A CONVOLUTION ALGORITHM FOR PRODUCT-FORM QUEUEING NETWORKS WITH BLOCKING*

S. Balsamo• and C. Clò ••

• Dipartimento di Matematica e Informatica, Università di Udine, Udine, Italy
•• Dipartimento di Informatica, Università di Pisa, Pisa, Italy

Abstract

Queueing network models with finite capacity and blocking can be used to represent various systems with finite resources, including communication and computer systems, as well as production and manufacturing systems to evaluate their performance. Various blocking types have been defined to represent various system behaviors. Queueing networks with blocking show a product-form solution under special constraints. We present a convolution algorithm for the class of product-form closed network models with blocking to evaluate queue length distribution and average performance indices. As a consequence, this class of networks can be analysed with a polynomial time computational complexity.

1. Introduction

The behavior of various systems, including communication and computer systems, as well as production and manufacturing systems, can be represented and analysed through queueing network models to evaluate their performance. System performance analysis consists of the derivation of a set of figures of merit. This usually includes the queue length distribution and some average performance indices such as mean response time, throughput, and utilization.

Because of the system's resource constraints, realistic models should have finite capacity queues, i.e., the queue length cannot exceed a maximum threshold. When the queue length reaches this threshold, the queue is said to be full. Then any individual job that attempts to enter a full queue is not accepted and blocking arises. In the literature several blocking mechanisms have been proposed to represent various system behaviors, including Blocking After Service (*BAS*), Blocking Before Service (*BBS*), Repetitive Service (*RS*), *Stop* Blocking, and *Recirculate* Blocking [1-4, 7, 9, 10, 15, 16, 19, 20].

* This work has been partially supported by CNR Project Research Funds and MURST Project Research

An important class of queueing networks is the class of product-form networks, which are characterised by a closed-form expression of the steady-state probability distribution of the joint queue length. Most product-form queueing networks have infinite capacity queues and they do not have blocking, such as the class of so-called BCMP networks [5, 14]. For BCMP networks, some efficient algorithms have been defined to evaluate performance indices. The two most important algorithms are convolution and Mean Value Analysis (MVA), whose computational complexity is polynomial in the numbers of service centers and customers [6, 17, 18].

Queueing networks with blocking can exhibit a product-form solution or not. Queueing networks without product-form solution can be analyzed by exact algorithms with an exponential computational complexity in the number of network components (i.e., service centers, customers and queue capacities). Product-form solutions of queueing networks with blocking have been derived, under special constraints, for different blocking mechanisms [1-4, 7, 9-11, 15, 16, 19, 20]. In this paper we present a convolution algorithm for product-form queueing networks with blocking, whose computational complexity is polynomial in the number of network components.
An MVA algorithm is presented in [7] for the class of product-form networks with cyclic topology and with blocking.

The queue length distribution of product-form networks with blocking can be expressed similarly to that of product-form networks without blocking. However, we cannot immediately apply the convolution algorithm already known for BCMP networks [6, 17]. Indeed, the finite capacity queues limit the state space of the network. This leads to a new definition of each step of the convolution algorithm taking into account a set of constraints on the queue lengths.

The proposed convolution algorithm could be used in approximate methods for queueing networks with blocking without product-form, similarly to queueing networks with infinite capacity queues. More precisely, approximate and possibly iterative methods could be defined based on product-form networks solution methods such as Marie's method for networks without blocking [13]. The approximate analysis of queueing networks with blocking is however beyond the scope of this paper.
A few results concerning the algorithm for queueing networks with blocking are known in the literature. In particular, for product-form networks with RS-RD blocking two computational algorithms have been proposed in [1, 16]. These methods have a computational complexity higher than that of the convolution algorithm presented in this paper, as we shall discuss with detail in Section 3.

Specifically, our recursive algorithm has a time computational complexity that is linear in the number of network components, i.e., the number of service centers and the number of customers.

In Section 2 we present the class of queueing network models with product-form and blocking. Section 3 provides the proposed convolution algorithm. In Section 4 we derive a closed-form expression for the marginal queue length distribution of each service center, as well as the average performance indices, by using the set of recursive equations of the algorithm. Section 5 and 6 present some numerical examples and the conclusions, respectively. The pseudo-code of the algorithm is presented in the Appendix.

2. Product-form networks with blocking

In this section we introduce the main blocking mechanisms and a class of product-form closed queueing networks with blocking. Specifically, we consider single class networks with load independent exponential service centers and with single type of customers.

Let M denote the number of network nodes, N the number of customers, and B_i the finite capacity of queue i, $1 \le i \le M$. Let F_i and μ_i respectively denote the service time distribution and the mean service rate of service center i, $1 \le i \le M$. Customers in the queue are served according to a service discipline. Let $\vec{n}=(n_1,...,n_M)$ denote the network state when there are n_i customers in node i, $1 \le i \le M$. Let $a_i = \max(0, N - \sum_{j=1, j \neq i}^{M} B_j)$ denote the minimum feasible queue length of service center i in the network, $1 \le i \le M$. That is, the number of customers in service center i satisfies the following constraints: $a_i \le n_i \le B_i$. Let $\boldsymbol{P} = [p_{ij}]$ be the routing matrix of the network, where p_{ij} denotes the probability that a customer leaving node i tries to enter node j, $1 \le i,j \le M$. Let $\vec{x}=(x_1,...,x_M)$ be a solution of the following linear system: $\vec{x}=\vec{x}\boldsymbol{P}$. The network routing matrix $\boldsymbol{P}$ is said to be reversible if $x_i p_{ij} = x_j p_{ji}$, $1 \le i,j \le M$ [11].

When the queue length of node j reaches its capacity, i.e., $n_j = B_j$, the queue is said to be full. When a customer attempts to enter a full queue, it is not accepted and blocking arises. Various blocking mechanisms have been defined in the literature to describe different behaviors of customer arrivals at a full capacity node and the servers' activity in the network. We now recall the definition of the most important blocking mechanisms.

Repetitive Service Blocking (RS): a customer upon completion of its service at queue *i* attempts to enter destination queue *j*. If node *j* is full, the customer is looped back into the sending queue *i*, where it receives a new independent service according to the service discipline.

Two different subcategorises have been introduced depending on whether the customer after receiving a new service, chooses a new destination node independently of the one that it had selected previously:

RS-RD (random destination) if a customer destination is randomly chosen at the end of each new service, whatever the previous choices;

RS-FD (fixed destination) if a customer destination is determined after the first service and cannot be modified.

Blocking Before Service (BBS): a customer declares its destination node *j* before it starts receiving service at node *i*. If at that time node *j* is full, the service at node *i* does not start and the server is blocked. If a destination node *j* becomes full during the service of a customer at node *i* whose destination is *j*, node *i* service is interrupted and the server is blocked. The service of node *i* will be resumed as soon as a departure occurs from node *j*. The destination node of a blocked customer does not change.

