RECAL—A New Efficient Algorithm for the Exact Analysis of Multiple-Chain Closed Queuing Networks

A. E. CONWAY AND N. D. GEORGANAS

University of Ottawa, Ottawa, Canada

Abstract. RECAL, a *Re*cursion by *C*hain *Al*gorithm for computing the mean performance measures of product-form multiple-chain closed queuing networks, is presented. It is based on a new recursive expression that relates the normalization constant of a network with r closed routing chains to those of a set of networks having $(r - 1)$ chains. It relies on the artifice of breaking down each chain into constituent subchains that each have a population of one. The time and space requirements of the algorithm are shown to be polynomial in the number of chains. When the network contains many routing chains, the proposed algorithm is substantially more efficient than the convolution or mean value analysis algorithms. The algorithm, therefore, extends the range of queuing networks that can be analyzed efficiently by exact means.

Categories and Subject Descriptors: C.2.4 [**Computer-Communication Networks**]: Distributed Systems—*network operating systems*; D.4.4 [**Operating Systems**]: Communications Management—*network communication*; D.4.8 [**Operating Systems**]: Performance—*modeling and prediction; queuing theory; stochastic analysis*

General Terms: Algorithms, Performance, Theory, Verification

Additional Key Words and Phrases: Closed queuing networks, dynamic scaling, exact analysis, multiple chains, normalization constants, product-form solution, recursion by chain

1. *Introduction*

Multiple-chain closed queuing networks, which have a product-form [1] state probability distribution, are widely used models of, for example, computer systems [11, 19] and computer communication networks [10]. The principal computational difficulty associated with these networks is that a simple closed-form expression for the normalization constant of the distribution is not known. In general, a direct determination of the normalization constant by straightforward summation is computationally intractable. Hence, the network performance measures cannot be computed by simply summing probabilities over an appropriate set of states. As a result, much effort has been directed to the development of efficient methods for analyzing multiple-chain networks.

This research was supported in part by the Natural Sciences and Engineering Research Council of Canada (NSERC) under grant A8450, by a University of Ottawa research scholarship, and by an NSERC postgraduate scholarship.

Authors' present addresses: A. E. Conway, Department of Electrical Engineering, McGill University, 3480 University Street, Montreal, P.Q., Canada H3A 2A7; N. D. Georganas, Department of Electrical Engineering, University of Ottawa, 770 King Edward, Ottawa, Ontario, Canada K1N6N5.

There now exist several efficient algorithms. The best known ones are the convolution algorithm and mean value analysis (MVA). The convolution algorithm was first proposed by Buzen [3] for single-chain networks and subsequently extended by Reiser and Kobayashi [17] for multiple chains. The convolution algorithm is an efficient means of obtaining the normalization constant. Most performance measures of interest, such as mean queue lengths, server utilizations, mean waiting times, and nodal throughputs, are readily computed using the normalization constant and certain intermediate results obtained in the convolution algorithm. This is termed the *normalization constant approach.* Another efficient algorithm is MVA [18], which is based on the so-called Arrival Theorem [12]. In this algorithm the performance measures are obtained directly, without recourse to normalization constants, but only first moment information is obtained. A major difficulty with both of these algorithms is that the time and space requirements increase exponentially with the number of chains. Other existing algorithms, such as LBANC and CCNC [5, 11, 19], also share this difficulty. As a result, the exact determination of performance measures has, in general, been limited to networks with a relatively small number of chains.

The analysis of queuing networks with many chains requires the use of specialized exact algorithms, analytical approximation techniques, or heuristic methods. The tree convolution algorithm [9] can obtain exact results for networks in which there are many chains by taking advantage of sparsity or locality properties of the routing. It is particularly useful for the analysis of window flow-controlled computer communication networks [10]. A powerful analytical approximation technique for networks of a large size is discussed in [14] and [15]. Heuristic methods are presented in [4], [11], [16], and [18].

In this paper we present RECAL, a *Re*cursion by *C*hain *Al*gorithm for the exact analysis of product-form multiple-chain closed queuing networks. It is based on a new recursive expression that relates the normalization constant of a network with r closed routing chains to those of a set of networks having $(r-1)$ chains. RECAL may therefore be classified as a new normalization constant approach. For obtaining the normalization constant of multiple-chain closed queuing networks, the new recursion has a time and space growth that is different from that of the recursions used in the convolution and MVA algorithms. We introduce the artifice of breaking down each chain into subchains that each have a population of one. The main idea of employing this artifice is the following. With each chain made to have a population of one, the recursion by chain expression is greatly simplified. The artifice alters the state space and hence the normalization constant of the network under consideration but leaves the performance characteristics (i.e., nodal throughputs, server utilizations, waiting times, queue lengths) unchanged. As a result, this simplified version of the recursion, which yields the normalization constant for the artificial network, may nevertheless be used to carry out an exact analysis of the original network.

The time and space requirements for obtaining the mean performance measures of all the routing chains in a network using RECAL is polynomial in the number of chains. When there are many routing chains in the network, RECAL is substantially more efficient than the hitherto adopted methods of analyzing queuing networks. In other situations, such as when the number of routing chains is small, it is, in general, less efficient. RECAL, therefore, extends the range of queuing networks that can be analyzed efficiently by exact means.

RECAL is general in the sense that it may accommodate the situation in which all service centers are overlapped by all chains. Hence, we do not exploit sparsity

or locality properties of the routing. Rather, the computational savings can be attributed to the definition of the recursion itself. When there is sparsity or locality, however, the computational requirements dictated by the mathematical definition of the recursion are indeed diminished, but in this paper we only concern ourselves with the algorithm in its general form.

The paper is organized as follows. In the following section we define the class of queuing networks to be considered. For simplicity, we initially assume constant-speed servers at the service centers. In Section 3 we present certain new results, which form the basis for the proposed algorithm. The algorithm is then developed in Section 4. The computational complexity of the algorithm is derived in Section 5. In Section 6 we compare this complexity with that of the convolution and MVA algorithms and demonstrate the situations in which the proposed algorithm is useful. In Section 7 we extend the algorithm to accommodate the situation in which there are state-dependent servers. In Section 8, we discuss how one may handle mixed networks [1] with constant speed or limited queue-dependent servers [11, sect. 3.6.1] and networks in which there is customer-class switching [1]. Finally, in Section 9 we present a simple dynamic scaling procedure, which can be incorporated easily within the framework of RECAL, to reduce the possibility of exceeding the floating-point range of a machine.

2. *The Queuing Network*

We are concerned here with the analysis of multiple-chain closed queuing networks of the product-form type, which have been considered in [1]. There are N service centers and R closed routing chains. We are only concerned with cases in which $R > 1$. The service discipline at the centers may be FCFS, LCFSPR, PS, or IS, as in [1]. The routing of customers belonging to chain r is specified by a transition probability matrix with a left eigenvector (visit ratio vector), associated with eigenvalue 1, given by $(e_{1r}, \ldots, e_{Nr})$. The mean service requirement for a chain r customer at center i is t_{ir}. We initially assume constant-speed servers. At centers with a FCFS discipline it is required that the service time distribution be exponential with mean $t_{ir} = t_i$ for $1 \le r \le R$. The population of chain r is K_r. We denote the queuing network described above as $\mathcal{N}$.

An aggregate system state of $\mathcal{N}$ is $\mathbf{n}^{(R)}$ where $\mathbf{n}^{(R)} = (\mathbf{n}_1^{(R)}, \ldots, \mathbf{n}_N^{(R)})$, $\mathbf{n}_i^{(R)} = (n_{i1}, \ldots, n_{iR})$ and n_{ir} is the number of customers at center i belonging to chain r. The space of feasible aggregate system states for $\mathcal{N}$ is $\mathcal{S}^{(R)}$, where

$$\mathcal{S}^{(R)} = \left\{ \mathbf{n}^{(R)} \mid n_{ir} \ge 0 \text{ for } 1 \le i \le N,\ 1 \le r \le R;\ \sum_{i=1}^{N} n_{ir} = K_r \text{ for } 1 \le r \le R \right\}.$$

The marginal state probability distribution for $\mathbf{n}^{(R)} \in \mathcal{S}^{(R)}$ is [1]

$$\Pr(\mathbf{n}^{(R)}) = G^{-1} \prod_{i=1}^{N} f_i(\mathbf{n}_i^{(R)}), \tag{2.1}$$

where

$$f_i(\mathbf{n}_i^{(R)}) = \begin{cases} n_i^{(R)}! \prod_{r=1}^{R} \dfrac{w_{ir}^{n_{ir}}}{n_{ir}!} & \text{if center } i \text{ is FCFS, LCFSPR, or PS,} \\ \prod_{r=1}^{R} \dfrac{w_{ir}^{n_{ir}}}{n_{ir}!} & \text{if center } i \text{ is IS,} \end{cases}$$

$$n_i^{(R)} = \sum_{r=1}^{R} n_{ir}, \qquad w_{ir} = t_{ir} e_{ir},$$

and the normalization constant is, by definition,

$$G = \sum_{\mathbf{n}^{(R)} \in \mathcal{S}^{(R)}} \prod_{i=1}^{N} f_i(\mathbf{n}_i^{(R)}).$$

Since the cardinality of the set $\mathcal{S}^{(R)}$ is generally large, the computation of G by direct summation is not feasible. In the following section we develop a new approach toward computing G efficiently.

3. *Preliminary Results*

In this section we present certain new relationships among the normalization constants of multiple-chain closed queuing networks and new expressions for the mean performance measures in terms of these normalization constants. They provide the mathematical basis for the algorithm developed in the succeeding section.

Consider a multiple-chain closed queuing network, like $\mathcal{N}$, but with state-dependent servers at those service centers that do not have an IS service discipline. Let the service rate function for center i be $u_i(n)$; that is, when there are n customers at center i the server accomplishes work at the rate $u_i(n)$. Let $u_i(n) = n/(n + c_i)$, where c_i is a nonnegative integer. The physical interpretation is that there are c_i customers at node i that cycle around continuously, at node i, and consume, on average, a fraction $1 - u_i(n)$ of the server capacity when the population of node i is n. Denote the normalization constant for such a network with state-dependent servers as $G_R(\mathbf{c})$ where $\mathbf{c} = (c_1, \ldots, c_N)$. An explicit expression for $G_R(\mathbf{c})$ is given in Appendix A. Clearly $G_R(\mathbf{0}) = G$ since, if $\mathbf{c} = \mathbf{0}$, then $u_i(n) = 1$. We have the following results.

