K. M. Chandy
U. Herzog
L. Woo

Approximate Analysis of General Queuing Networks

Abstract: An approximate iterative technique for the analysis of complex queuing networks with general service times is presented. The technique is based on an application of Norton's theorem from electrical circuit theory to queuing networks which obey local balance. The technique determines approximations of the queue length and waiting time distributions for each queue in the network. Comparison of results obtained by the approximate method with simulated and exact results shows that the approximate method has reasonable accuracy.

Introduction

Queuing network models are being widely used in the analysis of computer systems and teleprocessing networks [1, 2]. Some efficient computational techniques for analyzing complex networks, limited to problems which satisfy local balance, have been described [3]. Efficient methods for the analysis of *specific* networks which do not satisfy local balance exist [4]. Approximating a *general* network by one which satisfies local balance results, usually, in unacceptable error for computer systems analysis.

Green and Tang [5] suggest that systems analysis techniques, which require modest computation time and provide results within accuracy limits of 10 to 20 percent, are adequate for the configuration phase of teleprocessing network design; subsequent phases may require more detailed, more accurate and more expensive techniques. The present work is concerned with the configuration phase.

This paper presents an approximate, iterative method for determining performance values of closed queuing networks with first come, first served discipline and general service times. The method is a direct application of Norton's theorem [6], and it gives exact solutions for networks which satisfy local balance. The method may be extended to networks with other disciplines and also to several classes of customers.

Norton's theorem for queuing networks

Consider a closed queuing network R with M queues indexed $1, 2, \cdots, M$ and with N customers (Fig. 1). The network is assumed to have only one class of customers. (Extension to several classes of customers is also presented in [6].) The service time for any queue may depend on the state of that queue (for instance, the number of customers in that queue) but is assumed to be independent of the rest of the network. A customer branches to queue j after service in queue i with probability p_{ij} independent of the state of the system, $i = 1, 2, \cdots, M$ and $j = 1, 2, \cdots, M$. The network is assumed to satisfy local balance [3, 6]. For example, a queue may have an exponential service time and a first come, first served discipline or a general service time with a rational Laplace transform and a last come, first served preemptive-resume or a processor-sharing discipline, etc.

Essential to Norton's theorem [6] is the exact equivalence of the complement of a queue. For any queue i in the given network, $i = 1, \cdots, M$, there exists a reduced two-queue network consisting of queue i and its "complementary queue" B_i (Fig. 2) such that the equilibrium queue length and wait time distributions of queue i in the reduced network are identical with those of queue i in the given network. We have shown in [6] that the complementary queue B_i can have a first come, first served discipline, an independent exponential service time and a service rate $r_i(n)$ that is a function of the number of customers n in queue B_i; the rate $r_i(n)$ is the conditional rate at which customers arrive at queue i of the given network R, given that there are $N - n$ customers in queue i (and n customers in the complementary queue B_i). The rate $r_i(n)$ is equal to the number of customers served per unit time in queue i of the given network when 1) the service time of queue i is identically zero and 2) there are n customers in the network, $n = 1, 2, \cdots, N$. The complement of a set of queues can be defined in the same manner.

We have shown how Norton's theorem can be applied in the parametric analysis of queuing networks. It is often required to study the behavior of a large network as

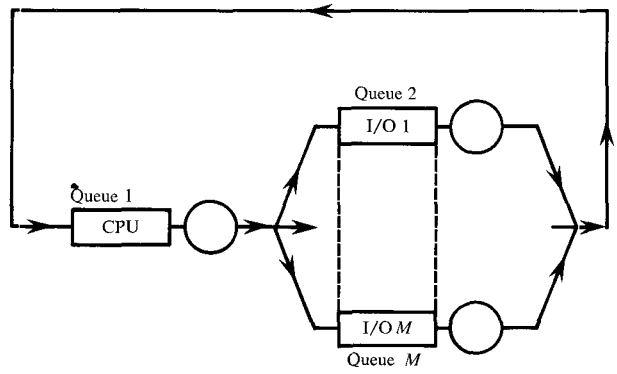


Figure 1 The central-server model: an example of a queuing network.

