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.

INVESTIGATION OF PACKET LOSS REDUCTION METHODS AND DECENTRALIZATION FOR DYNAMIC ROUTING IN PACKET-SWITCHING NETWORKS

by

Luc D. Lamontagne

A thesis
presented to the University of Ottawa
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Systems Science

ISBN 0-315-60094-2

UNIVERSITÉ D'OTTAWA
UNIVERSITY OF OTTAWA

Abstract

In this thesis, we slightly modify a known routing model in packet-switching networks. Those modifications give priority to messages transiting in the network (internal) over external arrivals (users) incoming to each node while reducing the number of lost packets. Mathematical programs are developed for the multiple and single buffer node models. Those take into account line capacities, finite buffers and channel errors during transmission. We investigate a simple decentralized routing policy, based on the proposed centralized model with multiple buffers, where partial information is available from neighboring nodes only. Integer linear programming is used to optimally solve the routing problem for each of the models studied. Simulation, conducted on a 5-node network, illustrates the impact of these modifications.

Acknowledgements

Je désire tout d'abord remercier Prof. N. U. Ahmed pour son aide dans le choix du sujet ainsi que pour ses judicieux conseils.

Á mes parents, pour le support ainsi que la confiance qu'ils ont su me témoigner.

Et principalement à ma bien-aimée épouse Irène Abi-Zeid qui m'a toujours appuyé, tant par ses idées constructives que par l'aide apportée à la rédaction de cette thèse, aux moments les meilleurs comme aux moments difficiles tout au long de ce périple.

Table of contents

CHAPTER 1
Introduction

Haven't we all, at one time or another experienced frustration because of lost electronic mail messages or any other problem caused by congestion in a computer network? As a matter of fact, even though computer and communication techniques have been improving faster than ever, users are still faced with many physical constraints. However, one may try to maximize the use of available resources. This thesis is intended to discuss message control problems in computer communication networks.

In this chapter, we examine the routing problem and its different aspects. This is followed by a short literature review surveying different methods used for the modelling and solving of routing problems. Special attention is paid to a model [1] which is used as a basis for this thesis. Finally, the motivation behind this work is provided and the thesis plan is outlined.

1.1 The routing problem

In this thesis, we are interested in the control problems of datagram store-and-forward communication networks where packets moving from one source to one destination are buffered at transit nodes along the path. The two elements of control in those packet-switching networks are flow control and routing. Flow control, as shown in figure 1.1, involves management of incoming traffic (arrivals) in order to prevent buffer overflow. We refer to

internal arrivals as the packets arriving from other nodes through the network, while external arrivals are packets coming either from the stations or users. On the other hand, figure 1.2 illustrates the principle of routing which consists of choosing a path for the packets (small boxes labelled a, b or c) to follow through the network according to certain policies. In this section, we explain the main elements of routing which are the decision place, the information structure, the performance indicators or criteria and the routing policy.

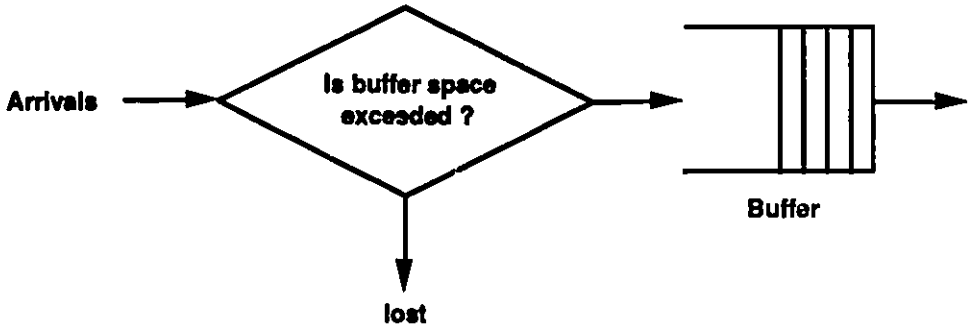


Figure 1.1: Flow control at one node

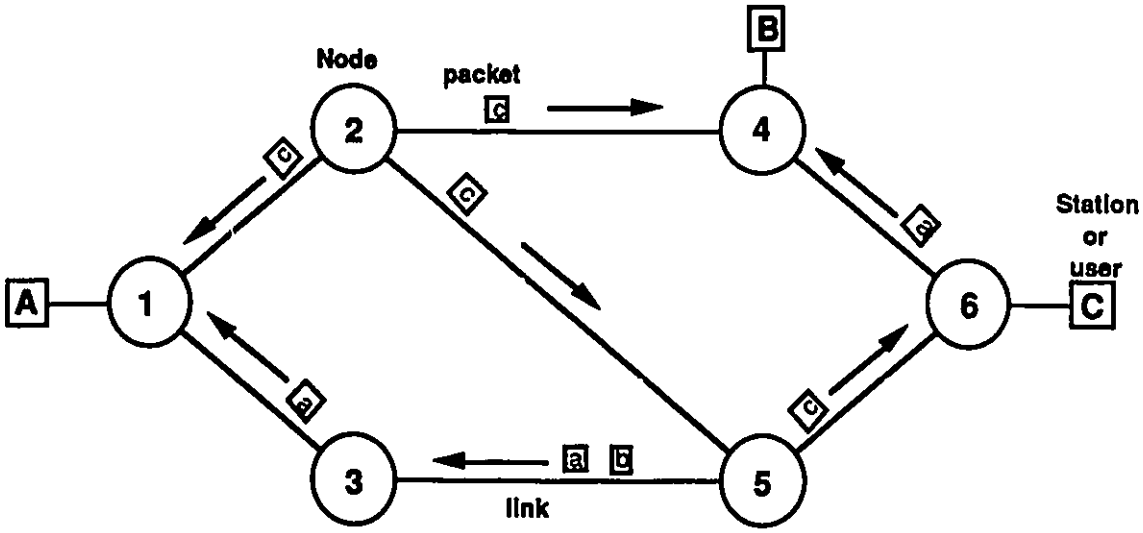


Figure 1.2: Routing in a datagram packet-switching environment

Decision can take place at different levels in a network. When the control of packet exchange in the whole network is assigned to a unique node (control center or central node) we say that the routing is centralized. This provides great coordination among nodes but is vulnerable to any failure involving the central node. Also, such configuration can hardly be implemented when the size of the network gets large. When more than one control center (decision maker) is operating in the same network, we say that the routing is distributed or decentralized. The most frequent case is that where every node is responsible for achieving routing of messages contained in its own local buffers. Hence each router can control its message flow in agreement with the other nodes, according to its individual goals. Some other distributed schemes may be exploited where different types of hierarchies are established among controllers of a network. Decentralized control is appealing in practice since the network reliability is improved, even though performance degradation may occur because of uncoordination between the decision makers.

Decisions are based on information describing the actual state of the network such as the current topology, line capacities, lengths of the queues, nodal buffers, etc... Sometimes, no information is available and actions are taken in the dark. When available, it can be global (from all nodes), local (from one part of the network), or sequentially acquired from nodes along the routing path. Global knowledge of the network is attractive even if data acquisition can be tedious, sometimes impossible in wide area networks, due to the time delays and the overhead created. Partial information moves around more easily but the lack of pertinent data creates uncoordination when nodes are largely

interconnected. The information structure and the decision place are highly related since information about the whole network is needed in centralized routing while decentralized schemes are fundamentally based on local information.

The concept of good performance can take on different meanings depending on one's goals and expectations. Performance indicators allow network controller(s) to respond to different needs. The most popular indicators are the cost, delay and throughput. Cost values can be assigned to network links in proportion to their propagation delays, operational costs or most of the time in an inverse proportion to the links capacities. When these costs are assumed equal or negligible for all links, minimum hop which consists of choosing routes that pass through the least number of nodes is considered. Delay, which is fundamentally defined as the time elapsed until a packet is successfully delivered to its destination, is often interpreted according to the modelling of the problem. When propagation and transmission delays are neglected, a time analysis of the model becomes irrelevant and artificial congestion measures are created such as the length of packet queues. Throughput designates the proportion of the offered load successfully transmitted through the network.

Finally, the routing policy is used to make decisions in conjunction with the last three elements of routing explained above. It is called centralized or decentralized according to where the decision place is located. Routing might react in different ways to parameter variations in the network. One extremum is called fixed routing, where fractions of the traffic at every link are kept constant in time and determined by the network topology and average traffic.

The opposite is called adaptive routing where the routing proportions change as the parameters of the network do. In between those two extrema, other methods such as quasi-static routing, partially react to specific network perturbations. The routing policy can simply be based on intuition as well as on extensive mathematical algorithms. Most of the operating networks use a shortest path algorithm which routes packets from source to destination along a path chosen in advance. Dijkstra shortest path (forward-search) method is used for centralized computation while backward-search algorithms are more appropriate for decentralized routing. Several other techniques have been used to model and optimize communication networks such as graph theory, linear and nonlinear programming, dynamic programming, queueing theory, team theory and optimal control. Analysis is possible under static conditions in the network but when varying, the problem gets more complicated to solve and the assumptions made become more restrictive, which weakens the solutions proposed.

1.2 Literature review

In this section, we review some centralized and decentralized models of the current literature and we explain in more details one model proposed in 1988 by N.U. Ahmed, T.E. Dabbbous and Y.W. Lee [1].

1.2.a Centralized model

The initial models ([14], [15]) proposed for computer communications made use of queueing theory. Leonard Kleinrock [15] introduced the fixed routing strategy which consists of keeping fractions of the traffic at one node destined to each of the other nodes, fixed in time. At every node, packets arriving from outside the network are assigned to one outgoing line. They are placed to wait in queue until they can be retransmitted. This procedure gave origin to the expression store-and-forward. The rate of arrival is Poisson distributed with independent arrival times and the packet lengths are exponentially distributed. The queues are associated with outgoing lines. The routing is static, constant in time and functional of the various transmission rates of the lines.

In 1977, Adrian Segall [31] approached the problem from a macroscopic point of view, which presents an alternative to Kleinrock's model using queueing theory. The model represents incoming and outgoing message flows at nodes, where external arrivals are deterministic. Capacities of the links are finite but buffer size is unlimited. Queues are associated with the nodes rather than with the links. Since the model is time continuous, instantaneous traffic

rates are the routing variables. The processing and propagation time delays are assumed negligible. An algorithm, very tedious to use, has been developed to minimize the total number of packets waiting in queues over a certain period of time. This measure is interpreted as the time spent in the network when the messages are idle (not in transit from one node to another). Furthermore, some variations of the problem are investigated but no solutions are proposed.

One solution to this problem formulation has been studied [23] by means of optimal control theory. This method involving deterministic dynamic programming, assumes time invariant inputs. Using maximum principle, necessary conditions for the optimal solution were obtained.

The dynamic routing can also be considered as an optimal evacuation time problem ([33], [34], [35]) which consists of finding a partitioning of the line capacities in order to clear the initial queues in a minimum time. This problem is meaningful under a low rate of traffic arrivals only.

Over time, various methods and models have been formulated but current literature about centralized routing is mainly based on those previous papers and their particular applications.

1.2.b Decentralized model

Distributed (continuous time model) or decentralized (discrete time model) routed network studies have been motivated by several commercial activities such as, very recently, the conversion of the telephone networks from

analog to digital technology. Originally, the implementation of ARPANET (Advanced Research Projects Agency NETwork), which was devoted to studying the possibilities of computer resources sharing in point-to-point networks, played an important role. Since the number of users was increasing quickly with years, the system became operational in 1975. Its actual routing policy is based on a backward-search algorithm.

A similar algorithm which achieves minimum steady state delay of a packet-switched network is introduced in [11]. This algorithm uses distributed computation in the sense that each node of the network can construct its own routing table, which is a matrix containing the expected estimated incremental delay to any destination from each individual node. Information is updated periodically and estimated using measurements in the network.

Two papers ([27], [28]) present decentralized dynamic routing as a minimum evacuation time. The optimization problem is to clear the congestion of a network which is governed by deterministic delay differential equations. The network where nodes with two outgoing links are subject to constant demand, as well as that of a single-source single-destination, have been solved. Propagation delays, due to the link transmission rate, are considered to be relevant in one case.

More recently, Aicardi, Davoli and Minciardi ([2], [3]) studied decentralized optimal control of Markov chains with delay in the information pattern where team theory is applied. Finite and infinite horizon problems are investigated and it is shown that a non classical dynamic programming approach can be relevant. It is extremely difficult to find a solution with this

method because the information required is unrealistic and the optimal strategies are not implementable.

An interesting attempt [24] has been done to provide a hierarchical two-level adaptive scheme where the decision making process consists of a distributed computation of the routing parameters. After a central node updates the parameters based on the actual state of the network, nodal computations, similar to [11] but with consideration given to buffer limitation, keep the network operating for a period of time without centralized intervention.

1.2.c Ahmed et al. model [1]

This model is a discrete time centralized routing policy where the network is considered to have channel capacity and finite buffers at each node. Let N denote the set of nodes in the network, L = { (i,k); i,k ∈ N} be the set of links, E(i) = { k ∈ N; (i,k) ∈ L} and I(i) = { l ∈ N; (l,i) ∈ L} be the set of nodes that receive from and send to node i respectively. The queue dynamic equations are based on Segall's model

$$\dot{Q}_i^j(t) = \sum_{l \in I(i)} u_{li}^j(t) - \sum_{k \in E(i)} u_{ik}^j(t) + A_i^j(t)$$

where $\dot{Q}_i^j(t)$ is the rate of change of the packet queue at node i with destination j, $u_{li}^j(t)$ is the rate of transmission of packets of type j sent through channel (l,i) and $A_i^j(t)$ is the rate of external mean arrivals at node i with destination j, the

latter being Poisson distributed. The first and third terms represent the arrivals while the middle one is the outgoing rate.

The first modification adopted in [1] is to consider separately the flow of arriving and outgoing packets constrained by (N-1) finite buffers of size B_i^j at each node. Assuming that arrivals are stored in the remaining buffer space associated with their destinations, the arriving load expression is defined as

$$R_i^j(dt) = \left[\sum_{l \in I(i)} \left(u_{li}^j(dt) - \xi_{li}^j(dt) \right) + A_i^j(dt) \right] \wedge \left[B_i^j - Q_i^j(t) \right]$$

where $A \wedge B$ is the minimum value of A and B, and $\xi_{li}^j(dt)$ is the number of packets not transmitted due to channel errors. In a similar way the outgoing flow at node i with destination j is defined as

$$O_i^j(dt) = \sum_{k \in E(i)} \left(u_{ik}^j(dt) - \xi_{ik}^j(dt) \right).$$

The resulting queue variation is the difference of the incoming and outgoing packets. Over a short period of time $(t, t+\Delta)$, denoted Γ_t, the dynamic queue equation can be approximated as the sum of the previous queue state at time t and the queue variation during this interval, and is expressed as

$$Q_i^j(t+\Delta) = Q_i^j(t) + \left[\sum_{l \in I(i)} \left(u_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t) \right) + A_i^j(\Gamma_t) \right] \wedge \left[B_i^j - Q_i^j(t) \right] - \sum_{k \in E(i)} \left(u_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t) \right). \quad (1.1)$$

The optimization goal is to minimize the delay in the network. Since propagation and processing time delays are assumed negligible, the length of the queues is considered to be the measure of delay for packets being transmitted. Equation (1.1) can be seen as a Markov chain where the next queue state is dependent on the previous state. The objective function to be

minimized is the summation of the expected value of the next queue states based on the available information at time t which is

$$J(Q) = \sum_{ij} \alpha_i^j E\left\{Q_i^j(t+\Delta) \mid Q_i^j(t)\right\} \tag{1.2}$$

The weighting factor, which has the purpose of increasing throughput under congestion in the network, is defined as

$$\alpha_i^j(\Gamma_t) = \begin{cases} 1 + Z_i^j(\Gamma_t), & \text{if } Z_i^j(\Gamma_t) > 0 \\ 1, & \text{otherwise} \end{cases} \tag{1.3}$$

where

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

The control variables and the queue states must be positive, and the maximum capacity of the links must not be exceeded, i.e..

$$u_{ik}^j(\Gamma_t) \geq 0,\ Q_i^j(t) \geq 0 \text{ and } \sum_{n \in N} u_{ik}^n(\Gamma_t) \leq C_{ik}. \tag{1.4}$$

Finally, one more constraint is added to avoid that the expected incoming load be refused at one node. This is given by

$$\begin{cases} \sum_{l \in I(i)} u_{li}^j(\Gamma_t) \leq B_i^j - Q_i^j(t) - E\left\{A_i^j(\Gamma_t)\right\}, & \text{if } B_i^j - Q_i^j(t) > E\left\{A_i^j(\Gamma_t)\right\} \\ \sum_{l \in I(i)} u_{li}^j(\Gamma_t) = 0 & \text{otherwise.} \end{cases} \tag{1.5}$$

A suboptimal solution, which must be of integer type, can be found by means of mathematical programming.

1.3 Motivation

Among the previous models explained in the literature review, we are more interested by the macroscopic deterministic approach initiated in [31] which provides a system dynamic analysis rather than the queueing theory analysis. Furthermore we want to study routing as a decision making problem under the stochastic nature of traffic. Stochastic control is difficult to work with because it implies far too extensive computations whereby information needed from the physical system is unrealistic. The model in [1], even though providing a suboptimal solution, seems a good alternative to other models mainly because of the assumption of finite buffers which allows a more realistic analysis of network congestion. Its use of mathematical programming provides a simple optimization scheme that can easily be implemented.

In the first section of chapter 2, we work on a problem related to the finite multiple buffer model which restricts the access of the packets to the network, even under noncongested operations. The multiple buffer model is modified to prevent excessive packet loss and is extended to a single buffer node configuration. In the second part, we investigate a simple decentralized routing policy based on the proposed multiple buffer centralized routing.

In chapter 3, simulations conducted on a 5-node network allow comparisons of the models proposed in chapter 2 and [1]. These are tested under both stable (constant in time) and unbalanced (variable in time) traffic.

Finally, chapter 4 concludes this work by reviewing the results obtained and proposes some topics for further research.

CHAPTER 2

The proposed centralized and decentralized models

Many computer networks give priority to packets which have already been routed through the system (internal) because, if lost along their path, their retransmission can congest the system and decrease efficient channel utilization. Consequently, when traffic gets critically congested at some point of the network, users (external) often see access to their local node restricted or even restrained. It is obvious that both priority assignment and resources allocation play a key role in the performance of the general model and are worthwhile considering.

In the first section of this chapter, we propose a model for the multiple buffer case, as in [1], where priority is given to internal packets coming from other nodes and where the number of packets rejected by the network is taken into account by the optimization process. A single buffer node model is also investigated to illustrate the problems caused by fixed buffer sharing.

In the second section, we propose an intuitive model based on that of the previous multiple buffer model, which allows us to investigate the behavior of decentralized routing in communication systems. It is already well known that this type of control provides considerable practical advantages such as improved reliability. The purpose of this part is not to promote any of these models, but to evaluate the level of degradation in the global performance of the network when the control is distributed among several decision makers.

2.1 Prioritized centralized model

2.1.a Multiple buffer case

Adaptive routing process

The adaptive routing process considered is shown in figure 2.1. The time axis is discretized in short periods of length Δ. At time t, information about the state of the network is transferred from every node to the central node or control center which performs an optimization based on that information. Results are sent back to each node so that transmission rates of every channel can be updated. During period τ, which is the difference between the total time period Δ and the time required for updating the rates μ, transmission between nodes is kept at a constant rate. At the end of the interval Δ, all transmission has ceased and the whole process starts again. For the sake of simplicity, it is assumed that the optimization time μ is negligible, the packet length fixed and that any number of packets lower than the assigned lines capacity can be transmitted within the time period τ.

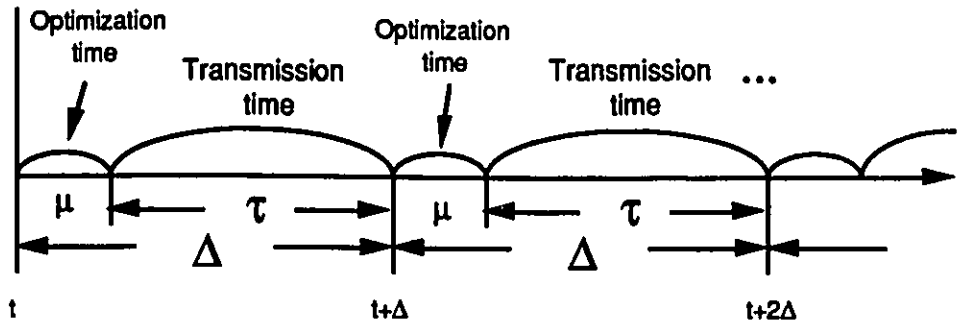


Figure 2.1: Adaptive routing process

We impose the restriction that a packet can not be received and retransmitted within the same time period. The time length Δ should be long

enough to avoid having to take a decision for almost every individual packet, a task physically impossible, due to the amount of information needed for a good optimization. On the other hand, Δ should be kept of a reasonable length because, as its value increases, static routing is performed and adaptivity is sacrificed.

Multiple buffer node

The internal architecture of a node, as shown in figure 2.2, consists of a router, I incomings links, M outgoing links and (N-1) buffers corresponding to all possible destinations. We want to favour a mechanism where the router sends the packets waiting in queue, while the incoming packets are received in buffers.

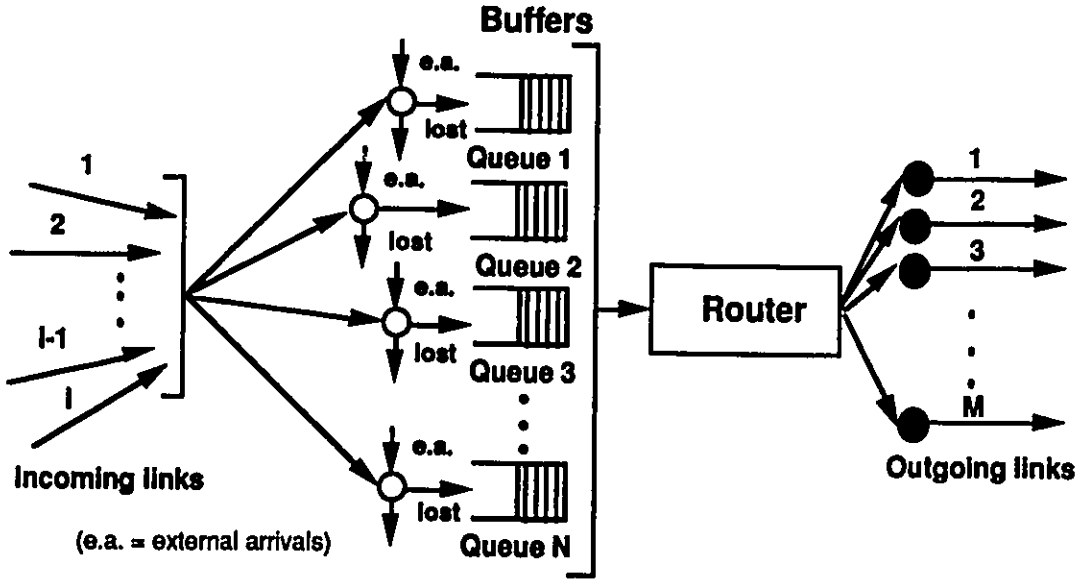


Figure 2.2: Multiple buffers node

It is assumed that the router, during time period (t,t+Δ), can at maximum, transmit the packets waiting in queue at time t. At every buffer, a

gate discards arriving packets if the buffer capacity has been exceeded. The transmission channels are considered to be of simplex type (one direction only).

Model with priority

The model in [1] is built such that, in a time period (t, t+Δ), it will only accept arrivals up to the maximum available buffer space at time t, even though the queue length may have been reduced by the router during this interval. This prudent policy leads directly to a deadlock situation (or premature saturation) in some buffers when the network is subject to an important load. This does not take place in all the buffers because of the modelling assumption that packets are not buffered once they get to their destination. It is natural to assume that we start delivering packets before other ones are received from incoming lines due to the physical transmission and propagation delays. The first priority is to empty the queues at time t so to avoid congestion when packets arrive.

The queue's dynamic equation under the above optimistic conditions is the difference between the number of packets waiting in queue at time t and that of the transmitted packets during the interval (t, t+Δ), to which is added the number of accepted arrivals into the remaining buffer space, i.e.

$$Q_i^j(t+\Delta) = Q_i^j(t) - \sum_{k \in E(i)} \left(u_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\right)$$

$$+ \min\left\{ \sum_{l \in I(i)} \left(u_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\right) + A_i^j(\Gamma_t),\ B_i^j - Q_i^j(t) + \sum_{k \in E(i)} \left(u_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\right) \right\}. \quad (2.1)$$

If the maximum buffer space B_i^j is exceeded, the number of lost packets, Ψ_i^j, is given by the following expression,

$$\Psi_i^j(t+\Delta) = \max\left\{0,\ Q_i^j(t) - \sum_{k \in E(i)} \left(u_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\right) + \sum_{l \in I(i)} \left(u_{li}^j(\Gamma_t) - \xi_{li}^j(\Gamma_t)\right) + A_i^j(\Gamma_t) - B_i^j\right\}. \quad (2.2)$$

Depending on the nature of the network, the external arrivals may follow different probability distributions. In this work, the external traffic is assumed to be Poisson distributed with mean

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

where $\lambda_i^j(t)$ is the instantaneous mean arrival rate at node i with destination j.

