APPROXIMATING THE MEAN TIME IN SYSTEM IN A MULTIPLE-SERVER QUEUE THAT USES THRESHOLD SCHEDULING

RANDOLPH NELSON
IBM Corporation, Yorktown Heights, New York

DON TOWSLEY
University of Massachusetts, Amherst, Massachusetts

(Received November 1985; revisions received April, September 1986; accepted September 1986)

In this paper we consider a queueing system consisting of a single queue with multiple exponential servers with different servicing rates. We assume that arrivals to the queue come from a Poisson source and are scheduled according to a threshold policy. Since determining the exact mean time in system appears to be difficult, we present an approximation that yields results very close to those obtained from simulation.

In this paper we study a multiple-server system sharing a common queue; the servers have different service rates. The control and analysis of such systems pose very interesting problems which, however, have received little attention. We will present an approximation method for obtaining the expected response time of a customer in any such system that uses a *threshold* scheduling policy.

Our motivation for studying threshold policies lies in the fact that these policies optimize performance of the system for a number of performance metrics. For example, Lin and Kumar (1984) show that a threshold schedule minimizes the mean response time of customers in a two-server system with Possion arrivals and exponential service times. Agrawala et al. (1984) consider a system with no arrivals, a fixed and finite number of customers in a queue with multiple servers, and exponential service times. They show that a threshold policy minimizes the total time required to process all the customers in the system. We have shown in another paper (Nelson and Towsley 1985) that a threshold policy maximizes the expected number of departures from a multiple-server system between successive arrivals when the arrival process is Poisson and service times are exponentially distributed. Finally, Ibe (1982) considered a multiple-server system in which customers were always assigned to the fastest available processor whenever a processor became idle. This policy is a degenerate form of a threshold policy in which all thresholds are identically zero. In his study, Ibe presented an approximation model with which to evaluate the performance of the system.

We will focus on multiple-server systems with an arbitrary number of servers, a Poisson arrival process and exponential service times. Queueing systems of this type are useful models for various computer system and communication network configurations—for instance, a multiprocessor system having different processors. In this case a job entering the queue (a central dispatch) would be assigned to one of the available processors according to some scheduling discipline. In another example, nodes of a communications network could be linked to one another by several channels of varying capacities (for example, a transmission group in a Systems Network Architecture (SNA) network). Messages passing through such nodes would be scheduled on the available links.

The paper is organized as follows: In Section 1 we describe the queueing system we consider, define a threshold scheduling policy, and discuss why finding an exact solution for the mean time in the system is difficult. In Section 2 we derive our approximation, and in Section 3 we compare the approximation with simulation results. Section 4 contains our conclusions.

1. The Model

Consider a set of N heterogeneous servers $\{P_1, P_2, \ldots, P_N\}$ that serve a common queue. Jobs arrive to this queue according to a time-invariant Poisson process with rate λ and are served in a first-come first-served

Subject classification: 684 mean time in system, 696 threshold scheduling.

0030-364X/87/3503-0419 $01.25

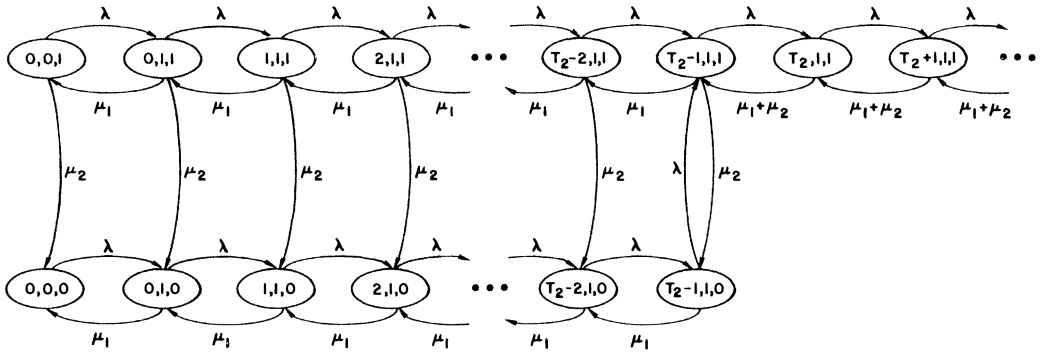

Figure 1. State diagram for $N = 2$.

