WIS-CS-200-74
COMPUTER SCIENCES DEPARTMENT
The University of Wisconsin
1210 West Dayton Street
Madison, Wisconsin 53706

Received October 22, 1973

DRUM MODELS USING AN ITERATIVE SOLUTION FOR CLOSED QUEUEING NETWORKS

by
J. H. Hine
and
D. R. Fitzwater

Computer Sciences Technical Report #200

January 1974

ABSTRACT

An algorithm by Buzen to find the normalizing constant and marginal probabilities in a closed queueing network is generalized to the model of Muntz and Baskett. A technique involving the iterative solution of a closed queueing network to implement queue dependent completion time distributions is presented. An accurate model of a paging drum for queueing networks is given as an example.

I. Introduction

Queueing network models have been studied in various forms as models for multiprogrammed computer systems. These models have generally taken one of two approaches. The first consider only two or three of the major interacting elements of a computer system in order to present a solution derived from accurate service and customer interarrival time distributions. The second approach assumes Poisson interarrival time distrubutions and exponential service time distributions to study larger, more generalized models. Recent work by Chandy [C1] and Muntz and Baskett [B3,M2] has opened the possibility of studying more generalized service time distributions in larger models by the use of the method of stages and various scheduling disciplines. Section II presents a generalization of an algorithm first suggested by Buzen [B4,B5]. This algorithm allows the efficient computation of the normalizing constant and marginal distribution of the closed network model developed by Muntz and Baskett.

Section III is addressed specifically to the accurate modelling of a paging drum within a network model. The derivation of the model is presented along with a discussion of its use in both open and closed network models. The use of this model in closed networks involves an iterative solution first suggested by Pinkerton [P1]. The algorithm and proof of convergence are both presented.

II. Solution of Closed Network Models

The classic solution of a closed queueing network model was presented by Gordon and Newell [G1]. Their model assumed an exponential service time distribution at all service centers, each of which could have a fixed number of parallel servers. Using the method of local balance Muntz and Baskett have been able to generalize the model of Gordon and Newell [G1]. These generalizations include the addition of classes of customers, last-come-first-served and processor sharing scheduling disciplines, as well as a restricted structure of exponential stages to represent generalized service time distributions.

The structure of stages for the model is shown in fig. 1. A customer enters at the first stage and proceeds to the i^{th} of K stages with probability a_{i-1} or exits with probability $b_{i-1} = 1 - a_{i-1}$. The use of stages was considered by Moore [M1] in modelling a large scale time-sharing system. Stages with $b_i = 0$ for $i < K$ were used by Pinkerton [P1] to model a drum in a central server model of a paging system.

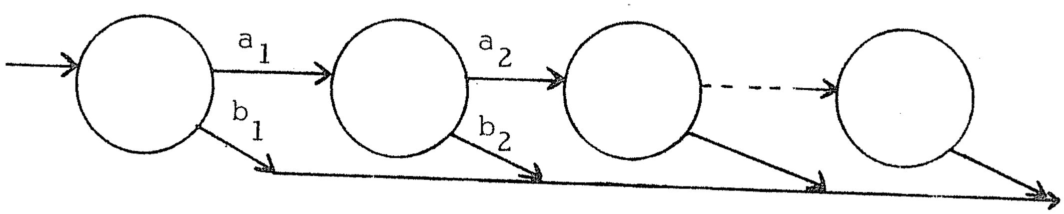

Fig. 1: A network of stages

The generalizations made by Muntz and Baskett may be used to model more accurately the various service time distributions of the components of a computer system. The solution of a closed network as given by Muntz and Baskett is

$$P(Z_1, Z_2, \ldots, Z_M) = G^{-1}(N) \prod_{i=1}^{M} F_i(Z_i) \qquad (1)$$

In eq. (1) $P(Z_1, Z_2, \ldots, Z_M)$ is the probability of the state described by all the Z_i and each Z_i is the description of the state of service center i. The particular form of Z_i depends on the absence or presence of classes of customers or stages. The $F_i(Z_i)$ are functions dependent on the scheduling discipline of service center i. Define $S(N,M)$ to be the set of possible states of a closed network with M service centers and number of customers, N. The normalizing constant, $G(N)$, may be written

$$G(N) = \sum_{S(N,M)} \prod_{i=1}^{M} F_i(Z_i) \qquad (2)$$

In his analysis of a central server model Buzen [B4] developed an extremely efficient algorithm for the evaluation of G(N). As an extension he also suggested an algorithm for service centers with parallel servers which he further emphasized in [B5]. In his development Buzen was primarily concerned with the model of Gordon and Newell. The generalizations of Muntz and Baskett preclude the direct application of Buzen's algorithm to find $G(N)$ as given by equation (2). However, assuming a constraint concerning the classes of customers and performing some relatively simple precalculation, the algorithm may be applied.

Buzen's second algorithm may also be speeded up and extended to provide not only the normalizing constant, but the marginal probability distributions at the individual service centers. To derive the modification to Buzen's algorithm consider the definition of $G(N)$ given by equation (2). Since the individual components of the vector Z_i appear only in $F_i(Z_i)$ equation (2) can be written as:

$$G(N) = \sum_{\Sigma n_i = N} \prod_{i=1}^{M} \left(\sum_{S(n_i,1)} F_i(Z_i) \right) \tag{3}$$

Step 1 of the algorithm is to calculate

$$f_i(n_i) = \sum_{S(n_i,1)} F_i(Z_i) \tag{4}$$

for $1 \leq i \leq M$ and $0 \leq n_i \leq N$. The remainder of the derivation follows the work of Buzen. Define the auxilliary function $g_1(m,n)$ to be the normalizing constant for a closed network with m service centers and n customers. $g_1(m,n)$ is recursively defined by

$$g_1(m,n) = \sum_{i=1}^{N} f_m(i)\, g_1(m-1,n-i) \tag{5}$$

Equation (5) is valid for $m > 1$ and $n > 0$. The necessary boundary conditions are

$$\begin{aligned} g_1(j,0) &= 1 \qquad & 1 \leq j \leq M \\ g_1(1,i) &= f_1(i) \qquad & 0 \leq i \leq N \end{aligned} \tag{6}$$

The normalizing constant for a network of M service centers with N customers, $G(N)$, is just $g_1(M,N)$. There is one constraint which must be placed on the model of Muntz and Baskett such that the above derivation is valid in the general case. Customers must be allowed to change classes in such a manner as to allow any individual customer to potentially become a member of any class. This arises from the computation of $f_i(n_i)$ which removes any designation of class from the customers. Given this constraint the normalizing constant may now be calculated by the following algorithm, which uses an array H with two rows and N columns.

Algorithm 1

1. Calculate $f_i(n_i)$ for $1 \le i \le M$ and $0 \le n_i \le N$.
2. $H(i,L) \Leftarrow f_i(L)$ for $0 \le L \le N$; $i \Leftarrow 1$; $j \Leftarrow 2$; $x \Leftarrow 2$.
3. $H(j,L) \Leftarrow \sum_{k=0}^{L} f_x(k) H(i,L-k)$ for $0 \le L \le N$.
4. $x \Leftarrow x + 1$; interchange i and j; if $x \le M$ go to step 3 otherwise the algorithm is complete and $g_1(M,N)$ is in $H(i,N)$.

This algorithm may be compared with the procedure of evaluating all possible states of the network. Since (1) we do not know a priori the scheduling discipline of the various service centers and (2) the values of f[I](N[I]) would most likely be evaluated beforehand in both cases, the computations are compared after the evaluation of f[I](N[I]). To evaluate (2) requires M-1 multiplications for each of

$$\binom{M+N-1}{N}$$

states and the same number of additions to sum the products. A network with five service centers and eight customers would have 495 states. The evaluation of G(N) would require 1980 multiplications and 495 additions. Step 3 of algorithm 1 uses

$$\sum_{I=0}^{N} (I+1) = (N+1)(N+2)/2$$

additions and multiplications for each of M-1 iterations. For the suggested network this totals 180 multiplications and additions.

