Numerical Computation of Sojourn-Time Distributions in Queuing Networks

BENJAMIN MELAMED

AT&T Bell Laboratories, Holmdel, New Jersey

AND

MICHA YADIN

Technion-Israel Institute of Technology, Haifa, Israel

Abstract. Sojourn time distributions in queuing networks seldom possess closed-form analytical solutions. When overtaking is permitted, the sojourn times at individual nodes are usually dependent, in which case the attendant distribution is mathematically intractable. In a previous paper the authors proposed a methodology utilizing randomization procedures to approximate sojourn time distributions in arbitrary discrete-state Markovian queuing networks. This paper addresses the computational aspects of the methodology pertaining to implementation. Ways of improving the accuracy of the approximated distribution functions are also discussed.

Categories and Subject Descriptors: D.4.8 **[Operating Systems]**: Performance—*queuing theory*; E.2 **[Data Storage Representations]**: Contiguous Representations; F.2.1 **[Analysis of Algorithms and Problem Complexity]**: Numerical Algorithms and Problems—*computations on matrices*; G.1.0 **[Numerical Analysis]**: General—*numerical algorithms*; G.m **[Miscellaneous]**: Queuing Theory

General Terms: Algorithms, Theory

Additional Key Words and Phrases: sojourn times, queuing networks, randomization, procedures, approximation of sojourn time distributions

1. *Introduction*

The problem of computing sojourn time distributions in queuing networks is among the hardest in *queuing network theory*. Closed-form analytical solutions are the exception rather than the rule. Roughly speaking, the difficulties are brought about when overtaking is allowed to occur. This introduces complicated and subtle dependencies among the sojourn times of a customer at the individual nodes even though each sojourn-time distribution at a node is far more amenable to computation.

The basic and fundamental result was given by Reich [13] who showed that in a sequence of M/M/1 queues in tandem the stationary sojourn-time distribution is a sum of independent sojourn times at the nodes, each of which is exponentially distributed. Such sojourn times are referred to as *simple*.

Authors' addresses: B. Melamed, AT&T Bell Laboratories, Holmdel, NJ 07733; M. Yadin, Technion-Israel Institute of Technology, Haifa, Israel.

Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the ACM copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Association for Computing Machinery. To copy otherwise, or to republish, requires a fee and/or specific permission.
© 1984 ACM 0004-5411/84/1000-0839 $00.75

A number of generalizations to the class of Jackson networks [5, 6] have emerged recently; these include branching networks [8] and overtake-free paths [11, 16]. Several counterexamples suggest that the scope of simple sojourn times cannot be further extended [15, 16]. Thus, approximation methods are called for to fill in the gap.

In [12] we proposed a general methodology that uses the so-called randomization procedure to approximate first passage-time distributions in discrete-state Markov processes—sojourn times in queuing systems being a special case.

The notion of randomization procedure has been known for some time [1–4, 7, 9, 14] (Keilson [7] calls it uniformization procedure). Grassmann [2–4] and Barzily and Gross [1] have used randomization to compute transient solutions of Markov processes. Grassmann [2] was the first to use this methodology to compute waiting-time distributions in an M/M/1 queue. The relevant part of the methodology is summarized in the next section; for an excellent additional discussion by Gross and Miller, see [4a].

2. *The Randomization Methodology*

Let $X = \{X(t) : t \geq 0\}$ be a regular continuous time Markov process with a countable state space E and a bounded infinitesimal generator matrix Q. The elements of Q are denoted $Q(x, y)$, $x, y \in E$, and $Q(x) \triangleq \sum_{y \in E-\{x\}} Q(x, y)$, $x \in E$, are the absolute values of the elements on the diagonal of Q. Furthermore, let ξ_t be the state probability vector of $X(t)$; that is, $\xi_t(x) = P\{X(t) = x\}$, $x \in E$.

X models the evolution of a queuing system during the sojourn of a particular (tagged) customer in it. The states in E have two major components: the state of the queuing system and the location of the tagged customer. Let A be the subset of states that describe the queuing system before that customer departed it; further let B be the subset of states that describe the queuing system just after that customer departed it.

Clearly A and B partition E, that is, $A \cup B = E$ and $A \cap B = \Phi$. Furthermore, if T is the time the process X spends in A up until hitting B (for the first time), then T is precisely the sojourn time of our tagged customer in the network. (An example showing how to construct A, B, and Q is deferred until Section 3.)

We assume that X will eventually land in B with probability 1. Since the evolution of the system is irrelevant once the tagged customer departs, we may assume that B is a closed set; that is, the process X cannot return to A from B.

The quantity of interest is the distribution function $\tau(t)$ of T. We observe that

$$\tau(t) \triangleq P\{T \leq t\} = P\{X(t) \in B\} = 1 - P\{X(t) \in A\}, \qquad t \geq 0, \tag{2.1}$$

because our assumptions ensure the equality of events $\{T \leq t\} = \{X(t) \in B\}$.

It follows from (2.1) that the problem of computing $\tau(t)$ is equivalent to computing the transient distribution of $X(t)$ on A. Thus we need to compute the vector ξ_t, $t \geq 0$. If P_t, $t \geq 0$, are the transition matrices of X, then

$$\xi_t = \xi_0 P_t, \qquad t \geq 0, \tag{2.2}$$

where P_t has the representation

$$P_t = \exp(Qt) \triangleq \sum_{n=0}^{\infty} \frac{t^n}{n!} Q^n, \qquad t \geq 0. \tag{2.3}$$

The so-called *randomization procedure* consists of using in eq. (2.2) an equivalent representation:

$$P_t = \exp(-\alpha t)\exp\left[\alpha t\left(I + \frac{1}{\alpha}Q\right)\right] \triangleq \exp(-\alpha t)\sum_{n=0}^{\infty}\frac{\alpha^n t^n}{n!}R^n, \tag{2.4}$$

where $R \triangleq I + (1/\alpha)Q$ is the *randomized matrix*, I the *identity matrix*, and α *any positive number bounding from above all* $Q(x)$, $x \in E$.

Although eq. (2.4) appears more complex than (2.3), it does, in fact, enjoy favorable computational properties. Most important, Q is not a stochastic matrix, whereas R is. Consequently, the computation using (2.4) is stable, whereas that using (2.3) may be unstable. Furthermore, the randomization procedure has an interesting probabilistic meaning that proves useful in deriving bounds for $\tau(t)$. Specifically, R being a stochastic matrix determines a discrete-time Markov process $Y = \{Y_n : n = 0, 1, \ldots\}$ provided we take $Y_0 = X(0)$. With this stipulation, the relation between the processes X and Y is quite simple, as follows.

Let us extend the discrete-time process Y to a continuous-time Markov process such that

(1) the time intervals between jumps are independent identically distributed (iid) exponential random variables with mean $1/\alpha$;
(2) the jumps themselves are governed by R.

It can be shown that the resulting process is precisely the original process X; however, whenever we have a sequence of Y jumps from a state $x \in E$ to itself, it will be perceived in X as one long sojourn in the state x [12].

