RECAL a new efficient algorithm for the exact analysis of multiple-chain closed queueing networks

A.E. Conway, N. D. Georganas

INRIA

CENTRE DE ROCQUENCOURT

Institut National
de Recherche
en Informatique
et en Automatique

Domaine de Voluceau
Rocquencourt
B.P.105
78153 Le Chesnay Cedex
France
Tel. (3) 954 90 20

Rapports de Recherche

N° 373

RECAL
A NEW EFFICIENT ALGORITHM FOR THE EXACT ANALYSIS OF MULTIPLE-CHAIN CLOSED QUEUEING NETWORKS

A. E. CONWAY
N. D. GEORGANAS

Mars 1985

RECAL- A NEW EFFICIENT ALGORITHM FOR THE EXACT ANALYSIS OF MULTIPLE-CHAIN CLOSED QUEUEING NETWORKS

*A. E. Conway and N.D. Georganas**
University of Ottawa
Ottawa, Ontario
Canada

Résumé : RECAL ("REcursion by Chain ALgoirthm").

Un aglorithme pour l'évaluation des mesures de performance de réseaux de files d'attente à plusieurs chaînes fermées est présenté. L'algorithme est basé sur une nouvelle expression récursive qui relie la constante de normalisation d'un réseau avec n chaînes de routage fermées à celle d'un ensemble de réseaux ayant (n-1) chaînes. Une simplification est obtenue en décomposant chaque chaîne en un ensemble de sous-chaînes, chacune ayant un client. L'algorithme a une complexité polynomiale en nombre de chaînes, en ce qui concerne le temps d'exécution et la mémoire exigée. Quand le réseau contient plusieurs chaînes l'algorithmes est plus efficace que celui de convolution, et offre donc une extension utile pour la catégorie des réseaux analysables de façon exacte. Un exemple numérique est donné.

* *The authors are currently on leave at INRIA, Le Chesnay, and the Advanced Studies Department, Bull-Transac, Massy, France*

RECAL - A NEW EFFICIENT ALGORITHM FOR THE EXACT ANALYSIS OF MULTIPLE-CHAIN CLOSED QUEUEING NETWORKS

A.E. Conway and N.D. Georganas[†]

University of Ottawa

Ottawa, Ontario,

Canada

ABSTRACT - RECAL, a Recursion by Chain Algorithm for computing the mean performance measures of product-form multiple-chain closed queueing networks, is presented. It is based on a new recursive expression which 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 sub-chains 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 queueing networks which can be efficiently analyzed by exact means. A numerical example is given.

[†]The authors are currently on leave at INRIA, Le Chesnay, and the Advanced Studies Department, BULL-Transac, Massy, France.

1. INTRODUCTION

Multiple-chain closed queueing networks, which have a product-form (1) state probability distribution, are widely used models of, for example, computer systems (9), (17) and computer communication networks (8). 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.

There now exist several efficient algorithms. The most well-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 (15) 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 (16) which is based on the so-called Arrival Theorem (10). 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 algorithms which exist, such as LBANC and CCNC (5), (9), (17) 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 queueing networks with many chains requires the use of specialized exact algorithms, analytical approximation techniques or heuristic methods. The tree convolution algorithm (7) 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 (8). A powerful analytical approximation technique for networks of a large size is discussed in (12) and (13). Heuristic methods are presented in (4), (9), (14) and (16).

In this paper we present RECAL, a Recursion by Chain Algorithm for the exact analysis of product-form multiple-chain closed queueing networks. It is based on a new recursive expression which 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. The new recursion, for obtaining the normalization constant of multiple-chain closed queueing networks, has a time and space growth which is different from that of the recursions used in the convolution and MVA algorithms. We introduce the artifice of breaking down each chain into sub-chains 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 queueing 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 queueing networks which can be efficiently analyzed 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 will 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 queueing networks to be considered here. 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. The paper concludes with an example of a network which has been analysed using RECAL.

2. THE QUEUEING NETWORK

We are concerned here with the analysis of multiple-chain closed queueing networks of the product-form type which have been considered in (1). There are N service centers and R closed routing chains. We will only be concerned with cases where 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 one, 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 $1\leq r\leq R$. The population of chain r is K_r. We denote the queueing network described above as N.

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

$$S^{(R)}=\{\ \underline{n}^{(R)}\ |\ n_{ir}\geq 0 \text{ for } 1\leq i\leq N,\ 1\leq r\leq R;\ \sum_{i=1}^{N} n_{ir}=K_r \text{ for } 1\leq r\leq R\ \}.$$

The marginal state probability distribution is (1), for $\underline{n}^{(R)} \in S^{(R)}$,

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

where

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

$n_i^{(R)} = \sum_{r=1}^{R} n_{ir}$, $w_{ir} = t_{ir} e_{ir}$ and the normalization constant is, by definition,

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

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

3. PRELIMINARY RESULTS

In this Section we present certain new relationships among the normalization constants of multiple-chain closed queueing networks and new expressions for the mean performance measures in terms of these normalization constants. They will provide the mathematical basis for the algorithm to be developed in the following Section.

Consider a multiple-chain closed queueing network, like N, but

with state-dependent servers at those service centers which 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 $G_R(\underline{c})$ where $\underline{c} = (c_1,\ldots,c_N)$. An explicit expression for $G_R(\underline{c})$ is given in Appendix 1. Clearly $G_R(\underline{0}) = G$ since, if $\underline{c} = \underline{0}$, then $u_i(n) = 1$. We have the following results.

Theorem 1 :

$G_R(\underline{0})$ is given recursively by

$$G_r(\underline{v}_r) = \sum_{\underline{\ell} \in L_r} g_r(\underline{v}_r,\underline{\ell}) G_{r-1}(\underline{v}_r+\underline{\ell}), \quad \text{for } 1\le r\le R,\ \underline{v}_r \in I_r,$$

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

$\underline{\ell} = (\ell_1,\ldots,\ell_N)$,

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

$$L_r = \{\, \underline{\ell} \mid \ell_i \ge 0 \text{ for } 1\le i\le N;\ \sum_{i=1}^{N} \ell_i = K_r \,\},$$

$$g_r(\underline{v}_r,\underline{\ell}) = \prod_{i=1}^{N} h_{ir}(v_{ir},\ell_i),$$

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

and the initial conditions are $G_0(\underline{v}_0) = 1$ for all $\underline{v}_0 \in I_0$.

Proof : The proof is contained in Appendix 1.

Theorem 1 gives a new recursive expression for computing the normalization constant $G = G_R(\underline{0})$. In effect, the normalization constant $G_r(\underline{v}_r)$ 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 to compute the mean performance measures, however, we will not use Theorem 1 but rather a simplified version of it (Corollary 1 below) which results when the population of all chains is one.

Corollary 1 : If $K_r = 1$ for $1 \leq r \leq R$, then $G_R(\underline{0})$ is given recursively by

$$G_r(\underline{v}_r) = \sum_{i=1}^{N} (1+v_{ir}\delta_i) w_{ir} G_{r-1}(\underline{v}_r + \underline{1}_i), \quad \text{for } 1 \leq r \leq R, \ \underline{v}_r \in I_r,$$

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

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

and the initial conditions are $G_0(\underline{v}_0) = 1$ for all $\underline{v}_0 \in I_0$, where now

$$I_0 = \{ \underline{v}_0 \mid v_{i0} \geq 0 \text{ for } 1 \leq i \leq N; \sum_{i=1}^{N} v_{i0} = R \}.$$

Proof : The Corollary follows directly from Theorem 1 when $K_r = 1$ for $1 \leq r \leq R$.

Theorem 2 :

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

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

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

and

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

Proof : The proof is contained in Appendix 2.

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

$$G_{R-1}(\underline{0}) = \sum_{\ell=1}^{N} G_{R-1}(\underline{1}_\ell) / (N+K-1)$$

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

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

Proof : The proofs are contained in Appendix 3.

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 (queueing+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

$$T_{iR} = \begin{cases} e_{iR} \sum_{\ell=1}^{N} G_{R-1}(\underline{1}_\ell) \; / \; G_R(\underline{0})(N+K-1), & \text{if there are no IS centers in the network,} \\ e_{iR} G_{R-1}(\underline{1}_x) \; / \; G_R(\underline{0}), & \text{if there are IS centers and x is any one of them,} \end{cases}$$

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

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

and $$W_{iR} = Q_{iR} \; / \; T_{iR}.$$

Proof : It is well-known (2), (9), (17) that, in terms of the notation adopted here, when $K_R = 1$, $T_{iR} = e_{iR} G_{R-1}(\underline{0}) / G_R(\underline{0})$. 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 (11).

