J,S) = IR(1)
C -------TOTAL ARRIVAL  ----------------
        TARR = TARR + RS(I,J,S)
        GOTO 3
2       RS(I,J,S) = 0
3       CONTINUE
C*****************************************************
C ** B1=B-Q-EA IF B-Q > EA AND B1 = 0 IF B-Q < EA **
```

```
C*************************************************
C
      DO 13 I=1,3
      DO 13 J=1,3
      IF(I.EQ.J) GOTO 13
      BS(I,J)=B(I,J)-Q(I,J)-EA(I,J,S)
        IF(BS(I,J).LT.0.0) THEN
        EER(I,J)=B(I,J)-Q(I,J)
        ELSE
        EER(I,J)=EA(I,J,S)
        ENDIF
      B1(I,J) = B(I,J) - Q(I,J) - EER(I,J)
C     ALF(I,J) = 1 + (Q(I,J)+EA(I,J,S))/B(I,J)
      ALF(I,J) = 1 - EER(I,J)/B(I,J)
13    CONTINUE
C
C ******  THE CONSTRAINTS *********
      BB(1) = C(1,2)
      BB(2) = C(1,3)
      BB(3) = C(2,1)
      BB(4) = C(2,3)
      BB(5) = C(3,1)
      BB(6) = C(3,2)
      BB(7) = B1(1,2)
      BB(8) = B1(1,3)
      BB(9) = B1(2,1)
      BB(10) = B1(2,3)
      BB(11) = B1(3,1)
      BB(12) = B1(3,2)
      BB(13) = Q(1,2) + EA(1,2,S)
      BB(14) = Q(1,3) + EA(1,3,S)
```

```
      BB(15) = Q(2,1) + EA(2,1,S)
      BB(16) = Q(2,3) + EA(2,3,S)
      BB(17) = Q(3,1) + EA(3,1,S)
      BB(18) = Q(3,2) + EA(3,2,S)
C
      DO 17 I = 1,MM
      DO 17 J = 1,NVAR
      A(I,J) = 0.0
17    CONTINUE
C
      DO 18 I = 1,6
      A(I,I*2) = 1.0
      A(I,I*2-1) = 1.0
18    CONTINUE
      DO 19 I = 13,16
      A(I,I-10) = 1.0
      A(I+2,I-6) = 1.0
19    CONTINUE
      DO 20 I = 13,14
      A(I,I-12) = 1.0
      A(I+4,I-2) = 1.0
20    CONTINUE
      A(7,10) = 1.0
      A(8,6) = 1.0
      A(9,11) = 1.0
      A(10,2) = 1.0
      A(11,7) = 1.0
      A(12,3) = 1.0
      A(13,10) = -1.0
      A(14,6) = -1.0
      A(15,11) = -1.0
```

```
      A(16,2) = -1.0
      A(17,7) = -1.0
      A(18,3) = -1.0
C     WRITE(9,111) (BB(J), J=7,18)
C     DO 55 I =1,18
C     IF(S.GT.2) GOTO 55
C     WRITE(9,111) (A(I,J), J=1,12)
C111   FORMAT(3X,12F6.2)
C55    CONTINUE
C
C ** OBJECTIVE FUNCTION  ***********
C
      CC(1) = -ALF(1,2)*(1-RO)
      CC(2) = (ALF(2,3)-ALF(1,3))*(1-RO)
      CC(3) = (ALF(3,2)-ALF(1,2))*(1-RO)
      CC(4) = -ALF(1,3)*(1-RO)
      CC(5) = -ALF(2,1)*(1-RO)
      CC(6) = (ALF(1,3)-ALF(2,3))*(1-RO)
      CC(7) = (ALF(3,1)-ALF(2,1))*(1-RO)
      CC(8) = -ALF(2,3)*(1-RO)
      CC(9) = -ALF(3,1)*(1-RO)
      CC(10) = (ALF(1,2)-ALF(3,2))*(1-RO)
      CC(11) = (ALF(2,1)-ALF(3,1))*(1-RO)
      CC(12) = -ALF(3,2)*(1-RO)
C
      DO 15 I=7,12
      IF(BB(I).EQ.0.0) GOTO 14
      IRTYPE(I)=1
      GOTO 15
14    IRTYPE(I)=0
15    CONTINUE
```

```
C  ------  SUBROUTINE TO MINIMIZE THE OBJECTIVE FUNCTION -------------
C
      CALL DLPRS(MM,NVAR,A,LDA,0.,BB,CC,IRTYPE,XLB,XUB,OBJ,XSOL,DSOL)
C
C *** THE CONTROL VARIABLES   **********
      T(1)=XSOL(1)
      T(2)=XSOL(2)
      T(3)=XSOL(3)
      T(4)=XSOL(4)
      T(5)=XSOL(5)
      T(6)=XSOL(6)
      T(7)=XSOL(7)
      T(8)=XSOL(8)
      T(9)=XSOL(9)
      T(10)=XSOL(10)
      T(11)=XSOL(11)
      T(12)=XSOL(12)
C
      DO 4000 I=1,12
C     IF(T(I).LT.51.0) GOTO 4000
      IF(T(I).LT.26.0) GOTO 4000
      T(I)=25.
C     T(I)=50.
4000  CONTINUE
C
C -------------------GENERATE CHANNEL ERRORS ------------
      NR=1
      DSEED=DSEED + S
      DO 4 I=1,12
      RLAM = RO*T(I)
      IF(RLAM.LE.0.0) GOTO 5
```

```
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      ER(I) = IR(1)
      T(I)= T(I)-ER(I)
      GOTO 4
5     ER(I) = 0.0
      T(I)=T(I)
4     CONTINUE
C
C -----------------------------
C THE ROUTING POLICY
C-----------------------------
200   Q1= - (Q(1,2)+T(10)+ RS(1,2,S) -T(1)-T(3))
      Q2= - (Q(1,3)+T(6) + RS(1,3,S) -T(2)-T(4))
      Q3= - (Q(2,1)+T(11)+ RS(2,1,S) -T(5)-T(7))
      Q4= - (Q(2,3)+T(2) + RS(2,3,S) -T(8)-T(6))
      Q5= - (Q(3,1)+T(7) + RS(3,1,S) -T(11)-T(9))
      Q6= - (Q(3,2)+T(3) + RS(3,2,S) -T(10)-T(12))
      IF(Q1.LE.1.0E-03) GOTO 181
      T(1) = T(1)-Q1*T(1)/(T(1)+T(3))
      T(3) = Q(1,2)+T(10)+RS(1,2,S)-T(1)
      IX=1
      IY=3
      GOTO 500
C
181   IF(Q2.LE.1.0E-03) GOTO 182
      T(4) = T(4)-Q2*T(4)/(T(4)+T(2))
      T(2) = Q(1,3)+T(6)+RS(1,3,S)-T(4)
      IX=4
      IY=2
      GOTO 500
182   IF(Q3.LE.1.0E-03) GOTO 183
```

```
      T(5) = T(5)-Q3*T(5)/(T(5)+T(7))
      T(7) = Q(2,1)+T(11)+RS(2,1,S)-T(5)
      IX=5
      IY=7
      GOTO 500
183   IF(Q4.LE.1.0E-03) GOTO 184
      T(8) = T(8)-Q4*T(8)/(T(8)+T(6))
      T(6) = Q(2,3)+T(2)+RS(2,3,S)-T(8)
      IX=8
      IY=6
      GOTO 500
184   IF(Q5.LE.1.0E-03) GOTO 185
      T(9) = T(9)-Q5*T(9)/(T(9)+T(11))
      T(11) = Q(3,1)+T(7)+RS(3,1,S)-T(9)
      IX=9
      IY=11
      GOTO 500
185   IF(Q6.LE.1.0E-03) GOTO 186
      T(12) = T(12)-Q6*T(12)/(T(12)+T(10))
      T(10) = Q(3,2)+T(3)+RS(3,2,S)-T(12)
      IX=12
      IY=10
500   NR=1
      RLAM = T(IX)*RO
      IF (RLAM) 503,503,502
502   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IX) = T(IX)-IR(1)
      GOTO 504
503   T(IX) = T(IX)
504   RLAM=T(IY)*RO
      IF (RLAM) 506,506,505
```

```
505     CALL GGPOS(RLAM,DSEED,NR,IR,IER)
        T(IY) = T(IY)-IR(1)
        GOTO 200
506     T(IY) = T(IY)
        GOTO 200
186     CONTINUE
C-----------------------------------------------
C   UPDATE THE QUEUES
C-----------------------------------------------
C
        U122=T(1)
        U123=T(2)
        U132=T(3)
        U133=T(4)
        U211=T(5)
        U213=T(6)
        U231=T(7)
        U233=T(8)
        U311=T(9)
        U312=T(10)
        U321=T(11)
        U322=T(12)
C
        IF(T(1)-U122.GT.0.5) U122 = U122 + 1
        IF(T(2)-U123.GT.0.5) U123 = U123 + 1
        IF(T(3)-U132.GT.0.5) U132 = U132 + 1
        IF(T(4)-U133.GT.0.5) U133 = U133 + 1
        IF(T(5)-U211.GT.0.5) U211 = U211 + 1
        IF(T(6)-U213.GT.0.5) U213 = U213 + 1
        IF(T(7)-U231.GT.0.5) U231 = U231 + 1
        IF(T(8)-U233.GT.0.5) U233 = U233 + 1
```

```
      IF(T(9)-U311.GT.0.5) U311 = U311 + 1
      IF(T(10)-U312.GT.0.5) U312 = U312 + 1
      IF(T(11)-U321.GT.0.5) U321 = U321 + 1
      IF(T(12)-U322.GT.0.5) U322 = U322 + 1
C
      Q(1,2) = Q(1,2)+RS(1,2,S)+U312-U122-U132
      Q(1,3) = Q(1,3)+RS(1,3,S)+U213-U123-U133
      Q(2,1) = Q(2,1)+RS(2,1,S)+U321-U231-U211
      Q(2,3) = Q(2,3)+RS(2.3,S)+U123-U233-U213
      Q(3,1) = Q(3,1)+RS(3,1,S)+U231-U321-U311
      Q(3,2) = Q(3,2)+RS(3,2,S)+U132-U312-U322
C
      DO 10 I=1,3
      DO 10 J=1,3
      IF(I.EQ.J) GO TO 10
      QT(I,J,S) = Q(I,J)
C-------- QLOST = BUFFER - QUEUE ---------
      IF(Q(I,J).GT.B(I,J)) GO TO 8
      QL(I,J) = 0.0
      GO TO 9
8     QL(I,J) = Q(I,J) - B(I,J)
9     Q(I,J) = Q(I,J) - QL(I,J)
      QQT(I,J,S) = Q(I,J)
      QLOST = QLOST + QL(I,J)
C --------TOTAL QUEUE --------------
      QTT = QTT + Q(I,J)
10    CONTINUE
      S = S + 1
      IF(S.LE.61) GOTO 1000
C -----  END OF THE LOOP -----------
C ------ COMPUTE THE DELAY AND THE THROUGHPUT-----
```

