Transient Analysis of Markovian Queueing Systems and Its Application to Congestion-Control Modeling

HARMEN R. VAN AS, MEMBER, IEEE

Abstract—**In applied queueing theory, it is often important to deal with transient system behavior. Performance evaluation of congestion-control mechanisms in a packet-switching network is an excellent example in which there is frequently a strong need to deal with the intrinsic dynamic character of congestion. In that case, the queueing models have to be analyzed for a transient environment. In this paper, we show that such problems can be treated in a uniform way, when the system of coupled differential equations describing the system-state or flow process is solved numerically. For this, the fourth-order Runge-Kutta procedure allows a good balance between memory requirements, computing time, and accuracy. To illustrate the explanatory power of this kind of transient queueing analysis, three models will be considered: the common-store queueing system showing the priority deadlock, the foreground-background congestion-control mechanism, and a two-level global congestion-control mechanism.**

I. Introduction

FOR the analysis of queueing systems exposed to a transient environment, it is often essential to reflect the dynamic system behavior. In this paper, it is shown that a straightforward numerical evaluation of the coupled differential equations describing the system-state or flow process is a very useful and uniform approach for transient investigations. Particularly, we demonstrate the application of this technique for congestion-control modeling of packet-switching networks. Here, the following problem exists. Whereas packet-switching technology has an excellent ability to share communication resources dynamically among a number of active users, serious performance degradations may result. Then, in an uncontrolled network, throughput typically increases linearly with growing offered traffic up to a maximum, and then decreases very rapidly. In addition, network transfer times become intolerably long. This performance degradation is caused by packet retransmissions and waiting for transmission capacity or nodal buffers which are free but not accessible. In extreme cases, a deadlock may occur. Because of time-out mechanisms, network performance can degrade even more, when traffic sources start to retransmit packets whose acknowledgments have not returned within the specified period of time and thus are presumed to be lost. To avoid a decrease in performance, an effective system of well-matched flow control, congestion control, and time-out mechanisms is necessary. A general overview on this subject is given in [7], [11], [12]. In this paper, only a few topics are addressed to illustrate the numerical approach to transient queueing analysis. In particular, we show the potential for a priority deadlock in a common-store queueing system, some dynamic properties of the foreground-background (FG-BG) congestion-control mechanism, and some interesting effects in a two-level global congestion-control mechanism.

Manuscript received June 12, 1985. This work was performed at the University of Siegen, West Germany.

The author is with the IBM Zurich Research Laboratory, 8803 Rüschlikon, Switzerland.

IEEE Log Number 8609935.

II. Analytical Technique

Although we consider a multi-dimensional class of Markovian queueing models, the theoretical fundamentals will be described for a one-dimensional Markovian process $\{X(t) = x,\ t \geq 0,\ x \in M$: state space$\}$. It is assumed that the process is in state i at time s and in state j at a later time t. The probability for this state transition is defined by the time-dependent transition probability $p_{ij}(s, t)$ for which one can derive the Kolmogorov forward differential equation

$$\frac{d}{dt} p_{ij}(s, t) = -q_j(t) \cdot p_{ij}(s, t) + \sum_{k \neq j} p_{ik}(s, t) \cdot q_{kj}(t), \qquad s < t, \quad (1)$$

and the Kolmogorov backward differential equation

$$\frac{d}{ds} p_{ij}(s, t) = -q_i(s) \cdot p_{ij}(s, t) + \sum_{k \neq i} q_{ik}(s) \cdot p_{kj}(s, t), \qquad s < t, \quad (2)$$

where $q_j(t)$, $q_{kj}(t)$, and $q_i(s)$, $q_{ik}(s)$ are time-dependent transition rates, and states $i, j, k \in M$ [6].

From these two sets of fundamental differential equations for the transition probabilities $p_{ij}(s, t)$, the differential equations describing the system-state process and those describing the waiting or flow processes are derived as will be described in the next two subsections.

A. System-State Process

The time-dependent state probabilities $P_j(t) = \text{Prob}\{X(t) = j\}$ are the basis for a major part of the transient queueing analysis considered in this paper. The state probabilities at time t are obtained from an initial system-state distribution $P_i(0)$ at time $s = 0$ and the transition probabilities $p_{ij}(s, t)$ by

$$P_j(t) = \sum_i P_i(0) \cdot p_{ij}(0, t). \quad (3)$$

0733-8716/86/0900-0891$01.00 © 1986 IEEE

Moreover, the normalization condition at time t must be taken into account,

$$\sum_j P_j(t) = 1. \tag{4}$$

If one inserts (1) into (3), the Kolmogorov forward differential equation for the time-dependent state probabilities at time t is obtained,

$$\frac{d}{dt} P_j(t) = -q_j(t) \cdot P_j(t) + \sum_{k \neq j} q_{kj}(t) \cdot P_k(t). \tag{5}$$

The complete set of differential equations is derived, for instance, from a system-state diagram by inspection. To do this, one has to apply the rule that the rate of flow change into the state j, given by $d/dt\ P_j(t)$, must be equal to the difference between the rate at which the system enters and leaves state j. Examples will be given in Sections IV-A and V-A. In the special case of transition rates which do not vary in time, and a system which has reached its steady state, the corresponding linear equations for the state probabilities P_j of the homogeneous Markovian process are obtained by setting all differential quotients $d/dt\ P_j(t)$ equal to zero.

B. Waiting or Flow Process

For the determination of waiting or flow-time measures in a Markovian queueing system with a complex structure or with time-dependent transition rates, the waiting or flow process is a very powerful concept [8], [9]. This technique is also called the method of first passage times or the life-time process. For instance, to obtain performance measures for the flow-time random variable $T_F(s)$, we consider the fate of a test packet. This test packet arrives at an arbitrary time s, and if accepted by the system, its flow process will begin. In this way, the fate of the test packet can be traced, while all possible influences from other packets can be taken into account. The flow process ends with an absorbing state which might be the end of service, for example. Thus, in contrast to the system-state process, a flow process has a finite duration. The flow process can be considered as a special system-state process with a set H of absorbing states from which a test packet does not return.

Mathematically, this flow process is described by the Kolmogorov backward differential equations for the conditional complementary flow-time distribution functions $f_i(s, t)$. This function is defined as the probability that the flow-time random variable $T_F(s)$ of the test packet is longer than the time interval $\bar{t} = t - s$, when the test packet encounters system state i at its arrival time s,

$$f_i(s, t) = \text{Prob}\ \{T_F(s) > \bar{t} | i\}. \tag{6}$$

The flow-time random variable is longer than $\bar{t}$ when the test packet remains outside the set H of absorbing states during that time interval. Therefore, the conditional complementary flow-time distribution function can be expressed as the sum of all transition probabilities outside the absorbing state space H. Thus,

$$f_i(s, t) = \sum_{j \in H} \tilde{p}_{ij}(s, t). \tag{7}$$

The special notation with a tilde should indicate that the transition probabilities of the flow process are different from those of the corresponding system-state process.

By summation of (2) according to (7), the Kolmogorov backward differential equation for the conditional complementary flow-time distribution functions $f_i(s, t)$ is obtained,

$$\frac{d}{ds} f_i(s, t) = -\tilde{q}_i(s) \cdot f_i(s, t) + \sum_{\substack{k \neq i \\ k \notin H}} \tilde{q}_{ik}(s) \cdot f_k(s, t), \qquad s < t. \tag{8}$$

Because each test packet accepted has a flow time larger than zero, the initial condition is

$$f_i(s, s) = 1. \tag{9}$$

It is advantageous to specify the flow process by a diagram containing all states a test packet can visit during its flow process. For some queueing models, the flow-process diagram also contains states which cannot be entered at the arrival time, but only as the flow process proceeds. We make the assumption that the flow-process diagram does not contain the test packet itself. Furthermore, if the test packet encounters a full system at its arrival time, it will be rejected. The flow process diagram must be constructed in such a way that it takes into account all packets influencing the fate of the test packet. An example will be given in Section V-A.

The complete set of Kolmogorov backward differential equations is obtained from the flow process diagram by inspection and application of the following rules:

- only transitions leaving state i must be considered,
- transition rate $\tilde{q}_i(s)$ is the sum of all rates leaving state i,
- each transition rate $\tilde{q}_{ik}(s)$ with $k \neq i$ is the transition rate from state i to its neighboring state k.

The complementary flow-time distribution for the test packet arriving at time s is finally obtained by weighting each function $f_i(s, t)$ for state i with its corresponding state probability $P_i(s)$ at arrival time,

$$F^c(s, t) = \sum_{\bar{M}} P_i(s) \cdot f_i(s, t), \tag{10}$$

with $\bar{M}$: all states which can be encountered by an arriving test packet.

III. Evaluation Technique

The numerical evaluation of both the time-dependent state probabilities $P_j(t)$ according to (5) and the conditional complementary flow-time distribution functions $f_i(s, t)$ according to (8) is done by solving the set of simultaneous first-order differential equations starting from a steady-state as an initial condition. In particular, the fourth-order Runge–Kutta procedure has been used [10], [13]. With respect to computing time, memory require-

ments, and accuracy, this method is very efficient for large systems of differential equations as they may occur in complex Markovian models. An algorithm especially tailored for this application has been published in [1].

Consider the coupled set of differential equations for the state probabilities. In the fourth-order Runge–Kutta procedure, the state probabilities $P_j(t + h)$ at the time $t + h$ are calculated from $P_j(t)$ at time t by taking into account four intermediate evaluations given by the Runge–Kutta (RK) coefficients k_{1j}, k_{2j}, k_{3j}, k_{4j},

$$P_j(t + h) = P_j(t) + \frac{h}{6} \cdot [k_{1j} + 2k_{2j} + 2k_{3j} + k_{4j}], \tag{11}$$

where h is the step of integration.