THEOREM 1. *$G_R(\mathbf{0})$ is given recursively by*

$$G_r(\mathbf{v}_r) = \sum_{\mathbf{l} \in \mathcal{L}_r} g_r(\mathbf{v}_r, \mathbf{l}) G_{r-1}(\mathbf{v}_r + \mathbf{l}), \qquad \text{for} \quad 1 \le r \le R, \quad \mathbf{v}_r \in \mathcal{J}_r,$$

where

$$\mathbf{v}_r = (v_{1r}, \ldots, v_{Nr}),$$

$$\mathbf{l} = (l_1, \ldots, l_N),$$

$$\mathcal{J}_r = \begin{cases} \left\{ \mathbf{v}_r \mid v_{ir} \ge 0 \text{ for } 1 \le i \le N; \sum_{i=1}^{N} v_{ir} = \sum_{s=r+1}^{R} K_s \right\} & \text{if} \quad 0 \le r \le R-1, \\ \{\mathbf{0}\} & \text{if} \quad r = R, \end{cases}$$

$$\mathcal{L}_r = \left\{ \mathbf{l} \mid l_i \ge 0 \text{ for } 1 \le i \le N; \sum_{i=1}^{N} l_i = K_r \right\},$$

$$g_r(\mathbf{v}_r, \mathbf{l}) = \prod_{i=1}^{N} h_{ir}(\mathbf{v}_r, \mathbf{l}),$$

$$h_{ir}(\mathbf{v}_r, \mathbf{l}) = \begin{cases} \binom{l_i + v_{ir}}{v_{ir}} w_{ir}^{l_i} & \text{if center } i \text{ is FCFS, LCFSPR, or PS,} \\ \dfrac{w_{ir}^{l_i}}{l_i!} & \text{if center } i \text{ is IS,} \end{cases}$$

and the initial conditions are $G_0(\mathbf{v}_0) = 1$ for all $\mathbf{v}_0 \in \mathcal{J}_0$.

PROOF. The proof is contained in Appendix A. □

Theorem 1 gives a new recursive expression for computing the normalization constant $G = G_R(\mathbf{0})$. In effect, the normalization constant $G_r(\mathbf{v}_r)$ of a network with r chains is decomposed into a sum of normalization constants for networks having $(r-1)$ chains and in which the customers of chain r are fixed at the nodes and cannot depart. The number of networks involved in this sum is the number of distinct ways K_r customers of chain r may be distributed over N nodes. In our algorithm, however, to compute the mean performance measures, we do not use Theorem 1, but rather a simplified version of it (Corollary 1 below), which results when the population of all chains is one. (We do not pursue the option of developing an algorithm based *directly* on Theorem 1 since we have been unable to obtain simple expressions for the mean performance measures in terms of the normalization constants that arise in the recursion of Theorem 1. We have, however, formulated an efficient algorithm, which is based directly on Theorem 1, to compute the normalization constant $G = G_R(\mathbf{0})$, which may be of interest in itself. This is presented in [6]. The time requirement of this algorithm is not significantly different from the one to be presented here, and the space requirement is, in general, considerably higher.)

COROLLARY 1. *If $K_r = 1$ for $1 \le r \le R$, then $G_R(\mathbf{0})$ is given recursively by*

$$G_r(\mathbf{v}_r) = \sum_{i=1}^{N} (1 + v_{ir}\delta_i) w_{ir} G_{r-1}(\mathbf{v}_r + \mathbf{1}_i), \qquad \text{for} \quad 1 \le r \le R, \quad \mathbf{v}_r \in \mathcal{S}_r,$$

where $\mathbf{1}_i$ is a unit vector pointing in the ith direction,

$$\delta_i = \begin{cases} 1 & \text{if center } i \text{ is FCFS, LCFSPR, or PS,} \\ 0 & \text{if center } i \text{ is IS,} \end{cases}$$

and the initial conditions are $G_0(\mathbf{v}_0) = 1$ for all $\mathbf{v}_0 \in \mathcal{S}_0$, where now

$$\mathcal{S}_0 = \left\{ \mathbf{v}_0 \,|\, v_{i0} \ge 0 \text{ for } 1 \le i \le N; \sum_{i=1}^{N} v_{i0} = R \right\}.$$

PROOF. The Corollary follows directly from Theorem 1 when $K_r = 1$ for $1 \le r \le R$. □

THEOREM 2

$$Pr(\mathbf{n}_R = \mathbf{k}) = G_R(\mathbf{0})^{-1} G_{R-1}(\mathbf{k}) \prod_{i=1}^{N} b_i(k_i),$$

where

$$\mathbf{n}_R = (n_{1R}, \ldots, n_{NR}),$$

$$\mathbf{k} = (k_1, \ldots, k_N),$$

and

$$b_i(k_i) = \begin{cases} w_{iR}^{k_i} & \text{if center } i \text{ is FCFS, LCFSPR, or PS,} \\ \dfrac{w_{iR}^{k_i}}{k_i!} & \text{if center } i \text{ is IS.} \end{cases}$$

PROOF. The proof is contained in Appendix B. □

Theorem 2 gives a new expression for the marginal distribution with respect to a particular chain R.

THEOREM 3

(*a*) *If* $K_R = 1$ *and there are no IS centers in the network, then*

$$G_{R-1}(\mathbf{0}) = \sum_{l=1}^{N} \frac{G_{R-1}(\mathbf{1}_l)}{N + K - 1}$$

where

$$K = \sum_{r=1}^{R} K_r.$$

(*b*) *If* x *is an IS center, then* $G_{R-1}(\mathbf{0}) = G_{R-1}(\mathbf{1}_x)$.

PROOF. The proofs are contained in Appendix C. □

We now give some results concerning the mean performance measures for chain R when $K_R = 1$. We denote the throughput, utilization, mean queue length (node population), and mean waiting time (queuing + service) of chain r customers at node i by T_{ir}, U_{ir}, Q_{ir}, and W_{ir}, respectively.

COROLLARY 2. *If* $K_R = 1$, *then*

$$T_{iR} = \begin{cases} \left(\dfrac{w_{iR}}{t_{iR}}\right) \displaystyle\sum_{l=1}^{N} \frac{G_{R-1}(\mathbf{1}_l)}{G_R(\mathbf{0})(N + K - 1)} & \text{if there are no IS centers in the network,} \\ \left(\dfrac{w_{iR}}{t_{iR}}\right) \dfrac{G_{R-1}(\mathbf{1}_x)}{G_R(\mathbf{0})} & \text{if there are IS centers and } x \text{ is any one of them,} \end{cases}$$

$$U_{iR} = t_{iR} T_{iR},$$

$$Q_{iR} = G_R(\mathbf{0})^{-1} G_{R-1}(\mathbf{1}_i) w_{iR},$$

$$W_{iR} = \frac{Q_{iR}}{T_{iR}}.$$

PROOF. It is well known [2, 11, 19] that, in terms of the notation adopted here, when $K_R = 1$, $T_{iR} = e_{iR} G_{R-1}(\mathbf{0})/G_R(\mathbf{0})$. The expressions for T_{iR} then follow from Theorem 3. The expression for Q_{iR} follows directly from its definition and Theorem 2. The expressions for U_{iR} and W_{iR} are well known and based on Little's result [13]. □

4. *The Recursion by Chain Algorithm—RECAL*

The new algorithm for multiple-chain networks is based directly on Corollaries 1 and 2 and the introduction of the artifice of breaking down each chain r, with K_r customers, into K_r identical subchains, with one customer each. The basic idea of employing this artifice is to create an artificial network in which $K_r = 1$ for all chains so that we may apply the simpler recursion of Corollary 1 and obtain the mean performance measures using Corollary 2. The artificial network has, in general, a different state space and normalization constant from the original network, but we may nevertheless employ the artifice and obtain the correct values for the performance measures since it in no way alters the physical characteristics of the original network.

Let the network created from $\mathcal{N}$, by breaking down each chain, be denoted by $\mathcal{N}^\circ$. (We use a dot ($^\circ$) to differentiate the notation associated with $\mathcal{N}^\circ$.) The total population K° of $\mathcal{N}^\circ$ is $K^\circ = \sum_{r=1}^{R} K_r$. The number of chains R° in $\mathcal{N}^\circ$ is, by the definition of $\mathcal{N}^\circ$, K°, and the population of chain r is $K_r^\circ = 1$ for $1 \le r \le K^\circ$. Let the customers in $\mathcal{N}^\circ$ be enumerated as 1, 2, ..., K°, and let the chain to which customer k belongs in $\mathcal{N}$ be $r(k)$. We then have, using Corollary 1, that $G_{R^\circ}^\circ(\mathbf{0})$ is given recursively by

$$G_k^\circ(\mathbf{v}_k) = \sum_{i=1}^{N} (1 + v_{ik}\delta_i) w_{ir(k)} G_{k-1}^\circ(\mathbf{v}_k + \mathbf{1}_i) \tag{4.1}$$

for $1 \le k \le K^\circ$, $\mathbf{v}_k \in \mathcal{S}_k^\circ$, where $\mathcal{S}_k^\circ = \{\mathbf{v}_k \mid v_{ik} \ge 0$ for $1 \le i \le N$; $\sum_{i=1}^{N} v_{ik} = K^\circ - k\}$. The initial conditions are $G_0^\circ(\mathbf{v}_0) = 1$ for all $\mathbf{v}_0 \in \mathcal{S}_0^\circ$.

Hence, by breaking down each chain in $\mathcal{N}$ into constituent subchains we can apply Corollary 1 and circumvent, in particular, the need to compute the terms $g_r(\mathbf{v}_r, \mathbf{l})$, which appear in the expression of Theorem 1. We note that, in general, $G_{R^\circ}^\circ(\mathbf{0}) \ne G_R(\mathbf{0})$ since, in general, $\mathcal{S}^{\circ(R^\circ)} \ne \mathcal{S}^{(R)}$. Hence, we are no longer able to compute the normalization constant $G = G_R(\mathbf{0})$, which is supposed to be of central interest. Nevertheless, using Corollary 2, we may write

$$T_{iR^\circ}^\circ = \begin{cases} \left(\dfrac{w_{ir(R^\circ)}}{t_{ir(R^\circ)}}\right) \displaystyle\sum_{l=1}^{N} \frac{G_{R^\circ-1}^\circ(\mathbf{1}_l)}{G_{R^\circ}^\circ(\mathbf{0})(N + K^\circ - 1)} & \text{if there are no IS centers in the network,} \\ \left(\dfrac{w_{ir(R^\circ)}}{t_{ir(R^\circ)}}\right) \dfrac{G_{R^\circ-1}^\circ(\mathbf{1}_x)}{G_{R^\circ}^\circ(\mathbf{0})} & \text{if there are IS centers and } x \text{ is any one of them,} \end{cases}$$

$$U_{iR^\circ}^\circ = t_{ir(R^\circ)} T_{iR^\circ}^\circ, \tag{4.2}$$

$$Q_{iR^\circ}^\circ = G_{R^\circ}^\circ(\mathbf{0})^{-1} G_{R^\circ-1}^\circ(\mathbf{1}_i) w_{ir(R^\circ)},$$

$$W_{iR^\circ}^\circ = \frac{Q_{iR^\circ}^\circ}{T_{iR^\circ}^\circ}.$$

The above expressions give the mean performance measures with respect to a *single* customer who belongs to chain $r(R^\circ)$ in $\mathcal{N}$. The normalization constants in eq. (4.2) are obtained in the computation of $G_{R^\circ}^\circ(\mathbf{0})$ using eq. (4.1). The key point remaining in the algorithm is that all customers who belong to a particular chain in $\mathcal{N}$ are statistically indistinguishable in equilibrium, so we finally have for the network $\mathcal{N}$

$$\begin{aligned} T_{ir(R^\circ)} &= K_{r(R^\circ)} T_{iR^\circ}^\circ, \\ U_{ir(R^\circ)} &= K_{r(R^\circ)} U_{iR^\circ}^\circ, \\ Q_{ir(R^\circ)} &= K_{r(R^\circ)} Q_{iR^\circ}^\circ, \\ W_{ir(R^\circ)} &= W_{iR^\circ}^\circ. \end{aligned} \tag{4.3}$$

Hence, we are able to obtain the mean performance measures for a particular chain $r(R^\circ)$ using eq. (4.1), even though the value for G itself is never obtained.