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 sub-chains, 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 the recursion of Corollary 1 can be applied. 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 alters in no way the physical characteristics of the original network.

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

$$G_k^\circ(\underline{v}_k) = \sum_{i=1}^{N} (1+v_{ik}\delta_i) w_{ir(k)} G^\circ{}_{k-1}(\underline{v}_k + \underline{1}_i) \qquad (4.1)$$

for $1 \leq k \leq K^\circ$, $\underline{v}_k \in I_k^\circ$, where $I_k^\circ = \{ \underline{v}_k \mid v_{ik} \geq 0$ for $1 \leq i \leq N$; $\sum_{i=1}^{N} v_{ik} = K^\circ - k \}$. The initial conditions are $G_0^\circ(\underline{v}_0) = 1$ for all $\underline{v}_0 \in I_0^\circ$.

Hence, by breaking down each chain in N into constituent sub-chains we can apply Corollary 1 and circumvent, in particular, the need to compute the terms $g_r(\underline{v}_r, \ell)$ which appear in the expression of Theorem 1. We note that, in general, $G_{R^\circ}^\circ(\underline{0}) \neq G_R(\underline{0})$ since, in general, $S^{\circ(R^\circ)} \neq S^{(R)}$. Hence, we are no longer able to compute the normalization

constant $G = G_R(\underline{0})$ which is supposed to be of central interest. Nevertheless, using Corollary 2, we may write

$$T^\circ_{iR^\circ} = \begin{cases} e_{ir(R^\circ)} \sum_{\ell=1}^{N} G^\circ_{R^\circ-1}(\underline{1}_\ell) \;/\; G^\circ_{R^\circ}(\underline{0})(N+K^\circ-1), & \text{if there are no IS centers in the network,} \\ e_{ir(R^\circ)} G^\circ_{R^\circ-1}(\underline{1}_x) \;/\; G^\circ_{R^\circ}(\underline{0}), & \text{if there are IS centers and x is any one of them,} \end{cases}$$

$$U^\circ_{iR^\circ} = t_{ir(R^\circ)} T^\circ_{iR^\circ},$$

$$Q^\circ_{iR^\circ} = G^\circ_{R^\circ}(\underline{0})^{-1} G^\circ_{R^\circ-1}(\underline{1}_i) w_{ir(R^\circ)}$$

and $$W^\circ_{iR^\circ} = Q^\circ_{iR^\circ} \;/\; T^\circ_{iR^\circ}. \qquad (4.2)$$

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

$$T_{ir(R^\circ)} = K_{r(R^\circ)} T^\circ_{iR^\circ},$$
$$U_{ir(R^\circ)} = K_{r(R^\circ)} U^\circ_{iR^\circ},$$
$$Q_{ir(R^\circ)} = K_{r(R^\circ)} Q^\circ_{iR^\circ}$$
and $$W_{ir(R^\circ)} = W^\circ_{iR^\circ}. \qquad (4.3)$$

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 N° so that $r(R^\circ)$ is changed to another chain for which one wishes to compute performance measures using eqs. 4.1, 4.2 and 4.3. More specifically, suppose that the customers in N° are initially enumerated so that the initial assignment of the customers in N° to chains in N is $r^{(1)}(k)$ as illustrated in Table 1 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 $G^\circ_{R^\circ}(\underline{0})$ using eq. 4.1 we may compute the mean performance measures for chain 1 in N using eqs. 4.2 and 4.3. We then reenumerate the customers in N° to obtain a new assignment of the customers in N° to chains in N. Let the new assignment be $r^{(2)}(k)$ as illustrated in Table 1 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 N may then be obtained after having recomputed $G^\circ_{R^\circ}(\underline{0})$. In general then, after having computed the performance measures for chain s in N we reenumerate the customers in N°

so that

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

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

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

so that in the determination of the performance measures for chain (s+1) in 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 which has been developed.

RECAL : A recursion by chain algorithm for computing mean performance measures.

Step 1 : Initialize $G^\circ_0(\underline{v}_0) = 1$ for all $\underline{v}_0 \in I^\circ_0$.

Step 2 : Enumerate the customers in N° so that the assignment of customers in N° to chains in N is $r^{(1)}(k)$, given by eq. 4.4, for $1 \le k \le K^\circ$.

Step 3 : Compute and store $G^\circ_x(\underline{v}_x)$ for all $\underline{v}_x \in I^\circ_x$, where $x=K^\circ-R$, using eq. 4.1.

Step 4 : For each chain s in N, $1 \le s \le R$:

Step 4a : Determine $G^\circ_{K^\circ}(\underline{0})$ using eq. 4.1 and the stored values of $G^\circ_x(\underline{v}_x)$, $\underline{v}_x \in I^\circ_x$.

Step 4b : Compute the mean performance measures for chain s using $G^{\circ}_{K^{\circ}}(\underline{0})$, $G^{\circ}_{K^{\circ}-1}(\underline{1}_i)$ for $1 \leq i \leq N$, eqs. 4.2 and 4.3. Stop if s=R.

Step 4c : Reenumerate the customers in N° so that the assignment of customers in N° to chains in N is $r^{s+1}(k)$, as defined by eq. 4.5.

Step 4d : Increment x by 1.

Step 4e : Compute and store $G^{\circ}_{x}(\underline{v}_x)$ for all $\underline{v}_x \in I^{\circ}_{x}$ using eq. 4.1 and the stored values of $G^{\circ}_{x-1}(\underline{v}_{x-1})$, $\underline{v}_{x-1} \in I^{\circ}_{x-1}$.

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 $G^{\circ}_{1}(\underline{v}_1)$ for all $\underline{v}_1 \in I^{\circ}_{1}$ is

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

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

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

With s=1, the number of operations to carry out Step 4a is

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

We will ignore the number of operations in Step 4b since it is relatively small.

With s=1, the number of operations to carry out Step 4e is

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

In general, the number of operations required in Step 4a is

$$\sum_{i=K^\circ-R+s}^{K^\circ} (4N-1)\binom{K^\circ-i+N-1}{N-1} \qquad (5.2)$$

and the number of operations involved in Step 4e is

$$(4N-1)\binom{R-s+N-1}{N-1} . \qquad (5.3)$$

The total number of operations required by the algorithm is, using eqs. 5.1, 5.2 and 5.3,

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

which simplifies to

$$(4N-1)\left(\binom{K^\circ+N-1}{N} + \binom{R+N-1}{N+1} + \binom{R+N-1}{N} - 1 \right) . \qquad (5.4)$$

Now suppose that $K_r = \kappa$ for $1 \leq r \leq R$. Then $K^\circ = \kappa R$. If we consider κ and N fixed, then eq. 5.4 is a polynomial in R of degree (N+1). When R is large, eq. 5.4 is of the order of

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

If we now consider R and κ fixed, then eq. 5.4 is a polynomial in N of degree (κR). When N is large, eq. 5.4 is of the order of

$$\begin{cases} 8N^R/(R-1)! \ , & \text{if } \kappa=1, \\ 4N^{\kappa R}/(\kappa R-1)! \ , & \text{if } \kappa>1. \end{cases}$$

5.2 Space requirement

The space requirement is dictated by the maximum storage space which 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 $G_k^\circ(\underline{v}_k)$ for all $\underline{v}_k \in I_k^\circ$ before incrementing k.

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

$$\sum_{i=1}^{N} d_i (K^\circ - x)^{i-1}. \qquad (5.5)$$

Such a mapping has been introduced in (6,Section 4.2). The domain of $M_x(\underline{d})$ is I°_x and the range is $\{1,2,\ldots,\binom{K_0-x+N-1}{N-1}\}$; the combinatorial term being the cardinality of the set I°_x. An example mapping is given in Table 2.