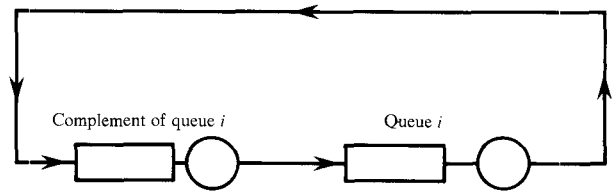


Figure 2 The reduced network of queue i.

the parameters of a single queue or a small subset of queues are varied over a wide range. It is much simpler to analyze the reduced network, consisting of the subsystem of queues and its complement, than to analyze the entire network as the subsystem parameters are varied over a range. For networks which satisfy local balance, Norton's theorem provides us with an exact representation of the complement and, therefore, an exact analysis.

Norton's theorem does not seem to be applicable to networks which do not satisfy local balance. In this paper, however, we describe a procedure for approximating complementary queues in networks which do not satisfy local balance.

Complementary algorithm for general networks

• *The principle*

The complement of a queue i in a network is a queue which completely captures the interface between queue i and the rest of the network. Because of complex interdependencies in general networks (which do not satisfy local balance), this complement cannot be determined exactly and must be approximated.

We approximate the complement of a queue in a general network by a queue with independent exponential service times; the service rate for this queue, however, is appropriately adjusted, as will be further shown, to account for the assumptions of independence and exponentiality. We refer to our approximation of the complementary queue as the *local balance interface* since it is computed by using assumptions of local balance. It is very likely that other interfaces with non-exponential service times will give even better results; however, this involves a great deal of additional computation.

The complementary algorithm is carried out in iterative steps, sequentially adjusting local balance interfaces to better approximate complementary queues. On the kth iteration of the algorithm, $k = 0, 1, 2, \cdots$, local balance interfaces are calculated using an auxiliary network $S^{(k)}$. The auxiliary network is identical to the actual network R except that

1. the service rates in $S^{(k)}$ are different and
2. $S^{(k)}$ is assumed to satisfy local balance.

It is therefore possible to use Norton's theorem to compute the complement of every queue in $S^{(k)}$. At the kth iteration we use the complement of queue i in $S^{(k)}$ to approximate the complement of queue i in R. In other words, the complement of queue i in $S^{(k)}$ is used as the local balance interface of queue i in R.

The queue length and wait time distributions of queue i in R are obtained by analyzing the two-queue (reduced) network consisting of queue i and its local balance interface. This two-queue network can be analyzed by the method of Herzog, Woo and Chandy [7] or Courtois and Georges [8]. The method suggested by Herzog, Woo and Chandy uses a decomposition technique to analyze Markovian systems with large, structured state spaces. It allows arbitrary service distributions with rational Laplace transforms. The method yields all steady state probabilities and can produce queue length and waiting time distributions. This technique also has application to models with different classes of customers and priorities. The technique developed by Courtois and Georges uses embedded Markov processes. It allows service distributions which may not have rational Laplace transforms. The technique yields the queue length distribution and mean waiting time, but not higher order moments of waiting time.

• *The algorithm*

Outline The outline of the complementary algorithm is presented here; details of the procedure are presented in following subsections.

Initialization

i) Set $k = 0$.

ii) Set the mean service times of $S^{(0)}$ to those of the given network R.

Step A

For each queue i in the network, $i = 1, \cdots, M$,

i) Determine the complement of queue i in $S^{(k)}$. Set the local balance interface of queue i in R equal to the complement of queue i in $S^{(k)}$.

ii) Determine queue length and waiting time distributions for queue i in R by analyzing the two-queue network consisting of queue i and its local balance interface.