In regard to step 1 note that this computation is subject to similar constraints as the evaluation of G(N) . The total number of customers of different classes and at different stages of service must total N[I]. A sequence of stages, for instance, may be considered as a network and algorithm 1 used to evaluate f[I](N[I]). This ability arises from the product form of F[I](Z[I]) where Z[I] represents classes of customers and service distributions with stages.

$g_2(i,N-k)$ represents the normalizing constant for the network obtained by removing service center i and k customers.

Thus, the normalizing constants of $M+1$ different networks are needed for all numbers of customers from 0 to N. M of the normalizing constants are found over a network with one service center of the original network removed. The $M+1$st normalizing constant is found for the entire network. Buzen suggested M repetitions of Algorithm 1 with the first $M-1$ repetitions over $M-1$ service centers and the last over M, however the amount of computing involved can be significantly reduced.

Algorithm 1 can be extended to generate an array, N by $M+1$, containing all values needed to find the marginal probabilities of the network. Examination of step 3 of Algorithm 1 reveals that the value $H(j,L)$ required only those values of $H(i,k)$ for $k \leq L$. By calculating $g_1(i,N)$ first, then $g_1(i,N-1)$ and so on through $g_1(i,0)$ successive $g_1(i,L)$ may be calculated using a single column of the array. Now note that the calculation of the normalizing constant for the network with the ith service center removed proceeds identically to the calculation of the normalizing constant for the entire network until x reaches i at step 4 of Algorithm 1. Algorithm 2, given below will generate an array of $M+1$ rows. The ith row, $1 \leq i \leq M$, will be the normalizing constants for the network with the ith service center removed. The $M+1$st row will be the normalizing constants for the complete network. As the algorithm proceeds the calculation of $g_2(i,n)$ will be started when $x = i$.

<u>Algorithm 2</u>

1. Calculate $f_i(n_i)$ for $1 \le i \le M$ and $0 \le n_i \le N$.

2. $g_2(1,k) \Leftarrow f_2(k)$; $g_2(2,k) \Leftarrow f_1(k)$ for $0 \le k \le N$; $x \Leftarrow 2$.

3. $x' \Leftarrow x-1$.

4. If $x' = 0$ or $x = 2$ go to step 5 else

   $$g_2(x',k) \Leftarrow \sum_{j=0}^{k} f_x(j)\, g_2(x',k-j) \quad \text{for } k = N, N-1, \ldots, 0$$

   $x' \Leftarrow x' - 1$

   Go to step 4

5. $$g_2(x+1,k) \Leftarrow \sum_{j=0}^{k} f_x(j)\, g_2(x,k-j) \quad \text{for } k = N, N-1, \ldots, 0$$

   $x \Leftarrow x+1$. If $x \le M$ go to step 3, otherwise the algorithm is complete.

In Algorithm 2 step 4 calculates the next step of Algorithm 1 for all normalizing constants for which a service center has already been skipped. Each computation proceeds uniquely from this point. Step 5 calculates the next iteration of Algorithm 1 for all normalizing constants for which a service center is yet to be skipped. The intermediate values of these normalizing constants are identical at this point. The marginal probability of k customers at service center i is given by equation (7) with $g_1(M,N) = g_2(M+1,N)$.

To evaluate the amount of computation necessary to compute the entire array $g_2(M+1,N)$, consider first the summation appearing in steps 4 and 5. The analysis of algorithm 1 revealed that this summation requires $(N+2)(N+1)/2$ additions and multiplications to compute the entire row. In algorithm 2 step 4 is performed

$$\sum_{I=3}^{M} (I-1)$$

times, step 5 is performed $M-2$ times and thus the calculation of a row is done a total of

$$1 + \sum_{I=3}^{M} I = (M+3)(M-2)/2$$

times. The array $g_2(M+1,N)$ may be computed in $(N+2)(N+1)(M+3)(M-2)/4$ multiplications and additions. For the previous example ($M=5$ and $N=8$) the total is 450 multiplications and additions. Even though this total increases as N squared and M squared the method is immensely improved over the method of evaluating each state, which grows with M and N factorial. It also shows a significant improvement over the 720 multiplications and additions it would require to do M iterations of algorithm 1 as suggested by Buzen. The array g_2 requires $(M+1) \times N$ words and a second array to hold $f[1](N[1])$ requires $M \times N$ words.

In many cases it is desirable to compute more than the normalizing constant. Virtually all information which may be obtained from such models depends on knowledge of the marginal probability distributions at the individual service centers. To find the probability that $n_i = k$ customers at service center i sum over all states with n_i fixed at k and let the n_j, $j \neq i$, take on all possible values with the remaining $N-k$ customers. This problem is analogous to that of finding the normalizing constant, except for the removal of a service center and the k customers known to be at that center.

$$P(n_i = k) = (1/G(N)) \sum_{\Sigma n_j = N \ \& \ n_i = k} \ \prod_{j=1}^{M} f_j(n_j)$$

$$= f_i(k)\, g_2(i, N-k)/g_1(M, N) \qquad (7)$$

III. A Drum Model for Queueing Networks

One of the drawbacks of queueing networks as models of computer systems has been the restricted scheduling and service time distribution for which solutions have existed. Very few input-output devices have an exponential service time distribution. First-come-first-served queueing is hardly the only algorithm used in scheduling within a computer system. A significant amount of work has appeared in which the scheduling and service time distribution of a processor have been studied [B1, B2]. However very little attention has been given to the service time distribution of devices such as drums and disks in queueing network models. Frequently a central server model is built to study the effects of parameters at the processor and an exponential service distribution tossed in to represent a paging drum or other I/O device.

1. A PAGING DRUM MODEL

To develop a model of a drum, or any other device, it is necessary to define a standard device representation for the purposes of comparison. The standard may represent either a service or queueing time distribution, whichever may be more suitable for modelling. Pinkerton [P1] developed a drum queueing time distribution based on the number of outstanding requests currently at the drum. These requests were assumed to be uniformly distributed over the sectors of the drum at a given instant in time. Denning [D1] has shown that this assumption is invalid. The probability of a request being present is smaller for sectors which have just passed the read/write heads. The standard representation must be developed in a manner which avoids this problem, since it should be as realistic as possible. Coffman's [C2] generating function for the queue length probabilities is derived independent of this assumption. His assumption concerned the distribution of sectors requested rather than the distribution of outstanding requests.

A queueing time distribution may be obtained from Coffman's results. Let P_n , $n \geq 0$, be the probability density function (pdf) obtained by repeated differentiation of Coffman's generating function. P_n is the probability in continuous time that n requests are queued at a given sector. The pdf for the queueing time conditioned on n previous requests in the queue is given by:

$$w_1(t|n) = \begin{cases} 0 & t < nr + r/m \\ 1/r & nr + r/m \leq t < (n+1)r+r/m \\ 0 & (n+1)r + r/m \leq t \end{cases} \qquad (8)$$

In eq. (8) m is the number of sectors and r the rotation time of the drum. This conditional pdf is uniformly distributed over a single rotation which occurs n rotations after the request initially enters the queue. The standard queueing time pdf is thus given by:

$$w_1(t) = \sum_{n=0}^{\infty} P_n \, w_1(t|n) \qquad (\ )$$

Pinkerton [Pl] has suggested an Erlang distribution as a model for the queueing time distribution of a drum. He employed a linear sequence of exponential stages imbedded in a cyclic model to generate this distribution. The parameters of the Erlang distribution were adjusted as a function of the number of requests at the drum. This led to an iterative procedure to find the solution. The generalized model of Muntz and Baskett may be used to improve upon this approximation. A service center with infinite servers is used to prevent queueing and a distribution with stages used to model the queueing time distribution at the drum. Parameters of this distribution may be adjusted to fit the pdf given by eq. (9) .