(FCFS) manner. The service time for a job executed on server P_i is an exponentially distributed random variable with mean $1/\mu_i$ for $i = 1, 2, \ldots, N$. We further assume that the servers are ordered so that $\mu_1 \geq \mu_2 \geq \ldots \geq \mu_N$. We assume that the servers are scheduled according to a *threshold* discipline. Specifically, for some thresholds $1 = T_1 \leq T_2 \ldots \leq T_N < \infty$, the policy schedules a job from the queue to an idle server P_i only if T_i is smaller than or equal to the threshold of any other idle server *and* if the queue length is greater than or equal to T_i.

2. Analysis

Let $x = (m, c)$ denote the state of the system; m represents the number of jobs waiting to be processed, and $c = (c_1, c_2, \ldots, c_N)$ where $c_i = 1$ if the ith server P_i is busy and $c_i = 0$ otherwise. It is clear that (m, c) forms a Markov process. Figure 1 shows the state diagram for the case in which $N = 2$. In this paper we assume that the Markov process is ergodic, and we are concerned with calculating the mean value of the stationary distribution of the time spent in the system. This value can be expressed as

$$W = \frac{1}{\lambda}\left(\bar{L}_q + \sum_{i=1}^{N} \rho_i\right), \tag{1}$$

where $\bar{L}_q$ is the average queue length (not including customers in service) and ρ_i is the utilization of the ith server. The difficulty in analyzing this Markov process exactly lies in the complexity of the state diagram. Specifically, there are 2^N possible states corresponding to a configuration of $(0, c)$ for all possible distinct values of c. In our approximation, we collapse the multidimensional process into a birth-dealth process as shown in Figure 2; the state of the system is the number of waiting customers. Note that some of the arrival transition rates are multiplied by a parameter p_i (the figures do not show a value of p_i if it equals 1). The parameters p_i for $i = 0, 1, \ldots$ are crucial to our approximation. For $i \neq T_j - 1$ for any j, we set $p_i = 1$. For $i = T_j - 1$ and $1 \leq j \leq N$, p_i is defined to be the probability that an arrival to a waiting line of $T_j - 1$ customers finds P_j busy. Only in this case will the queue length increase by 1, since if P_j is idle when the arrival occurs, the customer at the head of the queue will be scheduled immediately on P_j, thus leaving $T_j - 1$ customers still waiting for service. Given these parameters, one can use standard analysis to solve the birth-death equations and calculate an approximation to $\bar{L}_q$ given in Equation 1. To obtain approximations for ρ_i, we define f_i to be the stationary conditional probability that P_i is busy, given that the queue length is less than T_i. Since we assume the Markov process is ergodic, we can calculate f_i by calculating the proportion of time that P_i is busy when the queue length

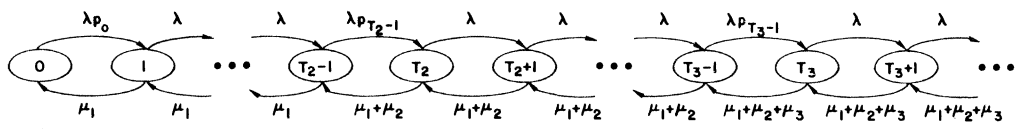

Figure 2. State diagram for birth-death approximation, unique thresholds.

is less than T_i. We can then write

$$p_i = P[L_q \geq T_i] + f_i(1 - P[L_q \geq T_i]) \quad 1 \leq i \leq N. \tag{2}$$

Knowing p_i and f_i, one can substitute (2) in Equation 1 to approximate the mean time in the system. In the remainder of this section, we give a method to calculate the parameters P_{T_i-1}, and f_i for $i = 1, 2, \ldots, N$.

2.1. Case 1: All Thresholds Different

In this section we show how we calculate the p_is and the f_is under the assumption that all thresholds are unique. We first state a useful lemma.

