control nets. He has described process, message types which implement his algorithm in detail. Our approach departs from Nutt's in its basis in discrete-event simulation. As in sequential simulation there are cases where discrete-event approaches are preferable to time driven simulations and there are cases where the reverse is true.

The running time of the distributed algorithm depends upon the model being simulated. It is known empirically [15] that the distributed scheme approaches ideal performance when there are no multiple loops in the network. Extensive experimentation with various models is necessary in order to predict the performance of the proposed algorithm.

Received 2/80; revised 9/80; accepted 12/80

References

1. Bryant, R. E. Simulation of packet communication architecture computer systems. M.I.T. Lab. Comptr. Sci., M.S. Thesis, Nov. 1977.
2. Chandy, K. M., Holmes, V., and Misra, J. Distributed simulation of networks. *Comptr. Networks 3,* 1 (Feb. 1979), 105–113.
3. Chandy, K. M. and Misra, J. Distributed simulation: A case study in design and verification of distributed programs. *IEEE Trans. on Software Engineering, SE-5,* 5 (Sept. 1979), 440–452.
4. Chandy, K. M. and Misra, J. Deadlock absence proofs for networks of communicating processes. *Information Processing Lett. 9,* 4, (Nov. 1979), 185–189.
5. Chandy, K. M. and Misra, J. Termination detection of diffusing computations in communicating sequential processes. Dept. of Comptr. Sci., Tech. Rept, TR-144, 1980, University of Texas, Austin, TX.
6. Dijkstra, E. W. and Scholten, C. S. Termination detection for diffusing computations. EWD687a, 5671 AL Nuenen, The Netherlands.
7. Ellis, C. A. Information control nets: A methematical model of office information flow. *Proc 1979 Conf. on Simulation, Measurement and Modeling of Computer Systems.* (Aug. 1979), 225–239.
8. Hoare, C. A. R. Communicating sequential processes. *Comm. ACM 21,* 8, (Aug. 1978) 666–677.
9. Holmes, V. Parallel algorithms for multiple processor architectures. Ph.D. Dissertation, Comptr. Sci. Dept. Univ. of Texas, Austin, TX, 1978.
10. Nutt, G. J. An experimental distributed modeling system. Tech. Rept, Jan. 1980, Xerox Palo Alto Research Center, Palo Alto, CA 94305.
11. Peacock, J. K., Wong, J. W., and Manning, E. G. Distributed simulation using a network of processors. *Comptr Networks, 3,* 1 (Feb. 1979), 44–56.
12. Peacock, J. K., Wong, J. W., and Manning, E. G. Synchronization of distributed simulation using broadcast algorithms. *Proc of the Winter Simulation Conference,* December, 1979.
13. Peacock, J. K., Wong, J. W., and Manning, E. G. A distributed approach to queueing network simulation. *Proc. 4th Berkeley Conf. on Distributed Data Management and Computer Networks,* Berkeley, CA, August, 1979, 237–259.
14. Sauer, C. H. Characterization and simulation of generalized queueing networks. RC-6057, IBM Research, Yorktown Heights, NY, May, 1978.
15. Seethalakshmi, M. Performance analysis of distributed simulation. M.S. Rept, 1978, Comptr. Sci. Dept., Univ. of Texas, Austin, TX.

Simulation Modeling and Statistical Computing
N. Adam Guest Editor

Use of Polya Distributions in Approximate Solutions to Nonstationary $M/M/s$ Queues

Gordon M. Clark
The Ohio State University

Delays are important processes represented by continuous simulation models; however, representing queueing delays efficiently within continuous simulations merits the development of new methodology. Rothkopf and Oren introduced the concept of using a surrogate distribution, viz., the negative-binomial, as a closure approximation to the infinite set of Chapman-Kolmogorov equations representing a nonstationary $M/M/s$ queue. The method presented in this paper uses the Polya-Eggenberger distribution as a surrogate for the true distribution of the number in the queueing system at a particular time and only requires the numerical integration of five differential equations. The paper presents numerical results comparing the Polya surrogate and Rothkopf and Oren's approximation for a number of diverse cases, and these results indicate that the Polya surrogate is, in general, more accurate, although exceptions were encountered. Moreover, queueing delays represented by a closure approximation involving a surrogate distribution, in particular, the Polya, are suitable for use within a larger continuous simulation.

Key Words and Phrases: continuous simulation, queueing delays, $M/M/s$ queue, queueing approximation, system dynamics.

CR Categories: 5.5, 8.1,

Working Paper Series Number 1980-004

Author's Present Address: Gordon M. Clark, Industrial and Systems Engineering, The Ohio State University, Columbus, OH 43210.

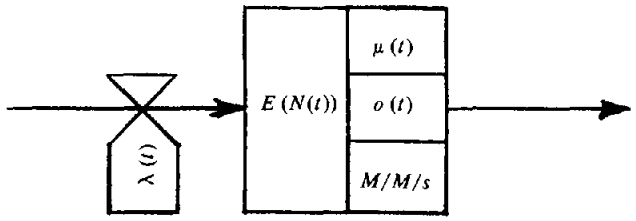
1. Introduction

System dynamics or continuous simulation models employ delays to represent processes requiring elapsed time before system quantities or entities change state [4]. Three processes frequently modeled as delays are

(1) Clerical processing by retailers or orders for more inventory prior to mailing the orders,
(2) Handling of orders received from retailers by a wholesaler prior to shipment, and
(3) Shipment of goods from a wholesaler to a retailer.

The basic modeling approach towards depicting delays in simulation languages such as DYNAMO [5] or SLAM [12] involves the use of an nth order exponential delay. If it is assumed that the delay process is equivalent to a queue with a time-dependent arrival process, an unlimited number of servers, and independent service times from the same negative exponential distribution [6], then the expected delay content is equivalent to a first order exponential delay. An nth order exponential delay describes a queue with an unlimited number of identical servers having nth order Erlang-distributed service times.

This paper presents a methodology for delineating $M/M/s$ queueing delays in dynamic continuous simulations. The most important characteristic of a queueing delay is the representation of a finite number of servers having a defined upper limit on their capacity to process entities. This methodology involves an approximation to the solution of a dynamic or nonstationary $M/M/s$ queue having arrivals from a time-dependent Poisson process, unlimited waiting space, s servers having service times from identical independent but possibly time-dependent Markov processes. The queueing delay is represented in a system-dynamics flow diagram by a symbol having the format



where

$N(t)$ = contents of the queueing delay at time t;
$E(N(t))$ = expected delay contents at time t;
$\mu(t)$ = single server service rate at time t;
$o(t)$ = expected delay output rate at time t; and
$\lambda(t)$ = delay input rate or arrival rate at time t.

The symbol $M/M/s$ implies an $M/M/s$ queueing delay. In addition to $E(N(t)$ and $o(t)$, the queueing delay model computes the expected queue waiting time for a new arrival at time t, i.e., $w(t)$, and the variance of $N(t)$, $V(N(t))$.

The lack of closed-form analytic solutions to dynamic queues complicates the continuous simulation of queues. Admittedly, expressions exist for the transient solutions to $M/M/1$ [8] and $M/M/1/k$ [11] queues[1]; however, these solutions are impractical for use in continuous simulations since they are very cumbersome to evaluate and are limited to single server queues.

Coupling the theory of Markov processes and numerical integration permits the solution or simulation of a much more comprehensive set of queueing situations. Kleinrock [8] describes the well-known procedure for constructing the Chapman–Kolmogorov equations for a constant-coefficient $M/M/s$ queue. These equations are simultaneous differential-difference equations involving the time–state probabilities,

$P_i(t)$ = probability the queueing system contains i entities at time t, i.e., $N(t) = i$.

