SRC TR 87-209-r1

An Approximation Method for General Tandem Queueing Systems Subject to Blocking

By

L. Gun and A. M. Makowski

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE	2. REPORT TYPE	3. DATES COVERED
1987		**00-00-1987 to 00-00-1987**

4. TITLE AND SUBTITLE	5a. CONTRACT NUMBER
An Approximation Method for General tandem Queueing Systems Subject to Blocking	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **University of Maryland,Electrical Engineering Department,College Park,MD,20742**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR'S ACRONYM(S)
	11. SPONSOR/MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

14. ABSTRACT
see report

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**		**26**	

To Appear in the *Proceedings of the First International Workshop on Queueing Networks with Blocking*, North Holland, 1989.

AN APPROXIMATION METHOD FOR GENERAL TANDEM QUEUEING SYSTEMS SUBJECT TO BLOCKING

by

Levent Gün* and Armand M. Makowski**

Electrical Engineering Department and Systems Research Center ***
University of Maryland, College Park, Maryland 20742.

ABSTRACT

An iterative approximation algorithm is presented for calculating the stationary queue size probabilities of *tandem* queueing systems subject to *blocking*. The algorithm combines a decomposition/aggregation technique with exact analytical results for two node systems. It applies to tandem lines where *failure* type servers with *phase-type* service and repair time distributions are in attendance under various blocking disciplines

* The work of this author was supported partially by the Office of Naval Research through Grant N00014-84-K-0614 and partially through a grant from AT&T Bell Laboratories.

** The work of this author was supported partially through NSF Grant ECS-83-51836 and partially through Grants from AT&T Bell Laboratories and GM Laboratories.

*** With the support of the National Science Foundation's Engineering Research Centers Program NSFD CDR 88-03012

1. INTRODUCTION

Consider a production system composed of many processing stages (servers) through which the parts (jobs) must pass in some prespecified order. In practice, the servers' behavior is modulated by several factors, such as fluctuations in the service times and interruptions of service due to server breakdowns. Although this service variability greatly affects the performance of such a system, its influence can be somewhat mitigated by using intermediate storage spaces (buffers) between the servers. However, because of physical limitations on the buffer sizes, the flow of jobs through the system may get *blocked.*

The blocking systems considered here are composed of a series configuration of service stations with *finite* intermediate buffers between the stations. A typical tandem configuration is shown in Figure 1. At each station some work is performed on a job which is then passed on to the next station; upon completion of service at the last station, the job immediately leaves the system.

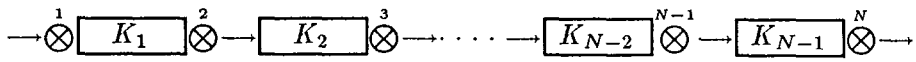

Figure 1.

The literature considers two distinct blocking policies for transfer lines with finite buffers. In order to describe them, let S be one of the servers in Figure 1. Under the first policy, called *immediate blocking*, at a time of service completion, the server S is blocked if the downstream buffer becomes full as a result of this service completion; the server S remains blocked until a space becomes available in the downstream buffer, at which time the server resumes service and *begins* processing its next job (if any). Under the second policy, called *non-immediate* blocking, the server S is blocked at a service completion time if the job that has just completed service finds the next buffer to be full. As soon as a space becomes available in the next buffer this job moves downstream *without* receiving any further service at server S, which then begins processing its next job (if any). As noted by Altiok and Stidham [2], these two blocking policies are in general *not* equivalent in that the solution of the system under one policy cannot be obtained from the solution of the system (with possibly different parameter values) under the other policy; such an equivalence does however hold for two station systems.

Queueing networks with blocking have been studied by researchers from different research communities, as evidenced by the annotated bibliography on blocking systems compiled in [7]. A basic assumption made in all the models studied in the surveyed literature is that *the last stage is never blocked*, i. e., there is always space available for a job whose service has been completed at the last server.

Queueing networks with blocking are typically very difficult to analyze, and closed form results at steady state (or otherwise) are usually not available. Although a rigorous Markovian analysis was performed early on by Hunt [9], to date strikingly few results have been obtained, with the bulk of the work focusing on *continuous-time* models. This state of affairs points to the need of developing

efficient approximation techniques to evaluate performance measures of interest. In this paper, an iterative approximation scheme is presented for finding the steady-state *marginal* probabilities of the queue sizes in tandem queueing systems with *finite* capacity buffers and *phase-type* (PH-type) servers. The algorithm is based on the analytical results obtained in [8] for two node tandem systems, and is presented under the *immediate* blocking policy. The same approximation scheme also applies to tandem systems under the *non-immediate* blocking policy in view of the above-mentioned equivalence for two node systems, and is only briefly outlined in the interest of brevity. In light of the results derived in [8], the proposed approximation scheme is also applicable to systems with failure type servers with PH-type service and repair distributions, and to systems where jobs serviced at the $(i+1)^{st}$ server can be fed back to the i^{th} buffer. Since the approximation scheme is based on the effective solution of two node systems, the relevant results of [8] are summarized in the Appendix. The reader is also referred to the work of Buzacott and Kostelski [4] for an algorithmic solution of two node production systems with two-stage Coxian service distributions.

Most approximation algorithms use the *flow conservation principle* to decompose the tandem model into simple two node models. The approximation reported here uses this viewpoint in the form taken by Altıok in [1], and represents the *effective* service distribution of a server by considering *all* the servers upstream of this server in order to capture the effect of *blocking*. However, the algorithm presented here differs from Altıok's in that instead of approximating the arrivals to the i^{th} buffer by a Poisson process, another similar effective representation is introduced to capture the effect of *idling*. Once this decomposition step is taken, approximations are made to express the effective representations iteratively in terms of quantities that can be obtained by solving two node models.

Although the discussion is given for the discrete-time formulation, the ideas presented here apply *mutatis mutandis* to the continuous-time formulation. The modeling of time as a discrete parameter could be motivated by the fact that service times in manufacturing systems are often constant, the main source of randomness being introduced by the possibility of server failures. The accuracy of the algorithm is validated through numerical examples, both for continuous and discrete-time systems. Comparison of the results against simulations indicates reasonable accuracy under both blocking policies even in the presence of significant blocking. The algorithm is also compared to another approximation algorithm proposed by Jun and Perros [10] for open tandem queueing systems with two-stage Coxian service distributions.

Phase Type Distributions

To fix the notation and terminology, it is convenient at this point to briefly describe the class of *discrete-time* PH-type distributions. The reader is invited to consult the monograph by Neuts [11] for additional information on this topic and for a similar development on continuous-time PH-distributions. The following notation is used hereafter: For any positive integer r, the $r \times r$ identity matrix is denoted by I_r and the $r \times 1$ column vector of ones is denoted by e_r, while the $1 \times r$ dimensional row vector with all zero entries is denoted by 0_r.

Loosely speaking, a *discrete-time* PH-type distribution is any probability distribution on the

non-negative integers which can be realized as the distribution of the time until *absorption* in a discrete-time finite state space Markov chain with a *single* absorbing state. The class of PH-type distributions includes well-known distributions such as the Generalized Negative Binomial and Hypergeometric distributions, as well as any distribution with *finite* support on the non-negative integers.

