Analysis of Closed Queuing Networks with Periodic Servers

GASTON H. GONNET AND DAVID E. MORGAN, MEMBER, IEEE

Abstract—A periodic network is a queuing network whose steady-state behavior is not constant in time, but repeats itself in a cycle. This behavior may be caused by the introduction of periodic servers, e.g., paging drums. The model presented is a generalization of some other models of queuing networks, and provides a more general definition of steady-state behavior. A theoretical solution is presented. Examples of theoretical and approximate solutions are presented for a well-known queuing network model of a computer system.

Index Terms—Bounded variation periodic functions, computer networks, computer system modeling, eigenvalue and eigenvector problem, periodic servers, queuing networks, time sharing systems.

I. Introduction

A PERIODIC network is a queuing network whose steady-state behavior is not constant in time, but repeats itself in a cycle. One important cause of this behavior is the introduction of periodic servers in the network. A periodic server is a special kind of server that starts service of its customers in the queue only at certain points in time. A rotating memory device used for swapping equal size blocks, such as a paging drum or disk, is an example of a periodic server. Fig. 1 shows a schematic graph on the behavior of a periodic server. Coffman and Denning [5] and Fuller and Baskett [7] have comprehensively studied the behavior of paging drums and disks. However, the authors are not aware of any comprehensive work that analyzes the effect such periodic servers have on queuing networks that contain them.

The classic analyses of queing networks containing disks or drums (Baskett *et al.* [2], Kleinrock [12], [13], Jackson [10], [11], Gordon and Newell [9], etc.) are based on the assumption that the requests to servers are started at random (not synchronous) times. In this case an exponential server is a good approximation. The interservice time may be considered random when we have a very low arrival rate, or the lengths of the records (i.e., service time) are variable. At the other extreme, when there is a heavy swapping activity of equal size pages and the queue waiting for service is essentially never empty, we may assume that the service time is constant

Manuscript received February 15, 1976; revised July 11, 1979. This work has been supported by the National Research Council and Defense Research Board of Canada, and the University of Waterloo.

G. H. Gonnet is with the Pontificia Universidade Catolica, Rio de Janeiro, Brazil, on leave at the Department of Computer Science, University of Waterloo, Waterloo, Ont., Canada.

D. E. Morgan is with the Department of Computer Science, University of Waterloo, Waterloo, Ont., Canada.

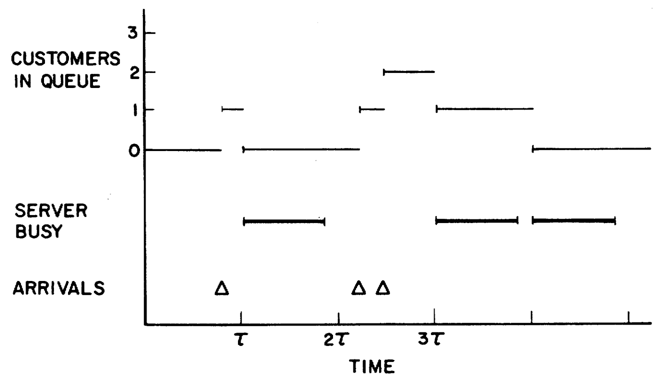


Fig. 1. Schematic graph of the behavior of a periodic server.

and approximate the behavior of the devices with an M/D/1 server. However, because the assumptions that support such approximations are different, they are not valid to describe situations between these two extreme cases. Considering the periodic nature of certain devices in a system allows us to model system behavior better across a broad spectrum of cases while including both extremes as special cases.

The purpose of this study is to develop theoretical solutions of queuing networks containing periodic servers. Section II of the paper defines terms and notation. Section III provides a steady-state solution for a periodic network. Section IV presents an example of a queuing network analyzed with these techniques.

II. General Definitions and Assumptions

Consistently we denote vectors and matrices in boldface capital letters (e.g., $\mathbf{A}, \mathbf{W}, \mathbf{P}$). Elements of matrices or vectors are denoted by the corresponding lower case boldface, with the proper subscripts (e.g., $\mathbf{a}_{ij}$). A superscript + or − indicates the limit to the quantity by the right or left, respectively (e.g., τ^+). Superscript T (e.g., $\mathbf{P}^T$) on a vector matrix means the transpose of that vector or matrix.

A *queuing network* will be defined by a directed graph in which each node represents a particular queue-server pair, and each edge represents a possible path for customers from one server to another.