Thus, the randomization procedure can be thought of as sprinkling the process X with random dummy jumps in between substantive jumps. The resultant process, say $\tilde{X}$ (in which the dummy jumps are visible), has the same probabilistic structure as X but with one advantage: The sequence of jump instants in $\tilde{X}$ (dummy and substantive) now forms a Poisson process (this is not generally the case in X). Observe that Y_n is the state of $\tilde{X}$ at the instant of its nth jump (dummy or substantive). Suppose now that $\tilde{X}$ hits set B on its nth jump. Given that, the sojourn time of $\tilde{X}$ (and therefore also of X) on A is the sum of n independent exponential random variables each with mean $1/\alpha$; that is, the sojourn time has an Erlang distribution of order n and parameter α (denoted $E_{n,\alpha}(t)$).

Let $h(n)$ be the probability that $\tilde{X}$ hits B on its nth jump. Further, let ϕ_n denote the state probability vector of Y_n; that is,

$$\phi_n = \xi_0 R^n. \tag{2.5}$$

The quantities $h(n)$ are given by two equivalent formulas:

$$h(n) = \begin{cases} \sum_{x\in B}\phi_0(x), & n = 0, \\ \sum_{x\in A}\sum_{y\in B}\phi_{n-1}(x)R(x, y), & n > 0; \end{cases} \tag{2.6}$$

or

$$h(n) = \begin{cases} 1 - \sum_{x\in A}\phi_0(x), & n = 0, \\ \sum_{x\in A}\phi_{n-1}(x) - \sum_{x\in A}\phi_n(x), & n > 0. \end{cases} \tag{2.7}$$

Given the probabilities $h(n)$, we can write (recall that we assumed $\sum_{n=0}^{\infty} h(n) = 1$)

$$\tau(t) = \sum_{n=0}^{\infty} h(n)E_{n,\alpha}(t) = 1 - \sum_{n=0}^{\infty} h(n)\bar{E}_{n,\alpha}(t), \qquad t \geq 0, \tag{2.8}$$

where $\bar{E}_{n,\alpha}(t) = 1 - E_{n,\alpha}(t)$ and $E_{0,\alpha}$ assigns the probability atom 1 to $t = 0$. Equation (2.8) can be formally derived from eqs. (2.1)–(2.5) as follows:

$$\tau(t) = \sum_{y \in B} \xi_t(y) = e^{-\alpha t} \sum_{n=0}^{\infty} \frac{\alpha^n t^n}{n!} \sum_{y \in B} \phi_n(y) = e^{-\alpha t} \sum_{n=0}^{\infty} \frac{\alpha^n t^n}{n!} \sum_{k=0}^{n} h(k)$$

because $\sum_{y \in B} \phi_n(y) = \sum_{k=0}^{n} h(k)$ is the probability of hitting B in at most n jumps of $\tilde{X}$; eq. (2.8) now follows by interchanging the summations.

Equation (2.8) leads to simple bounds on $\tau(t)$ which can, in principle, be made arbitrarily tight. For any integer $K \geq 0$ define

$$L_K(t) \triangleq \sum_{n=0}^{K} h(n)E_{n,\alpha}(t), \qquad t \geq 0, \tag{2.9}$$

$$U_K(t) \triangleq 1 - \sum_{n=0}^{K} h(n)\,\bar{E}_{n,\alpha}(t), \qquad t \geq 0, \tag{2.10}$$

whence we obtain the bounds

$$L_K(t) \leq \tau(t) \leq U_K(t), \qquad t \geq 0.$$

Furthermore, if for some $\epsilon > 0$, K is chosen as the stopping rule,

$$K = \min\left\{n \geq 0 : \sum_{k=0}^{n} h(k) \geq 1 - \epsilon\right\} \triangleq K(\epsilon), \tag{2.11}$$

or, equivalently (because $\sum_{k=0}^{n} h(k) + \sum_{y \in A} \phi_n(y) = 1$, for all $n \geq 0$),

$$J = \min\left\{n \geq 0 : \sum_{y \in A} \phi_n(y) \leq \epsilon\right\} = J(\epsilon), \tag{2.12}$$

then it is straightforward to show that

$$| \tau(t) - L_{J(\epsilon)}(t) | \leq \epsilon \qquad \text{and} \qquad | \tau(t) - U_{J(\epsilon)}(t) | \leq \epsilon$$

uniformly in $t \geq 0$.

The purpose of this paper is to discuss the computational aspects of these bounds. We shall also describe how these bounds may be combined to yield improved approximations.

Notational Note. In the remainder of the paper ϕ_n^A is the subvector of ϕ_n restricted to elements in A, $R^{A,A}$ is the submatrix of R restricted to elements in $A \times A$, and so on.

3. *The Approximation Paradigm with a Finite State Space*

We have seen from eqs. (2.9) and (2.10) that the computation of the bounds $L_K(t)$ and $U_K(t)$ of $\tau(t)$ requires the computation of the probabilities $h(n)$. These, in turn, require the computation of the vectors ϕ_n but only on the subset A of the state space. When the state space E is finite (as is the case in closed queuing networks), both the $h(n)$ and ϕ_n can, in principle, be computed exactly (ignoring round-off

errors). The randomization-based paradigm to approximate $\tau(t)$ for this case is summarized below.

THE RANDOMIZATION-BASED PARADIGM TO APPROXIMATE $\tau(t)$: E FINITE

(1) Input:
  (1.1) the infinitesimal generator Q,
  (1.2) the initial distribution vector ϕ_0 of Y_0,
  (1.3) the finite sets A and B which partition E,
  (1.4) some $\epsilon > 0$.

(2) Computational procedure:
  (2.1) Let $\alpha = \sup\{Q(x) : x \in A\}$ and form $R = I + 1/(\alpha Q)$.
  (2.2) Set $n \leftarrow 0$.
  (2.3) Compute $h(n)$ using eq. (2.6) or (2.7).
  (2.4) If $\sum_{x \in A} \phi_n(x) < \epsilon$ (i.e., $n = J(\epsilon)$, see eq. (2.12)), then stop; otherwise set $n \leftarrow n + 1$ and go to Step (2.5).
  (2.5) Compute $\phi_n^A = \phi_{n-1}^A R^{A,A}$.
  (2.6) Go to Step (2.3).

(3) Output:
  (3.1) the lower bound $L_{J(\epsilon)}(t)$ for $\tau(t)$, computed via eq. (2.9);
  (3.2) the upper bound $U_{J(\epsilon)}(t)$ for $\tau(t)$, computed via eq. (2.10).

4. *The Approximation Paradigm with a Truncated State Space*

In practice, the state space E is often infinite, or finite but prohibitively large. On such occasions it becomes necessary to truncate E to a finite subset $\hat{E}$ usually chosen to satisfy, for some prescribed $\epsilon_0 > 0$,

$$\sum_{x \in \hat{E}} \phi_0(x) > 1 - \epsilon_0. \tag{4.1}$$

The result is that another level of approximation is added since the $h(n)$, ϕ_n, etc., must now be approximated. From now on we consistently denote the approximating quantities by appending carets to the original symbols. Specifically we define $\hat{A} \triangleq A \cap \hat{E}$, $\hat{B} \triangleq B \cap \hat{E}$, and denote by $\hat{R} \triangleq I^{\hat{E} \times \hat{E}} + (1/\alpha) Q^{\hat{E} \times \hat{E}}$ the restriction of R to elements in $\hat{E} \times \hat{E}$. Let $\hat{\phi}_0 = \phi_0^{\hat{E}}$ be the restriction of ϕ_0 to elements in $\hat{E}$ and define

$$\hat{\phi}_n = \hat{\phi}_{n-1} \hat{R}, \qquad n \geq 1. \tag{4.2}$$

Because transitions from $E - \hat{E}$ into $\hat{E}$ are discarded, it is clear that, for all $n \geq 0$,

$$\hat{\phi}_n(x) \leq \phi_n(x), \qquad x \in \hat{E}.$$

Furthermore, eq. (2.6) leads to a lower bound $\hat{h}(n)$ on $h(n)$ given by

$$\hat{h}(n) \triangleq \begin{cases} \sum_{x \in \hat{B}} \hat{\phi}_0(x), & n = 0, \\ \sum_{x \in \hat{A}} \sum_{y \in \hat{B}} \hat{\phi}_{n-1}(x) \hat{R}(x, y), & n > 0. \end{cases} \tag{4.3}$$