Lemma. *Let a Markov process be contained on the state space* $1, 2, \ldots, r, r+1, \ldots, r+s$, *having the infinitesimal generator matrix G given by*

$$G = \begin{bmatrix} A & B \\ 0 & 0 \end{bmatrix},$$

where A is an $r \times r$ *matrix and B is an* $r \times s$ *matrix. Then the* (i, j) *entry of* $-(A^{-1})$ *is the expected amount of time that the process spends in the transient state j starting from the transient state i,* $1 \leq i, j \leq r$ *before being absorbed in states* $r+1, r+2, \ldots, r+s$.

Neuts and Meier (1981) proved this lemma. The approximation uses it to calculate the parameters p_i and f_i by analyzing a sequence of transient Markov processes. Specifically, to derive P_{T_i-1} and f_i we define states (j, P_i busy) and (j, P_i idle) for $0 \leq j \leq T_i - 1$, and state (T_i, P_i busy) where the first index is the number of customers waiting in the queue. We then view the system from the time it makes a transition from (T_i, P_i busy) to ($T_i - 1$, P_i busy)—it cannot go to ($T_i - 1$, P_i idle)—to the time it returns to (T_i, P_i busy) as a transient Markov process with transient states (j, P_i busy), (j, P_i idle) for $0 \leq j \leq T_i - 1$ and absorbing state (T_i, P_i busy). The parameter T_i is calculated as the average amount of time spent in the transient states during which P_i is busy, divided by the average total amount of time spent in the transient states. The parameter p_{T_i-1} is calculated by finding the average time spent in a state that has $T_i - 1$ waiting customers and P_i is busy, and dividing this quantity by the average time the system has $T_i - 1$ customers waiting.

Let us describe this procedure in more detail for two cases, $T_1 = 1$ and $T_j > 1$ for $1 \leq j \leq N$.

Case 1. $T_1 = 1$: Consider the absorbing Markov process shown in Figure 3. State values in this figure are the number of jobs waiting in the queue which for this case is zero. The upper level represents those states for which P_1 is busy and the lower level those in which this processor is idle. In the original birth-death model, any transitions from state 1 to state 0 will necessarily have P_1 busy. Then, using the lemma with absorbing state (1, P_1 busy), and ordering the states in the generator matrix G as (0, P_1 busy), (0, P_1 idle), (1, P_1 busy), we find that the generator is given by

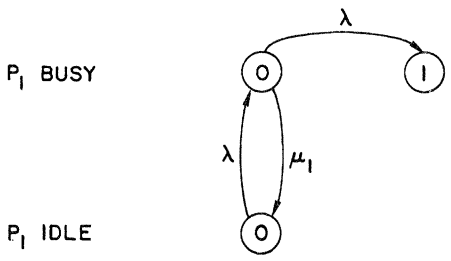

Figure 3. State diagram used to calculate p_0 and f_1.

$$G = \begin{bmatrix} -(\lambda + \mu) & \mu & \lambda \\ \lambda & -\lambda & 0 \\ 0 & 0 & 0 \end{bmatrix}.$$

By defining

$$A = \begin{bmatrix} -(\lambda + \mu) & \mu \\ \lambda & -\lambda \end{bmatrix}$$

and applying the lemma, we obtain

$$f_1 = p_0 = \frac{A^{-1}(0, 0)}{A^{-1}(0, 0) + A^{-1}(0, 1),}.$$

Case 2. $T_j > 1$: In Figure 4 we show the transition diagram in which the upper level of states corresponds to states for which P_j is busy. Assume that we have calculated p_{T_i-1} and f_i for $i = 1, 2, \ldots, j-1$; we will now show how to calculate p_{T_j-1} and f_j. If we order the states as (i, P_j busy) for $i = 0, 1, \ldots, T_1, \ldots, T_j - 1$, followed by ($i$, P_j idle) for $i = 0, 1, \ldots, T_1, \ldots, T_j - 1$, followed by the absorbing state (T_j, P_j busy), then the infinitesimal generator, G, is given by