Specifically, consider a discrete-time Markov chain on the state space $\{1, 2, \ldots, m+1\}$, where the states $\{1, \ldots, m\}$ are *transient* and the state $m+1$ is *absorbing.* The chain starts in state j with probability α_j and evolves according to the matrix P of one-step transition probabilities with

$$P = \begin{pmatrix} Q & p \\ 0_m & 1 \end{pmatrix} , \quad (\alpha, \alpha_{m+1}) \quad \text{and} \quad \alpha = (\alpha_1, \alpha_2, \ldots, \alpha_m) , \tag{1.1}$$

where

Q is a $m \times m$ substochastic matrix,

p is a $m \times 1$ column vector of absorption probabilities into the absorbing state $m+1$ from the transient states $\{1, \ldots, m\}$,

α is a $1 \times m$ row vector of initialization probabilities.

The *PH-type* probability distribution associated with the pair (α, Q) is the distribution function F on the non-negative integers with probability mass function $\{q_k,\ k = 0, 1, \ldots\}$ given by

$$q_k = \begin{cases} \alpha_{m+1} , & k = 0 , \\ \alpha Q^{k-1} p , & k \geq 1 . \end{cases} \tag{1.2}$$

The pair (α, Q) is called the *representation* of the distribution F, which it uniquely determines. The converse is not true in that a given PH-type distribution function F admits infinitely many PH-representations.

The following probabilistic construction is given in Neuts [11, p. 48]. Upon absorption into state $m+1$, *independent* multinomial trials with probabilities (α, α_{m+1}) are *instantaneously* and *repeatedly* performed until one of the alternatives $1, 2, \ldots, m$ occurs, say j. The process is now restarted in the state j and the same procedure is repeated at the next absorption. Upon indefinitely continuing this procedure, a new Markov process is constructed on $\{1, 2, \ldots, m\}$ with the state $m+1$ as an *instantaneous* state, and with matrix Q^* of one-step transition probabilities given by

$$Q^* = Q + \frac{1}{1 - \alpha_{m+1}} \, p \, \alpha . \tag{1.3}$$

The representation (α, Q) is said *irreducible* if the Markov chain on $\{1, \ldots, m\}$ with one-step transition matrix Q^* is *irreducible.* Every PH-type distribution function F admits an irreducible representation [11, p. 49].

2. THE DISCRETE-TIME MODEL

Consider the tandem system shown in Figure 1, which consists of N nodes with $N-1$ intermediate *finite* capacity buffers of sizes K_i, $1 \leq i < N$, including the jobs being served by the servers. Each node is attended by a *single* server which operates according to the FCFS (first-come-first-served) queueing discipline. For $1 \leq i \leq N$, the service times at the i^{th} node are assumed to be *independent* and *identically distributed (i.i.d)* with common discrete PH-distribution F_i given by the *irreducible* PH-representation (α_i, Q_i) of order m_i. The $m_i \times 1$ column vectors of absorption (service completion at the i^{th} node) probabilities is denoted by p_i, and in order to avoid cases of limited interest, the vector of initialization probabilities α_i is assumed to satisfy $\alpha_i e_{m_i} = 1$. The matrix $(I_{m_i} - Q_i)$ is assumed *nonsingular*, so that the service completion from any initial phase is certain [11, p. 45]. The service times at different servers are also assumed mutually independent. It is assumed that the last server is *never* blocked while the first server is *always busy*, i. e., there is an infinite supply of exogenous jobs. The situation where the first server is generating arrivals to the first buffer according to a PH-renewal process can also be approximated by a similar analysis.

Let the set S^i of service phases for the i^{th} server be defined by

$$S^i := \left\{ s_l^i : \ 1 \leq l \leq m_i \right\} \ , \qquad 1 \leq i \leq N,$$

and define the scalars

$$\Sigma_i^j := \sum_{k=i}^{j} m_k \ , \ \ 1 \leq i \leq j \leq N.$$

3. DECOMPOSITION AND APPROXIMATION METHOD

The steady-state marginal queue length distribution for the i^{th} node could be calculated from Theorem A.1 of the Appendix if the $(i+1)^{st}$ server were *not* subject to blocking and the i^{th} server were *always* busy. However, at a service completion epoch for the $(i+1)^{st}$ server, this server may be blocked and may remain blocked until there is a departure from the $(i+1)^{st}$ buffer. Similarly, at a service completion at the i^{th} server, this server may become idle. The decomposition method proposed here amounts to finding *equivalent* PH-representations for the i^{th} and the $(i+1)^{st}$ servers that incorporate the effects of idling and blocking, respectively.

For $1 \leq i, j \leq N$, $1 \leq l \leq m_j$ and $t \geq 0$, define the events

$B^i(t)$: the i^{th} server is *blocked* at time t,

$B_l^{i,j}(t)$: the servers k, $i \leq k < j$, are *all blocked* at time t, and the j^{th} server is *not blocked* and in service phase s_l^j,

$I^i(t)$: the i^{th} server is *idle* at time t,

$I_l^{i,j}(t)$: the servers k, $j < k \leq i$, are *all idle* at time t, and the j^{th} server is *not idle* and in service phase s_l^j.

Assuming that the limits exist, define the probabilities

$$\begin{aligned}
P_B^i &:= \lim_n P[B^i(T_n^i)] , && 1 \le i \le N, \\
P_{B_l}^{i,j} &:= \lim_n P[B_l^{i,j}(T_n^i)] , && 1 \le i < j \le N, \ \ 1 \le l \le m_j, \\
P_I^i &:= \lim_n P[I^i(T_n^i)] , && 1 \le i \le N, \\
P_{I_l}^{i,j} &:= \lim_n P[I_l^{i,j}(T_n^i)] , && 1 \le j < i \le N, \ \ 1 \le l \le m_j,
\end{aligned} \tag{3.1}$$

where $\{T_n^i,\ n = 1, 2, \ldots\}$ is the sequence of service completion times at node i, and $\lim_n$ is taken with n going to infinity.

Decomposition

Fix $1 \le i < N$ and consider the n^{th} service completion time T_n^{i+1} in the $(i+1)^{st}$ server. This server is either blocked with probability $P[B^{i+1}(T_n^{i+1})]$ or with complementary probability $1 - P[B^{i+1}(T_n^{i+1})]$ it starts serving its next job (if any) according to the initialization vector α_{i+1}. In the case of blocking, the event $B_l^{i+1,j}(T_n^{i+1})$ takes place for some j, $i+1 < j \le N$, and l, $1 \le l \le m_j$, in which case the effective service time of the blocked job at the $(i+1)^{st}$ server is equal to the residual service time in the j^{th} server plus the sum of the service times of all the blocked servers $j-1, \ldots, i+2$ (time to unblock), plus the service time in the $(i+1)^{st}$ server. Therefore, *if* the $(i+1)^{st}$ server is blocked at time T_n^{i+1}, then the service phase of the $(i+1)^{st}$ server is effectively reinitialized at phase s_l^j with probability $P[B_l^{i+1,j}(T_n^{i+1})]$ and the transitions occur within the set S^j according to the matrix Q_j. Upon service completion in the j^{th} server, service in the $(j-1)^{st}$ server is initialized according to α_{j-1}, thus causing a transition from the set S^j to the set S^{j-1} in the effective representation of the $(i+1)^{st}$ server. This argument can be repeated upon moving upstream from station j to station $(i+1)$ and finally, upon service completion in the $(i+2)^{nd}$ server, the blocking period of the $(i+1)^{st}$ server ends and this server resumes service according to the initialization vector α_{i+1}.