We assume that there is a fixed number n of indistinguishable customers circulating through the network, i.e., it is a closed network. Let $k = k_e + k_p$ be the total number of servers (exponential + periodic) in the network. Each server has arbitrary mean service time. Associated with each server there

0098-5589/79/1100-0653$00.75

is a queue of capacity at least n. The analysis given here is essentially the same for any simple queuing discipline, i.e., FIFO, LIFO, etc.

The state of the network will be defined by the number of customers in each exponential server, and by the number of customers in queue and in service in each periodic server. Notice that for a periodic server, the server may be idle although its queue may have customers awaiting service. If $R(k_e, k_p, n)$ is the number of possible distinguishable states, then we have [12]:

$$R(k_e, 0, n) = \binom{n + k_e - 1}{n} \tag{1}$$

if there are no periodic servers. The following recursion formula yields the number of distinguishable states r counting both exponential and periodic servers. The formula results from considering how the number of states changes when one replaces an exponential server with a periodic server.

$$r = R(k_e, k_p, n) = 2R(k_e + 1, k_p - 1, n) - R(k_e, k_p - 1, n) \tag{2}$$

We find that

$$r \leqslant 2^{k_p} \binom{n + k - 1}{n}. \tag{3}$$

We will number each state with an unique integer from 1, 2, $\cdots, r$.

Let $\mathbf{P}(t)$ be any r-dimensional column vector that contains the probability that each of the r states is the state of the network at time t. Normally $\mathbf{P}(t)$ is a function of the topology of the network (reflected by the global balance equations), the initial state, and time.

We say that the network is in *periodic steady state* if the probability state vector $\mathbf{P}_s(t)$, is a function of time which repeats its behavior every τ units of time (e.g., see Fig. 2). τ will be called the *cycle* of the network. Because a network that is τ-periodic is also 2τ-periodic, we will take as *canonical form* the smallest possible τ. Thus we have:

$$\mathbf{P}_s(t) = \mathbf{P}_s(t + \tau) \quad [0 \leqslant t \leqslant \tau]. \tag{4}$$

Note that this definition of steady state covers the normal notion, since given any $\mathbf{P}(t)$ for which $\mathbf{P}'(t) = 0 \Rightarrow \mathbf{P}(t) = constant = \mathbf{P}(t + \tau)$.

III. Theoretical Solution

First we will consider only the time intervals where the periodic servers do not cause a change in state of the network; that is, those intervals that do not include points in which an instantaneous state transition occurs as a result of any periodic server activity. An *instantaneous state transition* at time T is a discontinuity of $\mathbf{P}(t)$ at $t = T$. An instantaneous state transition may occur by the starting or completion of service in a synchronous server. In any of these intervals, the global balance equations yield:

$$\mathbf{P}'(t) = \mathbf{A}.\mathbf{P}(t). \tag{5}$$

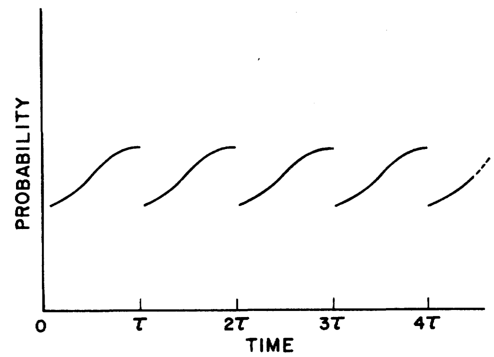


Fig. 2. Example component of probability state vector in steady state.

$\mathbf{A}$ is an $r \times r$ matrix that defines all possible state transitions for the exponential servers. Each element $\mathbf{a}_{ij}$ of the matrix $\mathbf{A}$ is determined by the global balance equations (or flow-conservation equations, as in Kleinrock [12]) and results in: $\mathbf{a}_{ij} = 0$ if there exists no single customer transition that takes the network from state j to state i; $\mathbf{a}_{ij} = \mu_{ji}$ iff departures from state j to state i occur at a rate μ_{ji} and finally $\mathbf{a}_{ii} = -\Sigma_j \mu_{ij}$ (total rate of departure from state i).

It follows immediately that the number of nonzero terms of each row of $\mathbf{A}$ is bounded by the topology of the network (i.e., the bound is the number of outgoing edges from exponential server nodes + 1). Moreover this matrix $\mathbf{A}$ is constant with respect to time, and the maximum norm, $\|\mathbf{A}\| \leqslant 2\Sigma\mu_k$, (where μ_k is the departure rate from exponential server k), is bounded by the characteristics of the network, but independent of the number of customers and possible states.