If we now compute the values of $G^\circ_k(\underline{v}_k)$, $\underline{v}_k \in I^\circ_k$, in the order which 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 $G^\circ_k(\underline{v}_k)$ using eq. 4.1 we may store the result at location $M_k(\underline{v}_k)$ of the *same* array which was used to store $G^\circ_{k-1}(\underline{v}_{k-1})$ for all $\underline{v}_{k-1} \in I^\circ_{k-1}$. We illustrate this mechanism in Figure 1. This storage procedure may be adopted since the value of $G^\circ_{k-1}(\underline{v}_{k-1})$ at location $M_k(\underline{v}_k)$ is never required, after $G^\circ_k(\underline{v}_k)$ has been computed, for any of the computations of $G^\circ_k(\underline{\gamma})$, where $\underline{\gamma}$ is any vector such that $\underline{\gamma} \in I^\circ_k$ and $M_k(\underline{\gamma}) > M_k(\underline{v}_k)$. This point is proved in Appendix 4. As a result, the required storage space to implement eq. 4.1 is only the cardinality of I°_0 and not the sum of the cardinalities of I°_0 and I°_1. The cardinality of I°_0 is $\binom{K^\circ+N-1}{N-1}$. In addition, Step 3 requires a storage space equal to the cardinality of $I^\circ_{K^\circ-R}$ which is $\binom{R+N-1}{N-1}$. The values obtained in Step 4e may be stored in the same storage space used for Step 3. The total required storage space for the algorithm is then

$$\binom{K^\circ+N-1}{N-1} + \binom{R+N-1}{N-1} \qquad (5.6)$$

With $K_r = \kappa$ for $1 \leq r \leq R$ and with N and κ considered fixed, eq. 5.6 is a polynomial in R of degree (N-1). When R is large, eq. 5.6 is of the order of

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

If we now consider R and κ fixed, then eq. 5.6 is a polynomial in N of degree (κR). When N is large, eq. 5.6 is of the order of

$$\begin{cases} 2N^R/R! \text{ , if } \kappa=1, \\ N^{\kappa R}/(\kappa R)! \text{ , if } \kappa>1. \end{cases}$$

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) which applies to networks with constant speed servers, as is the case here, the total number of operations to arrive at G is

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

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

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

The MVA algorithm requires approximately the same time requirement as eq. 6.1 (16) and the required storage space is (2)

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

We show in Appendix 5 that no computational advantage can be obtained in the convolution or MVA algorithms by breaking down the chains in N into sub-chains.

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. 5.4 and 5.6, respectively, will be less than those given by eq. 6.1 and 6.2 or 6.3.

In Figure 2 we compare, for the purposes of illustration, the number of operations given by eq. 5.4 with eq. 6.1 in the case where N = 4 and $K_r = \kappa$ for $1 \leq r \leq R$, for various values of κ and R. In Figure 3 we compare the storage requirements given by eq. 5.6 with eq. 6.2 in the case where N = 4. In Figure 4 we determine, for various values of κ, the region in the space of queueing networks ($N \times R$) in which the number of operations required by RECAL (eq. 5.4) 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. 5.6) is less than that required with convolution (eq. 6.2). In Figures 6 and 7 we give, respectively, several iso-time and iso-storage curves in the space ($N \times R$) for RECAL and convolution in the case where $\kappa = 3$. 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. 5.6, is linear in R.

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), (9), (17).

Consider a queueing network of type N, but with state-dependent servers at the service centers. Let the service rate function for center i depend on the total number of customers at center i and be denoted by $\beta_i(n)$. At IS centers, with the definition of $f_i(\underline{n}_i^{(R)})$ that has been adopted in Section 2, we have $\beta_i(n)=1$. With state-dependent servers, the marginal state distribution for the network is (1), for $\underline{n}^{(R)} \in S^{(R)}$,

$$\Pr(\underline{n}^{(R)}) = G^{-1} \prod_{i=1}^{N} \left(f_i(\underline{n}_i^{(R)}) \Big/ \prod_{a=1}^{n_i^{(R)}} \beta_i(a) \right). \qquad (7.1)$$

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 1), we may show that $G_R(\underline{0})$ is given by the same recursive formula which appears in Theorem 1 but with

$$h_{ir}(\underline{v}_r,\underline{\ell}) = \binom{\ell_i+v_{ir}}{v_{ir}} w_{ir}^{\ell_i} \Big/ \prod_{b=1}^{\ell_i} \beta_i(b+v_{ir}), \qquad (7.2)$$

if center i is FCFS, LCFSPR or PS. The quantity $G_r(\underline{v}_r)$, in Theorem 1, is now

the normalization constant for a network of type N but with servers who have state-dependent service rate functions $u_i(n)=\beta_i(n+v_{ir})n/(n+v_{ir})$.

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

$$G^\circ_k(\underline{v}_k) = \sum_{i=1}^{N} (1+v_{ik}\delta_i)w_{ir(k)}G^\circ_{k-1}(\underline{v}_k+\underline{1}_i)/\beta_i(1+v_{ik}) \quad . \tag{7.3}$$

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

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

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

$$Q^\circ_{iR^\circ} = G^\circ_{R^\circ}(\underline{0})^{-1}G^\circ_{R^\circ-1}(\underline{1}_i)w_{ir(R^\circ)}/\beta_i(1) \quad .$$

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

$$T^\circ_{iR^\circ} = e_{ir(R^\circ)}G^\circ_{R^\circ-1}(\underline{1}_x)/G^\circ_{R^\circ}(\underline{0})$$

where x is any one of the IS centers. This expression for the throughput follows from the identity $G_{R-1}(\underline{0})=G_{R-1}(\underline{1}_x)$ and the result that, when $K_R=1$, $T_{iR} = e_{iR}G_{R-1}(\underline{0})/G_R(\underline{0})$ which is known to hold *even* in the case of state-dependency (2,page 71). We also have, using Little's result (11),

$$W^{\circ}_{iR^{\circ}} = Q^{\circ}_{iR^{\circ}}/T^{\circ}_{iR^{\circ}} \cdot$$

If we assume that there are no IS centers 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-1) operations. The total number of operations required by the algorithm is then, using eq. 5.4,

$$(5N-1)(\binom{K^{\circ}+N-1}{N} + \binom{R+N-1}{N+1} + \binom{R+N-1}{N} - 1)$$

When there are IS centers 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. 5.6 except, of course, for the additional space required to store the network parameters $\beta_i(n)$, which we will ignore.

8. EXAMPLE ANALYSIS USING RECAL

RECAL has been programmed in C-language and run on the VAX-11/780 operating under UNIX IS-3. We present the analysis of a queueing network with 4 service centers (N=4) and 10 closed routing chains (R=10). The population of each chain is 3 (K=3). Service center 4 is an IS center. The parameters of the network and the mean performance measures, obtained using RECAL, are given in Table 3. (We simply specify the visit ratio vectors rather than the routing matrices.) The required storage space for RECAL is 5742 elements (see Figure 3). The convolution algorithm would require, according to eq. 6.2, 2.1×10^{6} elements. The theoretical number of operations for RECAL and convolution are 6.4×10^{5} (see Figure 2) and 6.3×10^{7} (using eq. 6.1), respectively.

9. CONCLUDING REMARKS

In this paper we have presented RECAL, a new recursive algorithm for computing the mean performance measures of multiple-chain closed queueing networks. It is based on certain new relationships which have been derived among the normalization constants of multiple-chain closed queueing networks and relies on the artifice of breaking down each chain into constituent sub-chains. We have derived the time and space requirements of the algorithm and shown that, when the population of each chain is 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 queueing networks which can be efficiently analyzed 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 which customer number k may actually visit.

ACKNOWLEDGEMENT : The authors are grateful to Dr. D. Potier of INRIA, France, for several useful suggestions.

APPENDIX 1

Proof of Theorem 1 :

$G_r(\underline{c})$ is the normalization constant of a queueing network of type 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 which do not have an IS discipline. Let the centers be enumerated so that centers 1,...,p are the ones with a FCFS, LCFSPR or PS discipline and p+1,...,N are the IS centers. The aggregate system state distribution for this system with state dependent service rate functions is (1)

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

where, by definition,

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

But

$$S^{(r)} = \bigcup_{\underline{\ell} \in L_r} S^{(r)}(\underline{\ell})$$

where L_r was defined in Theorem 1 and

$$S^{(r)}(\underline{\ell}) = \{\underline{n}^{(r)} \mid \underline{n}^{(r)} \in S^{(r)},\ n_{ir}=\ell_i \text{ for } 1 \le i \le N\}$$

so that

$$G_r(\underline{c}) = \sum_{\underline{\ell} \in L_r} \sum_{\underline{n}^{(r)} \in S^{(r)}(\underline{\ell})} \prod_{i=1}^{p} (\cdot) \prod_{i=p+1}^{N} (\cdot) .$$

Now $$\prod_{a=1}^{n_i^{(r-1)}+\ell_i} (a+c_i) = (1+c_i)\ldots(\ell_i+c_i)(\ell_i+c_i+1)\ldots(n_i^{(r-1)}+\ell_i+c_i)$$

