Monte Carlo Summation and Integration Applied to Multiclass Queuing Networks

KEITH W. ROSS

University of Pennsylvania, Philadelphia, Pennsylvania

DANNY H. K. TSANG

The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

AND

JIE WANG

University of Pennsylvania, Philadelphia, Pennsylvania

Abstract. Although many closed multiclass queuing networks have a product-form solution, evaluating their performance measures remains nontrivial due to the presence of a normalization constant. We propose the application of Monte Carlo summation in order to determine the normalization constant, throughputs, and gradients of throughputs. A class of importance-sampling functions leads to a decomposition approach, where separate single-class problems are first solved in a setup module, and then the original problem is solved by aggregating the single-class solutions in an execution model. We also consider Monte Carlo methods for evaluating performance measures based on integral representations of the normalization constant; a theory for optimal importance sampling is developed. Computational examples are given that illustrate that the Monte Carlo methods are robust over a wide range of networks and can rapidly solve networks that cannot be handled by the techniques in the existing literature.

Categories and Subject Descriptors: D.4.8 [**Operating systems**]: Performance—*modeling and prediction*; *queuing theory*; G.3 [**Probability and statistics**]: Probabilistic algorithms (including Monte Carlo); G.m [**Miscellaneous**]: Queuing theory

General Terms: Performance, Theory

Additional Key Words and Phrases: Gradient estimation, product-form queuing networks, variance reduction

---

K. W. Ross and J. Wang were supported in part by NSF grant DDM-9000481. D. H. K. Tsang was supported in part by NSERC grant OPGP0046314.

Authors' addresses: K. W. Ross and J. Wang, Department of Systems, University of Pennsylvania, Philadelphia, PA 19104; email: ross@enia.seas.upenn.edu; wangj@claire.seas.upenn.edu; D. H. K. Tsang, Department of Electrical and Electronic Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong; email: eetsang@usthk.bitnet.

1. *Introduction*

It is well known that many multidimensional stochastic processes give rise to a product-form solution for their equilibrium state probabilities. One of the more important examples is the multiclass queuing network [Baskett et al. 1975; Kelly 1979], where the underlying stochastic process models multiple classes of customers circulating according to a class-dependent routing matrix. Another example is product—form loss networks [Kelly 1986], where the stochastic process models a telephone network supporting calls with different routes and bandwidth requirements.

The appeal of the product-form solution is that it enables one to circumvent solving the global balance equations. Indeed, without the product-form solution, one would have to solve a system of linear equations with a dimension that grows exponentially in the size of problem. The principle limitation of the product-form solution is that it often involves a normalization constant: If the state space is finite, as for closed multiclass queuing networks or for loss networks, then a sum must be performed over an excessively large number of states, a number that again grows exponentially in the problem size. In fact, brute-force summation must be ruled out for all but the "smallest" networks.

Due to the occurrence of product-form queuing networks in numerous important applications, much research effort has been devoted to developing methods to calculate the normalization constant and related performance measures. Currently, there are two schools of thought, the first based on combinatorial algorithms, and the second based on asymptotic expansions.

In 1973, Buzen [1973] initiated the research on combinatorial algorithms with an efficient convolution algorithm to determine the normalization constant in single-class closed queuing networks. In 1975, Reiser and Kobayashi [1975] developed a generalization of the convolution algorithm for closed multiclass queuing networks, and numerous variations and refinements have since been proposed (e.g., see Conway and Georganas [1989], Lam and Lien [1983], Reiser and Lavenberg [1980], and Sauer and Chandi [1981]). Although these algorithms can offer considerable computational savings over brute-force summation, their computational requirements continue to grow exponentially with either the number of classes or the number of stations, rendering them of little use for many problems of practical interest.

Research based on asymptotic expansions began in 1982 with the work of McKenna, Mitra, and Ramakrishnan.[1] The idea here is to expand the normalization constant in $1/N$, where N is a "large parameter" reflecting the population of the network. This technique has given impressive results for large multiclass queuing networks, rapidly solving problems that are completely off limits to the combinatorial algorithms.The technique also forms the core of the software package PANACEA, developed at Bell Laboratories. However, the theory requires that each class visit an infinite-server station and that the "normal-usage" condition be satisfied. Furthermore, although the expansions are applicable to a popular class of multiclass queuing networks, they have not been extended to more exotic product-form networks, such as networks with state-dependent routing [Towsley 1980; Yao and Buzacott 1987] or networks with service and arrival rates that depend on global state information [Kelly

[1]See McKenna and Mitra [1982; 1984], McKenna et al. [1981], Mitra and McKenna [1986], and Ramakrishnan and Mitra [1982].

1979; Serfozo 1989, 1990]. Finally, the theory of asymptotic expansions for product–form networks is mathematically sophisticated and therefore inaccessible to many practitioners who may want to incorporate it in a more comprehensive performance-modeling package.

In this paper, we introduce a new approach, *Monte Carlo summation and integration*, for evaluating performance measures and their gradients of product–form queuing networks. The method is quite simple and can be easily explained to any one familiar with elementary probability. Perhaps more importantly, the method is quite flexible since it can be adapted to arbitrary product–form networks.

We first consider Monte Carlo summation for closed multiclass queuing networks with multiple-server and infinite-server stations. The rough idea is to randomly sample the product–form solution over the state space and then average to obtain a consistent estimate. The performance of the Monte Carlo method is largely determined by the computational effort required to generate a sample and by the variance of the estimate. We develop sampling distributions from which samples are generated in $O(JM)$ time, where J is the number of classes and M the number of stations. Moreover, these sampling distributions give rise to small variances for many important classes of problems.

We then investigate the application of Monte Carlo integration to an integral representation of the normalization constant for closed multiclass networks with single- and infinite-server stations sampling method requiring $O(L)$ effort per sample, where L is the number of single-server stations, and a theory for choosing the optimal parameters of this sampling distribution is developed. A heuristic is also given, which leads to a significant reduction in variance; the sampling distribution and heuristic are given theoretical justification in Ross and Wang [1993].

We have developed a software package, MonteQueue, which implements both Monte Carlo summation and integration for multichain queuing networks; it is available in the public domain at an ftp site.

We give computational results for a variety of networks, all of which are much too large for the combinatorial methods. One set of computational results compares the accuracy of Monte Carlo integration with PANACEA for multiclass networks consisting of single-server and infinite-server stations satisfying the normal-usage conditions (restrictions required by the most recent version of PANACEA): when the utilizations at the single-server stations are not large, both PANACEA and Monte Carlo integration perform quite well; but when the utilizations are increased at the single-server stations, PANACEA breaks down whereas Monte Carlo integration with our sampling heuristic continues to give very accurate results. Another set of computational results compares Monte Carlo summation with Monte Carlo integration. When Monte Carlo integration applies, it typically gives better results than Monte Carlo summation; but Monte Carlo summation is not limited to networks with integral representations; Computational results for Monte Carlo summation are given for a network with multiple-server stations; computational results for summation and integration are also given for a network with no infinite-server nodes. The last set of computational results illustrates that Monte Carlo summation and integration can obtain performance sensitivities.

In a recent paper, Ross and Wang [1992] discuss the successful application of the Monte Carlo summation and importance sampling to product–form loss

networks; see also Ross and Wang, 1990] for some initial ideas on the application of Monte Carlo summation to queuing networks.

In Section 2, we briefly review the product–form formulas for closed multiclass queueing networks. In Section 3, we consider the problem of estimating the normalization constant with Monte Carlo summation. A class of importance-sampling functions leads to a decomposition approach, where separate single-class problems are first solved in a setup module, and then the original problem is solved by aggregating the single-class solutions in an execution module. In Section 4, we consider the problem of estimating throughputs and sensitivities of throughputs with the Monte Carlo summation method. In Section 5, we consider the multidimensional integral representation of the normalization constant, which holds for a large class of product–form networks. In Section 6, we present computational results. In Section 7, we summarize our results and list several problems that merit attention in future research.

2. *Product–Form Queuing Networks*

The queuing networks being considered consist of M stations, J classes, and a population of N_j customers for each class j. There are two different types of stations depending on the service discipline: first-come-first-serve (FCFS) stations and infinite-server (IS) stations. Throughout we suppose that stations 1 through L are FCFS stations, and stations $L + 1$ through M are IS stations. All customers are required to have the same exponential service time distribution of an FCFS service center; denote $1/\mu_m$, $m = 1, \ldots, L$, for the mean service time at FCFS station m. The customer classes may have distinct service time distributions at IS stations; denote $1/\mu_{jm}$ for the mean service time of a class-j customer at IS station m. The number of servers at station m is denoted by s_m. An FCFS station can have an arbitrary finite number of servers; an IS station has an infinite number of servers, so that each customer receives its own server immediately.

We make the usual assumption of Markovian routing. For each class j, let λ_{jm}, $m = 1, \ldots, M$, be the relative visit ratio of a class-j customer to service station m. Denote $\rho_{jm} := \lambda_{jm}/\mu_m$ for $m = 1, \ldots, L$ and $\rho_{jm} := \lambda_{jm}/\mu_{jm}$ for $m = L + 1, \ldots, M$.