Let $e^{\mathbf{A}}$ be defined in the usual way (Frazer *et al.* [6]) as:

$$e^{\mathbf{A}} = \mathbf{I} + \mathbf{A} + \frac{\mathbf{A}^2}{2} + \frac{\mathbf{A}^3}{3!} + \cdots \tag{6}$$

In the intervals (t_0, t) where no activity in the periodic servers occurs, the solution of the system (5) is

$$\mathbf{P}(t) = e^{\mathbf{A}t}.\mathbf{P}(t_0), \tag{7}$$

where $\mathbf{P}(t_0)$ is the initial state vector. It is important to note that the behavior of the $\mathbf{P}(t)$ vector in intervals with no periodic-server activity is completely characterized by $\mathbf{P}(t_0)$.

A. Single Periodic Server Networks

Suppose there is only one periodic server in the system, and it has both cycle and service time equal to τ. Let $\mathbf{R}$ be the matrix that defines the state transition caused by the start or completion of service in the periodic server, based on the probability state vector before such an operation, hence:

$$\mathbf{P}(\tau^+) = \mathbf{R}.\mathbf{P}(\tau^-). \tag{8}$$

$\mathbf{R}$ is an $r \times r$ matrix consisting of ones and zeros only. $\mathbf{R}$ describes the discontinuous transitions that occur in the network.

Now we will define $\mathbf{P}_s(0^+) = \mathbf{P}_s$, a probability state vector, that will characterize the steady-state solution $\mathbf{P}_s(t)$ of the periodic network. Using (4), (7), and (8) we obtain:

$$\mathbf{P}(0^+) = \mathbf{P}(\tau^+) = \mathbf{R}.\mathbf{P}(\tau^-) = \mathbf{R}.e^{\mathbf{A}t}.\mathbf{P}(0^+). \tag{9}$$

Let $\mathbf{P}_s$ satisfy:

$$\mathbf{P}_s = \mathbf{R}.e^{\mathbf{A}\tau}.\mathbf{P}_s \tag{10}$$

(i.e., $\mathbf{P}_s$ is the eigenvector of $\mathbf{R}.e^{\mathbf{A}\tau}$ whose corresponding eigenvalue is 1). Since $\mathbf{P}_s$ is a probability state vector, we normalize $\mathbf{P}_s$ in such a way that the sum of all its elements is 1.

Then:

$$\mathbf{P}_s(t) = e^{\mathbf{A}t}.\mathbf{P}_s \quad [0 < t < \tau] \tag{11}$$

defines the periodic steady state. Note that if $\mathbf{R}.e^{\mathbf{A}\tau}$ contains no unit eigenvalue then there is no possible *steady-state* solution for the network.

Let $\mathbf{Q}(t) = \mathbf{W}.\mathbf{P}_s(t) = (w_1, w_2, \cdots, w_r).\mathbf{P}_s(t)$ be a performance index that characterizes behavior of the network that we want to evaluate (e.g., probability of certain configuration; mean queue length; utilization of certain server; or simply moments of marginal distributions). Clearly we are interested in the average behavior of $\mathbf{Q}(t)$ rather than in its value for a certain t. Because of the periodic behavior of $\mathbf{P}_s(t)$ and consequently $\mathbf{Q}(t)$, we only need to consider the average over a τ period of time. Then the averaged $\mathbf{Q}(t)$, denoted $\mathbf{Q}$, will be:

$$\mathbf{Q} = \tau^{-1} \int_0^\tau \mathbf{W}.\mathbf{P}(t)\, dt = \mathbf{W}.\tau^{-1} \int_0^\tau e^{\mathbf{A}t}\, dt.\mathbf{P}_s. \tag{12}$$

We will call

$$\tau^{-1} \int_0^\tau e^{\mathbf{A}t}\, dt.\mathbf{P}_s \tag{13}$$

the *average probability state vector* (APSV). Notice that for a given network this vector does not depend on $\mathbf{W}$, so we may calculate any value $\mathbf{Q}$ for any vector $\mathbf{W}$ while only having to perform the APSV calculation once.

B. Multiple Periodic Servers

Assume now that there is more than one periodic server, or that there is more than one discontinuous event in the cycle of the periodic server. Both situations imply that we will have more than one discontinuity in the probability state vector. If there is no rational relation between their periods, it is intuitively clear that there is no way to define a periodic steady state as before. Suppose then that there exists a rational relation between their cycles, such that after a sequence $0 = \tau_0 < \tau_1 < \cdots < \tau_j = \tau$ of transition points in time, we complete a period, and the network is in the same state as in time 0. Associated with the end of each interval $[\tau_{i-1}, \tau_i]$ there is a corresponding transition matrix $\mathbf{R}_i$. Let $\Delta_i = \tau_i - \tau_{i-1}$.