$$G = \begin{bmatrix} A_j & \Lambda_j \\ 0 & 0 \end{bmatrix}, \tag{3}$$

where Λ_j is a $T_j \times 1$ column vector of all zeros except for the (T_j, 1) element, which equals λ. A_j is given by

$$A_j = \begin{bmatrix} M_j & \mu_j I_j \\ Z_j & M_j - \mu_j I_j \end{bmatrix},$$

and I_j is a $T_j \times T_j$ identity matrix, Z_j is a $T_j \times T_j$ matrix of all zeros except for the (T_j, T_j) entry, which

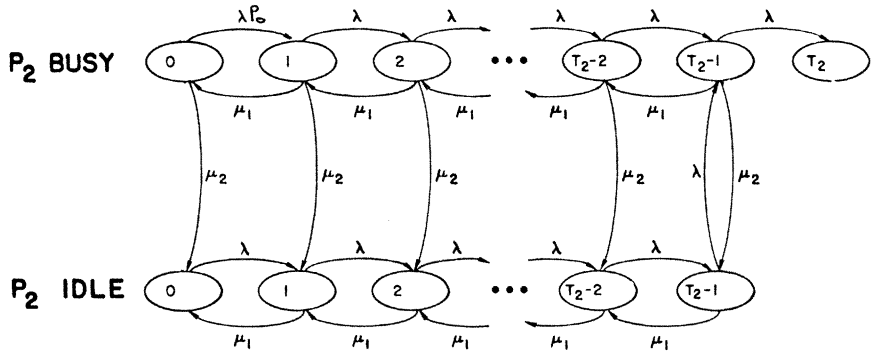

Figure 4. State diagram used to calculate p_{T_j-1} and f_j.

is λ, and M_j is a tridiagonal $T_j \times T_j$ matrix having elements, $1 \leq k \leq j$,

$$M_j(i, m) = \begin{cases} -(\lambda p_0 + \mu_j) & i = m = 0, \\ -\left(\lambda + \mu_j + \sum_{n=1}^{M(i)} \mu_n\right) & \\ & i = T_j - 1, \quad m = T_j - 1, \\ -\left(\lambda p_i + \mu_j + \sum_{n=1}^{M(i)} \mu_n\right) & \\ & i = 0, 1, \ldots, T_j - 2, \quad m = i, \\ \lambda p_i & i = 0, 1, \ldots, T_j - 2, \quad m = i + 1, \\ \sum_{n=1}^{M(i)} \mu_n & i = 1, 2, \ldots, T_j - 1, \quad m = i - 1, \\ 0 & \text{otherwise,} \end{cases} \tag{4}$$

where we define $M(i) = \max\{k: i \geq T_k\}$.

We calculate p_{T_j-1} and f_j as

$$f_j = \frac{\sum_{k=0}^{T_j-1} A_j^{-1}(T_j - 1, j)}{\sum_{j=0}^{2T_j-1} A_j^{-1}(T_j - 1, j)},$$

$$p_{T_j-1} = \frac{A_j^{-1}(T_j - 1, T_j - 1)}{A_j^{-1}(T_j - 1, T_j - 1) + A_j^{-1}(T_j - 1, 2, T_j - 1)}.$$

Once we know the values of p_i for $i = 0, 1, \ldots$, we can easily solve for the steady-state probability π_i that i customers are in the queue by solving the following set of equations:

$$\pi_i = \begin{cases} \pi_0 p_k \rho_k^{i-T_k+1} \prod_{j=1}^{k} p_j \rho_j^{T_{j+1}-T_j} T_k \leq i < T_k & \\ & 1 \leq k \leq N - 1, \\ \pi_0 p_N \rho_N^{i-T_N+1} \prod_{j=1}^{N-1} p_j \rho_j^{T_{j+1}-T_j} T_N \leq i, \end{cases} \tag{5}$$

where π_0 can be determined from the normalization condition. The value of $\bar{L}_4$ can be determined from (5).