Two different subcategorises can be introduced [15] depending on whether the server can be used as a service center buffer when the node is blocked:

BBS-SO (server occupied) when the server of the blocked node is used to hold a customer;

BBS-SNO (server is not occupied) when the server of the blocked node cannot be used to hold a customer.

Blocking After Service (BAS): if a job attempts to enter a full capacity queue *j* upon completion of a service at node *i*, it is forced to wait in node *i* server, until the destination node *j* can be entered. The server of source node *i* stops processing jobs (it is blocked) until destination node *j* releases a job. Node *i* service will be resumed as soon as a departure occurs from node *j*. At that time the job waiting in node *i* immediately moves to node *j*.

If more than one node is blocked by the same node *j*, then a scheduling discipline must be considered to define the unblocking order of the blocked nodes when a departure occurs from node *j*. First Blocked First Unblocked is a possible discipline [15] which states that the first node to be unblocked is the one which was first blocked.

Stop Blocking: the service rate of each node is delayed by a factor $d(N) \geq 1$, when there is a network population $N \geq 0$. In other words, the actual job service rate of each node depends on the state $\vec{n}$ of the entire network according to function $d(N)$. When $d(N) = 0$ then the service at each node in the network is stopped. Services will be resumed at each node as soon as an exogenous arrival occurs.

We shall now define the class of closed single class queueing networks with various blocking mechanisms and with product-form.
The state space E of the network is the set of all feasible states.
We assume that the queueing network can be represented by a continuous-time ergodic Markov chain with discrete state space E. The stationary and transient behavior of the network can be analyzed by the underlying Markov process.
Under the hypothesis of an irreducible routing matrix $\boldsymbol{P}$ and deadlock-free network, there exists a unique steady-state joint queue length distribution $\vec{\pi} = \{\pi(\vec{n}), \vec{n} \quad E\}$.

Under certain constraints, depending both on network parameters and the blocking mechanisms, $\vec{\pi}$ has a product-form solution [1-4, 7, 9-11, 15, 16, 19, 20]. It can be defined as follows:

$$\pi(\vec{n}) = \frac{1}{G} \prod_{i=1}^{M} g_i(\boldsymbol{n}_i) \tag{1}$$

where the normalising constant G is given by

$$G = \sum_{\vec{n} \quad E} \prod_{i=1}^{M} g_i(\boldsymbol{n}_i) \tag{2}$$

and the definition of function g_i, $1 \leq i \leq M$, depends on network parameters, the i-th queue length, the blocking type and some additional constraints.
A detailed survey of the various cases of product-form networks with blocking is given in [3, 4].
In order to define the convolution algorithm we consider the following two definitions of function g_i, which hold for various blocking types and under different constraints:

$$g_i(n_i) = \rho_i^{n_i}, \; n_i > 0, \; 1 \leq i \leq M \tag{3}$$

where

Case I

$$\rho_i = \left(\frac{\lambda_i}{\mu_i} \right) \tag{4}$$

Case II $$\rho_i = 1/\varepsilon_i \quad (5)$$

where $\vec{\varepsilon}=(\varepsilon_1,...,\varepsilon_M)$ is obtained by solving the system $\vec{\varepsilon}=\vec{\varepsilon}\,\boldsymbol{P}'$ with $\boldsymbol{P}'=[p'_{ij}]$, $p'_{ij}=\mu_j\, p_{ji}$, $j\neq i$, $p'_{ii}=1-\sum_{j\neq i} p_{ij}$, $1\leq i,j\leq M$.

These simplified expressions can easily be obtained by the general definition of product-form networks with blocking presented in [2, 3, 4, 9, 10, 11, 15, 16] and the assumption above. In particular, case I holds for the following networks:

- a network with *RS-RD* or *Stop* blocking, reversible routing, nodes with arbitrary scheduling [19];
- a two-node network with blocking types *BAS*, *BBS-SO*, *RS-RD*, *RS-FD* and *Stop*, and FIFO service discipline [15, 19].

Case II holds for [3, 9]:

- a network with arbitrary topology, *RS-RD* blocking and where the number of customers N verifies the strictly non-empty condition, which can be written as follows:

$$N > \sum_{j=1}^{M} B_j - \min_{1\leq j\leq M} B_j \quad (6)$$

This condition states that the number of customers is such that each node can never be empty.

- a network with *RS-FD* or *BBS-SO* blocking, when the strictly non-empty condition (6) holds and when each node i with finite capacity has a single sending node j, i.e., if $p_{ij}>0$ then $p_{ij}=1$, $1\leq j\leq M$;
- a network with cyclic topology, *BBS* or *RS* blocking and where the number of jobs N verifies the following non-empty condition:

$$N \geq \sum_{j=1}^{M} B_j - \min_{1\leq j\leq M} B_j \quad (6')$$

This condition implies that at most one of the nodes with the minimum capacity can be empty.

3. Convolution algorithm

We now present a convolution algorithm to compute the normalising constant G. From constant G we can derive the steady-state distribution $\pi(\vec{n})$, according to the formula (1) and some average performance indices. The direct computation of constant G based on formula (2) as a summation over all the feasible states of the network, takes an exponential time in the number of nodes and customers of the network. We propose a

convolution algorithm to compute the normalizing constant G with a polynomial time computational complexity.

We define function $G_j(n)$ as follows:

$$G_j(n) = \sum_{\substack{(n_1,\ldots,n_j) \\ a_i \le n_i \le B_i,\ 1 \le i \le j,\ \sum_{i=1}^{j} n_i = n}} \prod_{i=1}^{j} \rho_i^{n_i} \qquad (7)$$

for $A_j \le n \le B(j)$, $1 \le j \le M$, where

$$A_j = \sum_{i=1}^{j} a_i \ , \ B(j) = \sum_{i=1}^{j} B_i \ , \quad a_i = \max(0,\ N - B(M) + B_i \ )$$

a_i and B_i are the minimum and maximum queue lengths of node i, $1 \le i \le M$.

For queueing networks with infinite capacity queues $G_j(n)$ is the normalizing constant of the network with the first j nodes and n customers, $1 \le j \le$ M and $0 \le n \le N$. However, note that this is not always true for networks with finite capacity when $j<M$ and $n<N$. Indeed, the constraint $n_i \ge a_i$ in formula (7) depends on the parameters of the entire network, i.e., on N and B_j, $1 \le j \le$ M. $G_j(n)$ is the normalising constant of the network with the first j nodes and n customers only when $a_i = 0$, $1 \le i \le j$. However, here after, we use this interpretation of function $G_j(n)$ in any case, for the sake of simplicity.