Therefore in steady state, the effect of blocking on the $(i+1)^{st}$ server can be captured by considering a new PH-distribution for this server with set of service phases given by $\bigcup_{j=i+1}^{N} S^j$. This *effective* service time distribution of order Σ_{i+1}^N is obtained by considering all the servers downstream from the $(i+1)^{st}$ server, and is denoted by the pair $(\alpha_2(i+1), Q_2(i+1))$. If the corresponding $\Sigma_{i+1}^N \times 1$ column vector of absorption probabilities is denoted by $p_2(i+1)$, then

$$Q_2(i+1) = \begin{pmatrix} Q_{i+1} & & & \\ p_{i+2}\,\alpha_{i+1} & Q_{i+2} & & \\ & \ddots & \ddots & \\ & & p_N\,\alpha_{N-1} & Q_N \end{pmatrix}, \quad p_2(i+1) = \begin{pmatrix} p_{i+1} \\ 0_{m_{i+2}}^T \\ \vdots \\ 0_{m_N}^T \end{pmatrix}, \tag{3.2a}$$

and the $1 \times \Sigma_{i+1}^N$ initialization probability vector $\alpha_2(i+1)$ of this effective representation of the $(i+1)^{st}$ server is given by

$$\alpha_2(i+1) = \left((1 - P_B^{i+1})\alpha_{i+1}, P_B^{i+1,i+2}, \ldots, P_B^{i+1,N} \right) \tag{3.2b}$$

for $1 \leq i < N$. For $1 \leq i < j \leq N$, the $1 \times m_j$ row vector $P_B^{i+1,j}$ has entries $P_{B_l}^{i+1,j}$, $1 \leq l \leq m_j$.

With this effective PH-representation, the $(i+1)^{st}$ server is *never* blocked, but a service completion is allowed only from the phases that correspond to a service in the $(i+1)^{st}$ server. The subscript 2 indicates that the representation $(\alpha_2(i+1), Q_2(i+1))$ is the effective representation for the *second* node server when considering the i^{th} buffer.

A similar argument can now be developed to capture the effect of idling, in which case for $1 \leq i \leq N$, the *effective* service of the i^{th} server can be represented by a PH-distribution of dimension Σ_1^i with set of service phases $\bigcup_{j=1}^{i} S^j$. The effective representation of the i^{th} server as the *first* node server in the two node equivalent representation when considering the i^{th} buffer is denoted by $(\alpha_1(i), Q_1(i))$. It is given by

$$Q_1(i) = \begin{pmatrix} Q_i & & & & \\ p_{i-1}\,\alpha_i & Q_{i-1} & & & \\ & \cdot & \cdot & & \\ & & \cdot & \cdot & \\ & & & p_1\,\alpha_1 & Q_1 \end{pmatrix}, \quad p_1(i) = \begin{pmatrix} p_i \\ 0_{i-1}^T \\ \cdot \\ \cdot \\ 0_1^T \end{pmatrix}, \tag{3.3a}$$

where $p_1(i)$ is the $\Sigma_1^i \times 1$ column vector of absorption probabilities, and

$$\alpha_1(i) = \left((1 - P_I^i)\alpha_i, P_I^{i,i-1}, \ldots, P_I^{i,1} \right) \tag{3.3b}$$

for $1 < i \leq N$. The $1 \times m_j$ row vector $P_I^{i,j}$ has entries $P_{I_l}^{i,j}$, $1 \leq l \leq m_j$ and $1 \leq j < i \leq N$.

Therefore, by considering the effective representations (3.2) and (3.3) for each buffer i, $1 \leq i < N$, the tandem system in Figure 1 can be *decomposed* into $N-1$ two node systems of the form shown in Figure 2.

$$(\alpha_1(i), Q_1(i)) \longrightarrow \otimes \boxed{K_i} \otimes \longrightarrow (\alpha_2(i+1), Q_2(i+1))$$

Figure 2.

Since the matrices $I_{m_i} - Q_i$, $1 \leq i \leq N$, are all invertible, so are the matrices $I_{\Sigma_1^i} - Q_1(i)$ and $I_{\Sigma_{i+1}^N} - Q_2(i+1)$, $1 \leq i < N$. Furthermore, since the first (resp. last) node server of Figure 1 is never starved (resp. blocked), the first (resp. second) node server of Figure 2 is also never starved (resp. blocked), and the solution technique given in [8] thus applies.

Approximations

Although the tandem line of Figure 1 has been decomposed into $N-1$ two node systems of the type displayed in Figure 2, these systems are *coupled* through the initialization vectors $\alpha_k(i)$, $k = 1, 2$ and $1 \leq i < N$, and approximations are therefore needed in order to compute the vectors $\alpha_k(i)$.

As in [8], a Markov chain is associated with each one of the two node models shown in Figure 2. For $1 \leq i < N$, let the vector $u(i)$ (resp. $v(i)$) be the steady-state probability vector of this

Markov chain embedded at arrival (resp. departure) epochs at the first (resp. second) node. The $1 \times \Sigma_1^i$ block component $v_0(i)$ of $v(i)$ (see Appendix) is the steady-state probability vector that the second server becomes idle after a service completion. If $k = \Sigma_j^i + l$, $1 < j \leq i$, $1 \leq l \leq m_{j-1}$, then the k^{th} component of $v_0(i)$ is the joint probability that at a service completion epoch the second server is idle and the phase of the first node server is s_l^{j-1}. In view of (3.1) and of the construction of $Q_1(i)$, the k^{th} component of the vector $v_0(i)$ approximates the l^{th} component of the vector $P_I^{i+1,j-1}$, i. e., the steady-state probability that at a service completion epoch of the $(i+1)^{st}$ server, the buffers $i, i-1, \ldots, j-1$ are all empty and the $(j-1)^{th}$ server is in s_l^{j-1} service phase.

Similarly, if $k = \Sigma_{i+1}^j + l$ for $i < j < N$ and $1 \leq l \leq m_{j+1}$, then the k^{th} component of the $1 \times \Sigma_{i+1}^N$ block $u_{K_i-1}(i)$ of $u(i)$ is the joint steady-state probability that at a service completion epoch in the first server, it is blocked and the phase of the second node server is s_l^{j+1}. The k^{th} component of the vector $u_{K_i-1}(i)$ therefore approximates the l^{th} component of the vector $P_B^{i,j+1}$, i. e., the steady-state probability that at a service completion epoch of the i^{th} server, the servers $i, i+1, \ldots, j$ are all blocked and the $(j+1)^{st}$ server is in its s_l^{j+1} service phase.