Thus, for each $n > 0$, the quantity

$$D_n \triangleq 1 - \sum_{k=0}^{n} \hat{h}(k) - \sum_{x \in \hat{A}} \hat{\phi}_n(x) \geq 0 \tag{4.4}$$

may be appropriately termed *the probability loss incurred in the nth step* in view of the fact that the exact $h(n)$ and ϕ_n satisfy

$$\sum_{k=0}^{n} h(k) + \sum_{x \in A} \phi_n(x) = 1, \qquad n \geq 0.$$

The presence of probability loss has several consequences. First, if we form the natural approximation $\hat{\hat{h}}(n)$ based on eq. (2.7), namely,

$$\hat{\hat{h}}(n) = \begin{cases} 1 - \sum_{x \in \hat{A}} \hat{\phi}_0(x), & n = 0, \\ \sum_{x \in \hat{A}} \hat{\phi}_{n-1}(x) - \sum_{x \in \hat{A}} \hat{\phi}_n(x), & n > 0, \end{cases} \tag{4.5}$$

then $\hat{h}(n)$ and $\hat{\hat{h}}(n)$ are not equal. In fact, letting $D_{-1} \triangleq 0$, we have

$$\hat{\hat{h}}(n) = \hat{h}(n) + D_n - D_{n-1} \geq \hat{h}(n), \qquad n \geq 0.$$

Moreover,

$$\sum_{k=0}^{n} \hat{\hat{h}}(k) \geq \sum_{k=0}^{n} h(k), \qquad n \geq 0,$$

so that the approximate distribution function induced by the $\hat{\hat{h}}(n)$ is an upper bound on the exact distribution function of the number of jumps to hit B. A second consequence of probability loss is that the stopping rules attendant on (2.11) and (2.12), namely,

$$\hat{K}(\epsilon) \triangleq \min\left\{n \geq 0 : \sum_{k=0}^{n} \hat{h}(k) \geq 1 - \epsilon\right\}, \tag{4.6}$$

$$\hat{J}(\epsilon) \triangleq \min\left\{n \geq 0 : \sum_{x \in \hat{A}} \hat{\phi}_n(x) \leq \epsilon\right\}, \tag{4.7}$$

are also *not equivalent.*

In fact, $\hat{K}(\epsilon) \geq K(\epsilon)$ may not even be a viable stopping rule when $\sum_{n=0}^{\infty} \hat{h}(n) < 1$ because the sum may never exceed $1 - \epsilon$. Fortunately, $\hat{J}(\epsilon) \leq J(\epsilon)$ is always a finite stopping rule and should be the one adopted in practice.

Finally, the functions

$$\hat{L}_{\hat{J}(\epsilon)}(t) \triangleq \sum_{n=0}^{\hat{J}(\epsilon)} \hat{h}(n) E_{n,\alpha}(t), \tag{4.8}$$

$$\hat{U}_{\hat{J}(\epsilon)}(t) \triangleq 1 - \sum_{n=0}^{\hat{J}(\epsilon)} \hat{h}(n) \bar{E}_{n,\alpha}(t), \tag{4.9}$$

are still lower and upper bounds, respectively, for $\tau(t)$ albeit less tight than $L_{J(\epsilon)}$ and $U_{J(\epsilon)}$.

With the obvious notation for restricted vectors and matrices, we now summarize the approximation paradigm when truncation to a finite $\hat{E}$ is called for.

THE RANDOMIZATION-BASED PARADIGM TO APPROXIMATE $\tau(t)$: E TRUNCATED TO $\hat{E}$.

(1) Input:

(1.1) the infinitesimal generator Q,

(1.2) the initial distribution ϕ_0 of Y_0,

(1.3) the sets A and B that partition E,
(1.4) some $\epsilon_0 > 0$ and some $\epsilon > 0$.

(2) Computational procedure:
(2.1) Define a finite $\hat{E}$ satisfying $\sum_{x\in\hat{E}} \phi_0(x) > 1 - \epsilon_0$.
(2.2) Let $\hat{A} = A \cap \hat{E}$, $\hat{B} = B \cap \hat{E}$, $\alpha = \sup\{Q(x) : x \in \hat{A}\}$ and form the finite matrix $\hat{R} = I^{\hat{E}\times\hat{E}} + (1/\alpha)Q^{\hat{E}\times\hat{E}}$.
(2.3) Set $n \leftarrow 0$.
(2.4) Compute $\hat{h}(n)$ using eq. (4.3).
(2.5) If $\sum_{x\in\hat{A}} \hat{\phi}_n(x) < \epsilon$ (i.e., $n = \hat{J}(\epsilon)$; see eq. (4.7)), then stop; otherwise set $n \leftarrow n + 1$ and go to Step (2.6).
(2.6) Compute $\hat{\phi}_n = \hat{\phi}_{n-1}\hat{R}$.
(2.7) go to Step (2.4).

(3) Output:
(3.1) the lower bound $\hat{L}_{\hat{J}(\epsilon)}(t)$ for $\tau(t)$, computed via eq. (4.8);
(3.2) the upper bound $\hat{U}_{\hat{J}(\epsilon)}(t)$ for $\tau(t)$, computed via eq. (4.9).

5. *Sojourn Times in Open Jackson Networks*

In this section we illustrate in some detail how to implement the approximation paradigm of the previous section in the context of sojourn times in open Jackson queuing networks [5, 6]. We also provide a storage scheme for the state space suitable for a computer implementation.

Consider a stable Jackson network [5] with node set $M = \{1, 2, \ldots, m\}$, external Poisson arrivals with rates $\lambda_1, \ldots, \lambda_m$, single exponential servers with rates $\mu_1, \ldots, \mu_m$, and a switching (substochastic) matrix $P = [p_{ij}]_{i,j\in M}$. The probability of leaving the network from node i is $p_{io} \triangleq 1 - \sum_{j\in M} p_{ij}$.

We are interested in the sojourn-time distribution experienced by customers entering the system through some node, say node k^*.