By definition, the overall normalising constant is $G = G_M(N)$. In order to evaluate this constant, we shall now present the set of recursive equations to compute functions $G_j(n)$, $A_j \le n \le B(j)$, $1 \le j \le M$.

Theorem 1

For $n \le N$ and $2 \le j \le M$,

i) if $A_{j-1} + B_j > B(j\text{-}1) + a_j$ then

$$G_j(n) = \rho_j^{a_j}\ G_{j-1}\left(A_{j-1}\right) \qquad \text{if } n = A_j \qquad (8.1)$$

$$G_j(n) = \rho_j^{a_j}\ G_{j-1}\left(n - a_j\right) + \rho_j\ G_j(n-1) \qquad \text{if } A_j + 1 \le n \le B(j\text{-}1) + a_j \qquad (8.2)$$

$$G_j(n) = \rho_j\ G_j(n-1) \qquad \text{if } B(j\text{-}1) + a_j + 1 \le n \le A_{j-1} + B_j \qquad (8.3)$$

$$G_j(n) = -\rho_j^{B_j+1}\ G_{j-1}\left(n - B_j - 1\right) + \rho_j\ G_j(n-1) \qquad (8.4)$$

$$\text{if } A_{j-1} + B_j + 1 \le n \le B(j) - 1$$

$$G_j(n) = \rho_j^{B_j}\ G_{j-1}\left(B(j-1)\right) \qquad \text{if } n = B(j) \qquad (8.5)$$

ii) if $A_{j-1}+B_j \le B(j-1)+a_j$ then

if $n = A_j$, $G_j(n)$ is given by formula (8.1);
if $A_j+1 \le n \le A_{j-1}+B_j$, $G_j(n)$ is given by formula (8.2);
if $A_{j-1}+B_j+1 \le n \le B(j-1)+a_j$, then

$$G_j(n) = \rho_j^{a_j} G_{j-1}(n-a_j) - \rho_j^{B_j+1} G_{j-1}(n-B_j-1) + \rho_j G_j(n-1) \qquad (8.6)$$

if $B(j-1)+a_j+1 \le n \le B(j)-1$, $G_j(n)$ is given by formula (8.4);
if $n = B(j)$, $G_j(n)$ is given by formula (8.5);
and with initial condition, for $j = 1$

$$G_1(n) = \rho_1^n \qquad \text{for } a_1 \le n \le B_1 = B(1) \qquad (8.7)$$

Proof:
The initial condition given by formula (8.7) derives from definition (7). To derive the set of recursive equations, we consider the generating function $G_j(z)$ of $G_j(n)$ and two different computational approaches.
First consider the generating function $\gamma_i(z)$ of node i, $1 \le i \le M$, defined as follows:

$$\gamma_i(z) = \sum_{k=a_i}^{B_i} \rho_i^k z^k = \frac{(\rho_i z)^{a_i} - (\rho_i z)^{B_i+1}}{1-\rho z}$$

Then consider the generating function $G_j(z)$ related to the first j nodes, $1 \le j \le M$:

$$G_j(z) = \prod_{i=1}^{j} \gamma_i(z)$$

We can write

$$G_j(z) = \frac{(\rho_j z)^{a_j} - (\rho_j z)^{B_j+1}}{1-\rho_j z} G_{j-1}(z)$$

from which

$$G_j(z) = (\rho_j z)^{a_j} G_{j-1}(z) - (\rho_j z)^{B_j+1} G_{j-1}(z) + \rho_j z\, G_j(z) \qquad (9)$$

On the other hand we can also write the generating function $G_j(z)$ as follows:

$$G_j(z) = \sum_{n=A_j}^{B(j)} G_j(n)\, z^n \qquad (10)$$

where function $G_j(n)$ is given by formula (7).

Then we can substitute expression (10) in equation (9), thus obtaining the following expression:

$$\sum_{n=A_j}^{B(j)} G_j(n) z^n =$$

$$= \sum_{n=A_{j-1}}^{B(j-1)} \rho_j^{a_j}\, G_{j-1}(n) z^{n+a_j} - \sum_{n=A_{j-1}}^{B(j-1)} \rho_j^{B_j+1} G_{j-1}(n) z^{n+B_j+1} + \sum_{n=A_j}^{B(j)} \rho_j\, G_j(n) z^{n+1}$$

which can be rewritten as follows

$$\sum_{n=A_j}^{B(j)} G_j(n)\, z^n =$$

$$= \rho_j^{a_j} \sum_{m=A_j}^{B(j-1)+a_j} G_{j-1}(m-a_j) z^m - \rho_j^{B_j+1} \sum_{m=A_{j-1}+B_j+1}^{B(j)-1} G_{j-1}(m-B_j-1) z^m + \rho_j \sum_{m=A_j+1}^{B(j)+1} G_j(m-1) z^m$$

We equate the coefficients of the two polynomials in z of the two sides of this equation by considering whether the following condition holds: $A_{j-1} + B_j > B(j-1) + a_j$. For example, the coefficients of z^n for $n = A_j$ are $G_j(A_j)$ and $\rho_j^{a_j} G_{j-1}(A_{j-1})$ on the left and the right hand sides of the equation, respectively.

By equating the two coefficients we immediately obtain the recursive equation (8.1). Likewise for $A_j < n \le B(j)$ we obtain the equations from (8.2) to (8.6).◊

The theorem defines the convolution algorithm to recursively compute the normalizing constant G.

The convolution algorithm for queueing networks with infinite capacity queues is based on the following recursive equation for load independent service rates:

$$G_j(n) = G_{j-1}(n) + \rho_j G_j(n-1) \qquad (11)$$

for $1 \le j \le M$, $0 \le n \le N$.

This equation can be interpreted by considering that the total number of customers *n* in the network with the first *j* nodes can be distributed on the nodes as follows:
(a) zero customers in node *j* and *n* customers in the first *j-1* nodes,
(b) at least one customer in node *j* and the remaining *n-1* customers in the first *j* nodes (node *j* included).
Case (a) corresponds to the first term and case (b) to the second term of the right hand side of equation (11).