The state of the system is denoted by $\mathbf{n} := (n_{jm}: 1 \le j \le J, 1 \le m \le M)$, where n_{jm} denotes the number of class-j customers at service station m. The set of all possible states is given by

$$\Omega = \{\mathbf{n}: n_{j1} + \cdots + n_{jM} = N_j, \quad j = 1, \ldots, J\}.$$

Denote $\mathbf{n}_m := (n_{1m}, \ldots, n_{Jm})$ for the state of service station m and $n_m = n_{1m} + \cdots + n_{Jm}$ for the number of customers present at service station m. Let

$$f_m(n) := \begin{cases} 1 & n \le s_m \\ \prod_{i=1}^{n-s_m} \dfrac{s_m + i}{s_m} & n > s_m \end{cases}$$

and

$$f(\mathbf{n}) := \prod_{m=1}^{M} f_m(n_m) \prod_{j=1}^{J} \frac{\rho_{jm}^{n_{jm}}}{n_{jm}!}.$$

It is well known [Baskett et al. 1975; Gelenbe and Mitrani 1980; Kelly 1979] that the equilibrium probability of being in state $\mathbf{n} \in \Omega$ is given by $\pi(\mathbf{n}) = f(\mathbf{n})/g$, where

$$g := \sum_{\mathbf{n} \in \Omega} f(\mathbf{n}).$$

Many performance measures of interest, including average throughputs and the gradients of average throughputs, can be expressed as simple functions of multidimensional sums. In order to be more specific, define

$$\Omega_j(N_j) = \{(n_{j1}, \ldots, n_{jM}) : n_{j1} + \cdots + n_{jM} = N_j\},$$
$$\Omega_{(j)} = \Omega_1(N_1) \times \cdots \times \Omega_j(N_j - 1) \times \cdots \times \Omega_J(N_J).$$

Note that $\Omega_{(j)}$ is the state space associated with a queuing network which is identical to the original network except that there is one less class-j customer present. Further define

$$g^l = \sum_{\mathbf{n} \in \Omega} n_l f(\mathbf{n}),$$
$$g_j = \sum_{\mathbf{n} \in \Omega_{(j)}} f(\mathbf{n}),$$

and

$$g_j^l = \sum_{\mathbf{n} \in \Omega_{(j)}} n_l f(\mathbf{n}).$$

With this notation, the throughput of class-j customers at service station m is given by

$$TH_{jm} = \lambda_{jm} \frac{g_j}{g}. \tag{1}$$

Differentiating the above expression with respect to the service rate at station l and rearranging terms given an expression for the sensitivity of the throughput with respect to the service rate:

$$\frac{\partial TH_{jm}}{\partial \mu_l} = -\frac{\lambda_{jm}}{\mu_l} \frac{g g_j^l - g_j g^l}{g^2}. \tag{2}$$

The above product-form result continues to hold if we also permit stations to be processor sharing or last-come-first-serve preemptive resume; the Monte Carlo method to be described subsequently is therefore unchanged if some of the stations have these other service disciplines.

3. *Estimating the Normalization Constant*

We now consider estimating the normalization constant g with Monte Carlo summation. (See Rubinstein [1981] for an overview of the Monte Carlo

method.) In Section 4, we make use of (1) and (2) for estimating throughput and the sensitivity of throughput. Let $\mathbf{V}^i$, $i = 1,2,\ldots,$ be a sequence of independent and identically distributed $J \times M$ dimensional random vectors. We refer to $\mathbf{V}^i$ as the ith sample. Write V^i_{jm}, $1 \le j \le J$, $1 \le m \le M$, for the components (random variables) of the ith vector. Let Λ_j be the M-fold cartesian product of $\{0,\ldots,N_j\}$, that is, $\Lambda_j := \{0,\ldots,N_j\}^M$, and let

$$\Lambda := \Lambda_1 \times \cdots \times \Lambda_j.$$

We require V^i_{jm} to take values in $\{0,\ldots,N_j\}$ so that $\mathbf{V}^i$ takes values in Λ. Let $p(\mathbf{n}) := P(\mathbf{V}^i = \mathbf{n})$, $n \in \Lambda$, be the *importance-sampling function*, which is to be specified in order to obtain the maximum efficiency from the Monte Carlo method.

Let

$$q(\mathbf{n}) := 1(\mathbf{n} \in \Omega) f(\mathbf{n}) \tag{3}$$

and

$$Z^i := \frac{q(\mathbf{V}^i)}{p(\mathbf{V}^i)}.$$

Then

$$\bar{Z}^I := \frac{1}{I} \sum_{i=1}^{I} Z^i$$

is an unbiased estimator for g (i.e., $E[\bar{Z}^I] = g$). Moreover, the Central Limit Theorem implies that for large I

$$P\left(|\bar{Z}^I - g| \le \frac{c(\alpha)\sigma_I(Z)}{\sqrt{I}}\right) \approx 1 - \frac{\alpha}{2}, \tag{4}$$

where $c(\alpha)$ is the critical value of the standard normal distribution and $\sigma_I^2(Z)$ the sample variance of Z^i, $i = 1,\ldots,I$, that is,

$$\sigma_I^2(Z) = \frac{1}{I-1} \sum_{i=1}^{I} (Z^i - \bar{Z}^I)^2.$$

Note that for any fixed I, $\bar{Z}^I$ is an estimate of g whose accuracy can be assessed by the confidence interval $\bar{Z}^I \pm c(\alpha)\sigma_I(Z)/\sqrt{I}$ induced by (4). As the samples are being drawn, the sample variance can be calculated, and the confidence intervals can be given explicitly. Furthermore, if greater accuracy is desired, more samples can be drawn, thereby decreasing the width of the confidence interval.

The designer of a Monte Carlo scheme has great latitude in choosing the importance-sampling function $p(\mathbf{n})$, $\mathbf{n} \in \Lambda$. From (3) and (4) we see that it should satisfy the following two criteria in order for the Monte Carlo procedure to be effective:

Criterion 1: The effort required to generate $\mathbf{V}^i$ from the distribution $p(\mathbf{n})$, $\mathbf{n} \in \Lambda$, should be minimal.

Criterion 2: The variance of $Z^i = q(\mathbf{V}^i)/p(\mathbf{V}^i)$, denoted by σ^2, should be minimal.

It has been repeatedly observed in the Monte Carlo integration literature that the variance σ^2 can often be significantly reduced by choosing the appropriate sampling function $p(\mathbf{n})$, $\mathbf{n} \in \Lambda$. In particular, it is desirable to sample more frequently the points $\mathbf{n}$ at which $q(\mathbf{n})$ is "important," which is typically done by considering functions $p(\cdot)$ that are similar to $q(\cdot)$. Ideally, one would like $q(\cdot)/p(\cdot)$ to be nearly constant; however, there exists a tradeoff between this similarity and the effort required to generate a sample from $p(\cdot)$.

The standard sampling procedure is to let $p(\mathbf{n})$ correspond to a (discrete) uniform distribution over Λ, that is,

$$p(\mathbf{n}) = \prod_{j=1}^{J} \frac{1}{(N_j + 1)^M}, \qquad n \in A.$$

More generally, for each sample i, we could generate each of the components V_{jm}^i, $1 \le j \le J$, $1 \le m \le M$, independently, but use different distributions for each component:

$$p(\mathbf{n}) = \prod_{j=1}^{J} \prod_{m=1}^{M} p_{jm}(n_{jm}), \qquad \mathbf{n} \in \Lambda.$$

We shall refer to this method as the *independent sampling method*. Since V_{jm}^i can be obtained in $O(1)$ time by the alias algorithm (e.g., see Bratley et al. [1987]), a vector sample $\mathbf{V}^i$ can be obtained in $O(JM)$ time. Hence the independent sampling method satisfies fairly well Criterion 1. However, we do not recommend that this method be employed because with high probability $V_{j1}^i + \cdots + V_{jM}^i \neq N_j$ for some class j, which in turn leads to an extremely large σ^2, a violation of Criterion 2.

In the following subsection we present an importance-sampling function from which the samples can be efficiently generated and which gives rise to a σ^2 that is significantly smaller than that of the independent sampling method.

3.1 THE CLASS DECOMPOSITION APPROACH. In order to satisfy the second criterion, that is, a small variance σ^2, it is desirable to choose an importance-sampling function such that $V_{j1}^i + \cdots + V_{jM}^i = N_j$ for all classes j. This can be accomplished with sampling functions of the form

$$p(\mathbf{n}) = \prod_{j=1}^{J} p_j(n_{j1}, \ldots, n_{jM}), \tag{5}$$

where $p_j(\cdot)$ is a probability mass function defined over $\Omega_j(N_j)$. Note that a sampling function of the form (5) calls for independent sampling across classes but dependent sampling across service stations within a given class.

But how do we *efficiently* sample vectors from a given probability mass function $p_j(\cdot)$ defined over $\Omega_j(N_j)$? In order to respond to this question, fix an iteration i and a class j. Suppose, for the moment, that we know the marginal probability mass function

$$h_{j1}(n) := P\left(V_{j1}^i = n\right), \qquad 0 \le n \le N_j,$$

and for each $m = 2, \ldots, M$ the conditional mass functions

$$h_{jm}(n|l) := P\left(V_{jm}^i = n | V_{j1}^i + \cdots + V_{jm-1}^i = l\right), \qquad 0 \le n \le N_j - l, \qquad 0 \le l \le N_j.$$

Also suppose that these functions are stored in memory. Then, we sample from $p_j(\cdot)$ as follows: We first sample from $h_{j1}(\cdot)$ to obtain V_{j1}^i; if $V_{j1}^i = l$, we then sample from $h_{j2}(\cdot|l)$ to obtain V_{j2}^i; continuing this process we obtain a sample $V_{j1}^i, \ldots, V_{jM}^i$ from the probability mass function $p_j(\cdot)$. Note that sampling from $h_{jm}(\cdot|l)$ can be done in $O(1)$ time using the alias method. The computational effort is therefore $O(M)$ to sample from $p_j(\cdot)$ and $O(JM)$ to sample from $p(\cdot)$. The amount of memory required by this procedure is $O(M\sum_{j=1}^J N_j^2)$.

But how do we construct the mass function $h_{j1}(\cdot)$ and the conditional mass functions $h_{jm}(\cdot|\cdot)$ for each of the J classes? (It is important to keep in mind that this is done in the setup module and not at each iteration in the execution module.) To this end, let us further require $p_j(\cdot)$ to take the form

$$p_j(n_{j1}, \ldots, n_{jM}) = \frac{1}{G_j} \prod_{m=1}^{M} p_{jm}(n_{jm}), \qquad (n_{j1}, \ldots, n_{jM}) \in \Omega_j(N_j), \tag{6}$$

where G_j is a normalizing constant. Let $\otimes$ denote the convolution operator. The following theorem gives a recipe for constructing the desired functions. The proof, which employs (6), is straightforward and left to the reader.

THEOREM 3.1.1. *For a given class j, the mass function $h_{j1}(\cdot)$ and the conditional mass functions $h_{jm}(\cdot|\cdot)$ are given by*

$$h_{j1}(n) = p_{j1}(n)\frac{G_j^{(2)}(N_j - n)}{G_j^{(1)}(N_j)}, \qquad 0 \le n \le N_j.$$

and

$$h_{jm}(n|l) = p_{jm}(n)\frac{G_j^{(m+1)}(N_j - n - l)}{G_j^{(m)}(N_j - 1)}, \qquad 0 \le n \le N_j - l, \qquad 0 \le l \le N_j,$$

where

$$G_j^{(m)}(l) := (p_{jm} \otimes p_{jm+1} \otimes \cdots \otimes p_{jM})(l), \qquad n = 0, \ldots, N_j.$$

Moreover, $G_j = G_j^{(1)}(N_j)$.