In vector form these approximations can be rewritten as

$$\left(P_I^{i+1,i}, \ldots, P_I^{i+1,1}\right) \approx v_0(i) \,, \quad 1 \leq i < N \tag{3.4a}$$

and

$$\left(P_B^{i,i+1}, \ldots, P_B^{i,N}\right) \approx u_{(K_i-1)}(i) \,, \quad 1 \leq i < N. \tag{3.4b}$$

The scalars $P_0(i)$ and $P_F(i)$ defined by the relations

$$P_0(i) = v_0(i)\, e_{\Sigma_1^i} \qquad \text{and} \qquad P_F(i) = u_{(K_i-1)}(i)\, e_{\Sigma_{i+1}^N} \,, \quad 1 \leq i < N,$$

thus provide approximations to P_I^{i+1} and P_B^i, respectively. Consequently, the initialization vectors $\alpha_1(i)$ and $\alpha_2(i+1)$ in (3.3b) and (3.2b) can be *approximated* by the quantities $\alpha_1^a(i)$ and $\alpha_2^a(i+1)$ given by

$$\alpha_1^a(i) = \left[\left(1 - P_0(i-1)\right)\, \alpha_i \,, v_0(i-1)\right] \tag{3.5a}$$

and

$$\alpha_2^a(i+1) = \left[\left(1 - P_F(i+1)\right)\, \alpha_{i+1} \,, u_{K_{i+1}-1}(i+1)\right] \,. \tag{3.5b}$$

This approximation leads to an iterative approach based on the two node analysis summarized in the Appendix. The marginal probability distribution of the queue sizes at buffer i can be obtained once the vectors $v_0(i-1)$ and $u_{(K_{i+1}-1)}(i+1)$ are known. These vectors are calculated from the two node approximations of the $(i-1)^{st}$ and $(i+1)^{st}$ queues, respectively. More precisely, with some abuse in the notation, denoting the iteration count by a superscript, the steps of the iterative procedure can be described as follows:

Step 1. Select an initial value for the vectors $u^0_{(K_i-1)}(i)$ for all $1 < i < N$ and compute $\alpha^0_2(i)$ from (3.5b). Set $\alpha^0_2(N) = \alpha_N$.

Step 2. At iteration n;

For $i = 1, 2, \ldots, N-1$,

Starting from the first queue, solve the effective two node systems for each buffer:

a. Set $\alpha^n_1(1) = \alpha_1$ and $\alpha^n_2(N) = \alpha_N$.

b. Compute $\alpha^n_1(i)$ and $\alpha^n_2(i)$ by using $v^n_0(i-1)$ and $u^{n-1}_{(K_i-1)}(i)$, respectively.

c. Obtain the vectors $v^n_0(i)$ and $u^n_{(K_i-1)}(i)$ by using the results of the Appendix.

Step 3. Test for a convergence criterion, namely that all the marginal queue length probabilities in successive iterations are required to be within ϵ of each other. If this requirement is not satisfied, set n to $n+1$ and go to Step 2.

Although no proof of convergence is available, the algorithm has been tested on many examples and has always converged with $\epsilon = 10^{-4}$ in about only 5 iterations. Since at each node all the servers are taken into account, the computational complexity of the algorithm grows quadratically with N, the number of servers in tandem. In fact, the proposed algorithm can be computationally prohibitive for *very* long tandem lines with high dimensional PH-type distributions. However, in such cases, at the expense of introducing further approximations, the following modifications can easily be incorporated. Firstly, the computational complexity of the algorithm can be made *linear* in N by considering only a few immediate neighboring nodes in the effective representations. Secondly, the effective representations $Q_k(i)$, $k = 1, 2$ and $1 \leq i \leq N$, can be *approximated* by lower order PH-distributions. On the other hand, the algorithm is eminently suitable for parallel computations and can be made much faster by implementing it on a parallel machine. With a different processor assigned to the solution of each two node system, processor i needs information only from its immediate neighbors, namely from processors $i+1$ and $i-1$, to update the initialization vectors $\alpha_1(i)$ and $\alpha_2(i+1)$.

4. DECOMPOSITION FOR THE NON-IMMEDIATE BLOCKING POLICY

The decomposition described in Section 3 for systems that operate under the immediate blocking strategy can also be made for systems operating under the non-immediate blocking strategy. Since only the type of blocking is different, the representation (3.3) given for the pair $(\alpha_1(i), Q_1(i))$ remains unchanged.

However, by an argument similar to the one given in Section 3, the effective representation for the pair $(\alpha_2(i+1), Q_2(i+1))$ is now seen to be

$$Q_2(i+1) = \begin{pmatrix} Q_{i+1} & p_{i+1} & (P_B^{i+1,i+2}, & \ldots, & P_B^{i+1,N}) \\ & Q_{i+2} & & & \\ & & \ddots & & \\ & & & Q_{N-1} & \\ & & & & Q_N \end{pmatrix}, \quad p_2(i+1) = \begin{pmatrix} (1 - P_B^{i+1}) p_{i+1} \\ p_{i+2} \\ \vdots \\ p_N \end{pmatrix} \tag{4.1a}$$

and

$$\alpha_2(i+1) = (\alpha_{i+1}, 0_{m_{i+2}}, \ldots, 0_{m_N}) \qquad (4.1b)$$

for $1 \leq i < N$. An iterative approximation algorithm similar to the one developed in Section 4 is also available for this case; details are omitted for the sake of brevity.

5. NUMERICAL EXAMPLES

The accuracy of the algorithm is tested against simulations, and both relative and absolute errors in the approximations are listed. Although relative errors are usually a more important measure of accuracy of an approximation scheme, absolute errors play a more important role when determining the relative or absolute errors associated with the computation of *some* of the performance measures of interest. To illustrate this point, *average queue sizes* at each buffer location are calculated. Both the relative and the absolute errors for this performance measure are better predicted by the *absolute* errors in the calculation of the probabilities. For instance, in Example 1, the approximation of the probability of having an empty buffer at the first queue yields 17.6% relative error, while the absolute error is only about 1%. The relative error in the average queue size for this buffer is 1.9%. Generally speaking, in all the examples the approximations are within ± 0.02 of the simulations. Such an accuracy is considered reasonable since, due to blocking, the system usually reaches steady state rather slowly. Therefore, the simulations are also believed to be accurate only up to the second digit after the decimal point.

The first five examples are for the *discrete-time* model under the *immediate blocking* strategy and involves only *geometric* servers. Examples 6-17 are for the *continuous-time* model operating under the *non-immediate* blocking policy and involves more general service time distributions. The accuracy of the approximations seems not to be affected by high blocking probabilities (e.g., Examples 1,6,7,9-12) or in the presence of bottleneck(s) (e.g., Examples 11 and 12). Also, the approximations work equally well even when service time distributions have small or large squared coefficient of variations (e.g., $c_i^2 = 0.2$, $i = 2, 3$ in Example 10 and $2 < c^2 < 10$ in Examples 13-17).