The transmission error ξ_{ik}^j is defined as a stochastic variable bounded by the number of packets transmitted through channel (i,k) and is assumed to have a binomial distribution with conditional probability

$$p\left(\xi_{ik}^j = m \mid u_{ik}^j\right) = \binom{u_{ik}^j}{m} \left(\tilde{\rho}_{ik}\right)^m \left(1 - \tilde{\rho}_{ik}\right)^{u_{ik}^j - m}$$

where $\tilde{\rho}_{ik}$ is the mean rate of transmission error associated with channel (i,k).

In equation (2.1), no preference is given to any of the internal or external arrivals. If priority were to be given to external arrivals, then no internal packets would get transmitted during a high traffic period thereby causing a deadlock situation. One solution is to give priority to internal packets. This way, once a packet gets into the network, it does not get discarded at some point because of poor buffer management, thereby avoiding a retransmission which

can slow down the system and diminish throughput performance significantly.

On the other hand, if no consideration is given to external packets, their access to the network might get blocked. Therefore, it might be interesting to allocate a portion of the buffer space solely to the external load, under the condition that the priority of the internal packets is preserved. In most models, this allocated portion of the buffer is chosen to be the expected value of the external arrivals. Using this scheme, it has been noticed that even under noncongested conditions in the network, the routing policy has the tendency to fill the minimum length queue with the consequence that some buffers are full most of the time while some others are underutilized. And in the case where the number of external arrivals exceeds that of their expected value, some packets will be discarded. This is regretful since the use of the underutilized buffers could have prevented such a loss.

We want to investigate two solutions to this problem. The first one is to keep the same multiple buffer node architecture while modifying the routing constraints. Those modifications ensure that the external arrivals are accepted at every node with a certain probability without significantly restricting the internal packets transmission through the network.

The second solution investigated is to consider a node configuration of one single buffer where all packets are stored at the same location, independently of their destination, thereby avoiding buffer waste. This is explained in the other part of this section.

First solution proposed

The problem with minimizing only the expected value of the buffered queue length as in [1] is keeping track of the lost packets. It might even be optimal under certain conditions to lose packets in order to diminish the total queue length. We propose to add those packets as a weighted term in the objective function written as

$$J(t) = \sum_{ij} \alpha_i^j E\left\{Q_i^j(t+\Delta) \mid Q_i^j(t)\right\} + \sum_{ij} \beta_i^j E\left\{\Psi_i^j(t+\Delta) \mid Q_i^j(t)\right\}$$

$$= \sum_{ij} \alpha_i^j E\left\{Q_i^{j^*}(t+\Delta) \mid Q_i^j(t)\right\} + \sum_{ij} (\beta_i^j - \alpha_i^j) E\left\{\Psi_i^j(t+\Delta) \mid Q_i^j(t)\right\} \qquad (2.3)$$

where $Q_i^{j^*}(t+\Delta)$ is the unbuffered queue at node i with destination j. This is a combination of the accepted and lost packets, whose expected value is given by

$$E\left\{Q_i^{j^*}(t+\Delta) \mid Q_i^j(t)\right\} = Q_i^j(t) + \sum_{l \in I(i)} u_{il}^j(\Gamma_t)(1-\tilde{\rho}_{il}) - \sum_{k \in E(i)} u_{ik}^j(\Gamma_t)(1-\tilde{\rho}_{ik}) + E\left\{A_i^j(\Gamma_t)\right\}. \qquad (2.4)$$

In the above expression, the mean number of packets which failed to be transmitted by channel (i,l), ρ_{il}^j is, according to the binomial distribution, proportional to the number of packets sent through channel (i,k), i.e.

$$\rho_{ik}^j(\Gamma_t) = E\left\{ \xi_{ik}^j(\Gamma_t) \mid u_{ik}^j(\Gamma_t) \right\} = \tilde{\rho}_{ik}\, u_{ik}^j(\Gamma_t).$$

The objective function consists of a linear part (the unbuffered queue) and a nonlinear term (the queue lost). If equal weight is given to the total lost packets and the total unbuffered queue in the minimization process, i.e.

$\alpha_i^j = \beta_i^j$, then the second term becomes null and a linear objective function is obtained which can be solved by means of integer linear programming.

Constraints

The constraints associated with this model are the following. The control variable and the queue size are positive (lower bound)

$$u_{ij}^k(\Gamma_t) \geq 0 \text{ and } Q_i^j(t) \geq 0 . \tag{2.5}$$

As mentioned previously, node i can only send, during time slot Γ_t, packets already stored in queue (i,j) at time t. External arrivals wait until the next time slot before they are routed. This condition is given by

$$\sum_{j \in E(i)} u_{ij}^k(\Gamma_t) \leq Q_i^k . \tag{2.6}$$

Packets transmitted through channel (i,j) are limited by its maximum capacity, i.e.

$$\sum_{k \in E(i)} u_{ij}^k(\Gamma_t) \leq C_{ij} . \tag{2.7}$$

As noted earlier, a buffer constraint which redirects the traffic when an unexpected burst of arrivals occurs is necessary to avoid excessive loss of external arrivals. We define this constraint as the probability that the unbuffered queue will be accepted into the buffer. We ignore channel errors to simplify the analysis. This expression is written as:

$$P\left(Q_i^j(t+\Delta) \le B_i^j\right) \ge \eta_i^j$$

i.e.

$$P\left(Q_i^j(t) + \sum_{l \in I(i)} u_{li}^j(\Gamma_t) - \sum_{k \in E(i)} u_{ik}^j(\Gamma_t) + A_i^j(\Gamma_t) \le B_i^j\right) \ge \eta_i^j$$

which, considering the physical limitations of the network, is stated as

$$\sum_{l \in I(i)} u_{li}^j(\Gamma_t) - \sum_{k \in E(i)} u_{ik}^j(\Gamma_t) \le S_i^j(\Gamma_t) \tag{2.8}$$

$$\text{where} \quad S_i^j(\Gamma_t) = \left\{ \begin{array}{ll} B_i^j - Q_i^j(t) - \gamma_i^j, & \text{if } Q_i^j(t) + \gamma_i^j \le B_i^j \\ 0 & \text{otherwise.} \end{array} \right\},$$

$$\tilde{\lambda}_i^j = E\left(A_i^j(\Gamma_t)\right) \text{ and } P\left(A_i^j(\Gamma_t) \le \gamma_i^j\right) \ge \eta_i^j .$$

η_i^j is called the priority scheme parameter at node i which determines γ_i^j, the buffer space to be reserved for external arrivals of type j.

One advantage of this scheme is that when γ_i^j is excessive, i.e. η_i^j and/or the expected external traffic $\tilde{\lambda}_i^j$ are too high, the buffer space constraint requirements are ignored and the condition settles to the '0 otherwise' argument. This preserves the priority of internal arrivals over external arrivals. The default value is set to 0 because if lower, it forces the node to transmit more than it receives, an impossible behavior in a scenario where congestion is present at every node. In this situation the routing policy fails to take any action.

If $\eta = 0$, full priority is given to internal arrivals. As η tends to 1, more buffer space is dedicated for external arrivals until the buffer size is exceeded.

Clearly η cannot be optimal under varying network parameters if it remains constant during the operation. However no attempt is made in this work to consider a way to adapt this parameter to variations in the network.

Weighting factor

The weighting factor is also modified in agreement with the last constraint as

$$\alpha_i^j = 1 + \frac{\gamma_i^j(\Gamma_i) + Q_i^j(t)}{B_i^j} \quad . \tag{2.9}$$

More emphasis is put during the optimization on queues which are longer and/or are expecting to receive a high external load, in order to prevent congestion before it occurs.

2.1.b Single buffer case *(Second solution proposed)*

The adaptive routing mechanism is the same as for the multiple buffer case. The node configuration, shown in figure 2.3, differs from the previous one by its unique buffer B_i containing one global queue Q_i which is the union of all the waiting packets destined to the other nodes. A single gate is used for the flow control of the arrivals at the buffer.

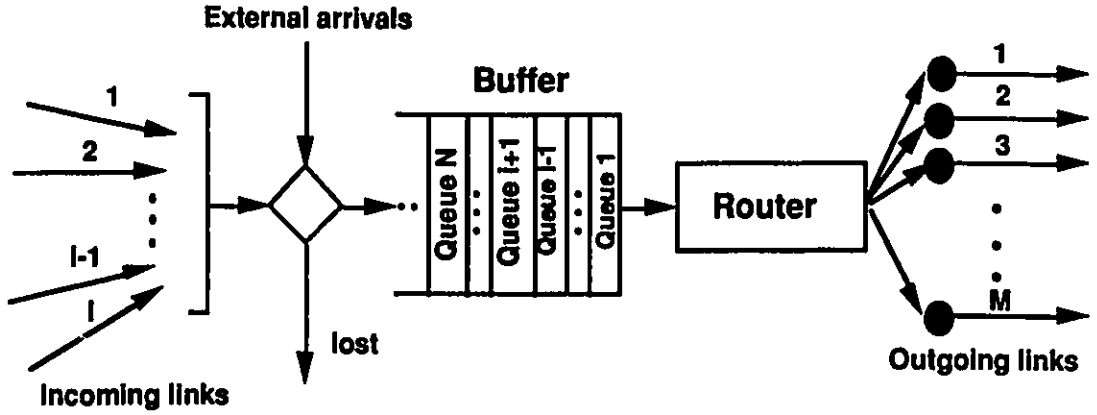


Figure 2.3: Single buffer node

The multiple buffer problem seen in the previous part is considered a special case of the single buffer model. It is equivalent to a partitioning of the memory space into N-1 sections, each fixed in time and dedicated to a specific destination. The fixed multiple buffer case is an extension of the Kleinrock's model which associates queues with outgoing lines instead of assigning queues to destinations as in the later models. The first models, especially the queuing theory ones, allocated infinite storage to waiting queues. Later on, finite queue buffers were assumed with the same "one queue to one line" concept. Finally when models associated queues to destinations instead of lines, the idea of "as many buffers as there are queues" remained. In this

part, the single buffer finite case is studied where a buffer is paired with every node. This approach is not seen often in the literature concerning macroscopic models. However this is a more realistic scenario, the reason being as the number of nodes increases so does the number of buffers dedicated to those new destinations. Clearly when many small dedicated buffers are used, underutilization and waste of space is considerably larger than with one general big buffer. A unique buffer configuration makes it easier to respond to the maximum total demand rather than trying to meet individual maximal requests all of the time in a multiple buffer situation, since

$$\max\left\{0, \sum_{j} (\text{Demand } j - \text{Available } j)\right\} \le \sum_{j} \max\{0, (\text{Demand } j - \text{Available } j)\}$$

Similarly to the previous model, the buffered queue length at time $(t + \Delta)$ and the corresponding lost packets are given by

$$Q_i(t+\Delta) = Q_i(t) - \sum_{j} \sum_{k \in E(i)} \left(u_{ik}^{j}(\Gamma_t) - \xi_{ik}^{j}(\Gamma_t)\right) \tag{2.10}$$

$$+ \min\left\{\sum_{j} \sum_{l \in I(i)} \left(u_{li}^{j}(\Gamma_t) - \xi_{li}^{j}(\Gamma_t)\right) + \sum_{j} A_i^{j}(\Gamma_t),\ B_i - Q_i(t) + \sum_{j} \sum_{k \in E(i)} \left(u_{ik}^{j}(\Gamma_t) - \xi_{ik}^{j}(\Gamma_t)\right)\right\}.$$

and (2.11)

$$\psi_i(t+\Delta) = \max\left\{0, Q_i(t) - \sum_{j} \sum_{k \in E(i)} \left(u_{ik}^{j}(\Gamma_t) - \xi_{ik}^{j}(\Gamma_t)\right) + \sum_{j} \sum_{l \in I(i)} \left(u_{li}^{j}(\Gamma_t) - \xi_{li}^{j}(\Gamma_t)\right) + \sum_{j} A_i^{j}(\Gamma_t) - B_i\right\}.$$

In a discrete event simulation, the destination of the lost packets is known since they arrive individually. However, it is not the case here. In the simulation performed in chapter 3, packets arrive in groups, thus making it hard to discern the destination of the discarded messages. Therefore, it is

necessary to assume that those lost packets are distributed following a multivariable hypergeometric.

$$P\left(x_i^1, \ldots, x_i^{i-1}, x_i^{i+1}, \ldots, x_i^N \mid \psi_i(t+\Delta)\right) = \frac{\binom{A_i^1(\Gamma_t)}{x_i^1} \cdots \binom{A_i^{i-1}(\Gamma_t)}{x_i^{i-1}} \binom{A_i^{i+1}(\Gamma_t)}{x_i^{i+1}} \cdots \binom{A_i^N(\Gamma_t)}{x_i^N}}{\binom{\sum_{j \neq i} A_i^j(\Gamma_t)}{\psi_i(t+\Delta)}}$$

where

$$x_i^j \leq A_i^j(\Gamma_t) \text{ and } \sum_j x_i^j = \psi_i(t+\Delta).$$

The x_i^j 's represent the number of discarded packets at node i destined to node j

Optimization

An objective function similar to the one proposed in the centralized model is used i.e.

$$J(t) = \sum_{ij} \alpha_i^j E\left\{Q_i^{j^*}(t+\Delta) \mid Q_i(t)\right\} + \sum_i (\beta_i - \sum_j \alpha_i^j) E\left\{\psi_i(t+\Delta) \mid Q_i(t)\right\} \qquad (2.12)$$

with the same weighting factor α_i^j. During simulation, equal weight is given to the unbuffered queue and to the packets lost in order to linearize the objective function (as described in the first solution proposed).

Constraints (2.5), (2.6) and (2.7) remain the same. Since the single buffer model is an alternative to the multiple buffer constraint (2.8), the latter is replaced by:

$$E\left\{\sum_j Q_i^{j^*}(t+\Delta) \mid Q_i(t)\right\} \le B_i$$

which is equivalent to:

$$\sum_j \left(\sum_{l \in I(i)} u_{li}^j(\Gamma_t) - \sum_{k \in E(i)} u_{ik}^j(\Gamma_t) \right) \le S_i(\Gamma_t) \qquad (2.13)$$

where $$S_i(\Gamma_t) = \left\{ \begin{array}{ll} B_i - Q_i(t) - \tilde{\lambda}_i, & \text{if } Q_i(t) + \tilde{\lambda}_i \le B_i \\ 0 & \text{otherwise.} \end{array} \right\},$$

$$\text{and } \tilde{\lambda}_i = \sum_j \tilde{\lambda}_i^j .$$

2.2. Decentralized routing

In this section, an intuitive scheme to decentralize the previous centralized model is investigated. Only the multiple buffer case is considered.

Information structure and decision place

During the normal operation of the centralized routing scheme, information about topology, queue states and expected users utilization are sent from each node to the control center of the network. As the number of nodes increases, the amount of information to be relayed to the central node limits the time available for message transmission and may cause long processing time delays during the optimization process. In order to obtain a solution by the centralized routing with multiple buffers, an optimization has to be done with a tableau of (N-1)*L variables and L+2N*(N+1) constraints. For a network of 100 nodes, a system of at least 20 000 variables would have to be solved. Decentralized routing lightens the information load exchange in wide area networks and decreases the processing time.

A scheme where decisions are taken at the node level is examined. Information about its own state as well as that of its neighbors is available to every node. Packets are transmitted after a data exchange between the sender and the receiver only, about queue states, buffer and channel limitations, and mean external arrivals. No regard is given to the rest of the network. Every subcontroller (decision-maker) makes a contribution to the routing process by simultaneously taking a decision based on different on-line information coming from adjacent subcontrollers.

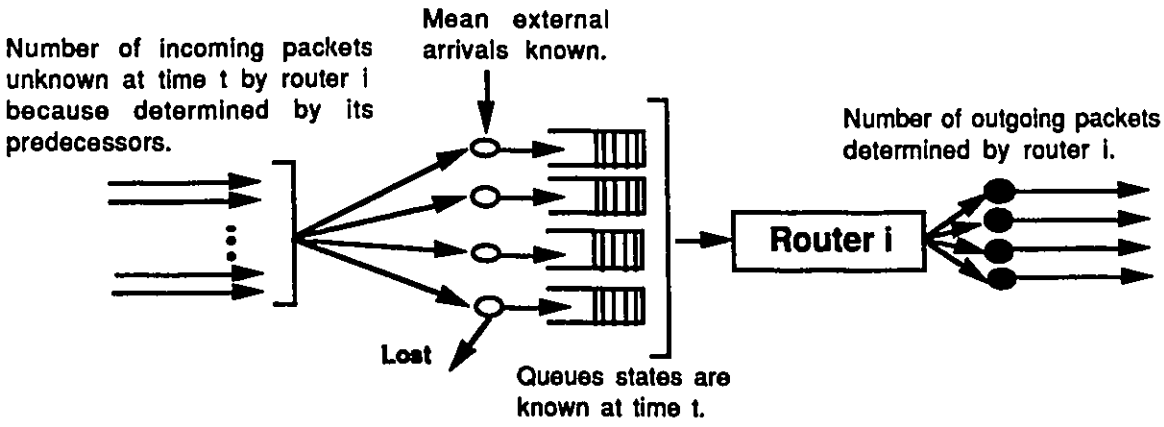


Figure 2.4: Incoming packets unknown at time t

This method avoids hierarchical information structure where some decision makers are favoured to the detriment of others.

Optimization scheme

The information structure $I_i(t)$ of node i at time t consists of the main parameters of the node itself and its neighbors, i.e.

$$I_i(t) = \left[\left\{ Q_i^j(t),\ A_i^j(\Gamma_t),\ \tilde{\rho}_{ik}^j(\Gamma_t),\ B_i^j,\ C_{ik} \right\} \cup \left\{ B_k^j,\ Q_k^j(t),\ A_k^j(\Gamma_t) \right\};\ k \in E(i);\ j \in N \right]$$

By definition of the problem, each node is responsible during time period $(t, t+\Delta)$ for transmitting packets contained in its buffer at time t. The control vector of packets transmitted from node i during Γ_t is defined as:

$$u_i(\Gamma_t) = \left\{ u_{ik}^j(\Gamma_t);\ k \in E(i);\ j \in N \right\}$$

One popular approach in the current literature is to optimize the global network performance where the control vector is multivariate. However we

are more interested in investigating a scheme where each controller wants to optimize its own performance. The local objective function at node i may be defined as the sum of its weighted unbuffered queue length, i.e.

$$J_i(t) = \sum_j \alpha_i^j E\left\{Q_i^j(t+\Delta) | I_i(t)\right\}$$

Every node would minimize this objective function under its control vector u_i . So the global decentralized performance would be expressed as

$$J_d(t) = \sum_i \min_{u_i(\Gamma_t)} J_i(t)$$

as opposed to the previous centralized objective function

$$J_c(t) = \min_{u(\Gamma_t)} \sum_i J_i(t)$$

There is a problem with this type of decentralized objective function. In the unbuffered queue dynamic equation (2.4), the total amount of incoming internal arrivals is unknown to router i since it is simultaneously determined by its predecessors. Therefore it is impossible for a node i to take into account this amount during the optimization process. So the objective function of the router i is reduced by dropping the terms representing the external arrivals and the queue length, to the term representing the outgoing packets to the successor nodes only, i.e.

$$J_i(t) = E\left\{ - \sum_j \sum_{k \in E(i)} \alpha_i^j \left[u_{ik}^j(\Gamma_t) - \xi_{ik}^j(\Gamma_t)\right] \middle| I_i(t)\right\} = - \sum_j \sum_{k \in E(i)} \alpha_i^j u_{ik}^j(\Gamma_t)\left(1 - \tilde{\rho}_{ik}\right)$$

The minimization of this linear combination of control variables at node i is equivalent to maximizing the throughput at this node.

If the impact of the decision taken by node i on its neighbors is not reflected in the objective function, a greedy policy is encouraged, thereby provoking uncoordination. For this reason, a term counterbalancing the load inflicted upon the neighboring node queues is included. The minimization of this function becomes

$$\min_{u_i(\Gamma_t)} J_i(t) = \min_{u_i(\Gamma_t)} \left\{ -\sum_j \sum_{k \in E(i)} \alpha_i^j \, u_{ik}^j(\Gamma_t)\left(1 - \tilde{\rho}_{ik}\right) + \sum_j \sum_{k \in E(i);\, j \neq k} \alpha_k^j \, u_{ik}^j(\Gamma_t)\left(1 - \tilde{\rho}_{ik}\right) \right\}$$

which is equivalent to

$$\max_{u_i(\Gamma_t)} \left\{ \sum_j \sum_{k \in E(i)} \alpha_i^k \, u_{ik}^k(\Gamma_t)\left(1 - \tilde{\rho}_{ik}\right) + \sum_j \sum_{k \in E(i);\, j \neq k} \left(\alpha_i^j - \alpha_k^j\right) \, u_{ik}^j(\Gamma_t)\left(1 - \tilde{\rho}_{ik}\right) \right\} \quad (2.14)$$

Physically this means that every node will try to find a trade off solution for conflicting interests i.e. maximization of its throughput and minimization of the future load incured to its immediate neighboring nodes.

Constraints

The lower bound (2.5), queue (2.6) and channel (2.7) constraints must still be observed. A problem with the buffer coordination constraint (2.8) still exists. Since the incoming packets are under the jurisdiction of node i, it is impossible to make sure with a certain probability that incoming traffic can be accepted wholly as in the previous centralized models. A solution in two steps is provided:

1- During the optimization process at every node, a constraint which puts an upper bound on the routing variables is included, i.e.

$$u_{ij}^{k}(\Gamma_t) \leq B_j^k - \gamma_j^k \qquad (2.15)$$

with γ_j^k defined as in section 2.1. Even when this upper bound is respected, uncooordination might arise in some cases where one node is heavily surrounded. For example, as figure 2.5 illustrates, node 1, 2 and 3 estimated during their individual optimization that the numbers of packets to be sent to destination 4 are respectively 20, 25 and 25. While every individual node constraint is respected, the total does not fit into the remaining buffer space at node 4.

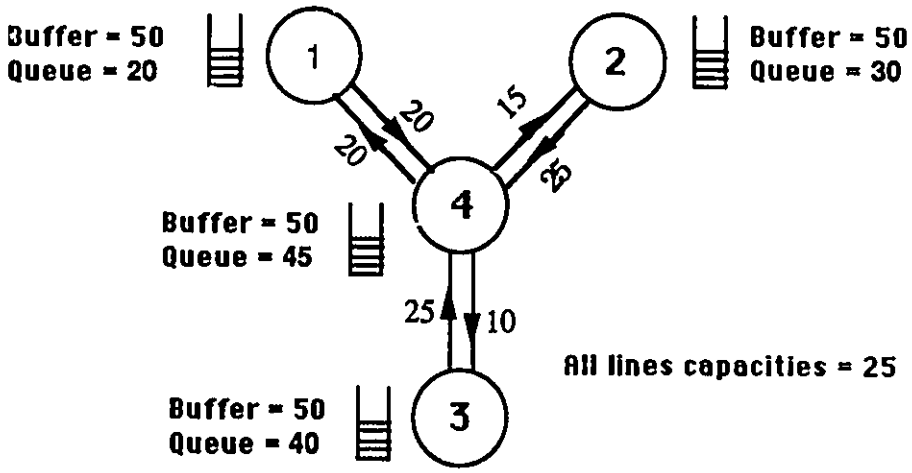


Figure 2.5: Uncoordination

2- In reaction to this situation, it is required that the results of every local optimization be submitted to the neighbors. If the total proposed arrivals is less or equal than the buffer space allocated to internal arrivals, an acknowledgment is sent and the transmission begins. Otherwise, the values are reestimated by the receiving nodes and passed back to the transmitting ones. This reestimation is arbitrary and can be deduced from logical arguments, an optimization scheme or even randomly. In this work, we

simply consider that the number of packets transmitted should be in proportion to the initial proposed estimates. The buffer space allocated to internal packets during time period Γ_t corresponds to the upper bound in (2.15).

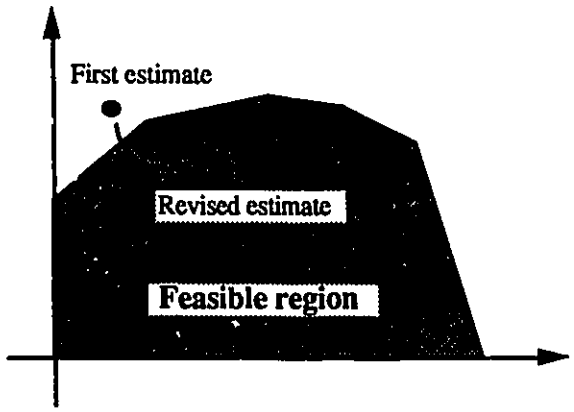


Figure 2.6: Correction of the first estimate

Even though this topic is not pursued in this thesis, consultation among nodes after the optimization may be used as a reliability mechanism. If no acknowledgment or reestimated values are sent back, the silent nodes are considered to be out of order.

Hence the combined policy motivates each node to maximize throughput while minimizing the future load in a sub-optimal manner based on adjacent node information. Corrections, when required, map the routing vector into or as close as possible to the feasible region to prevent excessive packet loss when buffer allocation is exceeded (fig. 2.6).