These differential equations are

$$\begin{aligned} P_0'(t) &= -\lambda P_0(t) + \mu P_1(t), \\ P_i'(t) &= -(\lambda + i\mu)P_i(t) + (i+1)\mu P_{i+1}(t) \\ &\quad + \lambda P_{i-1}(t) \text{ for } i = 1, 2, \cdots, s-1 \\ P_i'(t) &= -(\lambda + s\mu)P_i(t) + s\mu P_{i+1}(t) + \lambda P_{i-1}(t) \\ &\quad \text{for } i = s, s+1, \cdots . \end{aligned} \tag{1}$$

To numerically integrate these equations, one approximates an unlimited waiting space by specifying that all values of $P_i(t)$ are zero for $i > k$. In essence, an $M/M/s/k$ queue is represented where k is large. This is accomplished by using Eq. (1) for $i < k$, and

$$\begin{aligned} P_k'(t) &= -\mu s P_k(t) + \lambda P_{k-1}(t) \\ P_i(t) &= 0 \text{ for } i > k. \end{aligned} \tag{2}$$

Equations (1) and (2) are readily extended to portray time-dependent arrival rates and/or service rates by regarding λ and μ as functions of time.

Numerical integration of the Chapman–Kolomogorov equations has been applied by Koopman [10] and Kolesar et al. [9]. However, this is a cumbersome approach for a continuous simulation because of the large number of equations integrated to represent congested queues, e.g., larger than 100 when the time average of $\lambda(t)/(s\mu(t))$ is 0.9 and $s = 1$. Also, the simulation may have several queueing delays along with other state variables (or levels) integrated numerically. Moreover, for each simulation run, the simulation user will have to verify whether k is sufficiently large to approximate an unlimited waiting space.

In an attempt to reduce the number of differential equations integrated numerically, Rider [13] and Chang [1], following an approach used earlier by Clarke [2], assumed a single server, multiplied each differential equation in Eq. (1) by i, and summed the resulting equations to obtain:

$$E'(N(t)) = \lambda(t) - \mu(t)(1 - P_0(t))$$

for an $M/M/1$ queue. Then they developed approximations for $P_0(t)$ and integrated $E'(N(t))$ numerically to approximate the expected queueing system contents.

[1] An $M/M/s/k$ queue is the same as an $M/M/s$ queue except that incoming arrivals balk and do not enter the queue when the contents of the queue plus the number being served equals k.

Rothkopf and Oren [14] extended this approach to represent multiple servers. They derived the following differential equations for an $M/M/s$ queue by the approach outlined above.

$$E'(N(t)) = \lambda(t) - \mu(t)s + \mu(t)\sum_{i=0}^{s-1}(s-i)P_i(t) \quad (3)$$

$$V'(N(t)) = \lambda(t) + \mu(t)s - \mu(t)\sum_{i=0}^{s-1}(2E(N(t)) + 1 - 2i)(s-i)P_i(t) \quad (4)$$

$V(N(t))$ = variance of $N(t)$

They approximated $P_i(t)$, $i = 0, 1, \cdots, s-1$, using a negative binomial distribution given values of $E(N(t))$ and $V(N(t))$. The fact that the geometric distribution is a special case of the negative binomial is one supporting reason for choosing the negative binomial. This implies that the approximation will converge to the true steady state distribution when there is only one server. To improve the approximation for the multiple server case, they published a table of constants that are a function of both the average value of $\lambda(t)/(s\mu(t))$ and s. Rothkopf and Oren found their approximation produced less error in approximating $E(N(t))$ than Rider's for a single server case. Moreover, Rothkopf and Oren tested their approximation using a variety of single and multiple server cases and published results indicating that their approximation produces errors of an acceptable magnitude.

2. Polya Surrogate Representation

The surrogate distribution approach to approximating dynamic queueing systems involves numerical integration of differential equations giving desired system moments. To illustrate the approach, consider the system moments $E(N(t))$ and $V(N(t))$ and their differential equations (3) and (4). The numerical integration is performed in time steps using a procedure such as the fourth-order Runge–Kutta procedure implemented in SLAM [12]. In SLAM, the simulation user must specify a subroutine that calculates derivatives of $E(N(t))$ and $V(N(t))$. All quantities in Eqs. (3) and (4) are known during the time step and available to this subroutine other than $P_i(t)$ for all i. Since the true values of $P_i(t)$ are unavailable, the user simply assumes a distribution as a surrogate for $P_i(t)$. The parameters for this surrogate distribution are calculated from the known moments, i.e., $E(N(t))$ and $V(N(t))$.

Of course, the choice of the surrogate distribution is crucial for developing an accurate approximation. Flexibility is a desirable property for the surrogate distribution because the distribution may be required to represent diverse values of the mean and variance. Thus, any limits imposed by the surrogate on possible combinations of mean and variance may be important. Also the distribution ought to accurately represent certain known special cases of the true distribution, i.e., the steady state distribution. However, the accuracy of the resulting approximation to desired queueing system quantities is the only real test.

The surrogate distribution approximation described in this paper differs from Rothkopf and Oren's approximation in two significant ways. First, the distribution of $N(t)$ is represented by two conditional distributions depending on whether $N(t)$ is larger than s, i.e., whether a queue exists. The desirability of this additional detail is suggested by the steady state distribution of an $M/M/s$ queue [11]. That is,

$$P_i = \begin{cases} P_0(\lambda/\mu)^i/i! & \text{for } 0 \le i \le s \\ P_0(\lambda/\mu)^i/(s^{i-s}s!) & \text{for } i \ge s. \end{cases} \quad (5)$$

Note the change in form of the steady state distribution once a queue exists. The second significant difference is the use of the Polya–Eggenberger distribution [7] hereafter referred to as simply the Polya distribution rather than the negative binomial. Accordingly, let the conditional random variables A and B represent $N(t)$, i.e.,

$A = N(t)$ given that there is no queue, i.e., $N(t) \le s$ and
$B = N(t) - s - 1$ given $N(t) > s$.

Also, A and B both have conditional Polya distributions with their own unique parameter values.

The probability a Polya random variable X assumes the value i is

$$P_i = \binom{n}{i} \frac{\left(\prod_{j=0}^{i-1} p + j\alpha\right)\left(\prod_{k=0}^{n-i-1} q + k\alpha\right)}{\left(\prod_{a=0}^{n-1} 1 + a\alpha\right)},$$

where $i = 0, 1, 2, \ldots, n$

$0 \le p \le 1$, $q = 1 - p$, and

$\alpha > -(\min(p, q))/(n-1)$.

Note that the binomial distribution is a special case of the Polya when $\alpha = 0$. The first two moments of the Polya are

$$E(X) = np$$
$$E(X^2) = np(np + q + n\alpha)/(1 + \alpha).$$

Also, the variance is

$$V(x) = npq\,(1 + n\alpha)/(1 + \alpha).$$

Observe that the variance of a Polya random variable can be less than the mean which is not possible for a negative binomial random variable. This property is important only for queues having more than one server. Rothkopf and Oren have successfully applied their approximation to $M/M/1$ queues where $V(N(0)) < E(N(0))$.

A random variable X having a Polya distribution can assume values on a finite set, i.e.,

$X = 0, 1, 2, \cdots n$;

however, under conditions specified in [7] the negative binomial distribution is a limiting distribution for the

Polya as $n \to \infty$. That is, let $n \to \infty$, $p \to 0$, $\alpha \to 0$ in a manner such that np and $n\alpha$ converge to the nonzero quantities θ and δ, respectively. Then X has a negative binomial distribution and

$$\theta = E(X)$$
$$\delta = V(X)/E(X) - 1.$$

To apply the Polya, the user must solve for the Polya parameters, given the first two moments. Thus,

$$p = E(X)/n. \tag{6}$$
$$\alpha = (E(X)(E(X) + q) - E(X^2))/(E(X^2) - E(X)n). \tag{7}$$

If $\alpha \leq -(\min(p, q))/(n - 1)$, then α is set to

$$\alpha = -(\min(p, q))/(n - 1) + 0.0001.$$

However, the conditional random variables A and B represent $N(t)$ so their moments are used in Eqs. (6) and (7).