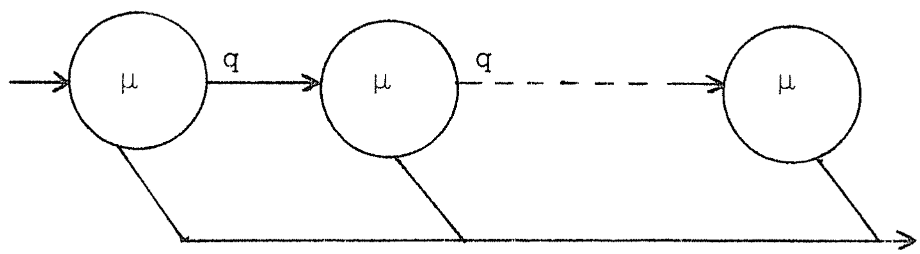

Fig. 2: A staged distribution for the drum model.

The network of stages in fig. 2 is similar to that in fig. 1. The values of a_j are q for all j except the final stage. Each individual stage has an exponential service time distribution with mean $1/\mu$. The queueing time pdf of the service center in fig. 2 may be found by initially considering two pdfs. $w_2(t|j)$ is the completion time pdf for exactly j stages and is obtained by the convolution of j exponential pdfs with mean $1/\mu$. $w_2(t|j)$ is an Erlang-j pdf. The probability of executing exactly j stages is given by:

$$\begin{cases} q^{j-1}(1-q) & 1 \le j < K \\ q^{K-1} & j = K \\ 0 & \text{otherwise} \end{cases} \tag{10}$$

K is the number of stages at the service center. With

$$w_2(t|j) = \mu^j t^{j-1} \exp(-\mu t)/(j-1)! \quad 0 \le t \tag{11}$$

the completion time pdf for the service center of fig. 2 is

$$w_2(t) = \left(\sum_{j=1}^{K} q^{j-1}\mu^j t^{j-1}/(j-1)! - \sum_{j=1}^{K-1} q^j \mu^j t^{j-1}/(j-1)!\right)\exp(-\mu t) \tag{12}$$

The pdf given by eq. (12) contains three parameters which may be varied to find a best fit of eq. (9), the number of stages K, the service rate μ, and the probability q. In attempting this fit the entire domain of the pdf is relevant. It is desirable to obtain a better fit than obtained by matching only the expected values or variances of the queueing time distributions. An error measure can be defined over the entire domain as

$$\text{m.s.e.} = \int_0^\infty (w_1(t) - w_2(t))^2 \, dt \tag{13}$$

This result does not lend itself to analytic minimization procedures. However, the effect of varying the parameters of eq. (12) may be studied. The number of variable parameters can be reduced to two by constraining the optimum approximation, $w_2(t)$, to have the same mean as $w_1(t)$. The mean queueing (and service) time for the model is

$$t_{queue} = (1/\mu)(Kq^{K-1} + \sum_{j=1}^{K-1} jq^{j-1}(1-q)) \qquad (14)$$

Then if q or μ is set to a specific value the other is also determined.

2. FITTING A MODEL TO A DRUM

Now consider the problem of selecting K, μ and q to produce the best fit for a particular drum. A drum can be characterized by a rotation time, r and the number of sectors, m. The usage of the drum, measured by the expected sector queue length, will also effect the value of the parameters of the model.

The effect of the rotation time of the drum on selecting an optimum model is nil. The magnitude of the m.s.e. decreases for drums with longer rotation times but the optimum values for the number of stages and a choice of μ or q remain the same. To totally remove the effect of the rotation time define a normalized error:

n.m.s.e. = (m.s.e. of model)/(m.s.e. of one stage model)

The one stage model is just an exponential service time distribution with an infinite number of servers.

The effect of q on the magnitude of the n.m.s.e. is shown in fig. 3a and fig. 3b for light and moderate loading. These figures reveal that improvements to the drum completion time approximation are obtained with values of $q < 1$. With q equal to 1 the queueing time distribution becomes Erlang-K. Fig. 3a indicates that at light loading models with more stages are increasingly sensitive to changes in q in terms of how well they fit the desired pdf. Since the minimum n.m.s.e. appears to be approximately the same regardless of the choice of the number of stages some simplification in defining a model might be obtained by fixing the number of stages. If two stages are selected the sensitivity to q at light loading is minimized. Of course only considering a model with a fixed number of stages may cause some accuracy to be lost.