Step B

Check whether the queue length and throughput for all M queues of R interface satisfactorily (for details see the following subsection). If they do, then stop; else go to step C.

Step C

Adjust the mean service times of queues in $S^{(k)}$ to obtain network $S^{(k+1)}$; set $k = k + 1$. The extent and direction of adjustment is determined from the degree and manner in which individual queue lengths and throughputs violate proper interface conditions (for details see the subsection on adjustment of mean service times). Go to step A.

An example illustrating the sequence of steps is shown in the Appendix. Clearly, the approximate solution is sensitive to the queue interface conditions checked in step B and the manner in which service times of $S^{(k)}$ are adjusted in step C. We shall now discuss steps B and C in detail.

The interface checks (step B) Several parameters can be checked to verify proper interface. We have chosen to check mean queue lengths and throughputs of each node. Let q_i be the queue length of queue i, and $\mathrm{E}(q_i)$ be its mean value. Let t_i be the throughput (number of customers served per unit time) of queue i. The two conditions checked are:

$$\sum_i \mathrm{E}(q_i) = N \tag{1}$$

and

$$\sum_i t_i p_{ij} = t_j, \quad j = 1, 2, \cdots, M. \tag{2}$$

The first condition checks that the sum of the mean queue lengths is equal to the number of customers in the system, and the second verifies that the throughput into a node is equal to the throughput out of a node.

Gordon and Newell [9] have defined a set of numbers y_i, $i = 1, \cdots, M$, such that

$$\sum_i y_i p_{ij} = y_j, \quad j = 1, 2, \cdots, M. \tag{3}$$

We find it convenient for the iterative procedure to define the normalized throughput t_i' for queue i as

$$t_i' = t_i / y_i. \tag{4}$$

Using normalized throughputs, Eq. (2) becomes

$$\sum_i t_i' y_i p_{ij} = t_j' y_j \quad \text{for } j = 1, \cdots, M. \tag{5}$$

The solution to these equations is obviously

$$t_1' = t_2' = \cdots = t_M'. \tag{6}$$

Equation (6) becomes the second interface condition, which states that all queues must have the same normalized throughput.

Adjustment of mean service times (step C) We will find it convenient to define an error tolerance ϵ which will be typically set to 0.01. If

$$\sum_i \mathrm{E}(q_i) > N(1+\epsilon), \tag{7}$$

there is said to be "excessive-queue-length" error, and if

$$\sum_i \mathrm{E}(q_i) < N(1-\epsilon), \tag{8}$$

there is "insufficient-queue-length" error. We will also define $t' = \Sigma_i t_i'/M$, where t_i' is the normalized throughput for queue i, computed by analysis of the reduced network consisting of queue i and its local balance interface. If $t_i' > t'(1+\epsilon)$, then queue i has "excessive-throughput" error. If $t_i' < t'(1-\epsilon)$, then queue i has "insufficient-throughput" error. The algorithm describing the adjustments in mean service times is presented next.

Step 1 If there is excessive-queue-length error then go to step 3a. If there is insufficient-queue-length error then go to step 3b. Otherwise proceed to step 2.

Step 2 Let $1/\mu_i$ be the mean service time of queue i in the network $S^{(k)}$ and let $1/\mu_i'$ be the mean service time of queue i in the modified network $S^{(k+1)}$. Set

$$\mu_i' = \mu_i t_i'/t' \quad \text{for } i = 1, \cdots, M.$$

Now if $\mu_i(1-\epsilon) < \mu_i' < \mu_i(1+\epsilon)$ for all $i = 1, \cdots, M$, then the algorithm stops; else start the next, $(k+1)$th, iteration.

Step 3a For all queues i with insufficient-throughput error, modify service rates in the following manner:

$$\mu_i' = \mu_i t_i'/t'.$$

If no queue has insufficient-throughput error, go to step 4.