```
      DO 300 I=1,3
      DO 300 J=1,3
      IF(I.EQ.J) GO TO 300
      QIL=QIL+QQT(I,J,1)-QQT(I,J,61)
C     WRITE(9,*) QIL
300   CONTINUE
      AVD=QTT/(TARR-QLOST+QIL)
      THR=(QIL+TARR-QLOST)/60.
      RETURN
      END
C
C-----------------------------------------------
      SUBROUTINE COMP(K,NITER,A1,A2,A3,A4,A5)
C-----------------------------------------------
      INTEGER     I,K,NITER
      REAL        A1(10),A2(10),A3(10),A4(10),A5(10),M1,M2,M3,M4,M5
     &            ,V1,V2,V3,V4,SD1,SD2,SD3,SD4,PCT1,PCT2,PCT3,PCT4
     &            ,UB1,UB2,UB3,UB4,LB1,LB2,LB3,LB4
C
      M1=0.0
      M2=0.0
      M3=0.0
      M4=0.0
      M5=0.0
      DO 1 I=1,NITER
         M1= M1+A1(I)
         M2= M2+A2(I)
         M3= M3+A3(I)
         M4= M4+A4(I)
         M5= M5+A5(I)
1     CONTINUE
```

```
      M1=M1/NITER
      M2=M2/NITER
      M3=M3/NITER
      M4=M4/NITER
      M5=M5/NITER
C
      V1=0.0
      V2=0.0
      V3=0.0
      V4=0.0
      DO 2 I=1,NITER
         V1= V1+((A1(I)-M1)**2)
         V2= V2+((A2(I)-M2)**2)
         V3= V3+((A3(I)-M3)**2)
         V4= V4+((A4(I)-M4)**2)
2     CONTINUE
      V1=V1/(NITER-1)
      V2=V2/(NITER-1)
      V3=V3/(NITER-1)
      V4=V4/(NITER-1)
C
      SD1=SQRT(V1/FLOAT(NITER))
      SD2=SQRT(V2/FLOAT(NITER))
      SD3=SQRT(V3/FLOAT(NITER))
      SD4=SQRT(V4/FLOAT(NITER))
C
      PCT1=SD1*2.26*100/M1
      PCT2=SD2*2.26*100/M2
      PCT3=SD3*2.26*100/M3
      PCT4=SD4*2.26*100/M4
C
```

```
      UB1=M1 + SD1*2.26
      UB2=M2 + SD2*2.26
      UB3=M3 + SD3*2.26
      UB4=M4 + SD4*2.26
C
      LB1=M1 - SD1*2.26
      LB2=M2 - SD2*2.26
      LB3=M3 - SD3*2.26
      LB4=M4 - SD4*2.26
      WRITE(9,60) K,M1,M2,M3,M4,M5
C
      WRITE(4,*) ('PERCENT STANDARD DIVIATION')
      WRITE(4,66) K,PCT1,PCT2,PCT3,PCT4
C
      WRITE(4,*) ('CONFID INTERVAL')
      WRITE(4,66) K,UB1,UB2,UB3,UB4
      WRITE(4,66) K,LB1,LB2,LB3,LB4
C
60    FORMAT(2X,I3,F8.4,F8.2,3F12.1)
66    FORMAT(2X,I3,4F12.4)
      RETURN
      END
```

A.2 The Program for the CS model

```
      INTEGER   K,II,NITER
      REAL      AVD,THR,QLOST,QTT,TARR,A1(10),A2(10),A3(10),A4(10)
     &          ,A5(10)
      DOUBLE PRECISION DSEED
      NITER=10
      DO 700 K=1,70
         DO 600 II=1,NITER
            CALL STSEED(II,DSEED)
            CALL SIM (K,DSSED,AVD,THR,QLOST,QTT,TARR)
            A1(II)=AVD
            A2(II)=THR
            A3(II)=QLOST
            A4(II)=QTT
            A5(II)=TARR
           WRITE(4,50) A1(II),A2(II),A3(II),A4(II),A5(II)
C          WRITE(4,50) K, AVD,THR,QLOST,QTT,TARR
50         FORMAT(2X,I3,2F9.3,3F12.1)
600        CONTINUE
           CALL COMP(K,NITER,A1,A2,A3,A4,A5)
700   CONTINUE
      STOP
      END
C-----------------------------------------------
      SUBROUTINE STSEED (II,DSEED)
C-----------------------------------------------
      INTEGER   II
      DOUBLE PRECISION NSEED,DSEED
C
      NSEED(1)=123457.0
```

```
      NSEED(2)=68374381.0
      NSEED(3)=964393174.0
      NSEED(4)=121742663.0
      NSEED(5)=61433579.0
      NSEED(6)=11572403.0
      NSEED(7)=15726055.0
      NSEED(8)=48108509.0
      NSEED(9)=999792099.0
      NSEED(10)=477424540.0
C
      DSEED=NSEED(II)
C
      RETURN
      END
C-------------------------------------------------------------
      SUBROUTINE SIM (K,DSEED,AVD,THR,QLOST,QTT,TARR)
C-------------------------------------------------------------
      INTEGER      N,NR,IR(100),IER,K,S,R,IX,LDA,MM,NVAR,I,KK
     &            ,U122,U123,U132,U133,U211,U213,U231,U233,
     &             U311,U312,U321,U322
      PARAMETER    (MM = 15, NVAR = 12, LDA = MM)
      INTEGER      IRTYPE(MM), ISEED,IISEED,M,NN,L
      REAL         XLB(NVAR),XUB(NVAR),T(12),A(LDA,NVAR),BB(MM),Y(3)
     &            ,CC(NVAR),OBJ,XSOL(NVAR),DSOL(NVAR),BF(3,3),QQ(3)
     &            ,RS(3,3,900),EA(3,3,900),TARR,QL(3),QIL,RR(3)
     &            ,Q(3,3),QLOST,QTT,B(3),C(3,3),BS(3),B1(3),B2(3,3)
     &            ,ERT(NVAR),RLAM,ER(NVAR),ALF(3,3),BFS(3,3),Q1,Q2,Q3
     &            ,Q4,Q5,Q6,AVD,THR,EER(3,3),QQT(3,900),RO,QLL(3,3)
C
      DOUBLE PRECISION DSEED
C  ***  IMSL SUBROUTINES    **********
```

```
      EXTERNAL     RNUND, DLPRS, RNHYP,GGPOS,RNGET,RNSET
      DATA XLB/12*0./
      DATA XUB/12*50./
      DATA IRTYPE/15*1/
C
C ***************************************************************************
C                          INITIALIZATION
C ***************************************************************************
      TARR  = 0.0
      QLOST = 0.0
      QTT   = 0.0
      QIL   = 0.0
      RO = .001
      IISEED = 123457
      ISEED = INT(DSEED)
C
      DO 1 I=1,3
      B(I) = 200.
      QQ(I) = 100.
      QL(I) = 0.0
      DO 1 J=1,3
      IF(I.EQ.J) GO TO 1
      C(I,J)  =50.0
      BF(I,J)  =B(I)
      Q(I,J)  =50.0
      QQT(I,1)  =100.
C ----  EXTERNAL ARRIVAL AT T=0 ----------
      EA(I,J,1)=20.
      RS(I,J,1) = EA(I,J,1)
1     CONTINUE
```

```
      S=2
1000  CONTINUE
C--------GENERATE EXTERNAL ARRIVAL RATE FOR THE UNBALANCED CASE----
C     NR=7
C     CALL RNSET (ISEED)
C     CALL RNUND (NR, K, IR)
C     CALL RNGET (ISEED)
C     EA(1,2,S) = IR(2)
C     EA(1,3,S) = IR(3)
C     EA(2,1,S) = IR(4)
C     EA(2,3,S) = IR(5)
C     EA(3,1,S) = IR(6)
C     EA(3,2,S) = IR(7)
C -------------- EXTERNAL ARRIVAL RATE IN THE BALANCED CASE = K ----
      EA(1,2,S) = K
      EA(1,3,S) = K
      EA(2,1,S) = K
      EA(2,3,S) = K
      EA(3,1,S) = K
      EA(3,2,S) = K
      NR=1
      DSEED=DSEED + S
      DO 3 I=1,3
      DO 3 J=1,3
      IF(I.EQ.J) GO TO 3
      RLAM = EA(I,J,S)
      IF(RLAM.LE.0.0) GO TO 2
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      RS(I,J,S) = IR(1)
C -------TOTAL ARRIVAL  -----------------
      TARR = TARR + RS(I,J,S)
```

```
      GOTO 3
2     RS(I,J,S) = 0
3     CONTINUE
C*****************************************************
C ** B1=B-Q-EA IF B-Q > EA AND B1 = 0 IF B-Q < EA **
C*****************************************************
C
      DO 13 I=1,3
      BS(I) = B(I) - QQ(I)
      DO 13 J=1,3
      IF(I.EQ.J) GOTO 13
      BS(I) = BS(I)-EA(I,J,S)
      B1(I) = MAX (0.,BS(I))
      BFS(I,J) = BF(I,J) - Q(I,J) -EA(I,J,S)
      B2(I,J) = MAX (0.,BFS(I,J))
      ALF(I,J) = 1 + (Q(I,J) + EA(I,J,S))/BF(I,J)
13    CONTINUE
C
      DO 15 I=1,3
      IF(B1(I).EQ.0.0) GOTO 14
      IRTYPE(12+I) = 1
      GOTO 15
14    IRTYPE(I+12) = 0
15    CONTINUE
C  ***  THE CONSTRAINTS   ******
      BB(1) = C(1,2)
      BB(2) = C(1,3)
      BB(3) = C(2,1)
      BB(4) = C(2,3)
      BB(5) = C(3,1)
      BB(6) = C(3,2)
```

```
      BB(7) = Q(1,2) + EA(1,2,S)
      BB(8) = Q(1,3) + EA(1,3,S)
      BB(9) = Q(2,1) + EA(2,1,S)
      BB(10) = Q(2,3)+ EA(2,3,S)
      BB(11) = Q(3,1)+ EA(3,1,S)
      BB(12) = Q(3,2)+ EA(3,2,S)
      BB(13) = B1(1)
      BB(14) = B1(2)
      BB(15) = B1(3)
      DO 17 I = 1,MM
      DO 17 J = 1,NVAR
      A(I,J) = 0.0
17    CONTINUE
C
      DO 18 I = 1,6
      A(I,I*2) = 1.0
      A(I,I*2-1) = 1.0
18    CONTINUE
      DO 19 I = 7,10
      A(I,I-4) = 1.0
      A(I+2,I) = 1.0
19    CONTINUE
      DO 20 I = 7,8
      A(I,I-6) = 1.0
      A(I+4,I+4) = 1.0
      A(I+7,I-5) = 1.0
      A(I+6,I+3) = 1.0
20    CONTINUE
      A(7,10) = -1.0
      A(8,6) = -1.0
      A(9,11) = -1.0
```