The mean performance measures for chains other than $r(R^\circ)$ may be obtained by reenumerating the customers in $\mathcal{N}^\circ$ so that $r(R^\circ)$ is changed to another chain for which one wishes to compute performance measures using eqs. (4.1)–(4.3). More specifically, suppose that the customers in $\mathcal{N}^\circ$ are initially enumerated so that the initial assignment of the customers in $\mathcal{N}^\circ$ to chains in $\mathcal{N}$ is $r^{(1)}(k)$, as

TABLE I. ASSIGNMENTS OF THE CUSTOMERS IN $\mathcal{N}^\circ$ TO THE CHAINS IN $\mathcal{N}$.

Customer number in $\mathcal{N}^\circ$ (k)	Assignment of customer k to chains in $\mathcal{N}^{(R)}$					
	$r^{(1)}(k)$	$r^{(2)}(k)$	$r^{(s)}(k)$	$r^{(s+1)}(k)$	$r^{(R-1)}(k)$	$r^{(R)}(k)$
K°	1	2	s	$s+1$	$R-1$	R
$K^\circ-1$	2	3	$s+1$	$s+2$	R	$R-1$
$K^\circ-2$	3	4	$s+2$	$s+3$	$R-2$	$R-2$
.	.	.	.	.	.	.
.	.	.	.	R	.	.
.	.	.	R	s	.	.
.	.	.	$s-1$	$s-1$	.	.
.	.	.	.	.	.	.
$K^\circ-R+3$	$R-2$	$R-1$	3	3	3	3
$K^\circ-R+2$	$R-1$	R	2	2	2	2
$K^\circ-R+1$	R	1	1	1	1	1
$K^\circ-R$	R					
.	.					
.	.					
$1+\sum_{r=1}^{R-1}(K_r-1)$	R					
$\sum_{r=1}^{R-1}(K_r-1)$	$R-1$					
.	.					
.	.					
.	.					
K_1+K_2-1	3			same as $r^{(1)}(k)$		
K_1+K_2-2	2					
.	.					
.	.					
K_1	2					
K_1-1	1					
.	.					
.	.					
1	1					

illustrated in Table I and given by

$$r^{(1)}(k)=\begin{cases}1 & \text{for } 1\le k\le K_1-1,\\ i & \text{for } 1+\sum_{r=1}^{i-1}(K_r-1)\le k\le \sum_{r=1}^{i}(K_r-1),\\ & \quad 2\le i\le R,\\ R-k+1+\sum_{r=1}^{R}(K_r-1) & \text{for } 1+\sum_{r=1}^{R}(K_r-1)\le k\le K^\circ.\end{cases}\tag{4.4}$$

Having arrived at $G_R^\circ(\mathbf{0})$ using eq. (4.1), we may compute the mean performance measures for chain 1 in $\mathcal{N}$ using eqs. (4.2) and (4.3). We then reenumerate the customers in $\mathcal{N}^\circ$ to obtain a new assignment of the customers in $\mathcal{N}^\circ$ to chains in $\mathcal{N}$. Let the new assignment be $r^{(2)}(k)$, as illustrated in Table I and given by

$$r^{(2)}(k)=\begin{cases}r^{(1)}(k) & \text{for } 1\le k\le \sum_{r=1}^{R}(K_r-1),\\ 1 & \text{for } k=1+\sum_{r=1}^{R}(K_r-1),\\ r^{(1)}(k)+1 & \text{for } 2+\sum_{r=1}^{R}(K_r-1)\le k\le K^\circ.\end{cases}$$

The mean performance measures for chain 2 in $\mathcal{N}$ may then be obtained after $G^{\circ}_{R^{\circ}}(\mathbf{0})$ has been recomputed. In general then, after having computed the performance measures for chain s in $\mathcal{N}$, we reenumerate the customers in $\mathcal{N}^{\circ}$ so that

$$r^{(s+1)}(k) = \begin{cases} r^{(s)}(k) & \text{for } 1 \le k \le s - 1 + \sum_{r=1}^{R} (K_r - 1), \\ s & \text{for } k = s + \sum_{r=1}^{R} (K_r - 1), \\ r^{(s)}(k) + 1 & \text{for } s + 1 + \sum_{r=1}^{R} (K_r - 1) \le k \le K^{\circ}. \end{cases} \tag{4.5}$$

We then recompute $G^{\circ}_{R^{\circ}}(\mathbf{0})$ and obtain the measures for chain $(s + 1)$. This procedure of reassignment and recomputation of $G^{\circ}_{R^{\circ}}(\mathbf{0})$ is rendered efficient by storing $G^{\circ}_{x-1}(\mathbf{v}_{x-1})$ for all $\mathbf{v}_{x-1} \in \mathcal{S}^{\circ}_{x-1}$, where

$$x = s + \sum_{r=1}^{R} (K_r - 1),$$

so that in the determination of the performance measures for chain $(s + 1)$ in $\mathcal{N}$ we need not reevaluate eq. (4.1) for all $1 \le k \le K^{\circ}$ but *only* for $x \le k \le K^{\circ}$.

We now summarize the algorithm that has been developed.

RECAL: A recursion by chain algorithm for computing mean performance measures.
Step 1: Initialize $G^{\circ}_{0}(\mathbf{v}_0) = 1$ for all $\mathbf{v}_0 \in \mathcal{S}^{\circ}_{0}$.
Step 2: Enumerate the customers in $\mathcal{N}^{\circ}$ so that the assignment of customers in $\mathcal{N}^{\circ}$ to chains in $\mathcal{N}$ is $r^{(1)}(k)$, given by eq. (4.4), for $1 \le k \le K^{\circ}$.
Step 3: Compute and store $G^{\circ}_{x}(\mathbf{v}_x)$ for all $\mathbf{v}_x \in \mathcal{S}^{\circ}_{x}$, where $x = K^{\circ} - R$, using eq. (4.1).
Step 4: For each chain s in $\mathcal{N}$, $1 \le s \le R$:
(a) Determine $G^{\circ}_{K^{\circ}}(\mathbf{0})$ using eq. (4.1) and the stored values of $G^{\circ}_{x}(\mathbf{v}_x)$, $\mathbf{v}_x \in \mathcal{S}^{\circ}_{x}$.
(b) Compute the mean performance measures for chain s using $G^{\circ}_{K^{\circ}}(\mathbf{0})$, $G^{\circ}_{K^{\circ}-1}(\mathbf{1}_i)$ for $1 \le i \le N$, eqs. (4.2) and (4.3). Stop if $s = R$.
(c) Reenumerate the customers in $\mathcal{N}^{\circ}$ so that the assignment of customers in $\mathcal{N}^{\circ}$ to chains in $\mathcal{N}$ is $r^{(s+1)}(k)$, as defined by eq. (4.5).
(d) Increment x by 1.
(e) Compute and store $G^{\circ}_{x}(\mathbf{v}_x)$ for all $\mathbf{v}_x \in \mathcal{S}^{\circ}_{x}$ using eq. (4.1) and the stored values of $G^{\circ}_{x-1}(\mathbf{v}_{x-1})$, $\mathbf{v}_{x-1} \in \mathcal{S}^{\circ}_{x-1}$.

5. *Computational Complexity*

In this section we determine the time and space requirements of the algorithm. We derive the number of operations (additions and multiplications) and the storage space (number of elements) required. For simplicity we assume that there are no IS centers in the network. When there are IS centers, however, the computations are simplified slightly, since $\delta_i = 0$ when i is an IS center.

5.1 TIME REQUIREMENT. Let us begin by evaluating the number of operations involved in step 3. The evaluation of the summation in eq. (4.1) requires $(4N - 1)$ operations. The number of operations to obtain $G^{\circ}_{1}(\mathbf{v}_1)$ for all $\mathbf{v}_1 \in \mathcal{S}^{\circ}_{1}$ is

$$(4N - 1)\binom{K^{\circ} + N - 2}{N - 1},$$

the combinatorial term being the cardinality of $\mathcal{S}^{\circ}_{1}$. The total number of operations to obtain $G^{\circ}_{x}(\mathbf{v}_x)$ for all $\mathbf{v}_x \in \mathcal{S}^{\circ}_{x}$, where $x = K^{\circ} - R$, is, therefore,

$$\sum_{i=1}^{K^{\circ}-R} (4N - 1)\binom{K^{\circ} - i + N - 1}{N - 1}. \tag{5.1}$$

With $s = 1$, the number of operations to carry out step 4(a) is

$$\sum_{i=K^\bullet-R+1}^{K^\bullet} (4N-1)\binom{K^\circ - i + N - 1}{N-1}.$$

In step 4(b) we require $(N + 2)$ operations to compute $T^\circ_{iR^\bullet}$ using eq. (4.2) (assuming that there are no IS centers in the network and that the constant $(N + K^\circ - 1)$ has been precomputed). We then require six more operations to obtain $T_{ir(R^\bullet)}$, $U_{ir(R^\bullet)}$, $Q_{ir(R^\bullet)}$, and $W_{ir(R^\bullet)}$ using eqs. (4.2) and (4.3). The number of operations involved in step 4(b) is, therefore, $(N + 8)$. We loop on step 4 R times, so that the total number of operations consumed in step 4(b) is

$$R(N + 8). \tag{5.2}$$

With $s = 1$, the number of operations to carry out step 4(e) is

$$(4N-1)\binom{R - 1 + N - 1}{N-1}.$$

In general, the number of operations required in step 4(a) is

$$\sum_{i=K^\bullet-R+s}^{K^\bullet} (4N-1)\binom{K^\circ - i + N - 1}{N-1}, \tag{5.3}$$

and the number of operations involved in step 4(e) is

$$(4N-1)\binom{R - s + N - 1}{N-1}. \tag{5.4}$$

The total number of operations required by the algorithm is, using eqs. (5.1)–(5.4),

$$\begin{aligned}
&\sum_{i=1}^{K^\bullet-R} (4N-1)\binom{K^\circ - i + N - 1}{N-1} \\
&\quad + \sum_{s=1}^{R} \sum_{i=K^\bullet-R+s}^{K^\bullet} (4N-1)\binom{K^\circ - i + N - 1}{N-1} \\
&\quad + \sum_{s=1}^{R-1} (4N-1)\binom{R - s + N - 1}{N-1} + R(N + 8),
\end{aligned}$$

which simplifies to

$$\begin{aligned}
(4N-1)\Bigg(&\binom{K^\circ + N - 1}{N} + \binom{R + N - 1}{N+1} \\
&+ \binom{R + N - 1}{N} - 1\Bigg) + R(N + 8).
\end{aligned} \tag{5.5}$$