The rationale of this modification is that, in the next iteration, the local balance interfaces of those queues without insufficient-throughput error (i.e., $\mu_i' = \mu_i$)

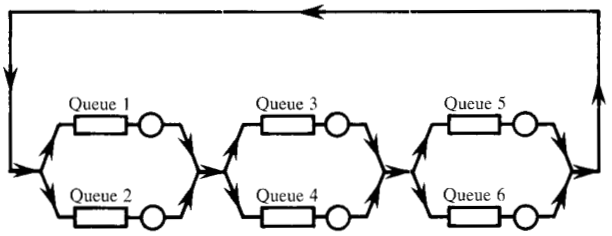
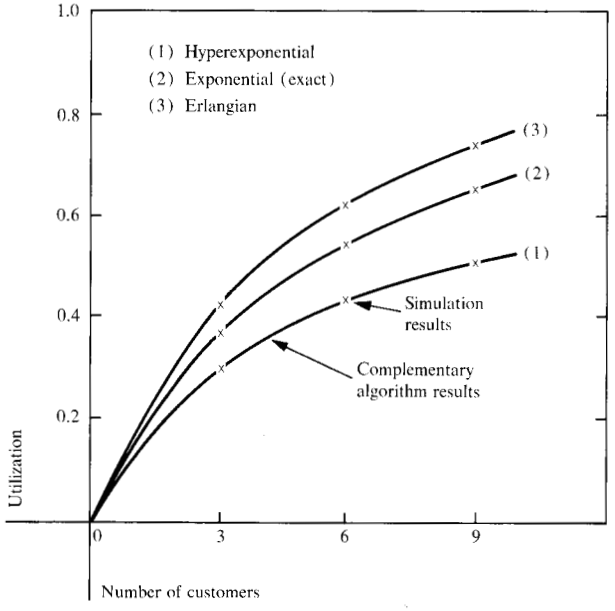


Figure 3 A six-queue example.

Figure 4 Comparison between simulation and complementary algorithm results: utilization for the central server.



will have slower input rates $r_i(n)$. In other words, in the reduced network analysis at the next iteration, the input rate $r_i(n)$ into queues without insufficient-throughput error will become smaller, thus reducing the mean queue length and throughputs of these queues.

Start the $(k+1)$th iteration with these modified rates.

Step 3b For all queues with excessive-throughput error modify service rates in the following manner:

$$\mu_i' = \mu_i t_i' / t'.$$

If no queue has excessive-throughput error, go to step 4.

Start the $(k+1)$th iteration with these modified rates. (The rationale for step 3b is the same as for step 3a.)

Step 4 The service rates for all queues are modified in the following manner:

$$\mu_i' = \mu_i N / \sum_j \mathrm{E}(q_j) \quad \text{for } i = 1, \cdots, M.$$

The effect of this step is to reduce all rates $r_i(n)$ if there is excessive-queue-length error, thus reducing mean queue lengths, and to increase all rates $r_i(n)$ if there is insufficient-queue-length error, thereby increasing mean queue lengths.

Start the $(k+1)$th step with the modified rates.

Validation

The algorithm was validated by comparing results obtained by the algorithm with those obtained by simulation and exact analysis. We first discuss the error "tolerance" of an algorithm.

• *Tolerance*

The error in a performance value is the difference between the exact value (obtained by Markov analysis or simulation) and the value obtained by the algorithm. The cycle time τ_i of a queue i is the mean time between successive arrivals of a particular customer at that queue, i.e., $\tau_i = N/t_i$, where t_i is the number of customers served per unit time in queue i of network R and N is the number of customers in the system. In validating our algorithm, we chose a set of criteria for assessing the accuracy of the approximation. The present objective is concerned with the configuration phase of teleprocessing networks and we feel that the following tolerance values are appropriate for ordinary ranges of M and N which are applicable to these networks.

