APPROXIMATION ANALYSIS OF OPEN ACYCLIC EXPONENTIAL QUEUEING NETWORKS WITH BLOCKING

HYO-SEONG LEE

Kyung Hee University, Seoul, Korea

STEPHEN M. POLLOCK

University of Michigan, Ann Arbor, Michigan

(Received July 1987; revisions received April 1988, March 1989; accepted September 1989)

An arbitrary configuration of an open queueing network with exponential service times and finite buffers is analyzed. We offer an iterative procedure for approximating the marginal occupancy probabilities for each queue of the system. The method decomposes the queueing network into individual queues and analyzes each in isolation using information from only its nearest neighbors. Based upon the SIMP approximation previously used for tandem queues, it replaces each server's service time with a clearance time, which includes blocking, and each server's arrival rate by an equivalent *acceptance* rate. The procedure is easy to implement and requires modest memory and computer time. Extensive numerical experiments, performed for various topologies, yield accurate results compared with those obtained by exact or simulation methods.

A queueing network, a set of arbitrarily connected queues, can represent many processes of interest in manufacturing systems, computer systems, telecommunications, etc. If the buffer space between servers is infinite and service times at each queue are exponential, these networks can be analyzed exactly by Jackson's decomposition method (see Jackson 1963). Jackson's method, however, ignores an important feature of many real queueing systems, i.e., blocking due to the finiteness of buffer space. In this case, the product form property does not hold, and very complicated conditions of dependency exist among the queues. The number of states needed for an exact numerical analysis grows combinatorially with the number of queues and buffers. For this reason, most analyses are based on approximation or simulation methods.

There are various configurations of queueing networks with blocking. The tandem (or serial) network with exponential servers, the most basic structural configuration, has been studied by Hillier and Boling (1967), Caseau and Pujolle (1979), Latouche and Neuts (1980), Boxma and Konheim (1981), Altiok (1982), Perros and Altiok (1986), Bocharov and Rokhas (1980), Brandwajn and Jow (1985) and Foster and Perros (1980). For nonexponential service times, Gershwin (1987) and Choong and Gershwin (1987) present algorithms for special service time distributions, representing the probabilistic failure and repair of the server. The SIMP approximation procedure of Pollock, Birge and Alden (1985) allows for general service time distributions.

Analysis of other configurations, particularly split and merge, have been reported by Boxma and Konheim (1981), Altiok and Perros (1986) and Lee and Pollock (1989). With the exception of allowing some servers to have general service time distributions in Lee and Pollock, these all assume exponential service times.

The general system, which is a combination of tandem, split and merge configurations, is the most complicated to analyze. Takahashi, Miyahara and Hasegawa (1980) assumed that *effective* service times follow an exponential distribution, and developed a set of simultaneous nonlinear equations that must be solved to get performance measures. Labetoulle and Pujolle (1980) and Kerbache and Smith (1986), allowing for nonexponential service times, use a diffusion approximation that may restrict its validity (see Perros and Snyder 1986). Altiok and Perros (1987) use phase-type distributions for approximately characterizing effective service times. Their procedure appears to be restricted to small networks due to the inherent complexity of the phase-type mechanism. Recently, Perros and Snyder developed a similar algorithm, using a two-phase Coxian distribution to approximate

Subject classifications: Queues, applications: analysis of networks with blocking. Queues, limit theorems: approximation method for networks with blocking.

0030-364X/90/3806-1123 $01.25

effective service times, as an improvement over the work by Altiok and Perros (1987). However, this algorithm is not accurate in important boundary cases, such as when queues that receive exogenous inputs have very large buffers. In these previous analyses, as with the work presented in this paper, the networks are restricted to have no feedback loops.

In this paper, we present an approximation method for analyzing the general configuration of an open queueing network with blocking. This algorithm is based on two earlier algorithms; one proposed by Pollock, Birge and Alden for tandem queues and the other by Lee and Pollock for merge queues. This new algorithm solves large networks quickly, and yields robust and accurate results.

1. DESCRIPTION OF THE NETWORK AND FORMULATION OF THE PROBLEM

The network that we consider is identical to that in Altiok and Perros (1987) and Perros and Snyder except that we also allow external arrivals at any server. It consists of the set $\{i: i = 1, 2, \ldots, M\}$ of single server queues, connected arbitrarily via arcs (i, j) with the restriction that there is no directed cycle. This restriction is made to avoid feedback loops, with which the approximation method would have difficulty. Since there is no directed cycle, we can number each queue in such a way that every arc (i, j) has i less than j. The service time at queue i follows an exponential distribution with rate μ_i and external arrivals to queue i are independent Poisson processes with rate λ_i. The capacity of the ith queue, including the one in service, is N_i and its buffer size is $N_i - 1$. Units at each queue are served in a FIFO manner. If an external arrival encounters a queue i when it is full, the arrival is simply lost. A unit that has completed service at queue i chooses destination queue j with routing probability r_{ij}. The probability that a unit leaves the queueing system after completing service at queue i is r_{i0}. Figure 1 shows an example that consists of four queues.

The blocking mechanism considered in this paper is as follows. Suppose that a unit has just finished service at queue i and the next service required is at queue j. If the buffer of queue j is full, the unit cannot leave the ith server. During this time the ith server cannot serve other units that might be waiting in its buffer: the ith server is said to be *blocked* and the jth queue is *blocking*. For example, in Figure 1 queue 1 cannot be blocking and queue 4 cannot be blocked.

Note that a queue may be simultaneously blocking more than one *upstream* queue at one time. It is assumed that the blocked units enter the destination queue on a *first blocked, first enter* basis.

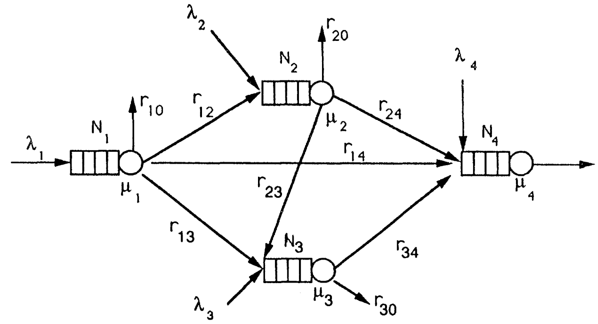

Figure 1. The general four-node network.

Since a unit blocked by queue j is ready to proceed to queue j whenever there is a space in queue j, it is effectively waiting in line to be served by server j. Therefore, we can interpret the server position of a blocked unit to be part of the buffer capacity of the blocking queue. Hence, the capacity of queue j is augmented by the number of upstream queues directly connected to it, so that its effective capacity is $N_j + k$. In the next section, we develop a procedure that exploits the augmented buffer size for each queue.

2. ANALYSIS OF THE MODEL

2.1. Approach and General Relationships

The approximation algorithm presented here produces the marginal steady-state occupancy probabilities for each queue. To do this, we analyze each individual queue in isolation using information only from its nearest neighbors. This requires the consideration of two parameters: 1) the *clearance time*, which has two components: the actual service time plus a term due to the occasional and probabilistic delay caused by blocking; and 2) an *effective interarrival rate*.

