ELSEVIER

Performance Evaluation 29 (1997) 273–292

PERFORMANCE EVALUATION
An International Journal

Implementation of Monte Carlo integration for the analysis of product-form queueing networks ★

Keith W. Ross [a,1], Jie Wang [b,*]

[a] *Department of Systems, University of Pennsylvania, Philadelphia, PA 19104, USA*
[b] *AT&T Laboratories, Room 1L-219 Holmdel, NJ 07733, USA*

Received 13 October 1994; revised 30 May 1996

Abstract

MonteQueue is a new public-domain software package which rapidly solves large and small multiclass product-form queueing networks with multiple- and single-server stations over a wide range of traffic conditions. MonteQueue obtains estimates of performance measures by applying importance sampling to sum and integral representations of the network's normalization constants. This paper discusses the implementation issues and surveys of the theoretical properties of the four importance sampling techniques included in MonteQueue. It also presents new numerical data which compare the performance of the four techniques.

Keywords: Queueing networks; Markov chain; Monte Carlo integration; Importance sampling; Normalization constant

1. Introduction

Closed product-form queueing networks are valuable tools for analyzing manufacturing systems, high-speed communication networks, and computer systems. MonteQueue 2.0 is a new public-domain software package which rapidly solves large and small multiclass product-form networks with multiple- and single-server stations over a wide range of traffic conditions.

MonteQueue 2.0 exploits the well-known fact that key performance measures – including throughputs, utilizations, and their gradients – can be represented as simple functions of normalization constants. MonteQueue 2.0 obtains estimates of performance measures by applying importance sampling to sum and integral representations of the normalization constants. By invoking the central limit theorem, MonteQueue 2.0 also provides confidence intervals for its estimates.

★ Supported partially by NSF grant NCR93-04601.
* Corresponding author. E-mail: jie@buckaroo.att.com.
[1] E-mail: ross@eniac.seas.upenn.edu.

PII S0166-5316(96)00051-X

In an earlier paper Ross et al. [14] discuss MohteQueue 1.0, a preliminary software package that applies Monte Carlo summation and integration to product-form queueing networks. MonteQueue 1.0 provides the user with two Monte Carlo techniques: integration with exponential sampling and summation with decomposition sampling. Both techniques apply to networks with a large or small number of stations, classes, and customers. Ross et al. [14] present the results of several numerical tests for symmetric and asymmetric networks. Integration with exponential sampling gives highly accurate results for light to moderately heavy traffic conditions and fairly accurate results for heavy traffic conditions. But it does not apply to networks with multiple-server stations (that is, stations with more than one but less than an infinite number of servers). Summation with decomposition sampling consumes more time and memory than integration with exponential sampling, gives less accurate results, but applies to networks containing multiple-server stations.

Ross et al. [14] also present a detailed comparison of the Monte Carlo methods with existing methodologies. Briefly, the Monte Carlo methods give highly accurate estimates with negligible CPU time for the networks that are tractable by combinatorial algorithms (that is, convolution and mean-value algorithms). But unlike the combinatorial algorithms, the Monte Carlo methods can also solve networks which simultaneously have a large number of stations, classes, and customers. We mention here that a new set of tools based on generating functions have recently been developed [3,4]. These tools can quickly solve many classes of networks, including some large networks with special structure. But they are also nonpolynomial combinatorial algorithms, and are ineffective when the numbers of stations, classes, and customers are all large.

PANACEA, a popular software package based on asymptotic expansions of the normalization constant, can solve many large networks [13]. PANACEA has the advantage over the Monte Carlo methods, in that it returns bounds for performance measures, whereas the Monte Carlo methods return confidence intervals. However, MonteQueue 1.0 has two advantages over PANACEA. First, it allows for multiple-server stations whereas PANACEA does not. Second, it gives meaningful results for a wider range of traffic conditions. Specifically, when PANACEA returns tight bounds, MonteQueue 1.0 returns tight confidence intervals in a comparable amount of time; but for moderate-to-heavy loads at single-server station, PANACEA often gives poor bounds when MonteQueue 1.0 continues to give respectably tight confidence intervals. We mention that the asymptotic expansion theory was extended to cover networks with multiple-server stations and higher traffic loads [11]. However, this theory is deep and intricate, and to our knowledge has never been implemented in software. An interesting research avenue would be to develop a numerically robust code that implements the theory in [11], and then compare the performance of this asymptotic method with MonteQueue 2.0.

MonteQueue 2.0, written in C and available in the public domain, improves on the earlier package in three important ways. [2] The first improvemnt is to include a novel and powerful importance sampling technique – summation with rejection sampling. This new technique applies to networks with multiple-server stations, and it is typically more accurate than summation with decomposition sampling. The second improvement is to incorporate a new heuristic, based on a pilot run, for choosing the importance sampling parameters for integration with exponential sampling. With this modification, the high accuracy of integration with exponential sampling extends to heavy traffic conditions. The third improvement is to include integration with truncated normal sampling, another novel sampling technique which has appealing theoretical properties and which performs better than the other sampling techniques for certain networks.

[2] MonteQueue 2.0 is available at the ftp server systems.seas.upenn.edu. in the directory/public/MonteQueue (use anonymous for login name and use your name for password).

Thus MonteQueue 2.0 includes four importance sampling techniques: integration with exponential sampling (IE); summation with rejection (SR); summation with decomposition (SD); and integration with truncated normal (ITN). Most networks are best solved by either IE or SR. Specifically, if the network does not contain any multiple-server stations, then IE with the new pilot-run heuristic gives remarkably tight confidence intervals for essentially all traffic conditions. If the network contains multiple-server stations and the loadings are moderately heavy or less, then SR gives tight confidence intervals. The remaining two sampling techniques are useful for a few particular classes of networks. Specifically, unlike the other techniques, SD can be applied to networks that have no infinite-server stations and at least one multiple-server station; however, the confidence intervals for the same number of iterations are relatively large. ITN can give good results if the network has a large number of nodes, no multiple-server stations, moderately heavy loads at the single-server stations, and symmetric routing.

The current paper has a threefold contribution. First, the paper collects and surveys the key results from our earlier papers that have a direct bearing on the design of MonteQueue 2.0. Second, the paper presents the following new analytical result: summation with rejection sampling is asymptotically optimal for estimating the normalization constant when the network is in normal usage. Third and most important, by appealing to analytical results and new numerical studies, the paper explains why and how the four sampling techniques indicated above are implemented in MonteQueue 2.0.

2. Preliminaries

2.1. *Product-form queueing networks*

We consider closed queueing networks consisting 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 at a 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 and general 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 has a 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 $m = L+1, \ldots, M$. Also let

$$\rho_{j0} = \sum_{m=L+1}^{M} \rho_{jm}, \quad 1 \le j \le J.$$

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

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

Denote $\boldsymbol{n}_m = (n_{1m}, \ldots, n_{Jm})$ 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

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

It is well known [1,5,8] that the equilibrium probability of being in state $\boldsymbol{n} \in \Omega$ is given by $\pi(\boldsymbol{n}) = f(\boldsymbol{n})/g$, where

$$g = \sum_{\boldsymbol{n} \in \Omega} f(\boldsymbol{n})$$

is the *normalization constant*. Many performance measures of interest can be expressed as simple functions of multidimensional sums. For example, the throughput of class-j customers through station m is

$$\mathrm{TH}_{jm} = \lambda_{jm} \frac{g_j}{g},$$

where g_j is the normalization constant for the same network except for one less class-j customer. Gradients, such as derivatives with respect to service rates, can also be expressed as simple functions of multidimensional sums [14].