The RK-coefficients are obtained by evaluation of the system of differential equations according to (5) with transition rates valid at time t_x and with the values $Z_j(t_x)$ as specified below for the state probabilities,

$$k_{mj} = -q_j(t_x) \cdot Z_j(t_x) + \sum_{k \neq j} q_{kj}(t_x) \cdot Z_k(t_x), \qquad m = 1, \cdots, 4, \tag{12}$$

where

K_{1j} is evaluated at time $t_x = t$, k_{2j} at time $t_x = t + h/2$, k_{3j} at time $t_x = t + h/2$, and k_{4j} at time $t_x = t + h$

with

$$\begin{aligned} Z_j(t_x = t) &= P_j(t), \\ Z_j(t_x = t + h/2) &= P_j(t) + k_{1j} \cdot h/2, \\ Z_j(t_x = t + h/2) &= P_j(t) + k_{2j} \cdot h/2, \\ Z_j(t_x = t + h) &= P_j(t) + k_{3j} \cdot h. \end{aligned}$$

Since in the fourth-order Runge–Kutta procedure, the RK-coefficients are needed only once per integration step, memory space can be saved by immediately carrying out the weighted summation of (11). In this version, the memory requirement for the calculation of the time-dependent state probabilities is only four times the size of the state space. For each state j, one needs old state probability, new state probability or an intermediate evaluation, sum of the RK-coefficients, and the current RK-coefficient. Moreover, to reduce computing time, an automatic step-size control based on a time-dependent factor

$$D_{\max}(t + h) = \max_j |P_j(t + h)| \tag{13}$$

has been introduced. It reflects system dynamics at time $t + h$. For high system dynamics, $D_{\max}(t + h)$ differs largely from its preceding value $D_{\max}(t)$. In this case, a small step-size h is required, whereas for a small difference a larger step size is possible.

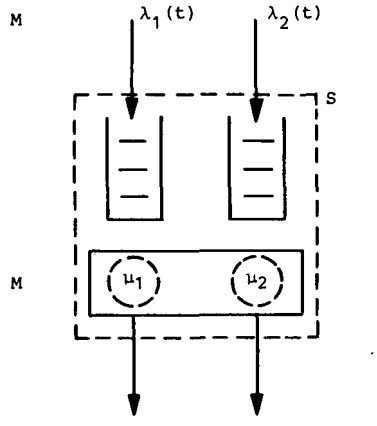

Fig. 1. Common-store queueing system.

IV. Priority Deadlock

In a first example illustrating the application of transient queueing analysis, we show the priority deadlock which may occur in a common-store queueing system operated by a nonpreemptive priority rule [4]. The priority deadlock is a situation in which packets of the high-priority traffic class must be discarded owing to lack of store caused by backlogged low-priority traffic. This effect can be demonstrated most effectively by considering system dynamics as a response to a rectangular-shaped overload peak.

Fig. 1 shows the model considered. It consists of a single-stage queueing system with one server and two nonpreemptive priority traffic classes sharing a finite store S which includes the packet in service. It is assumed that each packet needs exactly one buffer. Packets of the high-priority class arrive according to a Poisson process with time-dependent arrival rate $\lambda_1(t)$. The same holds for packets of the low-priority class which arrive with rate $\lambda_2(t)$. High-priority packets are always served before low-priority ones, but they cannot interrupt a low-priority packet in service. The queueing model represents the complete system capacity of a network node, where packets leave the node with a rate μ_1 or μ_2 depending on its class. The service times are assumed to be negative exponentially distributed. This means, we assume Markovian (M) conditions. The number of high-priority packets in the system is denoted by X_1, and correspondingly, the number of low-priority packets by X_2, so that an arriving packet can be accepted when $X_1 + X_2 < S$.

A. Analysis

In Fig. 2, the structure of the three-dimensional system-state diagram is given. Each state is denoted by the triplet (i, j, k), where

- i Number of high-priority packets in the system, $i = 0, \cdots, S$.
- j Number of low-priority packets in the system, $j = 0, \cdots, S$.
- k Server state, $k = 0$: empty system, $k = 1$: service of a high-priority packet, and $k = 2$: service of a low-priority packet.

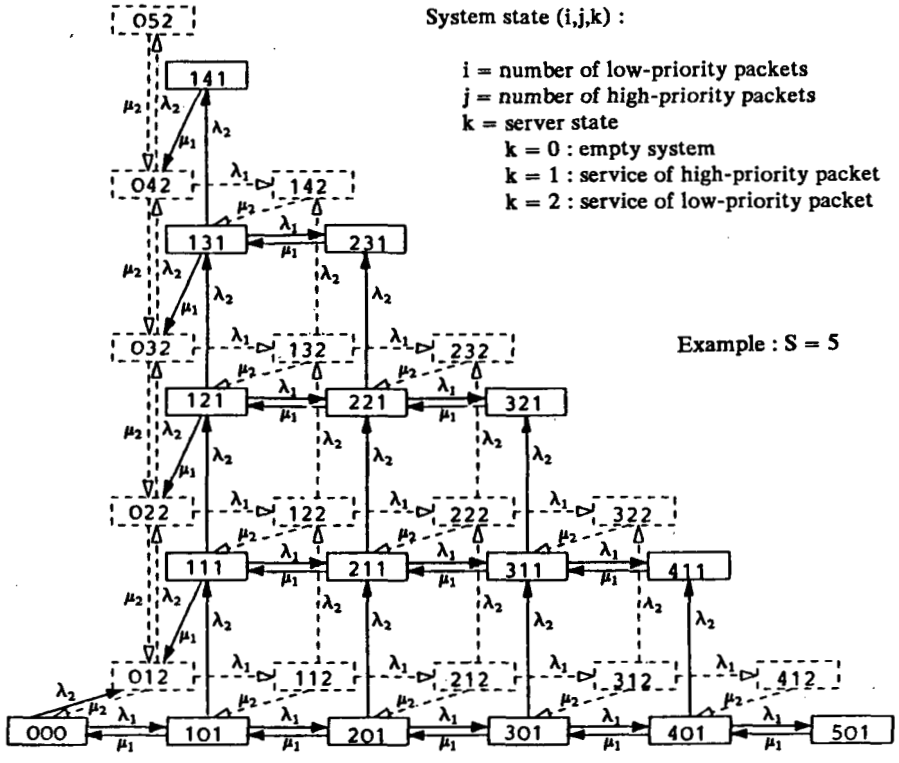

Fig. 2. System-state diagram for the common-store queueing system.

In this diagram, the empty state (0, 0, 0), all states (i, j, 1) representing the service of a high-priority packet, and the corresponding state-transition arrows have been drawn solid. The states (i, j, 2) for a low-priority packet in service and its transitions are dashed. Because of the common store, the system-state diagram is bounded diagonally.

The state of the system at time t is described by the random variables $X_1(t)$ for the number of high-priority packets, $X_2(t)$ for the number of low-priority packets, and $Y(t)$ for the server state, so that the time-dependent state probabilities can be defined by $P_{ijk}(t) = \text{Prob}\,\{X_1(t) = i, X_2(t) = j, Y(t) = k\}$.

From the system-state diagram, the Kolmogorov forward differential equations needed for the time-dependent analysis can easily be obtained by the rule given in Section II-A. For example, the differential equations for states (2, 2, 1) and (1, 3, 2) are found to be

$$\frac{d}{dt}P_{221}(t) = -[\lambda_1(t) + \lambda_2(t) + \mu_1] \cdot P_{221}(t) + \lambda_1(t) \cdot P_{121}(t) + \lambda_2(t) \cdot P_{211}(t) + \mu_1 \cdot P_{321}(t) + \mu_2 \cdot P_{232}(t),$$

$$\frac{d}{dt}P_{132}(t) = -[\lambda_1(t) + \lambda_2(t) + \mu_2] \cdot P_{132}(t) + \lambda_1(t) \cdot P_{032}(t) + \lambda_2(t) \cdot P_{122}(t). \quad (15)$$

The complete set of differential equations has been specified by using 14 types of equations. Each row of states which has to be considered separately is defined by three equation types: left boundary, middle field, and right boundary. In the system-state diagram of Fig. 2, equation types have to be defined for the state rows beginning at the left side with states (0, 0, 0), (0, 1, 2), (1, 1, 1), or (0, 2, 2). Furthermore, the two upper states (1, 4, 1) and (0, 5, 2) have to be treated individually.

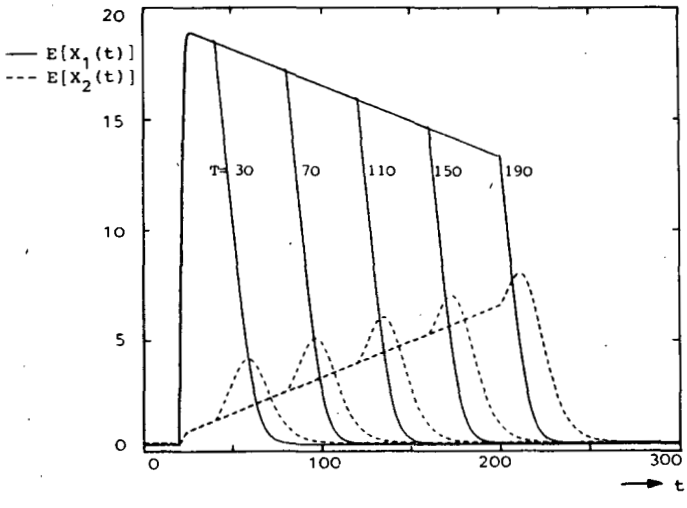

Fig. 3. Mean system occupancies $E[X_1(t)]$ and $E[X_2(t)]$ showing a potential priority deadlock. High-priority traffic overload peak with duration T = 30, 70, 110, 150, 190 starting at t = 10. Traffic parameters: $\lambda_1(\infty) = \lambda_2(\infty) = 0.2$, $\lambda_{1\max} = 6.0$, $h_1 = h_2 = 1$.