Now suppose that $K_r = \kappa$ for $1 \le r \le R$. Then $K^\circ = \kappa R$. If we consider κ and N fixed, then eq. (5.5) is a polynomial in R of degree $(N + 1)$. When R is large, eq. (5.5) is on the order of

$$\frac{4N-1}{(N+1)!} R^{N+1}.$$

TABLE II. EXAMPLE OF THE MAPPING $M_x(\mathbf{d})$ ($N = 3$, $K^\circ = 3$, $x = 0$)

$\mathbf{d}$	$M_x(\mathbf{d})$
300	1
210	2
120	3
030	4
201	5
111	6
021	7
102	8
012	9
003	10

If we now consider R and κ fixed, then eq. (5.5) is a polynomial in N of degree (κR). When N is large, eq. (5.5) is on the order of

$$\frac{8N^R}{(R-1)!} \quad \text{if} \quad \kappa = 1,$$

$$\frac{4N^{\kappa R}}{(\kappa R - 1)!} \quad \text{if} \quad \kappa > 1.$$

5.2. SPACE REQUIREMENT. The space requirement is dictated by the maximum storage space that is required at any one time in the implementation of eq. (4.1). It is assumed, for the sake of simplicity in implementation, that we compute $G_k^\circ(\mathbf{v}_k)$ for all $\mathbf{v}_k \in \mathcal{J}_k^\circ$ before incrementing k.

Suppose that $G_{k-1}^\circ(\mathbf{v}_{k-1})$ has been computed and stored for all $\mathbf{v}_{k-1} \in \mathcal{J}_{k-1}^\circ$. Furthermore, suppose that the value for $G_{k-1}^\circ(\mathbf{v}_{k-1})$ is held in an array at location $M_{k-1}(\mathbf{v}_{k-1})$, where $M_x(\mathbf{d})$ is a mapping that gives increasing values of the storage location index to vectors $\mathbf{d}$, which give increasing values for the sum

$$\sum_{i=1}^{N} d_i(K^\circ - x)^{i-1}. \tag{5.6}$$

Such a mapping has been introduced in [7, sect. 4.2]. The domain of $M_x(\mathbf{d})$ is $\mathcal{J}_x^\circ$, and the range is $\{1, 2, \ldots, \binom{K_0 - x + N - 1}{N - 1}\}$, the combinatorial term being the cardinality of the set $\mathcal{J}_x^\circ$. An example mapping is given in Table II.

If we now compute the values of $G_k^\circ(\mathbf{v}_k)$, $\mathbf{v}_k \in \mathcal{J}_k^\circ$, in the order that gives increasing values for the sum

$$\sum_{i=1}^{N} v_{ik}(K^\circ - k)^{i-1},$$

then a useful consequence is that, after having computed $G_k^\circ(\mathbf{v}_k)$ using eq. (4.1), we may store the result at location $M_k(\mathbf{v}_k)$ of the *same* array that was used to store $G_{k-1}^\circ(\mathbf{v}_{k-1})$ for all $\mathbf{v}_{k-1} \in \mathcal{J}_{k-1}^\circ$. We illustrate this mechanism in Figure 1. This storage procedure may be adopted since the value of $G_{k-1}^\circ(\mathbf{v}_{k-1})$ at location $M_k(\mathbf{v}_k)$ is never required, after $G_k^\circ(\mathbf{v}_k)$ has been computed, for any of the computations of $G_k^\circ(\boldsymbol{\gamma})$, where $\boldsymbol{\gamma}$ is any vector such that $\boldsymbol{\gamma} \in \mathcal{J}_k^\circ$ and $M_k(\boldsymbol{\gamma}) > M_k(\mathbf{v}_k)$. This point is proved in Appendix D. As a result, the required storage space to implement eq. (4.1) is only the cardinality of $\mathcal{J}_0^\circ$ and not the sum of the cardinalities of $\mathcal{J}_0^\circ$ and $\mathcal{J}_1^\circ$. The cardinality of $\mathcal{J}_0^\circ$ is $\binom{K^\circ + N - 1}{N - 1}$. In addition, step 3 requires a storage space

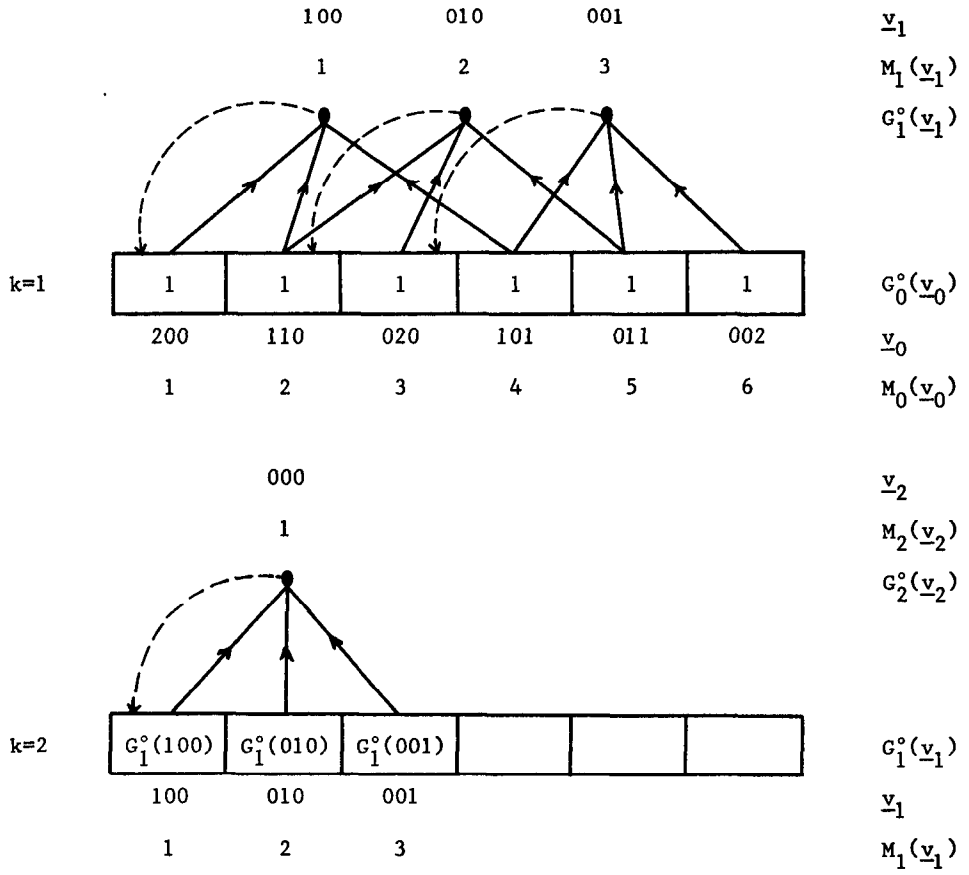

FIG. 1. Illustration of the implementation of eq. (4.1) ($N = 3$, $K^\circ = 2$): ⇢ pointer to storage location of $G_k^\circ(\mathrm{v}_k)$; ⟶, elements needed in the computation of $G_k^\circ(\mathrm{v}_k)$.

equal to the cardinality of $\mathcal{I}_{K^\circ-R}^\circ$ which is $\binom{R+N-1}{N-1}$. The values obtained in step 4(e) may be stored in the same storage space used for step 3. The total required storage space for the algorithm is then

$$\binom{K^\circ + N - 1}{N - 1} + \binom{R + N - 1}{N - 1}. \tag{5.7}$$

With $K_r = \kappa$ for $1 \le r \le R$ and with N and κ considered fixed, eq. (5.7) is a polynomial in R of degree $(N - 1)$. When R is large, eq. (5.7) is on the order of

$$\frac{\kappa^{N-1} + 1}{(N-1)!} R^{N-1}.$$

If we now consider R and κ fixed, then eq. (5.7) is a polynomial in N of degree (κR). When N is large, eq. (5.7) is on the order of

$$\frac{2N^R}{R!} \quad \text{if} \quad \kappa = 1,$$

$$\frac{N^{\kappa R}}{(\kappa R)!} \quad \text{if} \quad \kappa > 1.$$

6. *Comparisons in Complexity*

In this section we compare the time and space requirements of RECAL with those of the convolution and MVA algorithms.

If we use the version of the convolution algorithm [2] that applies to networks with constant speed servers, as is the case here, the total number of operations to arrive at G is

$$2R(N-1)\prod_{r=1}^{R}(K_r+1), \tag{6.1}$$

and the required storage space, in number of elements, is

$$2\prod_{r=1}^{R}(K_r+1). \tag{6.2}$$

The MVA algorithm has approximately the same time requirement as eq. (6.1) [18] and the required storage space is [2]

$$N\prod_{r=1}^{R}(K_r+1). \tag{6.3}$$

We show in Appendix E that no computational advantage can be obtained in the convolution or MVA algorithms by breaking down the chains in $\mathcal{N}$ into subchains.

In the previous section we have seen that with N and κ considered fixed, the time and space requirements of RECAL are polynomial in R. Hence, if R is sufficiently large, the time and space requirements given by eqs. (5.5) and (5.7), respectively, will be less than those given by eqs. (6.1) and (6.2) or (6.3).