Let $Z = \{Z(t) : t \geq 0\}$ be the state process of the network in which $Z(t) = (Z_1(t), \ldots, Z_m(t))$ is the vector of customer totals at each node at time t. Let further $C = (K, S)$ be the process that tracks a tagged customer during his or her sojourn in the network; more specifically, $K(t)$ is the node and $S(t)$ the position in the queue at that node occupied by the tagged customer at time t. It is convenient to take $(K(t), S(t)) = (0, 0)$ to mean that at time t the customer has already left the network. Thus,

$$\tau(t) = P\{(K(t), S(t)) = (0, 0)\}. \tag{5.1}$$

The process Z is a Markov process with values in N_+^m, N_+ being the set of nonnegative integers. The process $X = (Z, C)$ is also Markov with values in the set

$$E = \{((n_1, \ldots, n_m), (k, s)) : (n_1, \ldots, n_m) \in N_+^m, k \in M, n_k \geq 1, 1 \leq s \leq n_k\} \cup \{((n_1, \ldots, n_m), (0, 0)) : (n_1, \ldots, n_m) \in N_+^m\} = A \cup B. \tag{5.2}$$

It is known that the stationary distribution of (Z, C) just after an arrival at node k^* is (see, e.g., Melamed [10])

$$\phi_0((n_1, \ldots, n_m), (k, s)) = \begin{cases} \dfrac{1}{\rho_k} \displaystyle\prod_{j=1}^{m} (1 - \rho_j)\rho_j^{n_j}, & \text{if } n_k > 0, k = k^* \text{ and } s = n_k, \\ 0, & \text{otherwise,} \end{cases} \tag{5.3}$$

where $\rho_i = \theta_i/\mu_i$ are the relative traffic intensities at the nodes and the θ_i solve the system of linear equations

$$\theta_i = \lambda_i + \sum_{j \in M} \theta_j p_{ji}, \qquad i \in M.$$

To simplify the notation, we henceforth denote $z = (n_1, \ldots, n_m) \in N_+^m$ so that $x = (z, k, s) \in E$. Further let e_i denote the m-dimensional unit vector along the ith coordinate. Let δ denote Kronecker's delta, $\bar{\delta} \triangleq 1 - \delta$ be the complementary Kronecker delta and $\beta(n) \triangleq \bar{\delta}_{n,0}$.

Observe that for the Markov process X

$$Q(z, k, s) = \sum_{i \in M} [\lambda_i + \mu_i[\beta(n_i)(1 - p_{ii})\bar{\delta}_{i,k} + \delta_{i,k}]], \qquad (z, k, s) \in E. \quad (5.4)$$

Note that for $n_k = 1$, the second term on the right-hand side (RHS) of (5.4) should, in fact, vanish for $i = k$ (when the target customer resides alone in node k, service completions followed by feedback do not change the state). It is however, convenient to regard such service completions as state changing to simplify the ensuing eqs. (5.6) that take account of this particular definition of $Q(x)$.

Now, choose

$$\alpha = \max_{1 \le k \le m} \sum_{i \in M} [\lambda_i + \mu_i((1 - p_{ii})\bar{\delta}_{i,k} + \delta_{i,k})]. \quad (5.5)$$

The recursive computational scheme of the ϕ_n becomes (for given ϕ_0)

$$\begin{aligned}
&\alpha\phi_n(z, k, s) \\
&\quad = \phi_{n-1}(z, k, s)[\alpha - Q(x)] && (5.6.1)\\
&\qquad + \sum_{i \in M-\{k\}} \phi_{n-1}(z - e_i, k, s)\lambda_i\beta(n_i) && (5.6.2)\\
&\qquad + \phi_{n-1}(z - e_k, k, s)\lambda_k\bar{\delta}_{s,n_k} && (5.6.3)\\
&\qquad + \sum_{j \in M-\{k\}} \phi_{n-1}(z + e_j, k, s)\mu_j p_{j0} && (5.6.4)\\
&\qquad + \phi_{n-1}(z + e_k, k, s + 1)\mu_k p_{k0} && (5.6.5)\\
&\qquad + \sum_{i \in M-\{k\}} \sum_{\substack{j \in M-\{k\} \\ j \ne i}} \phi_{n-1}(z + e_j - e_i, k, s)\mu_j p_{ji}\beta(n_i) && (5.6.6)\\
&\qquad + \sum_{i \in M-\{k\}} \phi_{n-1}(z + e_k - e_i, k, s + 1)\mu_k p_{ki}\beta(n_i) && (5.6.7)\\
&\qquad + \phi_{n-1}(z, k, s + 1)\mu_k p_{kk}\bar{\delta}_{s,n_k} && (5.6.8)\\
&\qquad + \sum_{j \in M-\{k\}} \phi_{n-1}(z + e_j - e_k, k, s)\mu_j p_{jk}\bar{\delta}_{s,n_k} && (5.6.9)\\
&\qquad + \sum_{j \in M} \phi_{n-1}(z + e_j - e_k, j, 1)\mu_j p_{jk}\delta_{s,n_k} && (5.6.10)\\
&\qquad\qquad \text{for all} \quad x = (z, k, s) \in A && (5.6)
\end{aligned}$$

The reader should note that these equations are correct whether or not $n_k = 1$. In particular, for $n_k = 1$, terms (5.6.1) and (5.6.10) combine to give the proper result since $Q(x) - (5.6.10)$ is the correct transition rate.

The derivation of the recursive scheme (5.6) requires careful analysis of the jumps of X. In our case we have ten terms on the RHS that upon division by α become the probabilities of transitions of the Y process into state x due to

(5.6.1): a dummy jump from x to itself,
(5.6.2): arrivals at nodes other than k,
(5.6.3): an arrival at node k,

(5.6.4): departures from nodes other than k,
(5.6.5): a departure from node k,
(5.6.6): transfers within the network not involving node k,
(5.6.7): transfers from node k to nodes other than k not involving the tagged customer,
(5.6.8): transfers from node k to itself (feedback) not involving the tagged customer,
(5.6.9): transfers to node k from nodes other than k not involving the tagged customer,
(5.6.10): transfers due to service completion of the tagged customer.

Note that the $\beta(n_i)$ are in fact superfluous; they were included merely to make the scheme more amenable to programming as they represent a check for $n_i > 0$. Likewise, δ_{s,n_k} and $\overline{\delta}_{s,n_k}$ (which are not superfluous) represent the checks $s = n_k$ and $s < n_k$, respectively.

Finally, the $h(n)$ are obtained from the ϕ_{n-1} by

$$h(n) = \sum_{z \in N_+^m} \sum_{j \in M} \phi_{n-1}(z + e_j, j, 1)\mu_j p_{j0}. \tag{5.7}$$

5.1. State Space Truncation and Storage. Since the state space E for open Jackson networks is infinite, a truncation of E to a finite subset $\hat{E}$ is obviously required for practical computation. To do that, we must select for each node i a finite waiting room d_i, $1 \le i \le m$, such that, for some prescribed ϵ_0,

$$\sum_{x \in \hat{E}} \phi_0(x) = \sum_{x \in \hat{A}} \phi_0(x) \ge 1 - \epsilon_0,$$

the equality above holding because $\phi_0(x) = 0$, $\forall x \notin A$ (see eq. (5.3)).