The solution of this general system, using similar notation to that of Section III-A is as follows.

For each period of time between transitions we have:

$$\mathbf{P}'(t) = \mathbf{A}.\mathbf{P}(t) \Rightarrow \mathbf{P}(t + \tau_i) = e^{\mathbf{A}t}.\mathbf{P}(\tau_i^+) \quad [0 \leq t < \Delta_{i+1}]. \tag{14}$$

For each transition we have:

$$\mathbf{P}(\tau_i^+) = \mathbf{R}_i.\mathbf{P}(\tau_i^-). \tag{15}$$

In the periodic steady state, putting this together we have:

$$\mathbf{P}_s = \mathbf{R}_j.e^{\mathbf{A}\Delta_j} \cdots \mathbf{R}_1.e^{\mathbf{A}\Delta_1}.\mathbf{P}_s. \tag{16}$$

$\mathbf{P}_s$ now is calculated as the eigenvector of the product of all the matrices whose eigenvalue is 1, and is normalized so that its elements add up to 1. $\mathbf{P}_s(t)$ is sectionally defined by:

$$\mathbf{P}_s(t + \tau_i) = e^{\mathbf{A}t}.\mathbf{P}_s(\tau_i^+) \quad [0 \leq t < \Delta_{i+1}] \tag{17}$$

$$\mathbf{P}_s(\tau_i^+) = \mathbf{R}_i.e^{\mathbf{A}\Delta_i}.\mathbf{P}_s(\tau_{i-1}^+) \tag{18}$$

$$\mathbf{P}_s(0^+) = \mathbf{P}_s. \tag{19}$$

The average probability state vector is calculated as follows:

$$\text{APSV} = \tau^{-1} \sum_{i=1}^{j} \int_0^{\Delta_i} e^{\mathbf{A}t}\, dt.\mathbf{P}(\tau_{i-1}^+). \tag{20}$$

C. Complexity of the Computations

In the following evaluations we will assume that $k = o(r)$. The fact that $\mathbf{A}$ is sparse becomes quite important for the numerical computation. The common Taylor expansion is best suited to compute the exponential forms. Since $\|\mathbf{A}\|$ is bounded, only a fixed number of terms of the expansion will be used. Since the number of nonzero elements in $\mathbf{A}$ is $O(r)$, each power of $\mathbf{A}$ can be calculated in $O(r^2)$ multiplications. Moreover, since the number of terms in the expansion is constant, the evaluation of $e^{\mathbf{A}}$ also takes $O(r^2)$ operations.

The same arguments apply for the evaluation of $\int_0^\tau e^{\mathbf{A}t}\, dt$.

Since $\mathbf{R}$ is a matrix of r 1's, the product $\mathbf{R}.e^{\mathbf{A}}$ will be performed in $O(r^2)$ operations.

The evaluation of the eigenvalue that determines $\mathbf{P}_s$ is accomplished by the solution of a system of linear equations of dimension $r - 1$. The solution of such a system of equations could be done in $O(r^2)$ operations depending on some conditions of the matrices that are not always true (Varga [16]). Unfortunately if we cannot use $O(r^2)$ techniques or if we have $j > 1$ and we have to actually multiply the matrices in (16), the solution process becomes $O(r^3)$.

The actual size and, in some cases, exponential growth of the problem indicates that only small networks may be solved exactly.

D. General Approximate Solution of Periodic Networks

To obtain an approximate solution we observe the fact that in periodic networks (as described in Section III-B), all the components of the network oscillate with the same cycle. In this steady state, each node has an arrival probability distribution, and probability state vector (of the node itself, ignoring the rest of the network) that repeats its behavior every τ units of time.

In [8] we discuss a method of computing approximate solutions based on the above remark. These approximate solutions are most accurate when we have only one periodic server.

IV. Example Results

In this section we will analyze three networks. The first is a simple loop containing two servers each with its queue, and the second and third are well-known models of a time sharing system [1], [14]. The first example will be used mainly to

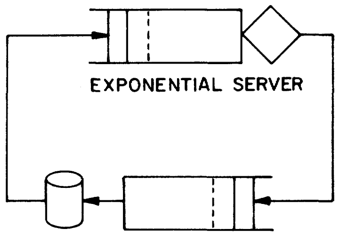


Fig. 3. Topology of simple loop.

illustrate the mechanisms of the theoretical solution. The other examples will be computed for situations similar to the ones studied in [1], [14], so we can compare results.