The motivation for an importance-sampling function of the form (6) is twofold: first, it guarantees that the sample $\mathbf{V}^i$ falls in the state space, thereby limiting the variance of the estimate; second, it is not difficult to sample from when using the approach discussed above.

In summary, this approach, henceforth referred as the *decomposition approach*, involves two modules in a software implementation. In the setup module, for each class j the functions $h_{j1}(\cdot)$ and $h_{jm}(\cdot|\cdot)$, $m = 2, \ldots, M$, are calculated using the convolution functions given in Theorem 3.1.1. (The convolutions can be efficiently carried out using the Fast Fourier Transform; see [Tsang and Ross 1990].) At each iteration in the execution module, these functions are sampled to generate a vector sample from $p(\cdot)$ satisfying (5) and

(6). The vector sample always falls in Ω. Furthermore, the effort to generate a vector sample, $O(JM)$, has the same order as that to generate a vector sample using the independent sampling method. Note that J separate single-class problems are first solved (in the setup module), and then the original problem is solved by aggregating the single-class solutions (in the execution module).

Combining (3), (5), (6), the summand, which must be evaluated at every iteration of the execution module, becomes

$$\frac{q(\mathbf{n})}{p(\mathbf{n})} = \left[\prod_{j=1}^{J} G_j\right]\left[\prod_{m=1}^{L} f_m(n_m)\right]\left[\prod_{m=1}^{M}\prod_{j=1}^{J} \frac{\rho_{jm}^{n_{jm}}/n_{jm}!}{p_{jm}(n_{jm})}\right].$$

It remains to specify $p_{jm}(\cdot)$ for each j and m. Our numerical testing indicates that functions of the following form

$$p_{jm}(n) = \begin{cases} (p_{jm}\Delta_m)^n & 1 \le m \le L \\ \rho_{jm}^n/n! & L+1 \le m \le M \end{cases} \tag{7}$$

lead to relatively small values of $\sigma^2 = \text{var}[q(\mathbf{V}^i)/p(\mathbf{V}^i)]$. (Moreover, the sampling function (7) with $\Delta_m = 1$, $m = 1, \ldots, L$, gives $\sigma^2 = 0$ when there is only one class and where all FCFS stations have one server.) With this choice the summand becomes

$$\frac{q(\mathbf{n})}{p(\mathbf{n})} = \left(\prod_{j=1}^{J} G_j\right)\left(\prod_{m=1}^{L} \frac{f_m(n_m)}{\Delta_m^{n_m}\prod_{j=1}^{J} n_{jm}!}\right).$$

Intuitively, in order to make $q(\mathbf{n})/p(\mathbf{n})$ "flat," Δ_m should be made relatively large for the stations m which typically have a long queue consisting of a mix of customers from several classes. (Numerical testing seems to justify this intuition; see Table V of Section 6.)

When evaluating the above expression for $q(\mathbf{n})/p(\mathbf{n})$ we recommend that the following method be employed. In the setup module, first calculate and store in tables the following quantities:

$$t(n) := n!, \qquad 0 \le n \le N,$$

$$\xi_m(n) = \frac{f_m(n)}{\Delta_m^n}, \qquad 0 \le n \le N, \qquad 1 \le m \le L,$$

where N is the maximum number of customers that can be present at this station (thus, $N \le \sum_{j=1}^{J} N_j$). Then

$$\prod_{m=1}^{L} \frac{f_m(n_m)}{\Delta_m^{n_m}\prod_{j=1}^{J} n_{jm}!} = \prod_{m=1}^{L} \frac{\xi_m(n_m)}{\prod_{j=1}^{J} t(n_{jm})}. \tag{8}$$

Note that right-hand side of (8) can be evaluated in $O(JL)$ time. Therefore, the effort required to obtain Z^i is $O(JL)$. Note that this work bound is independent of the population sizes.

We must take care when evaluating the right-hand side of (8) at a sample $\mathbf{V}^i$ due to potential numerical overflow or underflow. This issue is addressed in Appendix A.

An alternative sampling technique is discussed in Appendix B; although promising, it has not been tested numerically.

4. *Estimating Performance Measures*

In this section we consider estimating the throughput and the throughput sensitivity as given by (1) and (2). We shall assume throughout this section that the class decomposition approach of Section 3.1 is employed.

In order to obtain estimates of performance, not only do we need to generate vectors taking values in Ω, but we must also must generate vectors taking values in $\Omega_{(j)}$, $j = 1, \ldots, J$. We recommend the following approach. Let $\mathbf{V}^i$ be a vector sample generated by the class decomposition method, which has dimension JM and takes values in Ω. We can write $\mathbf{V}^i = (\mathbf{V}_1^i, \ldots, V_J^i)$, where $\mathbf{V}_j^i$ is the vector sample (of dimension M) obtained from $p_j(n_{j1}, \ldots, n_{jM})$ defined over $\Omega_j(N_j)$. Let $p_j'(n_{j1}, \ldots, n_{jM})$ be a probability distribution over $\Omega_j(N_j - 1)$ that is defined in a manner analogous to the definition of $p_j(n_{j1}, \ldots, n_{jM})$ (i.e., (6) and (7) are used with the same values of Δ_m, $m = 1, \ldots, L$). Let $\mathbf{W}_j^i$ be a vector sample taking values in $\Omega_j(N_j - 1)$ generated from $p_j'(n_{j1}, \ldots, n_{jM})$. Our desired vector, taking value in Ω_j^-, is given by

$$\mathbf{V}_{(j)}^i = \left(\mathbf{V}_1^i, \ldots, \mathbf{W}_j^i, \ldots, \mathbf{V}_J^i\right);$$

denote $p_{(j)}(\cdot)$ for the probability distribution of $\mathbf{V}_{(j)}^i$. Note that $\mathbf{V}_{(j)}^i$ is identical to $\mathbf{V}^i$ except for the components corresponding to the jth class. One of the advantages of this method for generating the vectors $V^i, V_{(j)}^i$, $j = 1, \ldots, J$, is that the computational effort required to obtain all these vectors is not more than twice that needed to generate $\mathbf{V}^i$.

Recall the definition $Z^i = f(\mathbf{V}^i)/p(\mathbf{V}^i)$. Further define

$$Z_j^i = \frac{f\left(\mathbf{V}_{(j)}^i\right)}{p_{(j)}\left(\mathbf{V}_{(j)}^i\right)},$$

$$Y_l^i = \left(V_{1l}^i + \cdots + V_{Jl}^i\right)\frac{f(\mathbf{V}^i)}{p(\mathbf{V}^i)},$$

$$Y_{jl}^i = \left(V_{1l}^i + \cdots + W_{jl}^i + \cdots + V_{Jl}^i\right)\frac{f\left(\mathbf{V}_{(j)}^i\right)}{p_{(j)}\left(\mathbf{V}_{(j)}^i\right)}.$$

Then an estimator for TH_{jm} is given by

$$\Phi_{jm}^I = \lambda_{jm}\frac{\Sigma_{i=1}^I Z_j^i}{\Sigma_{i=1}^I Z^i},$$

and an estimator for $\partial TH_{jm}/\partial \mu_l$ is given by

$$S_{jml}^I = -\frac{\lambda_{jm}}{\mu_l}\frac{\left(\Sigma_{i=1}^I Z^i\right)\left(\Sigma_{i=1}^I Y_{jl}^i\right) - \left(\Sigma_{i=1}^I Z_j^i\right)\left(\Sigma_{i=1}^I Y_l^i\right)}{\left(\Sigma_{i=1}^I Z^i\right)^2}.$$

Although these estimates converge to the desired values as $I \to \infty$, both estimators have the undesirable property of being biased. However, the bias diminishes as n becomes large (see the discussion in Ross and Wang [1992]).

Confidence intervals for Φ_{jm}^I and S_{jml}^I can be easily constructed by invoking the Central Limit Theorem for nonlinear functions of sums of i.i.d. random variables (e.g., see Theorem 1 of Glynn and Iglehart [1988]).

5. *An Alternative Approach: Monte Carlo Integration*

Throughout this section, we assume that each First-Come, First-Serve (FCFS) station has exactly one server. With this restriction, the normalization constant of the multiclass queuing network described in Section 2 can be expressed as a multidimensional integral [Ramakrishnan and Mitra 1982]:

$$g = \frac{1}{\prod_{j=1}^{J} N_j!} \int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u}) \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u})^{N_j}\, d\mathbf{u}, \tag{9}$$

where

$$\begin{aligned}
\mathbf{u} &= (u_1, \ldots, u_L)' \\
\mathbf{1} &= (1, \ldots, 1)' \\
\rho_{j0} &= \sum_{m=L+1}^{M} \rho_{jm}, \qquad 1 \le j \le J \\
\boldsymbol{\rho}_j &= (\rho_{j1}, \ldots, \rho_{jL})' \\
\mathbf{Q}^+ &= \{\mathbf{u} \in R^L : u_l \ge 0, \quad l = 1, \ldots, L\}.
\end{aligned}$$

Moreover the quantities g^l, g_j, and g_j^l defined in Section 2 can also be expressed as multidimensional integrals. For example,

$$g_j = \frac{N_j}{\prod_{k=1}^{J} N_k!} \int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u}) \frac{\prod_{k=1}^{J} (\rho_{k0} + \boldsymbol{\rho}_k'\mathbf{u})^{N_k}}{\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u}}\, d\mathbf{u},$$

$$g^l = \begin{cases} \dfrac{1}{\prod_{k=1}^{J} N_k!} \displaystyle\int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u})(u_l - 1) \prod_{k=1}^{J} (\rho_{k0} + \boldsymbol{\rho}_k'\mathbf{u})^{N_k}\, d\mathbf{u} & 1 \le l \le L \\ \displaystyle\sum_{j=1}^{J} \frac{N_j \rho_{jl}}{\prod_{k=1}^{J} N_k!} \int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u}) \frac{\prod_{k=1}^{J} (\rho_{k0} + \boldsymbol{\rho}_k'\mathbf{u})^{N_k}}{\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u}}\, d\mathbf{u} & L+1 \le l \le M. \end{cases}$$