Let

$Q(t)$ = Probability a queue exists at time t,
$\bar{Q}(t) = 1 - Q(t)$,

$$\bar{Q}(t) = \sum_{i=0}^{s} P_i(t), \tag{8}$$
$$C(t) = \sum_{i=0}^{s} iP_i(t) + sQ(t), \tag{9}$$
$$D(t) = \sum_{i=0}^{s} i^2P_i(t) + s^2Q(t), \tag{10}$$

then

$$E(A) = (C(t) - sQ(t))/\bar{Q}(t), \tag{11}$$
$$E(A^2) = (D(t) - s^2Q(t))/\bar{Q}(t), \tag{12}$$
$$E(B) = (E(N(t)) - C(t) + sQ(t))/Q(t) - (s + 1), \tag{13}$$
$$E(B^2) = (E(N^2(t)) - D(t) + s^2Q(t))/Q(t) - 2(s + 1)E(B) + (s + 1)^2. \tag{14}$$

The surrogate representation requires four Polya parameters:

p_a, α_a = Polya parameters for A
p_b, α_b = Polya parameters for B.

Substitution of Eqs. (11) and (12) into Eqs. (6) and (7) gives p_a and α_a. Similarly, substitution of Eqs. (13) and (14) into Eqs. (6) and (7) gives p_b and α_b.

The surrogate representation requires values for $E(N(t))$, $E(N^2(t))$, $C(t)$, $D(t)$, and $\bar{Q}(t)$ during each time step in order to use Eqs. (11), (12), (13), and (14). These quantities are obtained by integration of differential equations. In particular, integration of Eq. (3) gives $E(N(t))$, but rewriting of Eq. (3) simplifies the computations considerably.

$$E'(N(t)) = \lambda(t) - \mu(t)C(t). \tag{15}$$

A differential equation for $E(N^2(t))$ results from summing each equation in Eq. (1) after multiplication by i^2:

$$E'(N^2(t)) = \lambda(t) + 2\lambda(t)E(N(t)) + \mu(t)C(t) - 2\mu(t)(D(t) + sE(N(t)) - sC(t)). \tag{16}$$

Table I. Values of P_s for Three-Server Case.

$\lambda/(s\mu)$	Steady State	Polya Approximation
0.1	0.00333	0.00269
0.5	0.134	0.130
0.7	0.225	0.221
0.9	0.309	0.306
0.95	0.328	0.325

Similarly,

$$C'(t) = \lambda(t)(\bar{Q}(t) - P_s(t)) - \mu(t)(C(t) - sQ(t)) \tag{17}$$
$$D'(t) = \lambda(t)(2C(t) - (2s + 1)(Q(t) + P_s(t)) + 1) + \mu(t)(C(t) - 2D(t) + (2s^2 - s)Q(t)) \tag{18}$$
$$\bar{Q}'(t) = \mu(t)sP_{s+1}(t) - \lambda(t)P_s(t). \tag{19}$$

Note that the above differential equations only require two probabilities from the Polya surrogate, $P_s(t)$ and $P_{s+1}(t)$.

A comparison between P_s computed by the Polya and P_s computed from the steady state solution for an $M/M/s$ queue reinforces the selection of the Polya as a surrogate. Table I presents the comparison for the three-server case. On a percentage basis, the accuracy is excellent until very low server utilization $\lambda/(s\mu)$ values are encountered.

For P_{s+1}, the surrogate representation could utilize the limiting case when $n \to \infty$, which gives the negative binomial distribution and exact values for P_{s+1} in steady state, at least. This is true because the random variable B has a geometric distribution under steady state conditions. However, tests of representing B with a negative binomial distribution give less accuracy than use of the Polya under conditions other than steady state. Since accuracy under nonstationary conditions is more important to a continuous simulation, the Polya is also used to approximate P_{s+1}. This is done by selecting a value for n as an upper limit on the random variable B given by

$$n = [b(E(B) + 1) + 0.5]$$

where $[y]$ is the greatest integer less than y and b is an empirically determined constant. To determine b, a sequence of runs were made where

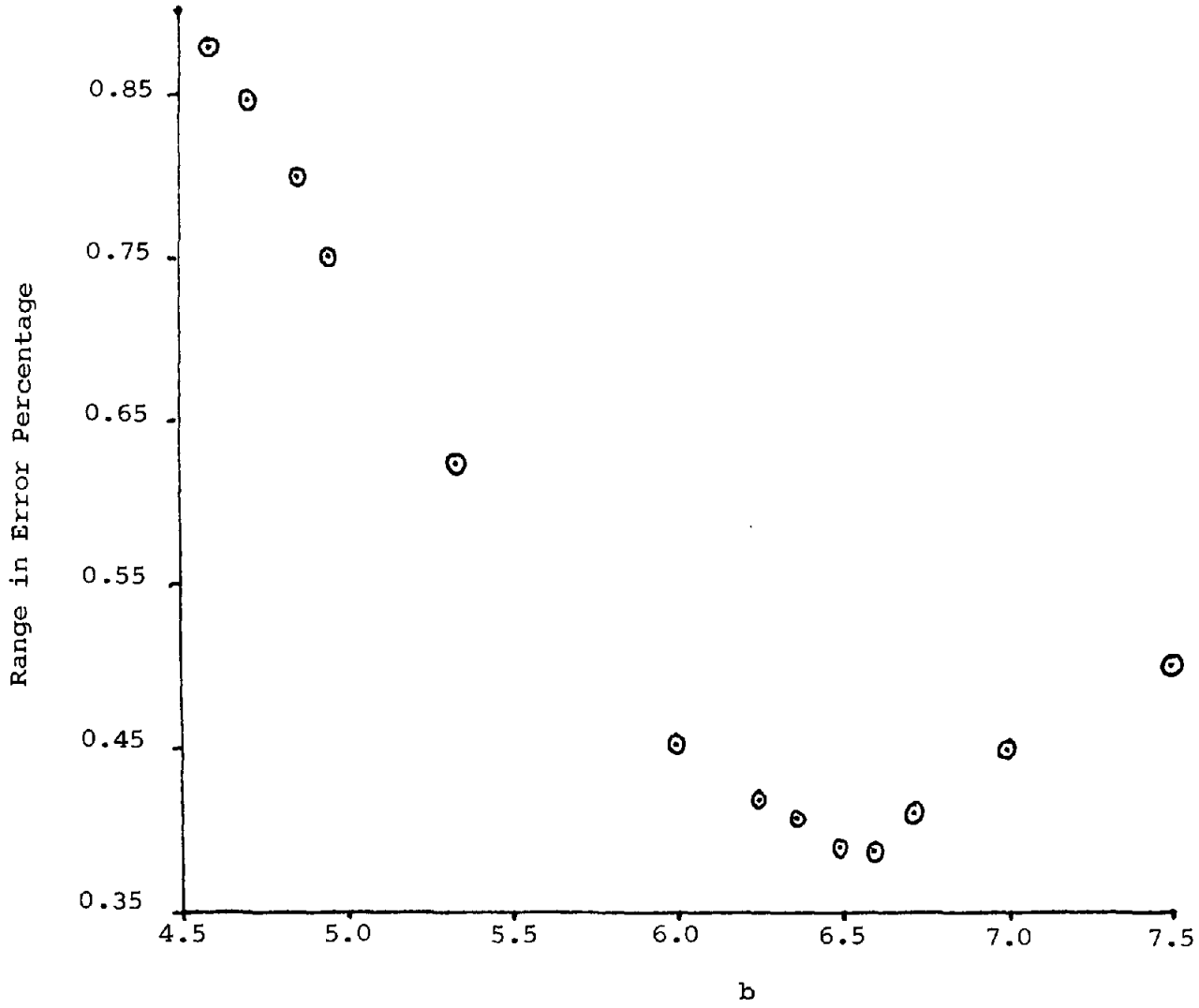
$$\lambda(t) = 0.7 + 0.25 \sin(2\pi t/25), \quad \mu(t) = 0.25, \; s = 3, \; 0 \leq t \leq 200.$$

At each integral value of time, a percentage error in approximating $E(N(t))$ was computed based upon use of results from integrating the Chapman–Kolmogorov equations (1) and (2). Figure 1 presents the range in error percentages, or difference between the minimum and maximum error percentages, as a function of b. Based upon these results, all other calculations for the Polya surrogate presented in this paper use a value of 6.6 for b.