The above product-form result continues to hold if we also permit stations to be processor sharing or last-come-first-serve pre-emptive resume; *the Monte Carlo method is therefore unchanged if the network includes stations with these disciplines.*

The quantity

$$\hat{y}_m = \frac{1}{s_m} \sum_{j=1}^{J} \frac{N_j \rho_{jm}}{\rho_{j0}}$$

is a measure of the amount of traffic at the mth FCFS station. This measure leads to the following important definitions [9,16] which facilitate the analysis of closed multiclass networks. The FCFS station m is said to be in *normal usage* if $\hat{y}_m < 1$. The FCFS station m is said to be in *critical usage* if $\hat{y}_m = 1$. The FCFS station m is said to be *near critical usage* if $\hat{y}_m \approx 1$. A network is said to be in normal usage if all its FCFS stations are in normal usage; we similarly define a network in critical usage and near critical usage. Finally, we say a network is in *mixed usage* if all its FCFS stations either are in normal usage or are in near critical usage.

Roughly speaking, a station m is in normal usage if its expected number of utilized servers is significantly below s_m. For example, if the network has 10 customers in each class, station m will typically be in normal usage if its expected number of utilized servers is less than $0.7\, s_m$; it can be in normal usage for a higher utilization if the customer populations are larger.

If a network is in normal usage and its populations tend to infinity, then the network tends to a collection of L independent $M/M/s_m$ stable queues [18]. If the network is in critical usage and the populations tend

to infinity, then the expected line lengths also tend to infinity but at a relatively slow rate [17]. We believe that essentially all networks of practical interest are either in normal usage, critical usage, or mixed usage. This network classification has had a profound impact on the design of MonteQueue 2.0.

2.2. The integral representation

There is an alternative representation for the normalization constant, a representation which gives rise to an efficient Monte Carlo method. *This representation requires each FCFS station to have exactly one server.* With this restriction, McKenna and Mitra [9] showed that the normalization constant g of the multiclass queueing network described in Section 2.1 can be expressed as a multidimensional integral:

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

where

$$\boldsymbol{u} = (u_1, \ldots, u_L)', \quad \mathbf{1} = (1, \ldots, 1)', \quad \boldsymbol{\rho}_j = (\rho_{j1}, \ldots, \rho_{jL})',$$
$$\mathbb{Q}^+ = \{\boldsymbol{u} \in R^L : u_l \geq 0, \ l = 1, \ldots, L\}.$$

Similarly, the normalization constant g_j can be expressed as multidimensional integral.

2.3. Monte Carlo integration and summation

Ross et al. [14] suggested estimating the integral representation of the normalization constant with Monte Carlo integration. We now summarize this method. Let $\boldsymbol{V}^i = (V_1^i, \ldots, V_L^i)', i = 1, 2, \ldots$, be a sequence of independent and identically distributed L dimensional random vectors with probability density function $p(\cdot)$ defined over $\mathbb{Q}^+$. The density $p(\cdot)$ is called the *importance sampling function*, and there is great latitude in its choice. Let

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

and define $\bar{Z}^I$ as the sample average of $Z^1, \ldots, Z^I$. Then $\bar{Z}^I$ is an unbiased estimator of g, that is, $E[\bar{Z}^I] = g$. Moreover, the central limit theorem implies that for large I

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

where $c(\alpha)$ is the critical value of the standard normal distribution and $\sigma_I^2(Z)$ is the sample variance of $Z^1, \ldots, Z^I$. 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 (1). 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 key to the success of the method is to find a probability density $p(\cdot)$ that is easy to sample from and has small $\text{var}(Z^1)$.

Ross and Wang [15] also suggested estimating the summation representation of the normalization constant with Monte Carlo summation. We now review this method. Let $\boldsymbol{V}^i, i = 1, 2, \ldots$, be a sequence of independent and identically distributed $J \times M$ dimensional random vectors. We refer to $\boldsymbol{V}^i$ 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

$$\Lambda = \{0, 1, 2, \ldots\}^{JM}.$$

We require $\boldsymbol{V}^i$ to take values in Λ. Denote $p(\boldsymbol{n}), \boldsymbol{n} \in \Lambda$, for the probability mass function (also called the importance sampling function) for $\boldsymbol{V}^i$. Let

$$Z^i = \frac{f(\boldsymbol{V}^i)}{p(\boldsymbol{V}^i)} 1(\boldsymbol{V}^i \in \Omega)$$

and $\bar{Z}^I$ be the sample average of $Z^1, \ldots, Z^I$. Then $\bar{Z}^I$ is again unbiased estimator for g. Moreover, the central limit theorem can again be applied to obtain confidence intervals.

For both Monte Carlo integration and summation, the estimate of throughput of class-j customers at station m takes the form of a ratio:

$$\Phi^I_{jm} = \lambda_{jm} \frac{\sum_{i=1}^I Z^i_j}{\sum_{i=1}^I Z^i},$$

where Z^i_j is an estimate for g_j [14]. We normally perform the sampling so that the correlation between Z^i and Z^i_j is as large as possible [14]. Although Φ^I_{jm} is a biased estimator for TH_{jm}, confidence intervals for TH_{jm} are easily constructed with the central limit theorem for nonlinear functions of sums of i.i.d. random variables (see Glynn and Iglehart [6], Theorem 1).

2.4. An asymptotic regime

Following [9], now consider an infinite sequence of closed queueing networks indexed by N. Suppose that each of the networks has J classes, L FCFS stations, and $M - L$ IS stations. Suppose station m has s_m servers. Suppose that the population size and the relative loads depends on N; in particular, for the Nth network, suppose that

$$N_j = \beta_j N + \alpha_j \sqrt{N}, \quad j = 1, \ldots, J$$

and

$$\frac{\rho_{jm}}{\rho_{j0}} = \frac{\Gamma_{jm}}{N}, \quad j = 1, \ldots, J, \quad m = 1, \ldots, L,$$

where the β_j's are given positive constants, the Γ_{jm}'s are given nonnegative constants, and the α_j's are given arbitrary constants. Thus as $N \to \infty$ the population size increase but the relative load, ρ_{jm}/ρ_{j0}, at each FCFS station decreases. For each network N we can define the load $\hat{y}_m$ at the FCFS stations (see Section 2.1); note that

$$\lim_{N \to \infty} \hat{y}_m = y_m, \quad m = 1, \ldots, L,$$

where

$$y_m = \frac{1}{s_m} \sum_{j=1}^{J} \beta_j \Gamma_{jm}.$$

Recall the definitions of normal and critical usage for a network with finite data. The above limit motivates the following definitions. We say that *the sequence of network* is in *normal usage* if $y_m < 1$ for all $m = 1, \ldots, L$, and that the sequence of networks is in critical usage if $y_m = 1$ for all $m = 1, \ldots, L$. Let $\boldsymbol{y} = (y_1, \ldots, y_L)$.