```
      A(10,2) = -1.0
      A(11,7) = -1.0
      A(12,3) = -1.0
      A(13,6) = 1.0
      A(15,7) = 1.0
C     DO 55 I =1,MM
C     IF(S.LT.42) GOTO 55
C     WRITE(9,*) Q(2,1), EA(2,1,S)
C     WRITE(9,111) (A(I,J), J=1,12)
C111   FORMAT(3X,12F6.2)
C     WRITE(9,*) I,BB(I),IRTYPE(I)
C55    CONTINUE
C  ***  THE OBJECTIVE FUNCTION  ***********
C
      CC(1) = -ALF(1,2)*(1-RO)
      CC(2) = (ALF(2,3)-ALF(1,3))*(1-RO)
      CC(3) = (ALF(3,2)-ALF(1,2))*(1-RO)
      CC(4) = -ALF(1,3)*(1-RO)
      CC(5) = -ALF(2,1)*(1-RO)
      CC(6) = (ALF(1,3)-ALF(2,3))*(1-RO)
      CC(7) = (ALF(3,1)-ALF(2,1))*(1-RO)
      CC(8) = -ALF(2,3)*(1-RO)
      CC(9) = -ALF(3,1)*(1-RO)
      CC(10) = (ALF(1,2)-ALF(3,2))*(1-RO)
      CC(11) = (ALF(2,1)-ALF(3,1))*(1-RO)
      CC(12) = -ALF(3,2)*(1-RO)
C  ------  SUBROUTINE TO MINIMIZE THE OBJECTIVE FUNCTION ----------
C
      CALL DLPRS(MM,NVAR,A,LDA,BB,BB,CC,IRTYPE,XLB,XUB,OBJ,XSOL,DSOL)
C
      T(1)=XSOL(1)
```

```
      T(2)=XSOL(2)
      T(3)=XSOL(3)
      T(4)=XSOL(4)
      T(5)=XSOL(5)
      T(6)=XSOL(6)
      T(7)=XSOL(7)
      T(8)=XSOL(8)
      T(9)=XSOL(9)
      T(10)=XSOL(10)
      T(11)=XSOL(11)
      T(12)=XSOL(12)
C
      DO 4000 I=1,12
      IF(T(I).LT.51.) GOTO 4000
      T(I)=50.
4000  CONTINUE
C -------------------GENERATE CHANNEL ERRORS ------------
      NR=1
      DSEED=DSEED + S
      DO 4 I=1,12
      RLAM = RO*T(I)
      IF(RLAM.LE.0.0) GOTO 5
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      ER(I) = IR(1)
      T(I)= T(I)-ER(I)
      GOTO 4
5     ER(I) = 0.0
      T(I)=T(I)
4     CONTINUE
C
C ----------------------------
```

```
C THE ROUTING POLICY
C------------------------------
200     Q1= - (Q(1,2)+T(10)+ RS(1,2,S) -T(1)-T(3))
        Q2= - (Q(1,3)+T(6) + RS(1,3,S) -T(2)-T(4))
        Q3= - (Q(2,1)+T(11)+ RS(2,1,S) -T(5)-T(7))
        Q4= - (Q(2,3)+T(2) + RS(2,3,S) -T(8)-T(6))
        Q5= - (Q(3,1)+T(7) + RS(3,1,S) -T(11)-T(9))
        Q6= - (Q(3,2)+T(3) + RS(3,2,S) -T(10)-T(12))
        IF(Q1.LE.1.0E-03) GOTO 181
        T(1) = T(1)-Q1*T(1)/(T(1)+T(3))
        T(3) = Q(1,2)+T(10)+RS(1,2,S)-T(1)
        IX=1
        IY=3
        GOTO 500
C
181     IF(Q2.LE.1.0E-03) GOTO 182
        T(4) = T(4)-Q2*T(4)/(T(4)+T(2))
        T(2) = Q(1,3)+T(6)+RS(1,3,S)-T(4)
        IX=4
        IY=2
        GOTO 500
182     IF(Q3.LE.1.0E-03) GOTO 183
        T(5) = T(5)-Q3*T(5)/(T(5)+T(7))
        T(7) = Q(2,1)+T(11)+RS(2,1,S)-T(5)
        IX=5
        IY=7
        GOTO 500
183     IF(Q4.LE.1.0E-03) GOTO 184
        T(8) = T(8)-Q4*T(8)/(T(8)+T(6))
        T(6) = Q(2,3)+T(2)+RS(2,3,S)-T(8)
        IX=8
```

```
      IY=6
      GOTO 500
184   IF(Q5.LE.1.0E-03) GOTO 185
      T(9) = T(9)-Q5*T(9)/(T(9)+T(11))
      T(11) = Q(3,1)+T(7)+RS(3,1,S)-T(9)
      IX=9
      IY=11
      GOTO 500
185   IF(Q6.LE.1.0E-03) GOTO 186
      T(12) = T(12)-Q6*T(12)/(T(12)+T(10))
      T(10) = Q(3,2)+T(3)+RS(3,2,S)-T(12)
      IX=12
      IY=10
500   NR=1
      RLAM = T(IX)*RO
      IF (RLAM) 503,503,502
502   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IX) = T(IX)-IR(1)
      GOTO 504
503   T(IX) = T(IX)
504   RLAM=T(IY)*RO
      IF (RLAM) 506,506,505
505   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IY) = T(IY)-IR(1)
      GOTO 200
506   T(IY) = T(IY)
      GOTO 200
186   CONTINUE
C-----------------------------------------------
C   UPDATE THE QUEUES
C-----------------------------------------------
```

```
C
      U122=T(1)
      U123=T(2)
      U132=T(3)
      U133=T(4)
      U211=T(5)
      U213=T(6)
      U231=T(7)
      U233=T(8)
      U311=T(9)
      U312=T(10)
      U321=T(11)
      U322=T(12)
C
      IF(T(1)-U122.GT.0.5) U122 = U122 + 1
      IF(T(2)-U123.GT.0.5) U123 = U123 + 1
      IF(T(3)-U132.GT.0.5) U132 = U132 + 1
      IF(T(4)-U133.GT.0.5) U133 = U133 + 1
      IF(T(5)-U211.GT.0.5) U211 = U211 + 1
      IF(T(6)-U213.GT.0.5) U213 = U213 + 1
      IF(T(7)-U231.GT.0.5) U231 = U231 + 1
      IF(T(8)-U233.GT.0.5) U233 = U233 + 1
      IF(T(9)-U311.GT.0.5) U311 = U311 + 1
      IF(T(10)-U312.GT.0.5) U312 = U312 + 1
      IF(T(11)-U321.GT.0.5) U321 = U321 + 1
      IF(T(12)-U322.GT.0.5) U322 = U322 + 1
C
      Q(1,2) = Q(1,2)+RS(1,2,S)+U312-U122-U132
      Q(1,3) = Q(1,3)+RS(1,3,S)+U213-U123-U133
      Q(2,1) = Q(2,1)+RS(2,1,S)+U321-U231-U211
      Q(2,3) = Q(2,3)+RS(2,3,S)+U123-U233-U213
```

```
      Q(3,1) = Q(3,1)+RS(3,1,S)+U231-U321-U311
      Q(3,2) = Q(3,2)+RS(3,2,S)+U132-U312-U322
C
      QQ(1) = Q(1,2) + Q(1,3)
      QQ(2) = Q(2,1) + Q(2,3)
      QQ(3) = Q(3,1) + Q(3,2)
C
      RR(1) =RS(1,2,S) + RS(1,3,S)
      RR(2) =RS(2,1,S) + RS(2,3,S)
      RR(3) =RS(3,1,S) + RS(3,2,S)
C ---------------- QLOST = QUEUE - BUFFER -----------
      DO 12 I=1,3
      QL(I) = 0.
      Y(I)  = 0.
      IF(QQ(I)- B(I).LE.0.5) GO TO 10
      QL(I) =  QQ(I) - B(I)
      IF(QL(I).LT.1.0) QL(I)=1.
      DO 7 J=1,3
      IF(I.EQ.J) GO TO 7
      NN = QL(I)
      M  = RS(I,J,S)
      L  =  RR(I)
      CALL RNSET (IISEED)
      CALL RNHYP (1,NN,M,L,IR)
      QLL(I,J) =  IR(1)
      Y(I) = Y(I) + QLL(I,J)
7     CONTINUE
      DO 9 J=1,3
      IF(I.EQ.J) GO TO 9
      IF(Y(I)-QL(I).EQ.0.0) GO TO 8
      QLL(I,J) = QLL(I,J) + (QL(I)-Y(I))*RS(I,J,S)/RR(I)
```

```
8       Q(I,J) = Q(I,J) - QLL(I,J)
9       CONTINUE
10      QQ(I) = QQ(I) - QL(I)
        QQT(I,S) = QQ(I)
        QLOST = QLOST + QL(I)
C  ------TOTAL QUEUE --------------
        QTT = QTT + QQ(I)
12      CONTINUE
        S = S + 1
C       IF(S.LE.61) GOTO 1000
C
C  ----  END OF THE LOOP -----------
        DO 300 I=1,3
        QIL=QIL+QQT(I,1)-QQT(I,61)
300     CONTINUE
        AVD=QTT/(TARR-QLOST+QIL)
        THR=(QIL+TARR-QLOST)/60.
        RETURN
        END
C
C-------------------------------------------------
        SUBROUTINE COMP(K,NITER,A1,A2,A3,A4,A5)
C-------------------------------------------------
        INTEGER     I,K,NITER
        REAL        A1(10),A2(10),A3(10),A4(10),A5(10),M1,M2,M3,M4,M5
     &              ,V1,V2,V3,V4,SD1,SD2,SD3,SD4,PCT1,PCT2,PCT3,PCT4
     &              ,UB1,UB2,UB3,UB4,LB1,LB2,LB3,LB4
C
        M1=0.0
        M2=0.0
        M3=0.0
```

```
      M4=0.0
      M5=0.0
      DO 1 I=1,NITER
         M1= M1+A1(I)
         M2= M2+A2(I)
         M3= M3+A3(I)
         M4= M4+A4(I)
         M5= M5+A5(I)
1     CONTINUE
      M1=M1/NITER
      M2=M2/NITER
      M3=M3/NITER
      M4=M4/NITER
      M5=M5/NITER
C
      V1=0.0
      V2=0.0
      V3=0.0
      V4=0.0
      DO 2 I=1,NITER
         V1= V1+((A1(I)-M1)**2)
         V2= V2+((A2(I)-M2)**2)
         V3= V3+((A3(I)-M3)**2)
         V4= V4+((A4(I)-M4)**2)
2     CONTINUE
      V1=V1/(NITER-1)
      V2=V2/(NITER-1)
      V3=V3/(NITER-1)
      V4=V4/(NITER-1)
C
      SD1=SQRT(V1/FLOAT(NITER))
```