Integration of Eqs. (15) through (19) automatically gives the mean system contents, i.e., $E(N(t))$. The variance of $N(t)$ is

$$V(N(t)) = E(N^2(t)) - (E(N(t))^2.$$

Fig. 1. Calibration of b.



The expected output rate is

$$o(t) = \mu(t)C(t).$$

In addition the expected waiting time for an arrival at time t is

$$w(t) = (P_s(t) + E(N(t)) - C(t) + Q(t))/(s\mu(t)).$$

3. Testing the Polya Surrogate

Results obtained by comparing approximations from the Polya surrogate with numerical values regarded as exact indicate its accuracy. Direct numerical integration of the Chapman–Kolmogorov equations provides results of sufficient accuracy to be regarded as exact. A double precision version of SLAM [12] implemented on an Amdahl 470 computer performed all numerical calculations. SLAM uses a fourth-order Runge–Kutta procedure with an automatic reduction of step size until the estimated truncation error on each step is within allowable limits. This truncation error bound is given by

$$|\mathbf{EERR}| \leq \mathbf{AAERR} + \mathbf{RRERR} \cdot |\mathbf{SS}|,$$

where **EERR** = estimate of truncation error derived by Fehlberg [3], **SS** = value of variable integrated, e.g., $P_i(t)$. When integrating the Kolmogorov equations, **AAERR** = 10^{-7}, **RRERR** = 10^{-6}. A comparison between numerical results by Runge–Kutta integration and the closed form transient solution [11] for an $M/M/1/k$ queue serves as a check on the numerical integration.

The cases compared are

λ	λ/μ	k	T
0.5	0.5	100	240
0.5	0.9	180	2400

Each run started in the empty and idle condition, and values of $E(N(t))$, $V(N(t))$, $w(t)$, and $o(t)$ were sampled and compared every 5 time units until time T. The absolute value of the error percentages expressed as a percentage of results computed by the transient solution never exceeded 2×10^{-5}.

Fig. 2. Starting from Exactly 9 in System.

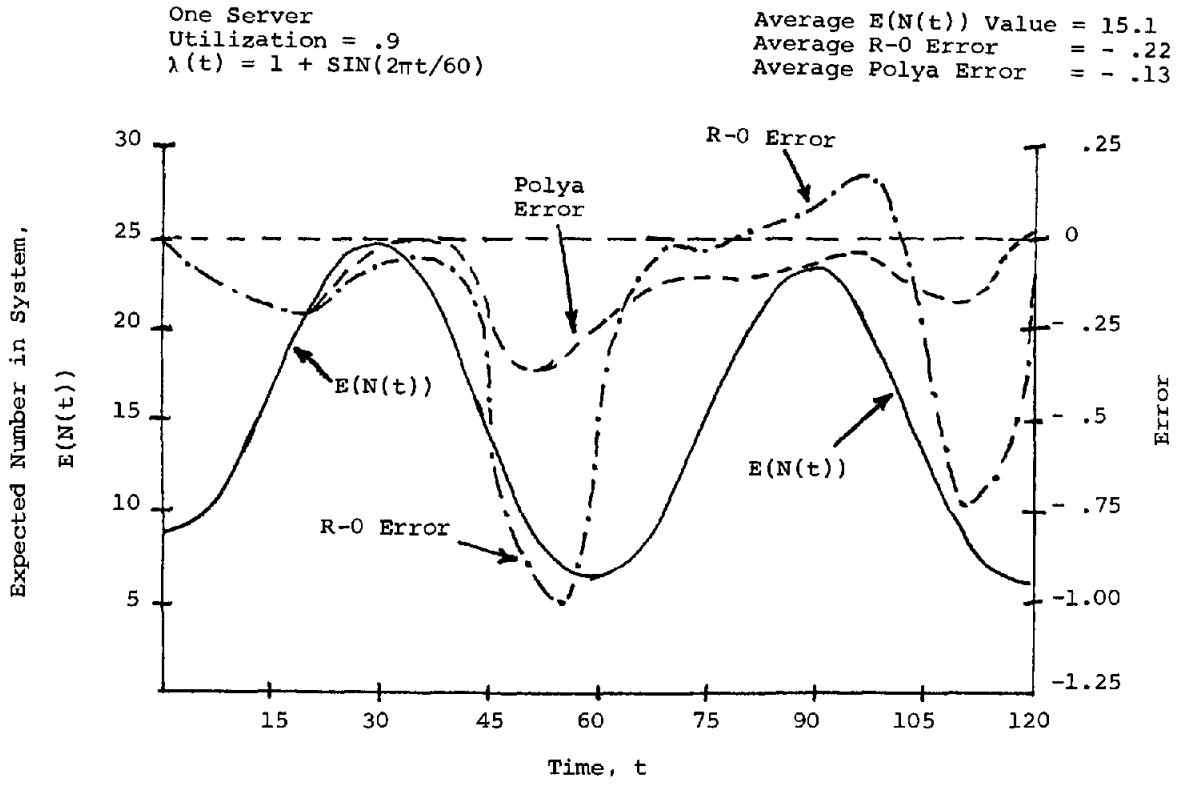


Fig. 3. Dynamic Low Utilization Single Server Comparison.