Examples 2-5 indicate that the *reversibility* property discussed in Chapter 3 of [6] (see also [3]) also hold for a general tandem line under the *immediate blocking* strategy. In Examples 2 and 3, where the tandem line is totally symmetric, reversing the order of the servers yield the exact same probabilities. Example 5 is the same as Example 4 except that the order of both the servers and the intermediate buffers is now reversed. The approximation scheme yields totally *symmetric* probabilities. In the simulation results for these examples, the corresponding probabilities change in the third digit after the decimal point, thereby providing an indication of the margin of accuracy of the simulations.

For the numerical values of the Examples 8-13, three independent simulation runs are performed on the Performance Analysis Workstation (PAW) [12]. The results obtained from each simulation were within ± 0.01 of each other. The results listed in the simulation column in Tables 5.8-5.12 are the arithmetic average of these three simulation runs. In these examples, S_i denotes the service time distribution of server i, while $Er(r;\mu)$ (resp. $Er(2;\mu_1,\mu_2)$) denotes the r-stage

Erlang (resp. 2-stage Generalized Erlang) distribution with parameter(s) μ (resp. μ_1 and μ_2), i. e., $\frac{1}{\mu}$ is the average time spent at each stage.

Examples 14-17 are given in order to compare the proposed algorithm (indicated by the column GM) against the one proposed by Jun and Perros (JP) [10]. In these examples, the first server generates Poisson arrivals with rate λ and jobs that arrive into the system when the first buffer is full are assumed lost. The service times at each node are two-stage Coxian with mean $1/\mu$ and squared coefficient of variation c^2. The corresponding PH-representation is given by

$$Q = \begin{pmatrix} -\gamma_1 & \theta\gamma_1 \\ 0 & -\gamma_2 \end{pmatrix} \quad \text{and} \quad \alpha = (1, 0) ,$$

where

$$\gamma_1 = 2\mu , \qquad \gamma_2 = \frac{\mu}{c^2} \qquad \text{and} \qquad \theta = \frac{1}{2c^2} .$$

The simulation data is taken from [10], where the average queue length and the probabilities of having an empty buffer and a full buffer, denoted respectively by $E(L)$, $p(0)$ and $p(N)$, are also available for each buffer. The comparison of the corresponding data shows that both approximation algorithms perform well relatively to the simulation data and have comparable accuracy except in Example 17 where all the servers have high variability. In this example the algorithm presented in this paper provides a better approximation in all the entries of Table 5.17. However, the algorithm of Jun and Perros was 2-3 times faster than the proposed algorithm. This is mainly because the proposed algorithm is implemented for general PH-distributions and does not make use of the special structure of the matrices in the effective representations. As mentioned at the end of Section 3, while several steps can be taken to reduce the CPU time, the emphasis in the present paper is mainly on the structure of the approximation methodology rather than on the implementation details.

Example 1 : $N = 3,\ K_1 = 2,\ K_2 = 1$. Immediate blocking.
Geometric servers; $p_i = 0.5,\ i = 1, 2, 3$.

Buffer	Queue Size	Approx.	Exact Sol'n.	Rel. Error	Abs. Error
1	0	0.0526	0.0638	0.1755	0.0112
	1	0.4211	0.4256	0.0106	0.0045
	2	0.5263	0.5106	-0.0307	-0.0157
Average queue length:		1.4737	1.4468	-0.0186	-0.0269
2	0	0.5263	0.5106	-0.0307	-0.0157
	1	0.4737	0.4894	0.0321	0.0157
Average queue length:		0.4737	0.4894	0.0321	0.0157

Table 5.1.

Example 2 : $N = 4,\ K_i = 2,\ i = 1, 2, 3$. Immediate blocking.
Geometric servers; $p_i = 0.5,\ i = 1, 2, 3, 4$.

Buffer	Queue Size	Approx.	Exact Sol'n.	Rel. Error	Abs. Error
1	0	0.1444	0.1652	0.1259	0.0208
	1	0.4470	0.4596	0.0274	0.0126
	2	0.4086	0.3754	-0.0884	-0.0332
Average queue length:		1.2643	1.2104	-0.0444	-0.0538
2	0	0.2643	0.2621	-0.0084	-0.0022
	1	0.4715	0.4757	0.0088	0.0042
	2	0.2643	0.2621	-0.0084	-0.0022
Average queue length:		1.0001	0.9999	-0.0002	-0.0002
3	0	0.4086	0.3754	-0.0884	-0.0332
	1	0.4470	0.4596	0.0274	0.0126
	2	0.1444	0.1652	0.1259	0.0208
Average queue length:		0.7358	0.7900	0.0686	0.0542

Table 5.2.

Example 3 : $N = 4,\ K_i = 4,\ i = 1,2,3$. Immediate blocking.
Geometric servers; $p_i = 0.5,\ i = 1,2,3,4$.

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.074	0.079	0.063	0.005
	1	0.174	0.186	0.065	0.012
	2	0.234	0.235	0.004	0.001
	3	0.313	0.305	-0.026	-0.008
	4	0.205	0.195	-0.054	-0.010
Average queue length:		2.401	2.311	-0.039	-0.090
2	0	0.131	0.127	-0.031	-0.004
	1	0.246	0.247	0.004	0.001
	2	0.247	0.247	0.000	0.000
	3	0.246	0.249	0.012	0.003
	4	0.131	0.130	0.008	0.001
Average queue length:		2.002	2.008	0.003	0.006
3	0	0.205	0.195	-0.051	-0.010
	1	0.313	0.308	-0.016	-0.005
	2	0.234	0.238	0.017	0.004
	3	0.174	0.182	0.044	0.008
	4	0.074	0.076	0.026	0.002
Average queue length:		1.599	1.634	0.021	0.035

Table 5.3.

Example 4 : $N = 5,\ K_1 = K_3 = 2,\ K_2 = K_4 = 3.$ Immediate blocking.
Geometric servers; $p_1 = 0.7,\ p_2 = 0.8,\ p_3 = 0.9,\ p_4 = 0.75,\ p_5 = 0.6.$

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.168	0.176	0.045	0.008
	1	0.599	0.601	0.003	0.002
	2	0.233	0.223	-0.045	-0.010
Average queue length:		1.065	1.047	-0.017	-0.018
2	0	0.109	0.118	0.076	0.009
	1	0.340	0.334	-0.018	-0.006
	2	0.389	0.377	0.032	0.012
	3	0.162	0.171	0.053	0.009
Average queue length:		1.604	1.601	-0.002	-0.003
3	0	0.116	0.103	-0.126	-0.013
	1	0.590	0.589	-0.002	-0.001
	2	0.294	0.308	0.045	0.014
Average queue length:		1.178	1.205	0.022	0.027
4	0	0.106	0.091	-0.165	-0.015
	1	0.331	0.305	-0.085	-0.026
	2	0.395	0.416	0.050	0.021
	3	0.168	0.188	0.106	0.020
Average queue length:		1.625	1.701	0.045	0.076

Table 5.4.