Let us allocate an error ϵ_0/m to each node in the sense that

$$d_i = \min \left\{ j : \sum_{n_i > j} \phi_0(x) = \rho_i^j \le \frac{\epsilon_0}{m} \right\};$$

that is, d_i is the smallest integer j for which $P\{Z_i > j\} \le \epsilon_0/m$, whence from (5.3)

$$d_i = \left\lceil \frac{\log(\epsilon_0/m)}{\log \rho_i} \right\rceil, \qquad 1 \le i \le m. \tag{5.8}$$

Equation (5.8) determines $\hat{E}$ as

$$\hat{E} = \{((n_1, \ldots, n_m), (k, s)) : 0 \le n_i \le d_i,\ i \in M,\ k \in M,\ n_k \ge 1,\ 1 \le s \le n_k\}$$
$$\cup \{((n_1, \ldots, n_m), (0, 0)) : 0 \le n_i \le d_i,\ i \in M\} = \hat{A} \cup \hat{B}. \tag{5.9}$$

We now proceed to describe how to store the arrays $\hat{\phi}_n$ whose coordinates are indexed by $\hat{A}$. Let $|\hat{A}|$ be the cardinality of $\hat{A}$ and let V be a vector such that $|V| = 2|\hat{A}|$. Our goal is to define two hashing functions C and C' that map each state $x = ((n_1, \ldots, n_m), (k, s)) \in \hat{A}$ into two coordinates $C(x)$ and $C'(x)$ of the vector V. We require both C and C' to be injective, to have disjoint ranges, and to be mutually adjoint in the sense that the mapping

$$f(C(x)) \triangleq C'(x), \qquad x \in \hat{A},$$

also satisfies

$$f(C'(x)) = C(x), \qquad x \in \hat{A}.$$

Thus, if we denote $f(C'(x)) \triangleq C''(x)$, adjointness means that $C''(x) = C(x)$, $x \in \hat{A}$.

The rationale for C and C' should now be clear. The vector V can store successive pairs of the vectors $\hat{\phi}_n$ and $\hat{\phi}_{n+1}$. Thus, initially the elements of $\hat{\phi}_0$ will be stored in the set of coordinates $\{C(x): x \in \hat{A}\}$ of V and as $\hat{\phi}_1$ is computed (from $\hat{\phi}_0$) its elements will be stored in the set of coordinates $\{C'(x): x \in \hat{A}\}$. Next, $\hat{\phi}_2$ is computed from $\hat{\phi}_1$ and stored in $\{C(x): x \in \hat{A}\}$, etc. The computation proceeds in a ping-pong manner. Observe that for a fixed state $x \in \hat{A}$, the values $\hat{\phi}_n(x)$ are always stored in either coordinate $C(x)$ or $C'(x)$ of V.

We now proceed to define C and C' and to show that they possess the requisite properties.

Define the quantities u_i, $1 \le i \le m+1$, by

$$u_i = \prod_{j=0}^{i-1} (d_j + 1) \qquad \text{where} \quad d_0 \triangleq 0. \tag{5.10}$$

Note that the number of states of Z restricted to lie in $\hat{A}$ is $u_{m+1} = \prod_{i=1}^{m} (d_i + 1)$.

If $\hat{A}(k)$ is the subset of $\hat{A}$ given that the tagged customer is at node k, then

$$|\hat{A}(k)| \triangleq \frac{u_{m+1}}{d_k + 1} \sum_{j=1}^{d_k} j = \frac{u_{m+1}}{d_k + 1} \frac{(1 + d_k)d_k}{2} = \frac{u_{m+1}d_k}{2}, \qquad 1 \le k \le m.$$

The total number of states of $\hat{A}$ and in V (the storage complexity) is then

$$|\hat{A}| = \sum_{k=1}^{m} \frac{u_{m+1}d_k}{2}, \qquad |V| = \sum_{k=1}^{m} u_{m+1}d_k. \tag{5.11}$$

Thus V can be partitioned into m successive blocks each of size

$$b_k = u_{m+1}d_k, \qquad 1 \le k \le m.$$

The kth block is used to store the C and C' coordinates of states in $A(k)$.

The leftmost coordinates $L(k)$ of the blocks are given by

$$L(k) = \begin{cases} 1 & \text{if} \quad k = 1, \\ L(k-1) + u_{m+1}d_{k-1} & \text{if} \quad k > 1, \end{cases} \tag{5.12}$$

and the rightmost coordinates $R(k)$ are given by

$$R(k) = \begin{cases} u_{m+1}d_1 & \text{if} \quad k = 1, \\ R(k-1) + u_{m+1}d_k & \text{if} \quad k > 1. \end{cases} \tag{5.13}$$

Finally, define

$$C((n_1, \ldots, n_m), (k, s)) \triangleq L(k) + \sum_{i=1}^{m} u_i n_i + u_{m+1}[s - 1], \tag{5.14}$$

$$C'((n_1, \ldots, n_m), (k, s)) \triangleq R(k) - \sum_{i=1}^{m} u_i n_i - u_{m+1}[s - 1], \tag{5.15}$$

which confirm that the coordinate of a state and its adjoint are both stored in the same block.

PROPOSITION. *C and C' are injective, have disjoint ranges, and are mutually adjoint.*

PROOF. Both $C(x)$ and $C'(x)$ are clearly 1–1. To show disjointness of ranges, note that

$$\sum_{i=1}^{m} u_i d_i = u_{m+1} - 1 \qquad \text{(by summation on } 1 \le i \le m \text{ in (5.10))},$$

whence

$$R(k) = L(k) + u_{m+1}d_k - 1 = L(k) + \sum_{i=1}^{m} u_i d_i + u_{m+1}[d_k - 1].$$

Hence

$$\begin{aligned} C'((n_1, \ldots, n_m), (k, s)) &= L(k) + \sum_{i=1}^{m} u_i d_i + u_{m+1}[d_k - 1] \\ &\quad - \sum_{i=1}^{m} u_i n_i - u_{m+1}[s - 1] \\ &= L(k) + \sum_{i=1}^{m} u_i[d_i - n_i] + u_{m+1}[d_k - s]. \end{aligned} \tag{5.16}$$

Let now $x = ((n_1, \ldots, n_m), (k, s))$ and $x' = ((n_1', \ldots, n_m'), (k', s'))$ be both in $\hat{A}$. Suppose $C(x) = C'(x')$. Then all coefficients of the u_i in (5.14) and (5.16) must coincide.

In particular, we must simultaneously have

$$n_k = d_k - n_k', \qquad s - 1 = d_k - s'.$$

Upon subtraction, we get the relation

$$n_k - s + 1 = s' - n_k'.$$

Now,

$$\begin{aligned} x \in \hat{E} &\Rightarrow s \le n_k \Rightarrow s < n_k + 1 \Rightarrow n_k - s + 1 > 0 \\ &\Rightarrow s' - n_k' > 0 \Rightarrow s' > n_k' \Rightarrow x' \notin \hat{A}, \end{aligned}$$

which contradicts the assumption that $x' \in \hat{A}$. Conclude that the ranges of $C(x)$ and $C'(x)$ are disjoint as claimed.

Finally $C''(x) = C(x)$ for all $x = (z, k, s) \in \hat{A}$ follows from the fact that

$$C(x) + C'(x) = L(k) + R(k). \qquad \square$$

Going back to eq. (5.6), we observe that if $C(z, k, s)$ is given, then

$$\begin{aligned} C'(z, k, s) &= L(k) + R(k) - C(z, k, s), \\ C(z \pm e_i, k, s) &= C(z, k, s) \pm u_i, \\ C(z + e_i - e_j, k, s) &= C(z, k, s) + u_i - u_j, \\ C(z, k, s + 1) &= C(z, k, s) + u_{m+1}, \\ C(z + e_j - e_k, j, 1) &= C(z, k, s) + u_j - u_k + L(j) - L(k) - (s - 1)u_{m+1}. \end{aligned}$$

The corresponding formulas for $C'(z, k, s)$ are symmetric, with the $R(k)$ replacing the $L(k)$, and vice versa.