We see from eqs. (5.7), (6.2), and (6.3) that RECAL has a smaller space requirement than either convolution or MVA when

$$\binom{\sum_{r=1}^{R}K_r+N-1}{N-1}+\binom{R+N-1}{N-1}<2\prod_{r=1}^{R}(K_r+1).$$

From eqs. (5.5) and (6.1) we see that RECAL has a smaller time requirement when

$$(4N-1)\left(\binom{\sum_{r=1}^{R}K_r+N-1}{N}+\binom{R+N-1}{N+1}+\binom{R+N-1}{N}-1\right)+R(N+8)<2R(N-1)\prod_{r=1}^{R}(K_r+1).$$

In Figure 2 we compare, for the purposes of illustration, the number of operations given by eq. (5.5) with those given by eq. (6.1) in the case in which $N=4$ and $K_r=\kappa$ for $1\le r\le R$, for various values of κ and R. In Figure 3 we compare the storage requirements given by eq. (5.7) with those given by eq. (6.2) in the case in which $N=4$. In Figure 4 we determine, for various values of κ, the region in the space of queuing networks ($N\times R$) in which the number of operations required by RECAL (eq. (5.5)) is less than that required with convolution (eq. (6.1)). In Figure 5 we determine the region in which the storage space requirement of RECAL (eq. (5.7)) is less than that of convolution (eq. (6.2)). In Figures 6 and 7 we give, respectively, several isotime and isostorage curves in the space ($N\times R$) for RECAL and convolution in the case in which $\kappa=3$. We see from Figures 2–7 that RECAL is useful when there are many chains in the network. We finally note that, when $N=2$, the storage requirement of RECAL, given by eq. (5.7), is linear in R.

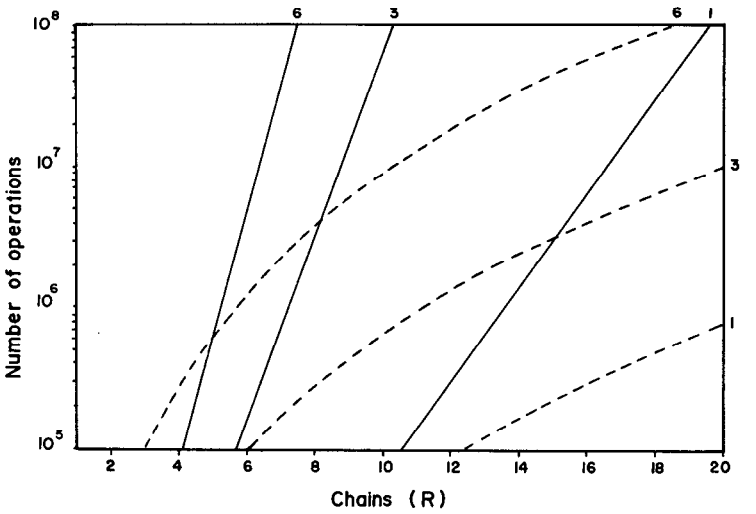

FIG. 2. Comparison of the number of operations in RECAL and convolution, for $N = 4$ and $\kappa = 1$, 3, and 6 (the values for κ are indicated at the ends of the curves): ——, convolution; ---, RECAL.

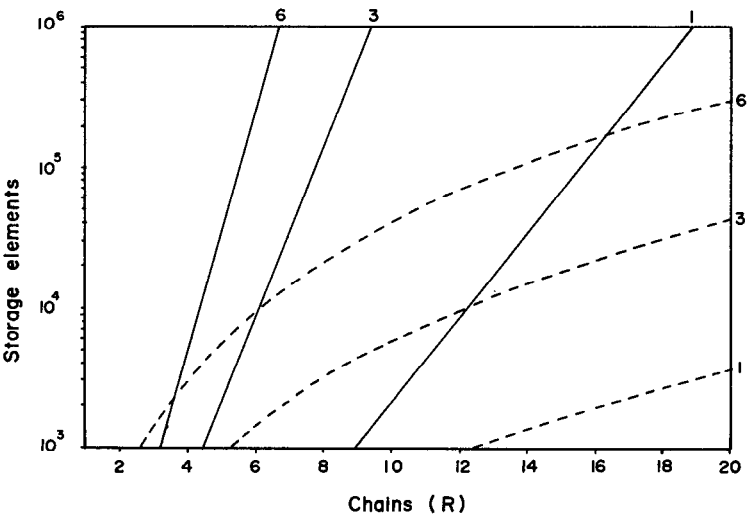

FIG. 3. Comparison of the storage space requirements in RECAL and convolution, for $N = 4$ and $\kappa = 1$, 3, and 6 (the values for κ are indicated at the ends of the curves): ——, convolution; ---, RECAL.

7. *Extensions for State-Dependent Service Centers*

In the previous sections we have assumed, for the sake of simplicity in presentation, that the servers at the service centers operate at a constant speed. In this section we extend RECAL to accommodate the situation in which there may be state-dependent service rates. An attractive feature of RECAL is that this extension may be made with little additional complexity. In the convolution and MVA algorithms such an extension is not so trivial [2, 11, 19].

Consider a queuing network of type $\mathcal{N}$, but with state-dependent servers at the service centers. Let the service rate function for center i depend on the total number

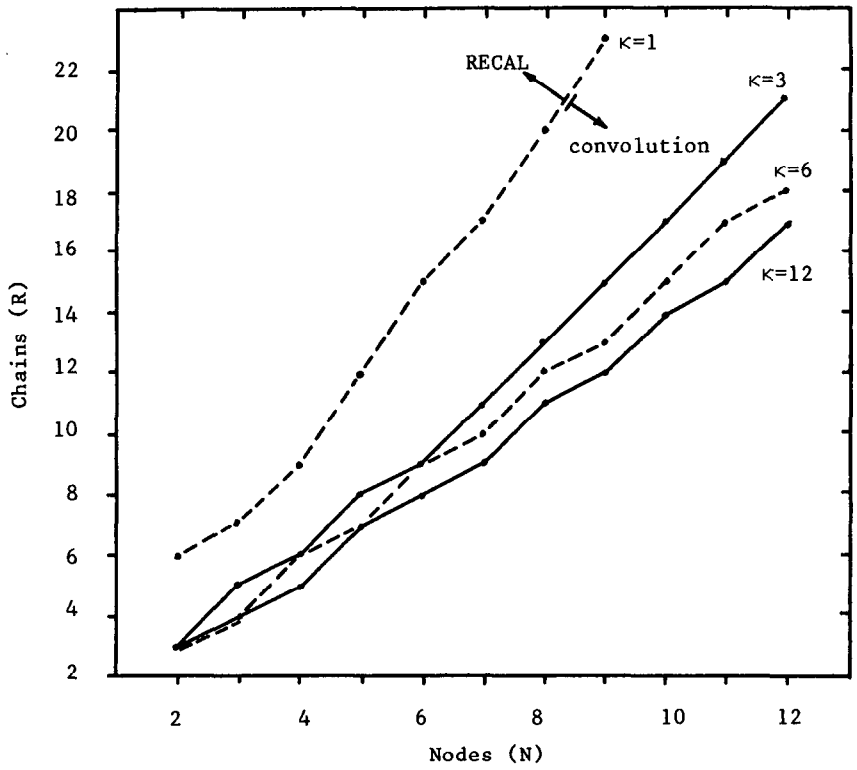

FIG. 4. Regions in the space ($N \times R$) in which the time requirement (number of operations) of RECAL is less than convolution, for $\kappa = 1$, 3, 6, and 12 (on or above the curves the time requirement of RECAL is less than convolution).

of customers at center i and be denoted by $\beta_i(n)$. At IS centers, with the definition of $f_i(\mathbf{n}_i^{(R)})$ that has been adopted in Section 2, we have $\beta_i(n) = 1$. With state-dependent servers, for $\mathbf{n}^{(R)} \in \mathcal{S}^{(R)}$, the marginal state distribution for the network is [1],

$$\Pr(\mathbf{n}^{(R)}) = G^{-1} \prod_{i=1}^{N} \left(\frac{f_i(\mathbf{n}_i^{(R)})}{\prod_{a=1}^{n_i^{(R)}} \beta_i(a)} \right). \tag{7.1}$$

Without loss of generality we need not explicitly consider the more general situation, which has been considered in [1], in which there may be different service rate functions, $\beta_{ir}(n)$, for each chain r.

Using eq. 7.1, and the same line of reasoning used in the proof of Theorem 1 (Appendix A), we may show that $G_R(\mathbf{0})$ is given by the same recursive formula that appears in Theorem 1 but with

$$h_{ir}(\mathbf{v}_r, \mathbf{l}) = \binom{l_i + v_{ir}}{v_{ir}} \frac{w_{ir}^{l_i}}{\prod_{b=1}^{l_i} \beta_i(b + v_{ir})}, \tag{7.2}$$

if center i is FCFS, LCFSPR, or PS. The quantity $G_r(\mathbf{v}_r)$ in Theorem 1 is now the normalization constant for a network of type $\mathcal{N}$, but with servers that have state-dependent service rate functions $u_i(n) = \beta_i(n + v_{ir})n/(n + v_{ir})$.

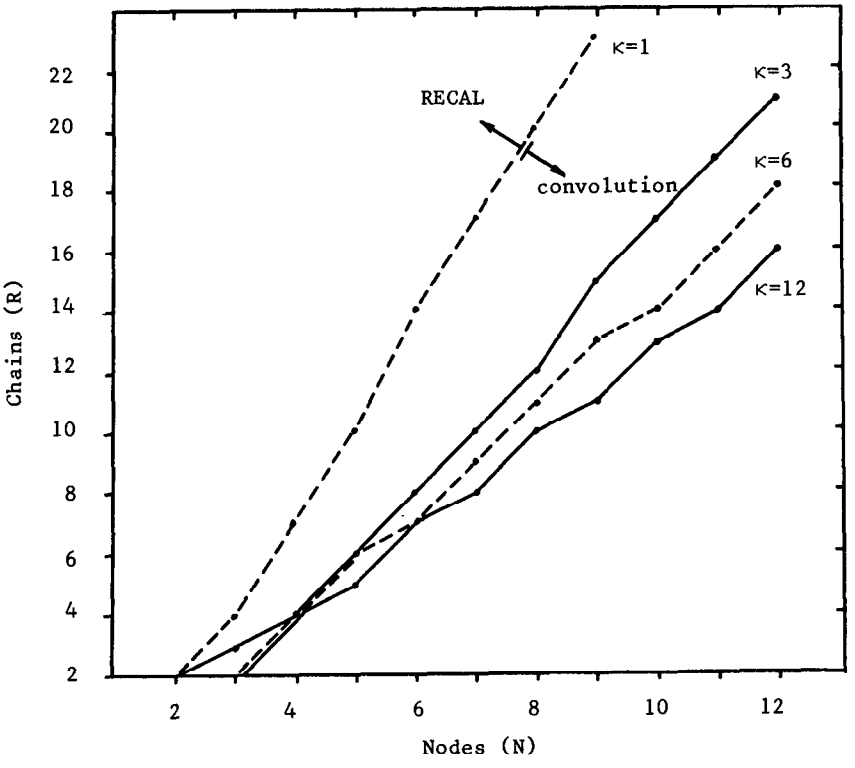

FIG. 5. Regions in the space ($N \times R$) in which the space requirement (number of elements) of RECAL is less than convolution, for $\kappa = 1$, 3, 6, and 12 (on or above the curves the space requirement of RECAL is less than convolution).