We first assume that:

a. Arrivals from queue i to queue j are Poisson with effective rate λ_{ij}^*, as long as the ith queue is not blocked by queue j. When the ith queue *is* blocked by queue j, there is no arrival from queue i to queue j.
b. The clearance time (having two components) at queue i is exponentially distributed with effective rate μ_i^*.
c. A unit at server i, at the instant service is completed, sees destination queues in the steady state.

Clearly, these assumptions are far different from what actually happens in the system.

2.2. Notation

We define, where, unless otherwise stated, the index i always runs from 1 to M:

- $\mu_i \equiv$ the service rate at server i, excluding any delay due to blocking,
- $T_i \equiv$ the clearance time for server i, that is, the time between when a unit enters service in queue i and when it leaves queue i,
- $F_i =$ the predecessor set of queue $i \equiv \{\nu$: queue ν can pass units directly to queue $i\}$,
- $k_i = |F_i| \equiv$ the number of upstream queues directly connected to queue i,
- $B_i =$ the successor set of queue $i \equiv \{\nu$: queue ν can receive units directly from queue $i\}$,
- $\overline{\lambda}_{ij} \equiv$ the flow rate from queue i to queue j, $i = 1, \ldots, M-1, j \in B_i$,
- $\overline{\lambda}_{0i}\ (\overline{\lambda}_{i0}) \equiv$ the flow rate from outside the system to queue i (from queue i to outside the system),
- $\lambda^*_{ij} \equiv$ the arrival rate to queue j from queue i as long as queue i is not blocked by queue j, $i = 1, \ldots, M-1, j \in B_i$,
- $P_i(k) \equiv$ the steady-state probability that there are k units at queue i,
- $b_{ij}(n) \equiv$ the probability that n units are blocked by queue j including one at queue i, $j = 2, \ldots, M, i \in F_j$,
- $\alpha_{ij}(k) \equiv$ the conditional probability that, upon service completion at server i, a unit that has queue j as its destination sees k units at queue j, $j = 2, \ldots, M, i \in F_j$,
- $f_i \equiv$ the probability $\{i$th queue is full$\}$.

2.3. Analysis of the Model

2.3.1. Analysis Given Effective Arrival Rates and Expected Clearance Times

For the moment, let us assume that we know the effective arrival rates λ^*_{ij} and the expected clearance time $E(T_i)$ for each queue. Then queue 1 can be analyzed by using an $M/M/1/N_1$ model, with occupancy probabilities

$$P_1(j) = \frac{(1-\rho_1)\rho_1^j}{1-\rho_1^{N_1+1}}, \quad j = 0, 1, \ldots, N_1 \tag{1}$$

where $\rho_1 = \lambda_1 E(T_1)$. The probability that queue 1 is full is

$$f_1 = P_1(N_1). \tag{2}$$

It is more difficult to analyze queues that have directly connected upstream queues. In order to obtain the occupancy and blocking probabilities of such queues, we use the following procedure, developed in Lee and Pollock for merge queues.

Consider queue j, which has k_j directly connected upstream queues. When queue j is not blocking, define its state to be the number of units in queue j. If queue j is blocking, the state is defined to be $(N_j + n, \mathbf{v})$, $n = 1, \ldots, k_j$ where $N_j + n$ represents the number of units in queue j, including the ones blocked by queue j, and $\mathbf{v}$ is the n-component vector that represents the order of units which are being blocked. To obtain the blocking probability $b_{ij}(n)$, we must find the occupancy probabilities prob. $\{S_j = N_j + n, \mathbf{v}\}$ for each ordered state. Unfortunately, these occupancy probabilities can be obtained only by solving the original problem via the very large set of steady-state balance equations. However, as shown in Lee and Pollock, we can obtain these occupancy probabilities in terms of $P_j(N_j + n)$ by considering, for each queue in isolation, an equivalent aggregated state space, and its associated simple birth-and-death equations. The aggregated state is the number of units in queue j, *disregarding* the order of units being blocked, so that blocking states which have the same number of units are aggregated into one state. Let $\hat{\lambda}_j(i)$ denote the arrival rate to aggregated state $i+1$ from aggregated state i. The following theorems allow us to compute appropriate values for the $\hat{\lambda}_j(i)$, and to find occupancy probabilities for the original states. Proofs of these theorems are not presented; they are similar to those in Lee and Pollock, with the modest extension that the network here has external arrivals to queue j.

Theorem 1. *The occupancy probabilities* $P_j(i)$, $i = 1, 2, \ldots, N_j + k_j$ *for the aggregated states of (isolated) queue j are equivalent to those obtained by using the original states of (isolated) queue j if the arrival rates to the aggregated states are*

$$\hat{\lambda}_j(i) = \lambda^*_j \quad \textit{for } i = 0, \ldots, N_j - 1 \tag{3a}$$

$$\hat{\lambda}_j(N_j + i) = \frac{(i+1)\Omega_{i+1,j}}{\Omega_{i,j}} \quad \textit{for } i = 0, \ldots, k_j - 1 \tag{3b}$$

where

$$\lambda^*_j \equiv \sum_{k \in F_j} \lambda^*_{kj} + \lambda_j$$

$$\Omega_{i,j} \equiv \textit{sum of all i-tuplet products from set } \{\lambda^*_{\nu j}: \nu \in F_j\}$$

$$\Omega_{0,j} \equiv 1.$$

For example, if $F_4 = \{1, 2, 3\}$, then $k_4 = 3$ and

$$\Omega_{2,4} = \lambda^*_{14}\lambda^*_{24} + \lambda^*_{14}\lambda^*_{34} + \lambda^*_{24}\lambda^*_{34}$$

$$\Omega_{3,4} = \lambda^*_{14}\lambda^*_{24}\lambda^*_{34}.$$

Equation (3a) takes into account the fact that the nonblocking aggregate states $0, 1, \ldots, N_j - 1$ are identical to the original states. Each state sees, under Assumption a, an arrival rate that is the sum of the effective rates from upstream queues and external arrivals. Equation (3b) produces an appropriately weighted combination of effective-arrival rates for the aggregated blocking states $N_j + n$, $n = 0, 1, \ldots, k_j - 1$. This in turn produces, under Assumption a, correct marginal probabilities for the aggregated blocking states.

From Theorem 1, and Assumptions a and b, the occupancy probability $P_j(i)$ for the aggregate states can be found as

$$P_j(i) = P_j(0) \prod_{k=1}^{i} \frac{\hat{\lambda}_j(k-1)}{\mu_j^*} \quad i = 1, 2, \ldots, N_j + k_j \qquad (4a)$$

$$\sum_{i=0}^{N_j+k_j} P_j(i) = 1 \qquad (4b)$$

where $\mu_j^* = 1/E(T_j)$.

Note that (4a), which is based on Assumption a, requires that in our approximation the ratio of successive marginal probabilities $P_j(i+1)/P_j(i)$ is constant for the nonblocking states $i = 0, 1, \ldots, N_j - 1$.