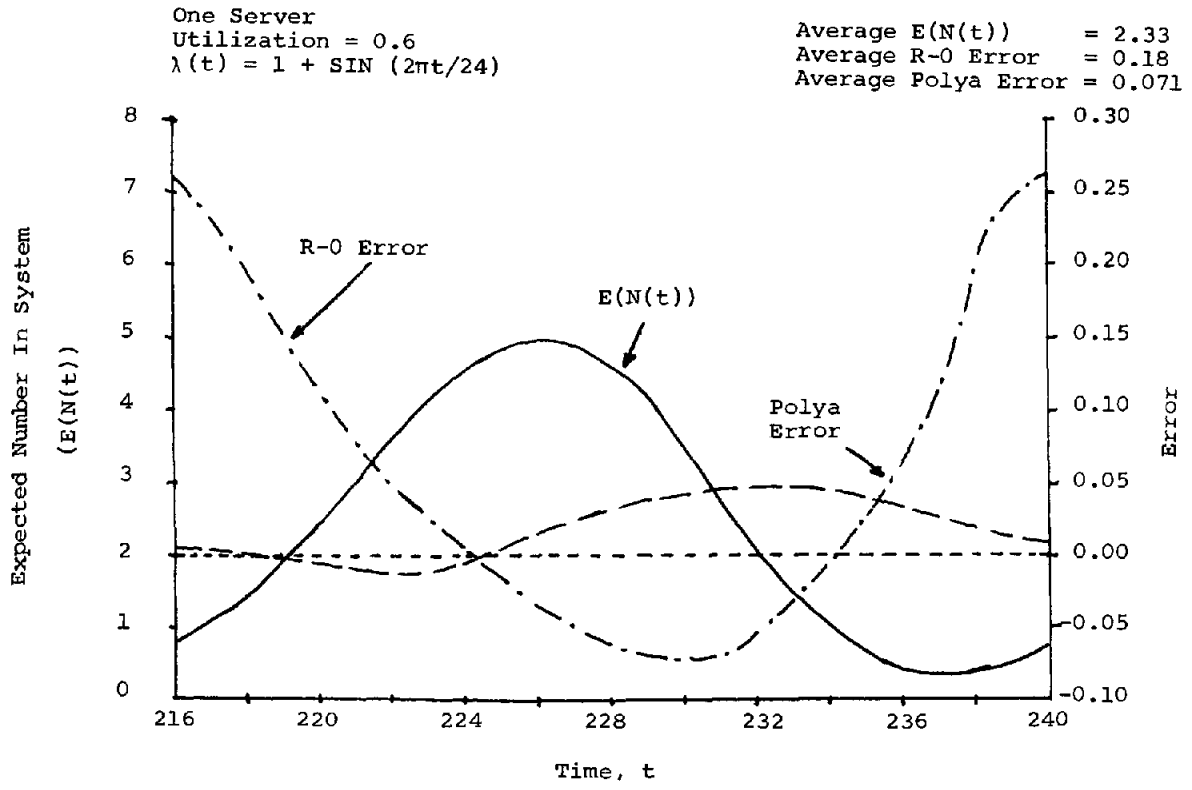
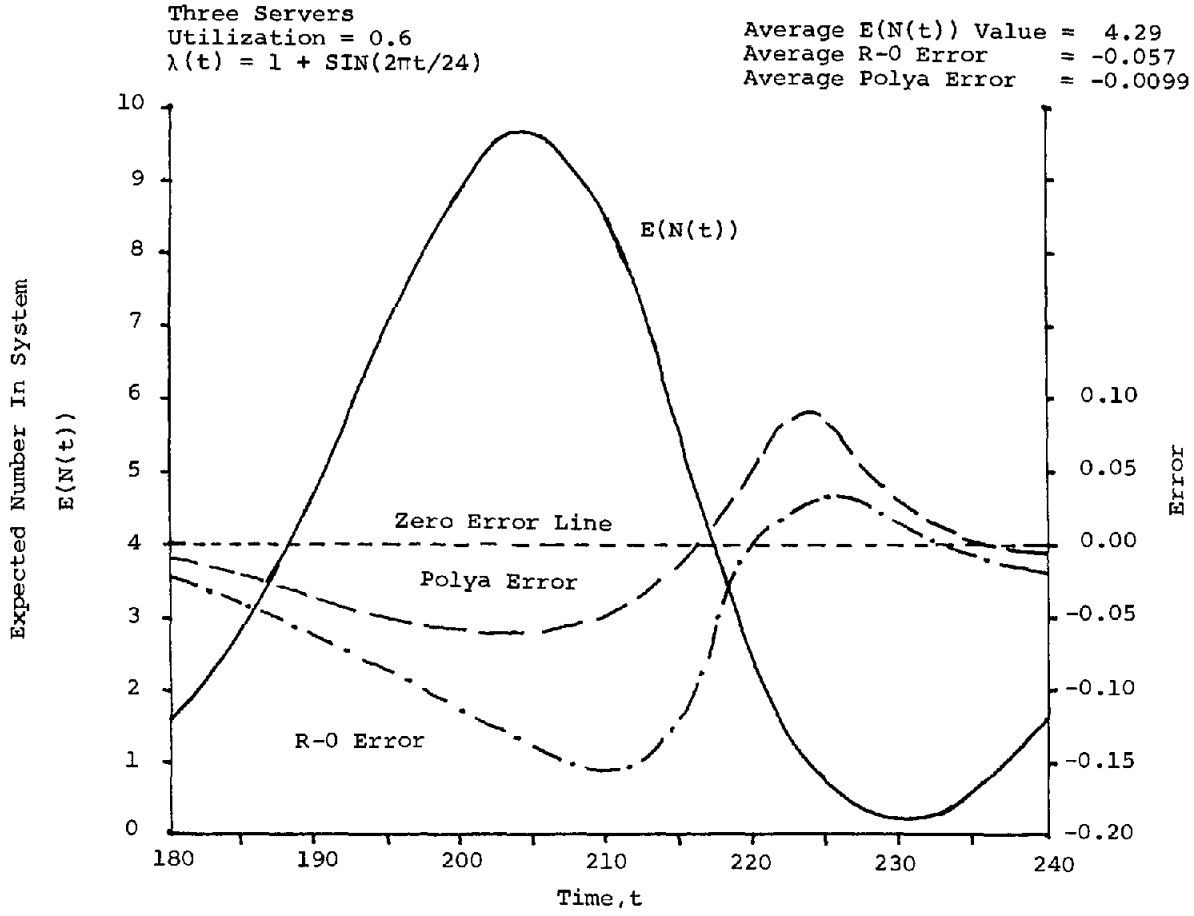


Fig. 4. Dynamic Low Utilization Multiple Server Comparison.



The use of a queueing system with capacity k is another source of error when integrating the Chapman–Kolmogorov equations. The value of k was varied from 100–190, depending on the average queue congestion, in order to control this error contribution. The potential error introduced by truncating an infinite queue can be visualized by realizing that the counting process defined by the number of rejections imposed by the finite queue is a time-dependent Poisson process with mean

$$r = \int_0^T \lambda(t) P_k(t)\, dt.$$

Thus, the probability an arrival is rejected at some time in the interval of duration T is approximately equal to r. Moreover, the error in $E(N(t))$ at a specified value of time would certainly be less than r. Since

$$V(Z) - V(X) \leq V(Y) + 2\sqrt{V(X)}\,\sqrt{V(Y)}$$

for all random variables where $Z = X + Y$, then the error in $V(N(t))$ at a specified point in time must be less than $r + 2\sqrt{r}\,\sqrt{V(N(t))}$. The value of r was never larger than 0.5×10^{-7}.

Figures 2–6 present the results from five different comparisons where the situations were taken from Rothkopf and Oren [14]. In each figure, the left vertical axis depicts the scale for the exact result computed by integrating Chapman–Kolmogorov equations. The right vertical axis gives the scale for the approximation error, i.e., approximation—exact. Of course, a large deviation from the zero error line by the approximation error is undesirable. The averages were computed by sampling the processes at each integral value of time. Figure 2 depicts a single server queue starting with exactly 9 in the system. Note that the average errors for both approximations are less than 1.5 percent of the time average for the expected number in the system. However, the Rothkopf and Oren error approaches 20 percent of $E(N(t))$ just prior to 60 time units. Figures 3–6 present results that are more representative of the limiting periodic condition. The period of the input arrival rate is 60 in Figures 4 and 5 and 24 in Figures 3 and 6. The performance of the approximations is compared subsequent to three cycles after the initial condition in Figures 4 and 5 and nine cycles in Figures 3 and 6. The initial condition was empty and idle for each figure other than Figure 5 which is a continuation of the situation depicted in Figure 2.

Figures 3 and 4 present dynamic results for the expected system contents for the one- and three-server cases, respectively. The average errors are less than 7.7 percent of the time average for $E(N(t))$ in each case. Figure 5 represents what Rothkopf and Oren term a worst case. Although the errors are as large as 11 percent of the exact values at some time during the interval

Fig. 5. Dynamic High Utilization Single Server Comparison.