In this section, we show how Monte Carlo integration can be applied to calculate the normalization constants and performance measures associated with the product-form queuing network. To this end, let $\mathbf{V}^i = (V_1^i, \ldots, V_L^i)'$, $i = 1, 2, \ldots$, be a sequence of independent and identically distributed L-dimensional random vectors with probability density function $p(\cdot)$ defined over $\mathbf{Q}^+$. As in the case of Monte Carlo summation, there is great latitude in the choice of importance-sampling density $p(\cdot)$. Let

$$Z^i := \frac{\exp(-\mathbf{1}'\mathbf{V}^i) \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{V}^i)^{N_j}}{p(\mathbf{V}^i)} \frac{1}{\prod_{j=1}^{J} N_j!}$$

and define $\bar{Z}^I$ as the average of $Z^1, \ldots, Z^I$ (as in Section 3). Then $\bar{Z}^I$ is an unbiased estimator of g. Similarly, if we define

$$Z_j^i := \frac{\exp(-\mathbf{1}'\mathbf{V}^i)\prod_{k=1}^{J}\left(\rho_{k0} + \boldsymbol{\rho}_k'\mathbf{V}^i\right)^{N_k}}{\left(\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{V}^i\right)p(\mathbf{V}^i)} \frac{N_j}{\prod_{j=1}^{J} N_j!},$$

then

$$\Phi_{jm}^I := \lambda_{jm}\frac{\sum_{i=1}^{I} Z_j^i}{\sum_{i=1}^{I} Z^i},$$

is a consistent estimator for TH_{jm}.

We shall refer to the approach outlined above for estimating normalization constants and performance measures as the *integration approach.* We make the following observations:

—The dimension of the sampling vector $\mathbf{V}^i$ for the integration approach is L, the number of FCFS stations. If each of the components of $\mathbf{V}^i$ is obtained independently, i.e., if

$$p(u_1, \ldots, u_L) = p_1(u_1) \times \cdots \times p_L(u_L), \tag{10}$$

then the effort required to generate $\mathbf{V}^i$ for the integration approach is $O(L)$. Moreover, if simple distributions are used for the marginal distributions $p_m(\cdot)$, $m = 1, \ldots, L$, such as the exponential distribution (see Section 5.1), then the amount of memory needed to obtain $\mathbf{V}^i$ is also $O(L)$. These work and memory bounds compare favorably with those of the class decomposition approach, where the computational effort and memory requirements to generate $\mathbf{V}^i$ are $O(LJ)$ and $O(L\sum_{j=1}^{J} N_j^2)$, respectively.

—For the integration approach with exponential sampling, the computational effort required to evaluate the integrand (i.e., to obtain Z^i from $\mathbf{V}^i$) requires $O(L\sum_{j=1}^{J}\log N_j)$ operations when successive squaring is used to evaluate the exponent involving N_j. Note that the computational effort grows (slowly) with the population size.

—To estimate the throughputs, TH_{jm}, $j = 1, \ldots, J$, $m = 1, \ldots, M$, the same sequence of random vectors $\mathbf{V}^i$, $i = 1, 2, \ldots$, can be used to evaluate the numerator and denominator for Φ_{jm}^I. In contrast, the class decomposition method needs a modified sequence for the numerator for each class (see Section 4).

—The summation approach has a wider range of applicability since it is not limited to product-form queuing networks with integral representations.

—The integral representation is valid even if there are no IS service centers; however, the asymptotic expansions [Ramakrishnan and Mitra 1982] that form the basis of PANACEA require that there be at least one IS station and that the normal-usage conditions be satisfied.

5.1 Exponential Sampling Functions. Throughout the remainder of this section, we restrict our attention to importance-sampling densities of the form (10). We further suppose that each component $p_m(\cdot)$ has an exponential distribution, that is, we suppose that

$$p_m(u) = \gamma_m \exp(-\gamma_m u), \qquad m = 1, \ldots, L, \tag{11}$$

where $\gamma' := (\gamma_1, \ldots, \gamma_L)$ is to be specified. Such a probability density function is easy to sample from; moreover, empirical testing seems to indicate that it leads to relatively narrow confidence intervals. Note that we now have

$$Z^i := \left(\prod_{m=1}^{L} \gamma_m\right)^{-1} \exp(-(\mathbf{1} - \boldsymbol{\gamma})'\mathbf{V}^i) \prod_{j=1}^{J} \frac{\left(\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{V}^i\right)^{N_j}}{N_j!}.$$

It is of interest to choose $\boldsymbol{\gamma}$ so that the confidence intervals associated with the various estimators are as small as possible. To this end, we consider the problem of minimizing $\text{var}(\bar{Z}_n)$ with respect to $\boldsymbol{\gamma} \in \mathbf{Q}^+$. It is easily seen that this problem is equivalent to minimizing $h(\boldsymbol{\gamma})$ with respect to $\boldsymbol{\gamma}$, where

$$h(\boldsymbol{\gamma}) := \left(\prod_{m=1}^{L} \gamma_m\right)^{-1} \int_{\mathbf{Q}^+} \exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u}) \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u})^{2N_j}\, d\mathbf{u}.$$

Let $\Gamma := \{\boldsymbol{\gamma} \in \mathbf{Q}^+ : 0 < \gamma_m < 2,\ m = 1, \ldots, L\}$. Note that $h(\boldsymbol{\gamma}) = \infty$ for all $\boldsymbol{\gamma} \in \mathbf{Q}^+ - \Gamma$ and that $0 < h(\boldsymbol{\gamma}) < \infty$ for all $\boldsymbol{\gamma} \in \Gamma$. Let $g(\mathbf{u}) = \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u})^{2N_j}$.

THEOREM 5.1.1. *$h(\boldsymbol{\gamma})$ is strictly convex over $0 \le \gamma_m \le 2$; consequently, $h(\boldsymbol{\gamma})$ has a unique minimum $\boldsymbol{\gamma}^* \in \Gamma$.*

PROOF. Note that

$$h(\boldsymbol{\gamma}) = \int_{Q^+} \frac{\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})}{\prod_{l=1}^{L} \gamma_l} g(\mathbf{u})\, d\mathbf{u}.$$

It is sufficient to show that $\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})/\prod_{l=1}^{L} \gamma_l$ is strictly convex with respect to $\boldsymbol{\gamma}$ since $g(\mathbf{u})$ is always positive. It is straightforward to obtain the following.

$$\frac{\partial^2 \{\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})/\prod_{l=1}^{L} \gamma_l\}}{\partial \gamma_m\, \partial \gamma_j}$$

$$= \begin{cases} \dfrac{\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})}{\prod_{l=1}^{L} \gamma_l} \left(u_m - \dfrac{1}{\gamma_m}\right)\left(u_j - \dfrac{1}{\gamma_j}\right) & m \ne j \\ \dfrac{\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})}{\prod_{l=1}^{L} \gamma_l} \left[\left(u_m - \dfrac{1}{\gamma_m}\right)^2 + \dfrac{1}{\gamma_m^2}\right] & m = j \end{cases}.$$

It is obvious that the Hessian of $\exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u})/\prod_{l=1}^{L} \gamma_l$ is positive definite; hence, $h(\cdot)$ is strictly convex. □

Now consider a closed multiclass queuing network with one IS station (indexed by 0), L FCFS stations (indexed from 1 through L), and J classes. Suppose that the population size of the jth class is $2N_j$. Also suppose that the relative traffic intensities, denoted by $\hat{\rho}_{jm}$, $j = 1, \ldots, J$, $m = 0, \ldots, L$, are given by $\hat{\rho}_{j0} = \rho_{j0}$ for the IS station and $\hat{\rho}_{jm} = \rho_{jm}/(2 - \gamma_m)$ for the FCFS stations. We shall refer to this new network as network $\Theta(\boldsymbol{\gamma})$. Denote $x_m(\boldsymbol{\gamma})$ for the expected number of customers present at station m in network $\Theta(\boldsymbol{\gamma})$. Also let $f_m(\boldsymbol{\gamma}) := 2/(2 + x_m(\boldsymbol{\gamma}))$, and let $\mathbf{f}(\boldsymbol{\gamma})$ be the corresponding vector.

THEOREM 5.1.2. *The vector* $\boldsymbol{\gamma}^*$ *that minimizes* $h(\boldsymbol{\gamma})$ *is given by the unique solution to the following fixed-point equation*:

$$\boldsymbol{\gamma} = \mathbf{f}(\boldsymbol{\gamma}), \qquad \boldsymbol{\gamma} \in \Gamma.$$

PROOF. By Theorem 5.1.1, the solution to $\nabla h(\boldsymbol{\gamma}) = 0$ is the unique minimum of $h(\cdot)$. Setting $\partial h / \partial \gamma_m = 0$, we obtain

$$\int_{\mathbf{Q}^+} \exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u}) \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u})^{2N_j} \, d\mathbf{u}$$

$$= \gamma_m \int_{\mathbf{Q}^+} \exp(-(2\mathbf{1} - \boldsymbol{\gamma})'\mathbf{u}) u_m \prod_{j=1}^{J} (\rho_{j0} + \boldsymbol{\rho}_j'\mathbf{u})^{2N_j} \, d\mathbf{u}.$$