Since the buffer space of each queue is augmented by the number of upstream queues directly connected to it, the probability that queue j is full is

$$f_j = \sum_{n=0}^{k_j} P_j(N_j + n). \qquad (5)$$

Once we obtain the occupancy probabilities of the aggregated states, we can find the occupancy probabilities of the original states by the following theorem.

Theorem 2. *The relationship between the occupancy probabilities of the original blocking states and those of aggregated blocking states is*

$$P_j(N_j + n, i_1 \ldots i_n) = \frac{\prod_{k=1}^{n} \lambda^*_{i_k j}}{n!\, \Omega_{n,j}} P_j(N_j + n)$$

$$\textit{for } n = 1, \ldots, k_j. \qquad (6)$$

2.3.2. Finding Effective Expected Clearance Times

Since we have values of $P_j(N_j + n, i_1 \ldots i_n)$ from (6), we can obtain $b_{ij}(n)$, the probability {n units are blocked by queue j including one at queue i}, by adding up the probabilities of the different orderings by which queue j is full and blocking n units including one at queue i

$$b_{ij}(n) = \sum_{i \in \{i_1 \ldots i_n\}} P_j(N_j + n, i_1 \ldots i_n).$$

Using Theorem 2, this gives

$$b_{ij}(n) = \frac{\lambda^*_{ij}\Omega_{n-1,j\setminus i}}{\Omega_{n,j}} P_j(N_j + n),$$

$$n = 1, \ldots, k_j, \quad i \in F_j \qquad (7)$$

where

$\Omega_{n-1,j\setminus i} \equiv$ sum of all $(n-1)$-tuplet products from the set $\{\lambda^*_{\nu j} : \nu \in F_j, \nu \neq i\}$.

From $b_{ij}(n)$, we can compute the conditional probability $\alpha_{ij}(k)$ that, upon service completion at server i, a unit which has queue j as its destination queue sees k units at queue j. From Assumption c in Section 2.1 and the fact that a unit cannot be served, and therefore cannot have *completed* service, at queue i if queue i is blocked by queue j, $\alpha_{ij}(N_j + n)$ is the conditional probability that there are $N_j + n$ units at queue j given that queue i is not blocked by queue j. Since the probability that queue i is blocked by queue j is $\sum_{n=1}^{k_j} b_{ij}(n)$

$$\alpha_{ij}(N_j + n) = \frac{P_j(N_j + n) - b_{ij}(n)}{1 - \sum_{m=1}^{k_j} b_{ij}(m)},$$

$$n = 0, \ldots, k_j - 1, \quad i \in F_j \qquad (8)$$

where $b_{ij}(0)$ is defined to be 0. Using this value of $\alpha_{ij}(N_j + n)$, we can obtain the expected clearance time at queue i given that queue j is the destination from

$$E(T_i \mid j) = \frac{1}{\mu_i} + \sum_{n=0}^{k_j - 1} (n+1)\alpha_{ij}(N_j + n) E(T_j). \qquad (9)$$

The first term in (9) is the expected service time at queue i. The second term is the expected delay due to blocking. If a unit whose destination is queue j sees n *other* units blocked by queue j at the instant of its service completion at queue i, it must wait n (independent) clearance times plus one residual clearance time at queue j before it feeds into queue j.

Since r_{ij} is the probability that queue j will be the destination queue, the unconditional expected clearance time for queue i is

$$E(T_i) = \sum_{j \in B_i} r_{ij} E(T_i \mid j). \qquad (10a)$$

If queue i has no directly connected downstream queues, i.e., $B_i = \varnothing$, $E(T_i)$ simply becomes

$$E(T_i) = 1/\mu_i. \tag{10b}$$

2.3.3. Finding the Flow Rates

Once $E(T_i)$ is obtained, with λ_{ij}^* already available, the full probability f_i can be calculated using $M/M/1/N_i + k_i$ analysis. Using these values of f_i, $\bar{\lambda}_{ij}$, the flow rate from queue i to queue j, can be calculated from the following relationships:

1. Since units can enter queue i from outside the system only when queue i is not full, the flow rate from outside is given by

$$\bar{\lambda}_{0i} = \lambda_i(1 - f_i). \tag{11}$$

2. The total flow rate into queue i, denoted by $\bar{\lambda}_i$, is

$$\bar{\lambda}_i = \sum_{k \in F_i} \bar{\lambda}_{ki} + \bar{\lambda}_{0i}. \tag{12}$$

3. By the conservation of average flow, the flow rate from queue i to queue j (or, when $j = 0$, outside of the system) is given by

$$\bar{\lambda}_{ij} = \bar{\lambda}_i r_{ij}. \tag{13}$$

2.3.4. Finding Updated Effective Arrival Rates

Using the values of $b_{ij}(n)$ and $\bar{\lambda}_{ij}$ obtained from (7) and (13), respectively, updated values of λ_{ij}^* can be calculated from

$$\lambda_{ij}^*\left(1 - \sum_{n=1}^{k_j} b_{ij}(n)\right) = \bar{\lambda}_{ij}, \tag{14}$$

a conservation equation which yields the arrival rate λ_{ij}^* needed in order to produce the given value $\bar{\lambda}_{ij}$.

2.3.5. Iterative Approach

We have just shown that values of $b_{ij}(n)$ and $\bar{\lambda}_{ij}$ can be computed from λ_{ij}^* and $E(T_i)$; conversely, given $b_{ij}(n)$ and $\bar{\lambda}_{ij}$, *updated* values of λ_{ij}^* and $E(T_i)$ can be computed: This is the basis for an iterative approach to finding performance measures of interest. In particular, we are now in a position to describe an iterative procedure which produces the occupancy probabilities for each queue. Each iteration consists of two sets of calculations: The effective arrival rates λ_{ij}^* are calculated in forward order and occupancy probabilities and $E(T_i \mid j)$ are calculated in backward order.

If external arrivals occur at only the first queue, only a single set of calculations is needed. However, if more than one queue has external arrivals, a two-way analysis is unavoidable since the value of f_i changes. Details are in the algorithm described below, but, in general, flow rates from queue to queue are produced in a series of "forward" calculations, and occupancy probabilities and expected clearance times are found in a series of "backward" calculations. At the end of the backward analysis, a convergence condition check is made on the values of $E(T_i)$. If convergence does not occur, another iteration is performed.

Note that, in the forward analysis, occupancy probabilities of disaggregated states are not obtained, since the only occupancy probability computed is f_i from (3) and (4). Thus, the computational effort of the two-way analysis is not critically increased over that needed for the one-way analysis.

2.4. Approximation Algorithm

The analysis above is incorporated into the following iterative algorithm to obtain an approximate solution to the system's steady-state probabilities.

Step 0. (Setup). Set the values of λ_i, μ_i, N_i for $i = 1, \ldots, M$.