The algorithm is said to be within a tolerance z for a given problem when

1. the utilization error does not exceed z for all queues in the network,
2. the error in mean and standard deviations of queue length for every queue in the network does not exceed zN, and
3. the error in mean and standard deviations of waiting time for queue i does not exceed $z\tau_i$ for $i = 1, \cdots, M$.

For our investigation, the accuracy is said to be good if the tolerance z is less than 0.05, and it is adequate if the tolerance is between 0.05 and 0.10.

• *Comparison with exact results*

Sixty cases were analyzed. The service times were hyperexponential (with a coefficient of variation of 2.13 or 3.10), second-order Erlang, or exponential. The number of customers varied from 2 to 5. The number of queues varied from 2 to 5, but 50 of the 60 cases had only two queues. Our validation shows that the approximation algorithm gives 57 good and three adequate results. In the three adequate cases, the utilization error is within 0.05, while the queue length error is slightly above $0.05N$.

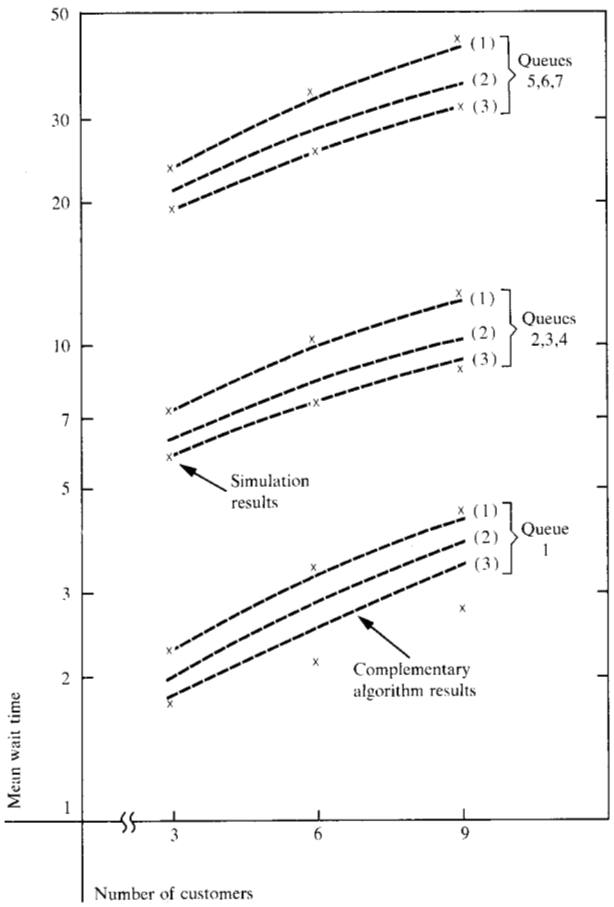


Figure 5 Comparison between simulation and complementary algorithm results: mean waiting time for all queues.