In this paper we repeatedly examine the Monte Carlo technique in the context of this asymptotic regime. In particular, fix N and let $p(\cdot)$ be an arbitrary importance sampling function (for Monte Carlo integration or summation); denote

$$\overline{\text{var}(N)} = \frac{\text{var}(\bar{Z}^I)}{E[\bar{Z}^I]^2}$$

for the relative variance $\bar{Z}^I$ associated with the sampling function $p(\cdot)$. Of course, $\overline{\text{var}(N)} \geq 0$. Say that the importance sampling function $p(\cdot)$ is *asymptotically optimal* if

$$\lim_{N \to \infty} \overline{\text{var}(N)} = 0.$$

3. Integration with exponential sampling

Throughout Sections 3 and 4 we assume that each FCFS station has one server. This assumption permits an integral representation of the normalization constant, as discussed in Section 2.2.

Recall the Monte Carlo integration method outlined in Section 2.3. We refer to the importance sampling function

$$q_\gamma(\boldsymbol{u}) = \prod_{m=1}^{L} \gamma_m \, \mathrm{e}^{-\gamma_m u_m}, \quad \boldsymbol{u} \in \mathbb{Q}^+,$$

as *independent exponential sampling* with vector parameter $\boldsymbol{\gamma} = (\gamma_1, \ldots, \gamma_L)$.

Generating the vector sample $\boldsymbol{V}^i$ with independent exponential sampling requires $O(L)$ computational effort. Evaluating Z^i from $\boldsymbol{V}^i$ requires $O(\sum_{j=1}^{J} \log N_j)$ operations when successive squaring is used to evaluate the exponents involving the N_j's. Thus the overall effort to generate an estimate Z^i grows linearly with the number of FCFS stations and grows very slowly with the population sizes. To estimate the throughout TH_{jm}, the same sequence of $\boldsymbol{V}^i$'s can be employed to evaluate the numerator and denominator of Φ^I_{jm}.

Independent exponential sampling is studied numerically and analytically in [14]; additional analytical results are reported in [16]. We now summarize some of the more important results and observations from these papers.

3.1. Asymptotic optimality

Recall the asymptotic regime discussed in Section 2.4. The following result is a special case of results proved in [16].

Theorem 1. *Suppose that the sequence of the networks is in normal usage, that is, $y_m < 1$ for all $m = 1, \ldots, L$. Further suppose the same sampling function $p(\boldsymbol{u}), \boldsymbol{u} \in \mathbb{Q}^+$, is used for each network in the sequence.*

(i) *If $p(\boldsymbol{u}) = q_{\mathbf{1}-y}(\boldsymbol{u}), \boldsymbol{u} \in \mathbb{Q}^+$, then*

$$\lim_{N\to\infty} \overline{\mathrm{var}(N)} = 0.$$

(ii) *If $p(\boldsymbol{u}) \neq q_{\mathbf{1}-y}(\boldsymbol{u})$ for some $\boldsymbol{u} \in \mathbb{Q}^+$, then*

$$\lim_{N\to\infty} \inf \overline{\mathrm{var}(N)} > 0.$$

Thus independent exponential sampling with parameters $\gamma_m = 1 - y_m, m = 1, \ldots, L$, is asymptotically optimal for estimating the normalization constant when the sequence of networks is in normal usage. Consequently, for a given network with finite data, if the N_j's are large and $\hat{y}_m < 1$ for all $m = 1, \ldots, L$, then we would expect independent exponential sampling with parameters $\gamma_m = 1 - \hat{y}_m, m = 1, \ldots, M$, to give accurate estimates of the normalization constant.

3.2. Implementation in MonteQueue 2.0

Although independent exponential sampling is not asymptotically optimal for estimating throughput, computational testing has shown that it nevertheless gives excellent throughput estimates when $\gamma_m = 1 - \mathrm{util}_m, m = 1, \ldots, L$ [14]. Here util_m is an estimate of the utilization of the single server at the mth FCFS station. But how do we obtain the estimates for the utilizations? If the network is normal usage, then $\hat{y}_m$ is an estimate for the utilization which is asymptotically correct, so that $\mathrm{util}_m = \hat{y}_m$ is a reasonable choice. If the network is near critical usage or in mixed usage, estimates for utilization can be obtained with pilot runs as discussed below.

MonteQueue 2.0 implements integration with exponential sampling, henceforth called the IE method, as follows. The program first calculates $\hat{y}_m, m = 1, \ldots, L$, from the network data. If $\hat{y}_m < 0.9$ for all m, the program considers the network as being in normal usage and runs the Monte Carlo technique with $\gamma_m = 1 - \hat{y}_m$ for a number of iterations specified by the user. On the other hand, if $\hat{y}_m \geq 0.9$ for some FCFS station m, then the program considers the network as being outside of normal usage and proceeds to estimate $\mathrm{util}_m, m = 1, \ldots, L$. This is done by automatically performing a pilot run of Monte Carlo integration for 5000 iterations with

$$\gamma_m = \begin{cases} 1 - \hat{y}_m, & \hat{y}_m < 0.9, \\ \dfrac{1}{\sqrt{\max(N_1, \ldots, N_J)}}, & \hat{y}_m \geq 0.9. \end{cases}$$

(The above heuristic for $\hat{y}_m \geq 0.9$ is motivated by the theory developed in [17].) The pilot run gives estimates which are fed back into the main program. The main program is then run with $\gamma_m = \max(1 - \mathrm{util}_m, 0.01), m = 1, \ldots, L$, for a number of iterations specified by the user. (MonteQueue 1.0 does not automatically determine the importance sampling parameters when $\hat{y}_m \geq 0.9$; instead it requires the user to provide the parameters as input.)

4. Integration with truncated normal sampling

One motivation for using exponential sampling for Monte Carlo integration is that it is asymptotically optimal for estimating the normalization constant for a sequence of networks in normal usage. But suppose we wish to analyze a network that is in – or near – critical usage: Is there an importance sampling function with some theoretical justification? Ross and Wang [16] show that the truncated normal distribution is such a distribution. But computational testing with this distribution has not been previously reported. In this section we summarize the theoretical results of Ross and Wang [16], indicate how truncated normal sampling can be implemented in software, and present computational results for Monte Carlo integration with truncated normal distributions.

4.1. Asymptotic optimality

Recall the asymptotic defined in Section 2.4. Let

$$\Gamma(l,m) = \sum_{j=1}^{J} \beta_j \Gamma_{jl} \Gamma_{jm},$$

and let $\boldsymbol{\Gamma}$ be the corresponding $L \times L$ matrix. We suppose that $\boldsymbol{\Gamma}$ is invertible. Further let

$$z_m = \sum_j \alpha_j \Gamma_{jm}, \quad m = 1, \ldots, L,$$

$$z = (z_1, \ldots, z_L)', \qquad \boldsymbol{u}_0 = \boldsymbol{\Gamma}^{-1} z.$$

Let

$$p^*(\boldsymbol{u}) = \frac{\exp\{-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{u}_0)'\boldsymbol{\Gamma}(\boldsymbol{u}-\boldsymbol{u}_0)\}}{\int_{\mathbb{Q}^+} \exp\{-\frac{1}{2}(\boldsymbol{u}-\boldsymbol{u}_0)'\boldsymbol{\Gamma}(\boldsymbol{u}-\boldsymbol{u}_0)\} d\boldsymbol{u}}, \quad \boldsymbol{u} \in \mathbb{Q}^+,$$

be an importance sampling function. Note that $p^*(\boldsymbol{u})$ is the density of a multivariate normal distribution with mean vector $\boldsymbol{u}_0$ and covariance matrix $\boldsymbol{\Gamma}^{-1}$ *truncated* to $\mathbb{Q}^+$. In particular, the components of $\boldsymbol{V}^i$ are *not* independent. The following result is proved in [16].