2.2. Case 2: Some Thresholds Might Be the Same

In this section we show how the approximation outlined in the previous section generalizes for the case in which the thresholds are not necessarily different. The approximation for this case again uses Equation 1, but parameters p_i and f_i are calculated in a more general manner. (Note that, if processors p_{j+m}, $m = 0, 1, \ldots, s$ have the same threshold, i.e., $T_j = T_{j+1} = \ldots = T_{j+s}$, then p_{T_j-1} is the probability that *all* processors P_{j+m} for $m = 0, 1, \ldots, s$ are busy when the queue length is $T_i - 1$.) The state space for this case grows exponentially with the number of identical threshold values, since the procedure requires the enumeration

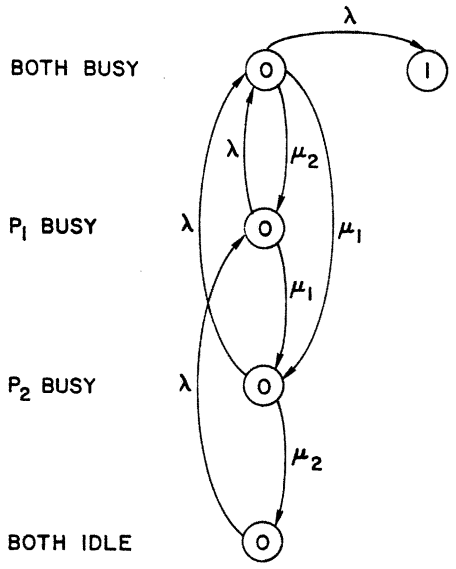

Figure 5. State diagram for $T_2 = T_1 = 1$.

of all possible configurations of busy processors. Our method would thus be useful only if the number of processors with identical thresholds is small. To avoid a cumbersome exposition we will exhibit two specific cases for which the thresholds are identical, and show how to calculate these parameters. The extension to the general case should be obvious.

Suppose, as shown in Figure 5, that $T_2 = T_1 = 1$. The Markov process used to solve for p_0 and f_i for $i = 1, 2$, as shown in Figure 5, consists of 4 states corresponding to all the possibilities for processors P_1 and P_2 being busy or idle. If we order the states as (0, Both busy), (0, P_1 busy), (0, P_2 busy), (0, Both idle), followed by (1, Both busy), then the generator matrix is given by

$$G = \begin{bmatrix} -(\lambda+\mu_1+\mu_2) & \mu_2 & \mu_1 & 0 & \lambda \\ \lambda & -(\lambda+\mu_1) & \mu_1 & 0 & 0 \\ \lambda & 0 & -(\lambda+\mu_2) & \mu_2 & 0 \\ 0 & \lambda & 0 & -\lambda & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

We define A to be

$$A = \begin{bmatrix} -(\lambda+\mu_1+\mu_2) & \mu_2 & \mu_1 & 0 \\ \lambda & -(\lambda+\mu_1) & \mu_1 & 0 \\ \lambda & 0 & -(\lambda+\mu_2) & \mu_2 \\ 0 & \lambda & 0 & -\lambda \end{bmatrix},$$

and thus we can write

$$f_i = \frac{A^{-1}(0, 0) + A^{-1}(0, i)}{\sum_{j=0}^{3} A^{-1}(0, j)}, \quad i = 1, 2$$

and

$$p_0 = \frac{A^{-1}(0, 0)}{\sum_{j=0}^{3} A^{-1}(0, j)}.$$

Figure 6 shows the case in which $T_1 < T_2 = T_3 < T_4$. In a manner similar to that for the case of Figure 5, we can write the A matrix for this case as

$$A = \begin{bmatrix} M_2 & \mu_2 I_2 & \mu_3 I_2 & 0 \\ Z_2 & M_2 - \mu_2 I_2 & \mu_3 I_2 & 0 \\ 0 & Z_2 & M_2 - \mu_3 I_2 & \mu_2 I_2 \\ 0 & 0 & Z_2 & M_2 - (\mu_2 + \mu_3) I_2 \end{bmatrix},$$