A. Simple Loop

Fig. 3 illustrates the topology of the first example. A fixed number of customers n circulate through the network. For this example (using the exact derivation), assume that $n = 2$. The number of possible states of the network, using (2) is

$$r = R(1,1,2) = 2R(2,0,2) - R(1,0,2) = 5. \tag{21}$$

A detailed description of each state is:

State	Periodic server	Exponential server
1	0-0	2
2	0-1	1
3	1-0	1
4	0-2	0
5	1-1	0

where the two numbers for the periodic server give the number of customers in service and in queue.

The global balance equations define the matrix $\mathbf{A}$ so that:

$$\mathbf{P}'(t) = \begin{pmatrix} p_1' \\ p_2' \\ p_3' \\ p_4' \\ p_5' \end{pmatrix} = \begin{pmatrix} -\mu & 0 & 0 & 0 & 0 \\ \mu & -\mu & 0 & 0 & 0 \\ 0 & 0 & -\mu & 0 & 0 \\ 0 & \mu & 0 & 0 & 0 \\ 0 & 0 & \mu & 0 & 0 \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{pmatrix}$$

In this case we compute $e^{\mathbf{A}t}$ explicitly, i.e.,

$$e^{\mathbf{A}t} = \begin{pmatrix} e^{-\mu t} & 0 & 0 & 0 & 0 \\ \mu t e^{-\mu t} & e^{-\mu t} & 0 & 0 & 0 \\ 0 & 0 & e^{-\mu t} & 0 & 0 \\ 1-(1+\mu t)e^{-\mu t} & 1-e^{-\mu t} & 0 & 1 & 0 \\ 0 & 0 & 1-e^{-\mu t} & 0 & 1 \end{pmatrix}$$

The matrix $\mathbf{R}$ is derived from the transitions that occur at τ caused by the periodic server. This in words is as follows: for the periodic server, when the customer in the server (if any) completes its service, one of the customers in the queue (if any) starts service.

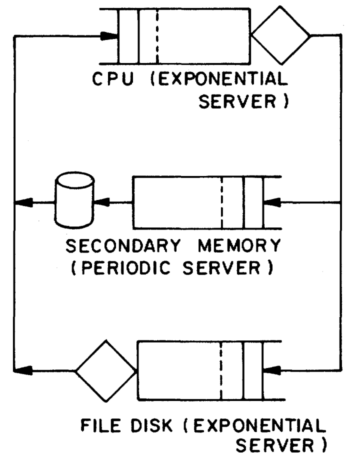


Fig. 4. Topology of resource loop model.

$$\mathbf{R} = \begin{pmatrix} 1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

If we denote $\alpha = \mu\tau$, then the matrix that describes a complete transition, from time 0^+ to time τ^+, using (9), is

$$\mathbf{R}.e^{\mathbf{A}\tau} = \begin{pmatrix} e^{-\alpha} & 0 & e^{-\alpha} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \alpha e^{-\alpha} & e^{-\alpha} & 1-e^{-\alpha} & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 1-(1+\alpha)e^{-\alpha} & 1-e^{-\alpha} & 0 & 1 & 0 \end{pmatrix}$$

The eigenvector whose eigenvalue is 1, that defines the steady-state solution, is

$$\begin{pmatrix} p_1 \\ p_2 \\ p_3 \\ p_4 \\ p_5 \end{pmatrix} = \begin{pmatrix} \dfrac{1}{e^{\alpha}+1-(1+\alpha)e^{-\alpha}} \\ 0 \\ \dfrac{e^{\alpha}-1}{e^{\alpha}+1-(1+\alpha)e^{-\alpha}} \\ 0 \\ \dfrac{1-(1+\alpha)e^{-\alpha}}{e^{\alpha}+1-(1+\alpha)e^{-\alpha}} \end{pmatrix}$$

Clearly this is not the way one would normally compute solutions; instead, they would be numerically computed.

B. Resource Loop Model of a Time Sharing System

Fig. 4 illustrates the network with which we intend to model part of a time sharing system. This model has been carefully described and analyzed in [1], [3], [14], among others. The model consists of three servers: CPU, secondary memory, and other I/O in the form of a file disk. The secondary memory is likely to behave like a periodic server since we will probably use a drum or disk to swap fixed size pages in and out. In this model the secondary memory plays a decisive role in the behavior of the overall performance, so an inaccurate representation in its modeling may invalidate the results.