Example 5 : $N = 5,\ K_1 = K_3 = 3,\ K_2 = K_4 = 2.$ Immediate blocking.
Geometric servers; $p_1 = 0.6,\ p_2 = 0.75,\ p_3 = 0.9,\ p_4 = 0.8,\ p_5 = 0.7.$

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.168	0.191	0.120	0.023
	1	0.395	0.413	0.044	0.018
	2	0.331	0.303	-0.092	-0.028
	3	0.106	0.093	-0.140	-0.013
Average queue length:		1.375	1.298	-0.059	-0.077
2	0	0.294	0.308	0.045	0.014
	1	0.590	0.589	-0.002	-0.001
	2	0.116	0.103	-0.126	-0.013
Average queue length:		0.822	0.776	-0.059	-0.046
3	0	0.175	0.172	0.058	0.010
	1	0.389	0.376	-0.035	-0.013
	2	0.340	0.333	-0.021	-0.007
	3	0.109	0.119	0.084	0.010
Average queue length:		1.396	1.337	-0.044	-0.059
4	0	0.230	0.224	-0.040	-0.009
	1	0.559	0.604	0.008	0.005
	2	0.168	0.172	0.023	0.004
Average queue length:		0.935	0.939	0.004	0.004

Table 5.5.

Example 6 : $N = 3,\ K_i = 1,\ i = 1,2$. Non-immediate blocking.
Exponential servers; $\mu_i = 1,\ i = 1,2,3$.

Buffer	Queue Size	Approx.	Exact Sol'n.	Rel. Error	Abs. Error
1	0	0.222	0.205	-0.083	-0.017
	1	0.778	0.795	0.021	0.017
2	0	0.444	0.436	-0.018	-0.008
	1	0.556	0.564	0.014	0.008

Table 5.6.

Example 7 : $N = 3,\ K_i = 1,\ i = 1,2$. Non-immediate blocking.
Exponential servers; $\mu_1 = 1,\ \mu_2 = 1.75,\ \mu_3 = 1.5$.

Buffer	Queue Size	Approx.	Exact Sol'n.	Rel. Error	Abs. Error
1	0	0.401	0.398	-0.008	-0.003
	1	0.599	0.602	0.005	0.003
2	0	0.509	0.519	0.019	0.010
	1	0.491	0.481	-0.021	-0.010

Table 5.7.

Example 8 : $N = 3,\ K_1 = 2,\ K_2 = 3$. Non-immediate blocking.
$S_1 = Er(2;2,1),\ S_3 = Er(2;2.5,1)$,
S_2 =Hyperexponential; $Q = \mathrm{diag}(-1,-0.5),\ \alpha = (0.3,0.7)$.

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.158	0.155	-0.019	-0.003
	1	0.242	0.248	0.024	0.006
	2	0.601	0.597	-0.007	-0.004
Average queue length:		1.444	1.442	-0.001	-0.002
2	0	0.355	0.351	-0.011	-0.004
	1	0.286	0.288	0.007	0.002
	2	0.185	0.195	0.051	0.010
	3	0.174	0.166	-0.048	-0.008
Average queue length:		1.178	1.176	-0.002	-0.002

Table 5.8.

Example 9 : $N = 3,\ K_1 = 3,\ K_2 = 2$. Nonimmediate blocking.
$S_1 = Er(2;1),\ S_2 = Er(2;\frac{1}{1.2}),\ S_3 = Er(2;1,\frac{1}{1.2})$.

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.035	0.027	-0.296	-0.008
	1	0.104	0.091	-0.143	-0.013
	2	0.205	0.207	0.010	0.002
	3	0.656	0.675	0.028	0.019
Average queue length:		2.482	2.530	0.019	0.048
2	0	0.235	0.231	-0.017	-0.004
	1	0.353	0.353	0.000	0.000
	2	0.412	0.415	0.007	0.003
Average queue length:		1.177	1.183	0.005	0.006

Table 5.9.

Example 10 : $N = 3,\ K_1 = 2,\ K_2 = 3$. Non-immediate blocking.
$S_1 = Er(1;0.25),\ S_2 = Er(5;1.5),\ S_3 = Er(5;1)$.

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.171	0.170	-0.006	-0.001
	1	0.269	0.251	-0.072	-0.018
	2	0.560	0.579	0.033	0.019
Average queue length:		1.389	1.397	0.006	0.008
2	0	0.042	0.058	0.276	0.016
	1	0.126	0.129	0.023	0.003
	2	0.240	0.225	-0.067	-0.015
	3	0.592	0.588	-0.007	-0.004
Average queue length:		2.382	2.343	-0.017	-0.039

Table 5.10.

Example 11 : $N = 6$, $K_i = 3$, $1 \leq i \leq 5$. Non-immediate blocking.
$S_i = Er(2;1)$, $1 \leq i \neq 4 \leq 6$, $S_4* = Er(2;0.5)$.

Average Queue Lengths

Buffer	Approx.	Simulation	Rel. Error
1	2.66	2.85	0.067
2	2.79	2.84	0.018
3	2.82	2.83	0.004
4	0.68	0.67	-0.015
5	0.68	0.69	-0.014

Table 5.11.

Example 12 : $N = 10$, Non-immediate blocking.

$$K_i = \begin{cases} 3, & i = 1,2,5,6,8 \\ 2, & i = 4,7,9 \\ 4, & i = 3 \end{cases}, \qquad S_i = \begin{cases} \text{Exponential, } \mu = 0.5, & i = 1,3,6,8 \\ \text{Exponential, } \mu = 1, & i = 10 \\ \text{Exponential, } \mu = 0.2, & i = 4^* \\ Er(2;1), & i = 2,5,7 \\ Er(2;0.5), & i = 9^*. \end{cases}$$

Average Queue Lengths

Buffer	Approx.	Simulation	Rel. Error
1	2.69	2.84	0.053
2	2.80	2.87	0.024
3	3.77	3.77	0.000
4	0.52	0.52	0.000
5	0.58	0.63	0.063
6	0.66	0.77	0.143
7	0.72	0.77	0.065
8	1.59	1.58	-0.006
9	0.21	0.21	0.000

Table 5.12.

* Bottleneck

Example 13 : $N = 3, K_1 = K_2 = 3$. Non-immediate blocking.
$Q_i = \text{diag}(-0.1, -10)$, $\alpha_i = (0.5, 0.5)$, $c_i^2 = 2.93$, $i = 1, 2, 3$.
(Hyperexponential)

Buffer	Queue Size	Approx.	Simulation	Rel. Error	Abs. Error
1	0	0.214	0.205	-0.044	-0.009
	1	0.104	0.114	0.088	0.010
	2	0.118	0.122	0.033	0.004
	3	0.563	0.559	-0.007	-0.004
Average queue length:		2.029	2.019	-0.005	-0.010
2	0	0.428	0.427	-0.002	-0.001
	1	0.135	0.133	-0.015	-0.002
	2	0.118	0.118	0.000	0.000
	3	0.319	0.322	0.009	0.003
Average queue length:		1.328	1.335	0.005	0.007

Table 5.13.

Example 14 : $N = 3, K_1 = K_2 = K_3 = 3$. Non-immediate blocking.
$\lambda = 3.0$, $\mu_i = 4$, $i = 1, 2, 3$, $c_1^2 = 2$, $c_2^2 = 4, c_3^2 = 8$.