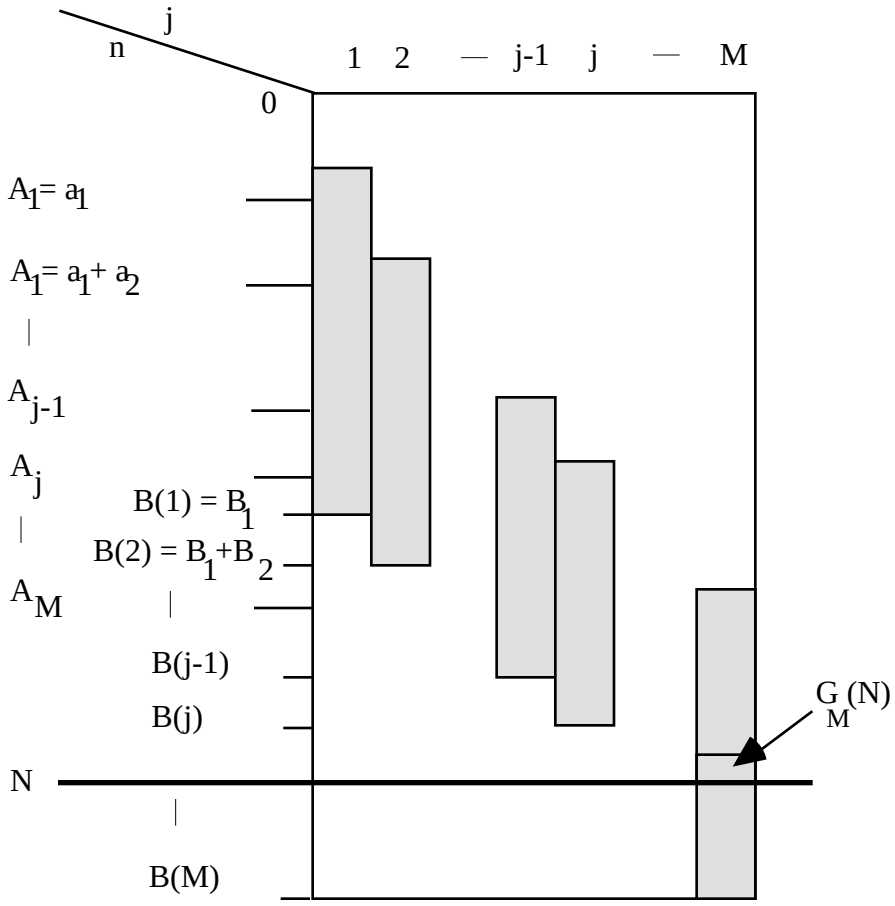

Figure 1 - Scheme of the range *n* for each function $G_j(n)$, $A_j \le n \le B(j)$, $1 \le j \le M$, to evaluate the normalizing constant $G_M(N)$.

Function $G_j(n)$ defined by formula (7) for queueing networks with finite capacity queues can be interpreted as the normalizing constant of the network with the first *j* nodes and *n* customers. Note that in the special case of $a_j=0$ equation (8.2) reduces to equation (11). In the general case ($a_j \ge 0$) the proposed convolution algorithm evaluates function $G_j(n)$ for a range of values of the number of customers depending on node *j* , $A_j \le n \le B(j)$. Note that, except for the first node, the upper bound can be greater than the total network population, i.e., it can be $B(j) > N$. So the evaluation of $G_j(n)$ for $n > N$ is not necessary.

Figure 1 shows the range of values n of each function $G_j(n)$, $1 \le j \le M$. Figures 2 and 3 show the recursive equations (8.1)-(8.6) for function $G_j()$ for the two cases (i) and (ii), respectively.

Equations (8.1) and (8.5) can be easily interpreted by noting that the number of customers $n=A_j$ and $B(j)$ determine a unique distribution of the customers on the nodes. In the two cases, the only feasible state is $(n_1, \ldots, n_j)$ with components $n_i = a_i$ and $n_i = B_i$, $1 \le i \le j$, respectively. Therefore these two equations relate the normalizing constant of the network with the first j nodes and n customers with the constant $G_{j-1}(n-n_j)$. Note that these expressions can be also directly obtained by formula (7).

Equation (8.2) is similar to equation (11) for infinite capacity queues. It relates the normalizing constant of the network with the first j nodes and n customers with $G_{j-1}(n-a_j)$ and $G_j(n-1)$, corresponding to the two following cases:

(a) a_j customers in node j, which is its minimum queue length, and the remaining $n-a_j$ customers in the first $j-1$ nodes,

(b) at least one customer in node j and the remaining $n-1$ customers in the first j nodes (node j included).

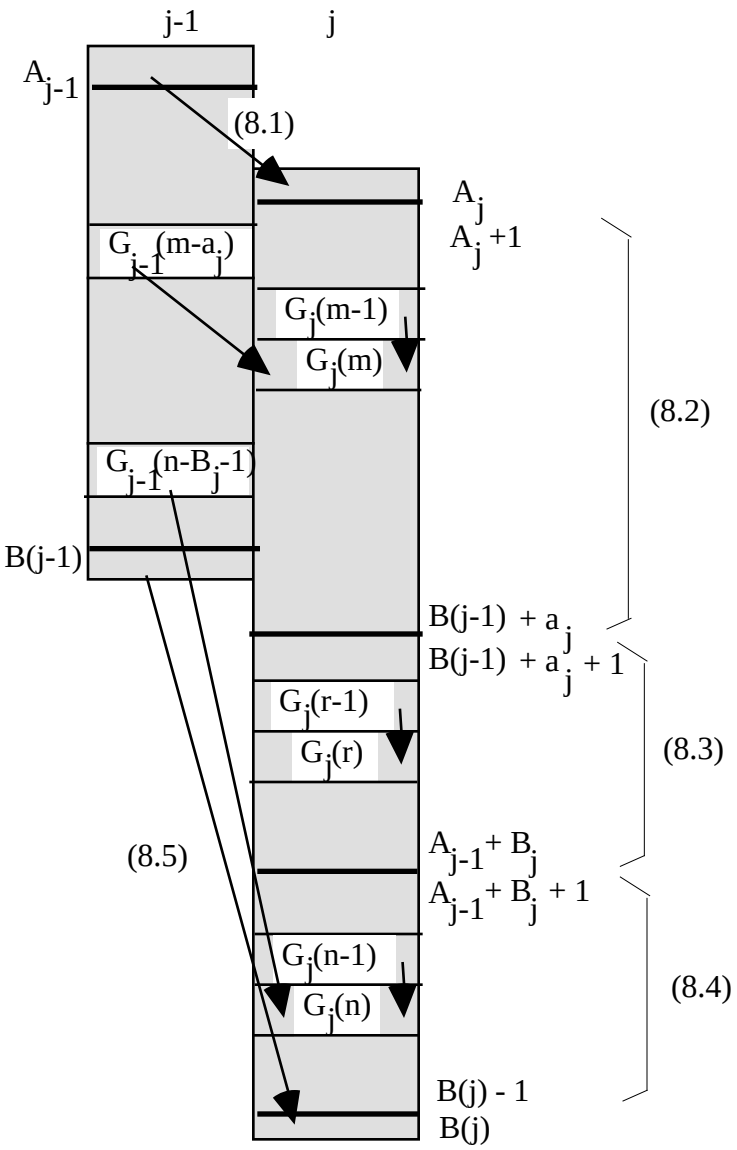

Figure 2 - Scheme of recursive equations (8.1)-(8.5) for function $G_j(n)$, $A_j \le n \le B(j)$, $1 \le j \le M$, for case (i).