Step 1. (Initialization—the conditions here are as if all the queues are unblocked)

Set $E(T_i) = 1/\mu_i$ for $i = 1, \ldots, M$, $\rho_1 = \lambda_1 E(T_1)$.
Find f_1 using (1) and (2)
Set $\bar{\lambda}_1 = \lambda_1(1 - f_1)$
Find $\bar{\lambda}_{1j}$ using (13) for all $j \in B_1$
Set $\lambda_{1j}^* = \bar{\lambda}_{1j}$ for all $j \in B_1$
For $i = 2, M - 1$ do
  begin
    find $\hat{\lambda}_i(n)$ using (3)
    find $P_i(n)$ using (4)
    find f_i using (5)
    find $\bar{\lambda}_{0i}$ using (11)
    find $\bar{\lambda}_i$ using (12)
    find $\bar{\lambda}_{ij}$ using (13) for all $j \in B_i$
    set $\lambda_{ij}^* = \bar{\lambda}_{ij}$ for all $j \in B_i$
  end

Step 2. (Backward analysis—find $E(T_i \mid j)$ and the occupancy probabilities for each queue)

For $j = M, 2, -1$ do
  begin
    find $E(T_j)$ using (10)
    find $\hat{\lambda}_j(n)$ using (3) for $n = 0, \ldots, N_j + k_j - 1$
    find $P_j(n)$ using (4) for $n = 0, \ldots, N_j + k_j$
    find $b_{ij}(n)$ using (7) for all $i \in F_j$
    find $\alpha_{ij}(N_j + n)$ using (8) for all $i \in F_j$
    find $E(T_i \mid j)$ using (9) for $i \in F_j$
  end
Find $E(T_1)$ using (10)

Step 3. (Convergence check)
If $\max_i |\text{updated } E(T_i) - E(T_i)| \leq \epsilon$ go to Step 4. Else, go to Step 4.

Step 4. (Forward analysis—find λ^*_{ij} for each queue)
Set $\rho_1 = \lambda_1 E(T_1)$
Find f_1 using (1) and (2)
Set $\bar{\lambda}_1 = \lambda_1(1 - f_1)$
Find $\bar{\lambda}_{1j}$ using (13) for all $j \in B_1$
Find λ^*_{1j} using (14) for all $j \in B_1$
For $i = 2, M - 1$ do
begin
find $\hat{\lambda}_i(n)$ using (3)
find $P_i(n)$ using (4)
find f_i using (5)
find $\bar{\lambda}_{0i}$ using (11)
find $\bar{\lambda}_i$ using (12)
find $\bar{\lambda}_{ij}$ using (13) for all $j \in B_i$
find λ^*_{ij} using (14) for all $j \in B_i$
end
Go to Step 2.

Step 5. (Calculate occupancy probabilities)
For queue 2 through M, these have already been obtained in Step 2. For queue 1, find $P_1(n)$ using (1) for $n = 0, \ldots, N_1$.

Step 6. Stop.

If external arrivals occur only at the first queue, the algorithm becomes simplified because all $\bar{\lambda}_{ij}$ are completely determined by f_1 which is obtained at the end of the backward analysis. Thus, in this case, (3), (4), (5) and (11) in Steps 1 and 4 are not used.

3. COMPUTATIONAL RESULTS

In order to test the accuracy of our approximation method, the algorithm was implemented on an IBM 3090-400 and tested on a variety of problems. Tables I–IX give comparisons with three- to eight-node network problems in the literature, all of which have only one queue with external arrivals. Tables X and XI give comparisons for the cases that have external arrivals at more than one queue. In those cases where exact solutions have not been obtained, we use simulation results. In all cases the convergence criterion of Step 3 was $\epsilon = 0.00001$.

Table I gives comparisons for the triangular network of Figure 2, as reported in Takahashi et al. (1980) and Altiok and Perros (1987). Arrivals are at queue 1 with rate 1, and every queue has a buffer of size one. The routing probabilities are $r_{10} = 0$, $r_{12} = r_{13} = 0.5$ and $r_{23} = 1$. Comparisons are based on $P_1(N_1)$, which determines the throughput of the system because other queues do not have external arrivals. As seen in the table, our method performs better than Takahashi's and is comparable with Altiok and Perros' method. Also note that Altiok and Perros' method underestimates $P_1(N_1)$, the probability that queue 1 is full, if the service rates are balanced and low (e.g., $\mu = 1, 1.1, 1.2$) and overestimates it if the service rates are unbalanced and high (e.g., $\mu = 1, 2, 3$). This pattern suggests that it might have a larger error for very high or low service rates, even though in the intermediate range shown their method is accurate. On the other hand, our method appears to be more robust in that it shows

Table I
Approximations to $P_1(N_1)$ from Takahashi et al., Altiok and Perros, and the New Algorithm

μ_1	μ_2	μ_3	Exact	Altiok and Perros	Takahashi	New
1	1.1	1.2	0.55963	0.54698	0.58669	0.56301
1	1.2	1.4	0.54634	0.53736	0.57344	0.55020
1	1.3	1.6	0.53681	0.53049	0.56324	0.54094
1	1.4	1.8	0.52980	0.52541	0.55538	0.53404
1	1.5	2.0	0.52451	0.52153	0.54904	0.52876
1	1.6	2.2	0.52043	0.51850	0.54398	0.52462
1	1.7	2.4	0.51724	0.51608	0.53975	0.52133
1	1.8	2.6	0.51469	0.51411	0.53619	0.51866
1	1.9	2.8	0.51264	0.51250	0.53318	0.51646
1	2.0	3.0	0.51096	0.51115	0.53058	0.51464

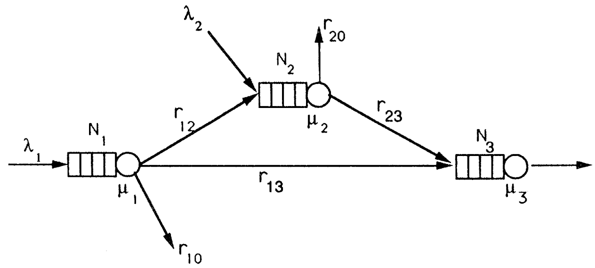

Figure 2. The three-node network.

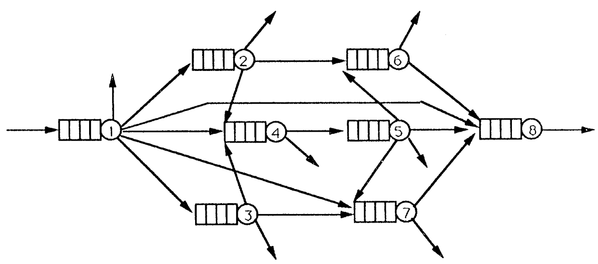

Figure 3. The eight-node network topology (the arrows indicate nonzero flows).