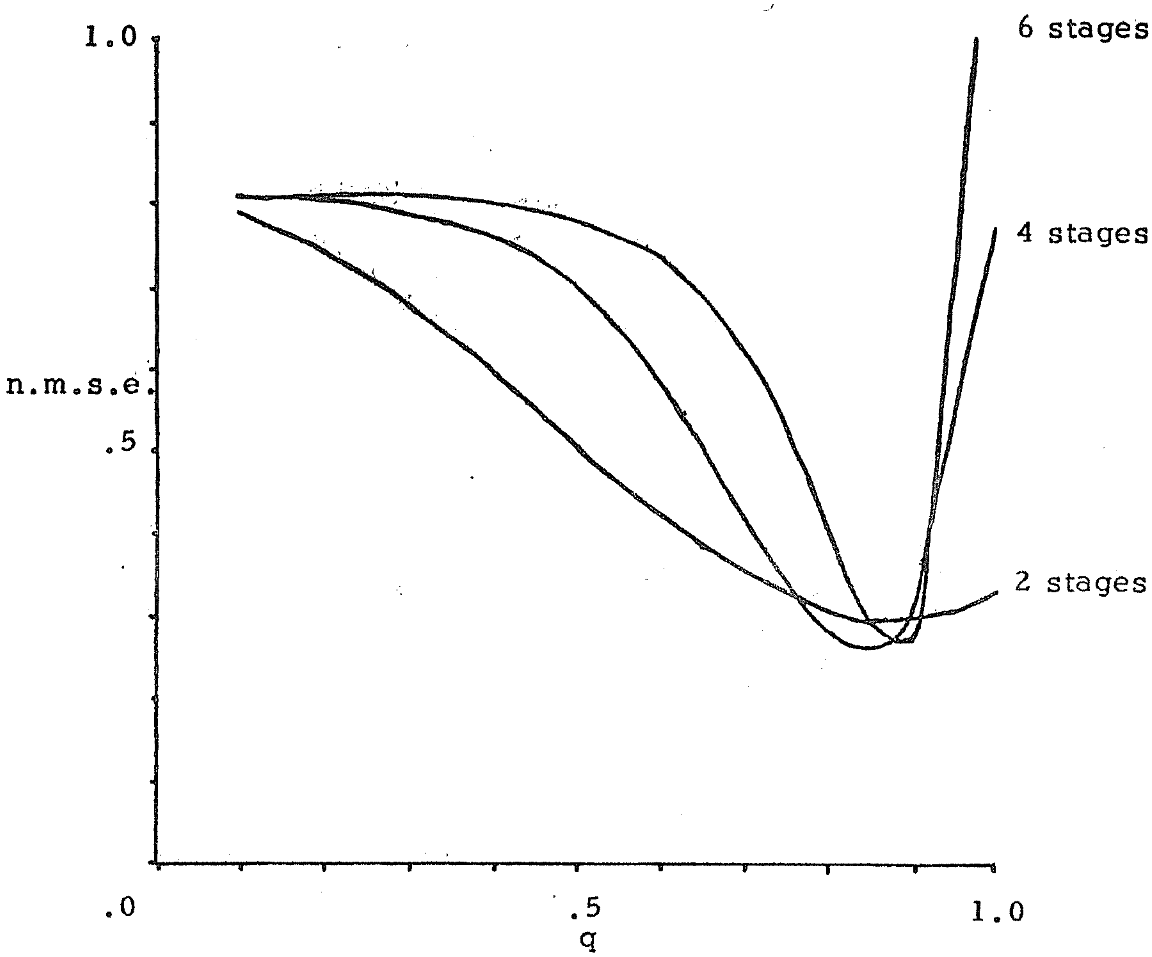

Fig. 3a: Normalized m.s.e. vs. 8
40 sectors
Exp. queue length = .128

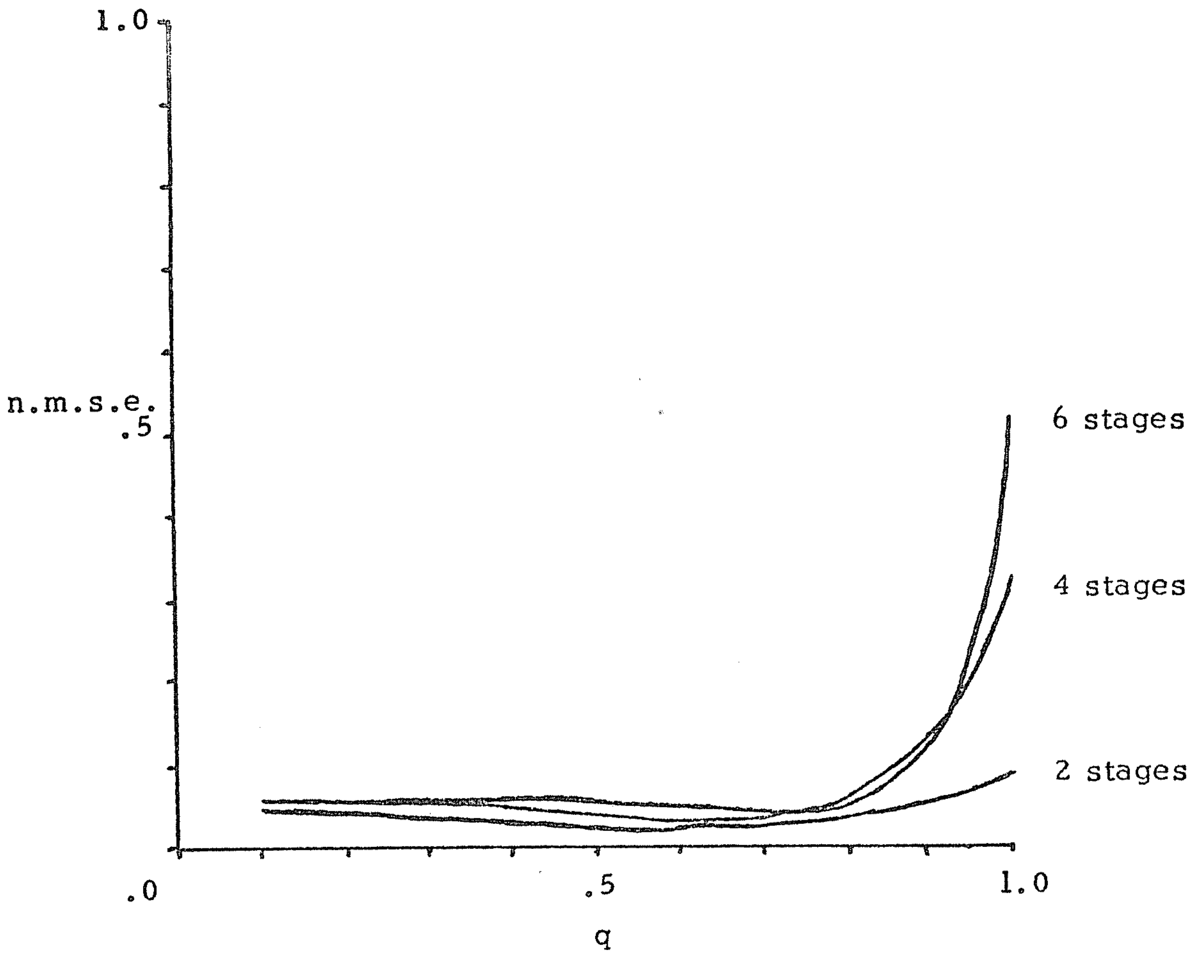

Fig. 3b: Normalized m.s.e. vs. q.
40 sectors
Exp. queue length = 1.157

Fig. 4 shows the effect of loading on the choice of q for models with different numbers of stages. Note that models with a greater number of stages are not as sensitive to changes in loading. Finally fig. 5 indicates how the choice of an optimum q for a given number of stages is affected by the number of sectors of the drum.