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

Hence

$$G_r(\underline{c}) = \sum_{\underline{\ell} \in L_r} \prod_{i=1}^{p} ((\ell_i+c_i)!\, w_{ir}^{\ell_i}/c_i!\ell_i!) \prod_{i=p+1}^{N} (w_{ir}^{\ell_i}/\ell_i!) \cdot$$

$$\cdot \sum_{\underline{n}^{(r)} \in S^{(r)}(\underline{\ell})} \prod_{i=1}^{p} \left(\left(\prod_{a=1}^{n_i^{(r-1)}} (a+\ell_i+c_i) \right) \prod_{s=1}^{r-1} w_{is}^{n_{is}}/n_{is}! \right) \cdot$$

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

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

$$G_r(\underline{c}) = \sum_{\underline{\ell} \in L_r} \left(\prod_{i=1}^{p} \binom{\ell_i+c_i}{c_i} w_{ir}^{\ell_i} \prod_{i=p+1}^{N} w_{ir}^{\ell_i}/\ell_i! \right) G_{r-1}(\underline{c}+\underline{\ell}).$$

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

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

$$G_r(\underline{v}_r) = \sum_{\underline{\ell} \in L_r} \left(\prod_{i=1}^{N} h_{ir}(\underline{v}_r, \underline{\ell}) \right) G_{r-1}(\underline{v}_r + \underline{\ell})$$

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

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

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

We show this by induction on r.

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

$$I_{R-1} = \bigcup_{\underline{v}_R \in I_R} \{ \bigcup_{\ell \in L_R} \{\underline{v}_R + \underline{\ell}\} \} = \bigcup_{\underline{\ell} \in L_R} \{\underline{\ell}\} = L_R$$

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

$$I_{t-1} = \bigcup_{\underline{v}_t \in I_t} \{ \bigcup_{\underline{\ell} \in L_t} \{\underline{v}_t + \underline{\ell}\} \} = \bigcup_{\underline{\ell} \in L_t} \{ \bigcup_{\underline{v}_t \in I_t} \{\underline{v}_t + \underline{\ell}\} \}$$

$$= \bigcup_{\underline{\ell} \in L_t} \{ (\underline{v}_t + \underline{\ell}) \mid v_{it} \geq 0 \text{ for } 1 \leq i \leq N; \ \sum_{i=1}^{N} v_{it} = \sum_{s=t+1}^{R} K_s \}\}$$

Now $\underline{\ell} \in L_t$ so that $\ell_i \geq 0$ for $1 \leq i \leq N$ and $\sum_{i=1}^{N} \ell_i = K_t$. Hence,

$$I_{t-1} = \{ (\underline{v}_t+\underline{\ell}) \mid v_{it} \geq 0 \text{ for } 1 \leq i \leq N;\ \ell_i \geq 0 \text{ for } 1 \leq i \leq N;\ \sum_{i=1}^{N} v_{it} = \sum_{s=t+1}^{R} K_s;$$

$$\sum_{i=1}^{N} \ell_i = K_t \}$$

$$= \{ (\underline{v}_t+\underline{\ell}) \mid (v_{it}+\ell_i) \geq 0 \text{ for } 1 \leq i \leq N;\ \sum_{i=1}^{N} (v_{it}+\ell_i) = \sum_{s=t}^{R} K_s \}.$$

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

APPENDIX 2 :

Proof of Theorem 2 : (we use some notation and definitions found in Appendix 1)

By definition

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

Now, using eq 2.1, we may write

$$\Pr(n_{1R}=k_1,\ldots,n_{NR}=k_N) = G^{-1} \prod_{i=1}^{N} w_{iR}^{k_i}/k_i! \cdot$$

$$\cdot \sum_{\underline{n}^{(R)} \in S^{(R)}(\underline{k})} \prod_{i=1}^{p} ((n_i^{(R-1)}+k_i)! \prod_{r=1}^{R-1} w_{ir}^{n_{ir}}/n_{ir}!) \prod_{i=p+1}^{N} (\prod_{r=1}^{R-1} w_{ir}^{n_{ir}}/n_{ir}!)$$

$$= G^{-1} \prod_{i=1}^{p} w_{iR}^{k_i} \prod_{i=p+1}^{N} w_{iR}^{k_i}/k_i! \cdot$$

$$\cdot \sum_{\underline{n}^{(R)} \in S^{(R)}(\underline{k})} \prod_{i=1}^{p} (\{ \prod_{a=1}^{n_i^{(R-1)}} (a+k_i) \} \prod_{r=1}^{R-1} w_{ir}^{n_{ir}}/n_{ir}!) \prod_{i=p+1}^{N} \prod_{r=1}^{R-1} w_{ir}^{n_{ir}}/n_{ir}!) .$$

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

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

APPENDIX 3

Proof of Theorem 3(a) :

When there are no IS centers in the network, using the definition of $G_r(\underline{c})$ in Appendix 1, we may write

$$G_{R-1}(\underline{1}_\ell) = \sum_{\underline{n}^{(R-1)} \in S^{(R-1)}} (n_\ell^{(R-1)}+1) \prod_{i=1}^{N} (n_i^{(R-1)}!(\prod_{r=1}^{R-1} w_{ir}^{n_{ir}}/n_{ir}!)) \quad .$$

Therefore

$$G_{R-1}(\underline{1}_\ell) = G_{R-1}(\underline{0})\, E(n_\ell^{(R-1)}) + G_{R-1}(\underline{0})$$

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

$$\sum_{\ell=1}^{N} G_{R-1}(\underline{1}_\ell) = G_{R-1}(\underline{0})(K-1) + NG_{R-1}(\underline{0})$$

since

$$\sum_{\ell=1}^{N} E(n_\ell^{(R-1)}) = \sum_{r=1}^{R-1} K_r = K-1.$$

Proof of Theorem 3(b) :

Consider the definition of $G_r(\underline{c})$ in Appendix 1. When $\underline{c} = \underline{1}_x$ and x is an IS center, we have $c_i = 0$ for all $1 \leq i \leq p$ in which case $G_r(\underline{1}_x) = G_r(\underline{0})$. Hence $G_{R-1}(\underline{0}) = G_{R-1}(\underline{1}_x)$.

APPENDIX 4 :

We need to show that the value of $G^\circ_{k-1}(\underline{v}_{k-1})$ at location $M_k(\underline{v}_k)$ is not required in the computation of $G^\circ_k(\underline{\gamma})$, where $\underline{\gamma}$ is any vector such that $\underline{\gamma} \in I^\circ_k$ and $M_k(\underline{\gamma}) > M_k(\underline{v}_k)$.

Proof :

As can be seen from eq. 4.1, the computation of $G^\circ_k(\underline{\gamma})$ requires the values of $G^\circ_{k-1}(\underline{\gamma}+\underline{1}_i)$ for $1 \leq i \leq N$. We need to show that the value of $G^\circ_{k-1}(\underline{v}_{k-1})$ at location $M_k(\underline{v}_k)$ is not one of these. Hence, we need to show that $M_{k-1}(\underline{\gamma}+\underline{1}_i) > M_k(\underline{v}_k)$ for any $1 \leq i \leq N$ and $\underline{\gamma}$, such that $\underline{\gamma} \in I^\circ_k$ and $M_k(\underline{\gamma}) > M_k(\underline{v}_k)$.

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

By the definition of $\underline{\gamma}$, $M_k(\underline{\gamma}) > M_k(\underline{v}_k)$ so that $M_{k-1}(\underline{\gamma}+\underline{1}_i) < M_k(\underline{\gamma})$. 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 $1 \leq k \leq K^\circ$ and $1 \leq i \leq N$ so there is a contradiction.

APPENDIX 5 :

Suppose for the sake of simplicity that $K_r = \kappa$ for $1 \leq r \leq R$. When we break down the chains in N, so that each sub-chain 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

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

since $2^\kappa \geq \kappa+1$. 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

$$2^{R\kappa+1} / 2(\kappa+1)^R = (2^\kappa)^R / (\kappa+1)^R \geq 1.$$

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

REFERENCES

(1) Baskett, F., Chandy, K.M., Muntz, R.R., and Palacios, F. 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, 6(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) Courtois, P.J. *Decomposability : Queueing and computer system applications*. Academic Press, New York, 1977.

(7) Lam, S.S., and Lien, Y.L. A tree convolution algorithm for the solution of queueing networks. *Commun.* ACM, 26, 3(March 1983), 203-215.