Table II
Summary of Comparisons With the Approximations of Altiok and Perros, and Perros and Snyder

Network Configuration	Measures[a]	Altiok and Perros	Perros and Snyder	New
3-node network (avg. of 9 problems)	Avg. abs. dev. (P)	0.0095	0.0082	0.0066
	Max. abs. dev. (P)	0.0322	0.0201	0.0152
	Avg. rel. err. (P)	0.0545	0.0547	0.0311
	Max. rel. err. (P)	0.1991	0.2132	0.1019
	Avg. rel. err. (L)	0.050	0.047	0.028
4-node network (avg. of 4 problems)	Avg. abs. dev. (P)	0.0213	0.0198	0.0135
	Max. abs. dev. (P)	0.0442	0.0442	0.0257
	Avg. rel. err. (P)	0.0679	0.0638	0.0433
	Max. rel. err. (P)	0.1691	0.1401	0.1045
	Avg. rel. err. (L)	0.057	0.060	0.035
8-node network (avg. of 10 problems)	Avg. abs. dev. (P)	—[b]	0.014	0.007
	Max. abs. dev. (P)	—[b]	0.043	0.020
	Avg. rel. err. (P)	—[b]	0.079	0.045
	Max. rel. err. (P)	—[b]	0.338	0.229
	Avg. rel. err. (L)	—[b]	0.055	0.035

[a] P represents the marginal occupancy probability and L represents the expected queue length.
[b] Not available due to memory constraints.

a consistent pattern of overestimating $P_1(N_1)$ for all explored service rates, by fairly small deviations, i.e., 0.004 − 0.005 in absolute error.

The new approximation algorithm was also tested for nine other three-node network problems, as well as the four four-node network problems and ten eight-node network problems analyzed in Altiok and Perros (1987) and Perros and Snyder (1986). Figures 1, 2 and 3 show the topologies of these networks. Tables III–IX present some numerical results selected from these problems, showing the average (and maximum) absolute deviations and average (and maximum) relative errors for the occupancy probabilities and average relative errors for the expected queue lengths with respect to exact or simulated values. Since Altiok and Perros' algorithm cannot solve the eight-node network due to memory limitations, we compare our results in this case with only those of Perros and Snyder's.

Table II is a summary of the comparisons with Altiok and Perros' and Perros and Snyder's for the averages of nine three-node network problems, four four-node network problems and ten eight-node network problems, for the various performance measures in Tables III–IX. As can be seen, our algorithm gives better results in both average (maximum) absolute deviation and average (maximum) relative error. Note that, while Perros and Snyder's algorithm is inaccurate if the first queue has infinite buffers (see, for example,

Table III
Comparisons With the Approximations of Altiok and Perros, and Perros and Snyder for a Three-Node Network

Measures[a]	Exact	Altiok and Perros	Perros and Snyder	New
$P_1(0)$	0.1078	0.1261	0.1248	0.1048
$P_1(1)$	0.0946	0.1096	0.1111	0.0938
$P_1(2)$	0.0836	0.0956	0.0961	0.0840
$P_1(3)$	0.0743	0.0835	0.0833	0.0752
$P_1(4)$	0.0662	0.0731	0.0724	0.0673
$P_1(5)$	0.0592	0.0639	0.0632	0.0603
L_1	8.6170	6.9948	6.9683	8.5424
$P_2(0)$	0.6231	0.6489	0.6490	0.6161
$P_2(1)$	0.2401	0.2294	0.2311	0.2399
$P_2(2)$	0.0919	0.0818	0.0805	0.0934
$P_2(3)$	0.0449	0.0399	0.0395	0.0506
L_2	0.5586	0.5127	0.5104	0.5784
$P_3(0)$	0.4563	0.4560	0.4561	0.4560
$P_3(1)$	0.2684	0.2608	0.2608	0.2679
$P_3(2)$	0.2753	0.2832	0.2831	0.2761
L_3	0.8190	0.8272	0.8271	0.8201
Avg. abs. deviation (P)		0.0102	0.0102	0.0018
Max. abs. deviation (P)		0.0258	0.0259	0.0070
Avg. rel. error (P)		0.0876	0.0880	0.0192
Max. rel. error (P)		0.1698	0.1744	0.1269
Avg. rel. error (L)		0.093	0.096	0.015

[a] P represents the marginal occupancy probability and L represents expected queue length; $\lambda = (0.8, 0, 0)$, $\mu = (1, 1, 1)$, $N = (\infty, 3, 2)$; $r_{10} = 0.2$, $r_{12} = 0.4$, $r_{13} = 0.4$, $r_{20} = 0.3$, $r_{23} = 0.7$; CPU time = 0.001 second.

Table IV
Comparisons With the Approximations of Altiok and Perros, and Perros and Snyder for a Three-Node Network

Measures[a]	Exact	Altiok and Perros	Perros and Snyder	New
$P_1(0)$	0.2154	0.2142	0.2135	0.2092
$P_1(1)$	0.7846	0.7858	0.7865	0.7909
L_1	0.7846	0.7858	0.7865	0.7909
$P_2(0)$	0.7051	0.7231	0.7241	0.6968
$P_2(1)$	0.2949	0.2770	0.2759	0.3032
L_2	0.2949	0.2770	0.2759	0.3032
$P_3(0)$	0.6123	0.6144	0.6158	0.6235
$P_3(1)$	0.3877	0.3856	0.3842	0.3765
L_3	0.3877	0.3856	0.3842	0.3765
Avg. abs. deviation (P)		0.0071	0.0081	0.0086
Max. abs. deviation (P)		0.0180	0.0190	0.0112
Avg. rel. error (P)		0.0170	0.0196	0.0207
Max. rel. error (P)		0.0607	0.0644	0.0289
Avg. rel. error (L)		0.023	0.025	0.022

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (3.0, 0, 0)$, $\mu = (1, 1, 1)$, $N = (1, 1, 1)$; $r_{10} = 0.2$, $r_{12} = 0.4$, $r_{13} = 0.4$, $r_{20} = 0.5$, $r_{23} = 0.5$; CPU time = 0.002 second.

Table V
Comparisons With the Approximations of Altiok and Perros, and Perros and Snyder for a Three-Node Network

Measures[a]	Exact	Altiok and Perros	Perros and Snyder	New
$P_1(0)$	0.1560	0.1722	0.1702	0.1516
$P_1(1)$	0.1297	0.1420	0.1440	0.1287
$P_1(2)$	0.1085	0.1175	0.1178	0.1091
$P_1(3)$	0.0914	0.0973	0.0967	0.0926
$P_1(4)$	0.0772	0.0806	0.0797	0.0786
$P_1(5)$	0.0654	0.0668	0.0659	0.0666
L_1	5.6100	4.8348	5.7497	5.5944
$P_2(0)$	0.6557	0.6744	0.6745	0.6473
$P_2(1)$	0.2349	0.2246	0.2259	0.2357
$P_2(2)$	0.1094	0.1010	0.0997	0.1170
L_2	0.4537	0.4266	0.4252	0.4697
$P_3(0)$	0.4901	0.4900	0.4900	0.4900
$P_3(1)$	0.2667	0.2600	0.2600	0.2662
$P_3(2)$	0.2431	0.2500	0.2500	0.2438
L_3	0.7529	0.7600	0.7599	0.7538
Avg. abs. deviation (P)		0.0083	0.0081	0.0023
Max. abs. deviation (P)		0.0187	0.0188	0.0084
Avg. rel. error (P)		0.0512	0.0495	0.0151
Max. rel. error (P)		0.1038	0.1103	0.0695
Avg. rel. error (L)		0.069	0.032	0.013

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (1.5, 0, 0)$, $\mu = (2, 2, 2)$, $N = (\infty, 2, 2)$; $r_{10} = 0.2$, $r_{12} = 0.4$, $r_{13} = 0.4$, $r_{20} = 0.3$, $r_{23} = 0.7$; CPU time = 0.002 second.