Buffer		Simulation	GM	Rel. Error	JP	Rel. Error
1	$p(0)$	0.253	0.264	-0.043	0.244	0.036
	$p(K)$	0.353	0.343	0.028	0.361	-0.023
	$E(L)$	1.646	1.609	0.022	1.668	-0.013
2	$p(0)$	0.336	0.345	-0.027	0.337	-0.003
	$p(K)$	0.374	0.364	0.027	0.386	-0.032
	$E(L)$	1.537	1.507	0.020	1.559	-0.014
3	$p(0)$	0.516	0.518	-0.004	0.521	-0.010
	$p(K)$	0.239	0.236	0.013	0.250	-0.046
	$E(L)$	1.053	1.045	0.008	1.069	-0.015

Table 5.14.

Example 15 : $N = 3$, $K_1 = K_2 = K_3 = 5$. Non-immediate blocking.
$\lambda = 4.0$, $\mu_i = 4$, $c_i^2 = 2$, $i = 1, 2, 3$.

Buffer		Simulation	GM	Rel. Error	JP	Rel. Error
1	$p(0)$	0.110	0.118	-0.073	0.116	-0.055
	$p(K)$	0.310	0.303	0.023	0.310	0.000
	$E(L)$	3.125	3.094	0.010	3.088	0.012
2	$p(0)$	0.193	0.199	-0.031	0.197	-0.021
	$p(K)$	0.325	0.323	0.006	0.319	0.018
	$E(L)$	2.822	2.796	0.009	2.798	0.009
3	$p(0)$	0.314	0.316	-0.006	0.310	0.013
	$p(K)$	0.201	0.201	0.000	0.198	0.015
	$E(L)$	2.108	2.101	0.003	2.115	-0.003

Table 5.15.

Example 16 : $N = 3$, $K_1 = K_3 = 5$, $K_2 = 3$. Non-immediate blocking.
$\lambda = 4.0$, $\mu_1 = \mu_3 = 3$, $\mu_2 = 5$, $c_i^2 = 4$, $i = 1, 2, 3$.

Buffer		Simulation	GM	Rel. Error	JP	Rel. Error
1	$p(0)$	0.059	0.070	-0.186	0.066	-0.119
	$p(K)$	0.495	0.478	0.034	0.495	0.000
	$E(L)$	3.763	3.333	0.114	3.711	0.014
2	$p(0)$	0.378	0.383	-0.013	0.386	-0.021
	$p(K)$	0.375	0.363	0.032	0.367	0.021
	$E(L)$	1.486	1.455	0.021	1.464	0.015
3	$p(0)$	0.327	0.326	0.003	0.326	0.003
	$p(K)$	0.298	0.295	0.010	0.296	0.007
	$E(L)$	2.373	2.369	0.002	2.388	0.006

Table 5.16.

Example 17 : $N = 3$, $K_1 = K_2 = K_3 = 3$. Non-immediate blocking.
$\lambda = 4.0$, $\mu_i = 4$, $c_i^2 = 8$, $i = 1, 2, 3$.

Buffer		Simulation	GM	Rel. Error	JP	Rel. Error
1	$p(0)$	0.187	0.196	-0.048	0.172	0.080
	$p(K)$	0.524	0.514	0.019	0.540	-0.031
	$E(L)$	2.004	1.969	0.017	2.051	-0.023
2	$p(0)$	0.333	0.338	-0.015	0.340	-0.021
	$p(K)$	0.438	0.431	0.016	0.469	-0.071
	$E(L)$	1.647	1.626	0.013	1.692	-0.027
3	$p(0)$	0.532	0.528	0.008	0.540	-0.015
	$p(K)$	0.254	0.256	-0.008	0.272	-0.071
	$E(L)$	1.060	1.071	-0.010	1.088	-0.026

Table 5.17.

APPENDIX

THE TWO NODE TANDEM SYSTEM

The discrete-time model

Consider the tandem system described in Section 2 when $N = 2$ and $K_1 = K$. A natural state space E for this system is the one that contains $r := (K-1)m_1m_2 + m_1 + m_2$ states with

$$E = \begin{cases} (i,0)\,, & 1 \le i \le m_1\ ,\ \ k = 0, \\ (i,k,j)\,, & 1 \le i \le m_1,\ 0 < k < K \text{ and } 1 \le j \le m_2, \\ (K,j)\,, & 1 \le j \le m_2\ ,\ \ k = K, \end{cases}$$

where k indicates the buffer size, and i and j represent the service phase in the first and the second node server, respectively.

A discrete-time Markov chain is naturally associated with E; its one-step transition matrix T is described in [6]. Any invariant probability vector for T is denoted by the $1 \times r$ row vector π, which is partitioned into $K+1$ blocks of components, say $\pi = (\pi_0, \pi_1, \ldots, \pi_K)$, with π_0, π_k, $0 < k < K$, and π_K being row vectors of dimension $1 \times m_1$, $1 \times m_1 m_2$ and $1 \times m_2$, respectively.

First, several matrices are defined in terms of the system parameters.

$$\begin{aligned}
&M = (I_{m_1} - e_{m_1}\,\alpha_1) \otimes I_{m_2} + (e_{m_1}\,\alpha_1 - Q_1) \otimes Q_2 && m_1 m_2 \times m_1 m_2, \\
&N = I_{m_1} \otimes (I_{m_2} - e_{m_2}\,\alpha_2) + Q_1 \otimes (e_{m_2}\,\alpha_2 - Q_2) && m_1 m_2 \times m_1 m_2, \\
&R = M\ N^{-1} && m_1 m_2 \times m_1 m_2, \\
&S = [(I_{m_1} - Q_1) \otimes \alpha_2]\ N^{-1} && m_1 \times m_1 m_2, \\
&U = (Q_2 \otimes p_1)\,(I_{m_2} - Q_2)^{-1} && m_1 m_2 \times m_2, \\
&W = I_{m_1} + \left[\textstyle\sum_{k=1}^{K-1}\ S\ R^{k-1}\right]\ (I_{m_1} \otimes e_{m_2}) && m_1 \times m_1, \\
&Z = W + S\,R^{K-2}\ U\,e_{m_2}\,x && m_1 \times m_1,
\end{aligned}$$

where the $1 \times m_1$ row vector $x = \alpha_1(I_{m_1} - Q_1)^{-1}/\alpha_1(I_{m_1} - Q_1)^{-1}e_{m_1}$ is the invariant probability vector of $Q_1 + p_1\alpha_1$.

When $K > 1$, if either the matrix M *or* N is invertible, then a closed-form solution can be obtained for the vector π. The main result of [6] is summarized in the following theorem when N is invertible.

Theorem A.1. *If the matrix N is invertible, then any invariant probability vector* $\pi = (\pi_0, \ldots, \pi_K)$ *has the following* **matrix geometric** *form*

$$\pi_k = \pi_0\ S\ R^{k-1}\ ,\quad 1 \le k < K \tag{A.1a}$$

and

$$\pi_K = \pi_0\ S\ R^{K-2}\ U\ , \tag{A.1b}$$

where the vector π_0 *satisfies the equation*

$$\pi_0\ Z = x\ . \tag{A.2}$$

When $K = 1$, i. e., there is no intermediate buffer, a simpler expression is available in [4, 6].