```
      SD2=SQRT(V2/FLOAT(NITER))
      SD3=SQRT(V3/FLOAT(NITER))
      SD4=SQRT(V4/FLOAT(NITER))
C
      PCT1=SD1*2.26*100/M1
      PCT2=SD2*2.26*100/M2
      PCT3=SD3*2.26*100/M3
      PCT4=SD4*2.26*100/M4
C
      UB1=M1 + SD1*2.26
      UB2=M2 + SD2*2.26
      UB3=M3 + SD3*2.26
      UB4=M4 + SD4*2.26
C
      LB1=M1 - SD1*2.26
      LB2=M2 - SD2*2.26
      LB3=M3 - SD3*2.26
      LB4=M4 - SD4*2.26
      WRITE(9,60) K,M1,M2,M3,M4,M5
C
      WRITE(4,*) ('CONFID INTERVAL')
      WRITE(4,66) K,UB1,UB2,UB3,UB4
      WRITE(4,66) K,LB1,LB2,LB3,LB4
C
      WRITE(4,*) ('PERCENT STANDARD DIVIATION')
      WRITE(4,66) K,PCT1,PCT2,PCT3,PCT4
      WRITE(4,*)
C
60    FORMAT(2X,I3,F8.4,F8.2,3F12.1)
66    FORMAT(2X,I3,4F12.4)
      RETURN
```

```
END
```

A.3 The Program for the SQRT model

```
      INTEGER   K,II,NITER
      REAL      AVD,THR,QLOST,QTT,TARR,A1(10),A2(10),A3(10),A4(10)
     &          ,A5(10)
      DOUBLE PRECISION DSEED
      NITER=10
      DO 700 K=1,70
         DO 600 II=1,NITER
            CALL STSEED(II,DSEED)
            CALL SIM (K,DSSED,AVD,THR,QLOST,QTT,TARR)
            A1(II)=AVD
            A2(II)=THR
            A3(II)=QLOST
            A4(II)=QTT
            A5(II)=TARR
          WRITE(4,50) A1(II),A2(II),A3(II),A4(II),A5(II)
C         WRITE(4,50) K, AVD,THR,QLOST,QTT,TARR
50        FORMAT(2X,I3,2F9.3,3F12.1)
600       CONTINUE
          CALL COMP(K,NITER,A1,A2,A3,A4,A5)
700   CONTINUE
      STOP
      END
C-----------------------------------------
      SUBROUTINE STSEED (II,DSEED)
C-----------------------------------------
      INTEGER   II
      DOUBLE PRECISION NSEED,DSEED
C
      NSEED(1)=123457.0
```

```
      NSEED(2)=68374381.0
      NSEED(3)=964393174.0
      NSEED(4)=121742663.0
      NSEED(5)=61433579.0
      NSEED(6)=11572403.0
      NSEED(7)=15726055.0
      NSEED(8)=48108509.0
      NSEED(9)=999792099.0
      NSEED(10)=477424540.0
C
      DSEED=NSEED(II)
C
      RETURN
      END
C-------------------------------------------------------------
      SUBROUTINE SIM (K,DSEED,AVD,THR,QLOST,QTT,TARR)
C-------------------------------------------------------------
      INTEGER      N,NR,IR(100),IER,K,S,R,IX,U122,LDA,MM,NVAR,I,
     &       U123,U132,U133,U211,U213,U231,U233,U311,U312,U321,U322
      PARAMETER    (MM = 21, NVAR = 12, LDA = MM)
      INTEGER      IRTYPE(MM), ISEED,IISEED, M,NN,L
      REAL        XLB(NVAR),XUB(NVAR),T(12),A(LDA,NVAR),BB(MM),Y(3)
     &            ,CC(NVAR),OBJ,XSOL(NVAR),DSOL(NVAR),BF(3,3),QQ(3)
     &           ,RS(3,3,900), EA(3,3,900),TARR,QL(3),QIL,QT(3,3)
     &           ,Q(3,3),QLOST,QTT,B(3),C(3,3),BS(3),B1(3),B2(3,3)
     &           ,RLAM,ER(NVAR),ALF(3,3),BFS(3,3),Q1,Q2,Q3,Q4,Q5,Q6
     &           ,AVD,THR,EER(3,3),QQT(3,900),RO
     &           ,QL1(3), RR(3),QL2(3),QLL1(3,3),QLL2(3,3),QLL(3,3)
      DOUBLE PRECISION DSEED
C ***  IMSL SUBROUTINES  *******
      EXTERNAL     RNSET, RNUND, DLPRS, RNHYP,RNGET,GGPOS
```

```
      DATA XLB/12*0./
      DATA XUB/12*50./
      DATA IRTYPE/21*1/
C
C ***************************************************************************
C                          INITIALIZATION
C ***************************************************************************
      TARR  = 0.0
      QLOST = 0.0
      QTT   = 0.0
      QIL   = 0.0
      RO = .001
      ISEED=INT(DSEED)
      IISEED=123457
C
      DO 1 I=1,3
      B(I) = 200.
      QQ(I) = 100.
      QL(I) =0.0
      DO 1 J=1,3
      IF(I.EQ.J) GO TO 1
      C(I,J)  =50.0
      BF(I,J)  =B(I)/SQRT(2.)
      Q(I,J)  =50.0
      QQT(I,1)  =100.0
C ----  EXTERNAL ARRIVAL AT T=0  ----
      EA(I,J,1) = 20.
      RS(I,J,1) = EA(I,J,1)
1     CONTINUE
C
```

```
      S=2
1000  CONTINUE
C--------GENERATE EXTERNAL ARRIVAL RATE IN THE UNBALANCED CASE-----
C     FR=7
C     CALL RNSET (ISEED)
C     CALL RNUND (NR, K, IR)
C     CALL RNGET (ISEED)
C     EA(1,2,S) = IR(2)
C     EA(1,3,S) = IR(3)
C     EA(2,1,S) = IR(4)
C     EA(2,3,S) = IR(5)
C     EA(3,1,S) = IR(6)
C     EA(3,2,S) = IR(7)
C ---------------- EXTERNAL ARRIVAL RATE IN THE BALANCED CASE = K ----
      EA(1,2,S) = K
      EA(1,3,S) = K
      EA(2,1,S) = K
      EA(2,3,S) = K
      EA(3,1,S) = K
      EA(3,2,S) = K
      NR=1
      DSEED=DSEED + S
      DO 3 I=1,3
      DO 3 J=1,3
      IF(I.EQ.J) GO TO 3
      RLAM = EA(I,J,S)
      IF(RLAM.LE.0.0) GO TO 2
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      RS(I,J,S) = IR(1)
C -------TOTAL ARRIVAL  ----------------
      TARR = TARR + RS(I,J,S)
```

```
      GOTO 3
2     RS(I,J,S) = 0
3     CONTINUE
C ** B1=B-Q-EA IF B-Q > EA AND B1 = 0 IF B-Q < EA **
C*************************************************
C
C     WRITE(9,*) S
      DO 13 I=1,3
      BS(I) = B(I) - QQ(I)
      DO 13 J=1,3
      IF(I.EQ.J) GOTO 13
      BS(I) = BS(I)-EA(I,J,S)
      B1(I) = MAX (0.,BS(I))
      BFS(I,J) = BF(I,J) - Q(I,J) -EA(I,J,S)
      B2(I,J) = MAX (0.,BFS(I,J))
      ALF(I,J) = 1 + (Q(I,J) + EA(I,J,S))/BF(I,J)
13    CONTINUE
C
      DO 15 I=1,3
      IF(B1(I).EQ.0.0) GOTO 14
      IRTYPE(I+18) = 1
      GOTO 15
14    IRTYPE(I+18) = 0
15    CONTINUE
C   ***** CONSTRAINTS *********
      BB(1) = C(1,2)
      BB(2) = C(1,3)
      BB(3) = C(2,1)
      BB(4) = C(2,3)
      BB(5) = C(3,1)
      BB(6) = C(3,2)
```

```
      BB(7) = B2(1,2)
      BB(8) = B2(1,3)
      BB(9) = B2(2,1)
      BB(10) = B2(2,3)
      BB(11) = B2(3,1)
      BB(12) = B2(3,2)
      BB(13) = Q(1,2) + EA(1,2,S)
      BB(14) = Q(1,3) + EA(1,3,S)
      BB(15) = Q(2,1) + EA(2,1,S)
      BB(16) = Q(2,3)+ EA(2,3,S)
      BB(17) = Q(3,1)+ EA(3,1,S)
      BB(18) = Q(3,2)+ EA(3,2,S)
      BB(19) = B1(1)
      BB(20) = B1(2)
      BB(21) = B1(3)
      DO 17 I = 1,MM
      DO 17 J = 1,NVAR
      A(I,J) = 0.0
17    CONTINUE
C
      DO 18 I = 1,6
      A(I,I*2) = 1.0
      A(I,I*2-1) = 1.0
18    CONTINUE
      DO 19 I = 13,16
      A(I,I-10) = 1.0
      A(I+2,I-6) = 1.0
19    CONTINUE
      DO 20 I = 13,14
      A(I,I-12) = 1.0
      A(I+4,I-2) = 1.0
```

```
20      CONTINUE
        DO 21 I = 7,8
        A(I+13,I-5) = 1.0
        A(I+12,I+3) = 1.0
21      CONTINUE
        A(7,10) = 1.0
        A(8,6) = 1.0
        A(9,11) = 1.0
        A(10,2) = 1.0
        A(11,7) = 1.0
        A(12,3) = 1.0
        A(13,10) = -1.0
        A(14,6) = -1.0
        A(15,11) = -1.0
        A(16,2) = -1.0
        A(17,7) = -1.0
        A(18,3) = -1.0
        A(19,6) = 1.0
        A(21,7) = 1.0
C *****   OBJECTIVE FUNCTION   ********
        CC(1) = -ALF(1,2)*(1-RO)
        CC(2) = (ALF(2,3)-ALF(1,3))*(1-RO)
        CC(3) = (ALF(3,2)-ALF(1,2))*(1-RO)
        CC(4) = -ALF(1,3)*(1-RO)
        CC(5) = -ALF(2,1)*(1-RO)
        CC(6) = (ALF(1,3)-ALF(2,3))*(1-RO)
        CC(7) = (ALF(3,1)-ALF(2,1))*(1-RO)
        CC(8) = -ALF(2,3)*(1-RO)
        CC(9) = -ALF(3,1)*(1-RO)
        CC(10) = (ALF(1,2)-ALF(3,2))*(1-RO)
        CC(11) = (ALF(2,1)-ALF(3,1))*(1-RO)
```

```
      CC(12) = -ALF(3,2)*(1-RO)
C
      DO 23 I=7,12
      IF(BB(I).EQ.0.0) GOTO 22
      IRTYPE(I) = 1
      GOTO 23
22    IRTYPE(I) = 0
23    CONTINUE
C     DO 55 I =1,MM
C     IF(S.LT.6) GOTO 55
C     WRITE(9,111) (A(I,J), J=1,12)
C111   FORMAT(3X,12F6.2)
C     WRITE(9,*) BB(I)
C55    CONTINUE
C  ------  SUBROUTINE TO MINIMIZE THE OBJECTIVE FUNCTION --------------
C
      CALL DLPRS(MM,NVAR,A,LDA,0.,BB,CC,IRTYPE,XLB,XUB,OBJ,XSOL,DSOL)
C
      T(1)=XSOL(1)
      T(2)=XSOL(2)
      T(3)=XSOL(3)
      T(4)=XSOL(4)
      T(5)=XSOL(5)
      T(6)=XSOL(6)
      T(7)=XSOL(7)
      T(8)=XSOL(8)
      T(9)=XSOL(9)
      T(10)=XSOL(10)
      T(11)=XSOL(11)
      T(12)=XSOL(12)
C
```