Table VI
Comparisons With the Approximations of Perros and Snyder for an Eight-Node Network

Measures[a]	Simulation	Perros and Snyder	New
$P_1(0)$	0.099	0.262	0.092
$P_1(1)$	0.071	0.191	0.083
$P_1(2)$	0.056	0.129	0.076
$P_1(3)$	0.048	0.090	0.069
$P_1(4)$	0.041	0.065	0.062
$P_1(5)$	0.037	0.048	0.057
L_1	17.17	14.39	9.92
$P_2(0)$	0.614	0.656	0.597
$P_2(1)$	0.235	0.233	0.252
$P_2(2)$	0.151	0.110	0.151
L_2	0.537	0.454	0.553
$P_3(0)$	0.607	0.656	0.599
$P_3(1)$	0.244	0.233	0.251
$P_3(2)$	0.149	0.111	0.150
L_3	0.542	0.455	0.551
$P_4(0)$	0.566	0.618	0.560
$P_4(1)$	0.240	0.239	0.255
$P_4(2)$	0.195	0.143	0.186
L_4	0.629	0.526	0.626
$P_5(0)$	0.461	0.600	0.420
$P_5(1)$	0.262	0.250	0.277
$P_5(2)$	0.277	0.150	0.303
L_5	0.817	0.550	0.883
$P_6(0)$	0.493	0.595	0.484
$P_6(1)$	0.282	0.249	0.266
$P_6(2)$	0.224	0.156	0.249
L_6	0.731	0.561	0.765
$P_7(0)$	0.405	0.471	0.414
$P_7(1)$	0.273	0.273	0.263
$P_7(2)$	0.322	0.256	0.323
L_7	0.917	0.785	0.909
$P_8(0)$	0.202	0.200	0.200
$P_8(1)$	0.189	0.177	0.199
$P_8(2)$	0.609	0.623	0.601
L_8	1.407	1.423	1.401
Avg. abs. deviation (P)		0.050	0.013
Max. abs. deviation (P)		0.163	0.041
Avg. rel. error (P)		0.348	0.108
Max. rel. error (P)		1.690	0.541
Avg. rel. error (L)		0.170	0.077

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (5, 0, 0, 0, 0, 0, 0, 0)$, $\mu = (4, 1, 1, 2, 2, 2, 2, 3.5)$; $N = (\infty, 2, 2, 2, 2, 2, 2, 2)$; $r_{10} = 0.0$, $r_{12} = 0.2$, $r_{13} = 0.2$, $r_{14} = 0.2$, $r_{17} = 0.2$, $r_{18} = 0.2$; $r_{20} = 0.0$, $r_{24} = 0.5$, $r_{26} = 0.5$, $r_{30} = 0.0$, $r_{34} = 0.5$, $r_{37} = 0.5$; $r_{40} = 0.0$, $r_{45} = 1.0$, $r_{50} = 0.0$, $r_{56} = 0.3$, $r_{57} = 0.3$, $r_{58} = 0.4$; $r_{60} = 0.0$, $r_{68} = 1.0$, $r_{70} = 0.0$, $r_{78} = 1.0$, $r_{80} = 1.0$; CPU time = 0.005 second.

Table VII
Comparisons With the Approximations of Perros and Snyder for an Eight-Node Network

Measures[a]	Simulation	Perros and Snyder	New
$P_1(0)$	0.321	0.323	0.299
$P_1(1)$	0.313	0.335	0.332
$P_1(2)$	0.366	0.342	0.369
L_1	1.044	1.019	1.070
$P_2(0)$	0.601	0.593	0.590
$P_2(1)$	0.254	0.254	0.254
$P_2(2)$	0.146	0.153	0.156
L_2	0.545	0.560	0.566
$P_3(0)$	0.584	0.587	0.570
$P_3(1)$	0.261	0.256	0.259
$P_3(2)$	0.155	0.157	0.171
L_3	0.571	0.571	0.602
$P_4(0)$	0.564	0.551	0.560
$P_4(1)$	0.258	0.253	0.255
$P_4(2)$	0.178	0.196	0.185
L_4	0.614	0.645	0.625
$P_5(0)$	0.557	0.579	0.548
$P_5(1)$	0.266	0.258	0.264
$P_5(2)$	0.177	0.163	0.188
L_5	0.620	0.585	0.640
$P_6(0)$	0.750	0.768	0.749
$P_6(1)$	0.195	0.180	0.190
$P_6(2)$	0.056	0.053	0.062
L_6	0.307	0.286	0.313
$P_7(0)$	0.539	0.529	0.533
$P_7(1)$	0.270	0.254	0.259
$P_7(2)$	0.191	0.216	0.208
L_7	0.652	0.687	0.675
$P_8(0)$	0.457	0.436	0.459
$P_8(1)$	0.262	0.250	0.262
$P_8(2)$	0.281	0.314	0.279
L_8	0.824	0.878	0.820
Avg. abs. deviation (P)		0.013	0.008
Max. abs. deviation (P)		0.033	0.022
Avg. rel. error (P)		0.046	0.033
Max. rel. error (P)		0.131	0.107
Avg. rel. error (L)		0.043	0.029

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (3, 0, 0, 0, 0, 0, 0, 0)$, $\mu = (4, 1, 1, 2, 2, 2, 2, 3.5)$; $N = (2, 2, 2, 2, 2, 2, 2, 2)$; routing is the same as in Table VI; CPU time = 0.006 second.

Table VIII
Comparisons With the Approximations of Perros and Snyder for an Eight-Node Network