Theorem 2. *Suppose the sequence of networks is in critical usage and that $\boldsymbol{\Gamma}$ is invertible. Further suppose that the same sampling function $p(\boldsymbol{u}), \boldsymbol{u} \in \mathbb{Q}^+$, is used for each network in the sequence.*

(i) *If $p(\boldsymbol{u}) = p^*(\boldsymbol{u})$ for all $\boldsymbol{u} \in \mathbb{Q}^+$, then*

$$\lim_{N\to\infty} \overline{\mathrm{var}(N)} = 0.$$

(ii) *If $p(\boldsymbol{u}) \neq p^*(\boldsymbol{u})$ for some $\boldsymbol{u} \in \mathbb{Q}^+$, then*

$$\lim_{N\to\infty} \inf \overline{\mathrm{var}(N)} > 0.$$

Thus $p^*(\cdot)$ is asymptotically optimal for estimating the normalization constant when the sequence of networks is in critical usage.

But because $p^*(\cdot)$ is a multivariate distribution truncated to the positive quadrant, it is not easy to sample from. We are, therefore, motivated to search another sampling function that approximates $p^*(\cdot)$ and that is easy to sample from. One such approximation is to neglect the off-diagonal terms $\boldsymbol{\Gamma}$, which gives

$$p(\boldsymbol{u}) = \frac{1}{c} \prod_{m=1}^{L} \exp\{-\tfrac{1}{2}(u_m - u_{m0})^2 \Gamma(m,m)\}, \quad \boldsymbol{u} \in \mathbb{Q}^+ \tag{2}$$

where

$$c = \prod_{m=1}^{L} \int_0^\infty \exp\{-\tfrac{1}{2}(u_m - u_{m0})^2 \Gamma(m,m)\}\, du_m;$$

note that this density calls for independent sampling of L univariate truncated normals. Such an approximation can be justified when the traffic has a symmetric pattern and the network is large; for example, when the number of stations is large, there is a distinct class routed through each pair of FCFS stations, the service rates are roughly equal across stations, and the populations are roughly equal across classes [17]. For these networks, the off-diagonal terms of $\boldsymbol{\Gamma}$ are much smaller than the diagonal terms, hence justifying the assumption. We refer to this sampling distribution as ITN sampling. In order to use ITN, the parameters u_{m0} and $\Gamma(m,m)$, $m = 1, \ldots, L$, in (2) must be specified. The computational effort for ITN is essentially the same as that for IE (see Section 3).

4.2. Computational testing

In view of the asymptotic result summarized above, we might expect truncated normal sampling to give accurate estimates of the normalization constant for a large network with symmetric patterns, and whose stations are nearly in critical usage. We might also hope that truncated normal sampling gives good estimates for throughputs under the same network conditions. We now present some new computational results which compare IE with truncated normal sampling.

Our test network has eight single-server FCFS stations and one IS station. For each set of two FCFS stations, there is one class that visits each of the two FCFS stations cyclically and then visits the IS station. (Thus there are 28 classes and each station is visited by seven classes.) Each class has 100 customers. Each server at the IS station works at rate $\mu_{j0} = 0.002$ for all classes. The servers at the FCFS stations all work at rate μ. Note that the network is symmetric and that

$$\hat{y}_m \sum_{j=1}^{J} N_j \frac{\rho_{jm}}{\rho_{j0}} = \frac{7 \times 100 \times 0.002}{\mu} = \frac{1.4}{\mu}.$$

Thus the network is in critical usage when $\mu = 1.4$ and is "near" critical usage when μ is "near" 1.4. Note also that the routing is symmetric.

In order to implement truncated normal sampling, we must first choose the parameters u_{m0} and $\Gamma(m,m)$, $m = 1, \ldots, L$. We now offer a heuristic to choose the parameters for the symmetric network. To this end, we first specify asymptotic parameters that match our test network. As in [13] we let

$$N = \max\left\{\frac{\rho_{j0}}{\rho_{jm}} : \rho_{jm} > 0, 1 \le j \le J, 1 \le m \le L\right\}.$$

For the symmetric network at hand, we have $N = 500\mu$ and

$$\Gamma_{jm} = N\frac{\rho_{jm}}{\rho_{j0}} = \begin{cases} 1 & \text{if } \rho_{jm} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

Critical usage requires that

$$1 = \sum_{j=1}^{J} \beta_j \Gamma_{jm} = \sum_{j:\,\rho_{jm}>0} \beta_j.$$

There is an infinite number of solutions to this equation; since the network is symmetric, we take $\beta_j = \frac{1}{7}$, $j = 1, \ldots, 28$. The asymptotic bias parameters must then be set to

$$\alpha_j = \frac{N_j - \beta_j N}{\sqrt{N}} = \frac{70 - 50\mu}{7\sqrt{5\mu}}.$$

Having specified the asymptotic parameters, we can now calculate the sampling parameters from their definitions:

$$\Gamma(m,m) = \sum_{j:\,\rho_{jm}>0} \beta_j \Gamma_{jm}^2 = 1, \qquad z_m = \sum_{j:\,\rho_{jm}>0} \alpha_j = \frac{70 - 50\mu}{\sqrt{5\mu}}.$$

Since we are (falsely) assuming that $\Gamma(l,m) = 0$ for all $l \neq m$, we also have

$$u_{m0} = \frac{z_m}{\Gamma(m,m)} = \frac{70 - 50\mu}{\sqrt{5\mu}}.$$

We ran MonteQueue 2.0 for 1000 iterations for ITN and IE. ITN consumes about 6 s of CPU time for all traffic conditions; IE consumes about 6 s when $\hat{y}_m < 0.9$ and about 36 s when $\hat{y}_m \geq 0.9$ (due to the pilot run consisting of 5000 additional iterations).

Fig. 1 compares the performance of the two techniques for estimating the normalization constant. It plots the normalized width of the estimate of the normalization constant as a function of the loading, $\hat{y}_m$. We define the normalized width of an estimate as the width of the 95% confidence interval divided by the midpoint of the interval. From the figure we observe that if $\hat{y}_m \leq 0.35$, so that the network is clearly in normal usage, then IE performs better than ITN as expected. For $0.35 < \hat{y}_m < 1.08$, so that the network is closer to critical usage, ITN performs better. The drop at $\hat{y}_m = 0.9$ in the graph for IE is due to the introduction of the pilot run; there is also an increase in CPU time for IE at this load. Finally, for $\hat{y}_m > 1.08$, which corresponds to heavy usage, IE performs better.

Fig. 2 compares the performance of the two techniques for estimating throughput, TH_{jm}. The normalized confidence interval width for a given loading and importance sampling technique is obtained by taking the maximum normalized width over all (j,m) pairs. From the figure we observe that both techniques give remarkably tight confidence intervals, even in the vicinity of critical usage. Furthermore, for $\hat{y}_m \leq 1.04$, the performance of the two techniques is roughly the same. We also tried running ITN with parameters that differ from those given by the heuristic – no significant reduction in the width of the confidence intervals was observed.