To sum up, the storage scheme for $\hat{A}$ is time and space efficient because

(1) The multidimensional subset $\hat{A}$ is "unraveled" into a vector; the overhead attendant on accessing a multidimensional array is largely avoided.

(2) The hashing functions C and C' allow an efficient access to coordinates of $\hat{\phi}_{n-1}$ necessary for computing $\hat{\phi}_n$.
(3) All coordinates of V are utilized so that no memory is wasted.

We conclude by pointing out that this scheme is applicable to the case of multiple servers. Equation (5.6) can be easily modified to cover that case by assigning the first position numbers to servers and the remaining ones to the waiting line.

6. *Obtaining Improved Approximation*

A randomization procedure produces two boundings curves for a sojourn-time distribution. It is natural to try to improve on those curves in order to produce an approximation that is better than both. This is especially important when the truncation to $\hat{E}$ produces a relatively high initial error, and the vertical spreads between $\hat{L}(t)$ and $\hat{U}(t)$ are relatively large.

Figure 1 displays the true sojourn-time distribution and the bounds on it produced in a network consisting of two M/M/1 queues in tandem with $\rho = 0.9$ (for further details, see [12]). The rightmost argument is $t_{\max}$, computed as $t_{\max} = (1/\alpha)\,(\hat{J}(\epsilon) + 4\sqrt{\hat{J}(\epsilon)})$, that is, four standard deviations beyond the mean of the highest order Erlang distribution $E_{\hat{J}(\epsilon),\alpha}$ used in the approximation. Requiring the initial probability loss to be bounded by $\epsilon_0 = 0.1$ produced a truncated state space of size 24795.

We tried a number of improvement schemes. The resultant errors are tabulated in Table I, which we now proceed to discuss.

6.1. Strict Bounds. This entry corresponds to the bounding curves in Figure 1. Note that the initial probability loss is very close to $\epsilon_0 = 0.1$ and that the terminal probability loss is about 60 percent larger. The maximal deviations from the true curve are comparable and of the same order of magnitude as the terminal probability loss. The deviations at $t \geq 0$ are defined as $\hat{L}(t) - \tau(t)$ and $\hat{U}(t) - \tau(t)$, respectively. Notice that the first deviation is negative and the second positive.

6.2. Conditional Initial Distribution. In this case we eliminated the initial probability loss by conditioning the initial distribution on the event $\{Y_0 \in \hat{E}\}$; that is, by taking $\hat{\phi}_0(x) \triangleq \phi_0(x)/\sum_{y\in\hat{E}} \phi_0(y)$, $x \in \hat{E}$. The randomization procedure produced new approximations $\hat{h}(n)$ for the $h(n)$ giving rise to $\hat{L}(t)$ and $\hat{U}(t)$ (see eqs. (4.7)–(4.9)). Compared with the strict bounds of Section 6.1, the attendant bounds almost halved the terminal probability loss. But the effect on these bounds is uneven: The maximal deviation of $\hat{L}(t)$ was just about cut in half, but that of $\hat{U}(t)$ remained about the same. Note that in this case the approximations $\hat{L}(t)$ and $\hat{U}(t)$ are no longer bounds on $L(t)$ and $U(t)$, respectively. Observe, for example, that $\hat{L}(t)$ has a positive maximal deviation.

6.3. Alternative Formula for $h(n)$. Suppose we truncate E to $\hat{E}$ and compute the approximations $\hat{\phi}_n$ of the vectors ϕ_n subject to the stopping rule (4.7) as in Section 4. Next, use the approximations $\hat{h}(n)$ for $h(n)$ of eq. (4.5) to compute $\hat{L}(t)$ and $\hat{U}(t)$. Computationally, this has the effect of "reclaiming" the probability loss and funneling it into the $\hat{h}(n)$, which explains why the terminal probability loss equals the initial one. Since $\sum_{k=0}^{n} \hat{h}(k) \geq \sum_{k=0}^{n} h(k)$, $n \geq 0$, one can reasonably expect that putting the $\hat{h}(n)$ in (4.8) and (4.9) will result in $\hat{L}(t)$ and $\hat{U}(t)$ which are compensated in part for the truncation of the infinite series (2.8) to $J(\epsilon)$ terms. From Table I, we see that the attendant $\hat{L}(t)$ and $\hat{U}(t)$ are comparable to those obtained in the previous case.

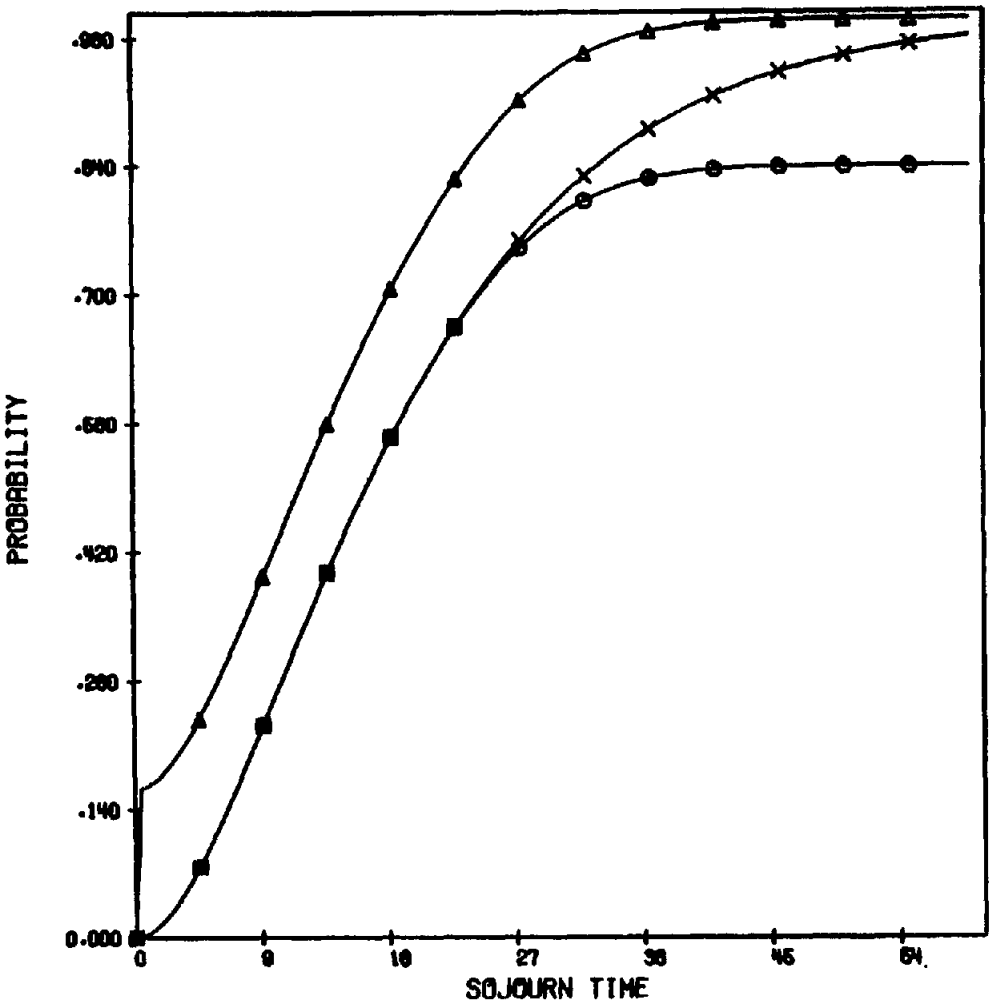

FIG. 1. Exact limiting sojourn time CDF versus its computed strict bounds. Tandem model: $\rho = 0.9$, $\epsilon_0 = 0.10$, $\epsilon = 0.001$. O = computed LB CDF; Δ = computed UB CDF; x = exact CDF.