where I_2 and Z_2 were defined previously, and M_2 is given by

$$M_2(i, m) = \begin{cases} -(\lambda p_1 + \mu_2 + \mu_3) & i = 0, \quad m = 0, \\ \lambda p_0 & i = 0, \quad m = 1, \\ \lambda & i = 1, 2, \ldots, T_2 - 2, \\ & \qquad m = i + 1, \\ -\left(\lambda + \sum_{j=1}^{3} \mu_i\right) & i = 1, 2, \ldots, T_2 - 1, \\ & \qquad m = i, \\ \mu_1 & i = 1, 2, \ldots, T_2 - 1, \\ & \qquad m = i - 1, \\ 0 & \text{otherwise.} \end{cases}$$

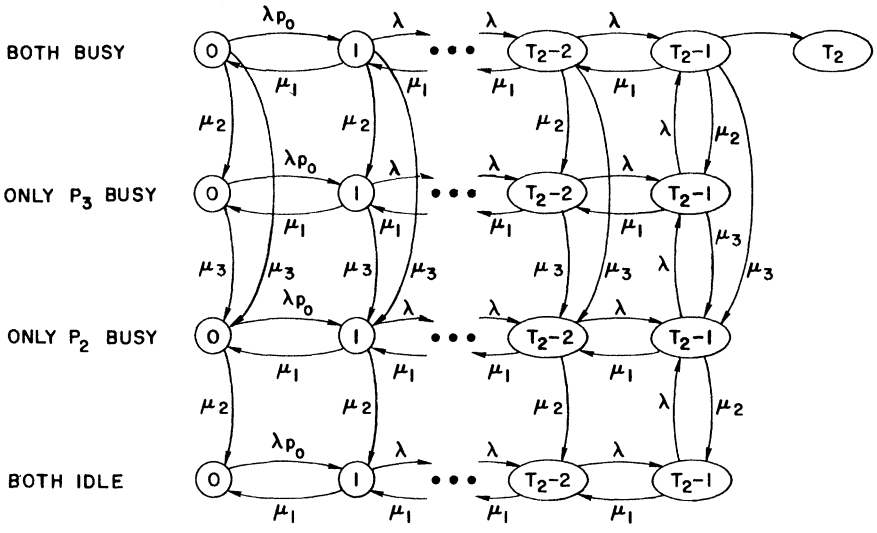

Figure 6. State diagram for $T_1 < T_2 = T_3 < T_4$, $\mu_1 \neq \mu_2$.

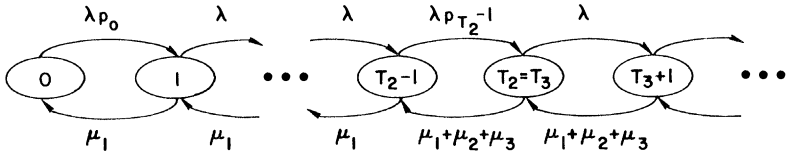

Figure 7. State diagram for birth-death approximation, non-unique thresholds.

For this case we can calculate

$$f_i = \frac{\sum_{j=1}^{T_2-1} A^{-1}(T_2-1,j) + A^{-1}(T_2-1,(i-1)T_2+j)}{\sum_{j=0}^{4T_2-1} A^{-1}(T_2-1,j)}$$

$$i = 2, 3$$

and

$$p_{T_2-1} = \frac{A^{-1}(T_2-1,\ T_2-1)}{\sum_{j=1}^{4} A^{-1}(T_2-1,\ jT_2-1)}$$

In solving for $\bar{L}_4$ of Equation 1 for this system, we would solve the birth-death system shown in Figure 7. The generalization to the more general threshold values is straightforward but notationally cumbersome.

The exponential state space growth with the number of identical thresholds is not a factor if all the processor speeds for these thresholds are identical. In Figure 8, we show the case similar to that of Figure 6, but with $\mu_2 = \mu_3$. The state space reduction that occurs arises from the fact that one need not preserve information about the identities of the busy processors, but only note their number. In cases like this one, the state space growth becomes linear.