Measures[a]	Simulation	Perros and Snyder	New
$P_1(0)$	0.486	0.487	0.482
$P_1(1)$	0.247	0.250	0.250
$P_1(2)$	0.127	0.127	0.129
$P_1(3)$	0.065	0.065	0.067
$P_1(4)$	0.033	0.034	0.035
$P_1(5)$	0.019	0.017	0.018
L_1	1.088	1.055	1.076
$P_2(0)$	0.795	0.799	0.799
$P_2(1)$	0.162	0.161	0.161
$P_2(2)$	0.034	0.032	0.033
$P_2(3)$	0.009	0.008	0.008
L_2	0.256	0.249	0.250
$P_3(0)$	0.790	0.797	0.791
$P_3(1)$	0.166	0.162	0.165
$P_3(2)$	0.033	0.033	0.035
$P_3(3)$	0.011	0.008	0.009
L_3	0.265	0.252	0.261
$P_4(0)$	0.798	0.798	0.798
$P_4(1)$	0.161	0.161	0.161
$P_4(2)$	0.032	0.032	0.032
$P_4(3)$	0.008	0.008	0.008
L_4	0.251	0.251	0.250
$P_5(0)$	0.789	0.796	0.789
$P_5(1)$	0.166	0.163	0.167
$P_5(2)$	0.036	0.033	0.035
$P_5(3)$	0.010	0.009	0.009
L_5	0.268	0.255	0.265
$P_6(0)$	0.775	0.780	0.779
$P_6(1)$	0.173	0.172	0.172
$P_6(2)$	0.040	0.038	0.038
$P_6(3)$	0.012	0.011	0.011
L_6	0.289	0.279	0.280
$P_7(0)$	0.585	0.579	0.578
$P_7(1)$	0.245	0.245	0.247
$P_7(2)$	0.102	0.103	0.105
$P_7(3)$	0.069	0.073	0.070
L_7	0.654	0.670	0.667
$P_8(0)$	0.753	0.750	0.750
$P_8(1)$	0.185	0.188	0.188
$P_8(2)$	0.046	0.047	0.047
$P_8(3)$	0.015	0.016	0.015
L_8	0.323	0.328	0.328
Avg. abs. deviation (P)		0.002	0.002
Max. abs. deviation (P)		0.007	0.007
Avg. rel. error (P)		0.035	0.028
Max. rel. error (P)		0.273	0.181
Avg. rel. error (L)		0.029	0.016

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (1, 0, 0, 0, 0, 0, 0, 0)$, $\mu = (2, 1, 1, 2, 2, 1, 1, 4)$; $N = (\infty, 3, 3, 3, 3, 3, 3, 3)$; routing is the same as in Table VI; CPU time = 0.008 second.

Table IX
Comparisons With the Approximations of Perros and Snyder for an Eight-Node Network

Measures[a]	Simulation	Perros and Snyder	New
$P_1(0)$	0.339	0.403	0.348
$P_1(1)$	0.204	0.239	0.227
$P_1(2)$	0.129	0.135	0.148
$P_1(3)$	0.087	0.079	0.096
$P_1(4)$	0.061	0.047	0.063
$P_1(5)$	0.045	0.030	0.041
L_1	2.458	2.194	1.874
$P_2(0)$	0.763	0.788	0.764
$P_2(1)$	0.180	0.168	0.182
$P_2(2)$	0.057	0.044	0.054
L_2	0.294	0.256	0.290
$P_3(0)$	0.750	0.782	0.742
$P_3(1)$	0.189	0.172	0.194
$P_3(2)$	0.061	0.046	0.064
L_3	0.312	0.264	0.322
$P_4(0)$	0.528	0.547	0.533
$P_4(1)$	0.253	0.254	0.259
$P_4(2)$	0.219	0.200	0.208
L_4	0.692	0.653	0.675
$P_5(0)$	0.535	0.579	0.532
$P_5(1)$	0.263	0.258	0.267
$P_5(2)$	0.202	0.164	0.201
L_5	0.668	0.585	0.669
$P_6(0)$	0.741	0.770	0.747
$P_6(1)$	0.194	0.178	0.191
$P_6(2)$	0.065	0.052	0.063
L_6	0.323	0.282	0.316
$P_7(0)$	0.525	0.543	0.526
$P_7(1)$	0.261	0.253	0.260
$P_7(2)$	0.215	0.204	0.214
L_7	0.690	0.661	0.689
$P_8(0)$	0.497	0.500	0.500
$P_8(1)$	0.257	0.252	0.261
$P_8(2)$	0.245	0.248	0.239
L_8	0.748	0.748	0.739
Avg. abs. deviation (P)		0.018	0.005
Max. abs. deviation (P)		0.064	0.023
Avg. rel. error (P)		0.099	0.033
Max. rel. error (P)		0.333	0.147
Avg. rel. error (L)		0.092	0.043

[a] P represents the marginal occupancy probability and L represents the expected queue length; $\lambda = (1, 0, 0, 0, 0, 0, 0, 0)$, $\mu = (4, 2, 2, 2, 2, 2, 2, 4)$; $N = (\infty, 2, 2, 2, 2, 2, 2, 2)$; routing is the same as in Table VI; CPU time = 0.011 second.

Table X
Comparisons With the Simulation for the Cases With More Than One External Arrivals: Three-Node Network

Case	Measures	Simulation	New	Rel. Error
$N = (5, 4, 3)$	$P_1(5)$	0.558	0.558	0.000
$\lambda = (2, 0.2, 0.1)$	L_1	4.246	4.241	0.001
$\mu = (1, 1, 1)$	$P_2(4)$	0.074	0.078	0.054
	L_2	1.150	1.156	0.005
	$P_3(3)$	0.279	0.282	0.011
	L_3	1.441	1.452	0.008
$N = (2, 2, 3)$	$P_1(2)$	0.269	0.272	0.011
$\lambda = (1.2, 0.3, 0.2)$	L_1	0.855	0.873	0.021
$\mu = (2, 1.5, 1)$	$P_2(2)$	0.193	0.206	0.067
	L_2	0.679	0.698	0.028
	$P_3(3)$	0.374	0.381	0.019
	L_3	1.733	1.741	0.005
$N = (\infty, 3, 3)$	L_1	2.462	2.386	0.031
$\lambda = (1.2, 0.8, 0.5)$	$P_2(3)$	0.176	0.177	0.006
$\mu = (2, 2, 2)$	L_2	1.189	1.183	0.005
	$P_3(3)$	0.312	0.306	0.019
	L_3	1.557	1.544	0.008
$N = (\infty, 2, 3)$	L_1	4.284	4.228	0.013
$\lambda = (1.5, 1.0, 0.5)$	$P_2(2)$	0.386	0.395	0.023
$\mu = (2.5, 2.0, 2.0)$	L_2	1.080	1.089	0.008
	$P_3(3)$	0.386	0.378	0.021
	L_3	1.758	1.741	0.010

$r_{12} = r_{13} = 0.35$, $r_{10} = 0.3$; $r_{23} = 0.65$, $r_{20} = 0.35$; CPU time less than 0.002 second.

Tables III, V, VI and IX), the new algorithm appears to work quite well for these cases as well.

Finally, Tables X and XI show the performance of our algorithm based on $P_i(N_i)$ and L_i (mean queue length at queue i) when there are external arrivals to more than one node. These results are only compared to simulation analysis because there are no other reported results in the literature, and an exact analysis of even the simplest (3-node) configuration involves precisely the onerous amount of computation that the approximation method has been developed to avoid. Note that values of $P_i(N_i)$ for $i = 1, \ldots, k$ completely determine the throughput for queue k. As we see in Tables X and XI, external arrivals to more than one node do not seem to degrade the performance of our algorithm.