TABLE I. ERRORS PRODUCED BY BOUNDS AND APPROXIMATIONS OF THE SOJOURN TIME DISTRIBUTION IN A TWO-NODE TANDEM QUEUING NETWORK

	Strict bounds (Section 6.1)		Approximations resulting from conditional initial distribution (Section 6.2)		Approximations resulting from alternative computation of $\hat{h}(n)$ (Section 6.3)		Approximations using weighted combinations of $\hat{L}(t)$ and $\hat{U}(t)$ (Section 6.4)	
	$\hat{L}(t)$	$\hat{U}(t)$	$\hat{L}(t)$	$\hat{U}(t)$	$\hat{L}(t)$	$\hat{U}(t)$	$w(t) = \hat{L}(t)$	$w(t) = \frac{\sum_{n=0}^{\hat{k}(t)} \hat{h}(n)}{t_{\max}} t$
Maximal deviation	−0.142	0.161	0.068	0.145	−0.073	0.147	0.112	0.054
Initial probability loss	0.092	0.092	0.000	0.000	0.092	0.092	0.092	0.092
Terminal probability loss	0.160	0.160	0.075	0.075	0.092	0.092	0.160	0.160

Note. Deviation = approximate value − exact value.

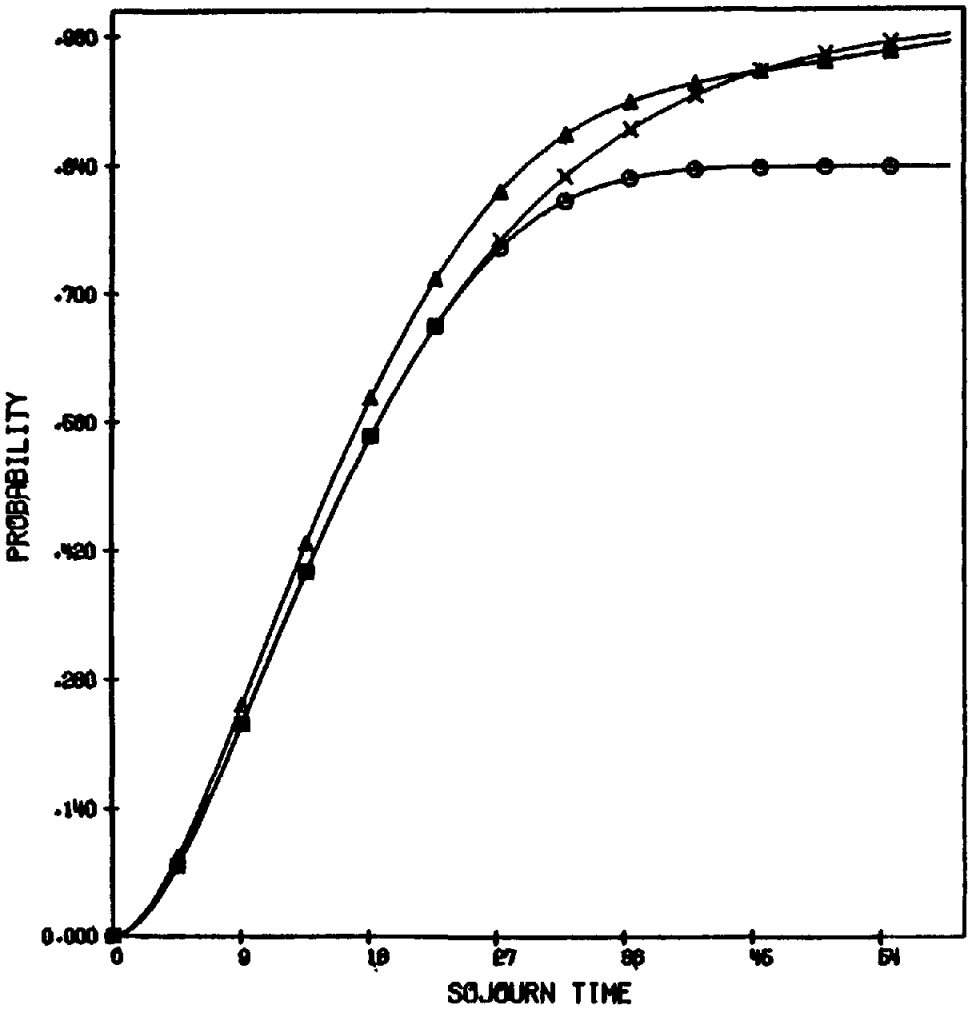

FIG. 2. Exact limiting sojourn time CDF versus its computed strict LB and corrected version. Tandem model linear weighting *fn*: $\rho = 0.9$, $\epsilon_0 = 0.1$, $\epsilon = 0.001$. O = computed LB CDF; Δ = corrected LB CDF; x = exact CDF.

6.4. WEIGHTED COMBINATION OF THE BOUNDING CURVES. The idea is to combine the strict bounds $\hat{L}(t)$ and $\hat{U}(t)$ of eqs. (4.8) and (4.9) for each $t \geq 0$ in a convex combination to produce a new approximation,

$$\hat{\tau}(t) = w(t)\hat{U}(t) + (1 - w(t))\hat{L}(t),$$

where the weighting function $w(t)$ takes values in the interval [0, 1]. The problem is to choose an appropriate weighting function.

An examination of Figure 1 reveals that $\tau(t)$ lies close to $\hat{L}(t)$ in the left-hand region and is relatively close to $\hat{U}(t)$ in the right-hand region. It switches from close proximity to $\hat{L}(t)$ to close proximity to $\hat{U}(t)$ in the middle region. We observed this phenomenon in every model we ran, and the explanation is quite simple. The first few $\hat{h}(n)$ can usually be computed exactly since no truncated states enter into their computation. Since, for small values of t, these $\hat{h}(n) = h(n)$ and $E_{n,\alpha}(t)$ dominate the sum in (2.8), we get $\hat{L}(t) \approx \tau(t)$ in the left-hand region. On the other hand, for $t \gg 0$, we get $\hat{U}(t) \approx 1 \approx \tau(t)$, which explains why $\hat{U}(t) \approx \tau(t)$ in the right-hand side