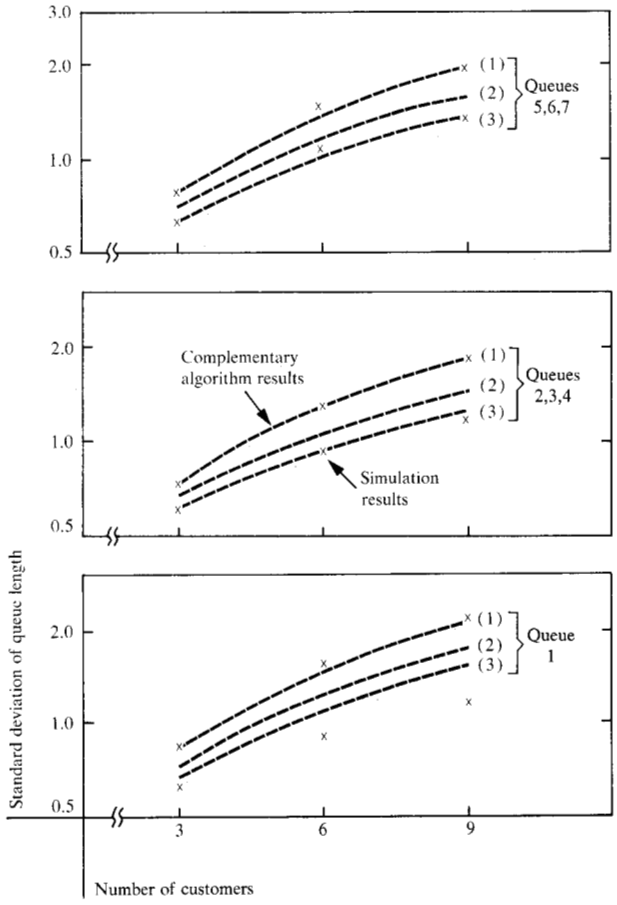


Figure 6 Comparison between simulation and complementary algorithm results: standard deviation of the queue length for all queues.

• *Comparison with simulation*

Eighteen cases of a seven-queue central server network (Fig. 1) and sixteen cases of a six-queue network (Fig. 3) were simulated. The number of customers was varied from two to twelve. Service times were picked from the following random variables: hyperexponential with a coefficient of variation of 2; Erlang of orders 2, 4 and 6; constant service; and exponential. Utilizations, mean and standard deviations of queue lengths, and waiting times were compared for each queue in the network. In every case the complementary algorithm gave *good* results. Some of the results are presented in figures 4 through 9. The results give the utilization, mean cycle time, mean and standard deviations of queue length, and waiting time in a central server model with seven queues. Figures 4 through 7 show results when all queues have 1) hyperexponential (coefficient of variation of 2), 2) exponential, and 3) sixth-order Erlangian service times. In addition, Figs. 8 and 9 present results when the central server (queue 1) has hyperexponential and other queues have sixth-order Erlangian service times. In all these cases, the mean service times for all queues are kept the same. When comparing the results of all these cases, it is observed that the utilizations, queue lengths, and waiting times are often quite different. Therefore, as was pointed out in the introduction, approximating a general network by an exponential network usually results in unacceptable errors.

The number of iterations of our algorithm for these problems varies from 2 to 8. The average number of iterations is less than four.

Conclusion

An approximate iterative technique for the analysis of queuing networks has been presented. This technique is an extension of the previously reported application [6] of Norton's theorem to queuing networks that do not necessarily satisfy local balance. The algorithm is adequate for the configuration phase of computer and teleprocessing system design, is general, and is limited only by the kinds of reduced networks that are analyzable. At the present time the algorithm has been programmed and