3. Results

In this section we compare the mean time in the system calculated by the approximation to that obtained by exact simulation methods. Our simulation used the Research Queueing Package (RESQ), a simulation package developed by IBM (see Sauer, MacNair and Salza 1980), and was run on an IBM 3081. Confidence intervals were generated using the spectral method given in Heidelberger and Welch (1981). In Figure 9 we show the results for a three-server system in which the servicing rates differ geometrically by a factor of 1/10. The thresholds for the three servers are 0, 5 and 10. The approximation does very well for low or high values of the utilization, and overestimates the time spent in the system over moderate values. For high utilizations the approximation is close because most of the servers are busy, and thus the underlying Markov process is similar to the approximating birth-death process. A similar observation can be made for low utilizations, where most of the servers are idle, and thus the existence of the thresholds is not a major factor in the scheduling of jobs. The overestimation of the approximation for moderate utilization is also seen in Figure 10 where,

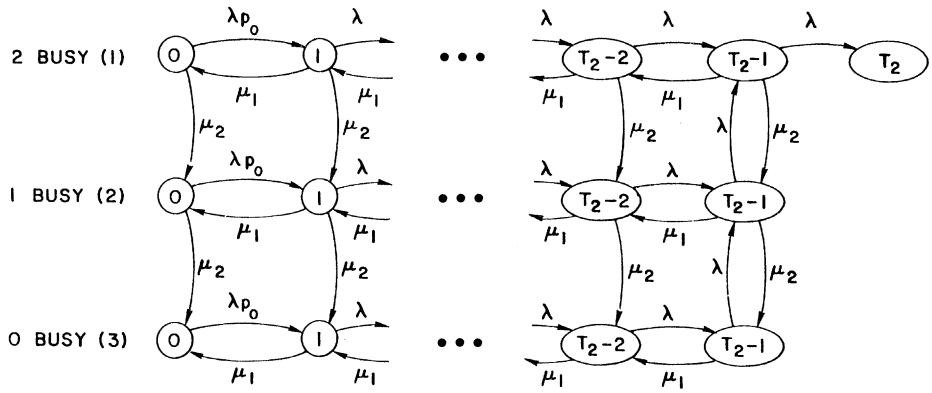

Figure 8. State diagram for $T_1 < T_2 = T_3 < T_4$, $\mu_1 = \mu_2$.

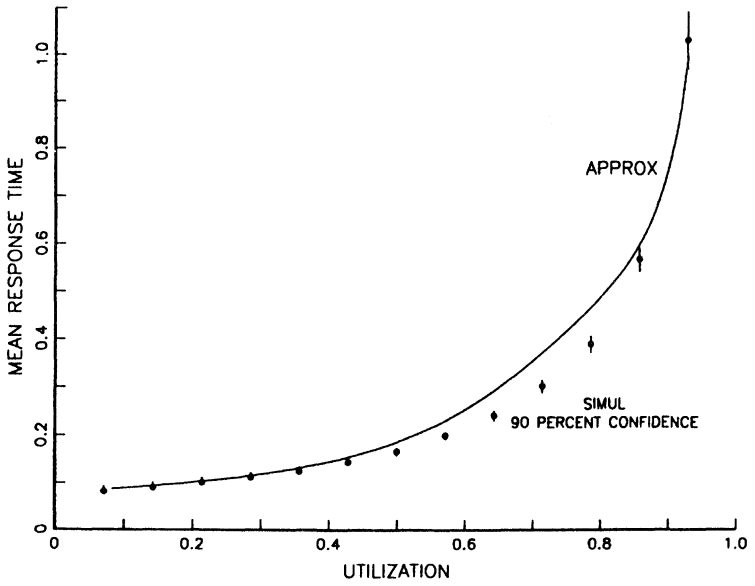

Figure 9. Comparison of simulation and approximation for three servers with rates 12.6, 1.26 and 0.126 for thresholds of 0, 5 and 10.

for the same set of servers, the thresholds depend on the utilization. Here the thresholds are chosen to maximize the throughput of the system between successive arrivals to the system. For a more thorough description of this performance metric and a proof that it is obtained by a threshold scheduling policy, see Nelson and Towsley.