Both IE and ITN are available in MonteQueue 2.0. With IE, the importance sampling parameters are fixed automatically; with ITN, they must be specified by the user.

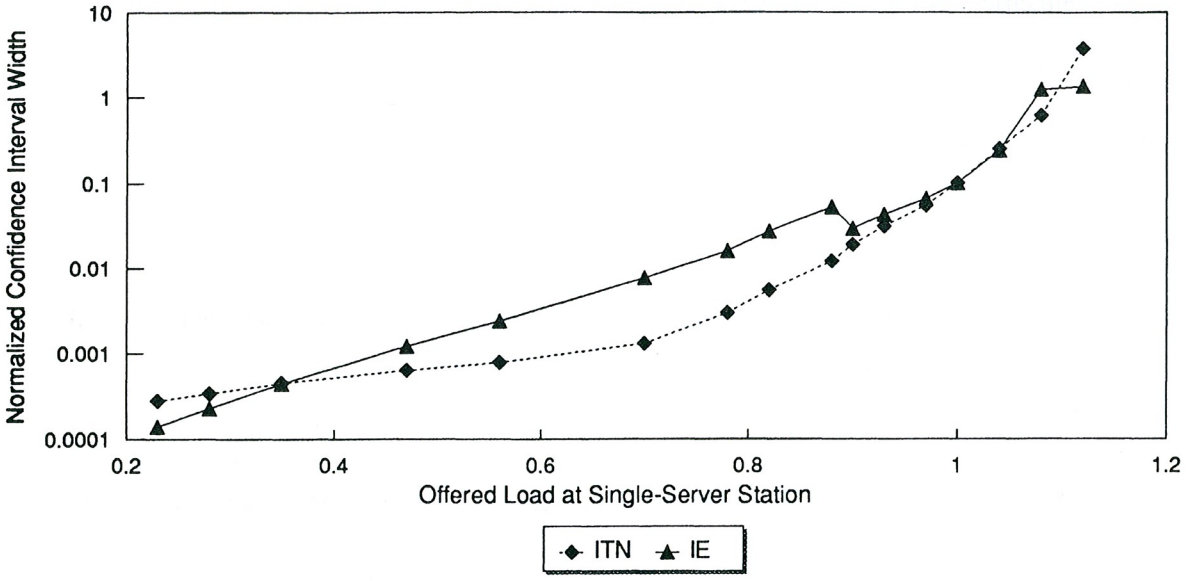


Fig. 1. The performance of IE and ITN for estimating the normalization constant as a function of the offered load, $\hat{y}_m$.

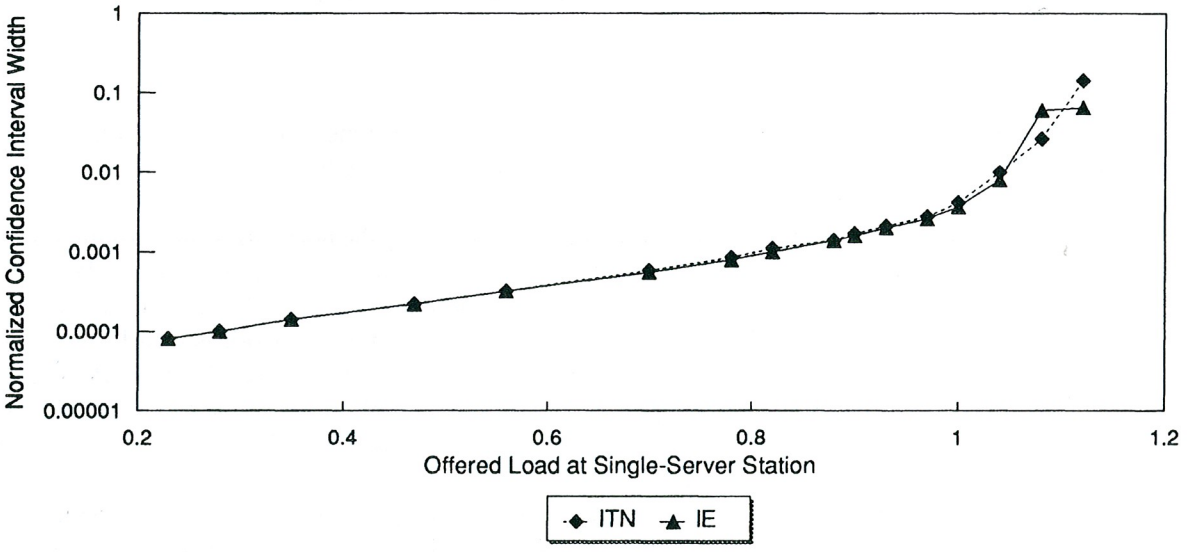


Fig. 2. The performance of IE and ITN for estimating throughput as a function of the offered load, $\hat{y}_m$.

For a given network with no multiple-server stations, should the user select IE or ITN? Keeping in mind that the above test network is optimized for ITN and that for this network their performance is roughly the same, we recommend that IE be first tried. If the network is large, the routing is symmetric, and the load is moderate to heavy at the single-server stations, the user may also want to try ITN.

5. Summation with decomposition sampling

We now permit FCFS stations to have multiple servers. Recall the Monte Carlo summation method discussed in Section 2.3.

For Monte Carlo summation, Ross et al. [14] proposed decomposition sampling, which we now discuss briefly. The technique generates component samples which are independent across classes, but dependent across stations so that every vector sample falls in the state space Λ. Since sampling is independent across classes, the importance sampling function takes the form

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

In order to specify how sampling is done across stations for the jth class, let

$$\Omega_j = \{(n_{j1}, \ldots, n_{jM}) : n_{j1} + \cdots + n_{jM} = N_j\},$$

which is the state space for a single-class Jackson network. MonteQueue uses

$$p_j(n_{j1}, \ldots, n_{jM}) = \frac{1}{G_j} \left(\prod_{m=1}^{L} \rho_{jm}^{n_{jm}} \right) \left(\prod_{n=L+1}^{M} \frac{\rho_{jm}^{n_{jm}}}{n_{jm}!} \right), \quad (n_{j1}, \ldots, n_{jM}) \in \Omega_j,$$

where G_j is defined so that the sum of the probabilities over Ω_j is equal to one. Since the component samples across stations are dependent, sampling from $p_j(\cdot)$ is nontrivial. Nevertheless, a technique was presented in [14] which requires a modest $O(JM)$ time for each sample $\boldsymbol{V}^i$ and which has memory requirements of $O(M \sum_{j=1}^{J} N_j^2)$. We refer to this technique as summation with decomposition (SD).

SD has several appealing features. Unlike the Monte Carlo integration techniques, it can be applied to networks with multiserver queues subject to arbitrary traffic conditions. Unlike rejection sampling discussed in the subsequent section, no samples are wasted as each sample falls in the state space. Nevertheless, SD has not been shown to be asymptotically optimal in a limiting regime, its memory requirements are substantial when the populations are large, and its performance, in terms of numerical tests, is inferior to that of IE when all FCFS stations have a single server [14]. For networks with multiple-server stations, new numerical results comparing decomposition sampling with rejection sampling are given in the subsequent section.