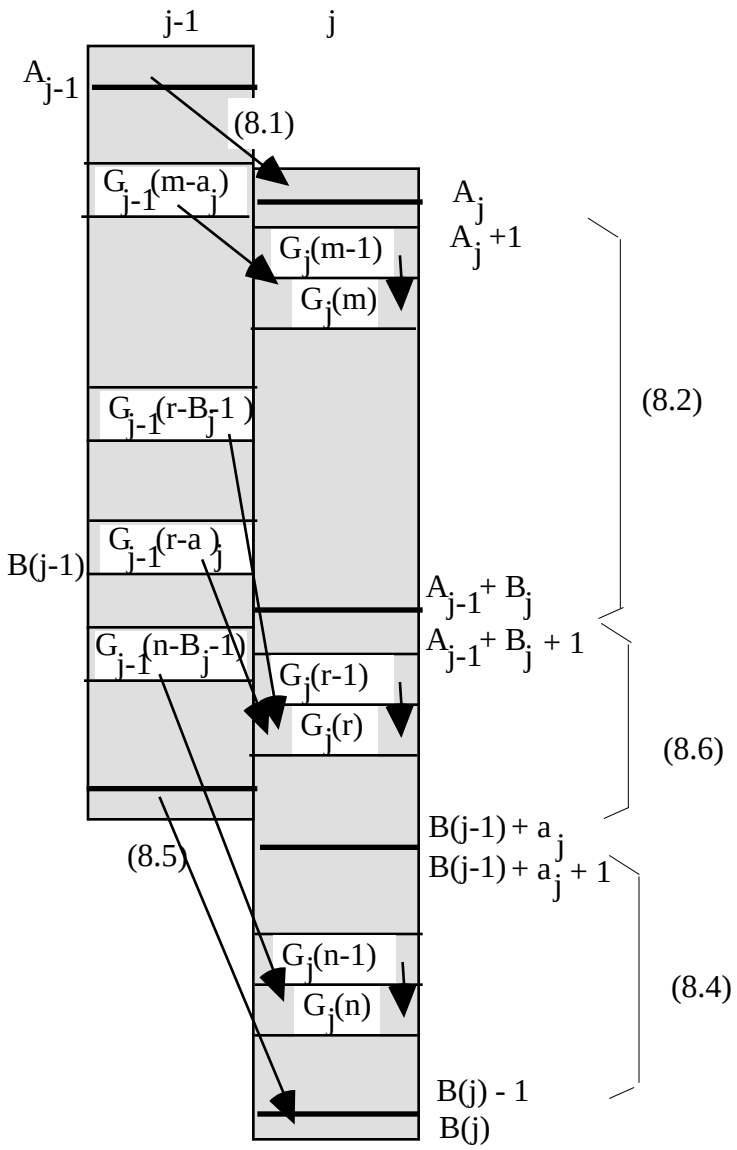

Figure 3 - Scheme of recursive equations (8.1), (8.2) and (8.4)-(8.6) for function $G_j(n)$, $A_j \le n \le B(j)$, $1 \le j \le M$, for case (ii).

Case (a) and case (b) correspond respectively to the first and to the second term of the right hand side of equation (8.2).

Equation (8.3) has the same meaning of equation (8.2), but without the term corresponding to case (a). Indeed, the number of customers is $n > a_j + B(j-1)$ and such that node j cannot attain its minimum queue length a_j even if all the first $j-1$ nodes have the maximum queue length (i.e., with $B(j-1)$ customers in the first $j-1$ nodes).

Equation (8.4) applies when the number of customers exceed the minimum overall capacity of the first j nodes (A_{j-1}) and node j capacity (B_j), i.e., for $n > A_{j-1} + B_j$. This

equation is similar to equation (8.3). Then the term corresponding to case (b) is obtained by considering the two following components:

(b1) at least one customer in node *j* and the remaining *n-1* customers in the first *j* nodes, given by the term $\rho_j G_j(n-1)$. Note that the distribution of these *n-1* customers includes the states where $n_j = B_j$. Hence the overall distribution of the *n* customers includes unfeasible states where $n_j = B_j+1$ and the remaining $n-B_j-1$ customers are in the first *j-1* nodes;

(b2) we discard these unfeasible states. This corresponds to the negative term $-\rho_j^{B_j+1} G_j(n-B_j-1)$.

Equation (8.6) can be interpreted as equation (8.4) by noting that the states where node *j* shows its minimum queue length ($n_j=a_j$) are feasible, which correspond to case (a) above. Hence the contribution of these states to function $G_j(n)$ is given by the first term of the right hand side of equation (8.6).

The pseudo-code of the algorithm is presented in the Appendix.

COMPUTATIONAL COMPLEXITY

The proposed algorithm has a polynomial time computational complexity in the number of network components. It requires O(*M N*) operations, and more precisely O(*MC*), where $C = \max \{B_i-a_i , 1 \le i \le M\}$.

It is easy to verify that this computational complexity is lower than that of the other methods proposed in the literature.

Akyildiz and von Brand [1] provide a method to evaluate the throughput, based on the computation of the normalising constant by a convolution algorithm. This method requires a total of $O(M^2)$ convolutions, each one of O(*N*) operations, so leading to a overall computational complexity of $O(M^2N)$.

Perros [16] proposes a recursive procedure to evaluate the constant *G* which can be applied only when $a_i = 0$, $1 \le i \le M$. This method is based on the following recursive equations:

$$G_j(n) = G_{j-1}(n) + \rho_j G_j(n-1) \qquad 1 \le n \le B_j$$

$$G_j(n) = G_{j-1}(n) + \rho_j G_j(n-1) - \rho_j^{B_j+1} \Phi_j\left(n-B_j-1\right) \qquad n > B_j$$

for $2 \le j \le M$, with $G_j(0)= 1$, for $1 \le j \le M$, $G_1(n)= \rho_1^n$ for $1 \le n \le B_1$, $G_1(n)=0$ for $n>B_1$, and where $\Phi_j(k) = \frac{1}{k} \sum_{s=1}^{j-1} \sum_{t=1}^{k} \rho_s^t \Phi_j(k-t)$, $1 \le k \le N$ and $\Phi_j(0)=1$. Note that the first equation is

identical to equation (11) for networks with infinite capacity queues [6]. In the second equation, function Φ_j is used to discard the terms corresponding to the unfeasible states. The computation of function Φ_j requires O(MN) operations for each recursive step. The number of these steps is O(MN), and more precisely O(MD), where D = max {NB_i, $1 \le i \le M$}. Hence the overall computational complexity of this method is O(M^2ND).