In this case we find 16 different states. They are indicated by (customers in CPU, in periodic server, in periodic queue, file disk):

1	(0, 0-0, 3)	9	(2, 0-1, 0)
2	(0, 0-1, 2)	10	(3, 0-0, 0)
3	(0, 0-2, 1)	11	(0, 1-0, 2)
4	(0, 0-3, 0)	12	(0, 1-1, 1)
5	(1, 0-0, 2)	13	(0, 1-2, 0)
6	(1, 0-1, 1)	14	(1, 1-0, 1)
7	(1, 0-2, 0)	15	(1, 1-1, 0)
8	(2, 0-0, 1)	16	(2, 1-0, 0)

The following paragraph shows a typical output of the program that computes the exact solution.

SOLUTION OF RESOURCE LOOP MODEL

degree of multiprogramming (number of customers) = 3
number of available memory pages= 128
mean compute time between i/o requests= 20.0
mean total compute time= 1000.0
locality and memory management parameter= 1.50
secondary memory service time= 5.0
file-disk mean service time= 30.0

DERIVED VALUES

mean time between memory faults= 2.787
rate of cpu= 0.4098
rate of file disk= 0.0333
probability of exit from the resource loop= 0.00244
probability of memory fault= 0.87555
probability of i/o fault= 0.12201
Average Probability State Vector (APSV)
0.08146 0.06053 0.01127 0.00189
0.05430 0.01513 0.00332 0.02179
0.00634 0.00959 0.10023 0.16283
0.17247 0.10645 0.15199 0.04040

probability of 0, 1, · · · , 3 customers at node 0
0.590669011 0.331203296 0.068537984 0.009589709
expected number of customers= 0.4970484 variance= 0.666787

probability of 0, 1, · · · , 3 customers at node 1
0.167146126 0.329084019 0.329416604 0.174353252
expected number of customers= 1.5109770 variance= 0.965856

probability of 0, 1, · · · , 3 customers at node 2
0.386003520 0.317475401 0.215064009 0.081457069
expected number of customers= 0.9919746 variance= 0.962721

CPU UTILIZATION 0.4093310

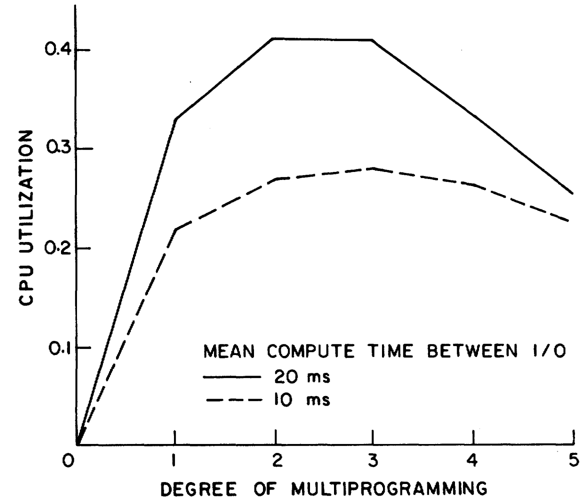


Fig. 5. Exact values for the resource loop model.

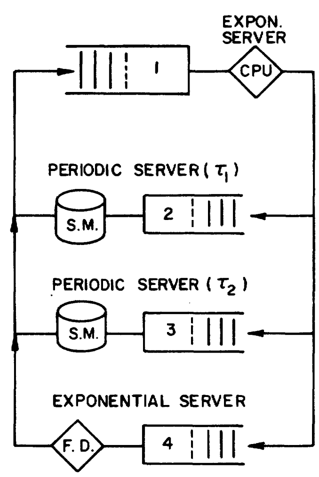


Fig. 6. Topology of resource loop model with two periodic servers.

The following table and Fig. 5 show the exact values for several degrees of multiprogramming. These results compare very well to those obtained by [1], [14]. The graph shows a slight shift to the right that may be interpreted in terms of the behavior of the secondary memory server, which is a periodic server.