B. Performance Measures

From the time-dependent state probabilities, we define the following two performance measures. The range of values for i and j follows from Fig. 2.

• *Mean system occupancy for high-priority packets*, given by the arithmetic mean of having i high-priority packets in the system weighted by the corresponding state probabilities,

$$E[X_1(t)] = \sum_i \sum_j i \cdot [P_{ij1}(t) + P_{ij2}(t)]. \quad (16)$$

• *Mean system occupancy for low-priority packets*, given by the arithmetic mean of having j low-priority packets in the system weighted by the corresponding state probabilities,

$$E[X_2(t)] = \sum_i \sum_j j \cdot [P_{ij1}(t) + P_{ij2}(t)]. \quad (17)$$

C. Numerical Results and Discussion

In order to demonstrate the priority deadlock in such a common-store queueing system, we consider a finite store with capacity $S = 20$ which includes the buffer for the packet in service. The mean service times of the priority classes have been chosen equal and they are identical to the time unit: $h_1 = 1/\mu_1 = 1$ and $h_2 = 1/\mu_2 = 1$. Thus, packets leave the queueing system with the unified service rate $\mu = 1$. At the normalized time $t = 0$, the system is in steady state with the arrival rates $\lambda_1(\infty) = \lambda_2(\infty) = 0.2$. To create an overload peak, the arrival rate $\lambda_1(t)$ of the high-priority traffic class is changed according to a rectangular function with maximum value $\lambda_{1\max} = 6.0$. The overload peak starts at the normalized time $t = 10$ and ends after an overload duration T varying from $T = 30$ to $T = 190$. Although the numerical method used allows for arbitrary changes of the arrival rates, a simple rectangular overload peak facilitates interpretation of the results.

In Fig. 3, the mean system occupancies $E[X_1(t)]$ and $E[X_2(t)]$ of both traffic classes have been plotted. Because low-priority packets must always wait until all high-priority packets have been served, a surplus of low-priority packets builds-up, which obstructs the acceptance of high-priority packets. In the same proportion as the mean system occupany $E[X_2(t)]$ increases, $E[X_1(t)]$ becomes smaller. The speed at which low-priority packets are backlogged depends on the arrival rate $\lambda_2(\infty)$. Because of the backlogged low-priority packets, the part of the common store which remains for high-priority packets reduces significantly. For a long overload peak of high-priority packets, in the long run the common store will be filled up completely by backlogged low-priority packets. In such a situation, the common store becomes useless for high-priority packets, and for those packets the queueing system is transformed into a loss system. To avoid this priority deadlock, a part of the common store must be reserved for the high-priority class.

V. FG-BG Congestion—Control Mechanism

The next example demonstrates some dynamic properties of the foreground-background (FG-BG) congestion-control mechanism [1], [5], recently introduced. It is based on selective removal of packets to a large and cheap background store. The fundamental idea of the mechanism is to let the network nodes autonomously cope with short overload peaks, instead of shifting the overload situation in an uncontrolled manner to neighboring nodes. The improvement in network performance results from the reduced number of packet rejections, so that fewer packet retransmissions are required. Thus, in fact, the blind traffic in the network becomes smaller. For its application, the packet-switching network must be able to distinguish between at least two priority classes. The first priority class is assigned to network management packets and packets for real-time or interactive applications.

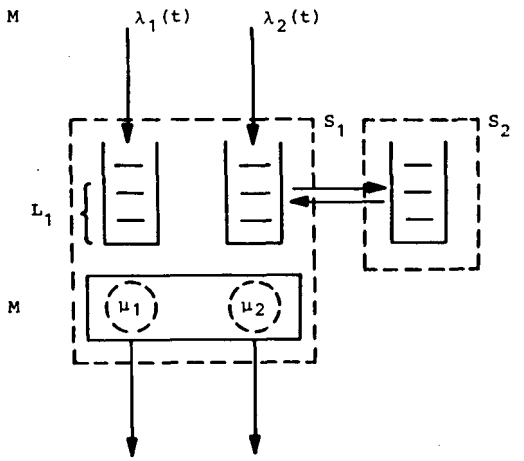

Fig. 4. Queueing model for the FG-BG congestion-control mechanism.

Packets for batch applications can be served with a lower priority, since they have less stringent delivery-time requirements. In this context, it should be mentioned that the time-outs for automatic retransmission of packets whose acknowledgments do not return to the source within the specified time are set differently for both priority classes. The priority class of the packet is identified by the packet header. For implementation of the FG-BG congestion-control mechanism, each network node must have, in addition to its normal switching store (foreground), a large and cheap secondary store (background). The background store is used to swap out low-priority packets during a short period of time. Then, because of longer tolerable delivery times, the low-priority packets can be handled in off-peak periods without penalty. So, this dynamic buffer management provides foreground store on demand for high-priority packets by transferring low-priority packets to the background store. In a traffic environment with a large share of batch traffic, a part of the foreground store should be permanently reserved for high-priority packets as will be shown later. In this way, it is prevented that packets for batch applications can hook up all buffers in both the foreground and the background stores, and the essential network management packets must be rejected. Apart from physical limitations, the storage access limits are software parameters and can easily be adapted to the traffic environment expected.

A. Analysis

Fig. 4 illustrates the queueing model for the performance evaluation of the FG-BG congestion-control mechanism. Essentially, it consists of a single-server queueing system with two nonpreemptive priority classes and queue-length limitations. Both priority classes are served at different rates. According to the queueing discipline, high-priority packets are always treated before low-priority ones, but they cannot preempt a packet of the low-priority class when it is in service. Furthermore, we assume that each packet needs exactly one buffer.

The system capacity accessible to each of the priority classes is determined by the following three parameters:

- S_1 Number of buffers in the foreground store accessible to both priority classes. It includes the buffer for the packet in service, too.
- S_2 Number of buffers in the background store used only for low-priority packets.
- L_1 Number of buffers in the foreground store reserved for high-priority packets.

The queueing model represents the total system capacity of a network node in which packets leave the node with a priority-dependent rate μ_1 and μ_2, respectively. It has been assumed that compared to packet transmission times the processing time for storage management can be neglected. The arrival processes of both priority classes have been modeled as Poisson processes. The time-dependent arrival rate of high-priority packets is denoted by $\lambda_1(t)$, the rate for low-priority packets by $\lambda_2(t)$. The stochastic service times are assumed to be negative exponentially distributed with mean $h_1 = 1/\mu_1$ and $h_2 = 1/\mu_2$, respectively. Thus, we are again dealing with a Markovian queueing model. Denoting the number of high-priority packets by X_1 and correspondingly, the number of low-priority packets by X_2, we can set up the following acceptance rules for the packets of both priority classes.

- High-priority packets are accepted if

$$X_1 + X_2 < S_1 + S_2 \quad \text{and}$$
$$\begin{cases} X_1 < S_1 - 1, & \text{if a low-priority packet is in service,} \\ X_1 < S_1, & \text{otherwise.} \end{cases} \tag{18}$$

- Low-priority packets are accepted if

$$X_1 + X_2 < S_1 + S_2 \quad \text{and} \quad X_2 < S_1 + S_2 - L_1. \tag{19}$$

System-State Process: Fig. 5 shows the structure of the system-state diagram for the FG-BG congestion-control mechanism. Each state is characterized by the triplet (i, j, k) where

- i Number of high-priority packets in the system, $i = 0, \cdots, S_1$.
- j Number of low-priority packets in the system, $j = 0, \cdots, S_1 + S_2 - L_1$.

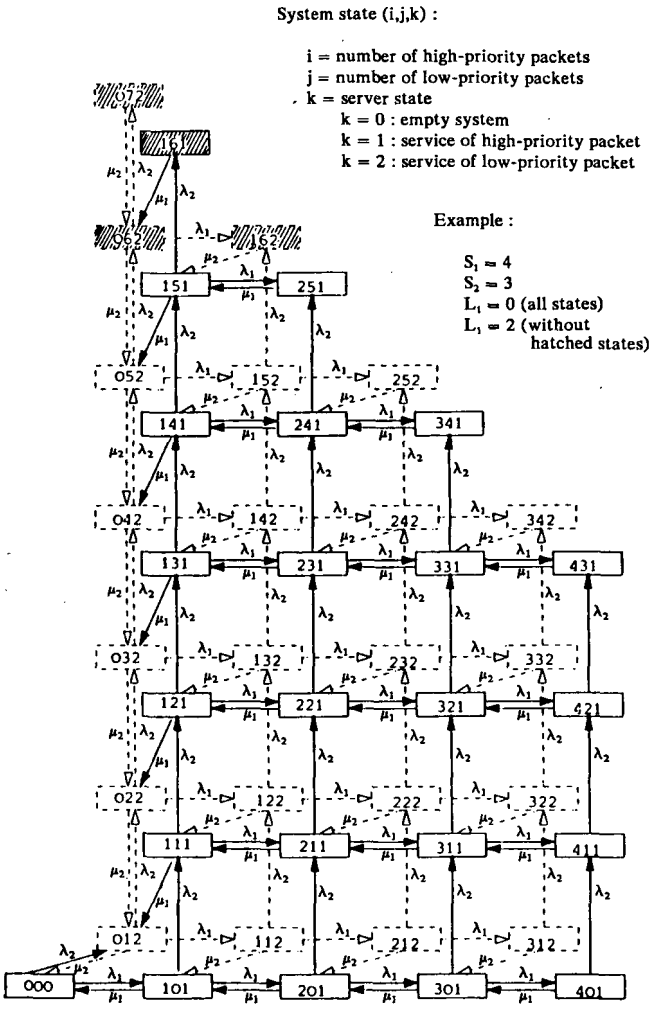

Fig. 5. System-state diagram for the FG-BG congestion-control mechanism.