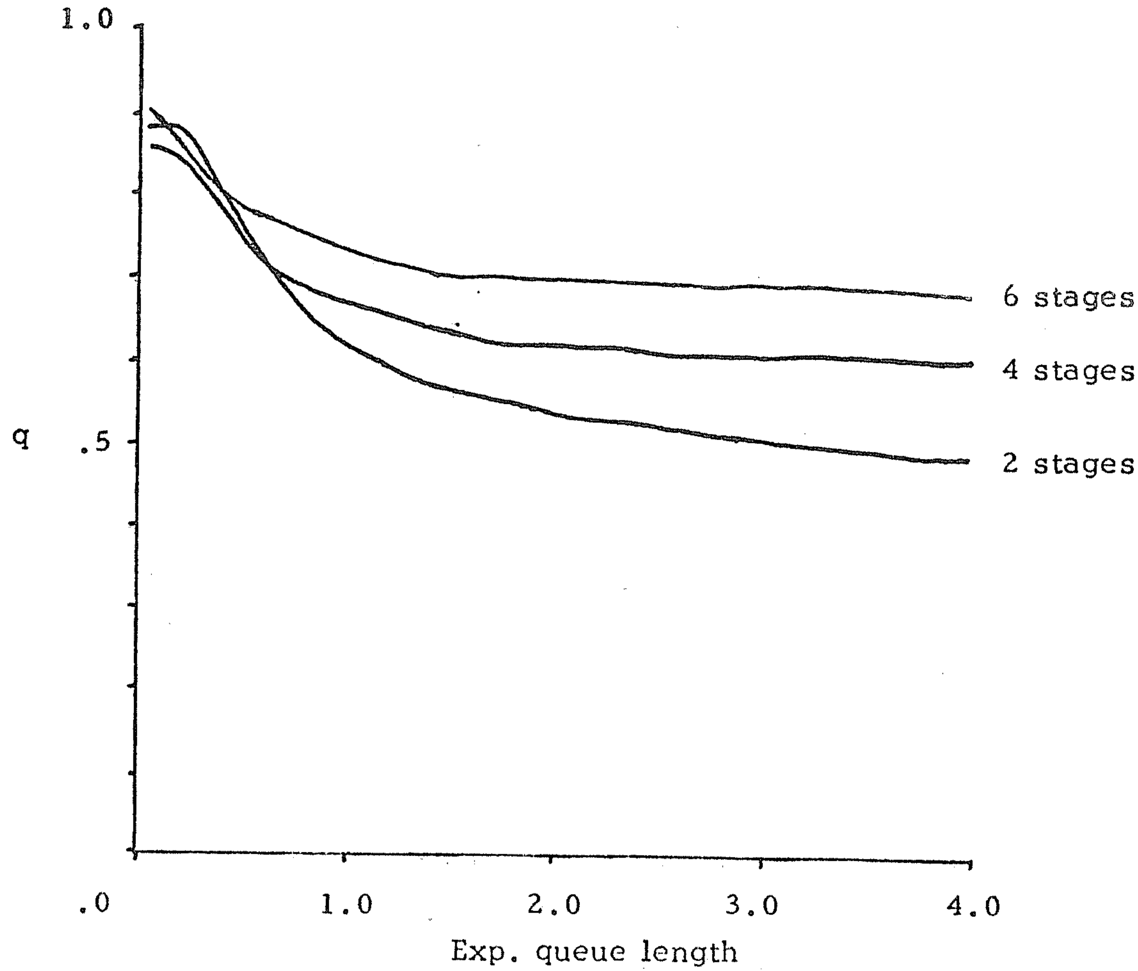

Fig. 4: Optimum q vs. loading
40 sectors

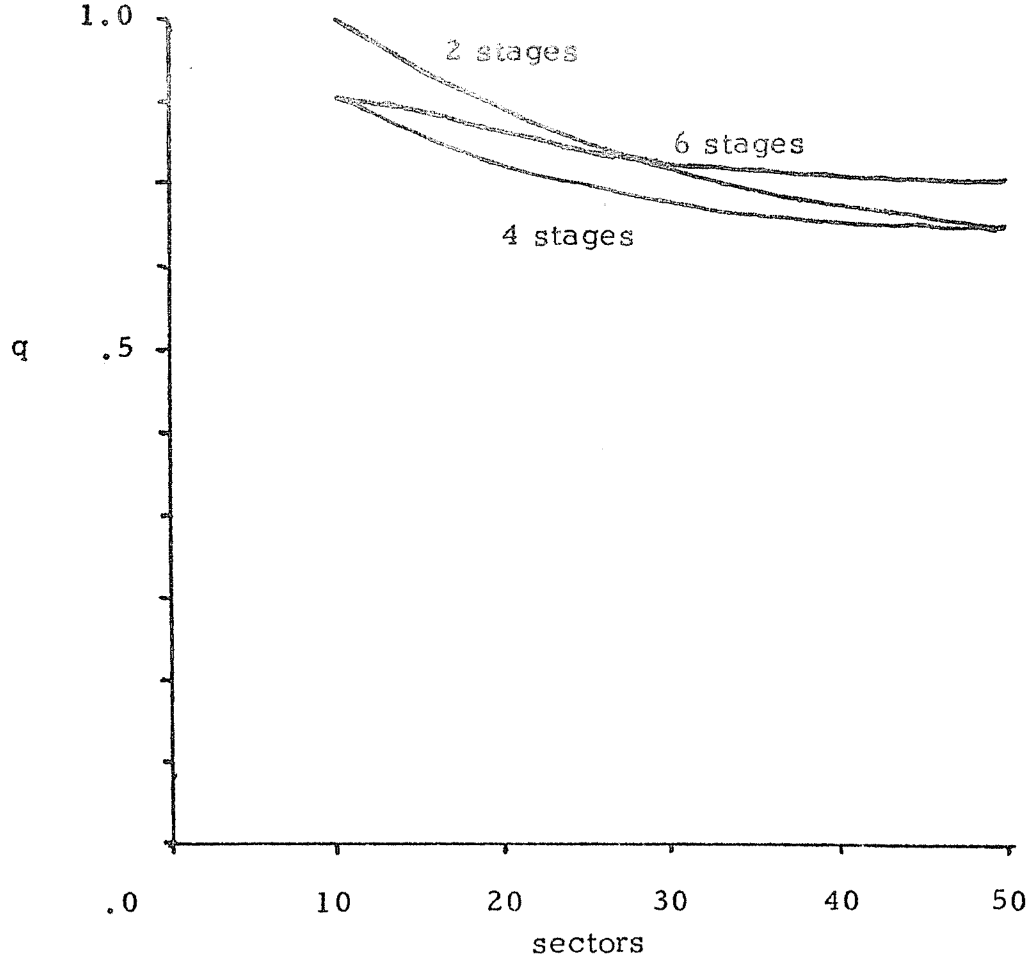

Fig. 5: Optimum q vs. sectors
Expected queue length = .36

The optimum model for a drum can vary in several aspects depending on the drum and its utilization. In a network model the utilization of the drum will not be known until the network has been solved. This suggests that the network will need to be solved repeatedly as the variables in the drum model are adjusted to reflect the correct utilization.

A practical aspect of such a solution is the minimization of the number of variables in the drum model. To this end it is of interest to consider the pdf obtained at optimal values of q and K and the pdf obtained by selecting values for K and q which give low, but not necessarily minimum values of the n.m.s.e. over a wide operating range. Fig. 6a and fig. 6b make just such a comparison. Also included is a more conventional drum model. This is an exponential service time distribution with mean equal to one half of the rotation time plus the transfer time. With FCFS queueing it represents the service presented by a single sector queue, and thus a service center would be needed for each sector.

As would be expected the staged approximations are more realistic than the exponential service distribution. This improvement increases with the loading of the drum. This is the effect of the increasing service time seen by customers queued at the drum. The exponential distribution does not react to this change. Also note the improvement gained at light loading by choosing the optimal configuration for the drum model. This is not seen at heavier loading since the optimum model is near the model chosen to be representative at all levels of loading.

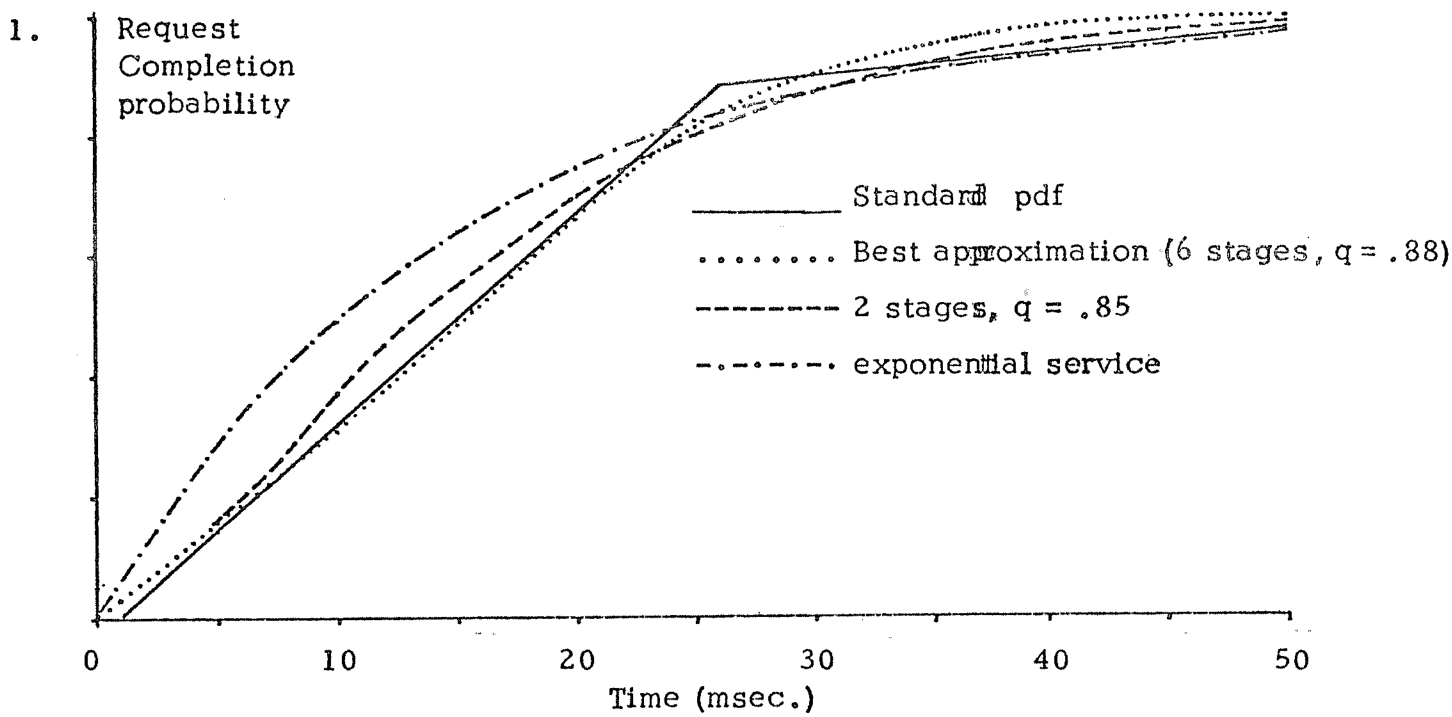

Fig. 6a: Drum model pdf approximations
40 sectors
Rotation time = 25 msec.
Exp. queue length = .128