As concerns numerical stability of the proposed algorithm, for networks with infinite capacity queues it is known that depending on the choice of the parameters, some values of function $G_j()$ may exceed the floating point range of some computers, or for constant near to the lower magnitude limit of the floating point range, one can observe truncation errors which lead to inaccuracy in the computation. Some static and dynamic scaling techniques have been proposed to substantially reduce this problem, although it cannot be completely eliminated [12]. MVA algorithm avoids the computation of the normalizing constants and their associated numerical problems.
As concerns the proposed convolution algorithms for networks with finite capacity queues we cannot exclude that numerical problems can be observed depending on the parameters values. In order to alleviate this problem static and dynamic techniques can be adopted. Moreover, note that an MVA algorithm which directly computes system related quantities, so avoiding possible overflow or underflow conditions cannot be applied to closed product-form networks with blocking and arbitrary topology. In [7] an MVA algorithm is presented for the special case of cyclic topology networks.

4. Performance indices

In this section we derive closed-form expressions for the marginal queue length probability and some average performance indices.
First we derive an intermediate result which states the relation between the functions G_{M-1} and G_M. Then we apply this result to derive the marginal queue length distribution of each node.
Let $d_M = B_M - a_M + 1$ and $p = \left\lfloor \left(m - A_{M-1} - 2d_M + 1 \right) / d_M \right\rfloor$.

Lemma 1
i) if $A_{M-1}+B_M > B(M-1)+a_M$:

$$G_{M-1}(A_{M-1}) = \frac{1}{\rho_M^{a_M}} G_M(A_M) \qquad (12.1)$$

for $A_{M-1}+1 \le m \le B(M-1)$

$$G_{M-1}(m) = \frac{1}{\rho_M^{a_M}} \left[G_M(m + a_M) - \rho_M G_M(m + a_M - 1) \right] \tag{12.2}$$

ii) if $A_{M-1}+B_M \le B(M-1)+a_M$

for $m=A_{M-1}$, $G_{M-1}(m)$ is given by formula (12.1);
for $A_{M-1}+1 \le m \le A_{M-1}+B_M+a_M$, $G_{M-1}(m)$ is given by formula (12.2);
for $A_{M-1} + B_M - a_M + 1 \le m \le B(M-1) + a_M - B_M - 1$
if $m > A_{M-1}+(p+1)d_M$ then

$$G_{M-1}(m) = \frac{1}{\rho_M^{a_M}} \sum_{i=0}^{p+1} \rho_M^{id_M} \left[G_M(m + a_M - id_M) - \rho_M G_M(m - 1 + a_M - id_M) \right] \tag{13.1}$$

if $m=A_{M-1}+(p+1)d_M$ then

$$G_{M-1}(m) = \tag{13.2}$$
$$= \frac{1}{\rho_M^{a_M}} \sum_{i=0}^{p} \rho_M^{id_M} \left[G_M(m + a_M - id_M) - \rho_M G_M(m - 1 + a_M - id_M) \right] + \rho_M^{(p+1)d_M} G_M(A_M)$$

for $B(M-1)+a_M - B_M \le m \le B(M-1)-1$

$$G_{M-1}(m) = \frac{1}{\rho_M^{B_M+1}} \left[\rho_M G_M(m + B_M) - G_M(m + B_M + 1) \right] \tag{13.3}$$

$$G_{M-1}(B(M-1)) = \frac{1}{\rho_M^{B_M}} G_M(B(M)) \tag{13.4}$$

Proof

Consider $j=M$.
Case i)
From equations (8.1) and (8.2) of theorem 1, we derive $G_{M-1}(m)$ as a function of G_M, thus immediately obtaining equations (12.1) and (12.2), respectively. Note that G_{M-1} is completely defined for each value of m, $A_{M-1} \le m \le B(M-1)$.

Case ii)

From equations (8.1) and (8.5) we can immediately derive the equations (12.1) and (13.4). From equation (8.2) in case ii), where $A_M+1 \le n \le A_{M-1}+B_M$, we derive equation (12.2) for $m=n+a_M$. Similarly, from equation (8.4) when $B(M-1)+a_M+1 \le n \le B(M)$, we derive equation (13.3) for $m=n-B_M-1$.

Finally, consider equation (8.3) where G_{M-1} appears twice. We derive $G_{M-1}(m)$ with $n=m-a_M$ as follows:

$$G_{M-1}(m) = \frac{1}{\rho_M^{a_M}} \left[G_M(m+a_M) - \rho_M G_M(m-1+a_M) \right] + \rho_M^{d_M} G_{M-1}(m-d_M)$$

where $d_M=B_M-a_M+1$. By recursively applying equation (8.3) p times until the argument of the last term is greater than $A_{M-1}+B_M-a_M$, we obtain

$$G_{M-1}(m) = \frac{1}{\rho_M^{a_M}} \sum_{i=0}^{p} \rho_M^{id_M} \left[G_M(m+a_M-id_M) - \rho_M G_M(m-1+a_M-id_M) \right] + \rho_M^{(p+1)d_M} G_{M-1}(m-(p+1)d_M)$$

where $p = \left\lfloor \left(m - A_{M-1} - 2d_M + 1 \right) / d_M \right\rfloor$.

If $m-(p+1)d_M=A_{M-1}$, then equation (8.1) applies and one obtains equation (13.2). Otherwise equation (8.2) holds, and leads to equation (13.1). ◊

Let $G_{M-\{j\}}(m)$ denote the normalizing constant for the network with all the nodes except node j and m customers, $1 \le j \le M$, $0 \le m \le N$.

Lemma 2

The relationships between G_{M-1} and G_M in lemma 1 also hold between $G_{M-\{j\}}$ and G_M for any node j of the network, with the following substitutions:

$$\begin{bmatrix} \rho_M \\ a_M \\ B_M \\ A_{M-1} \\ B(M-1) \end{bmatrix} \text{ is replaced by } \begin{bmatrix} \rho_j \\ a_j \\ B_j \\ A_M - a_j \\ B(M) - B_j \end{bmatrix}$$

Proof

The lemma proof is trivial by considering a reordering of the nodes so that node j is the last one. ◊

We shall now use this result to prove the following theorem.

Theorem 2

The marginal queue length probability density of each node j can be written as a function only of $G_M(n)$, as follows:

for $a_j \le k \le B_i$, $1 \le j \le M$,

if $A_M - B_M - a_j > B(M) - B_j + a_j$ then

$$prob\{n_j = k\} = \frac{\rho_j^{k-a_j}}{G_M(N)}\left[G_M(N-k+a_j) - \rho_j G_M(N-k+a_j-1)\right] \qquad (14)$$

if $A_M - B_M - a_j < B(M) - B_j + a_j$, then

$$prob\left\{ n_j = k \right\} = \rho_j^k \frac{G_{M-\{j\}}(N-k)}{G_M(N)} \qquad (15)$$

where $G_{M-\{j\}}(m)$ is given by lemma 2.