- k Server state, $k = 0$: empty, $k = 1$: high-priority packet in service, and $k = 2$: low-priority packet in service.

The empty system state (0, 0, 0) and all states $(i, j, 1)$ denoting the service of a high-priority packet have been drawn solid. All states $(i, j, 2)$ for a low-priority packet in service are dashed. The same holds for the corresponding state-transition arrows. Since the high-priority packets have only access to the capacity S_1 of the foreground store, the system-state diagram is bounded at the right side by the solid states $(S_1, j, 1)$. In the dashed-state plane, this boundary is formed by states $(S_1 - 1, j, 2)$, since the buffer for the low-priority packet in service also belongs to the capacity S_1 of the foreground store. The possible buffer reservation L_1 for high-priority packets has been indicated by the hatched states.

In the system-state diagram, the partly common usage of the foreground store is taken into account by the diagonal boundary. Because of the nonpreemptive priority mode of operation, the high-priority class is served next whenever a service ends and high-priority packets are present. When a low-priority packet leaves, it causes a

change from the dashed to the solid-state plane. However, when no high-priority packets are waiting, the next low-priority packet is served denoted by the left dashed column formed by states $(0, j, 2)$. After service of the last packet, the system returns to the empty state $(0, 0, 0)$.

The Kolmogorov forward differential equations are again obtained from the system-state diagram by inspection. For example, the differential equations for the states in the lowest row of the diagram are given by

$$\frac{d}{dt}P_{000}(t) = -[\lambda_1(t) + \lambda_2(t)] \cdot P_{000}(t) + \mu_1 \cdot P_{101}(t) + \mu_2 \cdot P_{012}(t),$$

$$\frac{d}{dt}P_{i,01}(t) = -[\lambda_1(t) + \lambda_2(t) + \mu_1] \cdot P_{i,01}(t) + \lambda_1(t) \cdot P_{i-1,01}(t) + \mu_1 \cdot P_{i+1,01}(t) + \mu_2 \cdot P_{i,12}(t), \quad i = 1, \cdots, S_1 - 1, \quad \text{with index } 001 \equiv 000,$$

$$\frac{d}{dt}P_{S_1,01}(t) = -[\lambda_2(t) + \mu_1] \cdot P_{S_1,01}(t) + \lambda_1(t) \cdot P_{S_1-1,01}(t). \tag{20}$$

Considering the case without reservation for high-priority packets, 20 types of equations are now needed to specify the complete set of differential equations. In the case of Fig. 5, equation types have to be defined for the state rows beginning at the left side with states (0, 0, 0), (0, 1, 2), (1, 1, 1), or (0, 2, 2) in the rectangular part of the system-state diagram, and with the states (1, 3, 1) or (0, 4, 2) in the triangular part. Finally, the two upper states (1, 6, 1) and (0, 7, 2) define the last two equation types.

Flow Process: The mean flow time $t_{F1}(s)$ of a high-priority test packet arriving at time s can be derived for the first-in/first-out queueing discipline with nonpreemptive priorities from the system state encountered. Then, in this case, the fate of the high-priority test packet considered is no longer influenced by packets arriving at a later time $t > s$. For other queueing disciplines or for determination of the mean flow time $t_{F2}(s)$ of low-priority test packets, the concept of a flow process as described in Section II-B must be used.

In the flow process for a low-priority test packet, we define a state (i, j, k) in the following way:

- i Number of high-priority packets served before the test packet of the low-priority class. These high-priority packets are either already present at the arrival time s or they arrive during the waiting time of the low-priority test packet.
- j Number of low-priority packets served before the test packet of the low-priority class. These low-priority packets are already present at the arrival time s.

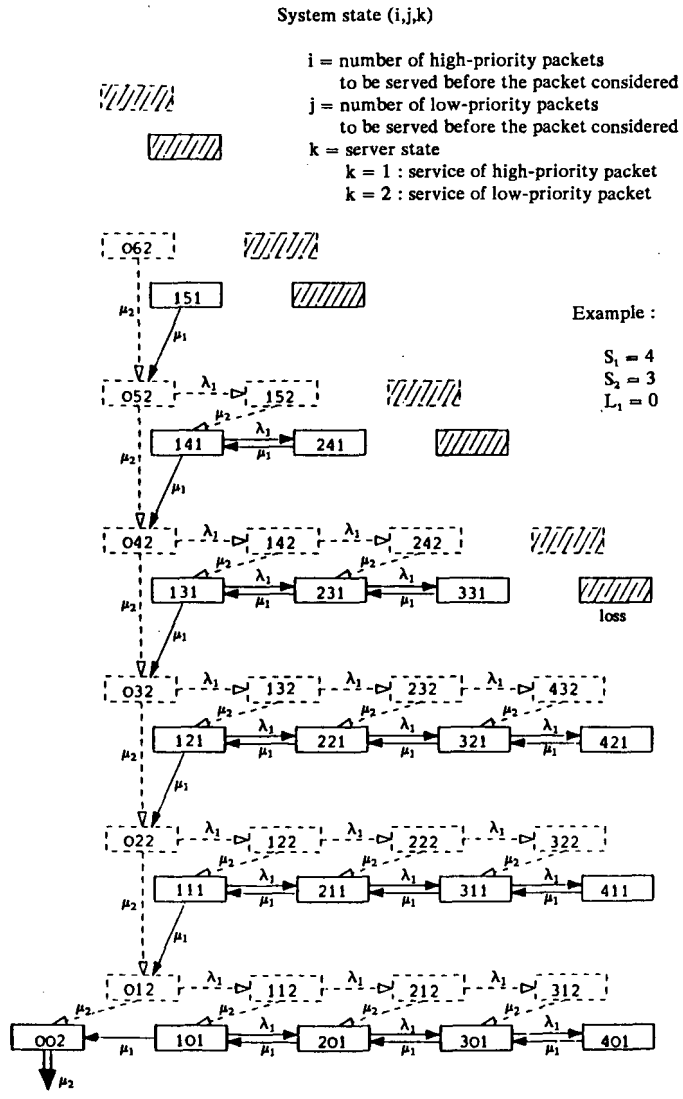

Fig. 6. Flow-process diagram for a low-priority test packet for the FG-BG congestion-control mechanism.

- k State of the server, where $k = 1$ means the service of a high-priority packet, and $k = 2$ the service of a low-priority packet or the test packet of the low-priority class itself.

Fig. 6 illustrates the flow process for a low-priority test packet. The queueing discipline is first-in/first-out with nonpreemptive priorities. Furthermore, the figure relates to the case where no buffers are reserved for high-priority packets. This diagram shows all system-state patterns which come into question for the initial state of the flow process. These system-state patterns do not contain the test packet itself. With respect to packets arriving after the test packet, only high-priority packets are to be considered. If the test packet encounters a full system at its arrival time, it is rejected. This is denoted by the hatched states. In all other states, the flow process starts and ends after service of the test packet marked by the bold arrow from state (0, 0, 2). This "empty" system state indicates that the test packet can no longer be influenced by other

packets. As indicated in the diagram, the test packet may be delayed by high-priority packets which arrive later, but are still served earlier. However, for these newly arriving high-priority packets, it must be taken into account that one buffer is needed for the low-priority test packet. The system states not accessible for the high-priority packets arriving after the test packet, are exactly the hatched states. Thus, these states are excluded from the flow-process diagram.

Let us take an example. A test packet of the low-priority class encounters system state (1, 2, 2). Owing to high-priority packets arriving later, the system state subsequently shifts to the right direction represented by states $(i, 2, 2)$ with $i = 2, \cdots, S_1 - 1$. Further arrivals of high-priority packets must be rejected. As the low-priority packet in service at arrival time s leaves the queueing system, a service cycle of high-priority packets starts. Of course, all high-priority packets arriving during this cycle are served, too. The possible states of this cycle are $(i, 1, 1)$ with $i = 1, \cdots, S_1$. Thereafter, the next low-priority packet is served, so that the next state of the flow process becomes (0, 1, 2). Should high-priority packets arrive during its service time, they are also served before the test packet. Finally, the test packet gets its turn which is represented by state (0, 0, 2).

Applying the rules given in Section II-B, the following differential equations are obtained for the lowest row of states in the flow-process diagram for a low-priority test packet,

$$\frac{d}{ds} f_{002}(s, t) = -\mu_2(t) \cdot f_{002}(s, t),$$

$$\frac{d}{ds} f_{i,01}(s, t) = -[\mu_1 + \lambda_1(s)] \cdot f_{i,01}(s, t) + \lambda_1(s) \cdot f_{i+1,01}(s, t) + \mu_1 \cdot f_{i-1,01}(s, t), \quad i = 1, \cdots, S_1 - 1, \text{ with index } 001 \equiv 002,$$

$$\frac{d}{ds} f_{S_1,01}(s, t) = -\mu_1 \cdot f_{S_1,01}(s, t) + \mu_1 \cdot f_{S_1-1,01}(s, t). \tag{21}$$

Similarly to the system-state process, the full set of differential equations is characterized by 20 types of equations. They are specified by the same states as used for the system-state process. All equation types belonging to a hatched state are set to zero.