Degree of multiprogramming	CPU utilization	Queue length at secondary memory
1	0.330	0.174
2	0.410	0.662
3	0.409	1.511
4	0.339	2.780
5	0.257	4.143
6	0.197	5.391

C. Resource Loop Model of a Timesharing System with Two Periodic Servers

Fig. 6 shows the topology of a particular timesharing model with two secondary memory devices modeled as periodic servers. The service times of these servers are τ_1 = 7.5 ms and

TABLE I
RESULTS FOR THE RESOURCE LOOP MODEL WITH TWO PERIODIC SERVERS

Degree of multiprogramming	μ_1	$\bar{n}_1$	μ_2	$\bar{n}_2$	μ_3	$\bar{n}_3$	μ_4	$\bar{n}_4$
1	0.2725	0.2725	0.0993	0.0993	0.2198	0.2198	0.4048	0.4048
2	0.3015	0.3539	0.2777	0.3041	0.6114	0.7764	0.4506	0.5655
3	0.2822	0.3567	0.4400	0.5511	0.8593	1.5368	0.4112	0.5554
4	0.2369	0.3074	0.5452	0.7882	0.9631	2.4535	0.3375	0.4509

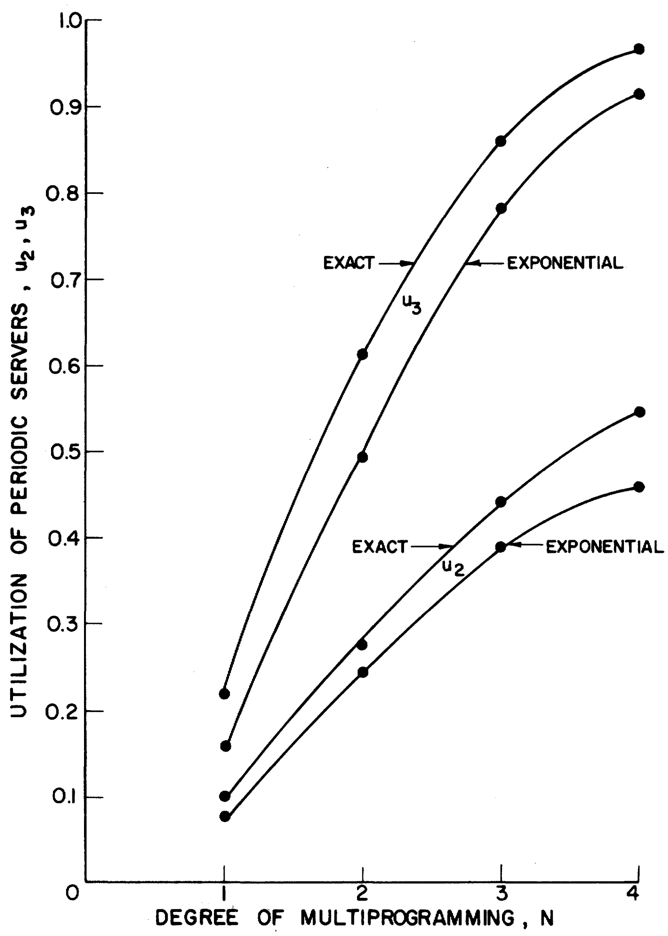


Fig. 7. Comparison of utilization of resource loop with two periodic servers under different models.

τ_2 = 15 ms, respectively. Note that the total throughput of the secondary memory is the same as for the previous example. In this case we used the solution described in Section III-B.

Table I and Fig. 7 show the average number of customers ($\bar{n}_i$) and utilization factor (μ_i) of each server for various numbers of customers.

V. CONCLUSIONS AND FURTHER WORK

In this paper we present a method for the exact solution of queuing networks that contain periodic servers. The method is primarily applicable to networks with a small number of servers.

The example applications presented here deal with computer systems, featuring values of the period τ in the order of milliseconds. Neither the problem statement, nor the assumptions, nor the methods of solution presented depend on the periods of the periodic server being small. Thus, the method is applicable to any kind of system that can be modeled as a queuing network having either periodic or exponential servers, regardless of the values of τ. A wide variety of kinds of systems can be modeled, from banks to supermarkets to computer networks. Note that any daily process is by nature periodic.

A number of example queuing networks were analyzed using these methods, and the results were compared with simulations. Discrepancies were found to be very small. No attempt has yet been made to compare measurements of a real system with results of using these methods. The necessary measurement tools are available [15], and plans exist to make such comparisons in the near future.

The work extends the class of functions that can be used as service distributions to include the class of all bounded variation periodic functions.

It appears possible to extend this work to provide solution of open networks, i.e., when external arrivals and departures are allowed. Further research should be done to see if it is possible to generalize the class of nonperiodic service distributions (e.g., M/G/1) that can be included with periodic servers in queuing networks.