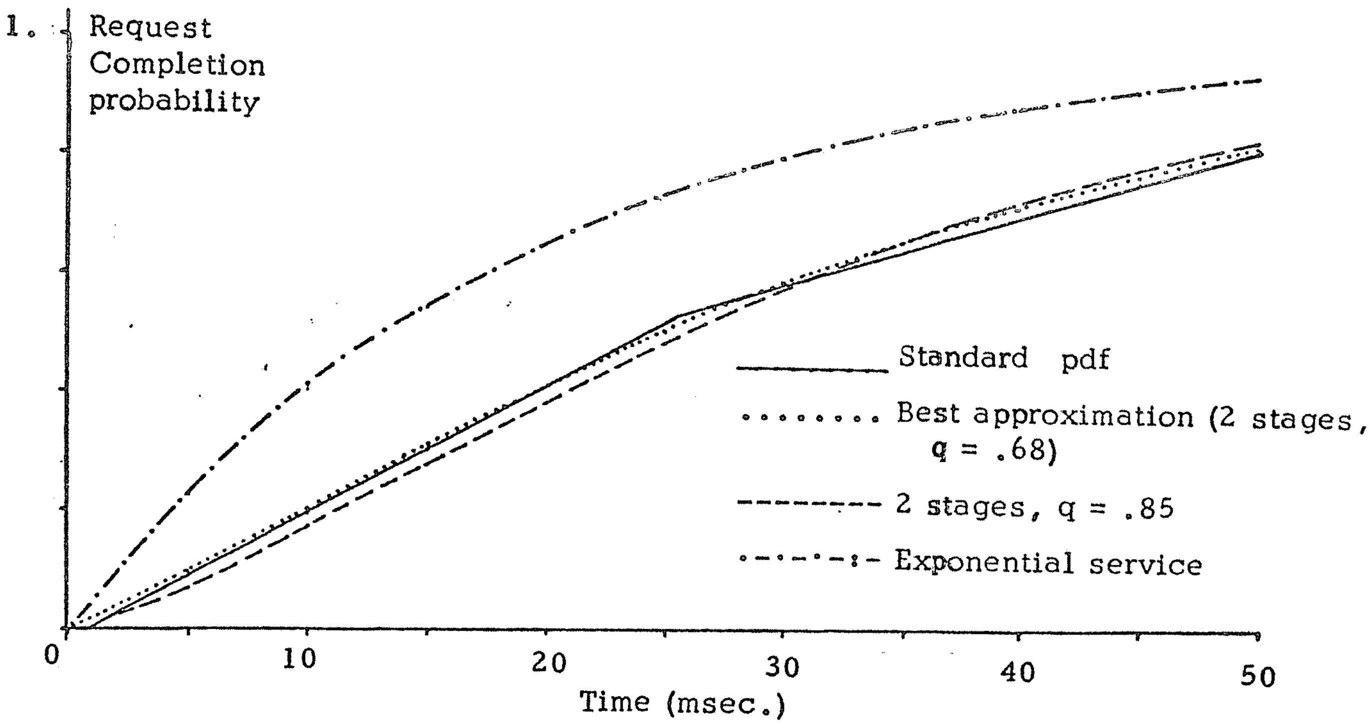

Fig. 6b: Drum model pdf approximations
40 sectors
Rotation time = 25 msec.
Exp. queue length = .737

3. NETWORK MODELS WITH QUEUE DEPENDENT SERVERS

The previous section developed a parameterized model for a paging drum. However the actual specification of the model's parameters required a priori knowledge of the expected queue length $E(n[i])$. Of course this measure will not be known until the network has been solved. This suggests an iterative approach to find an eventual solution.

An iterative approach has been used successfully by Teorey [T1] and Pinkerton and Teorey[P2] for single server models, and also by Pinkerton [P1]to solve a cyclic network. In this approach the mean service time or queueing time, t_{queue}, is expressed as a function of the expected queue length. In turn the expected queue length may be derived from the model as a function of the mean service time or queueing time. Let h be the function relating t_{queue} to $E(n_i)$.

$$t_{queue} = h(E(n_i)) \tag{15}$$

Once the function is found the model may be repeatedly solved until t_{queue} converges.

3.1 Convergence

Before considering algorithms for the iterative solution of closed queueing network models it is important to insure that these algorithms will indeed converge. Consider the model, with k stages, for a device with a queue dependent server in Figure 2. It is assumed that μ and q are the same for all classes. Note that either μ or t_{queue}, given by

$$t_{queue} = (1/\mu)(kq^{k-1} + \sum_{j=1}^{k-1} jq^{j-1}(1-q)) \tag{16}$$

may serve as the variable of iteration. Consider the nature of the marginal queue length probability distribution at the queue dependent service center. Let the vector $p = (p_0, p_1, \ldots, p_N)$ represent the marginal distribution where p_j is the probability of exactly j customers being present in the queue at the service center. p_j for service center M may be written:

$$p_j = \frac{\sum_{S(N,M)\ \&\ n_M=j} \prod_{m=1}^{M} F_m(Z_m)}{\sum_{S(N,M)} \prod_{m=1}^{M} F_m(Z_m)} \tag{17}$$

Define the auxilliary function $b(n_M)$ by

$$F_M(Z_M) = (1/\mu)^{n_M}\ b(n_M) \tag{18}$$

This factorization is possible regardless of the number of classes or number of stages at the service center. p_j may now be written to reflect the effect of μ on its value.

$$p_j = \frac{(i/\mu)^j \sum_{S(N,M)\ \&\ n_M=j} b_M(j) \prod_{m=1}^{M-1} F_m(Z_m)}{\sum_{k=0}^{N} (1/\mu)^k \sum_{S(N,M)\ \&\ n_M=k} b_M(k) \prod_{m=1}^{M-1} F_m(Z_m)} \tag{19}$$

Lemma 1. For the marginal probability distribution given by $p = (p_0, p_1, \ldots, p_N)$ and equation (19) there exists a k, $0 \le k \le N$, such that

(1) If μ is decreased p_j is not decreased for $j \geq k$ and not increased for $j < k$.

(2) If μ is increased p_j is not increased for $j \geq k$ and not decreased for $j < k$.

<u>Proof.</u> Consider case (1). Let $p' = (p'_0, p'_1, \ldots, p'_N)$ be a new distribution and $G'(N)$ a new normalizing constant resulting from a decrease in μ to μ'. A decrease in μ will increase all terms of $G(N)$ for $k > 0$ and leave the initial term unchanged. Therefore $G'(N) > G(N)$. p'_j may be written in terms of p_j.

$$p'_j = (G(N)/G'(N))(\mu/\mu')^j \, p_j \tag{20}$$

Since $G(N)/G'(N)$ is independent of j and $\mu > \mu'$

$$(\mu/\mu')^{j+1} > (\mu/\mu')^j \geq 1 \tag{21}$$

and

$$p'_{j+1}/p_{j+1} > p'_j/p_j \tag{22}$$

Both p_j and p'_j are density functions, thus

$$\sum_{j=0}^{N} p_j = \sum_{j=0}^{N} p'_j = 1 \tag{23}$$

From equation (20) with $j = 0$, $p'_0 < p_0$ and thus

$$\sum_{j=1}^{N} p'_j > \sum_{j=1}^{N} p_j \tag{24}$$

Equation (24) implies the existence of some j such that $p_j' > p_j$. From equation (22) for any $L \geq j$ it must also be true that $p_L' > p_L$. k is just the minimum j such that $p_j' > p_j$. Case (2) proceeds with identical arguments. Q.E.D.

The convergence of t_{queue} can now be shown.

Theorem 1. The iterative solution of a closed queuing network model with a single queue dependent server is convergent provided (1) the function h of equation (15) is monotonic over the interval $0 \leq E(n_i) \leq N$ and (2) the initial value of t_{queue}, t_0, is in the interval $h(0) \leq t_0 \leq h(N)$.

Proof. Since $E(n_i)$ is an expected value it is certainly true that $0 \leq E(n_i) \leq N$ and thus for the iterative variable t_{queue}:

$$h(0) \leq t_{queue} \leq h(N)$$

Since the sequence of values of t_{queue} is bounded it is sufficient to show the sequence is monotonic, i.e., the sign of the difference of any two consecutive values is the same as the sign of the difference of the initial guess and the first iteration.