B. Performance Measures

In order to evaluate the FG-BG congestion-control mechanism, the following performance measures are defined. If not explicitly specified, the range of valid values for i and j follows from the structure of the system-state diagram shown in Fig. 5 or the flow-process diagram defined by Fig. 6.

- *Loss probability for high-priority packets*, given by the sum of all state probabilities bounding the system-state diagram by condition 1 and at the right side,

$$B_1(t) = \sum_{\text{cond.1}} [P_{ij1}(t) + P_{ij2}(t)] + \sum_{j=1}^{S_2} [P_{S_1,j-1,1}(t) + P_{S_1-1,j,2}(t)]. \tag{22}$$

- *Loss probability for low-priority packets*, given by the sum of all state probabilities bounding the system-state diagram in the upward direction,

$$B_2(t) = \sum_{\text{cond.1}} [P_{ij1}(t) + P_{ij2}(t)] + \sum_{\text{cond.2}} P_{ij1}(t) + \sum_{\text{cond.3}} P_{ij2}(t), \tag{23}$$

where with $S = S_1 + S_2$ we get,

Cond. 1: $i + j = S$ and $j \leq S - L_1$,

Cond. 2: $L_1 > 0$ and $j = S - L_1$ and $1 \leq i \leq S - j$,

Cond. 3: $L_1 > 0$ and $j = S - L_1$ and $0 \leq i \leq S - j$.

- *Mean system occupancy for high-priority packets*, given by the arithmetic mean of having i high-priority packets in the system weighted by the corresponding state probabilities,

$$E[X_1(t)] = \sum_i \sum_j i \cdot [P_{ij1}(t) + P_{ij2}(t)]. \tag{24}$$

- *Mean system occupancy for low-priority packets*, given by the arithmetic mean of having j low-priority packets in the system weighted by the corresponding state probabilities,

$$E[X_2(t)] = \sum_i \sum_j j \cdot [P_{ij1}(t) + P_{ij2}(t)]. \tag{25}$$

- *Throughput of high-priority packets*, given by the probability that high-priority packets are present multiplied by the corresponding service rate,

$$T_1(t) = \mu_1 \cdot \sum_i \sum_j P_{ij1}(t). \tag{26}$$

- *Throughput of low-priority packets*, given by the probability that low-priority packets are present multiplied by the corresponding service rate,

$$T_2(t) = \mu_2 \cdot \sum_i \sum_j P_{ij2}(t). \tag{27}$$

- *Mean flow-time for an accepted high-priority test packet arriving at time s*, given by the sum of
  - — the mean service time of the test packet,
  - — the mean residual service time of a low-priority packet, provided a low-priority packet is in service at arrival time s; the probability for this case is given by the sum of the probabilities of all states $(i, j, 2)$, which does not cause a rejection of the test packet. Because of the Markovian condition, the residual service time is also negative exponentially distributed, so that its mean is identical to the mean service time h_2 itself. Since only accepted high-priority packets are taken into account, we must correct this term by the loss probability $B_1(s)$,

— the total mean service time of all high-priority packets encountered at the arrival of the test packet. The mean system occupancy of high-priority packets is obtained by the expectation of all states, which do not cause a rejection of the test packet. This term is again corrected by the loss probability $B_1(s)$, to obtain the conditional mean flow-time.

$$t_{F1}(s) = h_1 + \frac{1}{1 - B_1(s)} \cdot \left[h_2 \cdot \sum_i \sum_j P_{ij2}(s) + h_1 \cdot \sum_i \sum_j i \cdot \{P_{ij1}(s) + P_{ij2}(s)\} \right],$$

$i, j \in \tilde{M}$ with $\tilde{M}$: all states which can be encountered by the test packet. (28)

• *Mean flow-time for an accepted low-priority test packet arriving at time s*, given by the numerical integration of the complementary flow-time distribution defined in (10); since the conditional mean flow-time is considered, this integration result must be corrected by the loss probability $B_2(s)$,

$$t_{F2}(s) = \frac{1}{1 - B_2(s)} \cdot \int_{\tau=s}^{\infty} F^c(s, \tau) \cdot d\tau. \quad (29)$$

C. Numerical Results and Discussion

With some numerical examples, the dynamic behavior of the FG-BG congestion-control mechanism is demonstrated. For the performance evaluation, a queueing system with foreground store $S_1 = 20$, and background store $S_2 = 0$, 10, 20, 30, and 40 is considered. The mean service time is equal for both priority classes and is identical to the time unit: $h_1 = 1/\mu_1 = 1$ and $h_2 = 1/\mu_2 = 1$. Thus, packets leave the system with the unified service rate $\mu = 1$. At the normalized time $t = 0$, the system is in steady state with arrival rates $\lambda_1(\infty) = 0.4$ for high-priority packets and $\lambda_2(\infty) = 0.2$ for low-priority packets. At time $t = 10$, the arrival rate of the low-priority traffic class is changed according to a rectangular function with value $\lambda_{2max} = 6.0$. First, a short overload peak of duration $T = 10$ is analyzed. In that case, the overload ends at time $t = 20$. Subsequently, the duration of the overload situation is extended to $T = 30$, so that the normal arrival rate is again offered at time $t = 40$.

a) Short Overload Peak of Low-Priority Packets: First, a short rectangular overload of duration $T = 10$ caused by the low-priority traffic class is dealt with. Moreover, no buffers have been reserved for high-priority packets, i.e., $L_1 = 0$.

Loss Probability: In Fig. 7, the loss probabilities $B_1(t)$ and $B_2(t)$ of both priority classes have been given. Since the overload peak is caused by low-priority packets, which are permitted to use the whole system capacity $S = S_1 + S_2$, high-priority packets are rejected whenever low-priority packets must be refused. Therefore, during the overload peak both loss probabilities are equal, i.e., $B_1(t) = B_2(t)$. Because of the sudden increase of the arrival rate and particularly, the high intensity of the overload peak, the curves show a very steep rise to their maximum value. The very long system recovery time after the overload peak is also typical. During this period, the system is of course very sensitive to further overloads. As can be concluded from the delayed rise of the loss-probability curves for increasing parameter S_2, a short overload peak of low-priority packets can be managed better when the capacity of the background store is chosen large. For $S = 40$, the system is not yet saturated.

Mean System Occupancy: Fig. 8 illustrates the rapid increase of the mean system occupancy $E[X_2(t)]$ for low-priority packets. Initially, the foreground and background stores are linearly filled with growing rate $\Lambda = \lambda_{2max} + \lambda_1(\infty) - \mu = 5.4$ packets per time unit. At the saturation points, the curves flatten and finally are limited by the corresponding capacity of the queueing system. With a background store of $S_2 = 40$ buffers, system saturation just starts. As the system becomes saturated, the mean system occupancy $E[X_1(t)]$ of high-priority packets decreases to a very small value. This happens first for $S_2 = 0$. The effect is essentially caused by the high arrival rate of low-priority packets. Then, during the overload, each buffer which becomes free is captured by both priority classes in proportion to their arrival rates. In addition, when occasionally a high-priority packet is accepted, it is served with priority. For a larger value of the background store, e.g., $S_2 = 40$, the decrease of $E[X_1(t)]$ is very small. After disappearance of the overload peak, the surplus of low-priority packets is worked off. Its initial unloading rate, which is near $M = \mu = 1$ for a completely saturated system (e.g., for $S_2 = 0$), soon decreases to $M = \mu - \lambda_1(\infty) - \lambda_2(\infty) = 0.4$ as the influence of the loss probabilities $B_1(t) = B_2(t)$ diminishes. Furthermore, the dashed curves for $E[X_2(t)]$ become sooner negative exponentially shaped as the parameter S_2 decreases.

b) Long Overload Peak of Low-Priority Packets: The rectangular overload peak with intensity $\lambda_{2max} = 6.0$ originated by the low-priority packets has now a duration $T = 30$.

Loss Probability: In Fig. 9, the curves of the loss probabilities $B_1(t)$ and $B_2(t)$ have been shown for a longer duration of the overload peak. They are used as reference curves for discussing the influence of buffer reservation for high-priority packets. This reference figure shows the case without reservation, i.e., $L_1 = 0$. As already discussed in Fig. 7, the loss probabilities for both priority classes are identical.

Fig. 10 illustrates that the loss probability $B_1(t)$ for high-priority packets can be improved significantly by buffer reservation. Particularly, if we dedicate $L_1 = 10$ buffers exclusively to high-priority packets, their loss probability will grow only up to $B_1(t) = 10^{-4}$. Because of this precaution, the number of buffers for low-priority packets is reduced, so that the corresponding dashed curves $B_2(t)$ rise

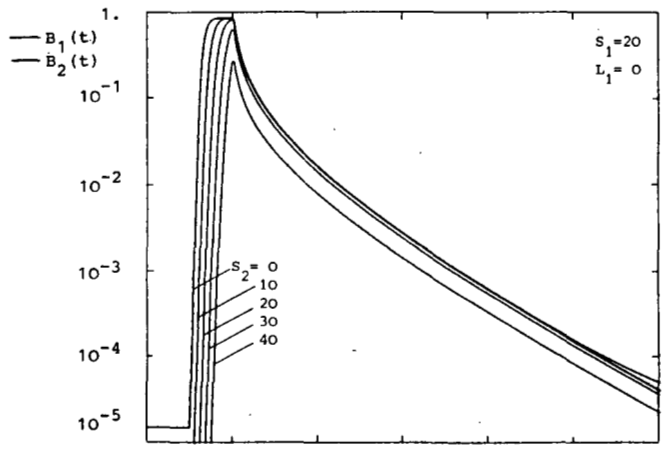

Fig. 7.