With these changes, we may then rewrite eq. (4.1) as

$$G_k^\circ(\mathbf{v}_k) = \sum_{i=1}^{N} \frac{(1 + v_{ik}\delta_i)w_{ir(k)}G_{k-1}^\circ(\mathbf{v}_k + \mathbf{1}_i)}{\beta_i(1 + v_{ik})}. \tag{7.3}$$

Comparing eq. (7.3) with eq. (4.1), we see that little additional complexity has been introduced to accommodate the situation of state dependency.

The expression for the marginal distribution with respect to a particular chain R, in the case of state dependency, is unchanged from Theorem 2, except that, if center i is FCFS, LCFSPR, or PS, then

$$b_i(k_i) = \frac{w_{iR}^{k_i}}{\prod_{a=1}^{k_i} \beta_i(a)}.$$

As a result, if $K_R = 1$, then $Q_{iR} = G_R(\mathbf{0})^{-1}G_{R-1}(\mathbf{1}_i)w_{iR}/\beta_i(1)$ so that

$$Q_{iR^*}^\circ = \frac{G_{R^*}^\circ(\mathbf{0})^{-1}G_{R^*-1}^\circ(\mathbf{1}_i)w_{ir(R^*)}}{\beta_i(1)}. \tag{7.4}$$

Under the additional condition that there is at least one IS center in the network we have, in the case of state dependency,

$$T_{iR^*}^\circ = \frac{e_{ir(R^*)}G_{R^*-1}^\circ(\mathbf{1}_x)}{G_{R^*}^\circ(\mathbf{0})}, \tag{7.5}$$

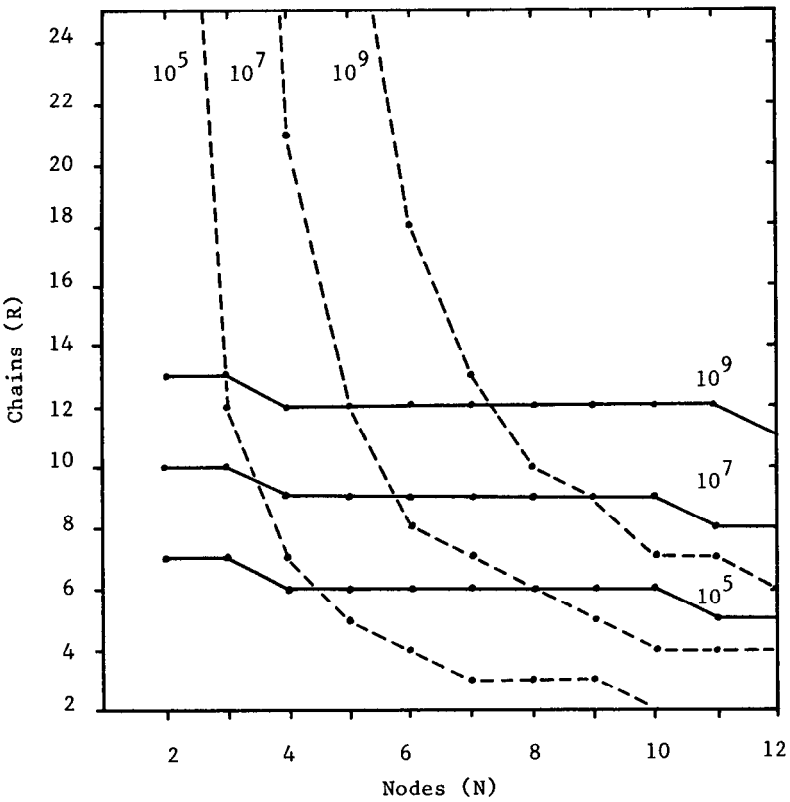

FIG. 6. Curves in the space ($N \times R$) on and above which the time requirement (number of operations) is equal to or greater than the number at the end of the curve, for $\kappa = 3$: ---, RECAL; ——, convolution.

where x is any one of the IS centers. This expression for the throughput follows from the identity $G_{R-1}(\mathbf{0}) = G_{R-1}(\mathbf{1}_x)$ and the result that, when $K_R = 1$, $T_{iR} = e_{iR}G_{R-1}(\mathbf{0})/G_R(\mathbf{0})$, which is known to hold *even* in the case of state dependency [2, p. 71]. We also have, using Little's result [13],

$$W^{\circ}_{iR^{\bullet}} = \frac{Q^{\circ}_{iR^{\bullet}}}{T^{\circ}_{iR^{\bullet}}}. \tag{7.6}$$

If we assume that there is one IS center in the network and that there is state dependency at all service centers, then the evaluation of the summation in eq. (7.3) requires $(5N - 4)$ operations. The total number of operations required by the algorithm to obtain the queue lengths, throughputs, and waiting times is then,

$$(5N-4)\left(\binom{K^{\circ}+N-1}{N} + \binom{R+N-1}{N+1} + \binom{R+N-1}{N} - 1\right) + 8R.$$

This result can be derived in a manner analogous to that of eq. (5.5). When there is more than one IS center, the number of operations is reduced slightly, since $\delta_i = 0$ when i is an IS center. The required storage space, in the case of state dependency, is unchanged from eq. (5.7), except, of course, for the additional space required to store the network parameters $\beta_i(n)$, which we ignore.

We finally mention that it does not appear possible for RECAL to accommodate state-dependent servers whose service-rate functions are of the form $\beta_i(\mathbf{n}_i^{(R)})$.

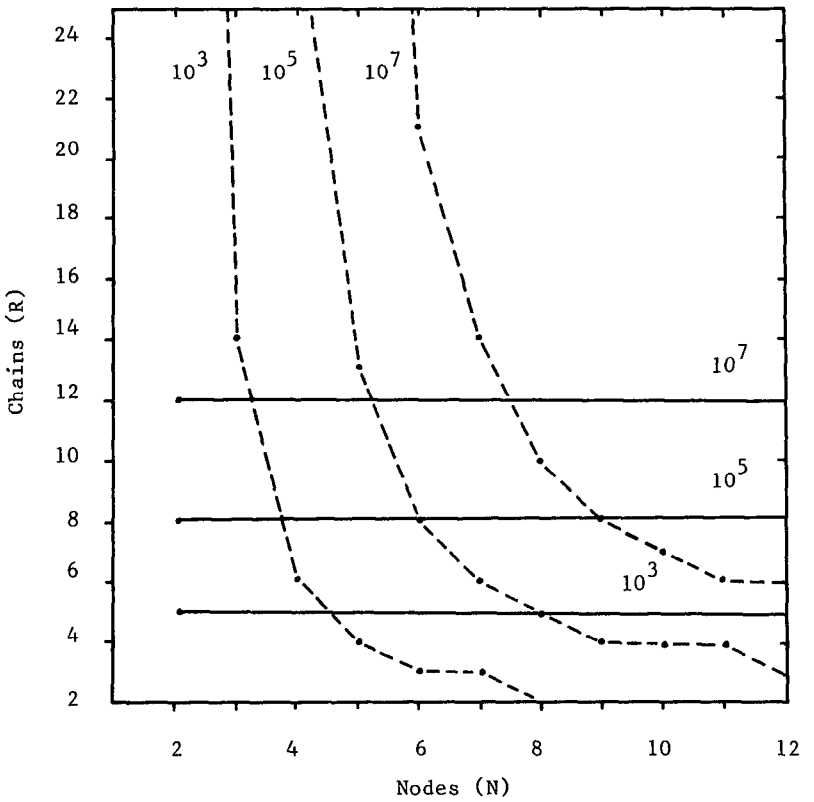

FIG. 7. Curves in the space ($N \times R$) on and above which the space requirement (number of elements) is equal to or greater than the number at the end of the curve, for $\kappa = 3$: ---, RECAL; ——, convolution.

8. *Mixed Networks and Class Switching*

RECAL can be used to analyze queuing networks having certain other more general features than those defined in Sections 2 and 7. It may be used to analyze the class of stable mixed [1] queuing networks in which there are constant-speed or limited queue-dependent servers [11, sect. 3.7]. This may be done since it is known that a mixed network can, for the purposes of analysis, be transformed into a closed network [11, sect. 3.7]. We may then obtain all the mean performance measures for the closed chains in the mixed network, by analyzing this closed network using RECAL, if in the original mixed network there are constant-speed servers. If there are limited queue-dependent servers in the mixed network, then we should use the version of RECAL that applies to networks with state-dependent servers (Section 7). The mean queue lengths can be obtained using eqs. (7.4) and (4.3). The throughputs and waiting times can be obtained using eqs. (7.5), (7.6), and (4.3), assuming that there is at least one IS center in the original mixed network. The mean queue lengths and waiting times for the open chains can be obtained readily if in the original mixed queuing network we have constant-speed servers [11, sect. 3.7]. The open-chain throughputs are obtained, as usual, from the equations of flow.

RECAL can also be used to analyze networks with customer class switching, since it is well known that the mean performance measures for the individual

classes can be obtained directly from the measures associated with the closed routing chains [11, sect. 3.5.4].

9. *Dynamic Scaling in RECAL*

It is well known that a genuine problem with the convolution algorithm is that the floating-point range of a machine may be exceeded in the course of computing the normalization constant G [8]. This problem also exists in RECAL. Lam [8] has developed a dynamic scaling procedure, which is applicable to the convolution algorithm, which alleviates the effect of numerical instability. Such scaling procedures can also be incorporated easily within RECAL. For the sake of simplicity we assume in the following that there are constant-speed servers. The same ideas, however, carry over to the state-dependent case.

Let s_r° be a scaling factor. Consider the computation of $G_k^\circ(\mathbf{v}_k)$, for all $\mathbf{v}_k \in \mathcal{S}_k^\circ$, using eq. (4.1) and the stored values of $G_{k-1}^\circ(\mathbf{v}_{k-1})$, where $\mathbf{v}_{k-1} \in \mathcal{S}_{k-1}^\circ$. According to eq. (4.1), we may write

$$(s_{r(k)}^\circ G_k^\circ(\mathbf{v}_k)) = \sum_{i=1}^{N} (1 + v_{ik}\delta_i)(s_{r(k)}^\circ w_{ir(k)}) G_{k-1}^\circ(\mathbf{v}_k + \mathbf{1}_i).$$

Hence, the scaling factor may be used as a control variable to reduce the possibility of encountering an underflow or overflow when computing $G_k^\circ(\mathbf{v}_k)$ for all $\mathbf{v}_k \in \mathcal{S}_k^\circ$.