A variation of this scheme is to replace the ρ_{jm}'s appearing in the sampling function $p(\boldsymbol{n})$ with some other values of γ_{jm}'s. Our computational testing has indicated that this variation can often reduce the confidence intervals if the γ_{jm}'s are chosen appropriately. Unfortunately, we have not yet designed a heuristic for choosing the γ_{jm}'s that works well for a wide variety of networks. When the γ_{jm}'s are chosen by trial and error, we refer to this variation as optimal SD. MonteQueue 2.0 allows the user either to select the nominal parameters – that is, $\gamma_{jm} = \rho_{jm}$ for all j, m – or to specify his own γ_{jm}'s.

6. Summation with rejection sampling

Summation with rejection (SR) sampling, another importance sampling technique for Monte Carlo summation, is presented in the appendix of [14]. Neither theoretical justification nor implementation issues or computational testing for summation with rejection is presented in [14]. In this section we show that

rejection sampling is asymptotically optimal for a sequence of networks in normal usage. We also present new computational results comparing rejection sampling with IE and decomposition sampling.

The summation technique discussed in this section is based on a minor modification of the summation representation of the normalization constant, which we describe now. Redefine $\boldsymbol{n} = (n_{jm}: 1 \le j \le J, m = 1, \ldots, L)$ and $\Lambda = \{0, 1, \ldots\}^{JL}$ (that is, neglect the components associated with the IS stations). Let

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

and

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

A simple argument [14] gives the following modification for the summation representation of the normalization constant:

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

where

$$c = \prod_{j=1}^{J} \frac{\rho_{j0}^{N_j}}{N_j!}.$$

We consider applying Monte Carlo summation to this representation of the normalization constant.

A good importance sampling function will have a shape resembling that of the summand. Motivated by the above expression for g, let γ_{jm}, $1 \le j \le J$, $1 \le m \le L$, be nonnegative constants satisfying

$$\gamma_m = \gamma_{1m} + \cdots + \gamma_{Jm} < s_m$$

for $m = 1, \ldots, L$. Consider the density

$$p\gamma(\boldsymbol{n}) = \prod_{m=1}^{L} \frac{f_m(n_m) \gamma_m^{n_m}}{n_m! \tau_m} \frac{n_m!}{n_{1m}! \cdots n_{Jm}!} \left(\frac{\gamma_{1m}}{\gamma_m} \right)^{n_{1m}} \cdots \left(\frac{\gamma_{Jm}}{\gamma_m} \right)^{n_{Jm}}, \quad \boldsymbol{n} \in \Lambda,$$

where

$$\gamma = (\gamma_{jm}, 1 \le j \le J, 1 \le m \le L) \quad \text{and} \quad \tau_m = \sum_{n=0}^{\infty} \frac{f_m(n)}{n!} \gamma_m^n.$$

With this sampling distribution, our one-sample estimate becomes

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

where $c' = c\tau_1 \cdots \tau_M$. We refer to this technique as *rejection* due to the presence of the $1(\boldsymbol{V}^i \in \Omega')$ term in the above expression.

6.1. Asymptotic optimality

Recall the asymptotic regime discussed in Section 2.4. Let $\tilde{p}(\cdot)$ denote the sampling function $p_{\gamma}(\cdot)$ with $\gamma_{jm} = \beta_j \Gamma_{jm}$, $1 \le j \le J$, $1 \le m \le L$. For the Nth network write $\Omega'(N)$ for Ω'.

Theorem 3. *Suppose the sequence of networks is in normal usage. Further suppose a sampling function $p(\boldsymbol{n})$, $\boldsymbol{n} \in \Lambda$, is used for each network in the sequence.*

(i) *If $p(\boldsymbol{n}) = \tilde{p}(\boldsymbol{n})$ for all $\boldsymbol{n} \in \Lambda$, then*

$$\lim_{N \to \infty} \overline{\text{var}(N)} = 0.$$

(ii) *If $p(\boldsymbol{n}) \neq \tilde{p}(\boldsymbol{n})$ for some $\boldsymbol{n} \in I^{JL}$, then*

$$\lim_{N \to \infty} \inf \overline{\text{var}(N)} > 0.$$

Proof.

(i) Suppose that $\gamma_{jm} = \beta_j \Gamma_{jm}$, $1 \le j \le J$, $1 \le m \le L$. Then for the Nth network, $Z^i = c'(N) X^i(N)$, where $c'(N)$ is the constant c' for the Nth network and

$$X^i(N) = 1(\boldsymbol{V}^i \in \Omega'(N)) \prod_{j=1}^{J} \phi(\beta_j N + \alpha_j \sqrt{N}, V_{j1}^i + \cdots + V_{jL}^i).$$

Note that $\overline{\text{var}(\text{N})}$ is equal to the relative variance of $X^i(N)$ divided by I. The result therefore follows from

$$X^i(N) \le 1,$$
$$\lim_{N \to \infty} X^i(N) = 1,$$

and the dominated convergence theorem.

(ii) Follows from Fatou's lemma (see the similar proof of Proposition 2 of Ross and Wang [16]). □

Thus, for estimating the normalization constant, rejection sampling is asymptotically optimal for Monte Carlo summation when the network is in normal usage. We note that Theorem 3 can be generalized to allow the sampling function $p(\boldsymbol{n})$ to depend on N with $p_N(\cdot)$ converging to $\tilde{p}(\cdot)$; the proof is longer but straightforward.

6.2. Implementation in MonteQueue 2.0

We can perform rejection sampling function as follows. For each m we select a V_m from the set $\{0, 1, \ldots\}$ according to the probabilities

$$P(V_m = n) = \frac{f_m(n) \gamma_m^n}{n! \tau_m}, \quad n = 0, 1, \ldots$$

For a fixed m, we then obtain $V_{1m}, \ldots, V_{Jm}$ by placing V_m balls in J boxes, with a ball placed in box j with probability γ_{jm}/γ_m. The balls can be placed in the box by running the alias algorithm over the set $\{1, \ldots, J\}$ with probabilities γ_{jm}/γ_m, $j = 1, \ldots, J$.

MonteQueue 2.0 actually implements a modified version of this sampling procedure. Let T_m denote the maximum number of customers that can be present at station m. The above sampling procedure rejects a sample whenever $V_m > T_m$. MonteQueue 2.0 eliminates this waste by sampling V_m over $\{0, 1, \ldots, T_m\}$ instead of over $\{0, 1, \ldots\}$. To implement this version, we must replace the ∞ appearing in the definition of τ_m with T_m and use the alias algorithm to sample over the finite set. The overall memory requirements of this modified procedure, $O(\sum_{m=1}^{L} T_m + JL)$, is not an obstacle. The overall expected computational effort is

$$O(E[V_1] + \cdots + E[V_L]).$$

It is not difficult to show that $E[V_m]$ is bounded by the expected number of customers in an $M/M/s_m$ queue with arrival rate γ_m and servers working at rate 1. Hence the computational effort of the sampling procedure is small if $\gamma_m \ll s_m$, $m = 1, \ldots, L$; the worst case expected effort is $O(\sum_{m=1}^{L} T_m)$, which is approached when all the γ_m's become large.