CHAPTER 3
Simulation

3.1 Simulation considerations

Simulation of model [1], and the proposed centralized and decentralized models of chapter 2 was conducted on the network topology of figure 3.1 using an HP 9000 workstation. The computer program listings, written in FORTRAN, are annexed in the appendix. The flowcharts of the programs are presented in section 3.2. Five comparisons of the results are carried out in section 3.3 in order to illustrate the impact of the various modifications. The first four analyses illustrate the particularities of the models seen in chapter 2. They are characterized as follow:

i) comparison of the multiple buffer centralized models of [1] and section 2.1.

ii) effect of the priority scheme on the multiple buffer centralized model.

iii) comparison of the single and multiple buffer centralized models.

iv) comparison of the multiple buffer centralized and decentralized models.

Results are obtained under a balanced traffic, which means that the mean external arrival rate is kept constant at every buffer of every node throughout the simulation. This stability makes the analysis easier and illustrates the main differences between the models. Each simulation is repeated many times so that the results are within a 95% confidence interval. This is to avoid having to judge performances based on biased results from one

experiment that depends on a sequence of pseudo-random generated numbers. All results obtained in the following sections have a maximum standard deviation of 2%.

The fifth simulation is conducted under very unstable external arrival rate to illustrate the reaction of the models in a realistic situation. The mean external arrivals (number of arrivals/time period) are selected according to a discrete uniform distribution and are corrupted according to a Poisson distribution.

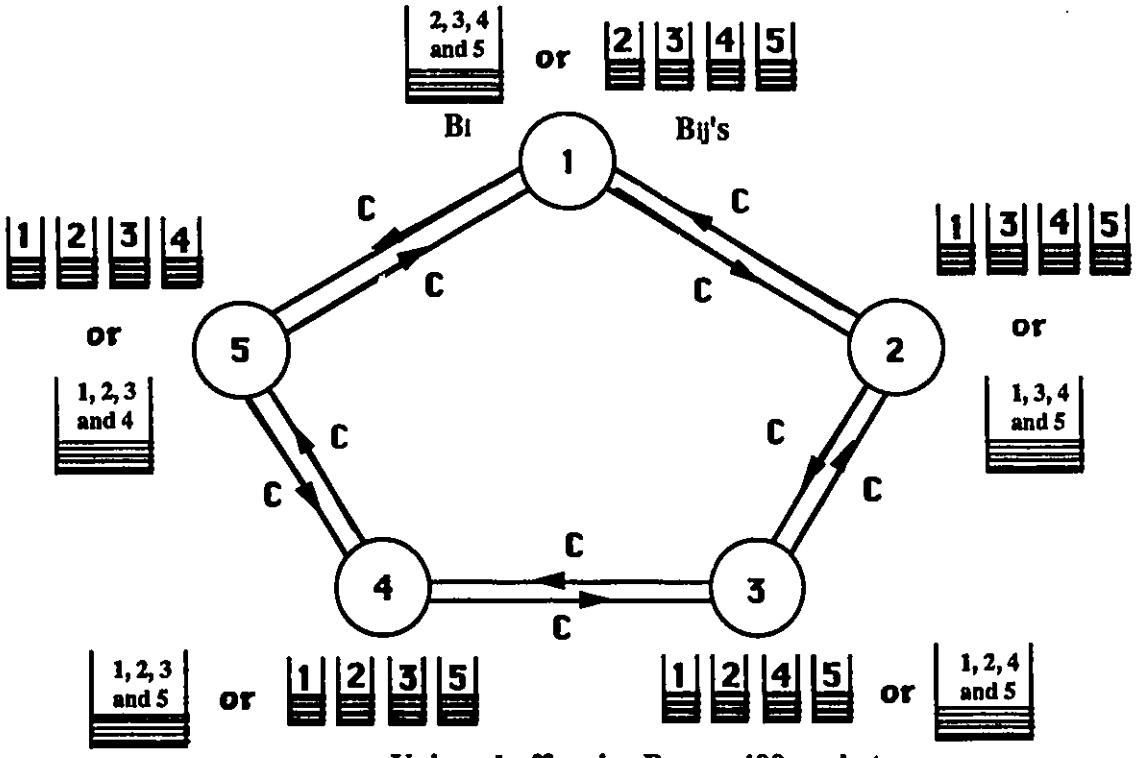


Figure 3.1: Network topology of 5 nodes

The simulation is conducted on a network of 5 nodes. Its relatively small size permits the analysis of the centralized models within a decent

computing time. For instance, each centralized multiple buffer optimization takes an average of 0.021 seconds. The single buffer model takes 0.015 seconds while each of the 5 nodal optimizations of the decentralized model needs less than 0.003 seconds. A ring topology is favoured because of its existing symmetry. Two incoming and outgoing links as well as two first and two second neighbors are associated with every node.

Total queue length, lost packets, throughput and average delay are the performance indicators used for the analysis of results. The total queue length is the sum of all the queues present at every buffer of every node over the entire simulation. The lost packets represent the total arrivals discarded at the buffer gate. The throughput is the number of submitted packets which have been successfully delivered to destination within one time period. In this case, the maximum throughput corresponds to the sum of all the channel capacities, i.e. 1000 packets/time period. Finally, the delay may be interpreted as the average number of hops made by a packet, which has never been discarded, to reach its destination.

3.2 Simulation flowchart

Figure 3.2 represents the general routing simulation flowchart, while figure 3.3 illustrates how the appropriate linear program is selected for the routing decision process in a multiple buffer node architecture.

The first step of the simulation is to describe the network topology. The initial state of the queues is then arbitrarily chosen. In this case, they are set to half the buffer size. The external arrival rates are then generated. For the balanced traffic case, the mean external arrival rates are kept constant throughout the simulation, while for the unbalanced case, the mean values are selected from an IMSL subroutine called RNUND generating discrete uniform numbers. The subroutine RNPOI is used to modify these mean values according to a Poisson distribution.

An optimization, using linear programming [17], is performed according to the state of the queues, the routing and weighting factor selected, which is detailed in figure 3.3 for the multiple buffer node. After a decision has been taken, the routing is executed and the queues are updated. This routing pattern is repeated for k time intervals. After completion, the performance indicators are computed and stored for analysis.

For the simulations with balanced traffic at every node, this routing process is repeated NITER times. The seeds differ each time but the initial conditions remains the same.

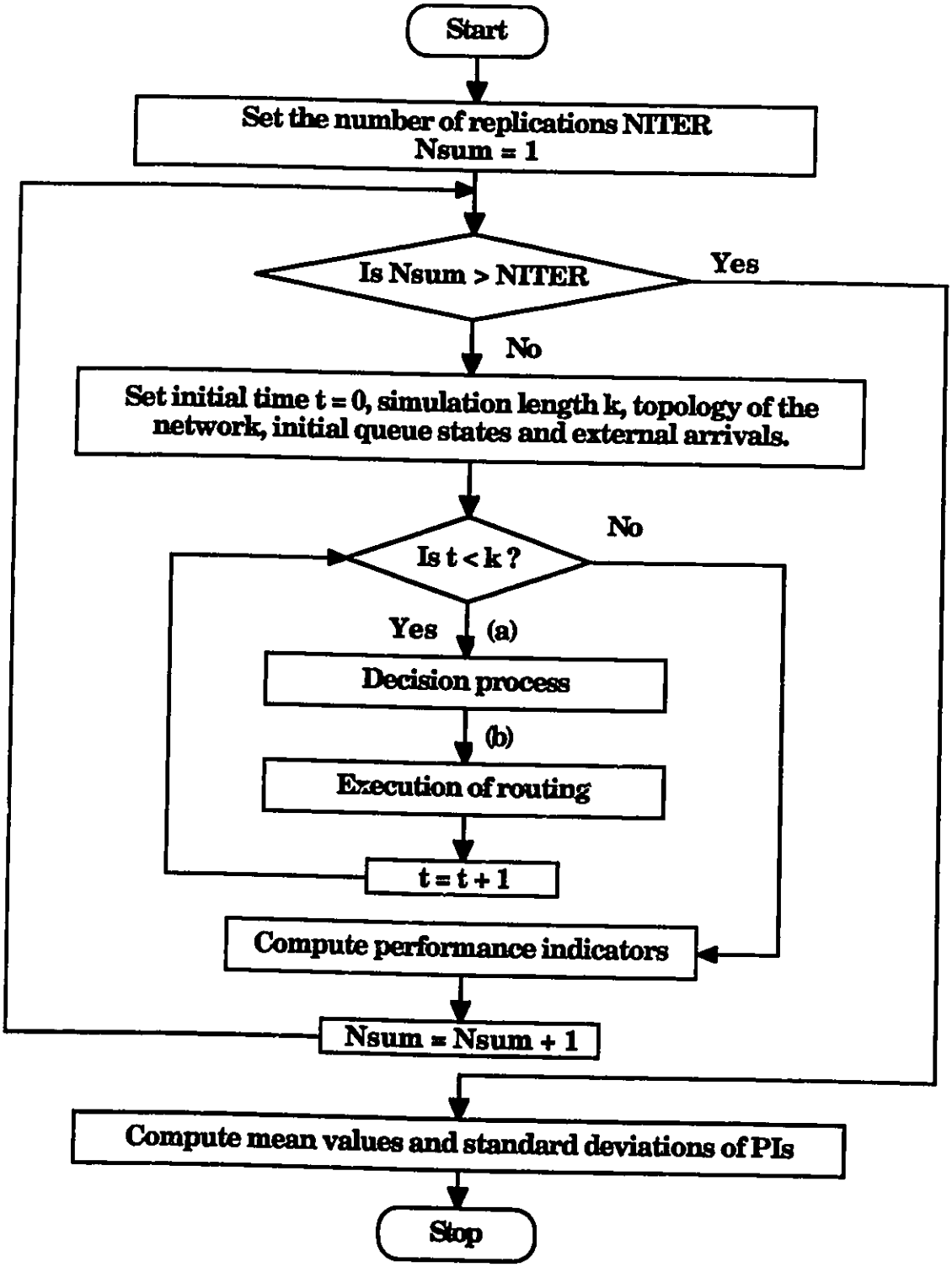


Figure 3.2: Flowchart of the routing simulation

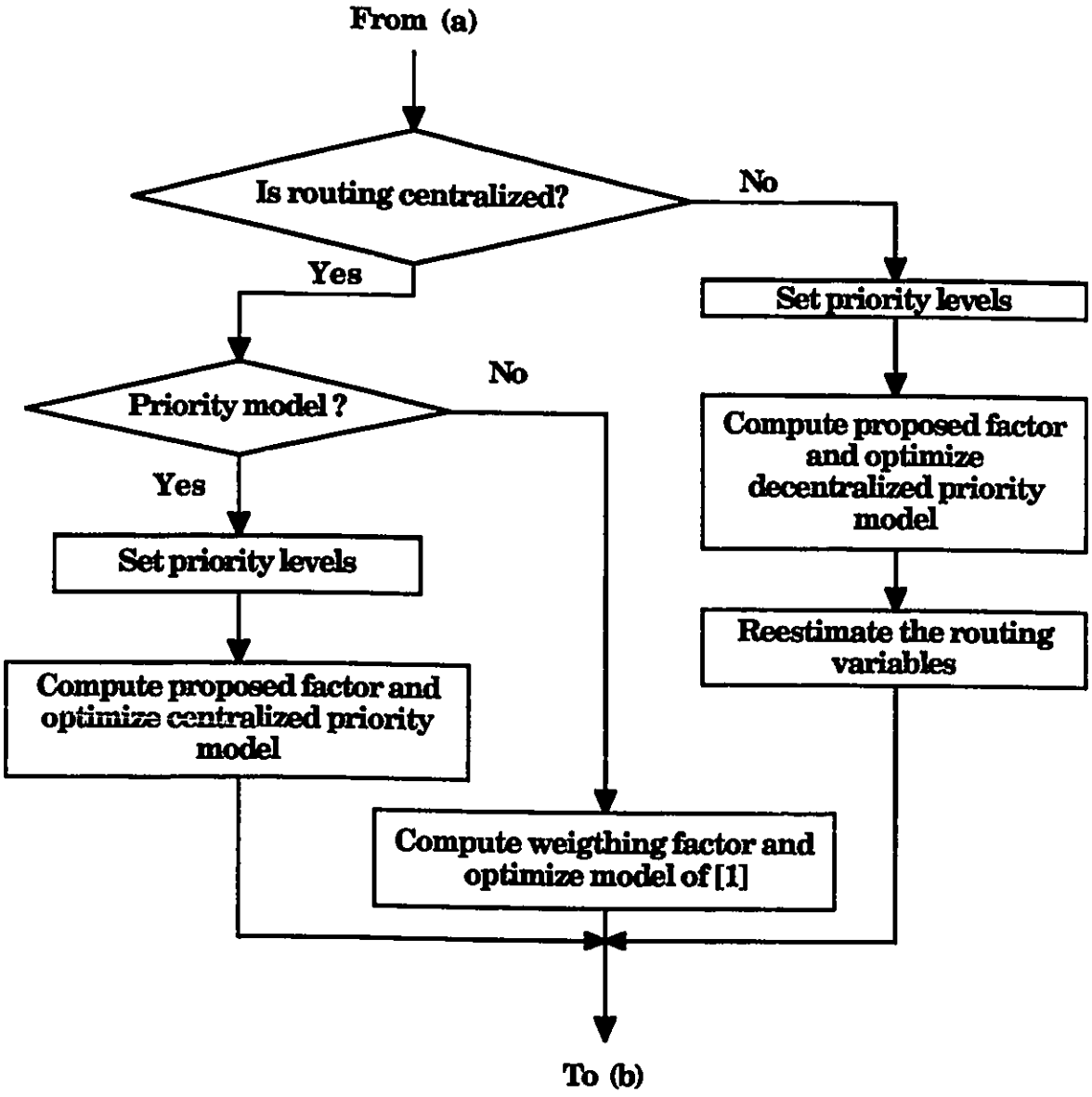


Figure 3.3: Decision process for the multiple buffer models

3.3 Simulation results

3.3.a Comparison of the multiple buffer centralized model and [1]

As mentioned earlier, results for this comparison are obtained under a balanced traffic, the mean arrival value of which increases over the range [1,100] packets/time period, i.e. the entire buffer range. It is important to mention that the mean arrival rates are measured in terms of packets/time period. We compare the model in [1] with a priority model of 50%, i.e. where η, the probability of accepting an entire unbuffered queue is equal to 0.5.

In figure 3.4, we notice that the model [1] looses more packets during the whole simulation than the proposed centralized model. We estimate that congestion occurs for [1] and the proposed model around 25 and 32 packets/time period respectively, where the graph lines break down. Why does the model [1] get congested before the other one? The following discussion provides an explanation of the main differences between the two schemes.

Both models tend to keep the queue length short which, under low load (no congestion), corresponds to a minimum hop problem. With the topology of figure 3.1, a node should send as many packets as possible to its first neighbors (the intended first destination) where messages will not be buffered and then send messages to the second neighbors by the minimum hop path (messages are buffered at the intermediate node). Those transmissions are limited by the line capacity and the buffer size. With the network topology of figure 3.1, the line capacity is exceeded before the buffers are filled since there are 2 outgoing lines of 100 packets/time unit each, while 4 buffers provide a total storage space of 400 packets.

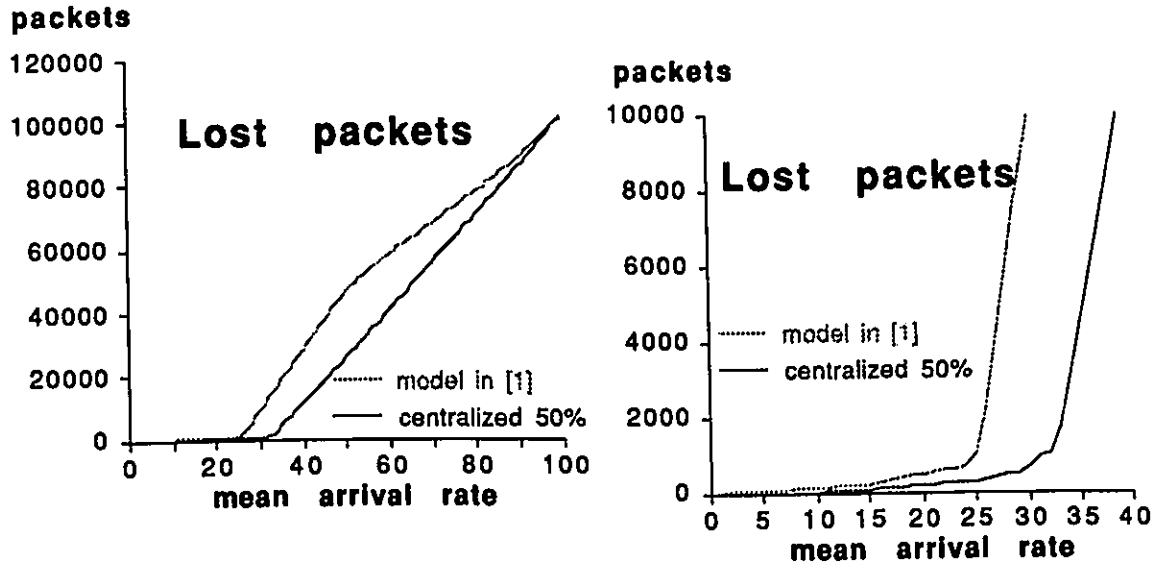


Figure 3.4: Lost packets of both models

In spite of the resemblance between those two models, congestion occurs in them for different reasons. In the proposed centralized model, performance degradation occurs when the maximum line capacity is reached. It happens at time t when the load in queue to be transmitted is at least equal to the total outgoing capacities, i.e.

$$\sum_{j} Q_i^j(t) \geq \sum_{k} C_{ik} .$$

For the sake of simplicity, let's assume that the external arrivals λ are constant in time and not corrupted. The approximated threshold arrival value at which congestion begins at one node is given by the following relation:

$$6\lambda_{thr} = 2C$$

where λ_{thr} is the mean external arrival rate at every buffer and C is the channel capacity. This can be easily deduced since each of the 4 buffers at each node i receives λ_{thr} packets from external arrivals during one time period. In the same time interval, node i expects to receive λ_{thr} from each of its two predecessors, destined to its two successors. With C = 100 packets/time unit, the threshold λ_{thr} is around 33.3 packets/time unit. We expect that above this value, the packet loss will increase linearly with the mean arrival. This is confirmed in figure 3.4, where the lost packet curve breaks down around a mean arrival rate of 32 packets/time unit. Some packets are lost before, due to the stochastic nature of the traffic.

In model [1], the threshold λ_{thr} is not 33.3. As mentioned previously, this model was modified because of inequality (1.5) which causes a premature saturation beginning at those buffers with packets destined to their immediate neighbors. The reason for this problem is the following:

Internal arrivals during (t, t+1) (λ)	$\leq$	Available buffer space at time t ($B - 2\lambda$)	-	Expected external arrivals during (t, t+1) (λ)

This inequality follows directly from (1.5). At time t, the length of the queue is of 2λ, i.e. one λ coming from a predecessor and the other from the external arrivals. Hence the buffer space available at time t is $(B-2\lambda)$. During time period (t,t+1) the expected external arrivals is λ. Normally, in this interval, the load incoming from internal lines would also be λ. Above a certain value of λ, λ_{thr}, not all packets can be accepted and some must be diverted elsewhere which causes congestion to start building up . With B equal

to 100 packets, the expected degradation of the performance is around a mean external rate of 25 packets/time period, which closely corresponds to the simulation results obtained. Clearly, the premature saturation occurring in this model makes it less efficient under high traffic load.

The total queue length is lower under small traffic in the proposed centralized model because of the proposed weighting factor which favours clearing longer queues even when the network is not congested. Under congestion, the inverse occurs because of the excessive packet loss of the model in [1]. As congestion increases, both schemes tend to converge toward a common solution.

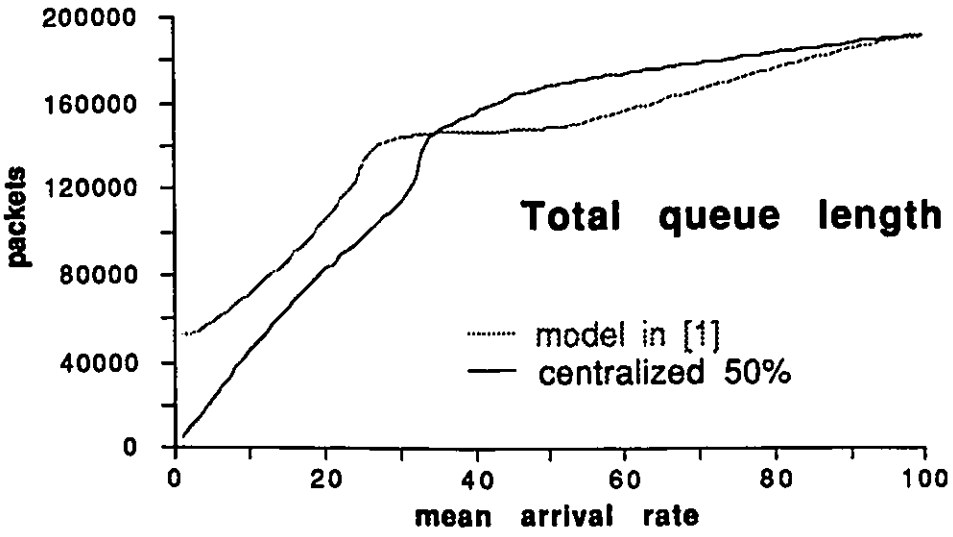


Figure 3.5: Total queue length

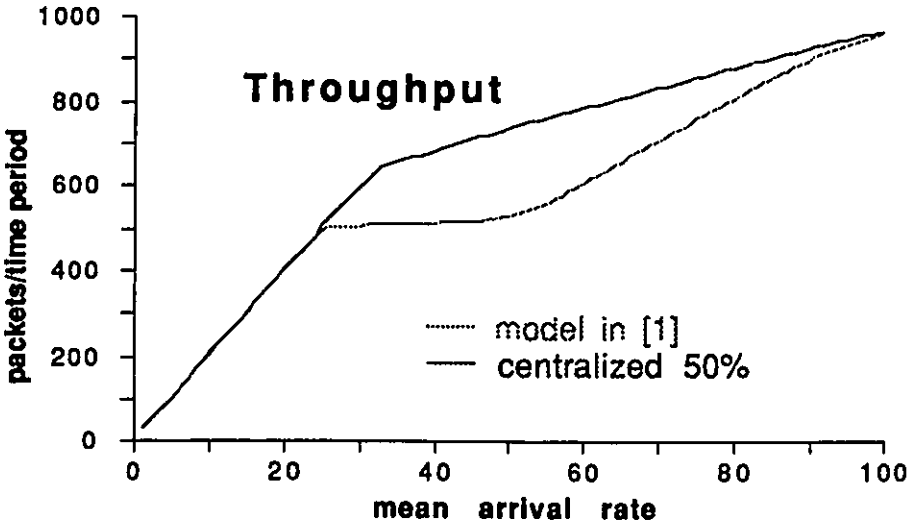


Figure 3.6: Throughput

As a consequence of the packet loss, the throughput of [1] reflects the saturation in the section [25,50], where a plateau is reached. The proposed model throughput increases constantly until the maximum channel capacity is reached, where more nearest neighbor packets are sent to the detriment of others.