Proof

Form the definition of the product-form of the steady-state distribution $\pi(\vec{n})$ (given by formula (1)), we can derive equation (15) for both cases.

Moreover, if $A_M - B_M - a_j > B(M) - B_j + a_j$, then by substituting in equation (15) equations (12.1) and (12.2) considered for node j as stated by lemma 2, we obtain equation (14).

◊

Corollary

The mean queue length of node j if $A_M - B_M - a_j > B(M) - B_j + a_j$ can be written as follows:

$$E[n_j] = \frac{1}{G_M(N)} \sum_{k=a_j}^{B_j} k \rho_j^{k-a_j} [G_M(N-k+a_j) - \rho_j G_M(N-k+a_j-1)]$$

Note that if $A_M - B_M - a_j \le B(M) - B_j + a_j$, then the mean queue length of node j can be derived by considering the marginal queue length distribution given by theorem 2.

Other performance indices of node j, including node utilization U_j, throughput X_j and average response time T_j in queueing networks with finite capacity queues depend on the probability that node j is blocked. Let PB_j denote this blocking probability of node j, $1 \le j \le M$, whose definition depends on the blocking type [2]. The other performance indices for single server nodes are defined as follows:

$$U_j = 1 - prob\{n_j = 0\} - PB_j, \qquad X_j = U_j \mu_j, \quad T_j = E[n_j] / X_j$$

Another interesting performance index is the average busy period during which node j is full, denoted by b_j. This busy period is defined as the time from an arrival that makes node j full to the next departure from node j [8].

Let $v_j = \sum_{i=1, i \neq j}^{M} \mu_i p_{ij}$, $1 \le j \le M$.

Theorem 3

The busy period of node j, $1 \le j \le M$, denoted by b_j , can be written as a function of $G_M(n)$ as follows:

if $A_M - B_M - a_j > B(M) - B_j + a_j$ then

$$b_j = \frac{\Delta_j}{G_M(N - B_j + a_j + 1) - \rho_j G_M(N - B_j + a_j)} \frac{\rho_j}{v_j} \tag{16}$$

where

$\Delta_j = G_M(A_M)$ if $N - B_j = A_M - a_j$

$\Delta_j = G_M(N - B_j + a_j) + \rho_j G_M(N - B_j + a_j - 1)$ if $N - B_j > A_M - a_j$

If $A_M - B_M - a_j \le B(M) - B_j + a_j$,

when the non-empty condition holds and $B(M) \ge N + 2$, then

$$b_j = \frac{\rho_j G_M(N) - G_M(N+1)}{\rho_j G_M(N+1) - G_M(N+2)} \frac{\rho_j}{v_j} \tag{17}$$

when the non-empty condition does not hold, then

$$b_j = \frac{G_{M-\{j\}}(N - B_j)}{G_{M-\{j\}}(N - B_j + 1)} \quad \frac{\rho_j}{v_j} \tag{18}$$

where $G_{M-\{j\}}$ is given by lemma 2.

Proof

Let b_j be the mean busy period of order B_j -1 of the node j. It is defined as the time between the arrival time of a customer that finds exactly B_j -1 customers in it and the next departure time of a customer that leaves less than B_j customers in the node [8].

Let S denote the set of states in which node j is full:

$$S = \{\vec{n} \quad E \mid n_j = B_j\}$$

Let S' denote its complementary set. We can compute the mean busy period as the mean sojourn time in S, as follows:

$$\pi(S) = h_{SS'} b_j$$

where $h_{SS'}$ is the ergodic flow rate from S to S' and $\pi(S)$ is the steady-state probability of set S. We can rewrite:

$$\sum_{\vec{n} \quad E:\, n_j = B_j} \pi(\vec{n}) = \sum_{\vec{n} \quad E:\, n_j = B_j - 1} \pi(\vec{n})\, v_j \quad b_j$$

where $v_j=\mu_{j-1}$ for cyclic topology and $v_j = \sum_{i=1, i \neq j}^{M} \mu_i p_{ij}$ for arbitrary topology.

By substituting the product-form (1) of $\pi(\vec{n})$ and after some algebra, we obtain:

$$(\rho_j)^{B_j} \sum_{(n_1,\ldots,B_j,\ldots,n_M) \quad E} \prod_{h=1,\, h\neq j}^{M} (\rho_h)^{n_h} =$$

$$= (\rho_j)^{B_j - 1} \sum_{(n_1,\ldots,B_j - 1,\ldots,n_M) \quad E} \prod_{h=1,\, h\neq j}^{M} (\rho_h)^{n_h} \; v_j \quad b_j$$

The summations in both sides of this equation define the two normalizing constants $G_{M-\{j\}}(N-B_j)$ and $G_{M-\{j\}}(N-B_j+1)$ of the network with all the nodes except node j and with $N-B_j$ and $N-B_j+1$ customers, respectively. We thus obtain formula (18).

By applying lemma 2 we can express b_j as a function of G_M for the two cases, thus obtaining formulas (16) and (17). ◊

Note that in the last case $B(M) \geq N+2$ because we also have to compute $G_M(n)$, for $n = N+1, N+2$.

4. Examples

In this section we present two examples to illustrate the application of the proposed convolution algorithm for product-form queueing networks with blocking.

Consider a queueing network which satisfies the condition of product-form in case I. In particular, we assume that the network is a central server model with $M=8$ service centers and $N = 50$ customers and the following routing probabilities: $p_{1j}=1/7$, $2 \leq j \leq M$, $p_{i1}=1$, $2 \leq i \leq M$. The blocking mechanism is *RS-RD*. Table I shows the values of the queue capacities and the service rates. The service rates are expressed in customers per units of time.

The number of states of the underlying Markov model is 11210.

By applying the proposed convolution algorithm, we compute the normalizing constant $G_8[50]= 0.0126$ with O(80) operations and we derive the average performance indices as shown in Table II.

The second example is a cyclic queueing network with *BBS* blocking which satisfies the condition of product-form in case II with $M = 10$ service centers and $N = 78$ customers. The network thus satisfies the non-empty condition. Table III shows the network parameters, i.e., the finite capacity of the queue and the service rates.