The scaling procedure we propose is the following. We initially (statically) scale the quantities w_{ir}, where $1 \le i \le N$ and $1 \le r \le R$, so that $\min_i \{w_{ir}\} = 1$ for $1 \le r \le R$. The *dynamic* scaling to be introduced into eq. (4.1) is to scale $w_{ir(k)}$, for $1 \le i \le N$, as follows, *prior* to computing $G_k^\circ(\mathbf{v}_k)$ for all $\mathbf{v}_k \in \mathcal{S}_k^\circ$. If

$$\min_{\mathbf{v}_{k-1} \in \mathcal{S}_{k-1}^\circ} \{G_{k-1}^\circ(\mathbf{v}_{k-1})\} < 1,$$

multiply $w_{ir(k)}$, for $1 \le i \le N$, by $10^{\zeta - \alpha}$, where ζ is the smallest exponent of the machine,

$$\alpha = [\log_{10} \min_{\mathbf{v}_{k-1} \in \mathcal{S}_{k-1}^\circ} \{G_{k-1}^\circ(\mathbf{v}_{k-1})\}],$$

and $[x]$ is the integer portion of x. Otherwise, multiply $w_{ir(k)}$, for $1 \le i \le N$, by 10^{ζ} and then by $10^{-\alpha}$. After having computed $G_k^\circ(\mathbf{v}_k)$ for all $\mathbf{v}_k \in \mathcal{S}_k^\circ$, we then remultiply $w_{ir(k)}$, for $1 \le i \le N$, by $10^{-\zeta}$ and then by 10^{α} so that, once again, $\min_i \{w_{ir}\} = 1$ for $1 \le r \le R$. If we are about to commence step 4(b), then we should leave this remultiplication until after step 4(b) so that the scaling factor will properly cancel out in the computation of the mean performance measures using eqs. (4.2) and (4.3). An attractive feature of this dynamic scaling procedure is that there is no need to store any scaling factor for future use.

10. *Concluding Remarks*

In this paper we have presented RECAL, a new recursive algorithm for computing the mean performance measures of multiple-chain closed queuing networks. It is based on certain new relationships that have been derived from among the normalization constants of multiple-chain closed queuing networks and relies on the artifice of breaking down each chain into constituent subchains. We have derived the time and space requirements of the algorithm and have shown that, when the chain populations are considered equal, they are polynomial in the number of routing chains. The efficiency of RECAL, compared with that of

convolution and MVA, becomes pronounced when there are many chains in the network. RECAL therefore extends the range of queuing networks that can be analyzed efficiently by exact means.

We conclude by noting that, when there is sparsity or locality in the routing, the number of operations involved in the summation of eq. (4.1) is reduced, since the sum need only range over those centers i that customer number k may actually visit.

Appendix A

PROOF OF THEOREM 1. $G_r(\mathbf{c})$ is the normalization constant of a queuing network of type $\mathcal{N}$ with $R = r$, as described in Section 2, but with state dependent service rate functions $u_i(n) = n/(n + c_i)$ at those centers that do not have an IS discipline. Let the centers be enumerated so that centers $1, \ldots, p$ are the ones with an FCFS, LCFSPR, or PS discipline and $p + 1, \ldots, N$ are the IS centers. The aggregate system state distribution for this system with state dependent service rate functions is [1]

$$\Pr(\mathbf{n}^{(r)}) = G_r(\mathbf{c})^{-1} \prod_{i=1}^{p} \left(\left\{ \prod_{a=1}^{n_i^{(r)}} (a + c_i) \right\} \prod_{s=1}^{r} \frac{w_{is}^{n_{is}}}{n_{is}!} \right) \prod_{i=p+1}^{N} \left(\prod_{s=1}^{r} \frac{w_{is}^{n_{is}}}{n_{is}!} \right),$$

where, by definition,

$$G_r(\mathbf{c}) = \sum_{\mathbf{n}^{(r)} \in \mathcal{S}^{(r)}} \prod_{i=1}^{p} \left(\left\{ \prod_{a=1}^{n_i^{(r)}} (a + c_i) \right\} \prod_{s=1}^{r} \frac{w_{is}^{n_{is}}}{n_{is}!} \right) \prod_{i=p+1}^{N} \left(\prod_{s=1}^{r} \frac{w_{is}^{n_{is}}}{n_{is}!} \right).$$

But

$$\mathcal{S}^{(r)} = \bigcup_{\mathbf{l} \in \mathcal{L}_r} \mathcal{S}^{(r)}(\mathbf{l}),$$

where $\mathcal{L}_r$ was defined in Theorem 1 and

$$\mathcal{S}^{(r)}(\mathbf{l}) = \{\mathbf{n}^{(r)} \mid \mathbf{n}^{(r)} \in \mathcal{S}^{(r)};\ n_{ir} = l_i \text{ for } 1 \le i \le N\},$$

so that

$$G_r(\mathbf{c}) = \sum_{\mathbf{l} \in \mathcal{L}_r} \sum_{\mathbf{n}^{(r)} \in \mathcal{S}^{(r)}(\mathbf{l})} \prod_{i=1}^{p} (\cdot) \prod_{i=p+1}^{N} (\cdot).$$

Now

$$\prod_{a=1}^{n_i^{(r-1)}+l_i} (a + c_i) = (1 + c_i) \cdots (l_i + c_i)(l_i + c_i + 1) \cdots (n_i^{(r-1)} + l_i + c_i)$$

$$= \frac{(l_i + c_i)!}{c_i!} \prod_{a=1}^{n_i^{(r-1)}} (a + l_i + c_i).$$

Hence,

$$G_r(\mathbf{c}) = \sum_{\mathbf{l} \in \mathcal{L}_r} \prod_{i=1}^{p} \left((l_i + c_i)! \frac{w_{ir}^{l_i}}{c_i!\, l_i!} \right) \prod_{i=p+1}^{N} \left(\frac{w_{ir}^{l_i}}{l_i!} \right)$$

$$\times \sum_{\mathbf{n}^{(r)} \in \mathcal{S}^{(r)}(\mathbf{l})} \prod_{i=1}^{p} \left(\left(\prod_{a=1}^{n_i^{(r-1)}} (a + l_i + c_i) \right) \prod_{s=1}^{r-1} \frac{w_{is}^{n_{is}}}{n_{is}!} \right)$$

$$\times \prod_{i=p+1}^{N} \left(\prod_{s=1}^{r-1} \frac{w_{is}^{n_{is}}}{n_{is}!} \right).$$

We now see that the sum on the far right is the definition of $G_{r-1}(\mathbf{c}+\mathbf{l})$ itself, so that

$$G_r(\mathbf{c}) = \sum_{\mathbf{l}\in\mathscr{L}_r} \left(\prod_{i=1}^{p} \binom{l_i + c_i}{c_i} w_{ir}^{l_i} \prod_{i=p+1}^{N} \frac{w_{ir}^{l_i}}{l_i!} \right) G_{r-1}(\mathbf{c}+\mathbf{l}).$$

The initial conditions, $G_0(\mathbf{c})$, are normalization constants of networks that contain no customers. The only network state is the empty state, and hence $G_0(\mathbf{c}) = 1$.

The above expressions are valid for any $\mathbf{c} \in \mathscr{C}$ where $\mathscr{C} = \{\mathbf{c} \mid c_i \geq 0 \text{ for } 1 \leq i \leq N\}$. They are therefore valid for $\mathbf{c} = \mathbf{v}_r$, where $\mathbf{v}_r \in \mathscr{S}_r$, since $\mathscr{S}_r \subset \mathscr{C}$. Hence,

$$G_r(\mathbf{v}_r) = \sum_{\mathbf{l}\in\mathscr{L}_r} \left(\prod_{i=1}^{N} h_{ir}(\mathbf{v}_r, \mathbf{l}) \right) G_{r-1}(\mathbf{v}_r + \mathbf{l}),$$

where $h_{ir}(\mathbf{v}_r, \mathbf{l})$ was defined in Theorem 1.

It now remains to be shown that the domain of $G_r(\mathbf{v}_r)$, in the computation of $G_R(\mathbf{0})$, is

$$\mathscr{S}_r = \begin{cases} \left\{ \mathbf{v}_r \mid v_{ir} \geq 0 \text{ for } 1 \leq i \leq N; \sum_{i=1}^{N} v_{ir} = \sum_{s=r+1}^{R} K_s \right\}, & \text{for } 0 \leq r \leq R-1, \\ \{\mathbf{0}\}, & \text{for } r = R. \end{cases} \tag{A1}$$

We show this by induction on r.

Clearly, $\mathscr{S}_R = \{\mathbf{0}\}$ since we only wish to determine $G_R(\mathbf{v}_R)$ for $\mathbf{v}_R = \mathbf{0}$. Now

$$\mathscr{S}_{R-1} = \bigcup_{\mathbf{v}_R\in\mathscr{S}_R} \left\{ \bigcup_{\mathbf{l}\in\mathscr{L}_R} \{\mathbf{v}_R + \mathbf{l}\} \right\} = \bigcup_{\mathbf{l}\in\mathscr{L}_R} \{\mathbf{l}\} = \mathscr{L}_R$$

so that eq. (A1) is true for $r = R - 1$.

We now assume that eq. (A1) is true for $r = t$ and show that it is true for $r = t - 1$. We have

$$\mathscr{S}_{t-1} = \bigcup_{\mathbf{v}_t\in\mathscr{S}_t} \left\{ \bigcup_{\mathbf{l}\in\mathscr{L}_t} \{\mathbf{v}_t + \mathbf{l}\} \right\} = \bigcup_{\mathbf{l}\in\mathscr{L}_t} \left\{ \bigcup_{\mathbf{v}_t\in\mathscr{S}_t} \{\mathbf{v}_t + \mathbf{l}\} \right\}$$

$$= \bigcup_{\mathbf{l}\in\mathscr{L}_t} \left\{ (\mathbf{v}_t + \mathbf{l}) \mid v_{it} \geq 0 \text{ for } 1 \leq i \leq N; \sum_{i=1}^{N} v_{it} = \sum_{s=t+1}^{R} K_s \right\}.$$

Now $\mathbf{l} \in \mathscr{L}_t$ so that $l_i \geq 0$ for $1 \leq i \leq N$ and $\sum_{i=1}^{N} l_i = K_t$. Hence,

$$\mathscr{S}_{t-1} = \left\{ (\mathbf{v}_t + \mathbf{l}) \mid v_{it} \geq 0 \text{ for } 1 \leq i \leq N; l_i \geq 0 \text{ for } 1 \leq i \leq N; \sum_{i=1}^{N} v_{it} = \sum_{s=t+1}^{R} K_s; \sum_{i=1}^{N} l_i = K_t \right\}$$

$$= \left\{ (\mathbf{v}_t + \mathbf{l}) \mid (v_{it} + l_i) \geq 0 \text{ for } 1 \leq i \leq N; \sum_{i=1}^{N} (v_{it} + l_i) = \sum_{s=t}^{R} K_s \right\}.$$