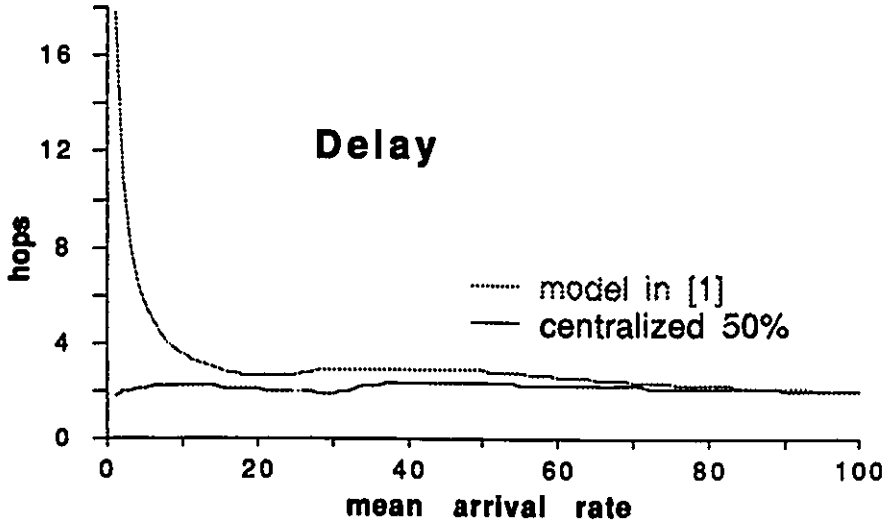


Figure 3.7: Delay

3.3.b Effect of priority scheme over centralized model

We define a priority scheme to be the selection of a parameter η which dedicates buffer space to external arrivals. We refer to (100%*η) as the level of priority accorded to external arrivals over internal arrivals. For the sake of illustration, three different significant levels are considered:

- full priority given to internal arrivals (0%)
- mixed priority where the buffer space reserved is around the expected value of the external arrivals (50%)
- reserve more buffer space for external arrivals than their expected value (95%)

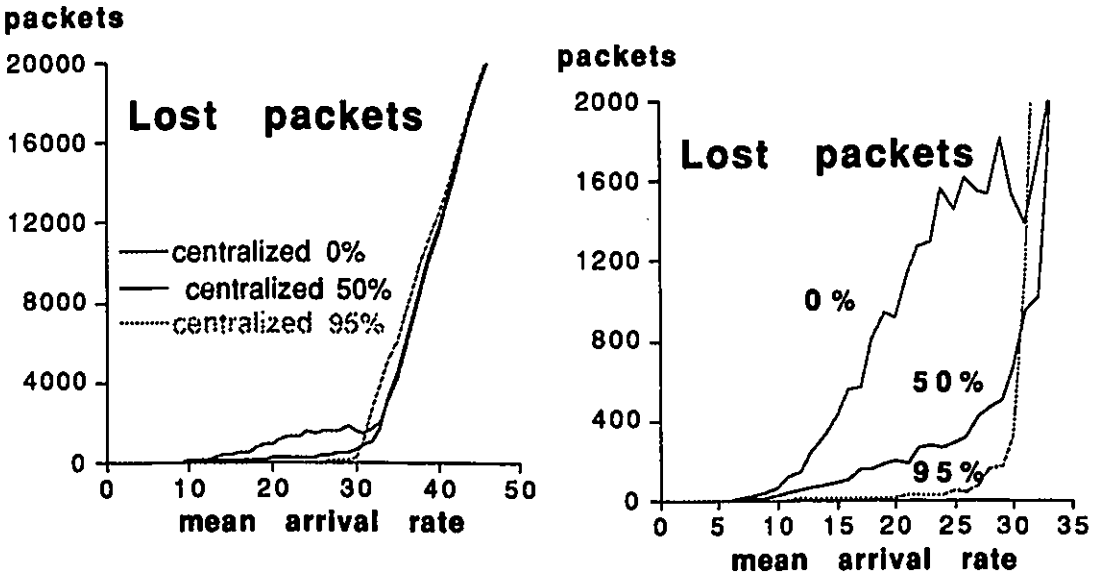


Figure 3.8: Lost packets performance under priority levels

A dramatic improvement in terms of lost packets is observed in the absence of congestion i.e. when the mean arrival rate is less than 30. When priority is high, the routing process is less dependent on the fluctuations of the arrivals. At a mean arrival rate of 25 packets, for example, about 1600 packets are lost with full priority to internal arrivals, about 400 with mixed priority and less than 100 with high priority As a reminder, note that the lost packets are always external.

A high level of priority implies that the routing process will try to minimize packet loss by reserving more space for external arrivals. However, there comes a point where a high priority is no longer desirable. This takes place when the traffic becomes important, making it impossible to avoid degradation. Trying to keep some buffer space available for hypothetical external arrivals leads to worse results. This is shown in figures 3.8 and 3.9 where, in the interval [30, 45], performance gets worse with increasing priority.

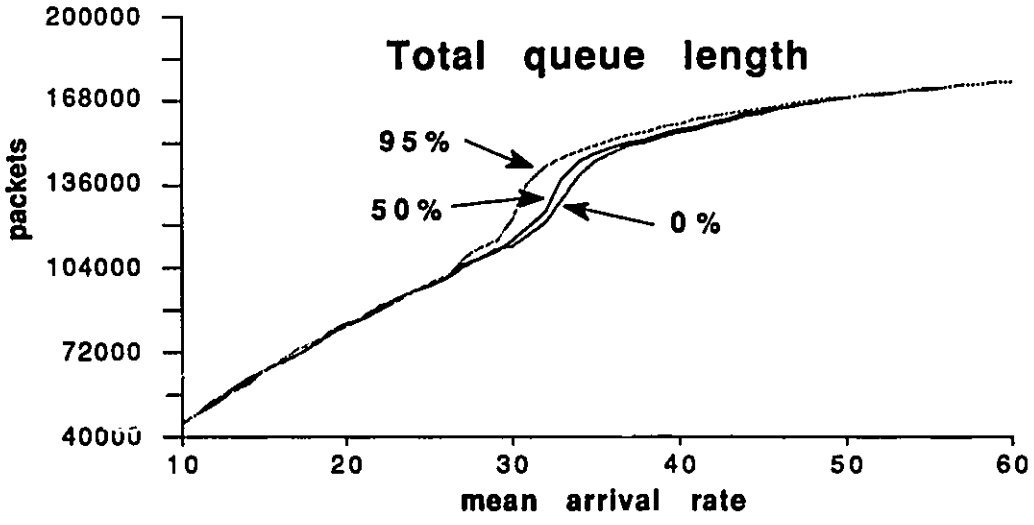


Figure 3.9: Total queue length

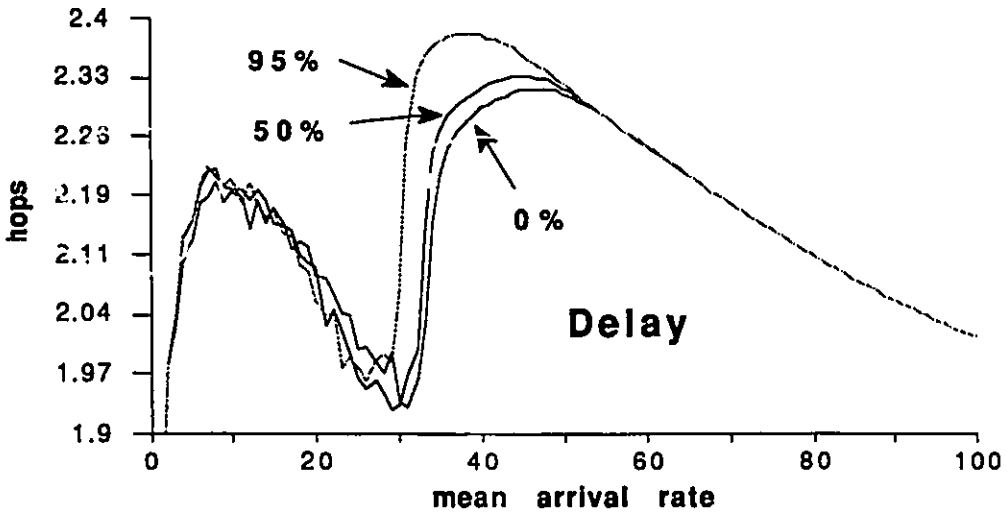


Figure 3.10: Delay comparison

In order to give priority to external packets and avoid packet loss, efficiency in terms of queue length has to be sacrificed causing traffic to be redirected. As a consequence, normally short queues become longer because other queues are keeping more space available for externals. Therefore, delay increases as shown in figure 3.10. At this point, it is up to the model designer to decide whether it is more important to minimize delay or minimize packet loss.

3.3.c Comparison of the single and multiple buffer models

As seen in figures 3.11 and 3.12, almost no packets are lost with the single buffer architecture before congestion. This result was easy to predict since no space is wasted with a unique buffer at every node.

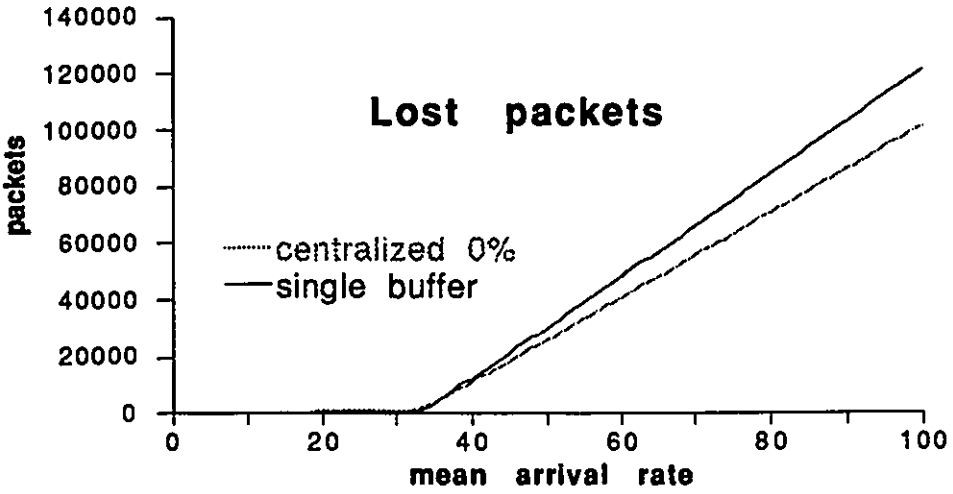


Figure 3.11: Lost packets over the entire buffer range

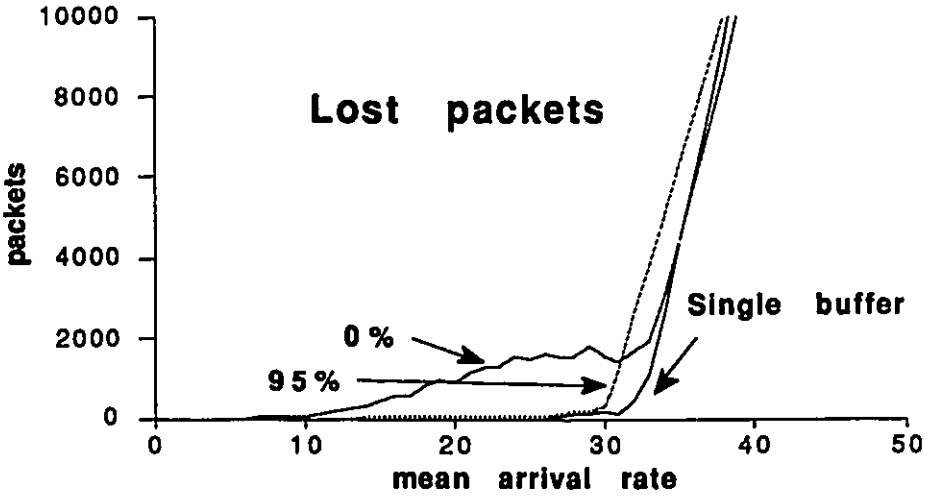


Figure 3.12: Lost packets for mean arrivals in between [0, 50]

Surprisingly, the performance of the single buffer model becomes worse than that of the multiple buffer model during congestion. This behavior is attributed to the following: when a multiple buffer configuration is present, the routing policy used in this thesis, has a tendency to favour the transmission of nearest neighbors packets, since they can be delivered to destination with no stopovers. This allows efficient throughput without increasing the queue lengths significantly. Hence the packets discarded are mainly destined to second neighbors. However, in the single buffer case, this biasedness toward nearest neighbors packets disappears. Lost packets are not mostly destined to second neighbors anymore, but are assumed to follow a multivariable hypergeometric distribution as explained in section 2.1.b.

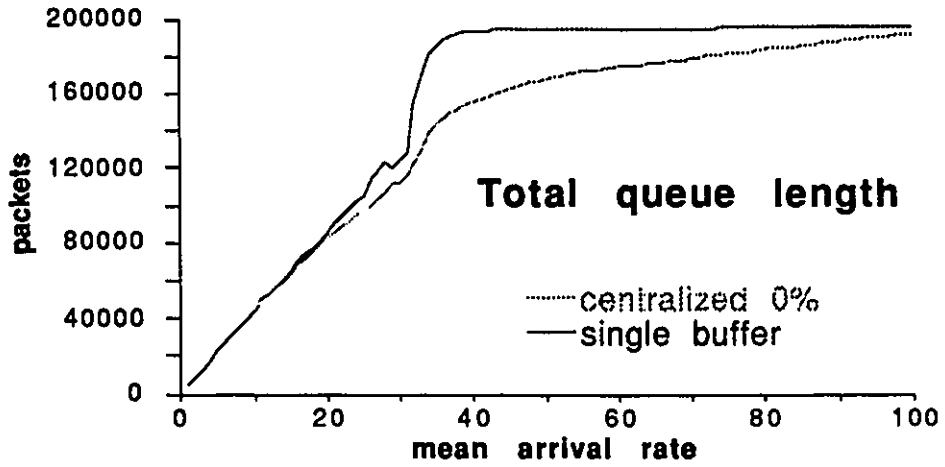


Figure 3.13: Total queue length comparison

Consequently, the expected number of hops increases during congestion as seen in the total queue length results (fig. 3.13) and the associated delays (fig. 3.15).

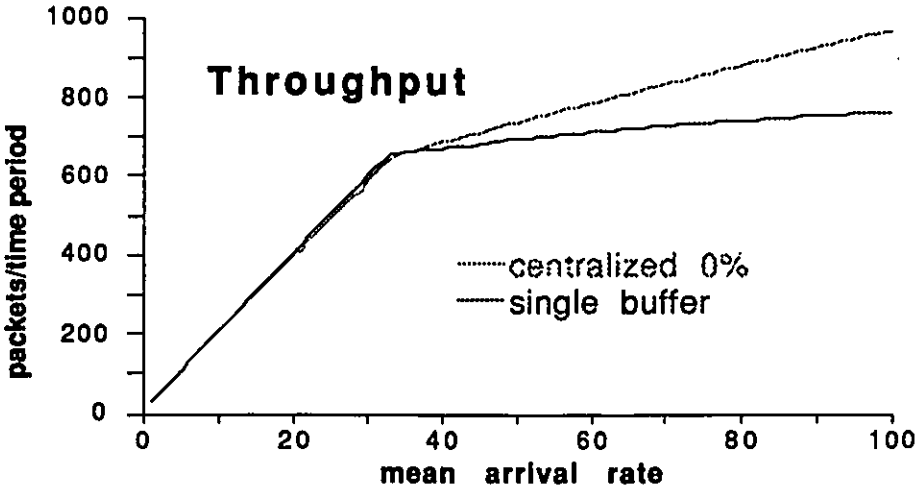


Figure 3.14: Throughput of both architectures

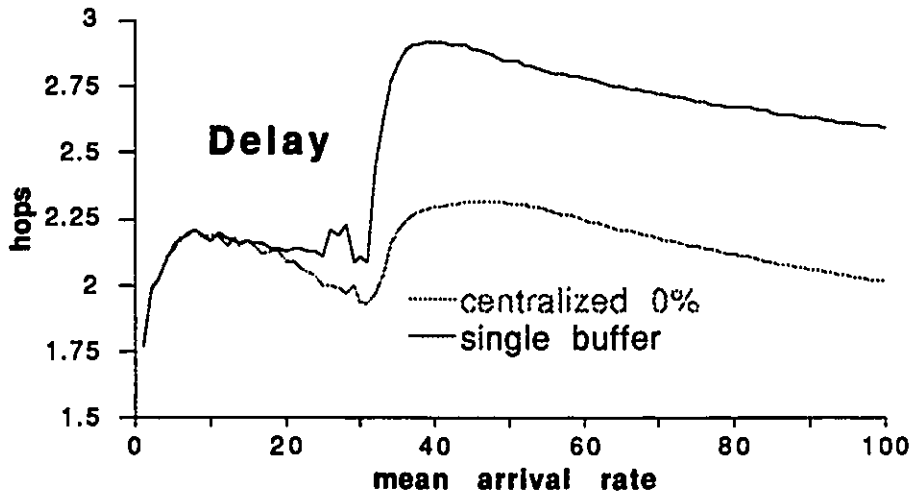


Figure 3.15: Delay performance

In summary, the unique node architecture is interesting and could lead to better results. However, some scheme for buffer management should be devised to improve the performance under congestion. This topic is not pursued any further in this work.

3.3.d Comparison of centralized and decentralized models

In this part, we compare the centralized and decentralized models with multiple buffers under the same priority level. The results shown in figures 3.16 and 3.17 are very surprising at first look. Under non congested conditions, the number of lost packets in the decentralized model is, most of the time, less than or equal to that of the centralized model. There is no major difference in terms of queue length neither, consequently, the delay (figure 3.17) is almost the same for both methods.

Why is it so? The decentralized model was built from the centralized one with the same properties. Since there are few nodes in the network, most of the global information is available at every node and the connectivity is low. With only two neighbors, few uncoordination will arise at the node buffer under low traffic.

The reestimation is efficient with two surrounding neighbors. Results without this second step are shown in appendix A. This reestimation gives a suboptimal solution which might even be advantageous in some situations. It will prevent some packet loss but may increase the queue length (trade off). At 95% priority level, slight degradation is present in the delay performance under congestion.

Finally, we believe that the number of nodes in the network is insufficient to conclusively assert the superiority of one model over the other. We do however expect uncoordination to increase, in the decentralized model, as the number of nodes in the network multiplies.

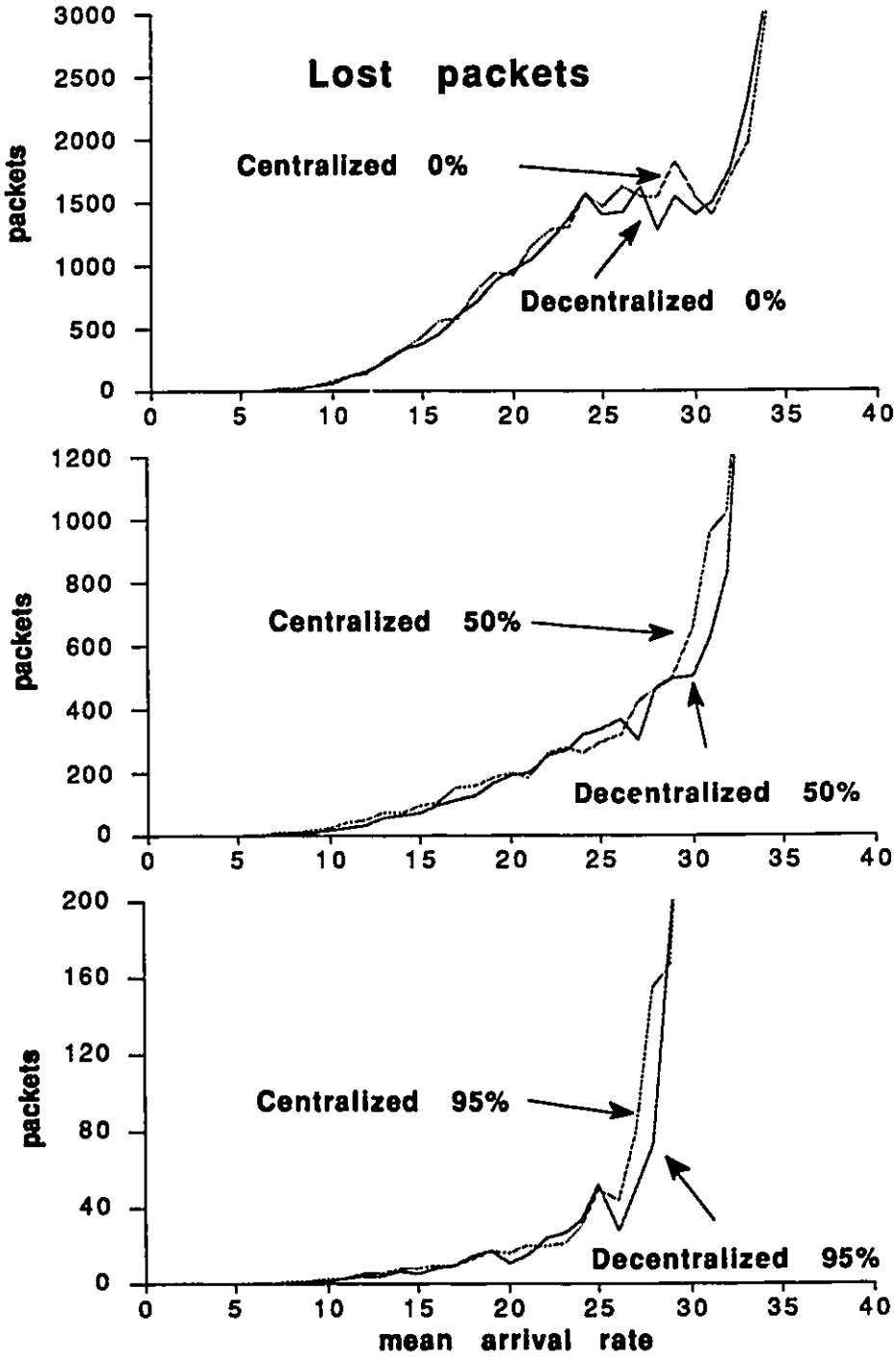


Figure 3.16: Packet loss of centralized and decentralized models

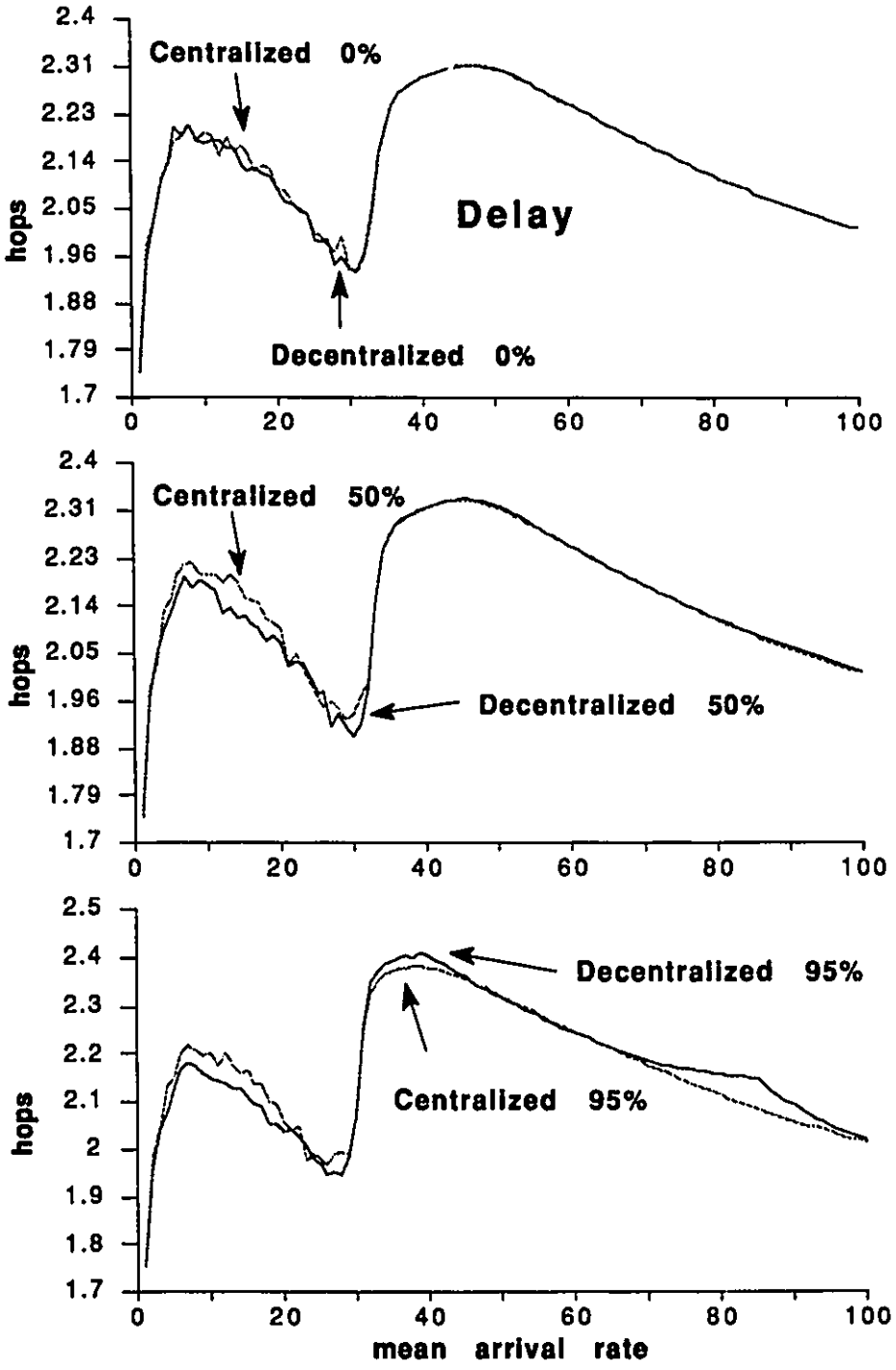


Figure 3.17: Delay of the centralized and decentralized models

3.3.e Comparison of the models under unbalanced traffic

Tables 3.1, 3.2 and 3.3 show how the models would react in a realistic scenario where the traffic is unstable. We examine three situations where the mean values of the external arrival rates are uniformly distributed over the ranges [0, 30], [0, 50] and [0, 60] respectively. This in order to test the models under increasing congestion.

In accordance with the results from previous sections, no model gets seriously congested in the first situation. The maximum mean value is lower than the threshold value estimated around 33.3 packets/time period for all models, except for [1] where it is around 25 packets/time period.

Mean arrivals selected from a discrete uniform distribution [0, 30]				
Model	**Total queues**	**Lost packets**	**Delay**	**Throughput**
Model in [1]	89532	283	2.92	309.3
Centralized 0%	62713	429	2.06	307.8
Centralized 50%	67305	90	2.18	311.2
Centralized 95%	65908	13	2.13	312.0
Single buffer	68203	0	2.21	312.1
Decentralized 0%	63467	656	2.10	305.5
Decentralized 50%	66294	87	2.15	311.2
Decentralized 95%	66677	1	2.16	312.1

Table 3.1: Unbalanced traffic with mean rate uniformly distributed [0,30]

Model [1] has some problems dealing with sudden traffic increases due to its lower threshold value. The priority scheme works well on both centralized and decentralized models. At 0%, the problem is equivalent to optimally routing the packets in queue without paying any attention to the external arrivals. This explains why the total queues are relatively shorter while the number of packets refused is higher than at other priority levels. It is interesting to note that at 95% priority level, the decentralized model accumulates 769 more packets in queue than the centralized version in order to accept only 12 supplemental packets into the network. No packets are lost with the single buffer case but the total queues are longer than all the other methods, except [1].