Without loss of generality assume h is nondecreasing over the interval $(0,N)$ and the initial iteration causes t_{queue} to increase. μ must be decreased to make this adjustment. Let $p = (p_1, \ldots, p_N)$ be the original marginal distribution and $p' = (p_1', \ldots, p_N')$ the marginal distribution obtained with the new smaller value of μ. From Lemma 3.1 there exists a k such that

$$d = \sum_{j=0}^{k-1} p_j - p_j' = \sum_{j=k}^{N} p_j' - p_j > 0 \tag{25}$$

Let $E'(n_M)$ be the new expected queue length.

$$E'(n_M) - E(n_M) = \sum_{j=0}^{N} jp_j' - \sum_{j=0}^{N} jp_j$$

$$= \sum_{j=0}^{k-1} j(p_j' - p_j) + \sum_{j=k}^{N} j(p_j' - p_j)$$

$$> \sum_{j=k}^{N} k(p_j' - p_j) - \sum_{j=0}^{k-1} k(p_j - p_j')$$

$$> dk - dk = 0$$

Since h is nondecreasing t_{queue} will again increase. The proof of the cases for nonincreasing h and an initial decrease in t_{queue} follows immediately in the same manner. Q.E.D.

Since t_{queue} is the iterative variable in Theorem 1 the number of stages or value of q may be altered and the iterative procedure resumed with a new initial value.

3.2 Algorithms for the Solution

Algorithms are presented for two cases. The first algorithm employs Algorithm 2 and is thus subject to the same constraint. All customers must have a non-zero probability of becoming members of all classes. The second algorithm is designed to yield an efficient iterative procedure when Algorithm 2 is not applicable and the normalizing constant must be constructed with a brute force approach.

Algorithm 3, presented below, performs an iterative calculation on a queueing network model with a single queue dependent server. Consider Algorithms 1 and 2. The parameters of the Mth service center are never used until the final pass of both algorithms. To iterate on a parameter of the Mth service center Algorithm 3 uses Algorithm 2 to solve a network of M-1 service centers. The iteration is then performed with the Mth service center finding only $g_2(M+1, j)$, $0 \leq j \leq N$, at each iteration. After the procedure converges the final values of $g_2(i,j)$, $1 \leq i < M$ and $0 \leq j \leq N$, are found. The amount of additional computation due to the iteration is independent of the number of service centers in the model.

Algorithm 3

1. The queue dependent server is the Mth service center.
2. Calculate $f_i(n_i)$ for $1 \leq i \leq M-1$ and $0 \leq n_i \leq N$.
3. $g_2(1,j) \Leftarrow f_2(j)$; $g_2(2,j) \Leftarrow f_1(j)$ for $0 \leq j \leq N$; $x \Leftarrow 2$.
4. $x' \Leftarrow x - 1$.
5. If $x' = 0$ or $x = 2$ go to step 6 else

$$g_2(x',j) \Leftarrow \sum_{i=0}^{j} f_x(i) g_2(x', j-i) \quad \text{for } j = N, N-1, \ldots, 0$$

$x' \Leftarrow x' - 1$, repeat step 5.

6.

$$g_2(x+1,j) \Leftarrow \sum_{i=0}^{j} f_x(i)\, g_2(x,j-i) \quad \text{for } j = N, N-1, \ldots, 0$$

$x \Leftarrow x + 1$. If $x < M$ go to step 4, otherwise step 7.

7. Select an initial value of t_{queue}.

8. Calculate $f_M(n_M)$ for $0 \le n_M \le N$.

9.

$$g_2(M+1,j) \Leftarrow \sum_{i=0}^{j} f_M(i)\, g_2(M,j-i) \quad \text{for } j = N, N-1, \ldots, 0$$

10. Solve for a new value of t_{queue} using the Mth and M+1st rows of g_2 and the function h of equation (8). If the new t_{queue} and the previous value differ by less than some specified parameter continue to step 11, otherwise repeat steps 8, 9, 10 with the new value of t_{queue}.

11. $x' \Leftarrow M - 1$.

12. If $x' = 0$ stop else

$$g_2(x',j) \Leftarrow \sum_{i=0}^{j} f_M(i)\, g_2(x',j-i) \quad \text{for } j = N, N-1, \ldots, 0$$

$x' \Leftarrow x' - 1$ go to step 12.

Not all networks with more than a single class of customer will satisfy the constraint necessary to use Algorithm 3. In the case of such networks the normalizing constant and summations needed to find the marginal probability distributions (see equation (17)) may be found by evaluating each term of the appropriate summations. To make an iterative solution feasible for moderate size models it is important to short cut this procedure, at least during the iteration.

Using the function $b_M(n_M)$ defined in equation (18) write the normalizing constant as a function of the service rate at the queue dependent server, μ. Again this is the Mth service center.

$$G(N) = \sum_{k=0}^{N} (1/\mu)^k \sum_{S(N,M)\ \&\ n_M=k} b_M(k) \prod_{m=1}^{M-1} F_m(Z_m)$$
$$= \sum_{k=0}^{N} (1/\mu)^k T_k .$$

Algorithm 4 evaluates N+1 constants, T_k, independent of μ and then iterates using these constants and succesive values of μ.

<u>Algorithm 4</u>

1. Initialize the temporary array. $T_i \Leftarrow 0,\ 0 \le i \le N$, and pick an initial μ.
2. For $k = 0,1,\ldots,N$

$$T_k \Leftarrow \sum_{S(N,M)\ \&\ n_M=k} b_M(n_M) \prod_{m=1}^{M-1} F_m(Z_m) .$$

3. Find the normalizing constant.

$$G(N) = \sum_{i=0}^{N} (1/\mu)^i T_i$$

4. Find the expected queue length at the queue dependent server.

$$E(n_M) = (1/G(N)) \sum_{i=0}^{N} i(1/\mu)^i T_i$$

5. Solve for a new value of t_{queue}. If the new t_{queue} and the previous value differ by less than some specified parameter continue to step 6, otherwise repeat steps 3 and 4 with a new μ derived from the new t_{queue}.

6. Evaluate the necessary summations to find the remaining marginal probabilities as given by equation (17).

In Algorithm 4 as in Algorithm 3 the amount of extra computation for the iteration is independent of the number of service centers in the network.

3.3 Open Networks

Hine and Fitzwater [HI] have shown that an open network developed from the model of Muntz and Baskett may be factored and treated essentially as a collection of single service centers. In this case the iterative approach of Teorey and Pinkerton [T1] may be applied. The function H is derived using Little's result.

$$t[queue] = (1/\lambda)\ E(N[I]) \tag{26}$$

λ is the mean arrival rate of customers at service center I. The iteration involves only the single service center. Since the expected queue length is essentially unbounded some other bounded variable such as utilization, must be identified to prove convergence.

IV. Conclusion

This paper has demonstrated that results of analyses of various peripheral devices may be applied to queueing network models of multiprogrammed computer systems. The resulting device models should improve the representation of the device in system models and lead to more accurate results from these models. Although the specific model presented was a paging drum with sector queueing the technique of approximating a service or queueing time distribution with a network of stages can be extended to other peripheral devices and scheduling disciplines.

An algorithm for the solution of a queueing network model with a queue dependent service time distribution was also presented. The algorithm allows the distribution of the service center representing this device to be adjusted to reflect the loading at various levels of multiprogramming. The amount of computation added by the iteration, while dependent on the level of multiprogramming, is independent of the number of service centers in the network model. This should encourage the use of more accurate modelling of devices in large network models.

The extension of algorithm 3 and theorem 1 to allow more than a single queue dependent server would be a welcome advance. However there is a serious question concerning the effect of a temporarily incorrect value of μ at one service center causing itself or another to move to an extreme, acting as a bottleneck. If Theorem 1 was valid in this situation the service center would remain a bottleneck even though it might not have been had it not been for the temporary imbalance during the iterative process.

An Implementation

A Fortran program for the iterative solution of a closed queueing network with the paging drum model developed in section 2 was implemented.

This program was used to check convergence rates and demonstrate the practicality of the solution technique. Many complex problems may now be studied with only minor amounts of computations. Since no closed form was known for identifying the optimum parameter values at a specific operating point, interpolation from a table was employed. A program was written to accept a drum description and an operating range. Using Coffman's analysis a sector queue length PDF was determined. From this information expected queue lengths and queueing times were produced.