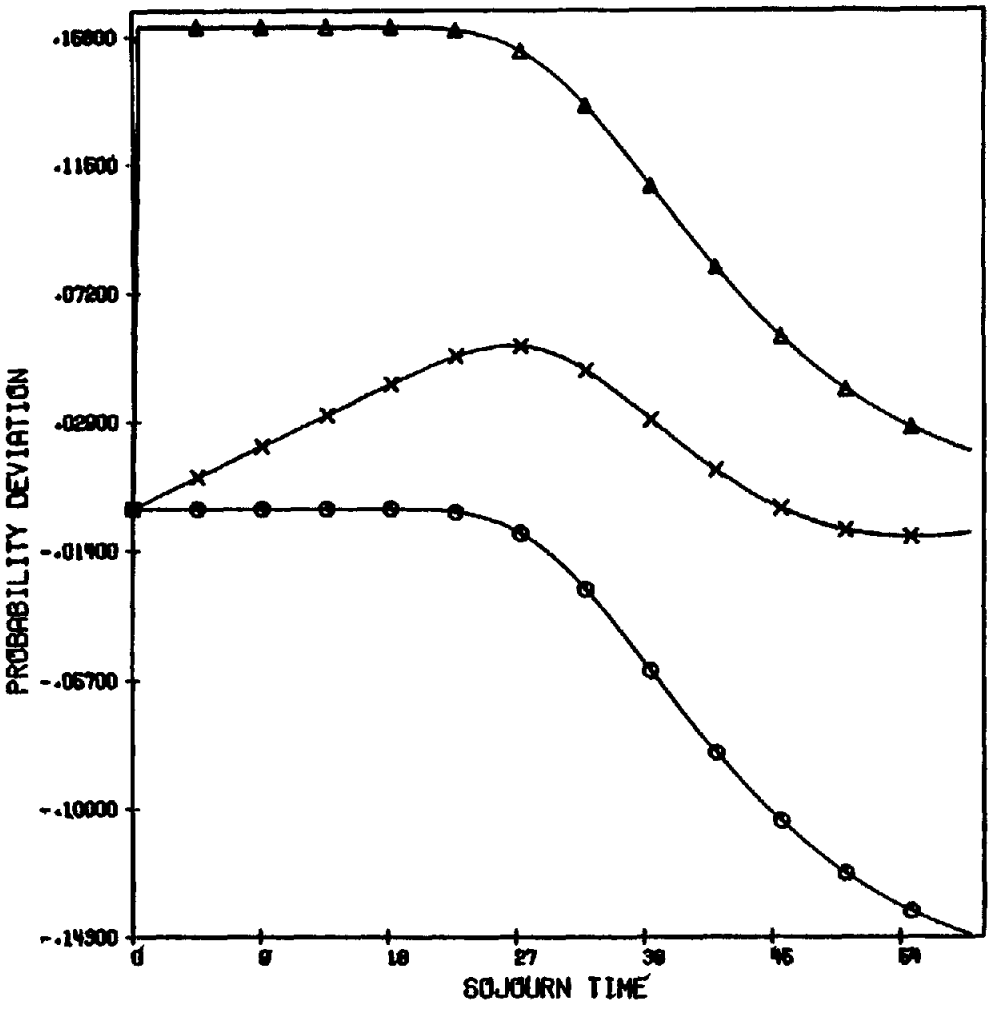

FIG. 3. Deviations of strict and corrected bounds from exact limiting sojourn time CDF. Tandem model linear weighting *fn*: $\rho = 0.9$, $\epsilon_0 = 0.1$, $\epsilon = 0.001$. O = deviation of LB CDF; Δ = deviation of UB CDF; x = deviation of correction.

region. Thus it makes sense to choose a weighting function $w(t)$ that assigns high weights (low weights) to $\hat{L}(t)$ ($\hat{U}(t)$) in the left-hand region, and increasingly higher (lower) weights to the same as t increases in magnitude to the right.

Two such weighting functions were tried out. First we observe that $\hat{L}(t)$ being an approximate distribution function behaves in precisely that way. Table I shows that this resulted in an improvement of some 20 percent over $\hat{L}(t)$ itself.

The second scheme used a linear function of t, specifically the straight line through the points (0, 0) and $(t_{\max}, \sum_{n=0}^{\hat{J}(\epsilon)} \hat{h}(n))$. The effect was to cut the maximal deviations of the strict bounding curves of Section 6.1 to about one-third of their values.

Figure 2 depicts the corrected approximate curve resulting from the linear weighting function $w(t) = [\sum_{n=0}^{\hat{J}(\epsilon)} \hat{h}(n)/t_{\max}]t$, the strict lower bounding curve $\hat{L}(t)$ resulting from Section 6.1, and the true curve $\tau(t)$. Figure 3 displays the deviation curves of this correction from the true curve $\tau(t)$ and the same deviation curves of the strict bounds.

7. *Discussion and Conclusions*

We have discussed computational aspects of randomization procedures when used to compute bounding curves for sojourn-time distributions in discrete-state Markovian queuing networks. An optimal storage scheme was described for open Jackson-type queuing networks, and the corresponding computational paradigm was exhibited for Jackson networks with single-server nodes. Several schemes were tried to obtain improved approximations from the bounding curves produced by the randomization-based approximation paradigms. For the case of two nodes in tandem we found that a linear weighting function produces the best results. We feel that this method is a quick and cheap way to obtain improved approximations for sojourn-time distributions.

REFERENCES

1. BARZILY, Z., AND GROSS, D. *Transient Solutions for Repairable Item Provisioning.* Program in Logistics, George Washington Univ., Washington, D.C., 1979.
2. GRASSMANN, W. K. Transient solutions in Markovian queues. *Eur. J. Op Res. 1* (1977), 396–402.
3. GRASSMANN, W. K. Transient solutions in Markovian queueing systems. *Comp. Op Res. 4* (1977), 47–56.
4. GRASSMANN, W. K. The GI/PH/1 queue: A method to find the transition matrix. Tech. Rep., Univ. of Saskatchewan, Saskatoon, Saskatchewan, Canada, 1981.

4a. GROSS, D., AND MILLER, D. R. The randomization technique as a modeling tool and solution procedure for transient Markov processes. *Oper Res. 32* (1984), 343–361.

5. JACKSON, J. R. Networks of waiting lines. *Op. Res. 5* (1957), 518–521.
6. JACKSON, J. R. Jobshop-like queueing systems. *Management Sci. 10* (1963), 131–142.
7. KEILSON, J. *Markov Chain Models: Rarity and Exponentiality.* Springer Verlag, New York, 1979.
8. LEMOINE, A. J. Total sojourn times in networks of queues. Tech. Rep. 79-020-1, System Control, Inc., Palo Alto, Cal., 1979.
9. LIPPMAN, S. A. Applying a new device in the optimization of exponential queueing systems. *Oper. Res. 23* (1975), 687–710.
10. MELAMED, B. On Markov jump processes imbedded at jump epochs and their queueing-theoretic applications. *Math. Oper. Res. 7* (1982), 111–128.
11. MELAMED, B. Sojourn times in queueing networks. *Math Oper. Res. 7* (1982), 223–244.
12. MELAMED, B., AND YADIN, M. Randomization procedures in the computation of cumulative time distributions over discrete state Markovian processes. *Oper Res. 32*, 4(1984), in press.
13. REICH, E. Waiting times when queues are in tandem. *Ann Math Stat. 28* (1957), 768–773.
14. SERFOZO, R. F. An equivalence between continuous and discrete time Markov decision processes. *Oper. Res. 27* (1979), 616–620.
15. SIMON, B., AND FOLEY, R. D. Some results on sojourn times in acyclic Jackson networks. *Management Sci. 25* (1979) 1027–1034.
16. WALRAND, J., AND VARAIYA, P. Sojourn times and the overtaking condition in Jacksonian networks. *Adv. App Prob. 12* (1980), 1000–1018.

RECEIVED SEPTEMBER 1981; REVISED FEBRUARY 1984; ACCEPTED FEBRUARY 1984