A change of variables $(2 - \gamma_l)u_l \to u_l$, $l = 1, \ldots, L$, in both sides of the above equation leads to

$$\int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u}) \prod_{j=1}^{J} (\rho_{j0} + \hat{\boldsymbol{\rho}}_j'\mathbf{u})^{2N_j} \, d\mathbf{u}$$

$$= \frac{\gamma_m}{2 - \gamma_m} \int_{\mathbf{Q}^+} \exp(-\mathbf{1}'\mathbf{u}) u_m \prod_{j=1}^{J} (\rho_{j0} + \hat{\boldsymbol{\rho}}_j'\mathbf{u})^{2N_j} \, d\mathbf{u},$$

where $\hat{\boldsymbol{\rho}}_j := (\hat{\boldsymbol{\rho}}_{j1}, \ldots, \hat{\boldsymbol{\rho}}_{jL})$. Dividing both sides of the above equation by $\prod_{j=1}^{J}(2N_j)!$ and converting the integral expressions into the corresponding summation expressions yields

$$\sum_{\mathbf{n} \in \hat{\Omega}} \prod_{l=1}^{L} \left(n_l! \prod_{j=1}^{J} \frac{\hat{\boldsymbol{\rho}}_{jl}^{n_{jl}}}{n_{jl}!} \right) \left(\prod_{j=1}^{J} \frac{\rho_{j0}^{n_{j0}}}{n_{j0}!} \right)$$

$$= \frac{\gamma_m}{2 - \gamma_m} \sum_{\mathbf{n} \in \hat{\Omega}} (n_m + 1) \prod_{l=1}^{L} \left(n_l! \prod_{j=1}^{J} \frac{\hat{\boldsymbol{\rho}}_{jl}^{n_{jl}}}{n_{jl}!} \right) \left(\prod_{j=1}^{J} \frac{\rho_{j0}^{n_{j0}}}{n_{j0}!} \right),$$

where $\hat{\Omega}$ is the state space for $\Theta(\boldsymbol{\gamma})$. Noting that the LHS of the above equation is the normalization constant for $\Theta(\boldsymbol{\gamma})$, we obtain the desired result. □

A function g: $\mathbf{Q}^+ \to \mathbf{R}$ is *symmetric* for every vector $\mathbf{u} = (u_1, u_2, \ldots, u_L) \in \mathbf{Q}^+$ and permutation $\mathbf{u}' = (u_{i_1}, u_{i_2}, \ldots, u_{i_L})$ of $\mathbf{u}$, we have $g(\mathbf{u}) = g(\mathbf{u}')$. Let

$$T_m = \sum_{j:\, \rho_j > 0} N_j$$

be the maximum number of customers that can be present at station m.

COROLLARY 5.1.3. *Let* $\boldsymbol{\gamma}^*$ *be the unique minimum of* $h(\boldsymbol{\gamma})$, $\boldsymbol{\gamma} \in \Gamma$. *Then*

$$\frac{1}{1 + T_m} < \gamma_m^* < 1 \qquad m = 1, \ldots, L.$$

If all stations are FCFS, we also have

$$\sum_{m=1}^{M} \frac{1-\gamma_m^*}{\gamma_m^*} = \sum_{j=1}^{J} N_j. \tag{12}$$

Furthermore, if all stations are FCFS, and $g(\mathbf{u}) = \prod_{j=1}^{J} (\boldsymbol{\rho}_j' \mathbf{u})^{2N_j}$ *is a symmetric function of* $\mathbf{u}$, *then*

$$\gamma_m^* = \frac{M}{M + \sum_{j=1}^{J} N_j}, \qquad m = 1, \ldots, M. \tag{13}$$

PROOF. From Theorem 5.1.2, we have

$$x_m(\gamma_m^*) = \frac{2(1-\gamma_m^*)}{\gamma_m^*}. \tag{14}$$

The first statement follows directly from (14) and the fact that

$$0 < x_m(\boldsymbol{\gamma}) < 2T_m$$

for all $\boldsymbol{\gamma} \in \Gamma$.

The second statement follows from (14) and the fact that

$$\sum_{m=1}^{M} x_m^*(\boldsymbol{\gamma}) = 2T_m$$

for all $\boldsymbol{\gamma} \in \Gamma$ when all stations are FCFS.

For the third statement, we first show that if $g(\mathbf{u})$ is symmetric then so is $h(\boldsymbol{\gamma})$. To this end, let $(i_1, i_2, \ldots, i_M)$ be an arbitrary permutation of $(1, 2, \ldots, M)$. Let σ be the permutation operator that maps $(x_1, x_2, \ldots, x_M)$ to $(x_{i_1}, x_{i_2}, \ldots, x_{i_M})$ for any $\mathbf{x} = (x_1, \ldots, x_M) \in \mathbf{Q}^+$, and let σ^{-1} be the corresponding inverse operator. For any $\boldsymbol{\gamma}$ and $\mathbf{u}$, it is easily verified that $\sigma(\boldsymbol{\gamma})'\mathbf{u} = \boldsymbol{\gamma}'\sigma^{-1}(\mathbf{u})$. Thus, if $g(\mathbf{u})$ is symmetric, then

$$\begin{aligned} h(\sigma(\boldsymbol{\gamma})) &= \frac{1}{\prod_{m=1}^{M} \sigma(\boldsymbol{\gamma})_m} \int_{\mathbf{Q}^+} \exp(-[2\mathbf{1} - \sigma(\boldsymbol{\gamma})]'\mathbf{u}) g(\mathbf{u})\, d\mathbf{u} \\ &= \frac{1}{\prod_{m=1}^{M} \gamma_m} \int_{\mathbf{Q}^+} \exp(-(2\mathbf{1} - \boldsymbol{\gamma})' p^{-1}(\mathbf{u})) g[\sigma^{-1}(\mathbf{u})]\, d\mathbf{u} \\ &= h(\boldsymbol{\gamma}), \end{aligned}$$

implying that $h(\boldsymbol{\gamma})$ is symmetric.

Since $\boldsymbol{\gamma}^*$ is the unique minimum of $h(\boldsymbol{\gamma})$ and since $h(\boldsymbol{\gamma})$ is symmetric, it follows that $\gamma_m^* = \gamma_1^*$, for $m = 1, \ldots, M$. Combining this with (12) gives (13). □

We see from Corollary 5.1.3 that the optimal importance-sampling parameters can be given explicitly when $g(\mathbf{u})$ is symmetric. As an example, consider a network consisting of M FCFS stations, J classes, with $N_j = N$, for $j = 1, \ldots, J$. Furthermore, suppose for each subset of K stations, there is one class that visits the K stations cyclically; hence $J = M!/K!(M-K)!$. For this network,

we can normalize the ρ_{jk}'s so that they take on values in $\{0, 1\}$. It is straightforward to verify that $g(\mathbf{u})$ is symmetric. Hence, by Corollary 1,

$$\gamma_m^* = \frac{M}{M + JN}, \qquad m = 1, \ldots, M.$$

It is easy to see that for a network of FCFS stations, the optimal importance-sampling parameters are given by (13) whenever $\gamma_m^* = \gamma_1^*$ for all $m = 1, \ldots, M$. An interesting research problem is to find a simple necessary and sufficient condition such that $\gamma_m^* = \gamma_1^*$ for $m = 1, \ldots, M$.

Corollary 1 does not specify an explicit expression for the sampling parameters for "asymmetric" networks. But it follows from Theorem 5.1.2 that if efficient methods for obtaining $x_m(\boldsymbol{\gamma})$ were available, we could then attempt to find a $\boldsymbol{\gamma}$ which satisfies the fixed-point equation in Theorem 5.1.2 by using successive approximations. However, the network $\Theta(\boldsymbol{\gamma})$ is not easier to solve than the original network. One possible approach is to approximate $x_m(\boldsymbol{\gamma})$, using one of the rapid approximation techniques found in Bard [1981], Baynat et al. [1992], Chandy and Neuse [1982] and Schweitzer [1979] and use this approximation within a successive-approximations procedure.

It turns out, however, that the following heuristic for choosing sampling parameters $\hat{\boldsymbol{\gamma}}$ leads to remarkably good results. First, obtain an approximation for util_m, the utilization at station m in the *original* network, for all FCFS stations. Second, set $1 - \hat{\gamma}_m$ equal to this approximation of util_m for all FCFS stations. Indeed, our empirical testing (see Section 6) indicates that when $\hat{\gamma}_m = 1 - \text{util}_m$ for all FCFS stations, a near-optimal reduction in variance is obtained when estimating the normalization constant; moreover, these sampling parameters also work remarkably well for estimating throughputs.

But how do we approximate the utilization at the FCFS stations? First suppose that the "normal-usage conditions" are satisfied [Ramakrishnan and Mitra 1982]; that is, suppose

$$\rho_{j0} > 0 \qquad \text{for all } j = 1, \ldots, J \tag{15}$$

$$\sum_{j=1}^{J} N_j \frac{\rho_{jm}}{\rho_{j0}} < 1 \qquad \text{for all } m = 1, \ldots, L. \tag{16}$$

We then suggest that util_m be approximated by the left-hand side of (16); indeed, it is shown in Ramakrishnan and Mitra [1982] that the util_m approaches this quantity in a natural limiting regime. In this case, our heuristic gives the following explicit sampling parameters:

$$\hat{\gamma}_m = 1 - \sum_{j=1}^{J} N_j \frac{\rho_{jm}}{\rho_{j0}}, \qquad m = 1, \ldots, L. \tag{17}$$

We show in Ross and Wang [1993] that independent exponential sampling with $\boldsymbol{\gamma} = \hat{\boldsymbol{\gamma}}$ minimizes variance in the asymptotic regime considered in McKenna et al. [1981]; in particular, $\boldsymbol{\gamma}^* - \hat{\boldsymbol{\gamma}}$ converges to zero in the asymptotic regime. Empirical justification of this heuristic is given in the subsequent section.

Now suppose that (15) or (16) is not satisfied.This would occur, for example, if the network does not have any IS stations. In this case, we recommend approximating util_m, $m = 1, \ldots, L$, by one of the rapid approximation tech-

TABLE I. PANACEA VS. MONTE CARLO INTEGRATION FOR A SYMMETRIC NETWORK

Util. at FCFS	PANACEA	Integration	Integration with Importance Sampling[1]	Integration with Optimal Imp. Sampling[2]	Exact Value
0.185	(0.492, 0.493)	(0.492, 0.493)	(0.492, 0.493)	(0.492, 0.493)	0.493
0.444	(1.156, 1.193)	(1.172, 1.193)	(1.183, 1.186)	(1.183, 1.186)	1.184
0.565	(0.970, 1.650)	(1.482, 1.534)	(1.508, 1.513)	(1.509, 1.513)	1.511

[1] Importance-sampling parameters are: 0.813, 0.531, and 0.381.
[2] Importance-sampling parameters are: 0.815, 0.556, and 0.435

niques cited above (or by the Monte Carlo techniques with a relatively small number of iterations).