It is important to state that we have not proven the convergence of our algorithm. However, in all of the problems we have tested to date, we have not found any which did not converge. The maximum number of iterations needed for convergence was 11; in most cases convergence to one part in 10^5 occurred within 4–7 iterations. From a practical point of view, note

Table XI
Comparisons With the Simulation for the Cases With More Than One External Arrivals: Four-Node Network

Case	Measures	Simu-lation	New	Rel. Error
$N = (2, 2, 2, 3)$	$P_1(2)$	0.374	0.385	0.029
$\lambda = (1, 0.2, 0.1, 0.05)$	L_1	1.080	1.100	0.019
$\mu = (1, 1, 1, 1)$	$P_2(2)$	0.155	0.163	0.052
	L_2	0.582	0.600	0.031
	$P_3(2)$	0.112	0.120	0.071
	L_3	0.475	0.485	0.021
	$P_4(3)$	0.269	0.276	0.026
	L_4	1.408	1.416	0.006
$N = (\infty, 2, 2, 2)$	L_1	5.184	4.888	0.057
$\lambda = (1.2, 0.2, 0.1, 0.05)$	$P_2(2)$	0.220	0.231	0.050
$\mu = (2, 1.5, 1.5, 1.5)$	L_2	0.731	0.747	0.022
	$P_3(2)$	0.183	0.197	0.077
	L_3	0.642	0.664	0.034
	$P_4(2)$	0.495	0.497	0.004
	L_4	1.222	1.229	0.006
$N = (\infty, 2, 2, 2)$	L_1	2.612	2.610	0.001
$\lambda = (1.2, 0.3, 0.2, 0.1)$	$P_2(2)$	0.145	0.155	0.069
$\mu = (2, 2, 2, 2)$	L_2	0.559	0.578	0.034
	$P_3(2)$	0.126	0.134	0.063
	L_3	0.506	0.522	0.032
	$P_4(2)$	0.381	0.392	0.029
	L_4	1.019	1.043	0.024

$r_{12} = r_{13} = 0.3$, $r_{14} = r_{10} = 0.2$; $r_{23} = 0.05$, $r_{24} = 0.8$, $r_{20} = 0.15$; $r_{34} = 0.85$, $r_{30} = 0.15$; CPU time less than 0.002 second.

that the maximum CPU time required for an eight-node network problem was 0.008 second. We also do not have a priori bounds on the accuracy of the method; these are currently being explored.

4. CONCLUSIONS

We have presented a new approximate algorithm for analyzing a general configuration of an open queueing network with blocking. Besides being accurate and fast, our algorithm has the following advantages over those previously reported.

Generality. It can solve not only networks with a large number of servers, but also general topologies including external arrivals at more than one queue.

Robustness. It yields accurate results regardless of whether the queues with external arrivals have infinite buffers or not, or whether the service rates are high or low.

Simplicity. There are no numerical procedures involving the solution of simultaneous nonlinear equations or fixed point problems.

Considering the generality, robustness, simplicity and accuracy of the algorithm, and its significant improvement over previous methods, it holds promise to be a useful tool in the study of networks of queues.

ACKNOWLEDGMENT

This study was supported in part by a contract from the General Motors Corporation to the University of Michigan for the development of analytical tools for production systems. We wish, in particular, to thank Jeff Alden and Larry Burns, of GM, for their continuing interest and support, and David Lederschnaider for his help in implementation of the method on a microcomputer. A referee also did us the considerable courtesy of suggesting numerous specific style and technical changes, all of which were taken seriously.

REFERENCES

ALTIOK, T. 1982. Approximate Analysis of Exponential Tandem Queues With Blocking. *Eur. J. Opnl. Res.* **11,** 390–398.

ALTIOK, T., AND H. G. PERROS. 1986. Open Networks of Queues With Blocking: Split and Merge Configurations. *AIIE Trans.* **18,** 251–261.

ALTIOK, T., AND H. G. PERROS. 1987. Approximate Analysis of Arbitrary Configurations of Open Queueing Networks With Blocking. *Ann. Opns. Res.* **9,** 481–509.

BOCHAROV, P. P., AND G. P. ROKHAS. 1980. On an Exponential Queueing System in Series With Blocking. *Problems of Control and Information Theory* **9,** 441–455.

BOXMA, O. J., AND A. G. KONHEIM. 1981. Approximate Analysis of Exponential Queueing Systems With Blocking. *Acta Informat.* **15,** 19–66.

BRANDWAJN, A., AND Y. L. JOW. 1985. An Approximate Method for Tandem Queues With Blocking. Manuscript, Amdahl Corp.

CASEAU, P., AND G. PUJOLLE. 1979. Throughput Capacity of a Sequence of Queues With Blocking Due to Finite Waiting Room. *IEEE Trans. Soft. Eng.* **SE-5,** 631–642.

CHOONG, Y. F., AND S. B. GERSHWIN. 1987. A Decomposition Method for the Approximate Evaluation of Capacitated Transfer Lines With Unreliable Machines and Random Processing Times. *IIE Trans.* **9,** 150–159.

FOSTER, F. G., AND H. G. PERROS. 1980. On the Blocking Process in Queue Networks. *Eur. J. Opnl. Res.* **5,** 276–283.

GERSHWIN, S. B. 1987. An Efficient Decomposition Method for the Approximate Evaluation of Tandem Queues With Finite Storage and Blocking. *Opns. Res.* **35,** 291–305.

HILLIER, F. S., AND R. BOLING. 1967. Finite Queues in Series With Exponential or Erlang Service Times—Numerical Approach. *Opns. Res.* **15,** 286–303.

JACKSON, J. R. 1963. Jobshop-Like Queueing Systems. *Mgmt. Sci.* **10,** 131–142.

KERBACHE, L., AND J. M. SMITH. 1986. The Generalized Expansion Method for Open Finite Queueing Networks. Technical Report, Dept. of Industrial Engineering and Operations Research, Univ. of Massachusetts, Amherst, Mass.

LABETOULLE, J., AND G. PUJOLLE. 1980. Isolation Method in a Network of Queues. *IEEE Trans. Soft. Eng.* **SE-6,** 373–381.

LATOUCHE, G., AND M. F. NEUTS. 1980. Efficient Algorithmic Solutions to Exponential Tandem Queues With Blocking. *SIAM J. Algebraic Discrete Methods* **1,** 93–106.

LEE, H. S., AND S. M. POLLOCK. 1989. Approximate Analysis for the Merge Configuration of an Open Queueing Network With Blocking. *IIE Trans.* **21,** 122–124.

PERROS, H. G., AND T. ALTIOK. 1986. Approximate Analysis of Open Networks of Queues With Blocking: Tandem Configurations. *IEEE Trans. Soft. Eng.* **SE-12.**

PERROS, H. G., AND P. M. SNYDER. 1986. A Computationally Efficient Approximation Algorithm for Analyzing Open Queueing Networks With Blocking. Manuscript, Computer Science Department, North Carolina State University.

POLLOCK, S. M., J. R. BIRGE AND J. M. ALDEN. 1985. Approximation Analysis for Open Tandem Queues With Blocking: Exponential and General Service Distributions. Technical Report, IOE Dept. 85-30, University of Michigan, Ann Arbor.

TAKAHASHI, Y., H. MIYAHARA AND T. HASEGAWA. 1980. An Approximation Method for Open Restricted Queueing Networks. *Opns. Res.* **28,** 594–602.