(8) Lam, S.S., and Wong, J.W. Queueing network models of packet switching networks, Part 2 : Networks with population size constraints. *Performance Evaluation*, 2, 3(1982), 161-180.

(9) Lavenberg, S.S. (Ed.) *Computer performance modeling handbook*. Academic Press, New York, 1983.

(10) Lavenberg, S.S., and Reiser, M. Stationary state probabilities at arrival instants for closed queueing networks with multiple types of customers. *J. of Applied Probability*, 17, (Dec. 1980), 1048-1061.

(11) Little, J.D.C. A proof of the queueing formula L=λW. *Operations Research*, 9, (1961), 383-387.

(12) McKenna, J., and Mitra, D. Integral representations and asymptotic expansions for closed Markovian queueing networks : Normal usage. *BSTJ*, 61, 5(May-June 1982), 661-683.

(13) 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.

(14) Neuse, D.M. Approximate analysis of large and general queueing networks. Ph.D. Thesis, The University of Texas at Austin, 1982.

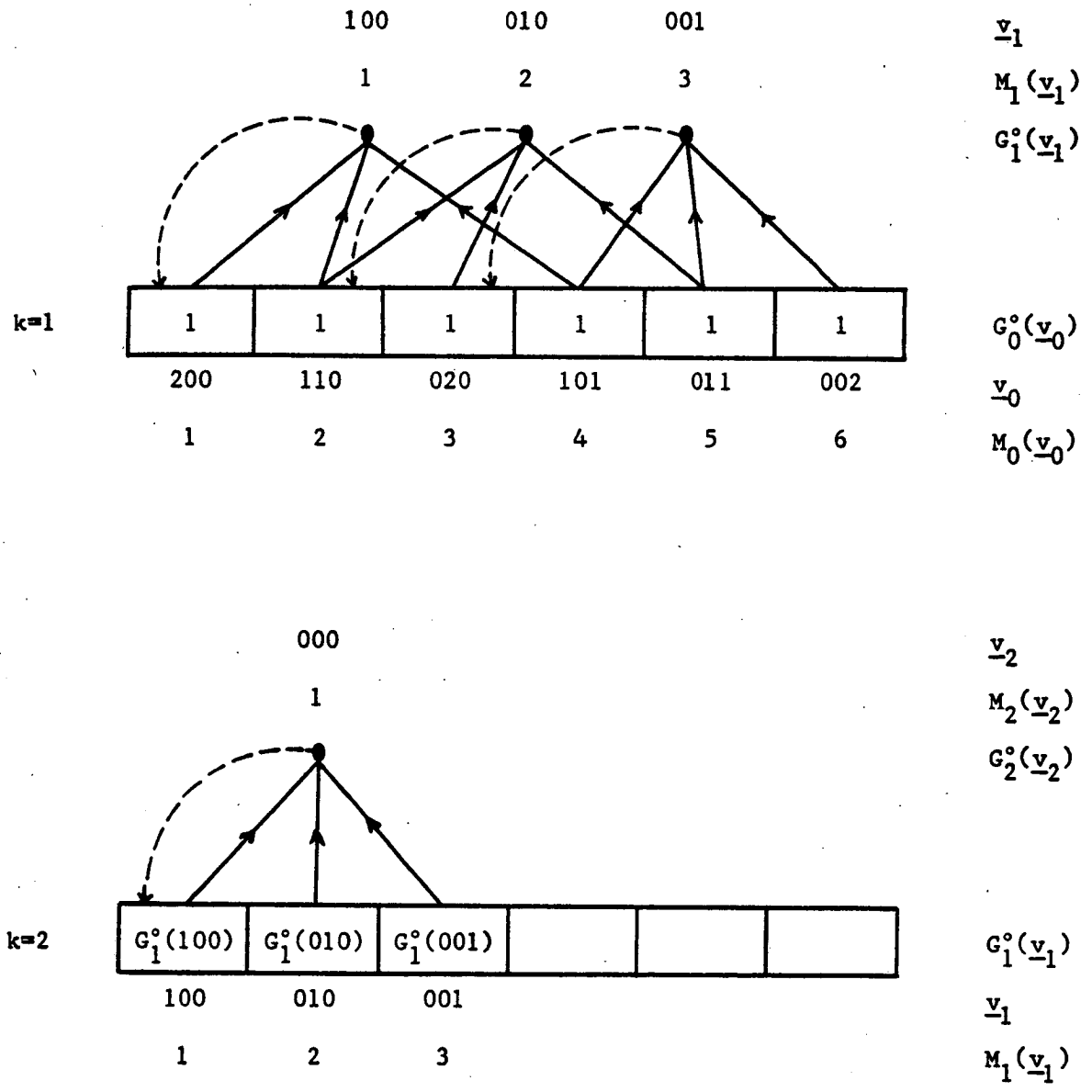
(15) Reiser, M., and Kobayashi, H. Queueing networks with multiple closed chains : Theory and computational algorithms. *IBM J. Res. and Dev.*, 19, (May 1975), 283-294.

(16) Reiser, M., and Lavenberg, S.S. Mean value analysis of closed multichain queueing networks. *J.ACM*, 27, 2(Apr. 1980), 313-322.

(17) Sauer, C.H., and Chandy, K.M. *Computer systems performance modeling*. Prentice Hall, New Jersey, 1981.

Figure 1 - Illustration of the implementation of eq. 4.1

(N=3, K°=2)



Legend : ----→ Pointer to storage location of $G_k^\circ(\underline{v}_k)$

——→ Elements needed in the computation of $G_k^\circ(\underline{v}_k)$

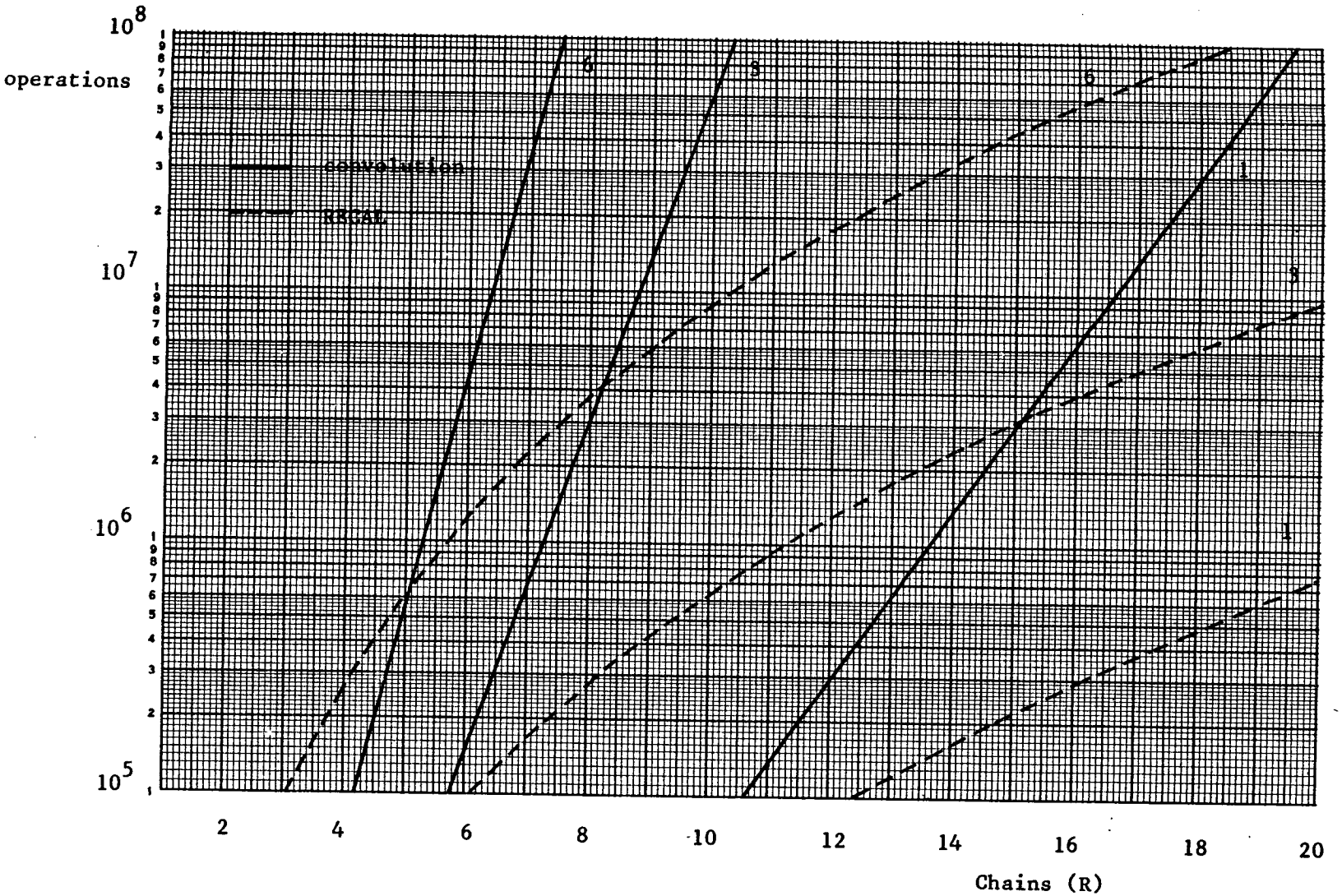


Figure 2 - Comparison of the number of operations in RECAL and convolution, for N=4 and κ=1,3 and 6. (The values for κ are indicated at the ends of the curves)