6. *Computational Results*

All computational tests discussed below are performed with a Sun SparcStation2. The PANACEA results are obtained from the most recent version of the PANACEA package, version 2.2.1, loaned to us from AT & T Bell Laboratories; this version requires that there be at least one IS station and that all FCFS stations have a single server. The Monte Carlo results are obtained from Monte Queue, a Fortran package that we have developed and that is available in the public domain at the ftp site systems.seas.upenn.edu with anonymous as a login name. Exact values of utilizations and throughputs presented in the various tables are obtained from long runs of the Monte Carlo methods. The results for optimal importance sampling presented in the various tables are obtained by varying the importance-sampling parameters using trial and error until the width of the confidence intervals is minimized.

Let Network I be defined as follows. There are 7 single-server FCFS stations and two IS stations. For each set of 5 FCFS stations, there is one class that visits each of the 5 FCFS stations cyclically and then visits one of the two IS stations with equal probability. Thus there is a total of 21 (i.e., 7 choose 5) classes.

It is unlikely that any of the existing combinatorial algorithms can handle such a network with 5 or more customers in each class. We begin by comparing Monte Carlo integration with PANACEA.

Our first set of computational results makes use of a symmetric version of Network I. In particular, we set the population sizes equal to 5 for all classes, the service rates to 40 for all of the FCFS stations, and the service rates for the two IS stations equal and to either 0.1, 0.25, 0.33. These three services rates lead to three different levels of the utilization of FCFS stations. Note that in this symmetric network, the throughput TH_{jm} will be the same for all classes at all FCFS stations. Table I presents results for estimating TH_{jm} by PANACEA, "Monte Carlo Integration," "Monte Carlo Integration with Importance Sampling," and "Monte Carlo Integration with Optimal Importance Sampling." The column "Integration" corresponds to the case of importance-sampling parameters set to $\gamma_m = 1$ for all FCFS stations. The column "Integration with Importance Sampling" corresponds to the case $\gamma_m = \hat{\gamma}_m$ for all FCFS stations, where $\hat{\gamma}_m$ is our heuristic defined by (17). The intervals in the PANACEA

column are the lower and upper bounds specified by the associated asymptotic expansions. For the Monte Carlo methods, 95% confidence intervals are given.

The CPU time consumed by PANACEA for a given service rate at the IS stations is 5.7 seconds. The number of iterations, 1900, for the Monte Carlo methods is chosen so that they also consume 5.7 seconds. In the case of low utilization at the FCFS stations, both PANACEA and the Monte Carlo methods give rise to very narrow "intervals," with PANACEA giving the narrowest interval. For moderate utilization at the FCFS stations, all of the methods give narrow intervals, although less narrow than those for the case of low utilization. Note that the intervals associated with the Monte Carlo methods are now narrower than the one associated with PANACEA; in particular, when the importance-sampling heuristic is used, the interval width given by Monte Carlo integration is about 8% of that given by PANACEA. For the third case of higher utilization at the FCFS stations, the performance of Monte Carlo integration is even more impressive: PANACEA gives a very wide interval, whereas Monte Carlo integration continues to give narrow intervals. When the importance-sampling heuristic is used, the width of the confidence intervals is less than 1% of that obtained from PANACEA.

Our second set of computational tests explores the performance of the Monte Carlo integration for a network with large population sizes. In particular, we consider Network I with N_j now set to 100 for each of the 21 classes. We also increase the service rates at the FCFS to 200 so that the utilizations are not too close to one. Since the computational effort for Monte Carlo integration grows with population size, albeit very slowly, we decrease the number of iterations to 1600 so that the Monte Carlo methods take the same amount of CPU time as PANACEA, 5.6 seconds. In Table II, we present the results of three different utilization levels corresponding to the service rates of two IS stations being equal and set to 0.04, 0.07, and 0.11. In the low-utilization case, we again see that PANACEA and the Monte Carlo integration techniques all give excellent results. In the moderate-utilization case, PANACEA gives a reasonably good estimate even though its interval significantly widens. For these same traffic conditions, Monte Carlo integration with the importance-sampling parameters set to 1 gives a poor result: its confidence intervals do not even cover the exact value. (We will explain this below.) Monte Carlo integration with the importance-sampling heuristic, however, gives excellent results with narrow confidence intervals. The results for the higher utilization are analogous to those for the moderate utilization except that PANACEA now gives a very wide interval.

How can it be that the confidence intervals for Monte Carlo integration with the sampling parameters set to 1 do not contain the exact value for the moderate- and high-utilization cases of Table II? We observed that the sample variance of the estimate of the normalization constant in these situations is extremely large. This along with the bizarre confidence intervals seems to indicate that the number of iterations, I, chosen for this example is not sufficient to invoke the Central Limit Theorem for nonlinear functionals [Glynn and Iglehart 1988]; Jackknifing [Miller 1974] can be used to eliminate the leading term in the bias, possibly leading to improved performance. This odd behavior does not seem to occur when importance sampling is employed.

It therefore appears from Tables I and II that Monte Carlo integration with importance sampling is more robust than PANACEA with respect to the traffic

TABLE II. PANACEA VS. INTEGRATION FOR A SYMMETRIC NETWORK WITH LARGE POPULATION

Util. at FCFS	PANACEA	Integration	Integration with Imp. Sampling[1]	Integration with Optimal Imp. Sampling[2]	Exact Value
0.523	(6.974, 6.975)	(6.969, 6.979)	(6.974, 6.975)	(6.974, 6.975)	6.975
0.815	(10.749, 10.870)	(10.900, 10.931)	(10.839, 10.845)	(10.840, 10.846)	10.843
0.873	(2.491, 11.900)	(11.780, 11.828)	(11.626, 11.637)	(11.628, 11.638)	11.636

[1]Importance-sampling parameters are: 0.475, 0.175, and 0.108.
[2]Importance-sampling parameters are: 0.480, 0.185, and 0.128.

TABLE III. MONTE CARLO INTEGRATION FOR AN ASYMMETRIC NETWORK

Station Number	Util.	Integration	Integration with Importance Sampling[1]	Exact Value
3	0.473	(1.622, 1.661)	(1.628, 1.634)	1.631
4	0.452	(1.248, 1.277)	(1.252, 1.257)	1.254
5	0.605	(1.148, 1.175)	(1.152, 1.156)	1.154
6	0.635	(1.192, 1.221)	(1.197, 1.201)	1.199
7	0.687	(1.348, 1.380)	(1.353, 1.358)	1.355

[1]Importance sampling parameters are: 0.49, 0.51, 0.34, 0.31, and 0.25.

conditions for Network I (for both small and large populations). Note that the optimal importance-sampling parameter, γ_m^*, is very close to $1 - \text{util}_m$ for each of the two population sizes and three traffic conditions. Moreover, our heuristic (17) for choosing the importance-sampling vector $\hat{\gamma}$ results in a confidence interval width that is very close to the optimal for each case.

Our third set of computational results involves an asymmetric version of Network I. The number of classes and stations are the same as in Network I; the population for each class is 5; but the routing and service rates are now asymmetric. In particular, we see that service rates at the 7 FCFS stations to 60, 60, 40, 40, 30, 30, 30 and the service rates at the two IS stations to 0.4 and 1.2. Each class visits 5 of the 7 FCFS stations and then one of the two IS stations with *different class-dependent* probabilities. Table III presents the results obtained by Monte Carlo integration for the throughput of class 1 at each of the FCFS stations that it visits. The number of iterations is 2000; it takes about 5.7 seconds to finish one run. It can be seen that Monte Carlo integration gives good results; the ratio of the width of the confidence interval over the estimate always being less than 3%. When the importance-sampling heuristic is implemented, the ratio is never more than 0.4%.

Our fourth set of computational results compares Monte Carlo integration with Monte Carlo summation for a network with a larger number of stations and classes. In this network, there are 10 FCFS stations, 1 IS station, and 45 classes. Each class visits 8 of the 10 FCFS stations, and then the IS station. The population size of each class is $N_j = 5$. To make the CPU times comparable. we run Monte Carlo integration with 11,300 iterations and Monte Carlo summation with 2000 iterations. The CPU time consumed is about 76 seconds in each case. Table IV illustrates the ability of the two Monte Carlo methods to handle this kind of large network. The column "summation" corresponds to

TABLE IV. MONTE CARLO INTEGRATION VS. SUMMATION FOR A LARGE NETWORK

Util. Level	Integration	Summation	Exact Value
Low	(0.246, 0.246)	(0.244, 0.251)	0.246
Medium	(0.473, 0.480)	(0.463, 0.490)	0.478
High	(0.675, 0.694)	(0.517, 0.784)	0.679

the case of importance-sampling parameters $\Delta_m = 1$ for all FCFS stations. Three levels of utilization are obtained with three different service rates for the IS station. For each level of utilization, although the exact throughput for each class is the same because the network is symmetric, the confidence intervals are different due to the random nature of the Monte Carlo procedure. Here we choose, among all the confidence intervals for throughputs obtained from Monte Carlo integration, the one with the largest width and the corresponding one from Monte Carlo summation. As one can see, Monte Carlo integration gives good results for all three levels since the width of the confidence interval is less than 3% of the exact value, while Monte Carlo summation gives reasonable results for the first two levels but gives poor estimates for the case of high utilization. The width of the confidence intervals can again be reduced if importance sampling is employed.

From Table IV and other computational results (not presented here), we observe that Monte Carlo integration almost always performs better than Monte Carlo summation when each FCFS station has a single server and normal traffic conditions are satisfied.

In Table V, we compare Monte Carlo integration and summation for a network with no IS stations. (Thus, the normal traffic conditions are violated, and PANACEA is inapplicable to this network.) The network has 21 FCFS stations, numbered from 1 to 21, 4 classes with each having a population of 5. Each class visits 5 of the first 20 stations and also visits the 21st station. The service rates at stations 11 and 21 are 0.1 while the rest of the stations have service rate 45.0, which results in most of the customers being clustered at stations 11 and 21. It takes about 69 seconds to run 10,000 iterations of Monte Carlo summation or 53,000 iterations of Monte Carlo integration. Since for each class the throughputs at all the stations are the same, we present one throughput value for each class in Table V. As one can see, summation gives reasonable results for all classes; with importance sampling the confidence intervals are significantly narrower. The results given by integration with $\gamma_m = 1$ for all FCFS stations are not meaningful since they are far away from the exact value. Monte Carlo integration with importance sampling, however, gives excellent performance.