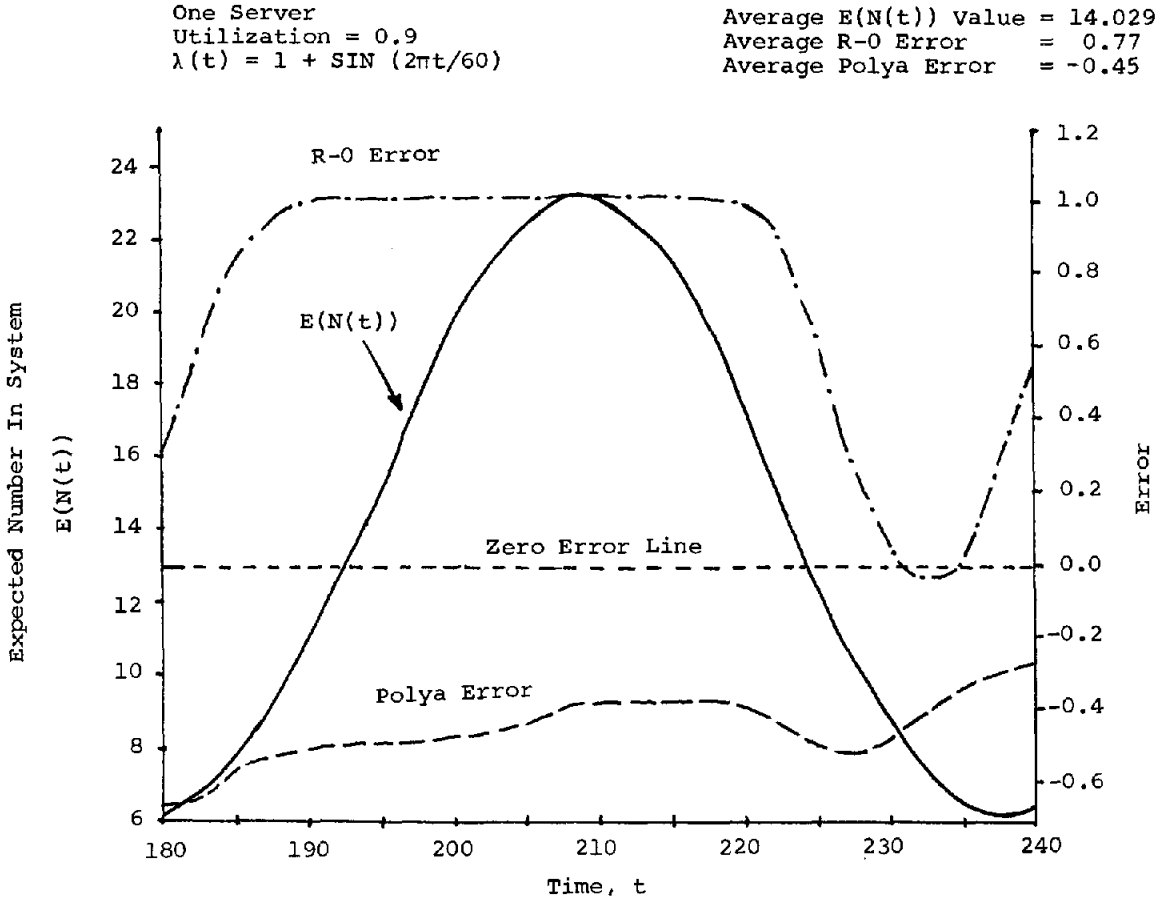


Fig. 6. Dynamic Variance Comparison.

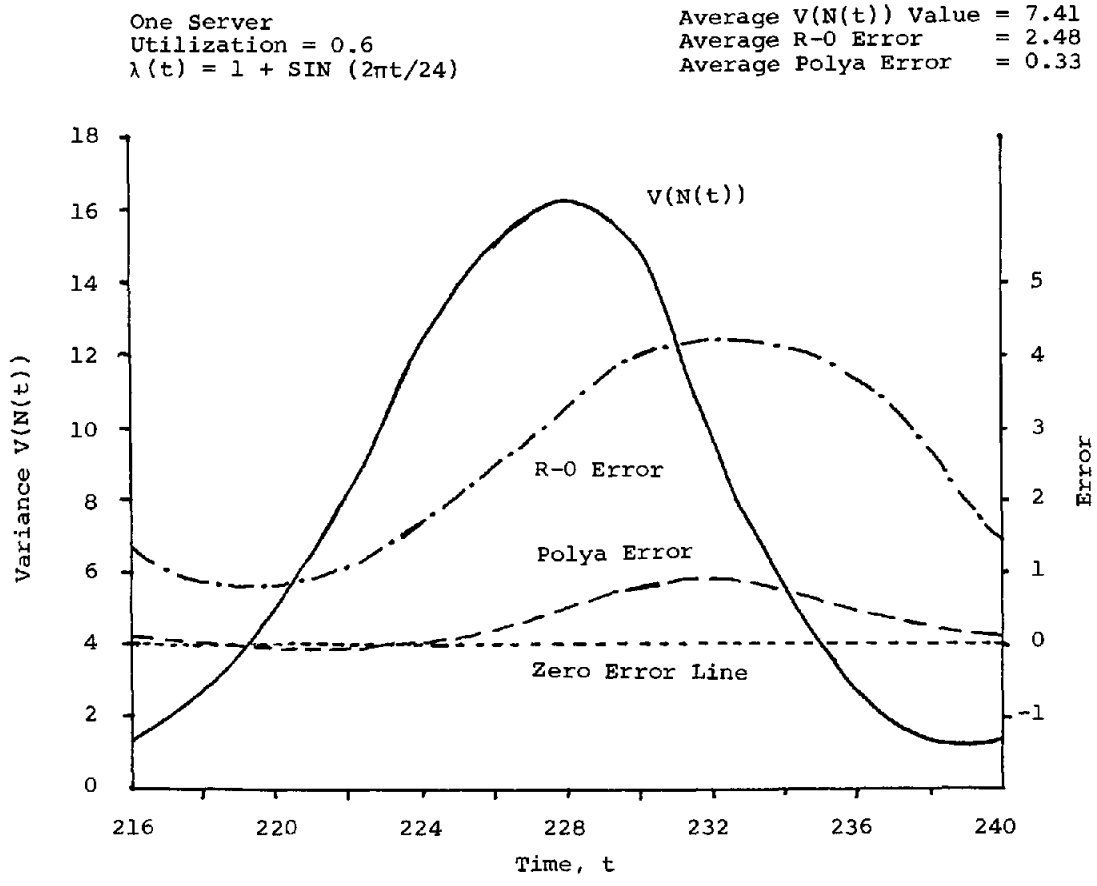


Table II. Test Case Specifications.

Case	s	ρ	a	T
1	1	0.5	0	240
2	1	0.5	0.25	240
3	1	0.9	0	2400
4	1	0.9	0.25	240
5	2	0.5	0	240
6	2	0.5	0.25	240
7	2	0.9	0	2400
8	2	0.9	0.25	240
9	3	0.5	0	240
10	3	0.5	0.25	240
11	3	0.9	0	2400
12	3	0.9	0.25	240
13	5	0.5	0	240
14	5	0.5	0.25	240
15	5	0.9	0	2400
16	5	0.9	0.25	240

portrayed, the average Rothkopf and Oren error is 5.4 percent of the average exact result while the Polya average error is 3.2 percent of the exact result. The results in Figure 5 depict a continuation of the conditions described in Figure 2, and this comparison illustrates the fact that the errors can increase for both approximations over time. Figure 6 illustrates another property of both approximations, viz., errors in approximating the variance exceed those for the mean on a percentage basis. For the Rothkopf and Oren approximation the average error in the variance is 33 percent of the time average for $V(N(t))$. The errors for the Polya approximation are much smaller for this comparison, but later results show that the second moment errors for the Polya tend to be larger than those for the first moment. Note than in these comparisons where the sinusoidal component of the arrival rate is very large, the Polya surrogate appears to be more accurate than the Rothkopf and Oren approximation.

Table II specifies 16 additional cases employed to test the approximations where the initial state for each case was always empty and idle. The input arrival rate was

$$\lambda(t) = 0.5 + a \cdot \sin(2\pi t/120)$$

where a was either 0 to specify a transient solution or 0.25 to specify a periodic solution. The simulated time T was either 240 time units for the periodic cases that represent two periods or long enough in the transient cases to permit $E(N(t))$ to achieve its steady state result within two significant figures. The single server service rate $\mu(t)$ was a constant in each case and chosen to represent either a 0.5 utilization factor or 0.9. The average utilization is

$$\rho = 0.5/s\mu.$$

At each integral value of time including the initial condition, the approximations were compared to the corresponding results from the Chapman-Kolmogorov equations integrations. Thus, at least 241 comparisons were made for each case, and the 0.9 utilization transient cases were subjected to 2401 comparisons. An error was computed for the output quantities: $E(N(t))$, $V(N(t))$, $w(t)$, and $o(t)$. Also a percentage error was calculated based upon a percentage of the Chapman-Kolmogorov results. Since each case started at the empty and idle condition this percentage error test could be severe. For example, a small error in a quantity such as $w(t)$ when t is small could result in a large percentage error. The performance measures tabulated are

average error,
error range = maximum error − minimum error,
error percent range = maximum percent error − minimum percent error.