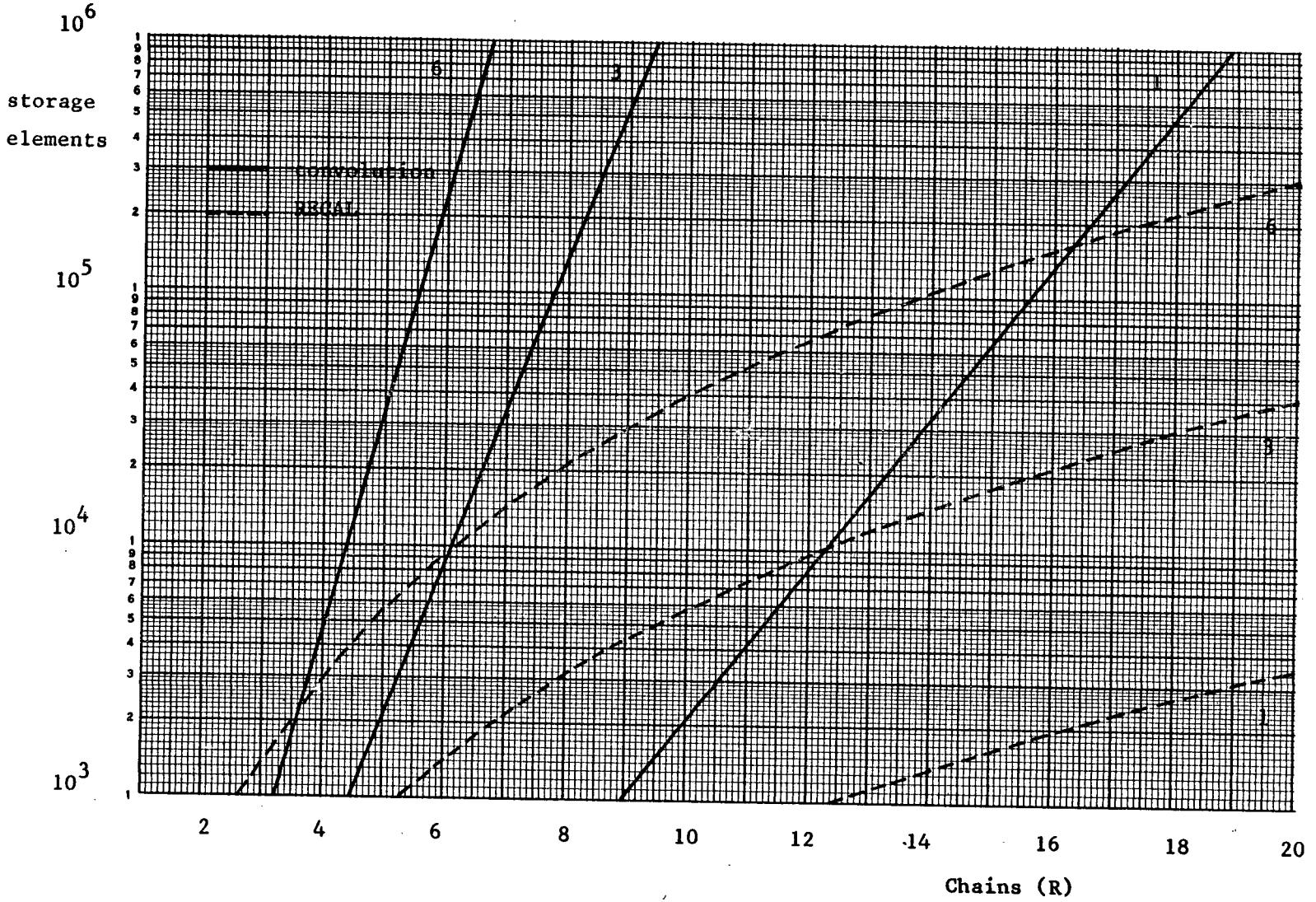


Figure 3 - Comparison of the storage space requirements in RECAL and convolution, for N=4 and κ=1,3 and 6. (The values for κ are indicated at the ends of the curves)

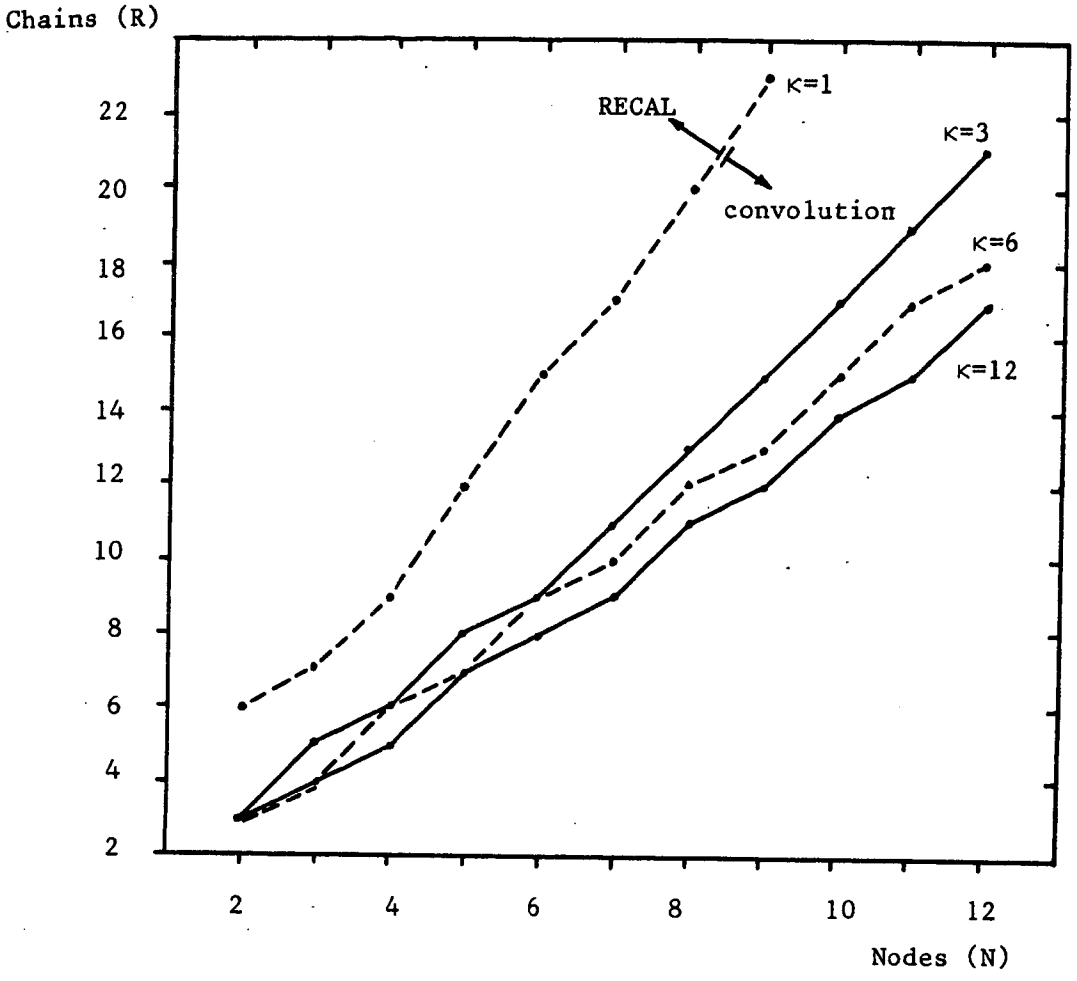


Figure 4 - Regions in the space (N×R) in which the time requirement (number of operations) of RECAL is less than convolution, for K=1,3,6 and 12. (On or above the curves the time requirement of RECAL is less than convolution)