We therefore conclude from the above computational experiments that Monte Carlo integration with importance sampling typically gives excellent results for a wide range of network sizes and traffic conditions. When the normal traffic conditions are satisfied, the heuristic (17) for choosing the sampling parameters consistently gives good performance. If normal traffic is violated, but a good approximation of util_m is available for all the FCFS stations, then setting $1 - \gamma_m$ to the approximate utilization gives good results. However, when such an approximation is not available, care has to be taken

TABLE V. MONTE CARLO INTEGRATION VS. SUMMATION FOR A NETWORK WITH ONLY FCFS NODES[1]

Class Number	Summation	Summation with Optimal Imp. Sampling[2]	Integration	Integration with Optimal Imp. Sampling[3]	Exact Value
1	(1.96,2.45)	(2.37,2.47)	(3.23,3.49)	(2.37,2.39)	2.38
2	(1.99,2.47)	(2.36,2.45)	(3.22,3.48)	(2.37,2.39)	2.38
3	(2.76,2.86)	(2.81,2.87)	(3.57,4.22)	(2.85,2.88)	2.86
4	(2.06,2.53)	(2.32,2.41)	(3.22,3.48)	(2.37,2.39)	2.38

[1]All numbers in this table should be multiplied by 10^{-2}.
[2]The importance-sampling parameter at node 11 is 3.5; that at node 21 is 7.0; the rest are 1.0.
[3]The importance-sampling parameter at node 11 is 0.3; that at node 21 is 0.01; the rest are 1.0. The utilizations of node 11 is 0.71; that of node 21 is 1.00.

TABLE VI. MONTE CARLO SUMMATION FOR A NETWORK WITH MULTIPLE-SERVER FCFS STATIONS

IS Station Service Rate	Monte Carlo Summation	Monte Carlo Sum With Imp. Sampling[1]	Exact Value
0.25	(1.101, 1.123)	(1.101, 1.123)	1.106
0.33	(1.373, 1.439)	(1.378, 1.437)	1.399
0.4	(1.528, 1.831)	(1.609, 1.735)	1.621

[1]Importance-sampling parameters corresponding to the three levels of IS service rates are: 1.00, 1.05, and 1.10.

since Monte Carlo integration can lead to incorrect confidence intervals. In this case we recommend Monte Carlo summation.

The next set of computational results explores Monte Carlo summation for networks with *multiple-server FCFS stations*. Consider again Network I with 5 customers in each class. Now let there be 4 servers at each FCFS station, each with service rate 10. Table VI gives the computational results for 2000 iterations for three levels of utilization; the CPU time for one run is 25 seconds. Accuracy is again good, particularly at the lower loads, but not as good as for Monte Carlo integration with single-server FCFS stations.

In Table VII we present some results for throughput sensitivities for the symmetric version of Network I with 5 customers in each class. For both Monte Carlo integration and summation, 200,000 iterations were employed. To be brief, we only present the confidence intervals for one class-node combination, $\partial TH_{13}/\partial\mu_l$ for $l = 1,\ldots,9$. We intend to explore in more depth the issue of sensitivity estimation (combined with importance sampling) in a future paper. For the time being we observe that (i) integration seems to give better performance than summation and (ii) a greater number of iterations is required in order to obtain narrow confidence intervals.

7. *Concluding Remarks*

Monte Carlo summation and integration are robust techniques that can solve a wide range of queuing networks, including many networks which are difficult, if not impossible, to solve by the combinatorial and asymptotic methods in the existing literature. We have discussed how the techniques can be applied to closed multiclass queuing networks consisting of multiple-server FCFS stations

TABLE VII. THROUGHPUT SENSITIVITIES FOR A SYMMETRIC NETWORK[1]

Class Number	Integration	Summation
1	(-0.09, -0.03)	(-0.17, 0.47)
2	(-0.09, -0.04)	(-0.57, 0.39)
3	(0.35, 0.47)	(-2.67, 1.60)
4	(0.35, 0.47)	(0.19, 1.05)
5	(0.44, 0.58)	(0.17, 1.32)
6	(0.38, 0.47)	(0.39, 1.63)
7	(0.37, 0.47)	(0.02, 1.02)
8	(2.17, 2.19)	(1.80, 2.41)
9	(2.17, 2.19)	(1.99, 2.55)

[1]All the numbers for the throughput sensitivities in first seven rows should be multiplied by 10^{-3}.

and of IS stations with class-dependent service distributions. The techniques apply without modification to networks containing, in addition to FCFS and IS stations, processor sharing and last-come-first-serve preemptive-resume stations, since the product-form result is unchanged.

With little effort the reader can generalize the methodology to handle class switching and FCFS stations with general load-dependent service rates. (The stationary probabilities continue to have a product form with these features [Baskett et al. 1975]. For example, general load-dependent service rates can be accommodated by simply redefining the evaluation function (8).

We feel that more work remains to be done on the application of Monte Carlo methods to queuing networks. Below we list some interesting problems for future research.

(1) It is of interest to investigate parallel implementations of Monte Carlo summation and integration. For example, if K processors are available, one should be able to run I/K iterations for Monte Carlo integration on each processor, thereby obtaining a speedup close to K.
(2) It would be interesting to design a network optimization procedure based on the gradient estimates given in this paper. In particular, when the current solution is far from the optimal, one may be satisfied with crude estimates obtained from a small number of iterations.
(3) It is of interest to develop a heuristic for choosing good sampling parameters for Monte Carlo summation with class decomposition. How does the second sampling technique for Monte Carlo summation, discussed in Appendix B, perform?
(4) It is of interest to extend the methodology developed here to exotic product-form networks, such as networks with multiple service stations with concurrent chains of customers [Chiola et al. 1988; LeBoudec 1986], networks with state-dependent routing [Towsley 1980; Yao and Buzacott 1987], and networks with service and arrival rates depending on *global* state information [Kelly 1979; Serfozo 1990; 1989].
(5) Is it possible to use the results in Mitra and McKenna [McKenna and Mitra 1984] to develop a Monte Carlo *integration* method for load-dependent FCFS stations?

(6) How do the Monte Carlo methods discussed in this paper, and the combinatorial and asymptotic methods discussed elsewhere, compete with discrete-event simulation? Using the algorithms in Rajasekaran and Ross [1993] the computational effort required to generate an event for discrete-event simulation can be independent of the problem size. Moreover, the estimate of throughput is bounded in discrete-event simulation. Therefore, one can expect discrete-event simulation to perform relatively well for large networks [Glynn and Iglehart 1988].

Appendix A. Avoiding Numerical Overflow and Underflow Problems

The dynamic range of floating numbers in double precision is from $10E-318$ to $10E318$ on most of the 32-bit machines like the VAX and the SUN. Since the arithmetic involving normalization constants can involve numbers exceeding these bounds, we must be careful to avoid potential overflow and underflow problems. The following technique has been used in our code. (Mitra and Ramakrishnan [1990] discuss a similar idea, as part of independent research.)

We set a threshold value, T, equal to $10E150$ and then represent a floating-point number, say A, as aT^I where $|a|$ is either zero or between $1/T$ and T; I is a signed integer. Using this representation, the largest positive number that can be handled is approximately $10E(150*MAX)$, where MAX is the largest possible integer. If integers are 4 bytes long, MAX becomes $2E31\text{-}1$. On the other hand, the smallest positive number that can be handled is approximately $10E(-150*MAX)$.

To compute $C=A*B$, where $A=aT^{I_a}$ and $B=bT^{I_b}$, we simply set $c=a*b$ and $I_c=I_a+I_b$ and normalize (c, I_c) so that $|c|$ satisfies the above-mentioned condition. The normalization process involves the following steps. If $|c|$ is greater than T, we perform the computations $c \leftarrow c/T$ and $I_c \leftarrow I_c+1$ so that $|c|$ satisfies the condition. Similarly, if $|c|$ is less than $1/T$, we perform the computations of $c \leftarrow c*T$ and $I_c \leftarrow I_c-1$ so that $|c|$ satisfies the condition. Here, overflow or underflow will not occur at the product of $a*b$ since the absolute values of both a and b are between $10E-150$ and $10E150$. Similarly, we can calculate $C=A/B$.

Computing $C=A+B$ is a bit tricky, since we must make sure the exponents are matched before we add. The following logic has been used in implementing the addition of two numbers represented by (a, I_a) and (b, I_b):

If $I_a=I_b$, then $c=a+b$ and $I_c=I_a$; if $I_a-I_b=1$, then $c=a+b/T$ and $I_c=I_a$; if $I_a-I_b=-1$, then $c=a+b*T$ and $I_c=I_a$; if $I_a-I_b=2$, then $c=a*T+b/T$ and $I_c=I_a-1$; if $I_a-I_b=-2$, then $c=a/T+b*T$ and $I_c=I_a+1$; if $I_a-I_b>2$, then $c=a$ and $I_c=I_a$; if $I_b-I_a>2$, then $c=b$ and $I_c=I_b$.

The last two steps are for the cases where the magnitudes of A and B differ by at least $T^2=10E300$ so that the loss of accuracy is unavoidable due to the finite precision of the floating-point number representation. After the addition, (c, I_c) needs to be normalized to ensure that the value of c satisfies the condition.

As an example, consider constructing the estimate $\overline{Z}^I$ using the class decomposition method. The entries in the table $t(\cdot)$ and $\xi_m(\cdot)$ created in the setup module should be represented according to their exponent form, as

discussed above. In the execution module, we generate the vector sample $\mathbf{V}^i$ (without using the exponent representation) and index the tables with the vector sample. In order to obtain Z^i, via the right-hand side of (8), we use the multiplication and division rules stated above. We then obtain $Z^1 + \cdots + Z^I$ according to addition rules stated above.