Note that the minimum error was no greater than 0 since each case was started at time zero when the approximation had no error. Values for cases 13, 14, 15, and 16 are omitted for Rothkopf and Oren's approximation since they did not publish correction constants for more than three servers. Tables III, IV, V, and VI present results for the four queueing system performance measures.

Table III. Error and Percentage Ranges for $E(N(t))$.

		Polya Surrogate			Rothkopf and Oren		
Case	Average	Error Avg.	Error Range	Percent Range	Error Avg.	Error Range	Percent Range
1	0.98	0.00013	0.00095	0.096	0.22*E*-5	0.0062	0.75
2	1.2	0.0017	0.027	2.5	0.36*E*-3	0.079	6.2
3	8.3	0.047	0.10	1.4	−0.0023	0.18	3.3
4	7.3	0.0065	0.32	4.9	−0.063	0.95	18.
5	1.3	0.84*E*-4	0.00059	0.044	−0.030	0.032	2.4
6	1.5	0.0012	0.029	1.8	−0.026	0.060	6.7
7	8.8	0.045	0.094	1.1	−0.048	0.096	1.1
8	7.7	0.0024	0.27	3.4	0.0072	0.64	11.
9	1.7	−0.0029	0.0031	0.18	−0.037	0.040	2.3
10	1.9	−0.0014	0.029	1.8	−0.038	0.087	6.
11	9.4	0.039	0.086	0.97	−0.13	0.23	2.2
12	8.1	−0.0085	0.22	2.5	0.014	0.40	6.5
13	2.6	−0.0092	0.0099	0.38	—	—	—
14	2.8	−0.0068	0.030	0.99	—	—	—
15	11.	0.023	0.075	0.77	—	—	—
16	9.2	−0.033	0.10	1.2	—	—	—

Table IV. Error Averages and Ranges for $V(N(t))$

		Polya Surrogate			Rothkopf and Oren		
Case	Average	Error Avg.	Error Range	Percent Range	Error Avg.	Error Range	Percent Range
1	1.9	−0.026	0.030	1.5	0.0044	0.042	2.6
2	3.2	−0.060	0.16	10.	0.089	0.67	23.
3	75.	−8.0	13.	15.	2.7	5.5	11.
4	38.	3.4	8.1	18.	9.9	24.	48.
5	2.2	−0.017	0.020	0.90	0.021	0.19	31.
6	3.4	−0.045	0.10	6.4	−0.050	0.74	50.
7	76.	−7.5	12.	14.	2.8	10.	320.
8	38.	2.9	6.9	15.	15.	28.	310.
9	2.4	−0.024	0.027	1.1	0.022	0.23	32.
10	3.6	−0.044	0.091	4.5	−0.15	1.1	53.
11	76.	−7.2	12.	13.	4.3	14.	680.
12	38.	2.4	6.0	13.	19.	32.	671.
13	3.1	−0.055	0.059	1.8	—	—	—
14	4.2	−0.065	0.12	2.7	—	—	—
15	76.	−6.6	11.	12.	—	—	—
16	37.	1.7	4.2	10.	—	—	—

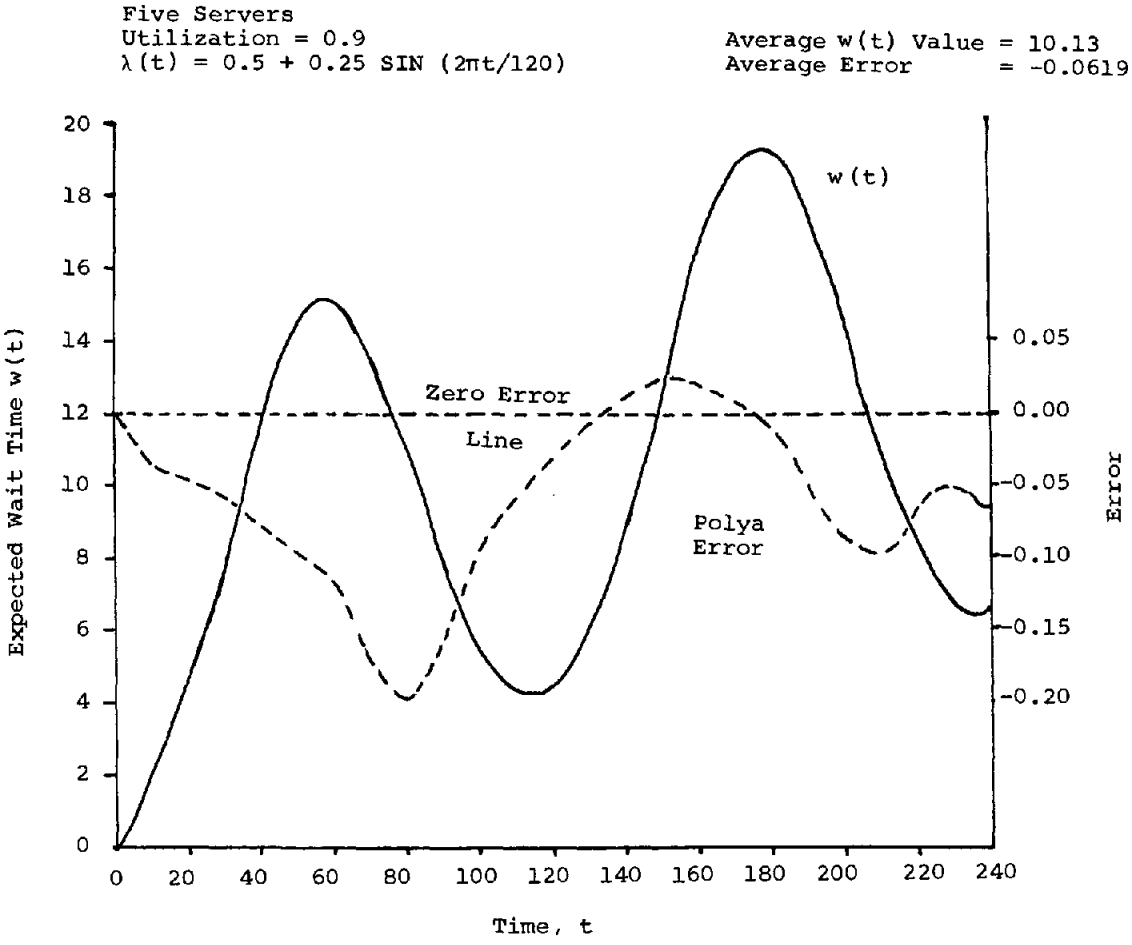
Table V. Error Averages and Ranges for $w(t)$.

		Polya Surrogate			Rothkopf and Oren		
Case	Average	Error Avg.	Error Range	Percent Range	Error Avg.	Error Range	Percent Range
1	0.98	0.00013	0.00095	0.096	0.18*E*-5	0.0062	0.75
2	1.2	0.0017	0.027	2.5	0.36*E*-3	0.079	6.2
3	15.	0.085	0.19	1.4	−0.0041	0.32	3.3
4	13.	0.012	0.58	4.9	−0.11	1.7	18.
5	0.65	0.83*E*-4	0.0009	0.09	−0.014	0.020	3.6
6	0.90	0.0011	0.025	3.2	−0.012	0.051	14.
7	1.5	−0.080	0.17	1.3	−0.066	0.17	2.0
8	12.	0.0047	0.48	3.9	0.039	1.2	14.
9	0.46	0.0060	0.0062	14.	−0.024	0.027	5.7
10	0.71	−0.0042	0.022	16.	−0.024	0.075	25.
11	13.	0.069	0.16	14.	−0.21	0.38	3.3
12	11.	−0.016	0.38	15.	0.064	0.84	11.
13	0.25	−0.019	0.020	62.	—	—	—
14	0.47	−0.015	0.026	64.	—	—	—
15	12.	0.038	0.14	65.	—	—	—
16	10.	−0.062	0.22	66.	—	—	—

Table VI. Error Averages and Ranges for $o(t)$.