Mean arrivals selected from a discrete uniform distribution [0, 50]				
Model	**Total queues**	**Lost packets**	**Delay**	**Throughput**
Model in [1]	128 232	3 966	2.74	472.7
Centralized 0%	100 115	3 509	2.11	477.3
Centralized 50%	109 738	1 073	2.20	501.9
Centralized 95%	109 972	429	2.18	508.4
Single buffer	122 647	6	2.41	512.7
Decentralized 0%	99 923	3 414	2.11	478.3
Decentralized 50%	110 981	1 043	2.23	502.2
Decentralized 95%	112 437	764	2.25	505.0

Table 3.2: Unbalanced traffic with mean rate uniformly distributed [0,50]

In the second situation, the range is extended to [0, 50], so that all methods face congestion at some moment in the simulation.

The priority scheme still works well on the centralized model. At 95%, 644 more packets are preserved compared to the 50% priority level while the queues are only 234 packets longer. The results are quite the same for the decentralized model, except that a sign of degradation is noticed at 95% where the number of lost packets is higher and the queues longer (335 more packets lost and 2465 supplemental packets in queues compared to the centralized model). The single buffer model reacts very well to packet conservation whereas the delay is increased significantly.

Mean arrivals selected from a discrete uniform distribution [0, 60]				
Model	**Total queues**	**Lost packets**	**Delay**	**Throughput**
Model in [1]	137 138	10 999	2.79	495.8
Centralized 0%	116 217	7 234	2.20	533.8
Centralized 50%	129 626	3 858	2.31	567.9
Centralized 95%	131 368	3 903	2.34	567.5
Single buffer	163 038	2 999	2.86	576.5
Decentralized 0%	115 953	7402	2.20	532.1
Decentralized 50%	130 267	4 289	2.33	563.6
Decentralized 95%	134 749	4 701	2.43	559.4

Table 3.3: Unbalanced traffic with mean rate uniformly distributed [0,60]

In the third experiment, the range is extended over [0, 60] to observe the impact of more frequent congestions.

We notice degradation in packet loss for the single buffer case which brings its performance closer to that of the other models (due to reasons explained in section 3.3.c). The priority scheme shows its limitations for both the centralized and decentralized routing. Improvements are made by increasing the priority level from 0% to 50% but not to 95%. We also notice that the performance of the decentralized model is inferior to the centralized one at every priority level.

CHAPTER 4
Conclusion

In this thesis, we presented some modifications to one known model for routing in datagram packet-switching networks. It has been shown that the previous prudent approach for routing policy [1] which consists of considering the arrivals before evaluating the transmission possibilities decrease significantly the throughput performance (increases the number of packets discarded at buffers) and increases the delay (average number of hops) of every packet transmitted through the network.

We have noticed that the multiple buffer architecture of one node induces storage waste when user utilization rates vary in time. By introducing a constraint which dedicates buffer space to external arrivals and provides a mean for flow control, we can partially remedy to this situation. This diminishes the number of lost packets during the operation of the network under low to medium traffic load. On the other hand, ambitious priority levels decrease the throughput performance under high congestion.

We studied another solution to the buffer waste problem which consists of having one single buffer containing all the queues at every node. This leads to interesting improvements under non congested conditions. However buffer management should be provided under congestion to prevent excessive packet loss.

Finally, we investigated a decentralized routing policy where decision making is performed at node level. Its performance was similar to that of the

multiple buffer centralized model on a network of 5 nodes. However, we do expect this performance to deteriorate as the number of nodes and the connectivity in the network increase. This suspicion is motivated by the degradation observed under unstable congested conditions.

For further research, it might be interesting to perform discrete-event simulations where more attention is paid to the communication aspects, such as the overhead needed by the different methods, the reliability of each, etc... This could allow to relate the results to those obtained from queuing theory.

Other types of feasible decentralized or distributed routing schemes, where it is possible to obtain simulation results, should be studied for the sake of comparison. It would be interesting to find a way to allow the nodes to perform asynchronous decision making without any restricting constraints such as assuming that transmissions have to cease during the optimization process.

REFERENCES

[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] M. Aicardi, F. Davoli and R. Minciardi, "Decentralized Optimal Control of Markov Chains with a Common Past Information Set," *IEEE Trans. on Automatic Control*, vol. AC-32, pp. 1028-1031, November 1987.

[3] M. Aicardi, F. Davoli and R. Minciardi, "Decentralized Dynamic Routing as a Markov Chain Optimization Problem," in *IEEE Conf. on decision and Control*, Los Angeles, pp. 1514-1517, December 1987.

[4] J. Banks and J. S. Carson, "*Discrete-event system simulation*," Prentice-Hall, Englewood Cliffs, N. J., 1984.

[5] P. R. Bell and K. Jabbour, "Review of Point-to-Point Network Routing Algorithms," *IEEE Communications Magazine*, vol. 24, pp. 34-38, January 1986.

[6] P. R. Bell and K. Jabbour, "Evaluation of Point-to-Point Network Routing Algorithms," *IEEE Trans. on Communications*, vol. COM-35, pp. 470-473, April 1987.

[7] D. P. Bertsekas, "Distributed Dynamic Programming", *IEEE Trans. on Automatic Control*, vol. AC-27, pp. 610-616, June 1982.

[8] D. G. Cantor and M. Gerla, "Optimal Routing in a Packet-Switched Computer Network," *IEEE Trans. on Computers*, vol C-23, pp. 1062-1069, October 1974.

[9] F. Chang and L. Wu, "An Optimal Adaptive Routing Algorithm," *IEEE Trans. on Automatic Control*, vol. AC-31, pp. 690-700, August 1986.

[10] A. Ephremides and S. Verdu, "Control and Optimization Methods in Communication Network Problems," *IEEE Trans. on Automatic Control*, vol AC-34, pp. 930-942, September 1989.

[11] R. G. Gallager, "A Minimum Delay Routing Algorithm Using Distributed Computation," *IEEE Trans. on Communications*, vol. COM-25, pp. 73-85, January 1977.

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

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

[14] L. Kleinrock, "*Communication Nets: Stochastic Message Flow and Delay*,"McGraw-Hill, New York, 1964.

[15] H. Kobayashi and A. G. Konheim, "Queueing Models for Computer Communications System Analysis," *IEEE Trans. on Communications*, vol. COM-25, pp. 2-29, January 1977.

[16] P. Kubat, "Estimation of Reliability for Communication/ Computer Networks- Simulation/ Analytic Approach," *IEEE Trans. on Communications*, vol COM-37, pp. 927-933, September 1989.

[17] J. L. Kuester and J. H. Mize, "*Optimization Techniques with Fortran*," McGraw-Hill, New York, 1973.

[18] D. F. Lazaroiu and E. Staicut, "Congestion-Reliability-Availability Relationship in Packet-Switching Computer Networks," *IEEE Trans. on reliability*, vol. R-32, pp. 354-365, October 1983.

[19] J. M. McQuillan, "Design Considerations for Routing Algorithms in Computer Networks," in *Proc. Seventh Hawaii Int. Conf. Syst. Sci.*, pp. 22-24, January 1974.

[20] J. M. McQuillan and F. C. Schoute, "A Comparison of Information Policies for Minimum Delay Routing Algorithms," *IEEE Trans. on Communications*, vol. COM-26, pp. 1266-1271, January 1977.

[21] J. M. McQuillan, G. Falk and I. Richer, "A Review of the Development and Performance of the ARPANET Routing Algorithm," *IEEE Trans. on Communications*, vol. COM-26, pp. 1802-1811, December 1978.

[22] J. M. McQuillan, I. Richer and E. C. Rosen, "The New Routing Algorithm for the ARPANET," *IEEE Trans. on Communications*, vol. COM-28, pp. 711-719, May 1980.

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

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

[25] A. Pattavina, "Reliability-Constrained Dimensioning of Transmission Resources in Packet-Switched Networks," *IEEE Trans. on Reliability*, vol. R-37, pp. 434-442, October 1988.

[26] I. B. Rhodes and J. M. Abram, "Some Shortest Path Algorithms with Decentralized Information and Communication Requirements," *IEEE Trans. on Automatic Control*, vol AC-27, pp. 570-582, June 1982.

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

[28] P. E. Sarachik and U. Ozguner, "On Decentralized Dynamic Routing for Congested Traffic Networks," *IEEE Trans. on Automatic Control*, vol AC-27, pp 1233-1238, December 1982.

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

[30] M. Schwartz and T. E. Stern, "Routing Techniques Used in Computer Communication Networks," *IEEE Trans. on communications*, vol COM-28, pp 539-552, April 1980.

[31] A. Segall, " The Modeling of Adaptive Routing in Data-Communication Networks," *IEEE Trans. on Automatic Control*, vol. AC-25, pp. 85-95, January 1977.

[32] W. Stallings, "*Data and Computer Communications*," New York, Macmillan, 1985.

[33] G. I. Stassinopoulos and P. Konstantopoulos, "Optimal Congestion Control in Single Destination Networks," *IEEE Trans. on communications*, vol COM-33, pp 792-800, August 1985.

[34] G. I. Stassinopoulos, "Optimal Dynamic Routing in Multidestination Networks," *IEEE Trans. on Communications*, vol. COM-35, pp. 472-475, April 1987.

[35] G. I. Stassinopoulos and H. Kukutos, "Optimal Dynamic Routing in Double Ring Networks", *IEEE Trans. on Communications*, vol. COM-37, pp. 890-896, August 1989.

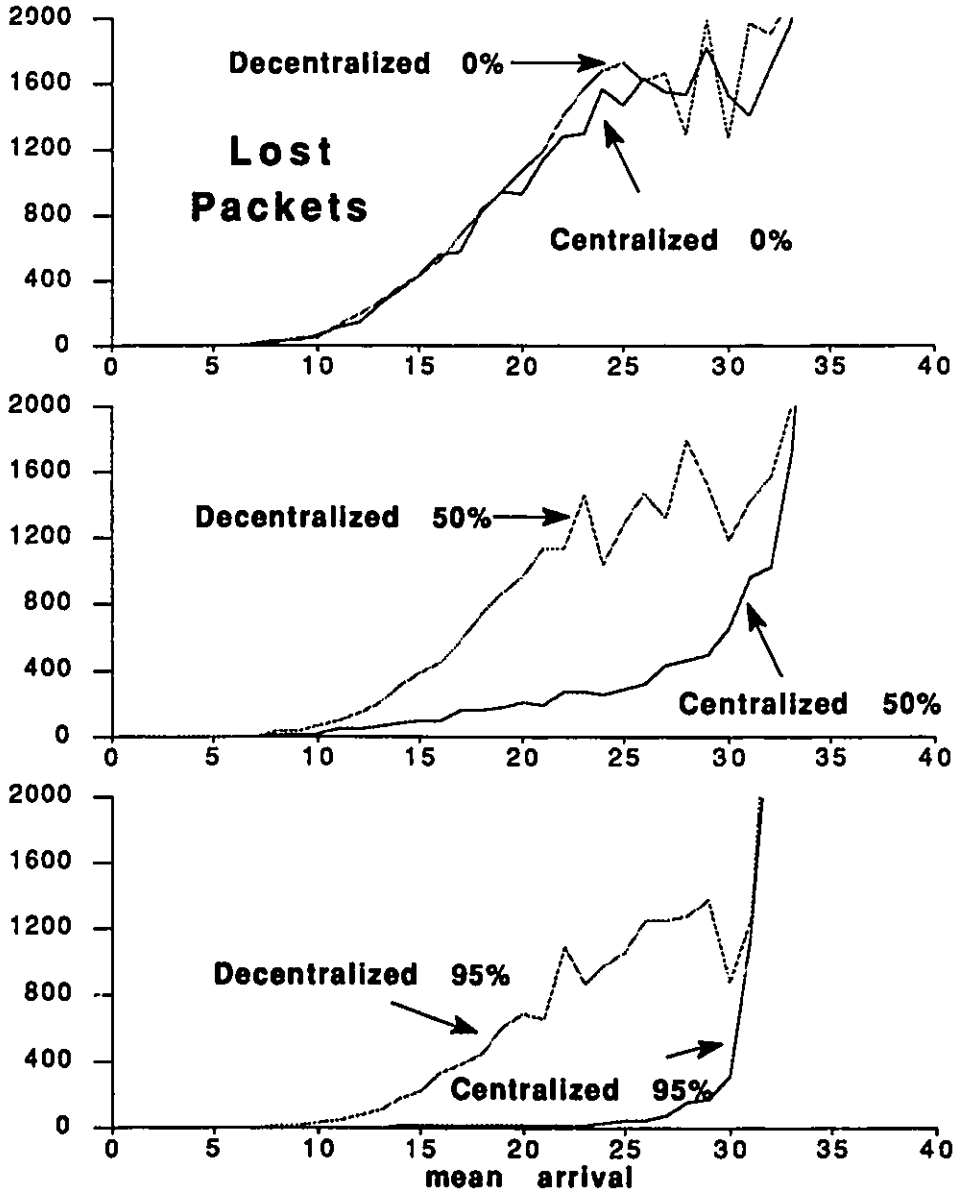
[36] A. Tanenbaum, "*Computer Networks*," Second Edition, Englewood Cliffs, N. J., Prentice-Hall, 1988.

[37] J. S. Turner, "Design of a Broadcast Packet Switching Network," *IEEE Trans. on Communications*, vol. COM-36, pp. 734-743, June 1988.

APPENDIX

A. More simulation results

The following results show the degradation in terms of number of lost packets in the decentralized scheme when the reestimation mechanism (the second step of the optimization) is omitted.



B. Computer program listings

Multiple buffer simulation

. Main program

```
C*****************************************************************C
C                                                                 C
C          THIS PROGRAMM IS SIMULATING A COMPUTER                 C
C          NETWORK OF 5 NODES WITH DYNAMIC ROUTING.               C
C                                                                 C
C  THE VARIABLES IN THIS PROGRAM ARE:                             C
C                                                                 C
C  CP(I,J)     =  CAPACITY OF THE LINK (I,J)                      C
C  RO(I,J)     =  TRANSMISSION ERROR RATE IF LINE (I,J)           C
C  BF(I,J)     =  CAPACITY OF THE BUFFER (I,J)                    C
C  Q(I,J)      =  NO.OF PACKETS IN QUEUE (I,J) (PAST STATE)       C
C  QT(I,J,S)   =  TOTAL PACKETS IN BUFFER (I,J) BEFORE DISCARTING C
C  QQ(I,J,S)   =    ''    ''    ''  ''      ''  AFTER  DISCARTING C
C  QLT(I,J,S)  =  LOST    ''    ''  ''      ''  AT TIME S         C
C  QL(I,J)     =  LOST    ''    ''  ''      ''  DURING ITERATION  C
C  ISEED       =  SEED FOR RANDOM NUMBER GENERATION               C
C  ER(I,J,S)   =  MEAN EXTERNAL ARRIVAL AT BUFFER(I,J) DURING (S,S+1)  C
C  AR(I,J,S)   =  EXTERNAL ARRIVALS AT BUFFER (I,J) DURING (S,S+1)    C
C  ALPHA(I,J)  =  WEIGHTING FACTOR OF THE OPTIMIZATION            C
C  A(I,J)      =  ELEMENTS OF THE CONSTRAINT MATRIX       | MIN CX  |  C
C  B(I)        =  CONSTRAINT VECTOR                       |  S.T.   |  C
C  C(I)        =  COEFFICIENTS OF THE OBJECTIVE FUNCTION  | AX = B  |  C
C  ATAB        =  INITIAL SIMPLEX TABLEAU                         C
C  UPBND       =  UPPER BOUND OF THE ROUTING VARIABLES            C
C  T           =  OPTIMAL VALUE OF THE ROUTING VARIABLES          C
C  TL          =  NUMBER OF PACKET LOST DURING TRANSMISSION       C
C  QTOT        =  TOTAL OF THE QUEUES DURING THE SIMULATION       C
C  PLOST       =  TOTAL PACKET LOST    ''     ''   ''            C
C  TARR        =  TOTAL ARRIVALS AT THE NODES                     C
C  AVD         =  AVERAGE DELAY                                   C
C  THR         =  THROUGHPUT                                      C
C  BOUND       =  PROBABILITY THAT THE QUEUE IS ACCEPTED IN THE BUFFER C
C*****************************************************************C
      DOUBLE PRECISION BOUND
      REAL CP(5,5),RO(5,5),RRO(40),BF(5,5),QL(5,5),PLOST,TARR,PUT,
     $  Q(5,5),QT(5,5,100),QLT(5,5,100),QQ(5,5,100),T(41),QTOT,SE(5,5),
     $  ER(5,5,100),AR(5,5,100),ALPHA(5,5),WT(10),WL(10),WA(10),WH(10)
      INTEGER S,ISEED,K,NM1,I,II,CONTRL,METH,NITER,L,NARVL
      COMMON /IO/ K,NM1
C-----------------------------------------------------------------
C  NARVL PERMIT TO CHOOSE THE TYPE OF EXTERNAL ARRIVALS AT THE BUFFERS
C   1  =  UNIFORMLY SPREAD THROUGH THE NETWORK
C   2  =  VARYING FOR REMOTE DESTINATIONS, CONSTANT FOR NEIGHBOURS
C-----------------------------------------------------------------
      NARVL  =  1
C-----------------------------------------------------------------
C  CONTRL AND METH SELECT THE TYPE OF ROUTING DESIRED
C    IF CONTRL = 1 : THEN CENTRALIZED ROUTING
C        - IF METH = 1 : AHMED'S MODEL ,  IF METH = 2 : PROPOSED MODEL
C    IF CONTRL = 2 : THEN DECENTRALIZED ROUTING
C-----------------------------------------------------------------
```

```
      CONTRL =   1
      METH   =   2
C-----------------------------------------------------------------------
C  NITER IS THE NUMBER OF REPETITION OF THE SIMULATION
C-----------------------------------------------------------------------
      NITER  =  10
C
      PUT    =   1.0
      BOUND  =  0.95
C
C     DO 100 I = 1,100
         DO 50 L = 1,NITER
            CALL STSEED(L,ISEED)
            K      =  99
            PLOST  = 0.0
            QTOT   = 0.0
            TARR   = 0.0
C
            CALL TOPO(CP,RO,RRO,BF)
            CALL INITL(Q,QT,QLT,QQ)
            CALL ARRVL(NARVL,PUT,ISEED,ER,AR)
            S = 1
 20         IF(S.LT.K) THEN
               IF(CONTRL.EQ.2) THEN
                    CALL NEWFAC(S,BF,BOUND,Q,ER,SE,ALPHA)
                    CALL OPTDCN(S,BF,Q,SE,ALPHA,RO,CP,T)
                    CALL ADJUST(S,BF,Q,SE,RRO,T)
               ELSE
                    IF(METH.EQ.1) THEN
                       CALL FACTOR(S,BF,Q,ER,ALPHA)
                       CALL CNTAHM(S,BF,Q,ER,ALPHA,RO,CP,T)
                    ELSE
                       CALL NEWFAC(S,BF,BOUND,Q,ER,SE,ALPHA)
                       CALL CNTNEW(S,BF,Q,SE,ALPHA,RO,CP,T)
                    ENDIF
               ENDIF
               DO 10 II = 1,40
 10            CALL ERROR(ISEED,II,RRO,T)
               S = S + 1
               CALL CALCUL(S,BF,AR,T,Q,TARR,QL,QQ,PLOST,QTOT,QLT,QT)
               GO TO 20
            ENDIF
            CALL TEST(QQ,QT,QLT,AR,ER)
            CALL COMPT(L,PLOST,TARR,QT,QTOT,PUT,WT,WL,WA,WH)
 50      CONTINUE
C
         CALL STATC(PUT,TARR,NITER,WT,WL,WA,WH)
         PUT = PUT + 1.0
100   CONTINUE
      STOP
      END
```