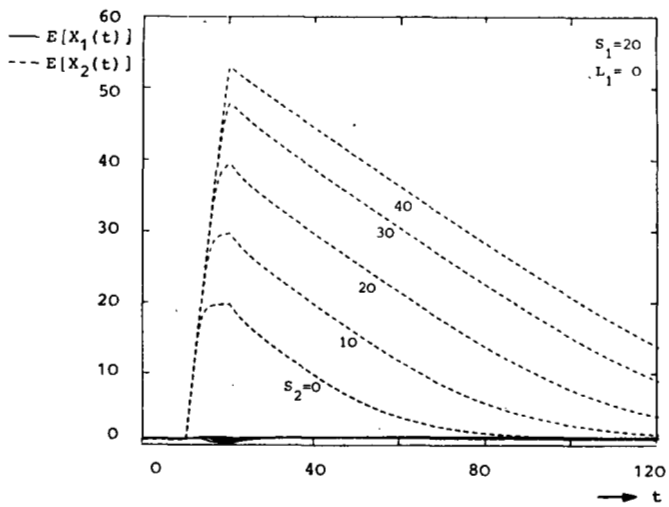

Fig. 7. Loss probabilities $B_1(t)$ and $B_2(t)$.
Fig. 8. Mean system occupancies $E[X_1(t)]$ and $E[X_2(t)]$.
Short overload peak of low-priority traffic. Duration $T = 10$, starting at $t = 10$. Traffic parameters: $\lambda_1(\infty) = 0.4$, $\lambda_2(\infty) = 0.2$, $\lambda_{2max} = 6.0$, $h_1 = h_2 = 1$.
Loss probabilities $B_1(t)$ and $B_2(t)$.

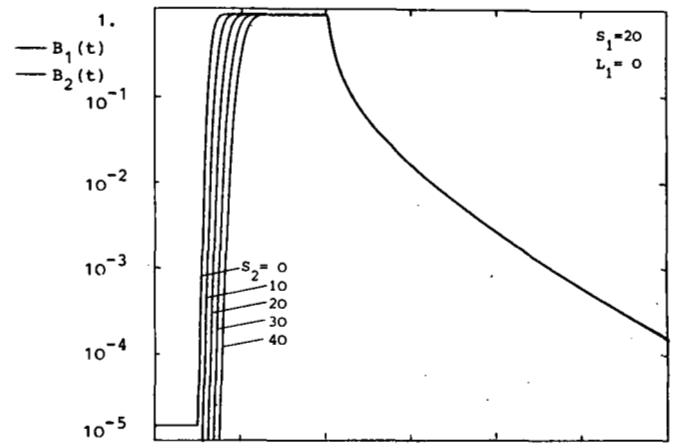

Fig. 9.

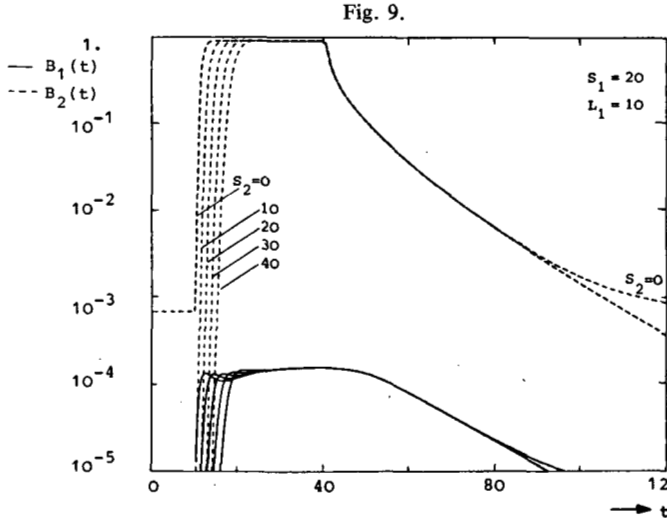

Fig. 9. No reservation for high-priority packets, $L_1 = 0$.
Fig. 10. Reservation for high-priority packets, $L_1 = 10$.
Overload peak of low-priority traffic. Duration $T = 30$, starting at $t = 10$. Traffic parameters: $\lambda_1(\infty) = 0.4$, $\lambda_2(\infty) = 0.2$, $\lambda_{2max} = 6.0$, $h_1 = h_2 = 1$.

earlier and need a longer time to return to their steady-state value.

Throughput: In Fig. 11, the dynamic behavior of the throughputs $T_1(t)$ and $T_2(t)$ is shown. The course of the curves looks rather complex. Therefore, the discussion will be restricted here to the major effects. For further details, we refer to [5]. Let us consider the case without background store, thus, $S_2 = 0$. For this system configuration, it can be observed that the throughput $T_2(t)$ of the low-priority packets, which cause the overload peak, rapidly becomes very high. At the same time, the throughput $T_1(t)$ of the high-priority packets is forced to decrease significantly. Then, during system saturation, a buffer becoming free is captured by both traffic classes in proportion to their arrival rates. A larger background store can only delay this degradation in throughput. However, as soon as the overload pressure disappears, the full steady-state throughput $T_1(\infty) = 0.4$ is again obtained. The time to approach the steady-state throughput $T_2(\infty) = 0.2$ depends on the surplus of low-priority packets which have to be worked off. This is, of course, related to the size of the background store. Moreover, one should note that for $S_2 = 40$, the total throughput has its maximum value $T_1(t) + T_2(t) = 1$ for quite a time after disappearance of the overload peak.

Fig. 12 demonstrates that, apart from a small degradation at the beginning, the throughput $T_1(t)$ can be maintained, if a buffer reservation $L_1 = 10$ for high-priority packets is made. Because of this, the throughput of low-priority packets grows only up to a maximum value $T_2(t) = 0.6$.

Mean Flow Time: Fig. 13 shows the mean flow times $t_{F1}(s)$ and $t_{F2}(s)$ for accepted test packets of both priority classes arriving at time s. Since we have an overload peak of low-priority packets, the mean flow time $t_{F1}(s)$ is only influenced by the increased probability of being delayed by the residual service time of a low-priority packet. Therefore, the curves for $t_{F1}(s)$ lie slightly above their steady-state values. Since the arrival rate $\lambda_1(\infty)$ of the high-priority packets remains constant, the curves for $t_{F2}(s)$ show much similarity to the curves for the mean

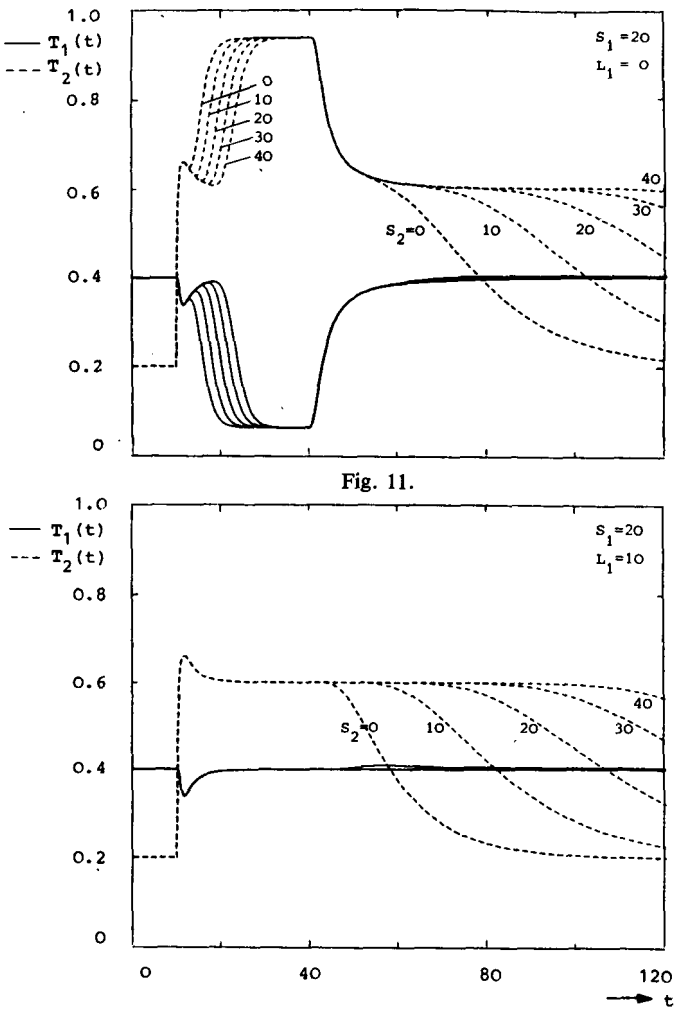

Fig. 11.

Throughputs $T_1(t)$ and $T_2(t)$.

Fig. 11. No reservation for high-priority packets, $L_1 = 0$.
Fig. 12. Reservation for high-priority packets: $L_1 = 10$.
Overload peak of low-priority traffic. Duration $T = 30$, starting at $t = 10$. Traffic parameters: $\lambda_1(\infty) = 0.4$, $\lambda_2(\infty) = 0.2$, $\lambda_{2max} = 6.0$, $h_1 = h_2 = 1$.

system occupancy $E[X_2(t)]$ for low-priority packets discussed in Fig. 8 for a short overload peak. The mean flow time $t_{F2}(s)$ can also be roughly calculated from the system state encountered at its arrival time s. However, a higher value results because the approximate calculation does not take into account the initial high value of the loss probability $B_1(t)$ immediately after the overload peak.