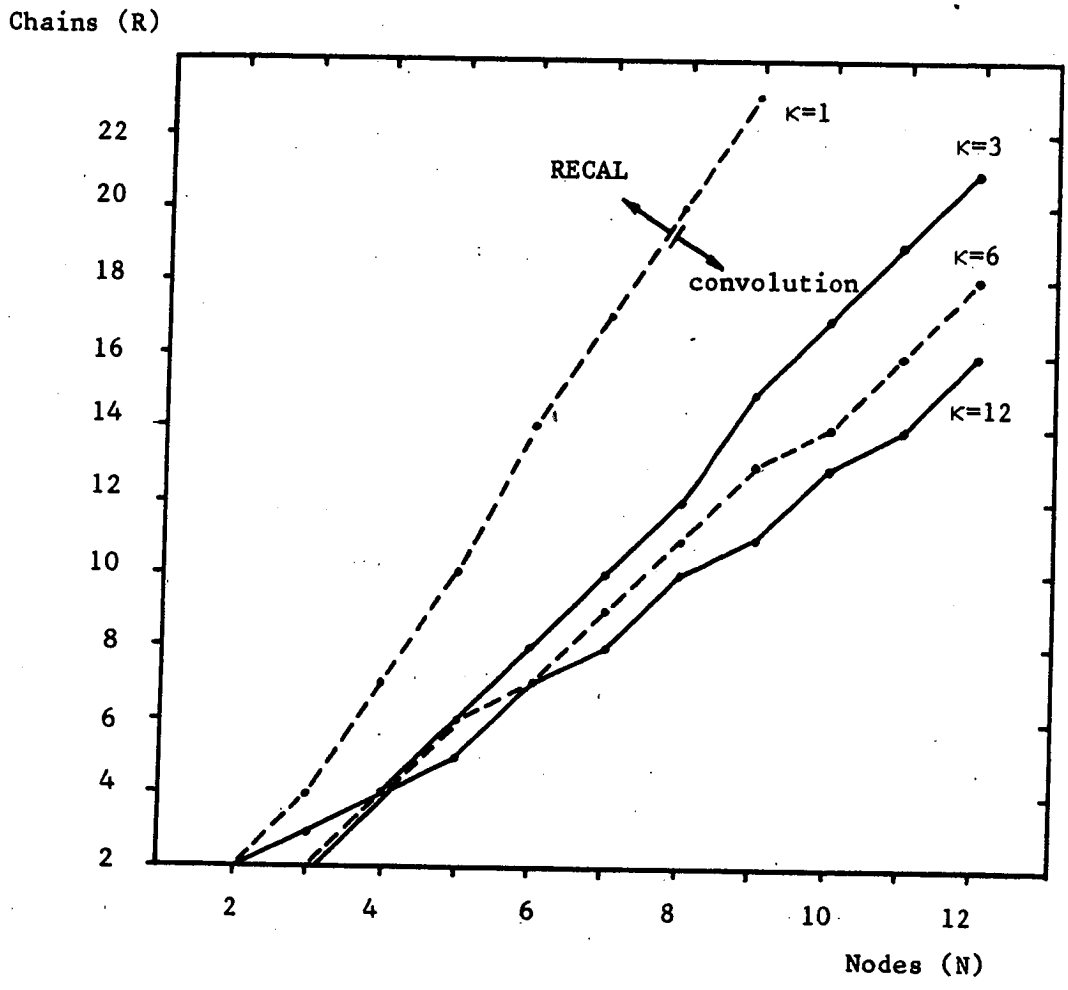


Figure 5 - Regions in the space (N×R) in which the space requirement (number of elements) of RECAL is less than convolution, for K=1,3,6 and 12. (On or above the curves the space requirement of RECAL is less than convolution)

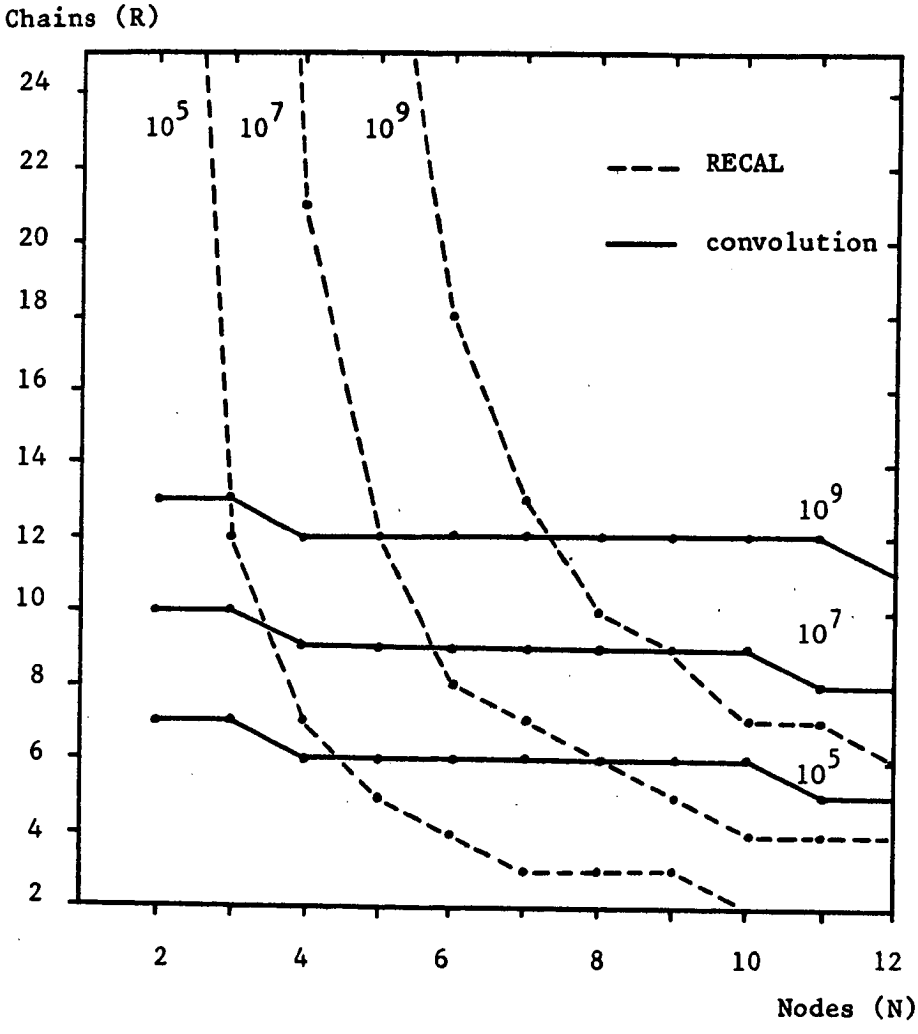


Figure 6 - Curves in the space (N×R) on and above which the time requirement (number of operations) is equal to or greater than the number at the end of the curve, for $\kappa=3$.