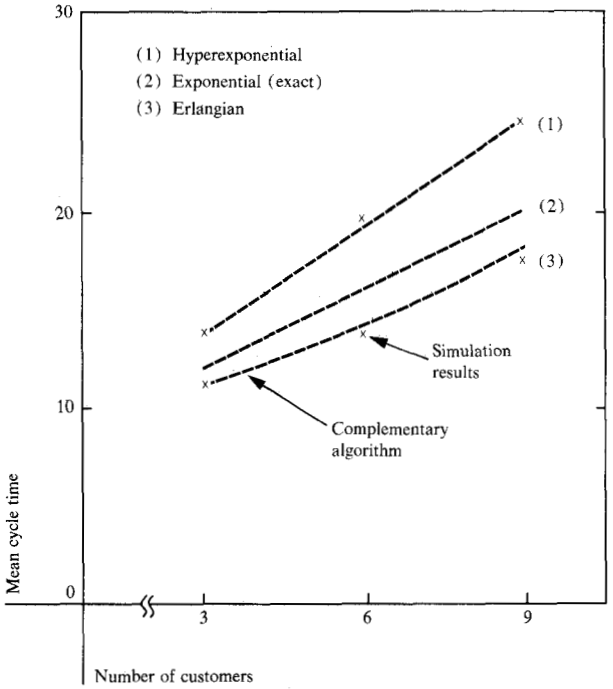
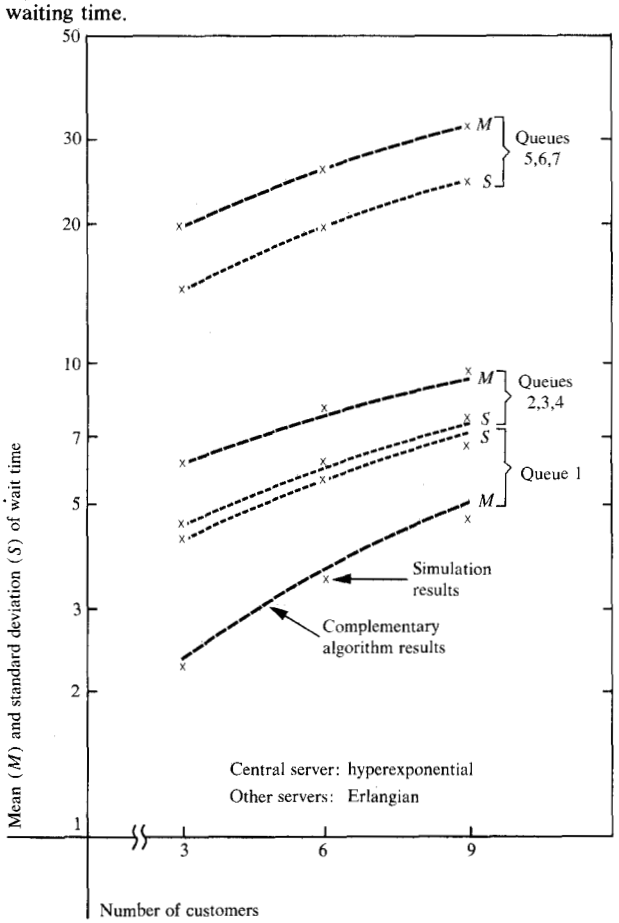


Figure 7 Comparison between simulation and complementary algorithm results: mean cycle time.

Figure 8 Comparison between simulation and complementary algorithm results: mean (M) and standard deviation (S) of the waiting time.



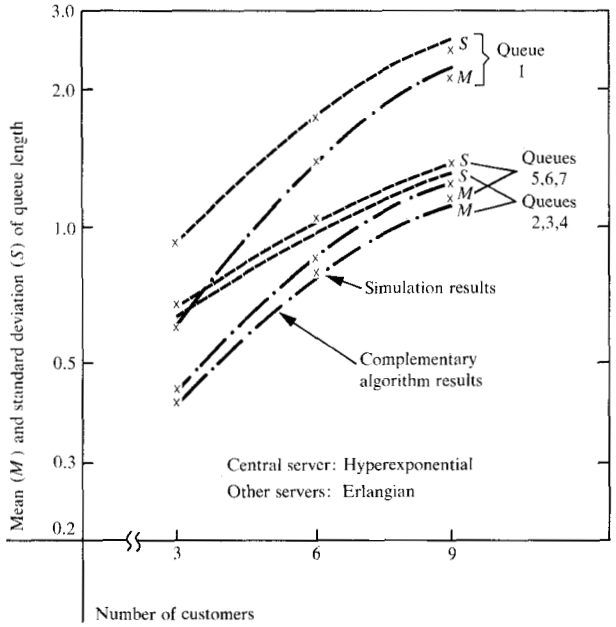


Figure 9 Comparison between simulation and complementary algorithm results: mean (M) and standard deviation (S) of the queue length for all queues.

validated to handle general networks with general service times and the first come, first served discipline. The algorithm has also been programmed to handle two classes of customers, where each class may have different branching probabilities, service rates and priorities; however, the program assumes (at this time) that service times for a given class are independent exponential random variables. The service disciplines may be preemptive or nonpreemptive priority disciplines or first come, first served. Validation of the complementary algorithm for these cases is not yet complete. We also plan to program the algorithm to handle state dependent service rates.