VI. Two-Level Global Congestion-Control Mechanism

As a third example, the transient behavior of a two-level global congestion-control mechanism is considered [2]. The storage occupancy of each network node is continuously monitored, and when overload threatens, messages are sent to the traffic sources or to the network access nodes to stop the traffic directly at the network boundary. Thus, the main objective of this congestion-control mechanism is to keep the excessive traffic out of the network, so that packet rejections at the network nodes

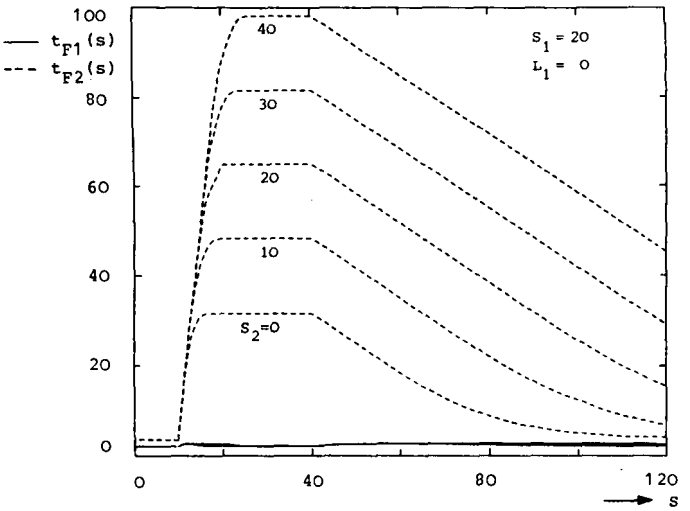

Fig. 13. Mean flow times $t_{F1}(s)$ and $t_{F2}(s)$ of accepted packets as a function of their arrival time s. Overload peak of low-priority traffic. Duration $T = 30$, starting at $t = 10$. Traffic parameters: $\lambda_1(\infty) = 0.4$, $\lambda_2(\infty) = 0.2$, $\lambda_{2max} = 6.0$, $h_1 = h_2 = 1$.

owing to lack of nodal storage can be largely avoided. Because now fewer packet retransmissions are necessary, the transmission capacity is used more efficiently. Since exchanging messages affects the control speed, this global congestion-control mechanism only tries to prevent the start or the propagation of overload situations. It must be assisted by a much faster local mechanism like the FG-BG congestion-control mechanism for which both detecting and executing functions are available at the same network node. The stochastic control delay caused by the geographic distance between the network node considered, somewhere in the packet-switching network, and the traffic sources or the network access nodes consists of two time components. First, the time to deliver the message. Because of the priority scheduling for network control packets, which contain these messages, its contribution to the delay is rather small. But the major part comes from the time elapsed until the influence of the traffic volume already within the network has disappeared, or until the traffic again appears at the network node.

Fig. 14 depicts the queueing model describing the fundamental effects of the two-level global congestion-control mechanism. Basically, it consists of a single-server queueing system with finite capacity S and on/off source control. The queueing system represents the storage capacity of a network node in which packets leave with a mean rate μ. Its occupancy is continuously monitored. An upward crossing of the high-occupancy level S_1 causes a message to be sent to cut off the arrival rate $\lambda(t)$. After a stochastic delay D with mean $d = 1/\nu$, the control becomes effective. The system can now recover, and a downward crossing of the low-occupancy level S_2 results in sending a message to switch on the arrival rate. Again, the control action becomes effective after the stochastic delay D. A two-level instead of a single-level control mechanism is necessary to prevent frequent generation of messages owing to statistical fluctuations around one fixed

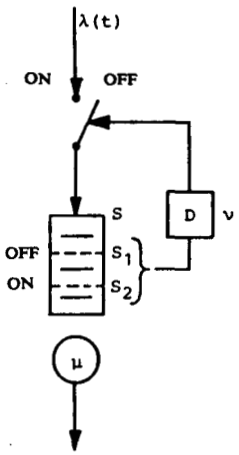

Fig. 14. Queueing model for a two-level global congestion-control mechanism.

occupancy level. With the switch and the stochastic delay, it is possible to model the arrival process of the network node such that the influence of the packet-switching network is taken into account approximately. The interarrival times, the service times, and the delay times are assumed to be negative exponentially distributed. Thus, a Markovian queueing system is considered. Finally, we make the assumption that each packet needs exactly one buffer.

A. Analysis

Fig. 15 illustrates the structure of the system-state diagram for the two-level global congestion-control mechanism. This structure is based on the assumption that with a neglecting probability more than one message is simultaneously underway to the switch. This can be achieved by a proper choice of the system parameters [3].

Each state is denoted by the triplet (i, j, k), where

- i Number of packets in the queueing system, $i = 0, 1, \cdots, S$.
- j Position of the switch, $j = 0$: off, $j = 1$: on.
- k Number of messages currently underway, $k = 0, 1, 2$.

We start with state (0, 1, 0) representing an empty system, the switch in the on-position, and no message underway. During normal load, the states of the left-hand column will be visited according to the arrival and service processes. For low loads, its behavior corresponds to an $M/M/1$ queueing system. When, for instance, because of an overload peak, the system occupancy reaches the high-occupancy level S_1, the state column is changed and the state process continues over the right-hand column states. This means, the switch remains in the on-position until the switch-off message arrives. In case the system is filled up faster with packets than the switch-off message becomes effective, packets are lost. However, if the system occupancy decreases down to the low-occupancy level S_2 before the switch-off message has reached the switch, a

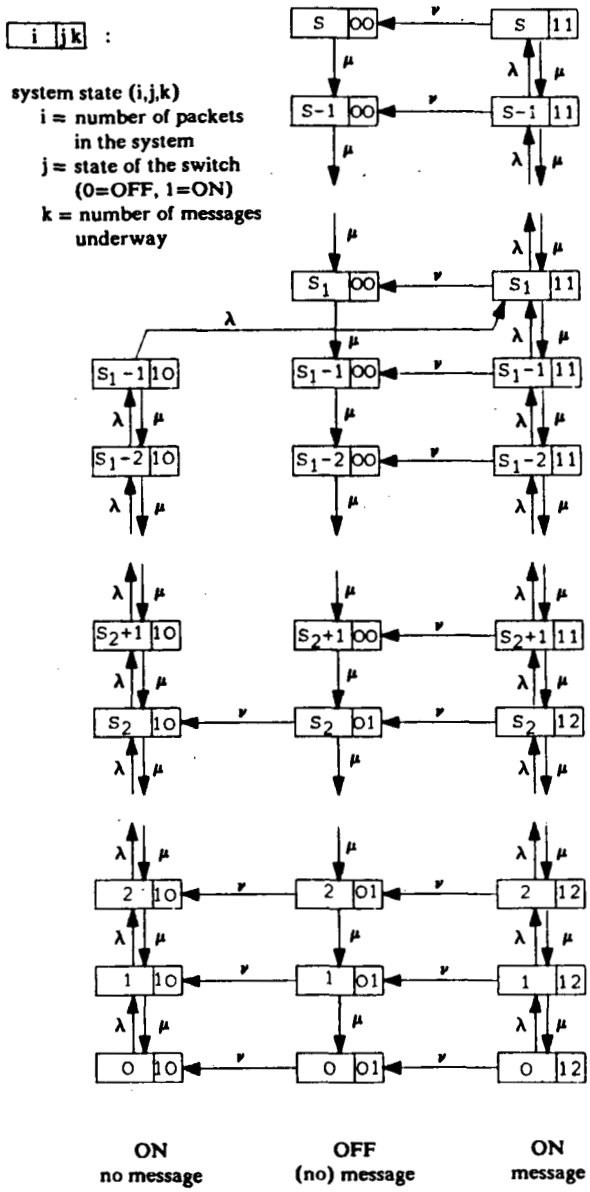

Fig. 15. System-state diagram for a two-level global congestion-control mechanism.

second message (switch-on message) is sent away, too. As discussed previously, this happens with very low probability. From all states of this right-hand column, a transition to the middle column occurs with rate ν, the rate at which messages arrive at the switch. Depending on the position of this transition, the new state is $(i, 0, 0)$ or $(i, 0, 1)$ which means that the switch is set in the off-position, and either no or the next switch-on message is already underway. In the middle column, the system occupancy can only decrease, and when a switch-on message finally switches the arrival process on again, the state process continues to visit the states of the left-hand column.

Having defined the structure of the system-state diagram, the differential equations for the state probabilities are immediately obtained by inspection.

B. Performance Measures

From the time-dependent state probabilities, the performance measures like mean system occupancy $E[X(t)]$, the loss probability at the switch $B_{SW}(t)$, and the loss probability at the queueing system $B_{SYS}(t)$ can be defined as follows.

- *Mean system occupancy*, given by a weighted sum of all state probabilities,

$$E[X(t)] = \sum_{i=0}^{S_1-1} i.P_{i10}(t) + \sum_{i=0}^{S_2} i.P_{i01}(t) + \sum_{i=S_2+1}^{S} i.P_{i00}(t) + \sum_{i=0}^{S_2} i.P_{i12}(t) + \sum_{i=S_2+1}^{S} i.P_{i11}(t). \quad (30)$$

- *Loss probability at the switch*, given by the sum of all state probabilities for which the switch is in the off-position,

$$B_{SW}(t) = \sum_{i=0}^{S_2} P_{i01}(t) + \sum_{i=S_2+1}^{S} P_{i00}(t). \quad (31)$$

- *Loss probability at the queueing system*, given by the state probability that the switch is in the on-position and that there are already S packets in the system,

$$B_{SYS}(t) = P_{S11}(t). \quad (32)$$

C. Numerical Results and Discussion

In the following, the system reaction to a long overload peak is considered. The queueing system analyzed has a capacity $S = 40$, a high-occupancy level $S_1 = 32$, and low-occupancy levels $S_2 = 4, 8, 12, 16, 20, 24, 28$. The mean service time $h = 1/\mu = 1$ is also the time unit. Furthermore, a mean control delay $d = 1$ has been chosen. At the normalized time $t = 0$, the system is in steady-state with arrival rate $\lambda(\infty) = 0.5$. From $t = 10$ to $t = 100$, the arrival rate has been changed according to a rectangular function with value $\lambda_{max} = 6.0$.