```
      DO 4000 I=1,12
      IF(T(I).LT.51.) GOTO 4000
      T(I)=50.
4000  CONTINUE
C ------------------GENERATE CHANNEL ERRORS -----------
      NR=1
      DSEED=DSEED + S
      DO 4 I=1,12
      RLAM = RO*T(I)
      IF(RLAM.LE.0.0) GOTO 5
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      ER(I) = IR(1)
      T(I)= T(I)-ER(I)
      GOTO 4
5     ER(I) = 0.0
      T(I)=T(I)
4     CONTINUE
C
C ----------------------------
C THE ROUTING POLICY
C----------------------------
200   Q1= - (Q(1,2)+T(10)+ RS(1,2,S) -T(1)-T(3))
      Q2= - (Q(1,3)+T(6) + RS(1,3,S) -T(2)-T(4))
      Q3= - (Q(2,1)+T(11)+ RS(2,1,S) -T(5)-T(7))
      Q4= - (Q(2,3)+T(2) + RS(2,3,S) -T(8)-T(6))
      Q5= - (Q(3,1)+T(7) + RS(3,1,S) -T(11)-T(9))
      Q6= - (Q(3,2)+T(3) + RS(3,2,S) -T(10)-T(12))
      IF(Q1.LE.1.0E-03) GOTO 181
      T(1) = T(1)-Q1*T(1)/(T(1)+T(3))
      T(3) = Q(1,2)+T(10)+RS(1,2,S)-T(1)
      IX=1
```

```
      IY=3
      GOTO 500
C
181   IF(Q2.LE.1.0E-03) GOTO 182
      T(4) = T(4)-Q2*T(4)/(T(4)+T(2))
      T(2) = Q(1,3)+T(6)+RS(1,3,S)-T(4)
      IX=4
      IY=2
      GOTO 500
182   IF(Q3.LE.1.0E-03) GOTO 183
      T(5) = T(5)-Q3*T(5)/(T(5)+T(7))
      T(7) = Q(2,1)+T(11)+RS(2,1,S)-T(5)
      IX=5
      IY=7
      GOTO 500 .
183   IF(Q4.LE.1.0E-03) GOTO 184
      T(8) = T(8)-Q4*T(8)/(T(8)+T(6))
      T(6) = Q(2,3)+T(2)+RS(2,3,S)-T(8)
      IX=8
      IY=6
      GOTO 500
184   IF(Q5.LE.1.0E-03) GOTO 185
      T(9) = T(9)-Q5*T(9)/(T(9)+T(11))
      T(11) = Q(3,1)+T(7)+RS(3,1,S)-T(9)
      IX=9
      IY=11
      GOTO 500
185   IF(Q6.LE.1.0E-03) GOTO 186
      T(12) = T(12)-Q6*T(12)/(T(12)+T(10))
      T(10) = Q(3,2)+T(3)+RS(3,2,S)-T(12)
      IX=12
```

```
      IY=10
500   NR=1
      RLAM = T(IX)*RO
      IF (RLAM) 503,503,502
502   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IX) = T(IX)-IR(1)
      GOTO 504
503   T(IX) = T(IX)
504   RLAM=T(IY)*RO
      IF (RLAM) 506,506,505
505   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IY) = T(IY)-IR(1)
      GOTO 200
506   T(IY) = T(IY)
      GOTO 200
186   CONTINUE
C-----------------------------------------------
C   UPDATE THE QUEUES
C-----------------------------------------------
C
      U122=T(1)
      U123=T(2)
      U132=T(3)
      U133=T(4)
      U211=T(5)
      U213=T(6)
      U231=T(7)
      U233=T(8)
      U311=T(9)
      U312=T(10)
      U321=T(11)
```

```
      U322=T(12)
C
      IF(T(1)-U122.GT.0.5) U122 = U122 + 1
      IF(T(2)-U123.GT.0.5) U123 = U123 + 1
      IF(T(3)-U132.GT.0.5) U132 = U132 + 1
      IF(T(4)-U133.GT.0.5) U133 = U133 + 1
      IF(T(5)-U211.GT.0.5) U211 = U211 + 1
      IF(T(6)-U213.GT.0.5) U213 = U213 + 1
      IF(T(7)-U231.GT.0.5) U231 = U231 + 1
      IF(T(8)-U233.GT.0.5) U233 = U233 + 1
      IF(T(9)-U311.GT.0.5) U311 = U311 + 1
      IF(T(10)-U312.GT.0.5) U312 = U312 + 1
      IF(T(11)-U321.GT.0.5) U321 = U321 + 1
      IF(T(12)-U322.GT.0.5) U322 = U322 + 1
C
      Q(1,2) = Q(1,2)+RS(1,2,S)+U312-U122-U132
      Q(1,3) = Q(1,3)+RS(1,3,S)+U213-U123-U133
      Q(2,1) = Q(2,1)+RS(2,1,S)+U321-U231-U211
      Q(2,3) = Q(2,3)+RS(2,3,S)+U123-U233-U213
      Q(3,1) = Q(3,1)+RS(3,1,S)+U231-U321-U311
      Q(3,2) = Q(3,2)+RS(3,2,S)+U132-U312-U322
C
      QQ(1) = Q(1,2) + Q(1,3)
      QQ(2) = Q(2,1) + Q(2,3)
      QQ(3) = Q(3,1) + Q(3,2)
C
      RR(1) =RS(1,2,S) + RS(1,3,S)
      RR(2) =RS(2,1,S) + RS(2,3,S)
      RR(3) =RS(3,1,S) + RS(3,2,S)
C -----PACKETS QLOST ------------------------
      DO 12 I=1,3
```

```
      QL(I) = 0.
      Y(I)  = 0.
      QL1(I) = 0.
      QL2(I)  = 0.
      DO 6 J=1,3
      IF(I.EQ.J) GO TO 6
      QLL1(I,J) = MAX(QT(I,J) - BF(I,J), 0.)
      QL1(I) =  QL1(I) + QLL1(I,J)
      Q(I,J) =  Q(I,J) - QLL1(I,J)
6     CONTINUE
      QQ(I) = QQ(I) - QL1(I)
      IF(QQ(I)-B(I).LE.0.5) GOTO 10
      QL2(I) = QQ(I) - B(I)
      IF(QL2(I).LT.1.) QL2(I)=1.
      DO 7 J=1,3
      IF(I.EQ.J) GO TO 7
      NN = QL2(I)
      M  = RS(I,J,S)
      L  =  RR(I)
      CALL RNSET (IISEED)
      CALL RNHYP (1,NN,M,L,IR)
      CALL RNGET (IISEED)
      QLL2(I,J) =  IR(1)
      Y(I) = Y(I) + QLL2(I,J)
7     CONTINUE
      DO 9 J=1,3
      IF(I.EQ.J) GO TO 9
      IF(Y(I)-QL2(I).EQ.0.0) GO TO 8
      QLL2(I,J) = QLL2(I,J) + (QL2(I)-Y(I))*RS(I,J,S)/RR(I)
8     Q(I,J)=Q(I,J)-QLL2(I,J)
9     CONTINUE
```

```
10      QL(I)=QL1(I)+QL2(I)
        QQ(I)=QQ(I)-QL2(I)
        QLOST = QLOST + QL(I)
        QTT = QTT + QQ(I)
        QQT(I,S) =  QQ(I)
12      CONTINUE
        S = S + 1
        IF(S.LE.61) GOTO 1000
C ------  END OF THE LOOP -----------------
C -- COMPUTE THE DELAY AND THE THROUGHPUT -------
        DO 300 I=1,3
        QIL=QIL+QQT(I,1)-QQT(I,61)
300     CONTINUE
        AVD=QTT/(TARR-QLOST+QIL)
        THR=(QIL+TARR-QLOST)/60.0
        RETURN
        END
C
C-----------------------------------------------
        SUBROUTINE COMP(K,NITER,A1,A2,A3,A4,A5)
C-----------------------------------------------
        INTEGER     I,K,NITER
        REAL        A1(10),A2(10),A3(10),A4(10),A5(10),M1,M2,M3,M4,M5
     &              ,V1,V2,V3,V4,SD1,SD2,SD3,SD4,PCT1,PCT2,PCT3,PCT4
     &              ,UB1,UB2,UB3,UB4,LB1,LB2,LB3,LB4
C
        M1=0.0
        M2=0.0
        M3=0.0
        M4=0.0
        M5=0.0
```

```
      DO 1 I=1,NITER
         M1= M1+A1(I)
         M2= M2+A2(I)
         M3= M3+A3(I)
         M4= M4+A4(I)
         M5= M5+A5(I)
1     CONTINUE
      M1=M1/NITER
      M2=M2/NITER
      M3=M3/NITER
      M4=M4/NITER
      M5=M5/NITER
C
      V1=0.0
      V2=0.0
      V3=0.0
      V4=0.0
      DO 2 I=1,NITER
         V1= V1+((A1(I)-M1)**2)
         V2= V2+((A2(I)-M2)**2)
         V3= V3+((A3(I)-M3)**2)
         V4= V4+((A4(I)-M4)**2)
2     CONTINUE
      V1=V1/(NITER-1)
      V2=V2/(NITER-1)
      V3=V3/(NITER-1)
      V4=V4/(NITER-1)
C
      SD1=SQRT(V1/FLOAT(NITER))
      SD2=SQRT(V2/FLOAT(NITER))
      SD3=SQRT(V3/FLOAT(NITER))
```

```
      SD4=SQRT(V4/FLOAT(NITER))
C
      PCT1=SD1*2.26*100/M1
      PCT2=SD2*2.26*100/M2
      PCT3=SD3*2.26*100/M3
      PCT4=SD4*2.26*100/M4
C
      UB1=M1 + SD1*2.26
      UB2=M2 + SD2*2.26
      UB3=M3 + SD3*2.26
      UB4=M4 + SD4*2.26
C
      LB1=M1 - SD1*2.26
      LB2=M2 - SD2*2.26
      LB3=M3 - SD3*2.26
      LB4=M4 - SD4*2.26
      WRITE(9,60) K,M1,M2,M3,M4,M5
C
      WRITE(4,*) ('PERCENT STANDARD DIVIATION')
      WRITE(4,66) K,PCT1,PCT2,PCT3,PCT4
C
      WRITE(4,*) ('CONFID INTERVAL')
      WRITE(4,66) K,UB1,UB2,UB3,UB4
      WRITE(4,66) K,LB1,LB2,LB3,LB4
C
60    FORMAT(2X,I3,F8.4,F8.2,3F12.1)
66    FORMAT(2X,I3,4F12.4)
      RETURN
      END
```

A.4 The Program for the SMA model