Theorem 3 compels us to set

$$\gamma_{jm} = \begin{cases} N_j \dfrac{\rho_{jm}}{\rho_{j0}} & \hat{y}_m < 0.9, \\ 0.9 \dfrac{N_j}{\hat{y}_m} \dfrac{\rho_{jm}}{\rho_{j0}} & \hat{y}_m \geq 0.9 \end{cases}$$

in MonteQueue 2.0. The heuristic for the case $\hat{y}_m \geq 0.9$ gives good results for a wide variety of networks. Note that it assures that $\gamma_m \leq 0.9 s_m$ for all $m = 1, \ldots, L$ and all traffic conditions. This heuristic also puts a limit on the computational effort, as discussed above.

6.3. Computational testing

Consider now the symmetric test network defined in Section 4.2, which has eight single-server stations, one IS station, 28 classes, 100 customers and $\mu_{j0} = 0.002$ for each class. Fig. 3 compares the performance of SR and IE for estimating throughput. It plots the normalized width of the estimate of throughput, TH_{jm}, as a function of the loading, $\hat{y}_m$. (The normalized width is again defined in terms of the maximum over all (j, m) pairs.) For a given value of μ_m, 10000 iterations are run for both IE and SR (thus, 15000 iterations are run for IE when $\hat{y}_m \geq 0.9$). The two methods consume a comparable amount of CPU time, which ranges from 50 to 80 s. From Fig. 3 we see that both techniques perform remarkably well, with IE performing slightly better at low loads, and significantly better above critical loading. In the light of these tests, and other tests not reported here, we recommend IE over SR for networks without multiple-server stations.

For a network with multiple-server stations, Fig. 4 compares the performance of SR and SD for estimating throughput. The test network is identical to our previous network, except that $s_m = 4$ for all eight FCFS stations, and $\mu_{j0} = 0.16$ and $N_j = 5$ for all 28 classes; thus

$$\hat{y}_m = \frac{1}{4} \frac{7 \times 5 \times 0.16}{\mu} = \frac{1.4}{\mu}.$$

For each method and loading, 10000 iterations are run. The CPU time for a run of SR ranges from 30 to 50 s; the CPU time for a run of SD is about 160 s.

From Fig. 4 we see that SR performs better than SD when the loadings are light to moderately heavy. But when critical loading is approached, SD and optimal SD perform better than SR. The reader should

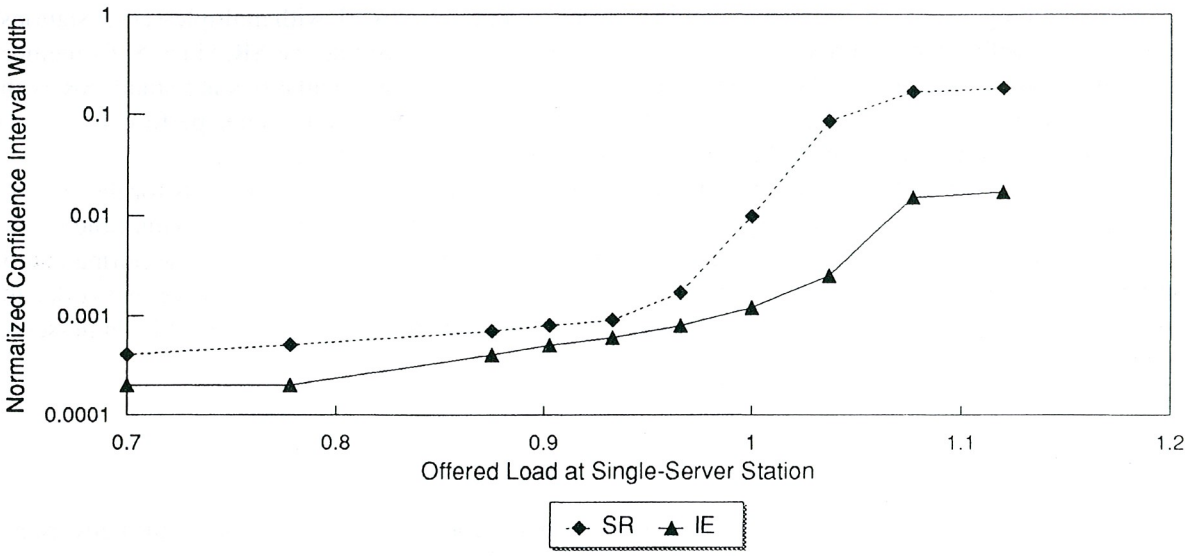


Fig. 3. The performance of SR and IE for estimating the throughput as a function of the offered load $\hat{y}_m$.

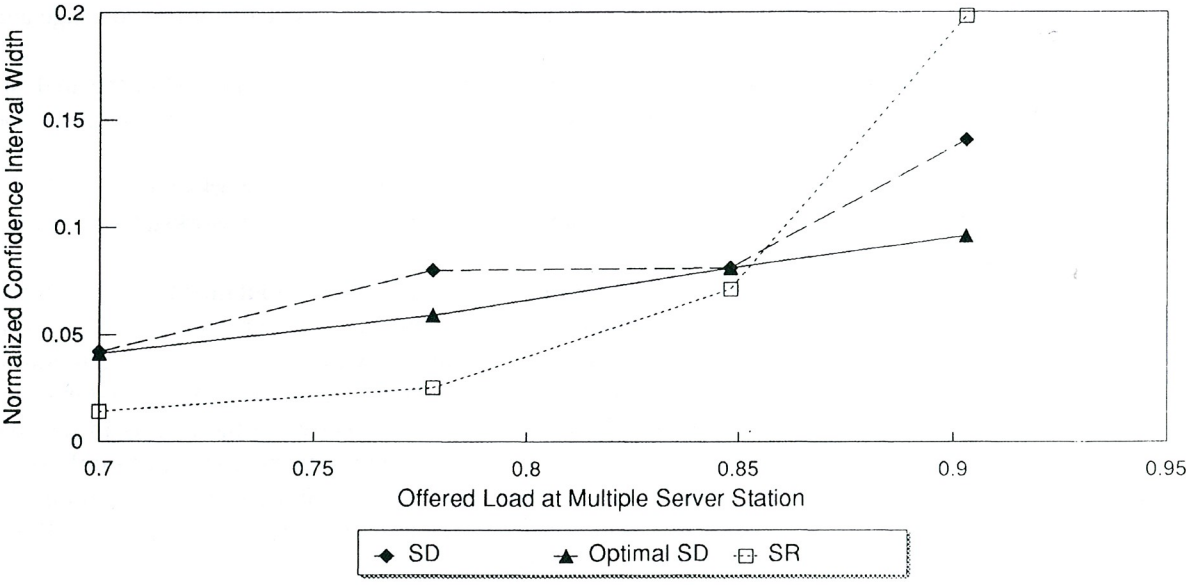


Fig. 4. The performance of SR and SD for estimating throughput as a function of the offered load, $\hat{y}_m$, for a network with multiple-server stations.

bear in mind, however, that the CPU time for SD is roughly three times that of SR for this test network. Furthermore, the optimal sampling parameters have to be found by trial and error, which requires additional CPU time. (The optimal parameters ranged from 0.95 to 1.0 for the loads from 0.7 to 0.9 in this example.)

Which sampling technique, SR or SD, should the user select for a network with multiple-server stations? For a network with a large number of customers the user's only choice may be SR, since SD's memory requirements grow quadratically with the population sizes. Even if the populations are small, SR is the preferred technique in normal usage. Nevertheless, SD can give substantially better performance if the populations are not large and the loadings at the FCFS stations are moderate to heavy.