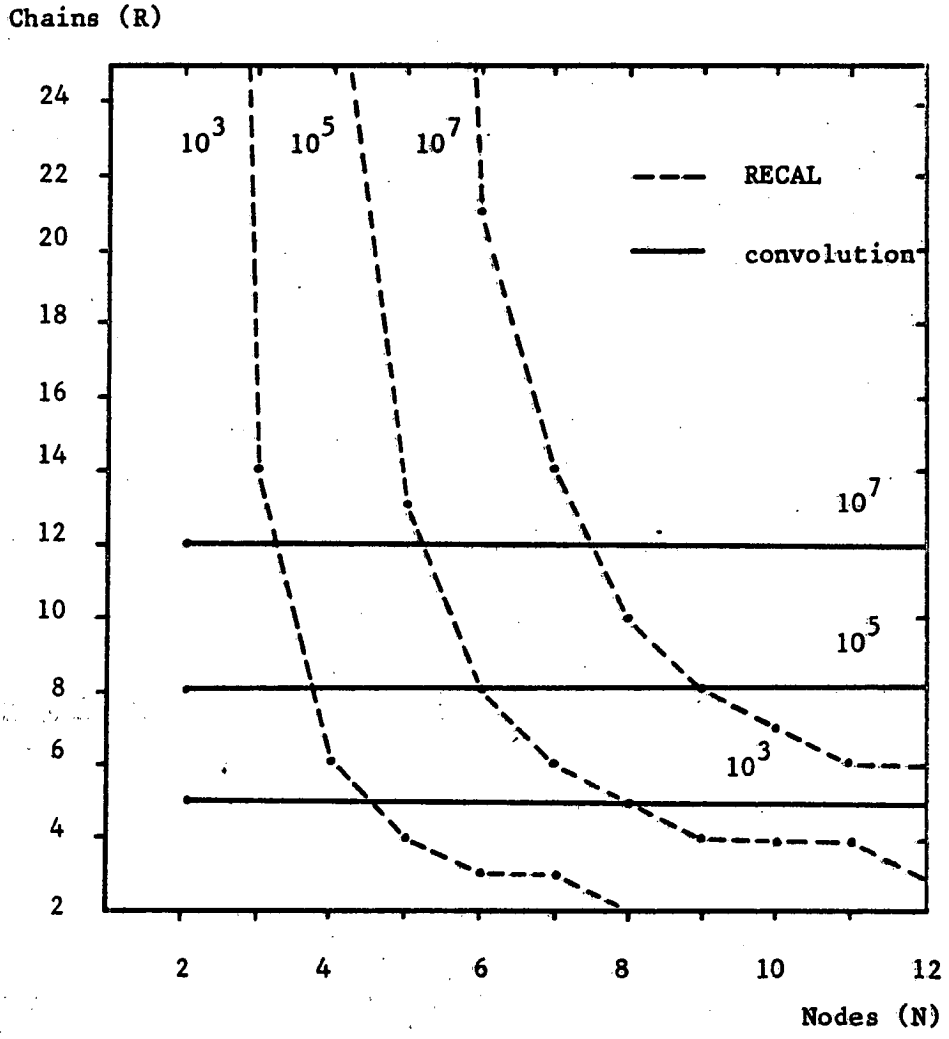


Figure 7 - Curves in the space (N×R) on and above which the space requirement (number of elements) is equal to or greater than the number at the end of the curve, for $\kappa=3$.

Table 1 - Assignments of the customers in N° to the chains in N.

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

Table 2 - Example of the mapping $M_x(\underline{d})$

($N=3$, $K^\circ=3$, $x=0$)

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

Table 3 - Example analysis using RECAL

number of nodes 4
number of chains 10

chain	chain population
1	3
2	3
3	3
4	3
5	3
6	3
7	3
8	3
9	3
10	3

node	discipline
1	fcfs
2	fcfs
3	ps (or lcfspr)
4	is

all nodes constant speed servers

mean service requirements at the nodes

	node 1	node 2	node 3	node 4
chain 1	1.00e+00	2.00e+00	2.00e+00	1.00e+00
chain 2	1.00e+00	2.00e+00	2.00e+00	1.00e+00
chain 3	1.00e+00	2.00e+00	5.00e+00	1.00e+00
chain 4	1.00e+00	2.00e+00	5.00e+00	2.00e+00
chain 5	1.00e+00	2.00e+00	5.00e+00	5.00e+00
chain 6	1.00e+00	2.00e+00	1.00e+00	1.00e+00
chain 7	1.00e+00	2.00e+00	1.00e+00	2.00e+00
chain 8	1.00e+00	2.00e+00	4.00e+00	4.00e+00
chain 9	1.00e+00	2.00e+00	1.00e+00	4.00e+00
chain 10	1.00e+00	2.00e+00	1.00e+00	5.00e+00

visit ratios

	node 1	node 2	node 3	node 4
chain 1	1.00e-01	1.00e-01	1.00e-01	1.00e-01
chain 2	1.00e-01	3.00e-01	3.00e-01	1.00e-01
chain 3	1.00e-01	1.00e-01	2.00e-01	2.00e-01
chain 4	3.00e-01	1.00e-01	6.00e-01	7.00e-01
chain 5	1.00e-02	1.00e-02	3.00e-02	1.00e-01
chain 6	1.00e-02	1.00e-02	2.00e-02	1.00e-01
chain 7	2.00e-02	5.00e-02	4.00e-02	2.10e-01
chain 8	1.00e-02	2.30e-01	1.00e-02	7.00e-02
chain 9	1.00e-03	1.00e-03	1.00e-03	1.10e-02
chain 10	2.00e-03	1.00e-03	1.00e-03	4.00e-03

mean performance measures for chain 1

	throughput	utilization	queue length	waiting time
node 1	5.54385e-02	5.54385e-02	8.26038e-02	1.49001e+00
node 2	5.54385e-02	1.10877e-01	1.10497e+00	1.99314e+01
node 3	5.54385e-02	1.10877e-01	1.75699e+00	3.16926e+01
node 4	5.54385e-02	na	5.54385e-02	1.00000e+00

mean performance measures for chain 2

	throughput	utilization	queue length	waiting time
node 1	1.90935e-02	1.90935e-02	2.90004e-02	1.51886e+00
node 2	5.72806e-02	1.14561e-01	1.13950e+00	1.98934e+01
node 3	5.72806e-02	1.14561e-01	1.81240e+00	3.16408e+01
node 4	1.90935e-02	na	1.90935e-02	1.00000e+00

mean performance measures for chain 3

	throughput	utilization	queue length	waiting time
node 1	1.71802e-02	1.71802e-02	2.60736e-02	1.51794e+00
node 2	1.71802e-02	3.43605e-02	3.73650e-01	2.17488e+01
node 3	3.43605e-02	1.71802e-01	2.56591e+00	7.46763e+01
node 4	3.43605e-02	na	3.43605e-02	1.00000e+00

mean performance measures for chain 4

	throughput	utilization	queue length	waiting time
node 1	1.87193e-02	1.87193e-02	2.83680e-02	1.51544e+00
node 2	6.23978e-03	1.24796e-02	1.39535e-01	2.23622e+01
node 3	3.74387e-02	1.87193e-01	2.74474e+00	7.33130e+01
node 4	4.36784e-02	na	8.73569e-02	2.00000e+00

mean performance measures for chain 5

	throughput	utilization	queue length	waiting time
node 1	1.00131e-02	1.00131e-02	1.52143e-02	1.51944e+00
node 2	1.00131e-02	2.00263e-02	2.18528e-01	2.18241e+01
node 3	3.00394e-02	1.50197e-01	2.26360e+00	7.54210e+01
node 4	1.00131e-01	na	5.00657e-01	5.00000e+00

mean performance measures for chain 6

	throughput	utilization	queue length	waiting time
node 1	4.72903e-02	4.72903e-02	7.05930e-02	1.49276e+00
node 2	4.72903e-02	9.45807e-02	9.48854e-01	2.00644e+01
node 3	9.45807e-02	9.45807e-02	1.50765e+00	1.59404e+01
node 4	4.72904e-01	na	4.72904e-01	1.00000e+00

mean performance measures for chain 7

	throughput	utilization	queue length	waiting time
node 1	2.91326e-02	2.91326e-02	4.38978e-02	1.50682e+00
node 2	7.28316e-02	1.45663e-01	1.38341e+00	1.89946e+01
node 3	5.82653e-02	5.82653e-02	9.60910e-01	1.64920e+01
node 4	3.05893e-01	na	6.11785e-01	2.00000e+00

mean performance measures for chain 8

	throughput	utilization	queue length	waiting time
node 1	6.23296e-03	6.23296e-03	9.56511e-03	1.53460e+00
node 2	1.43358e-01	2.86716e-01	2.38147e+00	1.66120e+01
node 3	6.23296e-03	2.49318e-02	4.34442e-01	6.97008e+01
node 4	4.36307e-02	na	1.74523e-01	4.00000e+00

mean performance measures for chain 9

	throughput	utilization	queue length	waiting time
node 1	3.66486e-02	3.66486e-02	5.46840e-02	1.49212e+00
node 2	3.66486e-02	7.32972e-02	7.30798e-01	1.99407e+01
node 3	3.66486e-02	3.66486e-02	6.01981e-01	1.64258e+01
node 4	4.03134e-01	na	1.61254e+00	4.00000e+00

mean performance measures for chain 10

	throughput	utilization	queue length	waiting time
node 1	1.01828e-01	1.01828e-01	1.47736e-01	1.45084e+00
node 2	5.09139e-02	1.01828e-01	9.98803e-01	1.96175e+01
node 3	5.09139e-02	5.09139e-02	8.35183e-01	1.64038e+01
node 4	2.03656e-01	na	1.01828e+00	5.00000e+00

na = not applicable