```
      INTEGER   K,II,NITER
      REAL      AVD,THR,QLOST,QTT,TARR,A1(10),A2(10),A3(10),A4(10)
     &          ,A5(10)
      DOUBLE PRECISION DSEED
      NITER=10
      DO 700 K=1,70
         DO 600 II=1,NITER
            CALL STSEED(II,DSEED)
            CALL SIM (K,DSSED,AVD,THR,QLOST,QTT,TARR)
            A1(II)=AVD
            A2(II)=THR
            A3(II)=QLOST
            A4(II)=QTT
            A5(II)=TARR
          WRITE(4,50) A1(II),A2(II),A3(II),A4(II),A5(II)
C         WRITE(4,50) K, AVD,THR,QLOST,QTT,TARR
50        FORMAT(2X,I3,2F9.3,3F12.1)
600       CONTINUE
          CALL COMP(K,NITER,A1,A2,A3,A4,A5)
700   CONTINUE
      STOP
      END
C-----------------------------------------
      SUBROUTINE STSEED (II,DSEED)
C-----------------------------------------
      INTEGER   II
      DOUBLE PRECISION NSEED,DSEED
C
      NSEED(1)=123457.0
```

```
      NSEED(2)=68374381.0
      NSEED(3)=964393174.0
      NSEED(4)=121742663.0
      NSEED(5)=61433579.0
      NSEED(6)=11572403.0
      NSEED(7)=15726055.0
      NSEED(8)=48108509.0
      NSEED(9)=999792099.0
      NSEED(10)=477424540.0
C
      DSEED=NSEED(II)
C
      RETURN
      END
C-------------------------------------------------------------
      SUBROUTINE SIM (K,DSEED,AVD,THR,QLOST,QTT,TARR)
C-------------------------------------------------------------
      INTEGER        NR,IR(100),IER,K,S,R,IX,U122,LDA,MM,NVAR,I
     & ,      U123,U132,U133,U211,U213,U231,U233,U311,U312,U321,U322
      PARAMETER     (MM = 27, NVAR = 12, LDA = MM)
      INTEGER       IRTYPE(MM), ISEED,IISEED,M,NN,L,D(3,3)
      REAL         XLB(NVAR),XUB(NVAR),T(12),A(LDA,NVAR),BB(MM),Y(3)
     &             ,CC(NVAR),OBJ,XSOL(NVAR),DSOL(NVAR),BF(3,3),QQ(3)
     &            ,RS(3,3,900), EA(3,3,900),TARR,QL(3),QIL
     &            ,Q(3,3),QLOST,QTT,B(3),C(3,3),BS(3),B1(3),B2(3,3)
     &            ,RLAM,ER(NVAR),ALF(3,3),BFS(3,3)
     &            ,Q1,Q2,Q3,Q4,Q5,Q6,AVD,THR,EER(3,3),QQT(3,900),RO
     &            ,RR(3),B0,A0,B3(3,3),N,QQQ(3)
     &            , QLL1(3,3), QLL2(3,3),QL1(3),QL2(3)
C ***   IMSL SUBROUTINES  ******
      EXTERNAL     RNSET, RNUND, DLPRS, RNBYP,RNGET,GGPOS
```

```
      DATA XLB/12* 0./
      DATA XUB/12* 50./
      DATA IRTYPE/27*1/
C
C ********************  INITIALIZATION **************
C -----N = NUMBER OF NODES IN THE NETWORK -------
      N = 3
      TARR  = 0.0
      QLOST = 0.0
      QTT   = 0.0
      QIL   = 0.0
      RO = .001
      ISEED = INT(DSEED)
      IISEED = 123457
C
      DO 1 I=1,3
      B(I) = 200.
      A0 = 10.
      B0 = B(I)-(N-1)*A0
      QQ(I) = 100.
      QL(I) = 0.0
      DO 1 J=1,3
      IF(I.EQ.J) GO TO 1
      C(I,J)  =50.0
      BF(I,J)  =A0+B0
      Q(I,J)  =50.0
      QQT(I,1)  =100.
C ----  EXTERNAL ARRIVAL AT T=0  ----
      EA(I,J,1)=20.
      RS(I,J,1) = EA(I,J,1)
1     CONTINUE
```

```
C
C---- THE LOOP STARTS FROM S=2 --------------------
      S=2
1000  CONTINUE
C
C ----  GENERATE EXTERNAL ARRIVAL RATE IN THE UNBALANCED CASE -------
C     NR=7
C     ISEED = 123457
C     CALL RNSET (ISEED)
C     CALL RNUND (NR, K, IR)
C     EA(1,2,S) =IR(2)
C     EA(1,3,S) = IR(3)
C     EA(2,1,S) = IR(4)
C     EA(2,3,S) = IR(5)
C     EA(3,1,S) = IR(6)
C     EA(3,2,S) = IR(7)
C ------------EXTERNAL ARRIVAL RATE IN THE BALANCED CASE = K-----
      EA(1,2,S) = K
      EA(1,3,S) = K
      EA(2,1,S) = K
      EA(2,3,S) = K
      EA(3,1,S) = K
      EA(3,2,S) = K
C ------------GENERATE EXTERNAL INPUT -------------------------
      NR=1
      DSEED=DSEED + S
      DO 3 I=1,3
      DO 3 J=1,3
      IF(I.EQ.J) GO TO 3
      RLAM = EA(I,J,S)
      IF(RLAM.LE.0.0) GO TO 2
```

```
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      RS(I,J,S) = IR(1)
C -------TOTAL ARRIVAL  ----------------
      TARR = TARR + RS(I,J,S)
      GOTO 3
2     RS(I,J,S) = 0
3     CONTINUE
C*****************************************************
C ** B1=B-Q-EA IF B-Q > EA AND B1 = 0 IF B-Q < EA **
C*****************************************************
      DO 11 I=1,3
      DO 11 J=1,3
      IF(I.EQ.J) GOTO 11
      IF((Q(I,J)+EA(I,J,S)).LE.A0) GOTO 14
      D(I,J) = 1
      GOTO 11
14    D(I,J) = 0
11    CONTINUE
      DO 13 I=1,3
      BS(I) = B0
      DO 13 J=1,3
      IF(I.EQ.J) GOTO 13
      BS(I) = BS(I)-D(I,J)*(Q(I,J)+EA(I,J,S)-A0)
      B1(I) = MAX (0.,BS(I))
      BFS(I,J) = BF(I,J) - Q(I,J) -EA(I,J,S)
      B2(I,J) = D(I,J)*MAX(0.,BFS(I,J))
      B3(I,J) = (1-D(I,J))*(A0-Q(I,J)-EA(I,J,S))
      ALF(I,J) = 1 + (Q(I,J) + EA(I,J,S))/BF(I,J)
13    CONTINUE
C
C ----- THE CONSTRAINTS ---------
```

```
BB(1) = C(1,2)
BB(2) = C(1,3)
BB(3) = C(2,1)
BB(4) = C(2,3)
BB(5) = C(3,1)
BB(6) = C(3,2)
BB(7) = Q(1,2) + EA(1,2,S)
BB(8) = Q(1,3) + EA(1,3,S)
BB(9) = Q(2,1) + EA(2,1,S)
BB(10) = Q(2,3)+ EA(2,3,S)
BB(11) = Q(3,1)+ EA(3,1,S)
BB(12) = Q(3,2)+ EA(3,2,S)
BB(13) = B1(1)
BB(14) = B1(2)
BB(15) = B1(3)
BB(16) = B2(1,2)
BB(17) = B2(1,3)
BB(18) = B2(2,1)
BB(19) = B2(2,3)
BB(20) = B2(3,1)
BB(21) = B2(3,2)
BB(22) = B3(1,2)
BB(23) = B3(1,3)
BB(24) = B3(2,1)
BB(25) = B3(2,3)
BB(26) = B3(3,1)
BB(27) = B3(3,2)
DO 16 I=13,27
IF(BB(I).EQ.0.0) GOTO 15
IRTYPE(I) = 1
GOTO 16
```

```
15      IRTYPE(I) = 0
16      CONTINUE
C
        DO 17 I = 1,MM
        DO 17 J = 1,NVAR
        A(I,J) = 0.0
17      CONTINUE
C
        DO 18 I = 1,6
        A(I,I*2) = 1.0
        A(I,I*2-1) = 1.0
18      CONTINUE
        DO 19 I = 7,10
        A(I,I-4) = 1.0
        A(I+2,I) = 1.0
19      CONTINUE
        DO 20 I = 7,8
        A(I,I-6) = 1.0
        A(I+4,I+4) = 1.0
20      CONTINUE
        A(7,10) = -1.0
        A(8,6) = -1.0
        A(9,11) = -1.0
        A(10,2) = -1.0
        A(11,7) = -1.0
        A(12,3) = -1.0
        A(13,6) =  D(1,3)
        A(13,10) = D(1,2)
        A(14,2) =  D(2,3)
        A(14,11) = D(2,1)
        A(15,3) =  D(3,2)
```

```
      A(15,7) =  D(3,1)
      A(16,10) = D(1,2)
      A(17,6) =  D(1,3)
      A(18,11) = D(2,1)
      A(19,2) =  D(2,3)
      A(20,7) =  D(3,1)
      A(21,3) =  D(3,2)
      A(22,10) = 1-D(1,2)
      A(23,6) =  1-D(1,3)
      A(24,11) = 1-D(2,1)
      A(25,2) =  1-D(2,3)
      A(26,7) =  1-D(3,1)
      A(27,3) =  1-D(3,2)
C     DO 55 I =7,MM
C     IF(S.GT.2) GOTO 55
C     WRITE(9,111) (A(I,J), J=1,12)
111   FORMAT(3X,12F6.2)
C     WRITE(9,*) BB(I)
55    CONTINUE
C
C ---------   OBJECTIVE FUNCTION  ----------
      CC(1) = -ALF(1,2)*(1-RO)
      CC(2) = (ALF(2,3)-ALF(1,3))*(1-RO)
      CC(3) = (ALF(3,2)-ALF(1,2))*(1-RO)
      CC(4) = -ALF(1,3)*(1-RO)
      CC(5) = -ALF(2,1)*(1-RO)
      CC(6) = (ALF(1,3)-ALF(2,3))*(1-RO)
      CC(7) = (ALF(3,1)-ALF(2,1))*(1-RO)
      CC(8) = -ALF(2,3)*(1-RO)
      CC(9) = -ALF(3,1)*(1-RO)
      CC(10) = (ALF(1,2)-ALF(3,2))*(1-RO)
```

```
      CC(11) = (ALF(2,1)-ALF(3,1))*(1-RO)
      CC(12) = -ALF(3,2)*(1-RO)
C  ------  SUBROUTINE TO MINIMIZE THE OBJECTIVE FUNCTION -------------
C
      CALL DLPRS(MM,NVAR,A,LDA,BB,BB,CC,IRTYPE,XLB,XUB,OBJ,XSOL,DSOL)
C
      T(1)=XSOL(1)
      T(2)=XSOL(2)
      T(3)=XSOL(3)
      T(4)=XSOL(4)
      T(5)=XSOL(5)
      T(6)=XSOL(6)
      T(7)=XSOL(7)
      T(8)=XSOL(8)
      T(9)=XSOL(9)
      T(10)=XSOL(10)
      T(11)=XSOL(11)
      T(12)=XSOL(12)
C
      DO 4000 I=1,12
      IF(T(I).LT.51.) GOTO 4000
C     WRITE(9,*) T(I)
      T(I)=50.
4000  CONTINUE
C -------------------GENERATE CHANNEL ERRORS ------------
      NR=1
      DSEED=DSEED + S
      DO 4 I=1,12
      RLAM = RO*T(I)
      IF(RLAM.LE.0.0) GOTO 5
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
```