.Subroutines

```
C*********************************************************************
      SUBROUTINE STSEED(L,ISEED)
C*********************************************************************
      IMPLICIT NONE
      INTEGER L,NSEED(10),ISEED
C
      NSEED( 1) = 2116429302
      NSEED( 2) =  683743814
      NSEED( 3) =  964393174
      NSEED( 4) = 1217426631
      NSEED( 5) =  618433579
      NSEED( 6) = 1157240309
      NSEED( 7) =   15726055
      NSEED( 8) =   48108509
      NSEED( 9) = 1797920909
      NSEED(10) =  477424540
C
      ISEED     =  NSEED(L)
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE TOPO(CP,RO,RRO,BF)
C*********************************************************************C
      IMPLICIT NONE
      REAL CP(5,5),RO(5,5),RRO(40),BF(5,5)
      INTEGER I,J
C----------  CAPACITY AND ERROR RATE OF CHANNELS  ------------
      DO 10 I = 1,5
      DO 10 J = 1,5
         CP(I,J) = 0.0
  10     RO(I,J) = 0.0
      CP(1,2) = 100.0
      CP(2,1) = 100.0
      CP(2,3) = 100.0
      CP(3,2) = 100.0
      CP(3,4) = 100.0
      CP(4,3) = 100.0
      CP(4,5) = 100.0
      CP(5,4) = 100.0
      CP(5,1) = 100.0
      CP(1,5) = 100.0
C
      RO(1,2) = 1E-2
      RO(2,1) = 5E-3
      RO(2,3) = 2E-3
      RO(3,2) = 9E-3
      RO(3,4) = 1E-2
      RO(4,3) = 7E-3
      RO(4,5) = 2E-3
      RO(5,4) = 9E-3
      RO(5,1) = 4E-3
      RO(1,5) = 1E-2
C-------------  ERROR VECTOR  ------------
      DO 20 J = 1,4
```

```
      I = 10 * (J-1)
      RRO(I + 1 ) = RO(1,2)
      RRO(I + 2 ) = RO(2,1)
      RRO(I + 3 ) = RO(2,3)
      RRO(I + 4 ) = RO(3,2)
      RRO(I + 5 ) = RO(3,4)
      RRO(I + 6 ) = RO(4,3)
      RRO(I + 7 ) = RO(4,5)
      RRO(I + 8 ) = RO(5,4)
      RRO(I + 9 ) = RO(5,1)
      RRO(I + 10) = RO(1,5)
  20  CONTINUE
C-------------   BUFFER   ----------------
      DO 30 I = 1,5
      DO 30 J = 1,5
      IF(I.NE.J) THEN
         BF(I,J) = 100.0
      ELSE
         BF(I,J) =   0.0
      ENDIF
 30   CONTINUE
      RETURN
      END
C*********************************************************************
      SUBROUTINE INITL(Q,QT,QLT,QQ)
C*********************************************************************
      IMPLICIT NONE
      REAL Q(5,5),QT(5,5,100),QLT(5,5,100),QQ(5,5,100)
      INTEGER I,J
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            Q(I,J) =  50.0
         ELSE
            Q(I,J) =  0.0
         ENDIF
         QT(I,J,1)  = Q(I,J)
         QQ(I,J,1)  = Q(I,J)
         QLT(I,J,1) = 0.0
 10   CONTINUE
      RETURN
      END
C*********************************************************************C
      SUBROUTINE ARRVL(NARVL,PUT,ISEED,ER,AR)
C*********************************************************************C
      IMPLICIT NONE
      REAL ER(5,5,100),AR(5,5,100),THETA,PUT,UNB
      INTEGER K,I,J,NM1,IA(1),IR(1),NR,ISEED,IS,NARVL
      COMMON /IO/ K,NM1
      NR=1
C----------  READ THE EXTERNAL MEAN ARRIVALS ----------
      IF(NARVL.EQ.1) THEN
           DO 12 IS = 1,K-1
           DO 12 I = 1,5
           DO 12 J = 1,5
              IF(I.NE.J) THEN
                 ER(I,J,IS) = PUT
```

```
            ENDIF
 12       CONTINUE
      ELSE
         DO 13 IS = 1, K-1
         DO 13 I  = 1,5
         DO 13 J  = 1,5
            IF(I.NE.J) THEN
               CALL RNSET(ISEED)
               CALL RNUND(NR,50,IA)
               CALL RNGET(ISEED)
               ER(I,J,IS) = IA(1)
            ENDIF
 13      CONTINUE
      ENDIF
C---------- GENERATE THE EXTERNAL ARRIVALS --------------
      DO 15 I = 1,5
      DO 15 J = 1,5
      IF(I.NE.J) THEN
         DO 20 IS = 1,K-1
            THETA = ER(I,J,IS)
            IF(THETA.GT.0.0) THEN
               CALL RNSET(ISEED)
               CALL RNPOI(NR,THETA,IR)
               CALL RNGET(ISEED)
               AR(I,J,IS) = IR(1)
            ELSE
               AR(I,J,IS) = 0.0
            ENDIF
 20      CONTINUE
      ENDIF
15    CONTINUE
      RETURN
      END
C*********************************************************************C
      SUBROUTINE FACTOR(S,BF,Q,ER,ALPHA)
C*********************************************************************C
      IMPLICIT NONE
      REAL BF(5,5),ER(5,5,100),Q(5,5),ALPHA(5,5),SPACE
      INTEGER S,I,J
C---------------  SET WEIGTHING FACTOR -------------
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            SPACE = BF(I,J) - Q(I,J) - ER(I,J,S)
            IF(SPACE.LT. 0.0) THEN
               ALPHA(I,J)  = (- SPACE / BF(I,J))
            ELSE
               ALPHA(I,J)  = 0.0
            ENDIF
         ENDIF
 10   CONTINUE
      RETURN
      END
```

```
C*********************************************************************C
      SUBROUTINE NEWFAC(S,BF,BOUND,Q,ER,SE,ALPHA)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION BOUND
      REAL BF(5,5),ER(5,5,100),Q(5,5),ALPHA(5,5),SE(5,5)
      INTEGER S,I,J
C---------------  SET WEIGTHING FACTOR -------------
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            CALL PRO(S,BOUND,I,J,ER,SE)
            ALPHA(I,J) = 1 + (( Q(I,J) + SE(I,J) ) / BF(I,J))
         ENDIF
  10  CONTINUE
      RETURN
      END
C*********************************************************************C
      SUBROUTINE PRO(S,BOUND,I,J,ER,SE)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION W,Y,PROB,BOUND,SUM
      REAL SE(5,5),ER(5,5,100)
      INTEGER S,I,J,IX,IFAC
      SE(I,J) = 0.0
      Y    = ER(I,J,S)
      W    = DEXP(-Y)
      IX   = 0
      IFAC = 1
      SUM  = W
      PROB = W
 10   IF(PROB.LT.BOUND) THEN
         IX = IX + 1
         IFAC = IFAC * IX
         SUM =  (SUM * Y) / IX
         PROB = PROB + SUM
         GO TO 10
      ELSE
         SE(I,J) = FLOAT(IX)
         RETURN
      ENDIF
      RETURN
      END
C*********************************************************************C
      SUBROUTINE CNTAHM(S,BF,Q,ER,ALPHA,RO,CP,T)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION ATAB(51,41), UPBND(41),U(41)
      REAL A(50,40),B(50),C(40),ROOM,ER(5,5,100),RO(5,5),
     $ CP(5,5),BF(5,5),Q(5,5),ALPHA(5,5),BR(5,5),T(41)
      INTEGER S,M,N,NM1,K,ISIZE,NZR1VR,I,J,M1,N1,II,IROW(51)
      COMMON /IO/ K,NM1
C
      ISIZE  = 90
      M      = 51
      N      = 41
      NZR1VR = 40
```

```
      NM1    = N - 1
C------------  UPPER BOUND ON THE VARIABLES --------
      DO 10 I = 1,NM1
  10  UPBND(I) = 100.0
C------- INEQUALITY OF CONSTRAINTS: -1=LE, 0=EQ AND +1=GE ---
      IROW(1) = 0
      DO 20 I = 2,M
  20  IROW(I) = -1
C--------------  CONSTRAINT MATRIX -----------
      DO 30 I = 1, 50
      DO 30 J = 1, 40
  30  A(I,J) = 0.0
      DO 40 I = 1,10
      A(I,I)    = 1.0
      A(I,I+10) = 1.0
      A(I,I+20) = 1.0
      A(I,I+30) = 1.0
      A(I+30,I) = 1.0
  40  CONTINUE
      A(11,19) = 1.D0
      A(12,34) = 1.D0
      A(13,11) = 1.D0
      A(14,36) = 1.D0
      A(15,13) = 1.D0
      A(16,38) = 1.D0
      A(17,15) = 1.D0
      A(18,40) = 1.D0
      A(19,17) = 1.D0
      A(20,32) = 1.D0
      A(21,12) = 1.D0
      A(22,14) = 1.D0
      A(23,16) = 1.D0
      A(24,18) = 1.D0
      A(25,20) = 1.D0
      A(26,22) = 1.D0
      A(27,24) = 1.D0
      A(28,26) = 1.D0
      A(29,28) = 1.D0
      A(30,30) = 1.D0
      A(21,29) = 1.D0
      A(22,21) = 1.D0
      A(23,23) = 1.D0
      A(24,25) = 1.D0
      A(25,27) = 1.D0
      A(26,39) = 1.D0
      A(27,31) = 1.D0
      A(28,33) = 1.D0
      A(29,35) = 1.D0
      A(30,37) = 1.D0
C
      A(31,20) = 1.D0
      A(32,33) = 1.D0
      A(33,12) = 1.D0
      A(34,35) = 1.D0
      A(35,14) = 1.D0
      A(36,37) = 1.D0
      A(37,16) = 1.D0
```

```
      A(38,39) = 1.D0
      A(39,18) = 1.D0
      A(40,31) = 1.D0
      A(41,11) = 1.D0
      A(41,30) = 1.D0
      A(42,13) = 1.D0
      A(42,22) = 1.D0
      A(43,15) = 1.D0
      A(43,24) = 1.D0
      A(44,17) = 1.D0
      A(44,26) = 1.D0
      A(45,19) = 1.D0
      A(45,28) = 1.D0
      A(46,21) = 1.D0
      A(46,40) = 1.D0
      A(47,23) = 1.D0
      A(47,32) = 1.D0
      A(48,25) = 1.D0
      A(48,34) = 1.D0
      A(49,27) = 1.D0
      A(49,36) = 1.D0
      A(50,29) = 1.D0
      A(50,38) = 1.D0
C------------  BUFFER RESTRICTIONS ----------------
      DO 45 I = 1,5
      DO 45 J = 1,5
         IF(I.NE.J) THEN
            ROOM = BF(I,J) - Q(I,J)
            IF(ER(I,J,S).LE.ROOM) THEN
               BR(I,J) = BF(I,J) - Q(I,J) - ER(I,J,S)
            ELSE
               BR(I,J) = 0.0
            ENDIF
         ENDIF
  45  CONTINUE
C---------------- CONSTRAINT B VECTOR ----------------
      B( 1)  = CP(1,2)
      B( 2)  = CP(2,1)
      B( 3)  = CP(2,3)
      B( 4)  = CP(3,2)
      B( 5)  = CP(3,4)
      B( 6)  = CP(4,3)
      B( 7)  = CP(4,5)
      B( 8)  = CP(5,4)
      B( 9)  = CP(5,1)
      B(10)  = CP(1,5)
      B(11) = BR(1,2)
      B(12) = BR(2,1)
      B(13) = BR(2,3)
      B(14) = BR(3,2)
      B(15) = BR(3,4)
      B(16) = BR(4,3)
      B(17) = BR(4,5)
      B(18) = BR(5,4)
      B(19) = BR(5,1)
      B(20) = BR(1,5)
      B(21) = BR(1,3)
```

```
      B(22) = BR(2,4)
      B(23) = BR(3,5)
      B(24) = BR(4,1)
      B(25) = BR(5,2)
      B(26) = BR(1,4)
      B(27) = BR(2,5)
      B(28) = BR(3,1)
      B(29) = BR(4,2)
      B(30) = BR(5,3)
      B(31) = Q(1,2)
      B(32) = Q(2,1)
      B(33) = Q(2,3)
      B(34) = Q(3,2)
      B(35) = Q(3,4)
      B(36) = Q(4,3)
      B(37) = Q(4,5)
      B(38) = Q(5,4)
      B(39) = Q(5,1)
      B(40) = Q(1,5)
      B(41) = Q(1,3)
      B(42) = Q(2,4)
      B(43) = Q(3,5)
      B(44) = Q(4,1)
      B(45) = Q(5,2)
      B(46) = Q(1,4)
      B(47) = Q(2,5)
      B(48) = Q(3,1)
      B(49) = Q(4,2)
      B(50) = Q(5,3)
C---------------- OBJECTIVE FUNCTION ------------
      C( 1)  = (-ALPHA(1,2) - 1.0)*(1.0-RO(1,2))
      C( 2)  = (-ALPHA(2,1) - 1.0)*(1.0-RO(2,1))
      C( 3)  = (-ALPHA(2,3) - 1.0)*(1.0-RO(2,3))
      C( 4)  = (-ALPHA(3,2) - 1.0)*(1.0-RO(3,2))
      C( 5)  = (-ALPHA(3,4) - 1.0)*(1.0-RO(3,4))
      C( 6)  = (-ALPHA(4,3) - 1.0)*(1.0-RO(4,3))
      C( 7)  = (-ALPHA(4,5) - 1.0)*(1.0-RO(4,5))
      C( 8)  = (-ALPHA(5,4) - 1.0)*(1.0-RO(5,4))
      C( 9)  = (-ALPHA(5,1) - 1.0)*(1.0-RO(5,1))
      C(10)  = (-ALPHA(1,5) - 1.0)*(1.0-RO(1,5))
      C(11) = (+ALPHA(2,3) - ALPHA(1,3))*(1.0-RO(1,2))
      C(12) = (+ALPHA(1,3) - ALPHA(2,3))*(1.0-RO(2,1))
      C(13) = (+ALPHA(3,4) - ALPHA(2,4))*(1.0-RO(2,3))
      C(14) = (+ALPHA(2,4) - ALPHA(3,4))*(1.0-RO(3,2))
      C(15) = (+ALPHA(4,5) - ALPHA(3,5))*(1.0-RO(3,4))
      C(16) = (+ALPHA(3,5) - ALPHA(4,5))*(1.0-RO(4,3))
      C(17) = (+ALPHA(5,1) - ALPHA(4,1))*(1.0-RO(4,5))
      C(18) = (+ALPHA(4,1) - ALPHA(5,1))*(1.0-RO(5,4))
      C(19) = (+ALPHA(1,2) - ALPHA(5,2))*(1.0-RO(5,1))
      C(20) = (+ALPHA(5,2) - ALPHA(1,2))*(1.0-RO(1,5))
      C(21) = (+ALPHA(2,4) - ALPHA(1,4))*(1.0-RO(1,2))
      C(22) = (+ALPHA(1,4) - ALPHA(2,4))*(1.0-RO(2,1))
      C(23) = (+ALPHA(3,5) - ALPHA(2,5))*(1.0-RO(2,3))
      C(24) = (+ALPHA(2,5) - ALPHA(3,5))*(1.0-RO(3,2))
      C(25) = (+ALPHA(4,1) - ALPHA(3,1))*(1.0-RO(3,4))
      C(26) = (+ALPHA(3,1) - ALPHA(4,1))*(1.0-RO(4,3))
      C(27) = (+ALPHA(5,2) - ALPHA(4,2))*(1.0-RO(4,5))
```

```
      C(28) = (+ALPHA(4,2) - ALPHA(5,2))*(1.0-RO(5,4))
      C(29) = (+ALPHA(1,3) - ALPHA(5,3))*(1.0-RO(5,1))
      C(30) = (+ALPHA(5,3) - ALPHA(1,3))*(1.0-RO(1,5))
      C(31) = (+ALPHA(2,5) - ALPHA(1,5))*(1.0-RO(1,2))
      C(32) = (+ALPHA(1,5) - ALPHA(2,5))*(1.0-RO(2,1))
      C(33) = (+ALPHA(3,1) - ALPHA(2,1))*(1.0-RO(2,3))
      C(34) = (+ALPHA(2,1) - ALPHA(3,1))*(1.0-RO(3,2))
      C(35) = (+ALPHA(4,2) - ALPHA(3,2))*(1.0-RO(3,4))
      C(36) = (+ALPHA(3,2) - ALPHA(4,2))*(1.0-RO(4,3))
      C(37) = (+ALPHA(5,3) - ALPHA(4,3))*(1.0-RO(4,5))
      C(38) = (+ALPHA(4,3) - ALPHA(5,3))*(1.0-RO(5,4))
      C(39) = (+ALPHA(1,4) - ALPHA(5,4))*(1.0-RO(5,1))
      C(40) = (+ALPHA(5,4) - ALPHA(1,4))*(1.0-RO(1,5))
C-----------------------------------------------------------------
C SUBMIT THE INITIAL LP TABLEAU ATAB : MIN CX SUBJECT TO AX <= B
C-----------------------------------------------------------------
       DO 50 I = 1,51
       DO 50 J = 1,41
  50   ATAB(I,J) = 0.0
       DO 52 I = 2,M
  52   ATAB(I,1) = B(I-1)
       DO 54 J = 2,N
  54   ATAB(1,J) = C(J-1)
       M1 = M-1
       N1 = N-1
       DO 56 I = 1,M1
       DO 56 J = 1,N1
  56   ATAB(I+1,J+1) = A(I,J)
C
       CALL MATH(M,N,NZR1VR,ATAB,UPBND,ISIZE,IROW,U)
C
       DO 70 I = 1,NM1
  70   T(I) = U(I)
C
       RETURN
       END
C*********************************************************************C
      SUBROUTINE CNTNEW(S,BF,Q,SE,ALPHA,RO,CP,T)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION ATAB(51,41), UPBND(41),U(41)
      REAL A(50,40),B(50),C(40),ROOM,SE(5,5),RO(5,5),
     $ CP(5,5),BF(5,5),Q(5,5),ALPHA(5,5),BR(5,5),T(41)
      INTEGER S,M,N,NM1,K,ISIZE,NZR1VR,I,J,M1,N1,II,IROW(51)
      COMMON /IO/ K,NM1
C
      ISIZE  = 90
      M      = 51
      N      = 41
      NZR1VR = 40
      NM1    = N - 1
C------------- UPPER BOUND ON THE VARIABLES ----------
      DO 10 I = 1,NM1
  10  UPBND(I) = 100.0
C------- INEQUALITY OF CONSTRAINTS: -1=LE, 0=EQ AND +1=GE ----
      IROW(1) = 0
      DO 20 I = 2,M
```

```
  20  IROW(I) = -1
C-------------  CONSTRAINT MATRIX -----------
      DO 30 I = 1, 50
      DO 30 J = 1, 40
  30  A(I,J) = 0.0
      DO 40 I = 1,10
      A(I,I) = 1.0
      A(I,I+10) =  1.0
      A(I,I+20) =  1.0
      A(I,I+30) =  1.0
      A(I+10,I) =  1.0
      A(I+30,I) = -1.0
  40  CONTINUE
      A(11,20) = 1.D0
      A(12,33) = 1.D0
      A(13,12) = 1.D0
      A(14,35) = 1.D0
      A(15,14) = 1.D0
      A(16,37) = 1.D0
      A(17,16) = 1.D0
      A(18,39) = 1.D0
      A(19,18) = 1.D0
      A(20,31) = 1.D0
      A(21,11) = 1.D0
      A(21,30) = 1.D0
      A(22,13) = 1.D0
      A(22,22) = 1.D0
      A(23,15) = 1.D0
      A(23,24) = 1.D0
      A(24,17) = 1.D0
      A(24,26) = 1.D0
      A(25,19) = 1.D0
      A(25,28) = 1.D0
      A(26,21) = 1.D0
      A(26,40) = 1.D0
      A(27,23) = 1.D0
      A(27,32) = 1.D0
      A(28,25) = 1.D0
      A(28,34) = 1.D0
      A(29,27) = 1.D0
      A(29,36) = 1.D0
      A(30,29) = 1.D0
      A(30,38) = 1.D0
C
      A(31,20) =-1.D0
      A(32,33) =-1.D0
      A(33,12) =-1.D0
      A(34,35) =-1.D0
      A(35,14) =-1.D0
      A(36,37) =-1.D0
      A(37,16) =-1.D0
      A(38,39) =-1.D0
      A(39,18) =-1.D0
      A(40,31) =-1.D0
      A(31,19) = 1.D0
      A(32,34) = 1.D0
      A(33,11) = 1.D0
```

```
      A(34,36) = 1.D0
      A(35,13) = 1.D0
      A(36,38) = 1.D0
      A(37,15) = 1.D0
      A(38,40) = 1.D0
      A(39,17) = 1.D0
      A(40,32) = 1.D0
      A(41,11) =-1.D0
      A(42,13) =-1.D0
      A(43,15) =-1.D0
      A(44,17) =-1.D0
      A(45,19) =-1.D0
      A(46,21) =-1.D0
      A(47,23) =-1.D0
      A(48,25) =-1.D0
      A(49,27) =-1.D0
      A(50,29) =-1.D0
      A(41,30) =-1.D0
      A(42,22) =-1.D0
      A(43,24) =-1.D0
      A(44,26) =-1.D0
      A(45,28) =-1.D0
      A(46,40) =-1.D0
      A(47,32) =-1.D0
      A(48,34) =-1.D0
      A(49,36) =-1.D0
      A(50,38) =-1.D0
      A(41,12) = 1.D0
      A(42,14) = 1.D0
      A(43,16) = 1.D0
      A(44,18) = 1.D0
      A(45,20) = 1.D0
      A(46,22) = 1.D0
      A(47,24) = 1.D0
      A(48,26) = 1.D0
      A(49,28) = 1.D0
      A(50,30) = 1.D0
      A(41,29) = 1.D0
      A(42,21) = 1.D0
      A(43,23) = 1.D0
      A(44,25) = 1.D0
      A(45,27) = 1.D0
      A(46,39) = 1.D0
      A(47,31) = 1.D0
      A(48,33) = 1.D0
      A(49,35) = 1.D0
      A(50,37) = 1.D0
C-------------  BUFFER RESTRICTIONS  -----------------
      DO 45 I = 1,5
      DO 45 J = 1,5
         IF(I.NE.J) THEN
            BR(I,J) = BF(I,J) - Q(I,J) - SE(I,J)
            IF(BR(I,J).LT.0.0) THEN
               BR(I,J) = 0.0
            ENDIF
         ENDIF
   45 CONTINUE
```

```
C---------------  CONSTRAINT B VECTOR ----------------
      B( 1)  = CP(1,2)
      B( 2)  = CP(2,1)
      B( 3)  = CP(2,3)
      B( 4)  = CP(3,2)
      B( 5)  = CP(3,4)
      B( 6)  = CP(4,3)
      B( 7)  = CP(4,5)
      B( 8)  = CP(5,4)
      B( 9)  = CP(5,1)
      B(10)  = CP(1,5)
      B(11) = Q(1,2)
      B(12) = Q(2,1)
      B(13) = Q(2,3)
      B(14) = Q(3,2)
      B(15) = Q(3,4)
      B(16) = Q(4,3)
      B(17) = Q(4,5)
      B(18) = Q(5,4)
      B(19) = Q(5,1)
      B(20) = Q(1,5)
      B(21) = Q(1,3)
      B(22) = Q(2,4)
      B(23) = Q(3,5)
      B(24) = Q(4,1)
      B(25) = Q(5,2)
      B(26) = Q(1,4)
      B(27) = Q(2,5)
      B(28) = Q(3,1)
      B(29) = Q(4,2)
      B(30) = Q(5,3)
      B(31) = BR(1,2)
      B(32) = BR(2,1)
      B(33) = BR(2,3)
      B(34) = BR(3,2)
      B(35) = BR(3,4)
      B(36) = BR(4,3)
      B(37) = BR(4,5)
      B(38) = BR(5,4)
      B(39) = BR(5,1)
      B(40) = BR(1,5)
      B(41) = BR(1,3)
      B(42) = BR(2,4)
      B(43) = BR(3,5)
      B(44) = BR(4,1)
      B(45) = BR(5,2)
      B(46) = BR(1,4)
      B(47) = BR(2,5)
      B(48) = BR(3,1)
      B(49) = BR(4,2)
      B(50) = BR(5,3)
C-----------------  OBJECTIVE FUNCTION -------------
      C( 1)  = -ALPHA(1,2)*(1.0 - RO(1,2))
      C( 2)  = -ALPHA(2,1)*(1.0 - RO(2,1))
      C( 3)  = -ALPHA(2,3)*(1.0 - RO(2,3))
      C( 4)  = -ALPHA(3,2)*(1.0 - RO(3,2))
      C( 5)  = -ALPHA(3,4)*(1.0 - RO(3,4))
```

```
      C( 6)  = -ALPHA(4,3)*(1.0 - RO(4,3))
      C( 7)  = -ALPHA(4,5)*(1.0 - RO(4,5))
      C( 8)  = -ALPHA(5,4)*(1.0 - RO(5,4))
      C( 9)  = -ALPHA(5,1)*(1.0 - RO(5,1))
      C(10)  = -ALPHA(1,5)*(1.0 - RO(1,5))
      C(11)  = (ALPHA(2,3) - ALPHA(1,3))*(1.0 - RO(1,2))
      C(12)  = (ALPHA(1,3) - ALPHA(2,3))*(1.0 - RO(2,1))
      C(13)  = (ALPHA(3,4) - ALPHA(2,4))*(1.0 - RO(2,3))
      C(14)  = (ALPHA(2,4) - ALPHA(3,4))*(1.0 - RO(3,2))
      C(15)  = (ALPHA(4,5) - ALPHA(3,5))*(1.0 - RO(3,4))
      C(16)  = (ALPHA(3,5) - ALPHA(4,5))*(1.0 - RO(4,3))
      C(17)  = (ALPHA(5,1) - ALPHA(4,1))*(1.0 - RO(4,5))
      C(18)  = (ALPHA(4,1) - ALPHA(5,1))*(1.0 - RO(5,4))
      C(19)  = (ALPHA(1,2) - ALPHA(5,2))*(1.0 - RO(5,1))
      C(20)  = (ALPHA(5,2) - ALPHA(1,2))*(1.0 - RO(1,5))
      C(21)  = (ALPHA(2,4) - ALPHA(1,4))*(1.0 - RO(1,2))
      C(22)  = (ALPHA(1,4) - ALPHA(2,4))*(1.0 - RO(2,1))
      C(23)  = (ALPHA(3,5) - ALPHA(2,5))*(1.0 - RO(2,3))
      C(24)  = (ALPHA(2,5) - ALPHA(3,5))*(1.0 - RO(3,2))
      C(25)  = (ALPHA(4,1) - ALPHA(3,1))*(1.0 - RO(3,4))
      C(26)  = (ALPHA(3,1) - ALPHA(4,1))*(1.0 - RO(4,3))
      C(27)  = (ALPHA(5,2) - ALPHA(4,2))*(1.0 - RO(4,5))
      C(28)  = (ALPHA(4,2) - ALPHA(5,2))*(1.0 - RO(5,4))
      C(29)  = (ALPHA(1,3) - ALPHA(5,3))*(1.0 - RO(5,1))
      C(30)  = (ALPHA(5,3) - ALPHA(1,3))*(1.0 - RO(1,5))
      C(31)  = (ALPHA(2,5) - ALPHA(1,5))*(1.0 - RO(1,2))
      C(32)  = (ALPHA(1,5) - ALPHA(2,5))*(1.0 - RO(2,1))
      C(33)  = (ALPHA(3,1) - ALPHA(2,1))*(1.0 - RO(2,3))
      C(34)  = (ALPHA(2,1) - ALPHA(3,1))*(1.0 - RO(3,2))
      C(35)  = (ALPHA(4,2) - ALPHA(3,2))*(1.0 - RO(3,4))
      C(36)  = (ALPHA(3,2) - ALPHA(4,2))*(1.0 - RO(4,3))
      C(37)  = (ALPHA(5,3) - ALPHA(4,3))*(1.0 - RO(4,5))
      C(38)  = (ALPHA(4,3) - ALPHA(5,3))*(1.0 - RO(5,4))
      C(39)  = (ALPHA(1,4) - ALPHA(5,4))*(1.0 - RO(5,1))
      C(40)  = (ALPHA(5,4) - ALPHA(1,4))*(1.0 - RO(1,5))
C-----------------------------------------------------------------
C SUBMIT THE INITIAL LP TABLEAU ATAB : MIN CX SUBJECT TO AX <= B
C-----------------------------------------------------------------
      DO 50 I = 1,51
      DO 50 J = 1,41
 50   ATAB(I,J) = 0.0
      DO 52 I = 2,M
 52   ATAB(I,1) = B(I-1)
      DO 54 J = 2,N
 54   ATAB(1,J) = C(J-1)
      M1 = M-1
      N1 = N-1
      DO 56 I = 1,M1
      DO 56 J = 1,N1
 56   ATAB(I+1,J+1) = A(I,J)
C-----------------------------------------------------------------
      CALL MATH(M,N,NZR1VR,ATAB,UPBND,ISIZE,IROW,U)
C-----------------------------------------------------------------
      DO 70 I = 1,NM1
 70   T(I) = U(I)
      RETURN
      END
```

```
C*********************************************************************C
      SUBROUTINE OPTDCN(S,BF,Q,SE,ALPHA,RO,CP,T)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION U(41)
      REAL CP(5,5),BF(5,5),SE(5,5),Q(5,5),ALPHA(5,5),
     $   BR(5,5),T(41),RO(5,5)
      INTEGER S,I,J
C---------------- BUFFER RESTRICTION ----------------
      DO 40 I = 1,5
      DO 40 J = 1,5
         IF(I.NE.J) THEN
            BR(I,J) = BF(I,J) - SE(I,J)
         ENDIF
  40  CONTINUE
C-------------------------------------------------------
C      THE FOUR NODES WILL PERFORM THEIR OWN OPTIMIZATION
C-------------------------------------------------------
C---------------------- NODE 1 -----------------------------
      CALL MATX(1,5,2,3,4,CP,BR,Q,ALPHA,RO,U)
      T(10) = U(1)
      T(20) = U(2)
      T(30) = U(3)
      T(40) = U(4)
      T( 1) = U(5)
      T(11) = U(6)
      T(21) = U(7)
      T(31) = U(8)
C---------------------- NODE 2 -----------------------------
      CALL MATX(2,1,3,4,5,CP,BR,Q,ALPHA,RO,U)
      T( 2) = U(1)
      T(12) = U(2)
      T(22) = U(3)
      T(32) = U(4)
      T( 3) = U(5)
      T(13) = U(6)
      T(23) = U(7)
      T(33) = U(8)
C---------------------- NODE 3 -----------------------------
      CALL MATX(3,2,4,5,1,CP,BR,Q,ALPHA,RO,U)
      T( 4) = U(1)
      T(14) = U(2)
      T(24) = U(3)
      T(34) = U(4)
      T( 5) = U(5)
      T(15) = U(6)
      T(25) = U(7)
      T(35) = U(8)
C---------------------- NODE 4 -----------------------------
      CALL MATX(4,3,5,1,2,CP,BR,Q,ALPHA,RO,U)
      T( 6) = U(1)
      T(16) = U(2)
      T(26) = U(3)
      T(36) = U(4)
      T( 7) = U(5)
      T(17) = U(6)
      T(27) = U(7)
```

```
      T(37) = U(8)
C---------------------- NODE 5 ----------------------------
      CALL MATX(5,4,1,2,3,CP,BR,Q,ALPHA,RO,U)
      T( 8) = U(1)
      T(18) = U(2)
      T(28) = U(3)
      T(38) = U(4)
      T( 9) = U(5)
      T(19) = U(6)
      T(29) = U(7)
      T(39) = U(8)
C
      RETURN
      END
C*************************************************************************C
      SUBROUTINE MATX(IA,IB,IC,ID,IE,CP,BR,Q,ALPHA,RO,U)
C*************************************************************************C
C
C       THIS SUBROUTINE CREATE THE DECENTRALIZED INITIAL
C       LINEAR PROGRAMMING TABLEAU.  THE TABLEAUX ARE THE
C       SAME FOR EVERY NODE BECAUSE THE NETWORK IS SYMMETRIC.
C-------------------------------------------------------------------
      IMPLICIT NONE
      DOUBLE PRECISION ATAB(51,41),UPBND(41),U(41)
      REAL A(6,8),B(6),C(8),BR(5,5),RO(5,5),CP(5,5),ALPHA(5,5),Q(5,5)
      INTEGER NZRlVR,ISIZE,I,J,IA,IB,IC,ID,IE,NM1,M,N,N1,M1,IROW(51)
      ISIZE  = 27
      NZRlVR =  8
      M      =  7
      N      =  9
      NM1    =  N - 1
C--------------  UPPER BOUND OF THE VARIABLES ----------
      UPBND(1) = 100.0
      UPBND(2) = BR(IB,IC)
      UPBND(3) = BR(IB,ID)
      UPBND(4) = BR(IB,IE)
      UPBND(5) = 100.0
      UPBND(6) = BR(IC,ID)
      UPBND(7) = BR(IC,IE)
      UPBND(8) = BR(IC,IB)
C-------  INEQUALITY CONSTRAINTS: -1=LE, 0=EQ AND +1=GE ------
      IROW(1)  = 0
      DO 20 I = 2,M
  20  IROW(I) = -1
C------------------  CONSTRAINT MATRIX  ----------------
      DO 30 I = 1, 6
      DO 30 J = 1, 8
  30  A(I,J) = 0.0
      A(1,1) = 1.0
      A(1,2) = 1.0
      A(1,3) = 1.0
      A(1,4) = 1.0
      A(2,5) = 1.0
      A(2,6) = 1.0
      A(2,7) = 1.0
      A(2,8) = 1.0
      A(3,1) = 1.0
```

```
      A(3,8) = 1.0
      A(4,2) = 1.0
      A(4,5) = 1.0
      A(5,3) = 1.0
      A(5,6) = 1.0
      A(6,4) = 1.0
      A(6,7) = 1.0
C------------------  CONSTRAINT B VECTOR  --------------
      B(1) = CP(IA,IB)
      B(2) = CP(IA,IC)
      B(3) =  Q(IA,IB)
      B(4) =  Q(IA,IC)
      B(5) =  Q(IA,ID)
      B(6) =  Q(IA,IE)
C----------------  OBJECTIVE FUNCTION  ----------------
      C(1) =  -ALPHA(IA,IB) *(1.D0 - RO(IA,IB))
      C(2) = ( ALPHA(IB,IC) - ALPHA(IA,IC))*(1.D0 - RO(IA,IB))
      C(3) = ( ALPHA(IB,ID) - ALPHA(IA,ID))*(1.D0 - RO(IA,IB))
      C(4) = ( ALPHA(IB,IE) - ALPHA(IA,IE))*(1.D0 - RO(IA,IB))
      C(5) =  -ALPHA(IA,IC) *(1.D0 - RO(IA,IC))
      C(6) = ( ALPHA(IC,ID) - ALPHA(IA,ID))*(1.D0 - RO(IA,IC))
      C(7) = ( ALPHA(IC,IE) - ALPHA(IA,IE))*(1.D0 - RO(IA,IC))
      C(8) = ( ALPHA(IC,IB) - ALPHA(IA,IB))*(1.D0 - RO(IA,IC))
C
      M1 = M - 1
      N1 = N - 1
      DO 50 I = 1,M
      DO 50 J = 1,N
 50   ATAB(I,J) = 0.0
      DO 52 I = 2,M
 52   ATAB(I,1) = B(I-1)
      DO 54 J = 2,N
 54   ATAB(1,J) = C(J-1)
      DO 56 I = 1,M1
      DO 56 J = 1,N1
 56   ATAB(I+1,J+1) = A(I,J)
C
      CALL MATH(M,N,NZR1VR,ATAB,UPBND,ISIZE,IROW,U)
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE UPDAT(IX,IY,SPC,T)
C*********************************************************************C
      IMPLICIT NONE
      REAL T(41),SPC,RMX,TX,TY
      INTEGER IX,IY
C
      T(41) = 0.0
      IF(SPC.GT.0.0) THEN
         IF((T(IX).GT.0.0).AND.(T(IY).GT.0.0)) THEN
            TX    = SPC * (T(IX)/(T(IX)+T(IY)))
            TY    = SPC - T(IX)
            RMX   = TX - INT(TX)
               IF(RMX.GE.0.5) THEN
                  T(IX) = INT(TX) + 1.0
                  T(IY) = INT(TY)
```

```
                ELSE
                   T(IX) = INT(TX)
                   T(IY) = INT(TY) + 1.0
                ENDIF
          ELSE
                IF(T(IX).GT.0.0) THEN
                   T(IX) = SPC
                ELSE
                   T(IY) = SPC
                ENDIF
          ENDIF
       ELSE
          T(IX) = 0.0
          T(IY) = 0.0
       ENDIF
       RETURN
       END
C*********************************************************************C
       SUBROUTINE ADJUST(S,BF,Q,SE,RRO,T)
C*********************************************************************C
       IMPLICIT NONE
       REAL T(41),P(40),SPC,Q(5,5),SE(5,5),RRO(40),BF(5,5),TQ
       INTEGER S,I
C------  KEEP PREVIOUS SUBOPTIMAL RESULTS FOR RECTIFICATION  --------
       DO 10 I = 1,40
  10   P(I) = T(I)
C----------------  NODE 1  -----------------
       SPC = BF(1,2) - SE(1,2) - Q(1,2) + P(1) + P(20)
       TQ  =  T(19)
       IF(TQ.GT.SPC) THEN
          CALL UPDAT(41,19,SPC,T)
       ENDIF
C
       SPC = BF(1,3) - SE(1,3) - Q(1,3) + P(11) + P(30)
       TQ  =  T(12) + T(29)
       IF(TQ.GT.SPC) THEN
          CALL UPDAT(12,29,SPC,T)
       ENDIF
C
       SPC = BF(1,4) - SE(1,4) - Q(1,4) + P(21) + P(40)
       TQ  =  T(22) + T(39)
       IF(TQ.GT.SPC) THEN
          CALL UPDAT(22,39,SPC,T)
       ENDIF
C
       SPC = BF(1,5) - SE(1,5) - Q(1,5) + P(31) + P(10)
       TQ  =  T(32)
       IF(TQ.GT.SPC) THEN
          CALL UPDAT(41,32,SPC,T)
       ENDIF
C
C----------  NODE 2  -----------------
       SPC = BF(2,1) - SE(2,1) - Q(2,1) + P(33) + P( 2)
       TQ  =  T(34)
       IF(TQ.GT.SPC) THEN
          CALL UPDAT(41,34,SPC,T)
       ENDIF
```

```
C
      SPC = BF(2,3) - SE(2,3) - Q(2,3) + P( 3) + P(12)
      TQ  =  T(11)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,11,SPC,T)
      ENDIF
C
      SPC = BF(2,4) - SE(2,4) - Q(2,4) + P(13) + P(22)
      TQ  =  T(14) + T(21)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(14,21,SPC,T)
      ENDIF
C
      SPC = BF(2,5) - SE(2,5) - Q(2,5) + P(23) + P(32)
      TQ  =  T(24) + T(31)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(24,31,SPC,T)
      ENDIF
C
C----------------  NODE 3  ----------------
      SPC = BF(3,1) - SE(3,1) - Q(3,1) + P(25) + P(34)
      TQ  =  T(26) + T(33)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(26,33,SPC,T)
      ENDIF
C
      SPC = BF(3,2) - SE(3,2) - Q(3,2) + P(35) + P( 4)
      TQ  =  T(36)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,36,SPC,T)
      ENDIF
C
      SPC = BF(3,4) - SE(3,4) - Q(3,4) + P(5) + P(14)
      TQ  =  T(13)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,13,SPC,T)
      ENDIF
C
      SPC = BF(3,5) - SE(3,5) - Q(3,5) + P(15) + P(24)
      TQ  =  T(16) + T(23)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(16,23,SPC,T)
      ENDIF
C
C----------------  NODE 4  ----------------
      SPC = BF(4,1) - SE(4,1) - Q(4,1) + P(17) + P(26)
      TQ  =  T(18) + T(25)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(18,25,SPC,T)
      ENDIF
C
      SPC = BF(4,2) - SE(4,2) - Q(4,2) + P(27) + P(36)
      TQ  =  T(28) + T(35)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(28,35,SPC,T)
      ENDIF
C
```

```
      SPC = BF(4,3) - SE(4,3) - Q(4,3) + P(37) + P( 6)
      TQ  =  T(38)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,38,SPC,T)
      ENDIF
C
      SPC = BF(4,5) - SE(4,5) - Q(4,5) + P( 7) + P(16)
      TQ  =  T(15)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,15,SPC,T)
      ENDIF
C
C----------------  NODE 5  -----------------
      SPC = BF(5,1) - SE(5,1) - Q(5,1) + P( 9) + P(18)
      TQ  =  T(17)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,17,SPC,T)
      ENDIF
C
      SPC = BF(5,2) - SE(5,2) - Q(5,2) + P(19) + P(28)
      TQ  =  T(20) + T(27)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(20,27,SPC,T)
      ENDIF
C
      SPC = BF(5,3) - SE(5,3) - Q(5,3) + P(29) + P(38)
      TQ  =  T(30) + T(37)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(30,37,SPC,T)
      ENDIF
C
      SPC = BF(5,4) - SE(5,4) - Q(5,4) + P(39) + P( 8)
      TQ  =  T(40)
      IF(TQ.GT.SPC) THEN
         CALL UPDAT(41,40,SPC,T)
      ENDIF
C
      RETURN
      END
C*************************************************************
       SUBROUTINE ERROR(ISEED,II,RRO,T)
C*************************************************************
      IMPLICIT NONE
      REAL RRO(40),TL(40),T(41),PRO,ISEED
      INTEGER IR(1),II,TI,NR
      NR = 1
      PRO = RRO(II)
      TI = INT(T(II))
         IF(TI.GT.0.0) THEN
            CALL RNSET(ISEED)
            CALL RNBIN(NR,TI,PRO,IR)
            CALL RNGET(ISEED)
            TL(II) = IR(NR)
         ELSE
            TL(II) = 0.0
         ENDIF
      T(II) = T(II) - TL(II)
```

```
      RETURN
      END
C*********************************************************************C
      SUBROUTINE CALCUL(S,BF,AR,T,Q,TARR,QL,QQ,PLOST,QTOT,QLT,QT)
C*********************************************************************C
      IMPLICIT NONE
      REAL QT(5,5,100),QQ(5,5,100),QLT(5,5,100),QTOT,QIL,AR(5,5,100),
     $   BF(5,5),Q(5,5),QL(5,5),PLOST,T(41),TARR
      INTEGER S,I,J
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            TARR = TARR + AR(I,J,S-1)
         ENDIF
 10   CONTINUE
      Q(1,2) = Q(1,2) + AR(1,2,S-1) - T( 1) - T(20) + T(19)
      Q(2,1) = Q(2,1) + AR(2,1,S-1) - T( 2) - T(33) + T(34)
      Q(2,3) = Q(2,3) + AR(2,3,S-1) - T( 3) - T(12) + T(11)
      Q(3,2) = Q(3,2) + AR(3,2,S-1) - T( 4) - T(35) + T(36)
      Q(3,4) = Q(3,4) + AR(3,4,S-1) - T( 5) - T(14) + T(13)
      Q(4,3) = Q(4,3) + AR(4,3,S-1) - T( 6) - T(37) + T(38)
      Q(4,5) = Q(4,5) + AR(4,5,S-1) - T( 7) - T(16) + T(15)
      Q(5,4) = Q(5,4) + AR(5,4,S-1) - T( 8) - T(39) + T(40)
      Q(5,1) = Q(5,1) + AR(5,1,S-1) - T( 9) - T(18) + T(17)
      Q(1,5) = Q(1,5) + AR(1,5,S-1) - T(10) - T(31) + T(32)
      Q(1,3) = Q(1,3) + AR(1,3,S-1) - T(11) - T(30) + T(12) + T(29)
      Q(2,4) = Q(2,4) + AR(2,4,S-1) - T(13) - T(22) + T(14) + T(21)
      Q(3,5) = Q(3,5) + AR(3,5,S-1) - T(15) - T(24) + T(16) + T(23)
      Q(4,1) = Q(4,1) + AR(4,1,S-1) - T(17) - T(26) + T(18) + T(25)
      Q(5,2) = Q(5,2) + AR(5,2,S-1) - T(19) - T(28) + T(20) + T(27)
      Q(1,4) = Q(1,4) + AR(1,4,S-1) - T(21) - T(40) + T(22) + T(39)
      Q(2,5) = Q(2,5) + AR(2,5,S-1) - T(23) - T(32) + T(24) + T(31)
      Q(3,1) = Q(3,1) + AR(3,1,S-1) - T(25) - T(34) + T(26) + T(33)
      Q(4,2) = Q(4,2) + AR(4,2,S-1) - T(27) - T(36) + T(28) + T(35)
      Q(5,3) = Q(5,3) + AR(5,3,S-1) - T(29) - T(38) + T(30) + T(37)
C
      DO 20 I = 1,5
      DO 20 J = 1,5
         IF (I.NE.J) THEN
            IF(Q(I,J).LE.BF(I,J)) THEN
               QL(I,J) = 0.0
               IF(Q(I,J).LT.0.0) THEN
                  WRITE(*,*)'ERROR'
               ENDIF
            ELSE
               QL(I,J) = Q(I,J) - BF(I,J)
            ENDIF
            QQ(I,J,S) = Q(I,J)
            Q(I,J)    = Q(I,J) - QL(I,J)
            PLOST     = PLOST + QL(I,J)
         ENDIF
 20   CONTINUE
C
      DO 30 I = 1,5
      DO 30 J = 1,5
      QT(I,J,S) = Q(I,J)
      QLT(I,J,S) = QL(I,J)
```

```
 30   QTOT = QTOT + QT(I,J,S)
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE STATC(PUT,TARR,NITER,WT,WL,WA,WH)
C*********************************************************************
      IMPLICIT NONE
      REAL PUT,TARR,WT(10),WL(10),WA(10),WH(10),MN1,MN2,MN3,MN4,ROOT,
     $  VA1,VA2,VA3,VA4,SD1,SD2,SD3,SD4,STDNT,PCT1,PCT2,PCT3,PCT4,TR
      INTEGER NITER,I
C
      STDNT = 2.571
      MN1 = 0.0
      MN2 = 0.0
      MN3 = 0.0
      MN4 = 0.0
      DO 10 I = 1,NITER
      MN1 = MN1 + WT(I)
      MN2 = MN2 + WL(I)
      MN3 = MN3 + WA(I)
 10   MN4 = MN4 + WH(I)
      MN1 = MN1 / NITER
      MN2 = MN2 / NITER
      MN3 = MN3 / NITER
      MN4 = MN4 / NITER
C
      VA1 = 0.0
      VA2 = 0.0
      VA3 = 0.0
      VA4 = 0.0
      DO 50 I = 1,NITER
      VA1 = VA1 + ((WT(I) - MN1)**2)
      VA2 = VA2 + ((WL(I) - MN2)**2)
      VA3 = VA3 + ((WA(I) - MN3)**2)
  50  VA4 = VA4 + ((WH(I) - MN4)**2)
      VA1 = VA1 / (NITER - 1)
      VA2 = VA2 / (NITER - 1)
      VA3 = VA3 / (NITER - 1)
      VA4 = VA4 / (NITER - 1)
      SD1 = SQRT(VA1)
      SD2 = SQRT(VA2)
      SD3 = SQRT(VA3)
      SD4 = SQRT(VA4)
      TR  = FLOAT(NITER)
      ROOT = SQRT(TR)
      PCT1 = (SD1 * STDNT * 100.0) / (ROOT * MN1)
      PCT2 = (SD2 * STDNT * 100.0) / (ROOT * MN2)
      PCT3 = (SD3 * STDNT * 100.0) / (ROOT * MN3)
      PCT4 = (SD4 * STDNT * 100.0) / (ROOT * MN4)
C
   4  FORMAT(3F10.2,2F10.4)
   5  FORMAT(4F10.2,2F10.4)
   7  FORMAT(20X,4F10.2)
      WRITE(*,4) PUT,MN1,MN2,MN3,MN4
C     WRITE(*,5) PUT,TARR,MN1,MN2,MN3,MN4
C     WRITE(*,7) PCT1,PCT2,PCT3,PCT4
```

```
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE TEST(QQ,QT,QLT,AR,ER)
C*********************************************************************C
      IMPLICIT NONE
      REAL ER(5,5,100),AR(5,5,100),QT(5,5,100),QQ(5,5,100),QLT(5,5,100)
      INTEGER K,IS,NM1
      COMMON /IO/ K,NM1
  5   FORMAT(I5,4F5.0)
C
C     WRITE(*,*)'TOTAL NUMBER OF PACKETS IN BUFFER'
C     DO 20 IS = 1,K
C20   WRITE(6,5) QT(1,2,IS),QT(1,3,IS),QT(1,4,IS),QT(1,5,IS),
C    $           QT(2,1,IS),QT(2,3,IS),QT(2,4,IS),QT(2,5,IS),
C    $           QT(3,1,IS),QT(3,2,IS),QT(3,4,IS),QT(3,5,IS),
C    $           QT(4,1,IS),QT(4,2,IS),QT(4,3,IS),QT(4,5,IS),
C    $           QT(5,1,IS),QT(5,2,IS),QT(5,3,IS),QT(5,4,IS)
C     WRITE(*,*)'TOTAL NUMBER OF PACKETS LOST'
C     DO 22 IS = 1,K
C22   WRITE(6,5) QLT(1,2,IS), QLT(1,3,IS), QLT(1,4,IS), QLT(1,5,IS),
C    $           QLT(2,1,IS), QLT(2,3,IS), QLT(2,4,IS), QLT(2,5,IS),
C    $           QLT(3,1,IS), QLT(3,2,IS), QLT(3,4,IS), QLT(3,5,IS),
C    $           QLT(4,1,IS), QLT(4,2,IS), QLT(4,3,IS), QLT(4,5,IS),
C    $           QLT(5,1,IS), QLT(5,2,IS), QLT(5,3,IS), QLT(5,4,IS)
C     WRITE(*,*)'TOTAL NUMBER OF PACKETS IN'
      DO 23 IS = 1,K
C23   WRITE(6,5) IS,QQ(1,2,IS),QQ(1,3,IS),QQ(1,4,IS),QQ(1,5,IS)
C23   WRITE(6,5) IS,QQ(2,1,IS),QQ(2,3,IS),QQ(2,4,IS),QQ(2,5,IS)
C23   WRITE(6,5) IS,QQ(3,1,IS),QQ(3,2,IS),QQ(3,4,IS),QQ(3,5,IS)
C23   WRITE(6,5) IS,QQ(4,1,IS),QQ(4,2,IS),QQ(4,3,IS),QQ(4,5,IS)
 23   WRITE(6,5) IS,QQ(5,1,IS),QQ(5,2,IS),QQ(5,3,IS),QQ(5,4,IS)
C     WRITE(6,*) '       MEAN ARRIVAL AT EACH BUFFER'
C     DO 24 IS = 1,K-1
C24   WRITE(6,5) ER(1,2,IS),ER(1,3,IS),ER(1,4,IS),ER(1,5,IS),
C    $           ER(2,1,IS),ER(2,3,IS),ER(2,4,IS),ER(2,5,IS),
C    $           ER(3,1,IS),ER(3,2,IS),ER(3,4,IS),ER(3,5,IS),
C    $           ER(4,1,IS),ER(4,2,IS),ER(4,3,IS),ER(4,5,IS),
C    $           ER(5,1,IS),ER(5,2,IS),ER(5,3,IS),ER(5,4,IS)
C     WRITE(6,*) '        EXTERNAL ARRIVAL AT EACH NODE'
C     DO 25 IS = 1,K-1
C25   WRITE(6,5) AR(1,2,IS),AR(1,3,IS),AR(1,4,IS),AR(1,5,IS),
C    $           AR(2,1,IS),AR(2,3,IS),AR(2,4,IS),AR(2,5,IS),
C    $           AR(3,1,IS),AR(3,2,IS),AR(3,4,IS),AR(3,5,IS),
C    $           AR(4,1,IS),AR(4,2,IS),AR(4,3,IS),AR(4,5,IS),
C    $           AR(5,1,IS),AR(5,2,IS),AR(5,3,IS),AR(5,4,IS)
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE COMPT(L,PLOST,TARR,QT,QTOT,PUT,WT,WL,WA,WH)
C*********************************************************************C
      IMPLICIT NONE
      REAL AVD,THR,QTOT,QINTL,QBAL,PLOST,QT(5,5,100),TARR,PUT,WT(10),
     $     WL(10),WA(10),WH(10)
      INTEGER K,L,I,J,NM1
```

```
      COMMON /IO/ K,NM1
      QINTL = 0.0
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            QINTL = QINTL + QT(I,J,1)
            QBAL  = QINTL - QT(I,J,K)
         ENDIF
   10 CONTINUE
      AVD    = QTOT / (TARR + QINTL - PLOST)
      THR    = ( TARR + QINTL - PLOST)/ K
C
      WT(L)  = QTOT
      WL(L)  = PLOST
      WA(L)  = AVD
      WH(L)  = THR
C
      RETURN
      END
```