The number of states of the network Markov process is 715. The convolution algorithm requires O(40) operations and it computes the normalising constant $G_{10}[78] = 3.15\ e^{32}$, and the average performance indices shown in Table IV.

i	B_i	μ_i
1	10	5
2	8	1
3	8	1
4	6	2
5	6	3
6	10	1
7	5	2
8	6	3

Table I - Parameters of example 1: queue capacities and service rates for each node.

i	U_i	X_i	$E_i[n_i]$	T_i
1	0.5401	2.7006	9.734	3.604
2	0.2132	0.2132	7.587	35.585
3	0.2132	0.2132	7.587	35.5585
4	0.2118	0.4237	4.780	11.279
5	0.2008	0.6025	3.342	5.547
6	0.2132	0.2132	9.587	44.966
7	0.2097	0.4194	3.834	9.141
8	0.2033	0.6099	3.546	5.814

Table II - Results of example 1: utilization, throughput, mean queue length and mean response time for each node.

i	B_i	μ_i
1	8	4
2	10	4
3	7	5
4	7	3
5	9	3
6	8	2
7	9	3
8	9	3
9	7	5
10	8	2

Table III - Parameters of example 2: queue capacities and service rates for each node.

i	U_i	X_i	$E_i[n_i]$	T_i
1	0.2315	0.9261	7.304	7.886
2	0.2315	0.9261	9.717	10.492
3	0.1852	0.9261	6.717	7.253
4	0.3087	0.9261	6.783	7.324
5	0.3087	0.9261	8.597	9.282
6	0.4630	0.9261	7.597	8.202
7	0.3087	0.9261	8.304	8.966
8	0.3087	0.9261	8.597	9.282
9	0.1852	0.9261	6.597	7.123
10	0.4630	0.9261	7.783	8.403

Table IV - Results of example 2: utilization, throughput, mean queue length and mean response time for each node.

5. Conclusions

A convolution algorithm for queueing networks with blocking and product-form solution has been proposed considering different blocking mechanisms introduced in the literature. The algorithm is based on a set of recursive equations. As a consequence, this class of networks can be analysed with a polynomial time computational complexity in the number of network components, i.e., the number of service centers and the number of customers. Closed-form expressions for the marginal queue length distribution of each service center and for average performance indices have been presented. The method has been presented for single class product-form queueing networks with blocking. We are currently extending this method to multiclass product-form networks.

Appendix

Figure 4 presents the pseudo-code of the convolution algorithm to compute the normalizing constant G.

[Computation of the values a_j , A_j,$B(j)$]

FOR j:=1 TO M DO

$$a_j := \max\left(0, N - \sum_{i=1,\, i\neq j}^{M} B_i\right);$$

$$A_j := \sum_{i=1}^{j} a_i ;$$

$$B(j) := \sum_{i=1}^{j} B_i ;$$

[initialisation]

FOR n :=a_1 to B_1 DO

$$G_1(n) := \rho_1^n ;$$

[computation of the functions $G_j(n)$]

FOR j:=2 TO M DO

BEGIN

MIN := min (A_{j-1}+B_j, $B(j-1)$+a_j);

MAX := max (A_{j-1}+B_j, $B(j-1)$+a_j;);

$$G_j(A_j) := \rho_j^{a_j} G_{j-1}(A_{j-1});$$

FOR n := A_j +1 TO min(*MIN*, N) DO

$$G_j(n) := \rho_j^{a_j} G_{j-1}(n - a_j) + \rho_j G_j(n-1);$$

FOR n := *MIN* +1 TO min(*MAX* ,N) DO

IF *MIN* = $B(j-1)$+a_j

THEN

$$G_j(n) := \rho_j G_j(n-1);$$

ELSE

$$G_j(n) := \rho_j^{a_j} G_{j-1}(n - a_j) - \rho_j^{B_j+1} G_{j-1}(n - B_j - 1) + \rho_j G_j(n-1)$$

FOR n :=*MAX* +1 TO min($B(j)$-1, N) DO

$$G_j(n) := -\rho_j^{B_j+1} G_{j-1}(n - B_j - 1) + \rho_j G_j(n-1);$$

IF $N > B(j)$ THEN $G_j(B(j)) := \rho_j^{B_j} G_{j-1}(B(j-1))$

END

Fig. 4. Convolution Algorithm

References

[1] I.F.Akyildiz and H. von Brand, Computational Algorithms for Networks of Queues with Rejection Blocking, Acta Informatica, 26, (1989) 559-576.

[2] S.Balsamo and V.De Nitto Personè, Closed queueing networks with finite capacities: blocking types, product-form solution and performance indices, Performance Evaluation, vol.12, n.4 (1991) 85-102.

[3] S.Balsamo and V.De Nitto Personè, A Survey of product-form queueing networks with blocking and their equivalences, Annals of Operations Research, 48, (1994) 31-61.

[4] S.Balsamo, Properties and analysis of queueing network models with finite capacities, in *Performance Evaluation of Computer and Communication Systems*, Lecture Notes in Computer Science, 729, Springer-Verlag, 1994.

[5] F.Baskett, K.M.Chandy, R.R.Muntz and G.Palacios, Open, closed, and mixed networks of queues with different classes of customers, J. ACM, 22, 248-260, 1985.

[6] J.P.Buzen, Computational Algorithms for Closed Queueing Networks with exponential servers, Comm. ACM, 16, 527-531, 1973.

[7] M.C.Clò, MVA for Product-Form Cyclic Queueing Networks with RS Blocking, *Proc. Third International Workshop on Queueing Networks with Finite Capacity*, Bradford, UK, 6th-7th July, 1995.

[8] H.Daduna, Busy Periods for Subnetworks in Stochastic Networks: Mean Value Analysis, J. ACM, 35, 3, 668-674, July 1988.

[9] W.J.Gordon and G.F.Newell, Cyclic queueing systems with restricted queues, Oper. Res., 15, 286-302, 1976.

[10] A.Hordijk and N. van Dijk, Networks of queues with blocking, in *Performance '81*, (Ed. K.J. Kylstra) North Holland, 1981, 51-65.

[11] F.P.Kelly, *Reversibility and Stochastic Networks*, Wiley (1979).

[12] S.S. Lavenberg , *Computer Performance Modelling Handbook*, Academic Press, New York, (1983).

[13] R.A. Marie, An approximate analytical method for general queueing networks, IEEE Trans. Softw. Eng., SE-5:530-538, 1979.

[14] R.D.Nelson, The mathematics of product-form queueing networks, ACM Computing Surveys, Vol. 25, 3 (1993).

[15] R.O.Onvural, Survey of Closed Queueing Networks with Blocking, ACM Computing Surveys, Vol. 22, 2 (1990) 83-121.

[16] H.Perros, *Queueing Networks with Blocking*, Oxford, 1994.

[17] M. Reiser and M.Kobayashi, Queueing networks with multiple closed chains: Theory and computational algorithms, IBM J. Res. and Develop. 19 (May 1975) 238-294.

[18] M.Reiser and S.S. Lavenberg, Mean Value Analysis of Closed Multichain Queueing Networks, J. ACM, 27, 2, 313-322, April 1980.

[19] N. van Dijk, On ‘stop = repeat’ servicing for non-exponential queueing networks with blocking, J. Appl. Prob., 28 (1991) 159-173.

[20] N.van Dijk, ‘Stop = recirculate’ for exponential product form queueing networks with departure blocking, Oper. Res. Lett., 10, 343-351, 1991.