The reader may have observed that for the same value of $\hat{y}_m$, SR's confidence intervals for the network with single servers and $N_j = 100$ are significantly narrower than those for the network with multiple servers and $N_j = 5$. This is not completely surprising since SR is asymptotically optimal (for the normalization constant) as the populations tend to infinity. If the $N_j's$ are set to 100 for this multiple-server networks, the width of the confidence intervals for SR are about the same as those for the networks with single-server stations.

7. Concluding remarks

MonteQueue 2.0 can quickly solve a wide variety of product-form queueing networks. If the network has no multiple-server stations, our numerical experiments have shown that IE (with the pilot runs introduced in this paper) is an excellent sampling technique for just about all network sizes and loads. (However, if the network is large, the routing is symmetric and the load is moderate-to-heavy, the user may want to try ITN; see Section 4.) If the network has multiple-server stations, then the user should choose (i) SR if the network is in normal usage (roughly, $\hat{y}_m < 0.9$), and (ii) SD if the network is in heavy or mixed usage and the populations are not too large.

The Monte Carlo techniques surveyed in this paper together form a powerful set of tools for the analysis of product-form queueing networks. Nevertheless, several important research problems in the area to be explored:

(1) *Parallel implementations of Monte Carlo summation and integration.* With K workstations available, one should be able to run $1/K$ iterations of Monte Carlo integration on each workstation, thereby obtaining a speedup close to K.

(2) *Comparison to discrete-event simulation.* Using the algorithms in [12] the computational effort required to generate an event for discrete-event simulation can be independent of the problem size. However, discrete-event simulation has a several disadvantages in comparison with the Monte Carlo techniques. First, the output observations are correlated so that many independent replications are needed in order to generate confidence intervals. Second, steady-state analysis is sensitive to the effects in the initial transient period, which may be very long. Third, variance reduction techniques have rarely been successful for queueing networks [7]. For these reasons, we feel that Monte Carlo summation and integration will generally have substantially better performance than discrete event simulation. But a fair and careful comparison between the Monte Carlo techniques and discrete-event simulation merits an in-depth study.

(3) *Gradient estimation.* Ross et al. [14] give estimates for gradients (that is, derivatives of confidence intervals); however, they report only limited success in finding importance sampling parameters that give small confidence intervals.

(4) *Heuristic for importance sampling parameters.* It is of interest to develop a good heuristic for choosing the sampling parameters for SD and ITN, and to further refine the heuristic for IE and SR for heavy usage (roughly, $\hat{y}_m \geq 0.9$).

References

[1] F. Baskett, M. Chandy, R. Muntz and J. Palacios, Open, closed and mixed networks of queues with different classes of customers, *J. ACM* **22** (1975) 248–260.

[2] J.A. Buzocott and J.G. Shantikumar, *Stochastic Models of Manufacturing Systems*, Prentice-Hall, Englewood Cliffs, NJ (1993).

[3] G.L. Choudhury, K.K. Leung and W. Whitt, Calculating normalization constants of closed queueing networks by numerically inverting their generating functions, *J. ACM* **42** (1995) 935–970.

[4] G.L. Choudhury, K.K. Leung and W. Whitt, Resource-sharing models with state-dependent arrivals of batches, *Proc. 2nd Internat. Workshop on Numer. Solutions of Markov Chains*, Rayleigh, N.C. 1995.

[5] E. Gelenbe and I. Mitrani, *Analysis and Synthesis of Computer Systems*, Academic Press, London (1980).

[6] P.W. Glynn and D.L. Iglehart, Simulation methods for queues: An overview, *Queueing Systems* **3** (1988) 221–256.

[7] P. Heidelberger and S.S. Lavenberg, Computer performance evaluation methodology, *IEEE Trans. Comput.* **C-33** (1984) 1195–1220.

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

[9] J. McKenna and D. Mitra, Integral representations and asymptotic expansions for closed Markovian queueing networks: Normal usage, *Bell Systems Tech. J.* **61** (1982) 661–683.

[10] D. Mitra, Asymptotically optmal design of congestion control for high speed data networks, *IEEE Trans. Commun.* **40** (1992) 301–311.

[11] D. Mitra and J. McKenna, Asymptotic expansions for closed Markovian queueing networks with state dependent service rates, *J. ACM* **33** (1986) 568–592.

[12] S. Rajasekaran and K.W. Ross, Fast algorithms for generating discrete random variates with changing distributions, *ACM Trans. Modeling and Comput. Simulation* **3** (1993) 1–19.

[13] K.G. Ramakrishnan and D. Mitra, An overview of PANACEA, a software package for analyzing Markovian queueing networks, *Bell Systems Tech. J.* **61** (1982) (568–592).

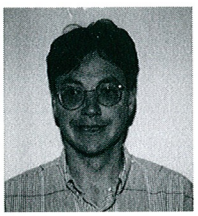
[14] K.W. Ross, D. Tsang and J. Wang, Monte Carlo summation and integration applied to multichain queueing networks, *J. ACM* **41** (1994) 1110–1135.

[15] K.W. Ross and J. Wang, Solving product form stochastic networks with Monte Carlo summation, *Proc. Winter Simulation Conf.*, 1990.

[16] K.W. Ross and J. Wang, Asymptotically optimal importance sampling for multichain queueing networks, *ACM Trans. Modeling and Computer Simulation* **3** (1993) 244–268.

[17] J. Wang and K.W. Ross, Asymptotic analysis for multiclass queueing networks in critical usage, *Queueing Systems Theory Appl.* **16** (1994) 167–191.

[18] W. Whitt, Heavy-traffic approximations for service systems with blocking, *Bell systems Tech. J.* **63** (1984) 689–708.



KeithW.Ross is an Associate Professor in the Department of Systems Engineering. He also holds secondary appointments in the Computer Information Science and in the Operations and Informations Management (Wharton) Departments. Keith Ross received his B.S. from Tufts University (1979), his M.S. from Columbia University (1981), and his Ph.D. from the University of Michigan (1985). He has been a visiting scholar at several research and academic institutions in France. He studies protocols and traffic management in high-speed telecommunication networks, including local area networks, wide area data networks, voice networks, and broadband integrated services digital networks. He is currently performing research in ATM and video on demand. He teaches courses on Internet technology and commerce, and on broadband networking. He also studies large-scale loss and queueing networks, with emphasis on developing computational procedures to predict the performance of large networks. He has published over 35 papers in leading journals and has completed a book on multiservice loss models for broadband telecommunication networks. Dr. Ross was the Program Chairman of the 1995 INFORMS Telecommunications Conference and has been on the program committees for INFOCOM and ITC for the past three years. He is an Associate Editor for Operations Research, for Probability in the Engineering and the Information Sciences, and for Telecommunication Systems. He is the recipient of numerous grants from NSF and AT&T, and has been consultant for major telecommunication companies.

JieWang is a Senior Technical Staff Member in the Department of Teletraffic Theory and Performance Analysis, AT&T Laboratories, AT&T. Jie Wang received his B.S. in mathematics from Peking University, China (1982), his M.S. in applied mathematics from University of Alberta, Canada (1989), and his Ph.D. in systems engineering from University of Pennsylvania (1993). After spending one year as postdoctoral fellow at the University of Pennsylvania, he joined AT&T Bell Laboratories in 1994. His main research interests include queueing networks, loss networks and stochastic modeling of ATM systems.