Single buffer simulation
. Main program

```
C*********************************************************************C
C       SINGLE BUFFER CASE FOR CENTRALIZED ROUTING
C*********************************************************************C
      IMPLICIT NONE
      REAL CP(5,5),RO(5,5),RRO(40),BF(5),QL(5,5),PLOST,TARR,PUT,
     $  Q(5,5),QT(5,100),QLT(5,100),QQ(5,5,100),T(41),QTOT,BR(5),GQ(5),
     $  ER(5,5,100),AR(5,5,100),ALPHA(5,5),WT(10),WL(10),WA(10),WH(10)
      INTEGER S,ISEED,K,NM1,I,II,NITER,L,NARVL
      COMMON /IO/ K,NM1
C
      NARVL  =  1
C
      NITER  =  10
C
      PUT    =  40.0
C
C     DO 100 I = 1,100
         DO 50 L = 1,NITER
            CALL STSEED(L,ISEED)
            K      =  99
            PLOST  = 0.0
            QTOT   = 0.0
            TARR   = 0.0
C
            CALL STOPO(CP,RO,RRO,BF)
            CALL SINITL(Q,QT,QLT,QQ)
            CALL ARRVL(NARVL,PUT,ISEED,ER,AR)
            S = 1
 20         IF(S.LT.K) THEN
                 CALL SEWFAC(S,BF,Q,ER,ALPHA)
                 CALL CNTBUF(S,BF,Q,ALPHA,ER,RO,CP,T)
               DO 10 II = 1,40
  10           CALL ERROR(ISEED,II,RRO,T)
               S = S + 1
               CALL SALCUL(S,ISEED,BF,AR,T,Q,TARR,QL,QQ,
     $                     PLOST,QTOT,QLT,QT)
               GO TO 20
            ENDIF
            CALL STEST(QQ,QT,QLT,AR,ER)
            CALL SCOMPT(L,PLOST,TARR,QT,QTOT,PUT,WT,WL,WA,WH)
   50    CONTINUE
C
         CALL STATC(PUT,TARR,NITER,WT,WL,WA,WH)
         PUT = PUT + 1.0
 100  CONTINUE
      STOP
      END
```