```
      ER(I) = IR(1)
      T(I)= T(I)-ER(I)
      GOTO 4
5     ER(I) = 0.0
      T(I)=T(I)
4     CONTINUE
C ---------------------------------
C THE ROUTING POLICY
C---------------------------------
C
200   Q1= - (Q(1,2)+T(10)+RS(1,2,S)-T(1)-T(3))
      Q2= - (Q(1,3)+T(6)+RS(1,3,S)-T(2)-T(4))
      Q3= - (Q(2,1)+T(11)+RS(2,1,S)-T(5)-T(7))
      Q4= - (Q(2,3)+T(2)+RS(2,3,S)-T(8)-T(6))
      Q5= - (Q(3,1)+T(7)+RS(3,1,S)-T(11)-T(9))
      Q6= - (Q(3,2)+T(3)+RS(3,2,S)-T(10)-T(12))
      IF(Q1.LE.1.0E-03) GOTO 181
      T(1) = T(1)-Q1*T(1)/(T(1)+T(3))
      T(3) = Q(1,2)+T(10)+RS(1,2,S)-T(1)
      IX=1
      IY=3
      GOTO 500
C
181   IF(Q2.LE.1.0E-03) GOTO 182
      T(4) = T(4)-Q2*T(4)/(T(4)+T(2))
      T(2) = Q(1,3)+T(6)+RS(1,3,S)-T(4)
      IX=4
      IY=2
      GOTO 500
182   IF(Q3.LE.1.0E-03) GOTO 183
      T(5) = T(5)-Q3*T(5)/(T(5)+T(7))
```

```
      T(7) = Q(2,1)+T(11)+RS(2,1,S)-T(5)
      IX=5
      IY=7
      GOTO 500
183   IF(Q4.LE.1.0E-03) GOTO 184
      T(8) = T(8)-Q4*T(8)/(T(8)+T(6))
      T(6) = Q(2,3)+T(2)+RS(2,3,S)-T(8)
      IX=8
      IY=6
      GOTO 500
184   IF(Q5.LE.1.0E-03) GOTO 185
      T(9) = T(9)-Q5*T(9)/(T(9)+T(11))
      T(11) = Q(3,1)+T(7)+RS(3,1,S)-T(9)
      IX=9
      IY=11
      GOTO 500
185   IF(Q6.LE.1.0E-03) GOTO 186
      T(12) = T(12)-Q6*T(12)/(T(12)+T(10))
      T(10) = Q(3,2)+T(3)+RS(3,2,S)-T(12)
      IX=12
      IY=10
500   NR=1
      RLAM = T(IX)*RO
      IF (RLAM) 503,503,502
502   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      T(IX) = T(IX)-IR(1)
      GOTO 504
503   T(IX) = T(IX)
504   RLAM=T(IY)*RO
      IF (RLAM) 506,506,505
505   CALL GGPOS(RLAM,DSEED,NR,IR,IER)
```

```
      T(IY) = T(IY)-IR(1)
      GOTO 200
506   T(IY) = T(IY)
      GOTO 200
186   CONTINUE
C-----------------------------------------------
C   UPDATE THE QUEUES
C-----------------------------------------------
      U122=T(1)
      U123=T(2)
      U132=T(3)
      U133=T(4)
      U211=T(5)
      U213=T(6)
      U231=T(7)
      U233=T(8)
      U311=T(9)
      U312=T(10)
      U321=T(11)
      U322=T(12)
C
      IF(T(1)-U122.GT.0.5) U122 = U122 + 1
      IF(T(2)-U123.GT.0.5) U123 = U123 + 1
      IF(T(3)-U132.GT.0.5) U132 = U132 + 1
      IF(T(4)-U133.GT.0.5) U133 = U133 + 1
      IF(T(5)-U211.GT.0.5) U211 = U211 + 1
      IF(T(6)-U213.GT.0.5) U213 = U213 + 1
      IF(T(7)-U231.GT.0.5) U231 = U231 + 1
      IF(T(8)-U233.GT.0.5) U233 = U233 + 1
      IF(T(9)-U311.GT.0.5) U311 = U311 + 1
      IF(T(10)-U312.GT.0.5) U312 = U312 + 1
```

```
      IF(T(11)-U321.GT.0.5) U321 = U321 + 1
      IF(T(12)-U322.GT.0.5) U322 = U322 + 1
C
      Q(1,2) = Q(1,2)+RS(1,2,S)+U312-U122-U132
      Q(1,3) = Q(1,3)+RS(1,3,S)+U213-U123-U133
      Q(2,1) = Q(2,1)+RS(2,1,S)+U321-U231-U211
      Q(2,3) = Q(2,3)+RS(2,3,S)+U123-U233-U213
      Q(3,1) = Q(3,1)+RS(3,1,S)+U231-U321-U311
      Q(3,2) = Q(3,2)+RS(3,2,S)+U132-U312-U322
C
      QQ(1) = Q(1,2) + Q(1,3)
      QQ(2) = Q(2,1) + Q(2,3)
      QQ(3) = Q(3,1) + Q(3,2)
C
      RR(1) =RS(1,2,S) + RS(1,3,S)
      RR(2) =RS(2,1,S) + RS(2,3,S)
      RR(3) =RS(3,1,S) + RS(3,2,S)
C----- QLOST = QUEUE - BUFFER ----------
      DO 12 I=1,3
      QL(I) = 0.
      Y(I)  = 0.
      QL1(I) = 0.
      QL2(I)  = 0.
      QQQ(I)  = 0.
      DO 6 J=1,3
      IF(I.EQ.J) GO TO 6
      QLL1(I,J) = MAX(Q(I,J) - BF(I,J), 0.)
      Q(I,J) = Q(I,J) - QLL1(I,J)
      QL1(I) =  QL1(I) + QLL1(I,J)
      QQQ(I) = QQQ(I)+MAX(Q(I,J)-A0,0.)
6     CONTINUE
```

```
      QQ(I) = QQ(I) - QL1(I)
      IF((QQQ(I)- BO).LE.0.5) GO TO 10
      QL2(I) =  QQQ(I) - BO
      IF(QL2(I).LT.1.0) QL2(I) = 1.
      DO 7 J=1,3
      IF(I.EQ.J) GO TO 7
      NN = QL2(I)
      M  = RS(I,J,S)
      L  =  RR(I)
      CALL RNSET (IISEED)
      CALL RNHYP (NR,NN,M,L,IR)
      CALL RNGET (IISEED)
      QLL2(I,J) =  IR(1)
      Y(I) = Y(I) + QLL2(I,J)
7     CONTINUE
      DO 9 J=1,3
      IF(I.EQ.J) GO TO 9
      IF(Y(I)-QL2(I).EQ.0.0) GO TO 8
      QLL2(I,J) = QLL2(I,J) + (QL2(I)-Y(I))*RS(I,J,S)/RR(I)
8     Q(I,J) = Q(I,J) - QLL2(I,J)
9     CONTINUE
10    QL(I) = QL1(I) + QL2(I)
      QQ(I) = QQ(I) - QL2(I)
      QQT(I,S) = QQ(I)
      QLOST = QLOST + QL(I)
C  ---TOTAL QUEUE --------------------
      QTT = QTT + QQ(I)
12    CONTINUE
C
      S = S + 1
      IF(S.LE.61) GOTO 1000
```

```
C
C  ------  END OF THE LOOP -----------
       DO 300 I=1,3
       QIL=QIL+QQT(I,1)-QQT(I,61)
300    CONTINUE
       AVD=QTT/(TARR-QLOST+QIL)
       THR=(QIL+TARR-QLOST)/60.0
       RETURN
       END
C
C-----------------------------------------------
       SUBROUTINE COMP(K,NITER,A1,A2,A3,A4,A5)
C-----------------------------------------------
       INTEGER     I,K,NITER
       REAL        A1(10),A2(10),A3(10),A4(10),A5(10),M1,M2,M3,M4,M5
     &             ,V1,V2,V3,V4,SD1,SD2,SD3,SD4,PCT1,PCT2,PCT3,PCT4
     &             ,UB1,UB2,UB3,UB4,LB1,LB2,LB3,LB4
C
       M1=0.0
       M2=0.0
       M3=0.0
       M4=0.0
       M5=0.0
       DO 1 I=1,NITER
          M1= M1+A1(I)
          M2= M2+A2(I)
          M3= M3+A3(I)
          M4= M4+A4(I)
          M5= M5+A5(I)
1      CONTINUE
       M1=M1/NITER
```

```
      M2=M2/NITER
      M3=M3/NITER
      M4=M4/NITER
      M5=M5/NITER
C
      V1=0.0
      V2=0.0
      V3=0.0
      V4=0.0
      DO 2 I=1,NITER
         V1= V1+((A1(I)-M1)**2)
         V2= V2+((A2(I)-M2)**2)
         V3= V3+((A3(I)-M3)**2)
         V4= V4+((A4(I)-M4)**2)
2     CONTINUE
      V1=V1/(NITER-1)
      V2=V2/(NITER-1)
      V3=V3/(NITER-1)
      V4=V4/(NITER-1)
C
      SD1=SQRT(V1/FLOAT(NITER))
      SD2=SQRT(V2/FLOAT(NITER))
      SD3=SQRT(V3/FLOAT(NITER))
      SD4=SQRT(V4/FLOAT(NITER))
C
      PCT1=SD1*2.26*100/M1
      PCT2=SD2*2.26*100/M2
      PCT3=SD3*2.26*100/M3
      PCT4=SD4*2.26*100/M4
C
      UB1=M1 + SD1*2.26
```

```
      UB2=M2 + SD2*2.26
      UB3=M3 + SD3*2.26
      UB4=M4 + SD4*2.26
C
      LB1=M1 - SD1*2.26
      LB2=M2 - SD2*2.26
      LB3=M3 - SD3*2.26
      LB4=M4 - SD4*2.26
      WRITE(9,60) K,M1,M2,M3,M4,M5
C
      WRITE(4,*) ('PERCENT STANDARD DIVIATION')
      WRITE(4,66) K,PCT1,PCT2,PCT3,PCT4
C
      WRITE(4,*) ('CONFID INTERVAL')
      WRITE(4,66) K,UB1,UB2,UB3,UB4
      WRITE(4,66) K,LB1,LB2,LB3,LB4
C
60    FORMAT(2X,I3,F8.4,F8.2,3F12.1)
66    FORMAT(2X,I3,4F12.4)
      RETURN
      END
```

A.5 The Program for the Fixed routing scheme