Appendix

Consider a central-server model (Fig. 1) with 5 customers; details regarding service times and branching probabilities are shown in Tables 1 and 2.

Zeroth step Let r_i be an N-vector whose jth element is the service rate of the local balance interface of queue i when there are j customers in the local balance interface. Then from $S^{(0)}$ we get r_1, r_2 and r_3 (see Table 3).

By analysis of the reduced networks we obtain the throughputs and mean queue lengths shown in Table 4. Note that we can have $y_1 = 1$, $y_2 = 0.25$ and $y_3 = 0.75$. From Eq. (4), we get the normalized throughputs shown in Table 4. With an error tolerance of 0.05 this solution is well within the allowable queue length error. We

Table 1 Service times of the example.

Queue number	*Service time*	*Mean*	*Coefficient of variation*
1	hyperexponential	3	2
2	Erlangian	4	$1/\sqrt{2}$
3	Constant	8/3	0

Table 2 Branching probability matrix of the example.

	1	*2*	*3*
1	0	0.25	0.75
2	1	0	0
3	1	0	0

Table 3 Rates of the local balance interfaces.

Number of customers	r_1	r_2	r_3
1	1/3	1/5	1/4
2	3/7	5/19	4/13
3	7/15	19/65	13/40
4	15/31	65/211	40/121
5	31/63	211/665	121/364

Table 4 Performance values obtained from reduced networks.

Queue number	*1*	*2*	*3*
mean queue lengths	3.30	0.385	1.122
throughputs	0.29	0.075	0.0225
normalized throughputs	0.589	0.596	0.601

therefore proceed to step 2 and modify the service rates of $S^{(0)}$ to get $S^{(1)}$. However, because the modified rates (or $S^{(1)}$) are within 5% of the original rates (of $S^{(0)}$), the algorithm terminates. The performance values obtained by analysis of the reduced networks are the final outputs of the algorithm.

References

1. F. Baskett, K. M. Chandy, R. R. Muntz, and F. Palacios-Gomez, "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," to be published in *J. ACM*.
2. J. Martin, *Design of Real-Time Computers*, Prentice-Hall, Inc., Englewood Cliffs, NJ 1967.
3. K. M. Chandy, "The Analysis and Solutions for General Queueing Networks," *Proceedings of the Sixth Annual Princeton Conference on Information Sciences and Systems*, Princeton University, Princeton, NJ 1972.
4. H. Kobayashi, "Some Recent Progress in Analytic Studies of System Performance," *First USA-Japan Computer Conference Proceedings*, AFIPS, New York 1972, p. 130.
5. P. E. Green and D. T. Tang, "Some Recent Developments in Teleprocessing System Optimization," *1973 IEEE Intercon Technical Papers* (Proceedings of IEEE International Conference and Exposition), IEEE, New York, March 26-30, 1973, p. 14/1.
6. K. M. Chandy, U. Herzog, and L. Woo, "Parametric Analysis of Queuing Network Models," *IBM J. Res. Develop.* **19,** 36 (1975, this issue).
7. U. Herzog, L. Woo, and K. M. Chandy, "Solution of Queuing Problems by a Recursive Technique," *Research Report RC4957*, IBM Thomas J. Watson Research Center, Yorktown Heights, NY 10598, July 1974.
8. P. J. Courtois and J. Georges, "On a Single Server Finite Queueing Model with State Dependent Arrival and Service Processes," *Oper. Res.* **19,** 424 (1971).
9. W. J. Gordon and G. F. Newell, "Closed Queueing Systems with Exponential Servers," *Oper. Res.* **15,** 254 (1967).

Received March 15, 1974

K. M. Chandy is located at the University of Texas, Austin, Texas 78712; U. Herzog is at the University of Stuttgart, Stuttgart, West Germany; and L. Woo is at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York 10598.