Acknowledgment

The authors wish to acknowledge the helpful comments of Prof. J. D. Lawson, Prof. J. Wong, and Prof. J. Majithia. The programming assistance of M. Unsoy and the very constructive criticisms of the referees are gratefully acknowledged.

References

[1] M. Badel, E. Gelenbe, J. Leroudier, and D. Potier, "Adpative optimization of a time-sharing system's performance," *Proc. IEEE*, vol. 63, June 1975, pp. 958–965.

[2] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers," *J. Ass. Comput. Mach.*, vol. 22, pp. 248–260, Apr. 1975.

[3] A. Brandwajn, "A model of a time sharing virtual memory system solved using equivalence and decomposition methods," *Acta Informatica*, vol. 4, pp. 11–47, 1974.

[4] K. M. Chandy, U. Herzog, and L. Woo, "Parametric analysis of queueing networks," *IBM J. Res. Develop.*, vol. 19, pp. 36–42, Jan. 1975.

[5] E. G. Coffman and P. J. Denning, *Operating Systems Theory*. Englewood Cliffs, NJ: Prentice-Hall, 1973, ch. 5.

[6] R. A. Frazer, W. J. Duncan, and A. R. Collar, *Elementary Matrices*. London: Cambridge Univ. Press, 1965.

[7] S. H. Fuller and F. Baskett, "An analysis of drum storage units," *J. Ass. Comput. Mach.*, vol. 22, pp. 83–105, Jan. 1975.

[8] G. H. Gonnet and D. E. Morgan, "Analysis of closed queueing networks with periodic servers," Univ. Waterloo, Waterloo, Ont., Canada, CCNG Rep. E-58.

[9] W. J. Gordon and G. F. Newell, "Closed queueing systems with exponential servers," *Operations Res.*, vol. 15, pp. 254–265, 1967.

[10] J. R. Jackson, "Networks of waiting lines," *Operations Res.*, vol. 5, pp. 518–521, 1957.

[11] ——, "Jobshop-like queueing systems," *Management Science*, vol. 10, pp. 131–142, Oct. 1963.

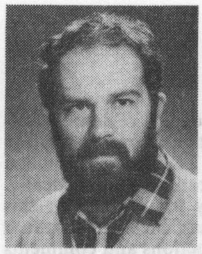
[12] L. Kleinrock, *Queueing Systems*. New York: Wiley, 1974, vol. I.

[13] ——, *Queueing Systems*. New York: Wiley, 1976, vol. II.

[14] J. Leroudier and M. Parent, "Discrete event simulation modeling of computer system for performance evaluation," IRIA, France, Res. Rep. 177, June 1976.

[15] D. E. Morgan, W. Banks, D. P. Goodspeed, and R. Kolanko, "A computer network monitoring system," *IEEE Trans. Software Eng.*, vol. SE-1, pp. 299–311, Sept. 1975.

[16] R. S. Varga, *Matrix Iterative Analysis*. Englewood Cliffs, NJ: Prentice-Hall, 1962.



Gaston H. Gonnet was born in Montevideo, Uruguay, on September 22, 1948. He received the M.S. and Ph.D. degrees in computer science from the University of Waterloo, Waterloo, Ont., Canada, in 1975 and 1977, respectively.

He is currently a faculty member at the University of Waterloo where he is on leave of absence from the Pontificia Universidade Catolica, Rio de Janeiro, Brazil. His research interests are in the areas of analysis of algorithms, operating systems, programming languages, and numerical analysis.

David E. Morgan (M '74) received the B.S. degree in mathematics from Rose Polytechnic Institute, Terre Haute, IN, the M.S. degree in mathematics from the University of Michigan, Ann Arbor, and the Ph.D. degree from the University of Waterloo, Waterloo, Ont., Canada.

He is currently an Associate Professor of Computer Science and Associate Director of the Computer Communications Networks Group at the University of Waterloo. He designed and developed a system for observing and controlling the behavior of computer networks and distributed systems. He has served as consultant to several U.S. and Canadian business and governmental organizations, and is Vice President of Telecom Network Technology, Inc., a firm specializing in consulting services and technology transfer. While on sabbatical leave at Digital Equipment Corporation, he contributed to the design of tools for software engineering and systems for managing the behavior of a computer network. Earlier, as a member of the technical staff at Bell Laboratories, he participated in the development of Bell's electronic switching systems and IBM's timesharing system for the 360/67. He has authored several papers relating to reliability, availability, maintainability, and performance of computer systems and networks.

Dr. Morgan is a member of the Association for Computing Machinery and a life member of Tau Beta Pi.