```
      INTEGER   K,II,NITER
      REAL      AVD,THR,QLOST,QTT,TARR,A1(10),A2(10),A3(10),A4(10)
     &          ,A5(10)
      DOUBLE PRECISION DSEED
      NITER=10
      DO 700 K=1,70
         DO 600 II=1,NITER
            CALL STSEED(II,DSEED)
            CALL SIM (K,DSSED,AVD,THR,QLOST,QTT,TARR)
            A1(II)=AVD
            A2(II)=THR
            A3(II)=QLOST
            A4(II)=QTT
            A5(II)=TARR
          WRITE(4,50) A1(II),A2(II),A3(II),A4(II),A5(II)
C         WRITE(4,50) K, AVD,THR,QLOST,QTT,TARR
50        FORMAT(2X,I3,2F9.3,3F12.1)
600       CONTINUE
          CALL COMP(K,NITER,A1,A2,A3,A4,A5)
700   CONTINUE
      STOP
      END
C-----------------------------------------
      SUBROUTINE STSEED (II,DSEED)
C-----------------------------------------
      INTEGER   II
      DOUBLE PRECISION NSEED,DSEED
C
      NSEED(1)=123457.0
```

```
      NSEED(2)=68374381.0
      NSEED(3)=964393174.0
      NSEED(4)=121742663.0
      NSEED(5)=61433579.0
      NSEED(6)=11572403.0
      NSEED(7)=15726055.0
      NSEED(8)=48108509.0
      NSEED(9)=999792099.0
      NSEED(10)=477424540.0
C
      DSEED=NSEED(II)
C
      RETURN
      END
C-------------------------------------------------------
      SUBROUTINE SIM (K,DSEED,AVD,THR,QLOST,QTT,TARR)
C-------------------------------------------------------
      INTEGER      N,NR,IR(100),IER,K,S,I
      INTEGER      ISEED,M,NN,L
      REAL        T(6),QQ(3),RS(3,3,900), EA(3,3,900),TARR,QL(3),Y(3)
     &           ,QIL,QT(3,3,900),Q(3,3),QLOST,QTT,B(3),C(3,3),U(3,3)
     &           ,RLAM,ER(6),AVD,THR,QQT(3,3,900),RO,QLL(3,3),RR(3)
      DOUBLE PRECISION DSEED
C  ***  IMSL SUBROUTINES   **********
      EXTERNAL     RNSET, RNUND, RNGET,RNHYP,GGPOS
C ***************************************************************************
C                        INITIALIZATION
C ***************************************************************************
      TARR  = 0.0
      QLOST = 0.0
      QTT   = 0.0
```

```
      QIL   = 0.0
      RO = .001
      ISEED = INT(DSEED)
C
      DO 1 I=1,3
      B(I) = 200.
      QQ(I) = 100.
      QL(I) = 0.0
      DO 1 J=1,3
      IF(I.EQ.J) GO TO 1
      C(I,J)  =50.0
      Q(I,J)  =50.0
      QQT(I,1)  =100.
C ----  EXTERNAL ARRIVAL AT T=0 ----------
      EA(I,J,1)=20.
      RS(I,J,1) = EA(I,J,1)
1     CONTINUE
C
      S=2
1000  CONTINUE
C--------GENERATE EXTERNAL ARRIVAL RATE FOR THE UNBALANCED CASE---
C     NR=7
C     CALL RNSET (ISEED)
C     CALL RNUND (NR, K, IR)
C     CALL RNGET (ISEED)
C     EA(1,2,S) = IR(2)
C     EA(1,3,S) = IR(3)
C     EA(2,1,S) = IR(4)
C     EA(2,3,S) = IR(5)
C     EA(3,1,S) = IR(6)
C     EA(3,2,S) = IR(7)
```

```
C --------- EXTERNAL ARRIVAL RATE IN THE BALANCED CASE = K ----
      EA(1,2,S) = K
      EA(1,3,S) = K
      EA(2,1,S) = K
      EA(2,3,S) = K
      EA(3,1,S) = K
      EA(3,2,S) = K
      NR=1
      DSEED=DSEED + S
      DO 3 I=1,3
      DO 3 J=1,3
      IF(I.EQ.J) GO TO 3
      RLAM = EA(I,J,S)
      IF(RLAM.LE.0.0) GO TO 2
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      RS(I,J,S) = IR(1)
C -------TOTAL ARRIVAL  -----------------
      TARR = TARR + RS(I,J,S)
      GOTO 3
2     RS(I,J,S) = 0
3     CONTINUE
C------------ COMPUTE CONTROL VARIABLES -----------
      DO 20 I=1,3
      DO 20 J=1,3
      IF(I.EQ.J) GOTO 20
      IF((Q(I,J)+RS(I,J,S)).GT.C(I,J)) GOTO 21
      U(I,J)=Q(I,J)+RS(I,J,S)
      GOTO 20
21    U(I,J)=C(I,J)
20    CONTINUE
      T(1)=U(1,2)
```

```
      T(2)=U(1,3)
      T(3)=U(2,1)
      T(4)=U(2,3)
      T(5)=U(3,1)
      T(6)=U(3,2)
C ------- GENERATE CHANNEL ERRORS----------
C
      NR=1
      DSEED=DSEED + S
      DO 4 I=1,6
      RLAM = RO*T(I)
      IF(RLAM.LE.0.0) GOTO 5
      CALL GGPOS(RLAM,DSEED,NR,IR,IER)
      ER(I) = IR(1)
      T(I)= T(I)-ER(I)
      GOTO 4
5     ER(I) = 0.0
      T(I)=T(I)
4     CONTINUE
C ----------------------------
C THE ROUTING POLICY
C----------------------------
      Q(1,2) = Q(1,2)+RS(1,2,S)-T(1)
      Q(1,3) = Q(1,3)+RS(1,3,S)-T(2)
      Q(2,1) = Q(2,1)+RS(2,1,S)-T(3)
      Q(2,3) = Q(2,3)+RS(2,3,S)-T(4)
      Q(3,1) = Q(3,1)+RS(3,1,S)-T(5)
      Q(3,2) = Q(3,2)+RS(3,2,S)-T(6)
C
      QQ(1) = Q(1,2) + Q(1,3)
      QQ(2) = Q(2,1) + Q(2,3)
```

```
      QQ(3) = Q(3,1) + Q(3,2)
C
      RR(1) =RS(1,2,S) + RS(1,3,S)
      RR(2) =RS(2,1,S) + RS(2,3,S)
      RR(3) =RS(3,1,S) + RS(3,2,S)
C ------- QLOST = QUEUE - BUFFER -----------
      DO 12 I=1,3
      QL(I) = 0.
      Y(I)  = 0.
      IF(QQ(I)- B(I).LE.0.5) GO TO 10
      QL(I) =  QQ(I) - B(I)
      IF(QL(I).LT.1.0) QL(I)=1.
      DO 7 J=1,3
      IF(I.EQ.J) GO TO 7
      ISEED=123457
      NN = QL(I)
      M  = RS(I,J,S)
      L  =  RR(I)
      CALL RNSET (ISEED)
      CALL RNHYP (1,NN,M,L,IR)
      QLL(I,J) =  IR(1)
      Y(I) = Y(I) + QLL(I,J)
      QT(I,J,S) = Q(I,J)
7     CONTINUE
      DO 9 J=1,3
      IF(I.EQ.J) GO TO 9
      IF(Y(I)-QL(I).EQ.0.0) GO TO 8
      QLL(I,J) = QLL(I,J) + (QL(I)-Y(I))*RS(I,J,S)/RR(I)
8     Q(I,J) = Q(I,J) - QLL(I,J)
9     CONTINUE
10    QQ(I) = QQ(I) - QL(I)
```

```
      QQT(I,S) = QQ(I)
      QLOST = QLOST + QL(I)
C  ------TOTAL QUEUE --------------
      QTT = QTT + QQ(I)
12    CONTINUE
      S = S + 1
      IF(S.LE.61) GOTO 1000
C
C  ----  END OF THE LOOP -----------
      DO 300 I=1,3
      QIL=QIL+QQT(I,1)-QQT(I,61)
300   CONTINUE
      AVD=QTT/(TARR-QLOST+QIL)
      THR=(QIL+TARR-QLOST)/60.
      RETURN
      END
C
C-------------------------------------------------
      SUBROUTINE COMP(K,NITER,A1,A2,A3,A4,A5)
C-------------------------------------------------
      INTEGER     I,K,NITER
      REAL        A1(10),A2(10),A3(10),A4(10),A5(10),M1,M2,M3,M4,M5
     &            ,V1,V2,V3,V4,SD1,SD2,SD3,SD4,PCT1,PCT2,PCT3,PCT4
     &            ,UB1,UB2,UB3,UB4,LB1,LB2,LB3,LB4
C
      M1=0.0
      M2=0.0
      M3=0.0
      M4=0.0
      M5=0.0
      DO 1 I=1,NITER
```

```
          M1= M1+A1(I)
          M2= M2+A2(I)
          M3= M3+A3(I)
          M4= M4+A4(I)
          M5= M5+A5(I)
1       CONTINUE
        M1=M1/NITER
        M2=M2/NITER
        M3=M3/NITER
        M4=M4/NITER
        M5=M5/NITER
C
        V1=0.0
        V2=0.0
        V3=0.0
        V4=0.0
        DO 2 I=1,NITER
          V1= V1+((A1(I)-M1)**2)
          V2= V2+((A2(I)-M2)**2)
          V3= V3+((A3(I)-M3)**2)
          V4= V4+((A4(I)-M4)**2)
2       CONTINUE
        V1=V1/(NITER-1)
        V2=V2/(NITER-1)
        V3=V3/(NITER-1)
        V4=V4/(NITER-1)
C
        SD1=SQRT(V1/FLOAT(NITER))
        SD2=SQRT(V2/FLOAT(NITER))
        SD3=SQRT(V3/FLOAT(NITER))
        SD4=SQRT(V4/FLOAT(NITER))
```

```
C
      PCT1=SD1*2.26*100/M1
      PCT2=SD2*2.26*100/M2
      PCT3=SD3*2.26*100/M3
      PCT4=SD4*2.26*100/M4
C
      UB1=M1 + SD1*2.26
      UB2=M2 + SD2*2.26
      UB3=M3 + SD3*2.26
      UB4=M4 + SD4*2.26
C
      LB1=M1 - SD1*2.26
      LB2=M2 - SD2*2.26
      LB3=M3 - SD3*2.26
      LB4=M4 - SD4*2.26
      WRITE(9,60) K,M1,M2,M3,M4,M5
C
C     WRITE(4,*) ('PERCENT STANDARD DIVIATION')
      WRITE(4,66) K,PCT1,PCT2,PCT3,PCT4
C
C     WRITE(4,*) ('CONFID INTERVAL')
C     WRITE(4,66) K,UB1,UB2,UB3,UB4
C     WRITE(4,66) K,LB1,LB2,LB3,LB4
C
60    FORMAT(2X,I3,F8.4,F8.2,3F12.1)
66    FORMAT(2X,I3,4F12.4)
      RETURN
      END
```