		Polya Surrogate			Rothkopf and Oren		
Case	Average	Error Avg.	Error Range	Percent Range	Error Avg.	Error Range	Percent Range
1	0.49	0.19*E*-7	0.00013	0.027	0.13*E*-6	0.0020	0.50
2	0.50	0.14*E*-3	0.0054	1.5	0.38*E*-3	0.0090	2.4
3	0.89	−0.84*E*-5	0.0012	0.15	−0.68*E*-5	0.0081	1.2
4	0.86	−0.16*E*-3	0.033	4.	0.0017	0.11	15.
5	0.99	0.63*E*-7	0.00014	0.014	0.27*E*-3	0.0030	0.35
6	0.99	1.6*E*-3	0.010	1.3	0.40*E*-3	0.018	2.3
7	1.8	1.7*E*-4	0.0018	0.11	0.14*E*-3	0.0058	0.39
8	1.7	1.8*E*-3	0.055	3.5	−0.0023	0.17	11.
9	1.5	0.39*E*-4	0.00095	0.097	0.50*E*-3	0.005	0.36
10	1.5	0.32*E*-3	0.017	1.6	0.0012	0.030	2.3
11	2.7	0.15*E*-4	0.030	0.18	0.51*E*-3	0.0048	0.25
12	2.6	0.40*E*-3	0.075	3.2	−0.87*E*-3	0.19	8.1
13	2.4	0.2*E*-3	0.0030	0.15	—	—	—
14	2.4	0.97*E*-3	0.021	1.1	—	—	—
15	4.5	0.22*E*-4	0.0084	0.30	—	—	—
16	4.2	−0.002	0.088	2.2	—	—	—

Fig. 7. Dynamic Waiting Time Comparison.



From an overall viewpoint, these tables clearly indicate the superiority of the Polya surrogate approximation over the one developed by Rothkopf and Oren. However, in some instances their approximation did produce better performance measure values. Relatively speaking, Rothkopf and Oren's approximation performs best for the single server cases. Note that the largest percentage error in predicting the mean number in the system using the Polya surrogate is 4.9 over all cases. The percentage errors for $V(N(t))$ are larger and range up to 18 percent; however, they are much less for many cases. Table V presents some interesting comparisons for the expected waiting times. Although the absolute errors are less for the Polya approximation, some of the percentage errors are better using Rothkopf and Oren's approximation. Investigation of the detailed calculations shows that the largest percentage error produced by the Polya surrogate for $w(t)$ occurs when $t = 1$ and the true value is quite small. This large error quickly damps out percentagewise although its magnitude grows. Figure 7 shows this pattern for case 16. Also, the Polya surrogate performs well approximating the expected output rate $o(t)$. The largest percentage error shown in Table VI is 4.0 over all cases. A more detailed examination indicates that, on a percentage basis, the errors do not grow as servers are added, but they are larger with a sinusoidal input and increase with congestion.

Although the Polya surrogate is more accurate, it does require the integration of five equations as opposed to two required by Rothkopf and Oren's approximation. Table VII presents the cpu times in seconds for each case on an Amdahl 470. Note that the compute time used by the Polya surrogate is larger but not significantly so. On

Table VII. cpu Time (seconds).

Case	Kolmogorov	Polya Surrogate	Rothkopf and Oren
1	3.7	4.1	3.8
2	15.	4.8	4.2
3	45.	20.	16.
4	19.	4.4	3.9
5	3.6	3.9	4.
6	14.	4.6	4.2
7	45.	20.	16.
8	18.	4.5	3.9
9	3.8	3.9	3.8
10	14.	4.6	4.2
11	45.	20.	16.
12	17.	4.4	4.
13	4.	3.9	—
14	14.	4.6	—
15	48.	20.	—
16	17.	4.6	—

the other hand, the Polya surrogate is much less expensive to use than integrating the Chapman–Kolmogorov equations. In several cases, the cpu times is as little as 25 percent of the Chapman–Kolmogorov equation approach. Interestingly, the light traffic transient cases show a compute time by the Chapman–Kolmogorov equation approach that is about identical to the two approximations. This result occurs because the SLAM Runge–Kutta algorithm automatically adjusts the step size and takes fewer steps with the Chapman–Kolmogorov equations for those cases.

4. Conclusions

The Polya surrogate representation closely approximates the important output quantities desired from a nonstationary $M/M/s$ queue. Moreover, these results are obtained with a significant reduction in cpu time over direct integration of the Chapman–Kolmogorov equations. Also, the user is relieved of the responsibility for selecting a truncation point for the maximum queue length in approximating an unlimited waiting line capacity. Both of the above problems with direct integration of the Chapman–Kolmogorov equations are particularly bothersome when $\lambda(t) > s\mu(t)$ during rush periods. In addition, the core savings may be significant for some applications. Most fourth-order Runge–Kutta algorithms implemented in continuous simulations require from eight to eleven words of core storage per variable integrated. These economies make the Polya surrogate approximation attractive when representing queueing delays as part of a larger simulation.

The computational algorithm for the Polya surrogate uses the parameter b, and the basis for selecting its value involves a series of runs to find a value of b giving desirable error results for the case specified on p. 209. The possibility does exist for improving the performance of the Polya by recalibrating the approximation by employing adjusted values of b for different cases.

However, when analyzing a single station queue with multiple servers without involving a computer simulation, the savings indicated in this paper may not be worth the possible introduction of error, no matter how small. If performed with care, integration of the Chapman–Kolmogorov equations is very accurate.

Received 4/80; revised 10/80; accepted 12/80.

References

1. Chang, S.S.C. Simulation of transient and time varying conditions in queueing networks. Proc. Seventh Ann. Pittsburgh Conf. on Modeling and Simulation, University of Pittsburgh, PA. April 21–22 1977, 1075–1078.
2. Clarke, A.B. A waiting line process of Markov type. *Annals Math. Stat. 27*, 2 (June 1956), 452–459.
3. Fehlberg, E. Low-order classical Runge–Kutta formulas with step-size control and their application to some heat transfer problems. NASA Rept TR R-315, Huntsville, AL, April 15, 1969.
4. Forrester, J.W. *Industrial Dynamics.* M.I.T. Press, Cambridge, MA, 1961.
5. Forrester, J.W. *Principles of Systems.* Wright Allen, Cambridge, MA, 1971.
6. Gross, D. and Harris, C.M. *Fundamentals of Queueing Theory.* Wiley, New York, 1974, 114–117.
7. Johnson, N.L. and Kotz, S. *Discrete Distributions.* Houghton Mifflin, Boston, 1969.
8. Kleinrock, L. *Queueing Systems Volume 1: Theory.* Wiley, New York, 1975.
9. Kolesar, P.J., Rider, K.L., Crabill, T.B., and Walker, W.E. A queueing-linear programming approach to scheduling police patrol cars. *Operations Res. 23*, 6 (Nov–Dec. 1975), 1045–1062.
10. Koopman, B.O. Air terminal queues under time-dependent conditions. *Operations Res. 20*, 6 (Nov–Dec. 1972), pp. 1089–1114.
11. Morse, P.M. *Queues, Inventory, and Maintenance.* Wiley, New York, 1958.
12. Pritsker, A., Alan B., and Pegden, C.D. *Introduction to Simulation and SLAM.* Wiley, New York, 1979.
13. Rider, K.L. A simple approximation to the average queue size in the time-dependent $M/M/1$ queue. *J. ACM 23*, 2 (April 1976), 361–367.
14. Rothkopf, M.H. and Oren, S.S. A closure approximation for the nonstationary $M/M/s$ queue. *Management Sci. 25*, 6 (June 1979), 522–534.