Mean System Occupancy: Fig. 16 shows the mean system occupancy $E[X(t)]$ with the low-occupancy level S_2 as a parameter. Owing to the sudden increase of the arrival rate $\lambda(t)$, the mean system occupancy $E[X(t)]$ increases with approximately $\Lambda = \lambda_{max} - \mu = 5$ packets per time unit. As the high-occupancy level $S_1 = 32$ is reached, a message to cut off the arrival stream is sent to the switch. During the mean delay $d = 1$ before the message becomes effective, the mean system occupancy continues to increase. Its upper value $E[X(t)] = 34.5$ is reached at time $t = 18.3$. As soon as the switch is in the off-position, the mean system occupancy can decrease with rate $\mu = 1$ packet per time unit. When a single realization of the system-state process reaches the low-occupancy level S_2, a switch-on message is sent away. However, taking $S_2 = 4$ as an example, it can be noted that the minimum value for the mean system occupancy remains much higher. This is a result of the interplay between the stochastic processes involved for arrival, service, and control delay. Owing to their time-dependent relationship, the course of the curve $E[X(t)]$ can no longer be predicted by simple mean value reasoning. This example demonstrates the importance of transient queueing analysis. Of course, the reason for this can also be explained. Then, since the stochastic delay D obeys a negative exponential distribution, its value is very often shorter than its mean d. Because of this, the switch frequently returns soon in the on-position, and the queueing system is filled up again with the high arrival rate $\lambda_{max} = 6.0$. Whereas on the other hand, for the cases of an occasional longer delay, the queueing system is only unloaded with the service rate $\mu = 1$. Furthermore, the curves for the mean system occupancy $E[X(t)]$ show that on–off switching results in damped oscillations. The amplitude of the oscillations increases with decreasing value of S_2. This parameter also determines the final value of $E[X(t)]$ during the overload peak and the speed to reach this value. It is important to note that for some values of

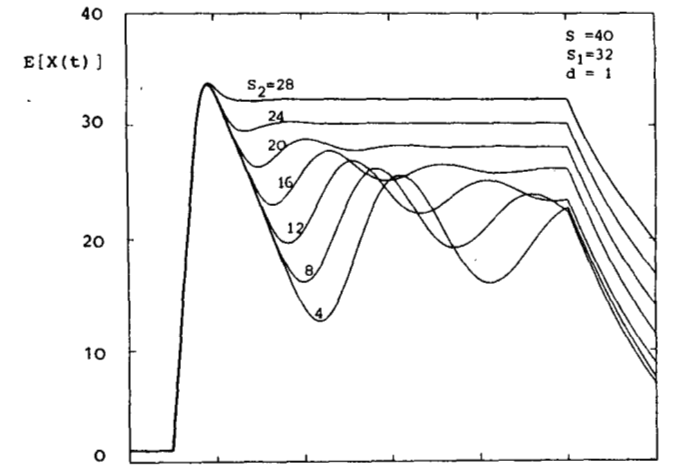

Fig. 16.

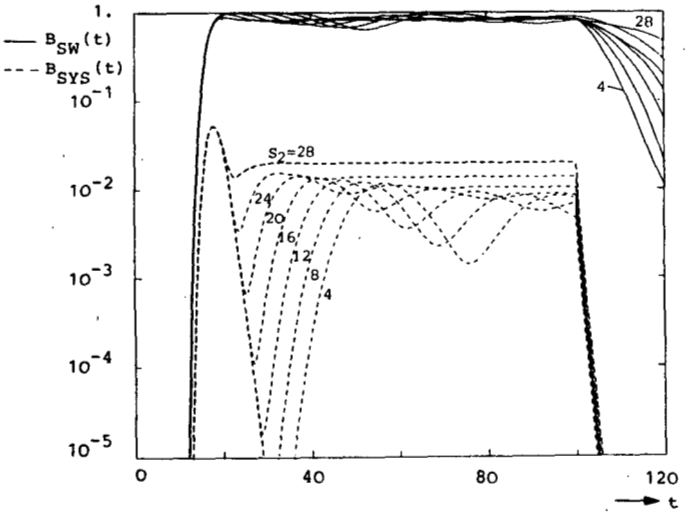

Fig. 16. Mean system occupancy $E[X(t)]$.
Fig. 17. Loss probabilities $B_{SW}(t)$ at the switch and $B_{SYS}(t)$ at the queueing system itself.
Overload peak of duration $T = 90$ starting at $t = 10$.
Traffic parameters: $\lambda(\infty) = 0.5$, $\lambda_{max} = 6.0$, $h = 1$.

the parameter S_2, $E[X(t)]$ does not become stationary during the overload peak.

Loss Probabilities: Fig. 17 gives the corresponding curves for the loss probabilities: $B_{SW}(t)$ at the switch and $B_{SYS}(t)$ at the queueing system itself. Similar to the mean system occupancy, the loss probability $B_{SYS}(t)$ reaches its maximal value immediately after the arrival rate has been cut off for the first time. Then, a damped oscillation follows for which the amplitude depends on the value of S_2. The curves demonstrate clearly the effectiveness of the on–off global congestion-control mechanism for a small mean control delay: a very high percentage of arriving packets is immediately rejected at the switch, thus preventing the occurrence of blind traffic within the network caused by packet transmissions. In contrast, the rejection rate at the network node, given by $B_{SYS}(t)$, is two orders of magnitude lower. Also the curves for $B_{SW}(t)$ show oscillations, but with a much smaller amplitude. For a long mean control delay, however, the control mechanism performs much poorer [2].

VII. Conclusion

It has been shown that the transient behavior of a wide variety of Markovian queueing models can be evaluated in a uniform and straightforward manner. This has been realized by a direct numerical approach of the set of coupled differential equations describing the system-state or flow process. Performance measures like loss probability, mean system occupancy, and throughput, have been calculated from time-dependent state probabilities. Moreover, following the concept of a flow process, the method also provides time-dependent flow-time measures like mean and distribution. The technique and the explanatory power of this kind of transient queueing analysis have been illustrated by three models: the common-store queueing system showing the priority deadlock, the foreground-background congestion-control mechanism, and a two-level global congestion-control mechanism.

Acknowledgment

The author would like to thank P. J. Kuehn for fruitful discussions and his continued interest in this study.

References

[1] H. R. van As, "Congestion control in packet switching networks by a dynamic foreground-background storage strategy," *2nd Int. Symp. Performance of Comput. Commun. Syst.*, Zurich, Switzerland, 1984, pp. 433–448; Amsterdam, The Netherlands: North-Holland, Eds., W. Bux and H. Rudin.

[2] H. R. van As, "Transient queueing analysis of a two-level global congestion control mechanism," *Int. Seminar Comput. Networking and Performance Evaluation*, Tokyo, Japan, 1985, pp. 423–435; Amsterdam, The Netherlands: North-Holland, Eds., T. Hasegawa, H. Takagi, and Y. Takahashi.

[3] H. R. van As and M. H. van Hoorn, "A queueing system with a remote two-level control," to be published.

[4] H. R. van As, "Transient analysis of a common-store queueing system," to be published.

[5] ——, "Transient queueing analysis of the foreground-background congestion-control mechanism," to be published.

[6] W. Feller, *An Introduction to Probability Theory and its Applications*. Vol. 1. New York: Wiley, 1968.

[7] M. Gerla and L. Kleinrock, "Flow control: A comparative survey," *IEEE Trans. Commun.*, vol. COM-28, pp. 553–574, 1980.

[8] P. J. Kuehn, "On the calculation of waiting times in switching and computer systems," 15th Rep. Studies in Congestion Theory, Institute of Switching and Data Technics, Univ. of Stuttgart, Stuttgart, Germany, 1972.

[9] ——, Analysis of busy periods and response times in queueing networks by the method of first passage times, in *Performance 83*. Amsterdam, The Netherlands: North-Holland, 1983, pp. 437–455.

[10] M. J. Maron, *Numerical Analysis: A Practical Approach*. New York: Macmillan, London, England: Collier-Macmillan, 1982.

[11] L. Pouzin, "Methods, tools, and observations on flow control in packet-switched data networks," *IEEE Trans. Commun.*, vol. COM-29, pp. 413–426, 1981.

[12] M. Reiser, "Performance evaluation of data communication systems," *Proc. IEEE*, vol. 70, pp. 171–196, 1982.

[13] J. A. White, J. W. Schmidt, and G. K. Bennett, *Analysis of Queueing Systems*. New York: Academic Press, 1975.

Harmen R. van As (M'79) was born in Rotterdam, The Netherlands, in 1943. He received the Dipl.-Ing. degree in electrical engineering in 1968, and a supplementary degree in operations research in 1970, from the Swiss Federal Institute of Technology (ETH), Zurich, Switzerland, and the Dr.-Ing. degree in electrical engineering in 1984 from the University of Siegen, Siegen, West Germany.

From 1970 to 1979, he was with Standard Telephon und Radio AG (ITT), Zurich, where he mainly worked in the fields of teletraffic theory, stochastic simulations, and computer-aided manufacturing. From 1979 to 1984, he worked as a Research Associate at the Department of Communications Switching and Transmission, University of Siegen. During this period, his research activities were in queueing theory applied to communication systems, advanced simulation techniques, and congestion control for packet-switching networks. In 1984, Dr. van As joined the IBM Zurich Research Laboratory, Rüschlikon, Switzerland. His current research is in performance evaluation and architecture of communication systems and networks.