Defining a new variable $\mathbf{v}_{t-1} = (\mathbf{v}_t + \mathbf{l})$, we see that this agrees in form with eq. (A1). □

Appendix B

PROOF OF THEOREM 2. (We use some notation and definitions found in Appendix A.) By definition

$$\Pr(\mathbf{n}_R = \mathbf{k}) = \sum_{\mathbf{n}^{(R)} \in \mathscr{S}^{(R)}(\mathbf{k})} \Pr(\mathbf{n}^{(R)}).$$

Now, using eq. (2.1), we may write

$$\begin{aligned}
&\Pr(n_{1R} = k_1, \ldots, n_{NR} = k_N) \\
&= G^{-1} \prod_{i=1}^{N} \frac{w_{iR}^{k_i}}{k_i!} \sum_{\mathbf{n}^{(R)} \in \mathscr{S}^{(R)}(\mathbf{k})} \prod_{i=1}^{p} \left((\mathrm{n}_i^{(R-1)} + \mathrm{k}_i)! \prod_{r=1}^{R-1} \frac{w_{ir}^{n_{ir}}}{n_{ir}!} \right) \prod_{i=p+1}^{N} \left(\prod_{r=1}^{R-1} \frac{w_{ir}^{n_{ir}}}{n_{ir}!} \right) \\
&= G^{-1} \prod_{i=1}^{p} w_{iR}^{k_i} \prod_{i=p+1}^{N} \frac{w_{iR}^{k_i}}{k_i!} \sum_{\mathbf{n}^{(R)} \in \mathscr{S}^{(R)}(\mathbf{k})} \prod_{i=1}^{p} \left(\left\{ \prod_{a=1}^{n_i^{(R-1)}} (a + k_i) \right\} \prod_{r=1}^{R-1} \frac{w_{ir}^{n_{ir}}}{n_{ir}!} \right) \prod_{i=p+1}^{N} \left(\prod_{r=1}^{R-1} \frac{w_{ir}^{n_{ir}}}{n_{ir}!} \right).
\end{aligned}$$

From the definition of $G_r(\mathbf{c})$ in Appendix A we see that the sum on the far right is the definition of $G_{R-1}(\mathbf{k})$ itself. Furthermore, $G = G_R(\mathbf{0})$, so that

$$\Pr(n_{1R} = k_1, \ldots, n_{NR} = k_N) = G_R(\mathbf{0})^{-1} G_{R-1}(\mathbf{k}) \prod_{i=1}^{p} w_{iR}^{k_i} \prod_{i=p+1}^{N} \frac{w_{iR}^{k_i}}{k_i!}. \qquad \square$$

Appendix C

PROOF OF THEOREM 3a. When there are no IS centers in the network, using the definition of $G_r(\mathbf{c})$ in Appendix A, we may write

$$G_{R-1}(\mathbf{1}_l) = \sum_{\mathbf{n}^{(R-1)} \in \mathscr{S}^{(R-1)}} (n_l^{(R-1)} + 1) \prod_{i=1}^{N} \left(n_i^{(R-1)}! \left(\prod_{r=1}^{R-1} \frac{w_{ir}^{n_{ir}}}{n_{ir}!} \right) \right).$$

Therefore

$$G_{R-1}(\mathbf{1}_l) = G_{R-1}(\mathbf{0}) E(n_l^{(R-1)}) + G_{R-1}(\mathbf{0}),$$

where $E(\cdot)$ denotes expectation. Hence, when $K_R = 1$,

$$\sum_{l=1}^{N} G_{R-1}(\mathbf{1}_l) = G_{R-1}(\mathbf{0})(K - 1) + N G_{R-1}(\mathbf{0}),$$

since

$$\sum_{l=1}^{N} E(n_l^{(R-1)}) = \sum_{r=1}^{R-1} K_r = K - 1. \qquad \square$$

PROOF OF THEOREM 3b. Consider the definition of $G_r(\mathbf{c})$ in Appendix A. When $\mathbf{c} = \mathbf{1}_x$ and x is an IS center, we have $c_i = 0$ for all $1 \le i \le p$, in which case $G_r(\mathbf{1}_x) = G_r(\mathbf{0})$. Hence $G_{R-1}(\mathbf{0}) = G_{R-1}(\mathbf{1}_x)$. $\square$

Appendix D

We need to show that the value of $G_{k-1}^{\circ}(\mathbf{v}_{k-1})$ at location $M_k(\mathbf{v}_k)$ is not required in the computation of $G_k^{\circ}(\boldsymbol{\gamma})$, where $\boldsymbol{\gamma}$ is any vector such that $\boldsymbol{\gamma} \in \mathscr{S}_k^{\circ}$ and $M_k(\boldsymbol{\gamma}) > M_k(\mathbf{v}_k)$.

PROOF. As can be seen from eq. (4.1), the computation of $G_k^\circ(\gamma)$ requires the values of $G_{k-1}^\circ(\gamma + \mathbf{1}_i)$ for $1 \le i \le N$. We need to show that the value of $G_{k-1}^\circ(\mathbf{v}_{k-1})$ at location $M_k(\mathbf{v}_k)$ is not one of these. Hence, we need to show that $M_{k-1}(\gamma + \mathbf{1}_i) > M_k(\mathbf{v}_k)$ for any $1 \le i \le N$ and γ, such that $\gamma \in \mathcal{S}_k^\circ$ and $M_k(\gamma) > M_k(\mathbf{v}_k)$.

We assume that $M_{k-1}(\gamma + \mathbf{1}_i) \le M_k(\mathbf{v}_k)$ and show that this leads to a contradiction.

By the definition of γ, $M_k(\gamma) > M_k(\mathbf{v}_k)$ so that $M_{k-1}(\gamma + \mathbf{1}_i) < M_k(\gamma)$. Therefore, using eq. (5.5), we have

$$(K^\circ - k + 1)^{i-1} + \sum_{j=1}^{N} \gamma_j (K^\circ - k + 1)^{j-1} < \sum_{j=1}^{N} \gamma_j (K^\circ - k)^{j-1} < \sum_{j=1}^{N} \gamma_j (K^\circ - k + 1)^{j-1}.$$

Hence $(K^\circ - k + 1)^{i-1} < 1$. Now $1 \le k \le K^\circ$ and $1 \le i \le N$, so there is a contradiction. □

Appendix E

Suppose, for the sake of simplicity, that $K_r = \kappa$ for $1 \le r \le R$. When we break down the chains in $\mathcal{N}$, so that each subchain consists of one customer, the number of operations to arrive at G° using the convolution algorithm is, according to eq. (6.1), $2R\kappa(N-1)2^{R\kappa}$. Now

$$\frac{2R\kappa(N-1)2^{R\kappa}}{2R(N-1)(\kappa+1)^R} = \frac{\kappa(2^\kappa)^R}{(\kappa+1)^R} \ge \kappa \ge 1,$$

since $2^\kappa \ge \kappa + 1$. Hence no advantage is obtained in the number of operations.

The storage space required in the convolution algorithm to obtain G° is, using eq. (6.2), $2^{R\kappa+1}$. Now

$$\frac{2^{R\kappa+1}}{2(\kappa+1)^R} = \frac{(2^\kappa)^R}{(\kappa+1)^R} \ge 1.$$

Hence, no advantage in the storage space is obtained either.

ACKNOWLEDGMENTS. The authors are grateful to the anonymous referees and Dr. D. Potier (France) for several useful suggestions.

REFERENCES

1. BASKETT, F., CHANDY, K.M., MUNTZ, R.R., AND PALACIOS, F.G. Open, closed, and mixed networks of queues with different classes of customers. *J. ACM 22*, 2 (Apr. 1975), 248–260.
2. BRUELL, S.C., AND BALBO, G. *Computational Algorithms for Closed Queueing Networks.* Elsevier-North Holland, New York, 1980.
3. BUZEN, J.P. Computational algorithms for closed queueing networks with exponential servers. *Commun. ACM 16*, 9 (Sept. 1973), 527–531.
4. CHANDY, K.M., AND NEUSE, D. Linearizer: A heuristic algorithm for queueing network models of computer systems. *Commun. ACM 25*, 2 (Feb. 1982), 126–133.
5. CHANDY, K.M., AND SAUER, C.H. Computational algorithms for product form queueing networks. *Commun. ACM 23*, 10 (Oct. 1980), 573–583.
6. CONWAY, A.E., AND GEORGANAS, N.D. A new method for computing the normalization constant of multiple chain queueing networks. *INFOR 24*, 3 (Aug. 1986).
7. COURTOIS, P.J. *Decomposability: Queueing and Computer System Applications.* Academic Press, Orlando, Fla., 1977.
8. LAM, S.S. Dynamic scaling and growth behavior of queuing network normalization constants. *J. ACM 29*, 2 (Apr. 1982), 492–513.
9. LAM, S.S., AND LIEN, Y.L. A tree convoluted algorithm for the solution of queueing networks. *Commun. ACM 26*, 3 (Mar. 1983), 203–215.

10. LAM, S.S., AND WONG, J.W. Queueing network models of packet switching networks. Part 2: Networks with population size constraints. *Perform. Eval. 2*, 3 (1982), 161–180.
11. LAVENBERG, S.S., ED. *Computer Performance Modeling Handbook.* Academic Press, Orlando, Fla. 1983.
12. LAVENBERG, S.S., AND REISER, M. Stationary state probabilities at arrival instants for closed queueing networks with multiple types of customers. *J. Appl. Prob. 17* (Dec. 1980), 1048–1061.
13. LITTLE, J.D.C. A proof of the queueing formula $L = \lambda W$. *Oper. Res.* 9 (1961), 383–387.
14. MCKENNA, J., AND MITRA, D. Integral representations and asymptotic expansions for closed Markovian queueing networks: Normal usage. *Bell Syst. Tech. J. 61*, 5 (May–June 1982), 661–683.
15. MCKENNA, J., AND MITRA, D. Asymptotic expansions and integral representations of moments of queue lengths in closed Markovian networks. *J. ACM 31*, 2 (Apr. 1984), 346–360.
16. NEUSE, D.M. Approximate analysis of large and general queueing networks. Ph.D. dissertation, Univ. of Texas, Austin, 1982.
17. REISER, M., AND KOBAYASHI, H. Queueing networks with multiple closed chains: Theory and computational algorithms. *IBM J. Res. Dev., 19* (May 1975), 283–294.
18. REISER, M., AND LAVENBERG, S.S. Mean-value analysis of closed multichain queuing networks. *J. ACM 27*, 2 (Apr. 1980), 313–322.
19. SAUER, C.H., AND CHANDY, K.M. *Computer Systems Performance modeling.* Prentice-Hall, Englewood Cliffs, N.J., 1981.

RECEIVED NOVEMBER 1984; REVISED OCTOBER 1985; ACCEPTED OCTOBER 1985