. Subroutines

```
C*********************************************************************C
      SUBROUTINE STOPO(CP,RO,RRO,BF)
C*********************************************************************C
      IMPLICIT NONE
      REAL CP(5,5),RO(5,5),RRO(40),BF(5)
      INTEGER I,J
C----------  CAPACITY AND ERROR RATE OF CHANNELS  ------------
      DO 10 I = 1,5
      DO 10 J = 1,5
         CP(I,J) = 0.0
 10      RO(I,J) = 0.0
      CP(1,2) = 100.0
      CP(2,1) = 100.0
      CP(2,3) = 100.0
      CP(3,2) = 100.0
      CP(3,4) = 100.0
      CP(4,3) = 100.0
      CP(4,5) = 100.0
      CP(5,4) = 100.0
      CP(5,1) = 100.0
      CP(1,5) = 100.0
C
      RO(1,2) = 1E-2
      RO(2,1) = 5E-3
      RO(2,3) = 2E-3
      RO(3,2) = 9E-3
      RO(3,4) = 1E-2
      RO(4,3) = 7E-3
      RO(4,5) = 2E-3
      RO(5,4) = 9E-3
      RO(5,1) = 4E-3
      RO(1,5) = 1E-2
C-------------  ERROR VECTOR  ------------
      DO 20 J = 1,4
      I = 10 * (J-1)
      RRO(I + 1 ) = RO(1,2)
      RRO(I + 2 ) = RO(2,1)
      RRO(I + 3 ) = RO(2,3)
      RRO(I + 4 ) = RO(3,2)
      RRO(I + 5 ) = RO(3,4)
      RRO(I + 6 ) = RO(4,3)
      RRO(I + 7 ) = RO(4,5)
      RRO(I + 8 ) = RO(5,4)
      RRO(I + 9 ) = RO(5,1)
      RRO(I + 10) = RO(1,5)
 20   CONTINUE
C-------------   BUFFER   ----------------
      DO 30 I = 1,5
 30      BF(I) = 400.0
      RETURN
      END
```

```
C*********************************************************************
      SUBROUTINE SINITL(Q,QT,QLT,QQ)
C*********************************************************************
      IMPLICIT NONE
      REAL Q(5,5),QT(5,100),QLT(5,100),QQ(5,100)
      INTEGER I,J
      DO 10 I = 1,5
         QT(I,1)  = 0.0
      DO 20 J = 1,5
         IF(I.NE.J) THEN
            Q(I,J) =  50.0
            QT(I,1)  = QT(I,1) + Q(I,J)
         ELSE
            Q(I,J) =  0.0
         ENDIF
 20      CONTINUE
         QQ(I,1)  = QT(I,1)
         QLT(I,1) = 0.0
 10   CONTINUE
      RETURN
      END
C*********************************************************************C
      SUBROUTINE SEWFAC(S,BF,Q,ER,ALPHA)
C*********************************************************************C
      IMPLICIT NONE
      REAL BF(5),ER(5,5,100),Q(5,5),ALPHA(5,5)
      INTEGER S,I,J
C---------------  SET WEIGTHING FACTOR ------------
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            ALPHA(I,J) = 1 + (( Q(I,J) + ER(I,J,S) ) / BF(I))
         ENDIF
  10  CONTINUE
      RETURN
      END
C*********************************************************************C
      SUBROUTINE CNTBUF(S,BF,Q,ALPHA,ER,RO,CP,T)
C*********************************************************************C
      IMPLICIT NONE
      DOUBLE PRECISION ATAB(51,41), UPBND(41),U(41)
      REAL A(50,40),B(50),C(40),ROOM,GA(5),GQ(5),RO(5,5),
     $ BOUND,CP(5,5),BF(5),Q(5,5),ALPHA(5,5),BR(5),T(41),ER(5,5,100)
      INTEGER S,M,N,NM1,K,ISIZE,NZR1VR,I,J,M1,N1,II,IROW(51)
      COMMON /IO/ K,NM1
C
      ISIZE  = 90
      M      = 36
      N      = 41
      NZR1VR = 40
      NM1    = N - 1
C------------- UPPER BOUND ON THE VARIABLES ----------
      DO 10 I = 1,NM1
  10  UPBND(I) = 100.0
C------- INEQUALITY OF CONSTRAINTS: -1=LE, 0=EQ AND +1=GE ----
      IROW(1) = 0
      DO 20 I = 2,M
```

```
  20  IROW(I) = -1
C--------------  CONSTRAINT MATRIX -----------
      DO 30 I = 1, 50
      DO 30 J = 1, 40
  30  A(I,J) = 0.0
      DO 40 I = 1,10
      A(I,I) = 1.0
      A(I,I+10) =  1.0
      A(I,I+20) =  1.0
      A(I,I+30) =  1.0
      A(I+10,I) =  1.0
  40  CONTINUE
      A(11,20) = 1.D0
      A(12,33) = 1.D0
      A(13,12) = 1.D0
      A(14,35) = 1.D0
      A(15,14) = 1.D0
      A(16,37) = 1.D0
      A(17,16) = 1.D0
      A(18,39) = 1.D0
      A(19,18) = 1.D0
      A(20,31) = 1.D0
      A(21,11) = 1.D0
      A(21,30) = 1.D0
      A(22,13) = 1.D0
      A(22,22) = 1.D0
      A(23,15) = 1.D0
      A(23,24) = 1.D0
      A(24,17) = 1.D0
      A(24,26) = 1.D0
      A(25,19) = 1.D0
      A(25,28) = 1.D0
      A(26,21) = 1.D0
      A(26,40) = 1.D0
      A(27,23) = 1.D0
      A(27,32) = 1.D0
      A(28,25) = 1.D0
      A(28,34) = 1.D0
      A(29,27) = 1.D0
      A(29,36) = 1.D0
      A(30,29) = 1.D0
      A(30,38) = 1.D0
C
      A(31, 1) =-1.D0
      A(31,11) =-1.D0
      A(31,21) =-1.D0
      A(31,31) =-1.D0
      A(31,10) =-1.D0
      A(31,20) =-1.D0
      A(31,30) =-1.D0
      A(31,40) =-1.D0
      A(31,12) = 1.D0
      A(31,22) = 1.D0
      A(31,32) = 1.D0
      A(31,19) = 1.D0
      A(31,29) = 1.D0
      A(31,39) = 1.D0
```

```
      A(32, 2) =-1.D0
      A(32,12) =-1.D0
      A(32,22) =-1.D0
      A(32,32) =-1.D0
      A(32, 3) =-1.D0
      A(32,13) =-1.D0
      A(32,23) =-1.D0
      A(32,33) =-1.D0
      A(32,11) = 1.D0
      A(32,21) = 1.D0
      A(32,31) = 1.D0
      A(32,14) = 1.D0
      A(32,24) = 1.D0
      A(32,34) = 1.D0
      A(33, 4) =-1.D0
      A(33,14) =-1.D0
      A(33,24) =-1.D0
      A(33,34) =-1.D0
      A(33, 5) =-1.D0
      A(33,15) =-1.D0
      A(33,25) =-1.D0
      A(33,35) =-1.D0
      A(33,36) = 1.D0
      A(33,13) = 1.D0
      A(33,16) = 1.D0
      A(33,26) = 1.D0
      A(33,23) = 1.D0
      A(33,33) = 1.D0
      A(34, 6) =-1.D0
      A(34,16) =-1.D0
      A(34,26) =-1.D0
      A(34,36) =-1.D0
      A(34, 7) =-1.D0
      A(34,17) =-1.D0
      A(34,27) =-1.D0
      A(34,37) =-1.D0
      A(34,18) = 1.D0
      A(34,28) = 1.D0
      A(34,38) = 1.D0
      A(34,15) = 1.D0
      A(34,25) = 1.D0
      A(34,35) = 1.D0
      A(35, 8) =-1.D0
      A(35,18) =-1.D0
      A(35,28) =-1.D0
      A(35,38) =-1.D0
      A(35, 9) =-1.D0
      A(35,19) =-1.D0
      A(35,29) =-1.D0
      A(35,39) =-1.D0
      A(35,17) = 1.D0
      A(35,27) = 1.D0
      A(35,37) = 1.D0
      A(35,20) = 1.D0
      A(35,30) = 1.D0
      A(35,40) = 1.D0
C-------------  BUFFER RESTRICTIONS  -----------------
```

```
      DO 45 I = 1,5
         GQ(I)   = 0.0
         GA(I) = 0.0
      DO 46 J = 1,5
         IF(I.NE.J) THEN
            GQ(I) = GQ(I) + Q(I,J)
            GA(I) = GA(I) + ER(I,J,S)
         ENDIF
  46  CONTINUE
         BR(I) = BF(I) - GQ(I) - GA(I)
         IF(BR(I).LT.0.0) THEN
            BR(I) = 0.0
         ENDIF
  45  CONTINUE
C--------------- CONSTRAINT B VECTOR ----------------
      B( 1)  = CP(1,2)
      B( 2)  = CP(2,1)
      B( 3)  = CP(2,3)
      B( 4)  = CP(3,2`
      B( 5)  = CP(3,4)
      B( 6)  = CP(4,3)
      B( 7)  = CP(4,5)
      B( 8)  = CP(5,4)
      B( 9)  = CP(5,1)
      B(10)  = CP(1,5)
      B(11)  = Q(1,2)
      B(12)  = Q(2,1)
      B(13)  = Q(2,3)
      B(14)  = Q(3,2)
      B(15)  = Q(3,4)
      B(16)  = Q(4,3)
      B(17)  = Q(4,5)
      B(18)  = Q(5,4)
      B(19)  = Q(5,1)
      B(20)  = Q(1,5)
      B(21)  = Q(1,3)
      B(22)  = Q(2,4)
      B(23)  = Q(3,5)
      B(24)  = Q(4,1)
      B(25)  = Q(5,2)
      B(26)  = Q(1,4)
      B(27)  = Q(2,5)
      B(28)  = Q(3,1)
      B(29)  = Q(4,2)
      B(30)  = Q(5,3)
      B(31)  = BR(1)
      B(32)  = BR(2)
      B(33)  = BR(3)
      B(34)  = BR(4)
      B(35)  = BR(5)
C----------------  OBJECTIVE FUNCTION ------------
      C( 1)  = -ALPHA(1,2)*(1.0 - RO(1,2))
      C( 2)  = -ALPHA(2,1)*(1.0 - RO(2,1))
      C( 3)  = -ALPHA(2,3)*(1.0 - RO(2,3))
      C( 4)  = -ALPHA(3,2)*(1.0 - RO(3,2))
      C( 5)  = -ALPHA(3,4)*(1.0 - RO(3,4))
      C( 6)  = -ALPHA(4,3)*(1.0 - RO(4,3))
```

```
      C( 7)  = -ALPHA(4,5)*(1.0 - RO(4,5))
      C( 8)  = -ALPHA(5,4)*(1.0 - RO(5,4))
      C( 9)  = -ALPHA(5,1)*(1.0 - RO(5,1))
      C(10)  = -ALPHA(1,5)*(1.0 - RO(1,5))
      C(11)  = (ALPHA(2,3) - ALPHA(1,3))*(1.0 - RO(1,2))
      C(12)  = (ALPHA(1,3) - ALPHA(2,3))*(1.0 - RO(2,1))
      C(13)  = (ALPHA(3,4) - ALPHA(2,4))*(1.0 - RO(2,3))
      C(14)  = (ALPHA(2,4) - ALPHA(3,4))*(1.0 - RO(3,2))
      C(15)  = (ALPHA(4,5) - ALPHA(3,5))*(1.0 - RO(3,4))
      C(16)  = (ALPHA(3,5) - ALPHA(4,5))*(1.0 - RO(4,3))
      C(17)  = (ALPHA(5,1) - ALPHA(4,1))*(1.0 - RO(4,5))
      C(18)  = (ALPHA(4,1) - ALPHA(5,1))*(1.0 - RO(5,4))
      C(19)  = (ALPHA(1,2) - ALPHA(5,2))*(1.0 - RO(5,1))
      C(20)  = (ALPHA(5,2) - ALPHA(1,2))*(1.0 - RO(1,5))
      C(21)  = (ALPHA(2,4) - ALPHA(1,4))*(1.0 - RO(1,2))
      C(22)  = (ALPHA(1,4) - ALPHA(2,4))*(1.0 - RO(2,1))
      C(23)  = (ALPHA(3,5) - ALPHA(2,5))*(1.0 - RO(2,3))
      C(24)  = (ALPHA(2,5) - ALPHA(3,5))*(1.0 - RO(3,2))
      C(25)  = (ALPHA(4,1) - ALPHA(3,1))*(1.0 - RO(3,4))
      C(26)  = (ALPHA(3,1) - ALPHA(4,1))*(1.0 - RO(4,3))
      C(27)  = (ALPHA(5,2) - ALPHA(4,2))*(1.0 - RO(4,5))
      C(28)  = (ALPHA(4,2) - ALPHA(5,2))*(1.0 - RO(5,4))
      C(29)  = (ALPHA(1,3) - ALPHA(5,3))*(1.0 - RO(5,1))
      C(30)  = (ALPHA(5,3) - ALPHA(1,3))*(1.0 - RO(1,5))
      C(31)  = (ALPHA(2,5) - ALPHA(1,5))*(1.0 - RO(1,2))
      C(32)  = (ALPHA(1,5) - ALPHA(2,5))*(1.0 - RO(2,1))
      C(33)  = (ALPHA(3,1) - ALPHA(2,1))*(1.0 - RO(2,3))
      C(34)  = (ALPHA(2,1) - ALPHA(3,1))*(1.0 - RO(3,2))
      C(35)  = (ALPHA(4,2) - ALPHA(3,2))*(1.0 - RO(3,4))
      C(36)  = (ALPHA(3,2) - ALPHA(4,2))*(1.0 - RO(4,3))
      C(37)  = (ALPHA(5,3) - ALPHA(4,3))*(1.0 - RO(4,5))
      C(38)  = (ALPHA(4,3) - ALPHA(5,3))*(1.0 - RO(5,4))
      C(39)  = (ALPHA(1,4) - ALPHA(5,4))*(1.0 - RO(5,1))
      C(40)  = (ALPHA(5,4) - ALPHA(1,4))*(1.0 - RO(1,5))
C-------------------------------------------------------------
C SUBMIT THE INITIAL LP TABLEAU ATAB : MIN CX SUBJECT TO AX <= B
C-------------------------------------------------------------
      DO 50 I = 1,51
      DO 50 J = 1,41
 50   ATAB(I,J) = 0.0
      DO 52 I = 2,M
 52   ATAB(I,1) = B(I-1)
      DO 54 J = 2,N
 54   ATAB(1,J) = C(J-1)
      M1 = M-1
      N1 = N-1
      DO 56 I = 1,M1
      DO 56 J = 1,N1
 56   ATAB(I+1,J+1) = A(I,J)
C-------------------------------------------------------------
      CALL MATH(M,N,NZR1VR,ATAB,UPBND,ISIZE,IROW,U)
C-------------------------------------------------------------
      DO 70 I = 1,NM1
 70   T(I) = U(I)
C
      RETURN
      END
```

```
C*********************************************************************C
      SUBROUTINE SALCUL(S,ISEED,BF,AR,T,Q,TARR,QL,QQ,PLOST,QTOT,QLT,QT)
C*********************************************************************C
      IMPLICIT NONE
      REAL QT(5,100),QQ(5,100),QLT(5,100),QTOT,QIL,AR(5,5,100),
     $   BF(5),Q(5,5),GQ(5),QL(5),PLOST,T(41),TARR,GE(5)
      INTEGER S,I,J,ISEED
      DO 10 I = 1,5
      DO 10 J = 1,5
         IF(I.NE.J) THEN
            TARR = TARR + AR(I,J,S-1)
         ENDIF
 10   CONTINUE
      Q(1,2) = Q(1,2) + AR(1,2,S-1) - T( 1) - T(20) + T(19)
      Q(2,1) = Q(2,1) + AR(2,1,S-1) - T( 2) - T(33) + T(34)
      Q(2,3) = Q(2,3) + AR(2,3,S-1) - T( 3) - T(12) + T(11)
      Q(3,2) = Q(3,2) + AR(3,2,S-1) - T( 4) - T(35) + T(36)
      Q(3,4) = Q(3,4) + AR(3,4,S-1) - T( 5) - T(14) + T(13)
      Q(4,3) = Q(4,3) + AR(4,3,S-1) - T( 6) - T(37) + T(38)
      Q(4,5) = Q(4,5) + AR(4,5,S-1) - T( 7) - T(16) + T(15)
      Q(5,4) = Q(5,4) + AR(5,4,S-1) - T( 8) - T(39) + T(40)
      Q(5,1) = Q(5,1) + AR(5,1,S-1) - T( 9) - T(18) + T(17)
      Q(1,5) = Q(1,5) + AR(1,5,S-1) - T(10) - T(31) + T(32)
      Q(1,3) = Q(1,3) + AR(1,3,S-1) - T(11) - T(30) + T(12) + T(29)
      Q(2,4) = Q(2,4) + AR(2,4,S-1) - T(13) - T(22) + T(14) + T(21)
      Q(3,5) = Q(3,5) + AR(3,5,S-1) - T(15) - T(24) + T(16) + T(23)
      Q(4,1) = Q(4,1) + AR(4,1,S-1) - T(17) - T(26) + T(18) + T(25)
      Q(5,2) = Q(5,2) + AR(5,2,S-1) - T(19) - T(28) + T(20) + T(27)
      Q(1,4) = Q(1,4) + AR(1,4,S-1) - T(21) - T(40) + T(22) + T(39)
      Q(2,5) = Q(2,5) + AR(2,5,S-1) - T(23) - T(32) + T(24) + T(31)
      Q(3,1) = Q(3,1) + AR(3,1,S-1) - T(25) - T(34) + T(26) + T(33)
      Q(4,2) = Q(4,2) + AR(4,2,S-1) - T(27) - T(36) + T(28) + T(35)
      Q(5,3) = Q(5,3) + AR(5,3,S-1) - T(29) - T(38) + T(30) + T(37)
C
      DO 20 I = 1,5
         GQ(I) = 0.0
         GE(I) = 0.0
         DO 30 J = 1,5
            IF (I.NE.J) THEN
              GQ(I) = GQ(I) + Q(I,J)
              GE(I) = GE(I) + AR(I,J,S)
            ENDIF
 30      CONTINUE
         IF(GQ(I).LE.BF(I)) THEN
            QL(I) = 0.0
            IF(GQ(I).LT.0.0) THEN
               WRITE(*,*)'ERROR'
            ENDIF
         ELSE
            QL(I) = GQ(I) - BF(I)
            CALL QDLM(S,ISEED,I,GE,QL,Q,AR)
         ENDIF
         QQ(I,S) = GQ(I)
         GQ(I)   = GQ(I) - QL(I)
         PLOST   = PLOST + QL(I)
 20   CONTINUE
```

```
C
      DO 40 I = 1,5
      QT(I,S) = GQ(I)
      QLT(I,S) = QL(I)
 40   QTOT = QTOT + QT(I,S)
C
      RETURN
      END
C*********************************************************************
      SUBROUTINE QDLM(S,ISEED,I,GE,QL,Q,AR)
C*********************************************************************
      IMPLICIT NONE
      REAL QL(5),Q(5,5),GE(5),AR(5,5,100)
      INTEGER S,I,J,ISEED,IA,IB,IC,ID,XTQ,XE,AQ(4),AT(4)
      IF(I.EQ.1) THEN
         IA = 2
         IB = 3
         IC = 4
         ID = 5
      ELSE
         IF(I.EQ.2) THEN
            IA = 1
            IB = 3
            IC = 4
            ID = 5
         ELSE
            IF(I.EQ.3) THEN
               IA = 1
               IB = 2
               IC = 4
               ID = 5
            ELSE
               IF(I.EQ.4) THEN
                  IA = 1
                  IB = 2
                  IC = 3
                  ID = 5
               ELSE
                  IA = 1
                  IB = 2
                  IC = 3
                  ID = 4
               ENDIF
            ENDIF
         ENDIF
      ENDIF
      XTQ = INT(QL(I))
      XE  = INT(GE(I))
      AT(1) = INT(AR(I,IA,S))
      AT(2) = INT(AR(I,IB,S))
      AT(3) = INT(AR(I,IC,S))
      AT(4) = INT(AR(I,ID,S))
C
      IF(AT(1).GT.0) THEN
         CALL RNSET(ISEED)
         CALL RNHYP(1,XTQ,AT(1),XE,AQ(1))
         CALL RNGET(ISEED)
```

```
      ELSE
         AQ(1) = 0
      ENDIF
      XTQ = XTQ - AQ(1)
      XE  = XE  - AT(1)
      IF(AT(2).GT.0) THEN
         CALL RNSET(ISEED)
         CALL RNHYP(1,XTQ,AT(2),XE,AQ(2))
         CALL RNGET(ISEED)
      ELSE
         AQ(2) = 0
      ENDIF
      XTQ = XTQ - AQ(2)
      XE  = XE  - AT(2)
      IF(AT(3).GT.0) THEN
         CALL RNSET(ISEED)
         CALL RNHYP(1,XTQ,AT(3),XE,AQ(3))
         CALL RNGET(ISEED)
      ELSE
         AQ(3) = 0
      ENDIF
      XTQ = XTQ - AQ(3)
      XE  = XE  - AT(3)
      AQ(4) = XTQ
      IF(XE.NE.AT(4)) THEN
         WRITE(*,*)'ERROR'
      ENDIF
C
      Q(I,IA) = Q(I,IA) - AQ(1)
      Q(I,IB) = Q(I,IB) - AQ(2)
      Q(I,IC) = Q(I,IC) - AQ(3)
      Q(I,ID) = Q(I,ID) - AQ(4)
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE STEST(QQ,QT,QLT,AR,ER)
C*********************************************************************C
      IMPLICIT NONE
      REAL ER(5,5,100),AR(5,5,100),QT(5,100),QQ(5,100),QLT(5,100)
      INTEGER K,IS,NM1
      COMMON /IO/ K,NM1
  5   FORMAT(20F5.0)
 10   FORMAT(I5,5F10.0)
C
C     WRITE(*,*)'TOTAL NUMBER OF PACKETS IN BUFFER'
C     DO 20 IS = 1,K
C20   WRITE(6,10) QT(1,IS),QT(2,IS),QT(3,IS),QT(4,IS),QT(5,IS)
C     WRITE(*,*)'TOTAL NUMBER OF PACKETS LOST'
C     DO 22 IS = 1,K
C22   WRITE(6,10) QLT(1,IS),QLT(2,IS),QLT(3,IS),QLT(4,IS),QLT(5,IS)
      WRITE(*,*)'TOTAL NUMBER OF PACKETS IN'
      DO 23 IS = 1,K
 23   WRITE(6,10) IS,QQ(1,IS),QQ(2,IS),QQ(3,IS),QQ(4,IS),QQ(5,IS)
C     WRITE(6,*) '       MEAN ARRIVAL AT EACH BUFFER'
C     DO 24 IS = 1,K-1
C24   WRITE(6,5) ER(1,2,IS),ER(1,3,IS),ER(1,4,IS),ER(1,5,IS),
```

```
C     $            ER(2,1,IS),ER(2,3,IS),ER(2,4,IS),ER(2,5,IS),
C     $            ER(3,1,IS),ER(3,2,IS),ER(3,4,IS),ER(3,5,IS),
C     $            ER(4,1,IS),ER(4,2,IS),ER(4,3,IS),ER(4,5,IS),
C     $            ER(5,1,IS),ER(5,2,IS),ER(5,3,IS),ER(5,4,IS)
C      WRITE(6,*) '        EXTERNAL ARRIVAL AT EACH NODE'
C      DO 25 IS = 1,K-1
C25    WRITE(6,5) AR(1,2,IS),AR(1,3,IS),AR(1,4,IS),AR(1,5,IS),
C     $            AR(2,1,IS),AR(2,3,IS),AR(2,4,IS),AR(2,5,IS),
C     $            AR(3,1,IS),AR(3,2,IS),AR(3,4,IS),AR(3,5,IS),
C     $            AR(4,1,IS),AR(4,2,IS),AR(4,3,IS),AR(4,5,IS),
C     $            AR(5,1,IS),AR(5,2,IS),AR(5,3,IS),AR(5,4,IS)
C
      RETURN
      END
C*********************************************************************C
      SUBROUTINE SCOMPT(L,PLOST,TARR,QT,QTOT,PUT,WT,WL,WA,WH)
C*********************************************************************C
      IMPLICIT NONE
      REAL AVD,THR,QTOT,QINTL,QBAL,PLOST,QT(5,100),TARR,PUT,WT(10),
     $     WL(10),WA(10),WH(10)
      INTEGER K,L,I,J,NM1
      COMMON /IO/ K,NM1
      QINTL = 0.0
      DO 10 I = 1,5
         QINTL = QINTL + QT(I,1)
         QBAL  = QINTL - QT(I,K)
  10  CONTINUE
      AVD    = QTOT / (TARR + QINTL - PLOST)
      THR    = ( TARR + QINTL - PLOST)/ K
C
      WT(L)  = QTOT
      WL(L)  = PLOST
      WA(L)  = AVD
      WH(L)  = THR
C
      RETURN
      END
```