Appendix B. An Alternative Sampling Technique for Monte Carlo Summation

This sampling technique requires each class to pass through an IS station. Without loss of generality, we assume that there is exactly one IS station (permitting class-dependent service distributions). Let station 0 be the IS station and stations $1,\ldots,L$ be the multiple-server FCFS stations. Redefine $\mathbf{n} = (n_{jm}: 1 \le j \le J,\ m = 1,\ldots,L)$ and let

$$\Omega' := \{\mathbf{n}: n_{j1} + \cdots + n_{jL} \le N_j, \quad j = 1,\ldots,J\}.$$

Since $n_{j0} = N_j - n_{j1} - \cdots - n_{jL}$, the normalization constant can be written as

$$g = \sum_{\mathbf{n}\in\Omega'} \left[\prod_{m=1}^{L} f_m(n_m) \prod_{j=1}^{J} \frac{\rho_{jm}^{n_{jm}}}{n_{jm}!}\right] \prod_{j=1}^{J} \frac{\rho_{j0}^{N_j - n_{j1} - \cdots - n_{jL}}}{(N_j - n_{j1} - \cdots - n_{jL})!}$$

$$= c \sum_{\mathbf{n}\in\Omega'} \left[\prod_{m=1}^{L} \frac{f_m(n_m)}{n_m!} \frac{n_m!}{n_{1m}! \cdots n_{Jm}!} \prod_{j=1}^{J} \left(\frac{N_j \rho_{jm}}{\rho_{j0}}\right)^{n_{jm}}\right]$$

$$\times \prod_{j=1}^{J} \sigma(N_j, n_{j1} + \cdots + n_{jL}),$$

where

$$c := \prod_{j=0}^{J} \frac{\rho_{j0}^{N_j}}{N_j!}$$

and

$$\sigma(N,n) := \prod_{i=N-n+1}^{N} \frac{i}{N}.$$

Recall that T_m is the maximum number of customers that can appear at station m.

A good importance-sampling function will have a shape resembling that of the summand. Motivated by the above expression for g, we suggest

$$p(\mathbf{n}) = \prod_{m=1}^{L} \frac{f_m(n_m) x_m^{n_m}}{n_m! \alpha_m} \frac{n_m!}{n_{1m}! \cdots n_{Jm}!} \left(\frac{x_{1m}}{x_m}\right)^{n_{1m}} \cdots \left(\frac{x_{Jm}}{x_m}\right)^{n_{Jm}},$$

$$0 \le n_m \le T_m, \qquad m = 1,\ldots,M,$$

where x_{jm}, $1 \le j \le J$, $1 \le m \le L$ are nonnegative sampling parameters to be chosen to minimize variance, $x_m := x_{1m} + \cdots + x_{Jm}$, and

$$\alpha_m := \sum_{n=0}^{T_m} \frac{f_m(n)}{n!} x_m^n.$$

We can sample from $p(\cdot)$ as follows: For each m, use the alias algorithm to select n_m from the set $\{0, \ldots, T_m\}$ with probability $f_m(n_m)x_m^{n_m}/\alpha_m n_m!$. We then obtain $n_{1m}, \ldots, n_{Jm}$ by distributing n_m balls into J boxes; a ball falls in the jth box with probability x_{jm}/x_m.

With this sampling distribution, we have

$$Z^i = c'1(\mathbf{V}^i \in \Omega')\left[\prod_{m=1}^{L}\prod_{j=1}^{J}\left(\frac{N_j\,\rho_{jm}}{x_{jm}\,\rho_{j0}}\right)^{V_{jm}^i}\right]\prod_{j=1}^{J}\sigma\left(N_j, V_{j1}^i + \cdots + V_{jL}^i\right),$$

where $c' := c\alpha_1 \cdots \alpha_M$. In particular, if we set $x_{jm} = N_j\,\rho_{jm}/\rho_{j0}$, we have

$$Z^i = c'1(\mathbf{V}^i \in \Omega')\prod_{j=1}^{J}\sigma\left(N_j, V_{j1}^i + \cdots + V_{jL}^i\right).$$

We expect this technique to work well if the normal-usage conditions are satisfied, that is, if $y_m < s_m$ with some margin for all $m = 1, \ldots, L$, in which case the majority of the samples $\mathbf{V}^i$ will fall in Ω'.

ACKNOWLEDGMENTS. We would like to thank AT&T Bell Laboratories, and in particular Debasis Mitra and Judith Seery, for permitting us to use PANACEA and helping us to install it on our system at the University of Pennsylvania. We thank Paul Glasserman for pointing out the Glynn and Iglehart [1980] reference to us. We would also like to thank the referees and the associate editor for their thoughtful comments.

REFERENCES

BARD, Y. 1981. A simple approach to systems modeling. *Perf. Eval. 3*, 225–248.

BASKETT, F., CHANDY, K. M., MUNTZ, R. R., AND PALACIOS, F. G. 1975. Open, closed, and mixed networks of queues with different classes of customers. *J. ACM 22*, 3 (Apr.), 248–260.

BAYNAT, B., DALLERY, Y., AND ROSS, K. W. 1992. A non-MVA method for the approximate analysis of multi-server, multi-class BCMP networks. Tech. Rep., Laboratoire MASI.

BRATLEY, P., FOX, B. L., AND SCHRAGE, L. E. 1987. *A Guide to Simulation*. Springer-Verlag, New York.

BUZEN, J. P. 1973. Computational algorithms for closed queueing networks with exponential servers. *Commun. ACM 16*, 9 (Sept.), 527–531.

CHANDY, K. M., AND NEUSE, D. 1982. Linearizer: A heuristic algorithm for queueing network models of computer systems. *Commun. ACM 25*, 2 (Feb.), 126–134.

CHIOLA, G., MARSAN, M. A., AND BALBO, G. 1988. Product-form solution techniques for the performance analysis of multiple-bus multiprocessor systems with nonuniform memory references. *IEEE Trans. Comput. 37*, 532–540.

CONWAY, A. E., AND GEORGANAS, N. D. 1989. *Queueing Networks—Exact Computational Algorithms: A Unified Theory Based on Decomposition and Aggregation*. MIT Press, Cambridge, Mass.

GELENBE, E., AND MITRANI, I. 1980. *Analysis and Synthesis of Computer Systems*. Academic Press. London, England.

GLYNN, P. W., AND IGLEHART, D. L. 1988. Simulation methods for queues: An overview. *Queue. Syst. 3*, 221–256.

KELLY, F. P. 1979. *Reversibility and Stochastic Networks*. Wiley, Chichester, England.

KELLY, F. 1986. Blocking probabilities in large circuit-switched networks. *Adv. Appl. Prob. 18*, 473–505.

LAM, S. S., AND LIEN, Y. L. 1983. A tree convoluted algorithm for the solution of queueing networks. *Commun. ACM 26*, 3 (Mar.), 203–215.

LEBOUDEC, J. Y. 1986. A BCMP extension to multiserver stations with concurrent classes of customers. *ACM Performance Evolution Preview 14*, 78–91.

McKENNA, J., AND MITRA, D. 1984. Asymptotic expansions and integral representations of moments of queue lengths in closed Markovian networks. *J. ACM 31*, 2 (Apr.), 346–360.

McKENNA, J., AND MITRA, D. 1982. Integral representations and asymptotic expansions for closed Markovian queueing networks: Normal usage. *Bell Syst. Tech. J. 61*, 661–683.

McKENNA, J., MITRA, D., AND RAMAKRISHNAN, K. G. 1981. A class of closed Markovian queueing networks: Integral representations, Asymptotic expansions, and generalizations. *Bell Syst. Tech. J. 60*, 599–641.

MILLER, R. G. 1974. The jackknife—A review. *Biometrika 61*, 1–5.

MITRA, D., AND McKENNA, J. 1986. Asymptotic expansions for closed Markovian networks with state-dependent service rates. *J. ACM 33*, 3 (July), 568–592.

MITRA, D., AND RAMAKRISHNAN, K. G. 1990. A numerical investigation into the optimal design of congestion controls for high speed data networks. Tech. Rep., AT & T Bell Laboratories.

RAJASEKARAN, S., AND ROSS, K. W. Fast algorithms for generating discrete random variates with changing distributions. *ACM Trans. Model. Comput. Simul., 3*, 1–19.

RAMAKRISHNAN, K. G., AND MITRA, D. 1982. An overview of PANACEA, a software package for analyzing Markovian queueing networks. *Bell Syst. Tech. J. 61*, 568–592.

REISER, M., AND KOBAYASHI, H. 1975. Queueing networks with multiple closed chains: Theory and computational algorithms. *IBM J. Res. Devel. 19*, 283–294.

REISER, M., AND LAVENBERG, S. S. 1980. Mean-value analysis of closed multichain queuing networks. *J. ACM 27*, 2 (Apr.), 313–322.

ROSS, K. W., AND WANG, J. Asymptotically optimal importance sampling for product-form queueing networks. *ACM Trans. Model. Comput. Simul. 3*, 244–268.

ROSS, K. W., AND WANG, J. 1992. Monte Carlo summation applied to product—form loss networks. *Prob. Eng. Inf. Sci. 6*, 323–348.

ROSS, K. W., AND WANG, J. 1990. Solving product form stochastic networks with Monte Carlo summation. In *Proceedings of the 1990 Winter Simulation Conference*. IEEE, New York.

RUBINSTEIN, R. Y. 1981. *Simulation and the Monte Carlo Method*. Wiley, New York.

SAUER, C. H., AND CHANDI, K. M. 1981. *Computer Systems Performance Modeling*. Prentice-Hall, Englewood Cliffs, N.J.

SCHWEITZER, P. 1979. Approximate analysis of multiclass closed queueing networks. In *Proceedings for International Conference on Stochastic Control and Optimization.*

SERFOZO, R. F. 1988. Reversibility and compound birth death and migration processes. Tech. Rep. School of Industrial and Systems Engineering, Georgia Inst. of Technology.

SERFOZO, R. F. 1989. Markovian network processes: Congestion dependent routing and processing. *Queue. Syst. Theory. Appl. 5*. 5–36.

TOWSLEY, D. 1980. Queuing network models with state-dependent routing. *J. ACM 27*, 2 (Apr.), 323–337.

TSANG, D., AND ROSS, K. W. 1990. Algorithms for determining exact blocking probabilities in tree networks. *IEEE Trans. Commun. 38*, 1266–1271.

YAO, D. D., AND BUZACOTT, J. A. 1987. Modeling a class of flexible manufacturing systems with reversible routing. *Oper. Res. 35*, 87–93.

RECEIVED JUNE 1992; REVISED JUNE 1993; ACCEPTED OCTOBER 1993