The program continues to tabulate values of the N.M.S.E. at points over the operating range for a variety of values of K, the number of stages, and q, the probability of continuing to the next stage. The suitable values of K and q are readily obtained from the tabulation. The data is tabulated with specific operating points defined by the expected queue length. Values of K, q and expected queueing time are obtained by linear interpolation. With K and q selected, μ is obtained by

$$\mu = (1/t[\text{queue}])(Kq^{K-1} + \sum_{j=1}^{K-1} jq^{j-1}(1-q)) \qquad (27)$$

μ is plotted as a function of the expected queue length (at each sector queue) in figure 7. To achieve maximum accuracy the operating points from which the table is constructed should be denser where the plot is non-linear. μ is bounded because the minimum expected queueing time is one half a rotation plus the transfer time. The relationship shown in figure 7 is monotone satisfying the convergence requirement.

Using linear interpolation, algorithm 3 converged with accuracy of .000001 in four to eight iterations. The number of iterations was only slightly affected by the accuracy of the initial values.

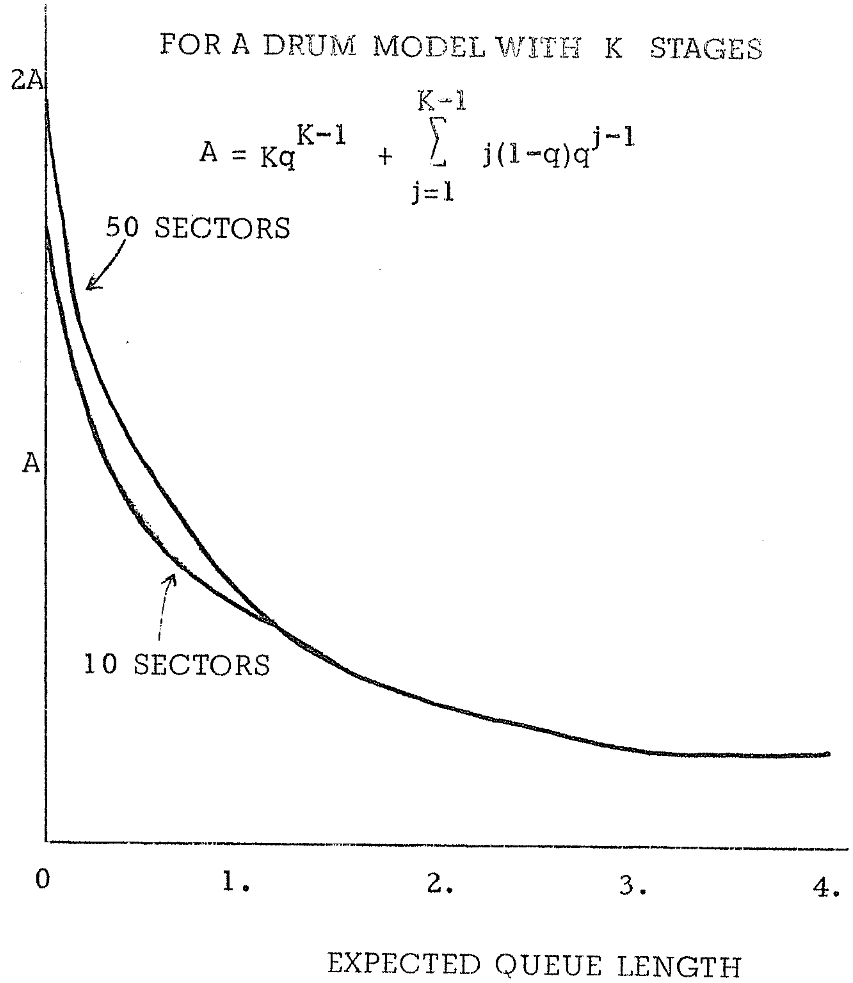

Fig. 7: u vs. loading

BIBLIOGRAPHY

B1 BASKETT, F., "The Dependence of Computer System Queues Upon Processing Time Distribution and Central Processor Scheduling," 3rd Symposium on Operating Systems Principles, Stanford Univ., 1971, pp. 109-113.

B2 BASKETT, F., and GOMEZ, F. P., "Processor Sharing in a Central Server Queueing Model of Multiprogramming with Applications, Proceedings of the 6th Annual Princeton Conference on Information Sciences and Systems, Princeton Univ., March 1972, pp. 598-603.

B3 BASKETT, F. and MUNTZ, R., "Queueing Network Models with Different Classes of Customers," 6th Annual IEEE Computer Society International Conference, 1972, pp. 205-209.

B4 BUZEN, J., "Queueing Network Models of Multiprogramming," Ph.D. Thesis, Division of Engineering and Applied Science, Harvard Univ., Cambridge, Mass., (1971), (ESD-TR-71-3451).

B5 BUZEN, J. P., "Computational Algorithms for Closed Queueing Networks with Exponential Servers," CACM, Vol. 16, No. 9, (September 1973), pp. 527-531.

C1 CHANDY, K. M., "The Analysis and Solution for General Queueing Networks," 6th Annual Princeton Conference on Information Systems and Science, Princeton Univ., 1972, pp. 224-228.

C2 COFFMAN, E. G. Jr., "Analysis of a Drum Input/Output Queue Under Scheduled Operation in a Paged Computer System," JACM, Vol. 16, No. 1, (January 1969(pp. 73-90.

D1 DENNING, P. J., "A Note on Paging Drum Efficiency," Computing Surveys, Vol. 4, No. 1, (March 1972), p. 1.

G1 GORDON, W. J. and NEWELL, G. F., "Closed Queueing Systems with Exponential Servers," Operations Research, Vol. 15 (1967), pp. 254-265.

H1 HINE, J. H. and FITZWATER, D. R., "A Note on Queueing Network Models," Draft in preparation.

M1 MOORE, C. G. III, "Network Models for Large-Scale Time-Sharing Systems," Tech. Report 71-1, Dept. of Industrial Engineering, Univ. of Michigan, April 1971.

M2 MUNTZ, R. and BASKETT, F., "Open, Closed and Mixed Networks of Queues with Different Classes of Customers," Stanford Univ., August 1972.

P1 PINKERTON, T. B., "Program Behavior and Control in Virtual Storage Computer Systems," CONCOMP TR4, Univ. of Michigan, 1968.

P2 PINKERTON, T. B. and TEOREY, T. J., "A Simple Iterative Technique in Computer System Modelling," (submitted to CACM).

T1 TEOREY, T. J., "Properties of Disk Scheduling Policies in Multiprogrammed Computer Systems," Proc. of 1972 FJCC, pp. 1-11.

BIBLIOGRAPHIC DATA SHEET	1. Report No. WIS-CS-200-74	2.	3. Recipient's Accession No.

4. Title and Subtitle

DRUM MODELS USING AN ITERATIVE SOLUTION FOR CLOSED QUEUEING NETWORKS

5. Report Date: January 1974

6.

7. Author(s)

J. H. Hine and D. R. Fitzwater

8. Performing Organization Rept. No. 200

9. Performing Organization Name and Address

COMPUTER SCIENCES DEPARTMENT
The University of Wisconsin
1210 West Dayton Street
Madison, Wisconsin 53706

10. Project/Task/Work Unit No.

11. Contract/Grant No.

12. Sponsoring Organization Name and Address

13. Type of Report & Period Covered

14.

15. Supplementary Notes

16. Abstracts

An algorithm by Buzen to find the normalizing constant and marginal probabilities in a closed queueing network is generalized to the model of Muntz and Baskett. A technique involving the iterative solution of a closed queueing network to implement queue dependent completion time distributions is presented. An accurate model of a paging drum for queueing networks is given as an example.

17. Key Words and Document Analysis. 17a. Descriptors

Modeling
queueing networks
operating systems
multiprogramming

17b. Identifiers/Open-Ended Terms

CR Categories: 4.39, 3.89, 5.5.

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report)	21. No. of Pages
Available to public	UNCLASSIFIED	43
	20. Security Class (This Page) UNCLASSIFIED	22. Price

FORM NTIS-35 (10-70)

USCOMM-DC 40329-P71