The following necessary and sufficient conditions for the invertibility of the matrices M and N are given in [6]. To state them, let $\mathcal{C}$ be the *open* disc centered at $(\frac{1}{2},0)$ in the complex plane with radius $\frac{1}{2}$ and let $Sp(X)$ denote the spectrum of the matrix X.

Lemma A.1. *If $Sp(A) \subseteq \mathcal{C}$ (resp. $Sp(B) \subseteq \mathcal{C}$) then the matrix N (resp. M) is nonsingular.*

Lemma A.2. *The matrix N (resp. M) is singular if the matrix A (resp. B) is singular.*

Let u (resp. v) be the steady state probability vector of the Markov chain associated with E embedded at points of arrival (resp. departure). These vectors are partitioned into K blocks of components, say $u = (u_0, u_1, \ldots, u_{K-1})$ and $v = (v_0, v_1, \ldots, v_{K-1})$, with u_k and v_k, $0 \leq k < K$, being $1 \times m_2$ and $1 \times m_1$ row vectors, respectively. The i^{th} component of u_k (resp. v_k) is the joint steady-state probability that at a service completion epoch at the first (resp. second) server, there are k jobs in the buffer, excluding the job that has just been serviced, and that the phase of the second (resp. first) server is i. The following theorem establishes simple relations between u, v and π; the proofs are similar to the ones given in [3] and are therefore omitted.

Theorem A.2. *Define the scalars c and d by*

$$c = \pi_0 p_1 + \sum_{1}^{K-1} \pi_k (p_1 \otimes e_{m_2}) \tag{A.3a}$$

and

$$d = \pi_K p_2 + \sum_{1}^{K-1} \pi_k (e_{m_1} \otimes p_2)\,. \tag{A.3b}$$

For $K > 1$ the relations

$$cu_0 = (\pi_0 p_1)\alpha_2 + \pi_1 (p_1 \otimes p_2 \alpha_2)\,, \tag{A.4a}$$

$$cu_k = \pi_k (p_1 \otimes Q_2) + \pi_{k+1}(p_1 \otimes p_2 \alpha_2), \quad 1 \leq k < K-1, \tag{A.4b}$$

$$cu_{K-1} = \pi_{K-1}(p_1 \otimes Q_2) \tag{A.4c}$$

and

$$dv_0 = \pi_1 (Q_1 \otimes p_2)\,, \tag{A.5a}$$

$$dv_k = \pi_k (p_1 \alpha_1 \otimes p_2) + \pi_{k+1}(Q_1 \otimes p_2) \quad 1 \leq k < K-1, \tag{A.5b}$$

$$dv_{K-1} = \pi_{K-1}(p_1 \alpha_1 \otimes p_2) + (\pi_K p_2)\alpha_1\,, \tag{A.5c}$$

hold true, while for $K = 1$, $u_0 = \alpha_2$ and $v_0 = \alpha_1$.

The continuous-time model

Equations (A.1) and (A.2) of Theorem A.1 also holds for the continuous-time formulation of the two node model. However, the matrices M, N, S and U are now given by

$$
\begin{aligned}
M &= (I_{m_1} - e_{m_1}\,\alpha_1) \otimes Q_2 + Q_1 \otimes I_{m_2} && m_1\,m_2 \times m_1\,m_2,\\
N &= Q_1 \otimes (I_{m_2} - e_{m_2}\,\alpha_2) + I_{m_1} \otimes Q_2 && m_1\,m_2 \times m_1\,m_2,\\
S &= [Q_1 \otimes \alpha_2]\ N^{-1} && m_1 \times m_1\,m_2,\\
U &= -(p_1 \otimes I_{m_2})\,Q_2^{-1} && m_1\,m_2 \times m_2
\end{aligned}
$$

and the matrices M and N are *always both* invertible.

For the continuous-time model, the equations (A.4) and (A.5) of Theorem A.2 become

$$cu_0 = (\pi_0 p_1)\alpha_2\,, \tag{A.6a}$$

$$cu_k = \pi_k(p_1 \otimes I_{m_2}), \quad 1 \le k < K, \tag{A.6b}$$

and

$$dv_k = \pi_{k+1}(I_{m_1} \otimes p_2) \quad 0 \le k < K-1, \tag{A.7a}$$

$$dv_{K-1} = (\pi_K p_2)\alpha_1\,, \tag{A.7b}$$

where the scalars c and d are again given by (A.3).

REFERENCES

[1] T.Altıok, "Approximate analysis of production lines with general service and repair times and with finite buffers," *IE Report* **84-4**, Rutgers University, New Brunswick, NJ, 1984.

[2] T. Altıok and S.Stidham, Jr. , "A note on transfer lines with unreliable machines, random processing times and finite buffers," *IIE Trans.* **14**, pp. 125-127, 1982.

[3] M.H. Ammar and S.B. Gershwin, "Equivalence relations in queueing models of manufacturing networks," *Proceedings of the 19th IEEE Conference on Decision and Control,* Albuquerque, NM, pp. 715-721, 1980.

[4] J.A. Buzacott and D. Kostelski, "Matrix-geometric and recursive algorithm solution of a two-stage unreliable flow line," *IIE Trans.* **19**, pp. 429-438, 1987.

[5] S. Chakravarthy and M. F. Neuts, "Algorithms for the design of finite capacity service units," *Working Paper* **87-002**, Systems and Industrial Engineering Department, The University of Arizona, Tucson, AZ, 1987.

[6] L. Gün, *Tandem queueing systems subjects to blocking with phase-type servers: Analytical solutions and approximations*, M. S. Thesis, Electrical Engineering Department, University of Maryland, College Park, MD, August 1986. Also available as *SRC Technical Report* **87-002**, Systems Research Center, University of Maryland, College Park, MD, 1987.

[7] L. Gün, "Annotated bibliography on blocking systems," *SRC Technical Report* **87-187**, Systems Research Center, University of Maryland, College Park, MD, 1987.

[8] L. Gün and A.M. Makowski, "Matrix-geometric solution for two node tandem queueing systems with phase-type servers subject to blocking and failures", *Stochastic Models*, forthcoming, 1989.

[9] G.C. Hunt, "Sequential arrays of waiting lines," *Operations Res.* 4, pp. 674-683, 1956.

[10] K.P. Jun and H.G. Perros, "An approximate analysis of open tandem queueing networks with blocking and general service times," *Technical Report*, Computer Science Department, North Carolina State University, Raleigh, NC, 1987.

[11] M.F. Neuts, *Matrix-geometric Solutions in Stochastic Models - An Algorithmic Approach,* The John Hopkins University Press, Baltimore, MD, 1981.

[12] B. Melamed, "The performance analysis workstation: An interactive animated simulation package for queueing networks," *Proceedings of Fall Joint Computer Conference,* Dallas, TX, 1986.