The same overestimation for moderate utilizations is seen in Figure 11, for a five-server system. The particular thresholds chosen are those calculated by

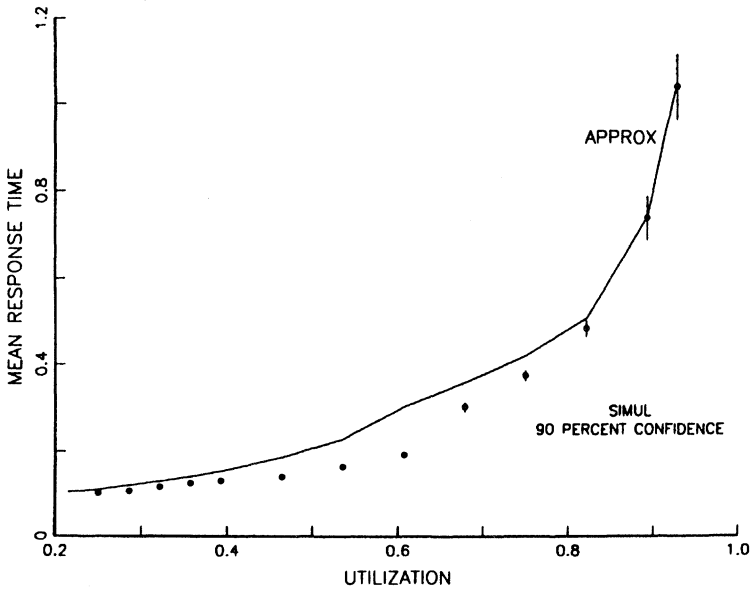

Figure 10. Comparison of approximation and simulation for different thresholds and rates 12.6, 1.26 and 0.126.

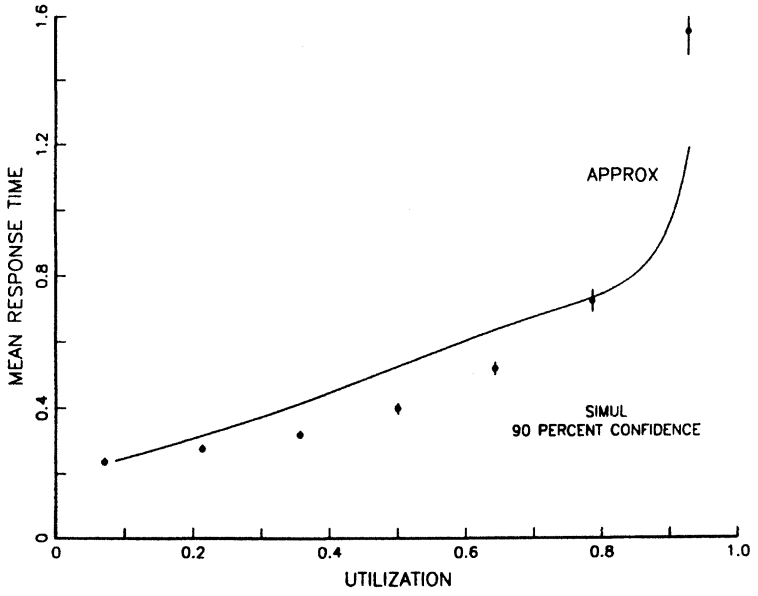

Figure 11. Comparison of simulation and approximation for five servers with rates 5.18, 4.14, 2.07, 1.55 and 1.03 for thresholds of 0, 1, 3, 5 and 9.

the threshold policy derived in Agrawala et al. The thresholds we chose minimize the completion of jobs in the system, assuming that there were no other arrivals to the system. In our last graph, Figure 12, we compare the approximation to an exact analysis for the two-server case. We assume that both thresholds are equal to zero. This assumption implies that we always schedule the fastest server if it is available. Ibe studied such a policy, and derived an approximation to the mean time in the system for the case in which the system had an arbitrary number of servers. The graph shows an interesting anomalous behavior in

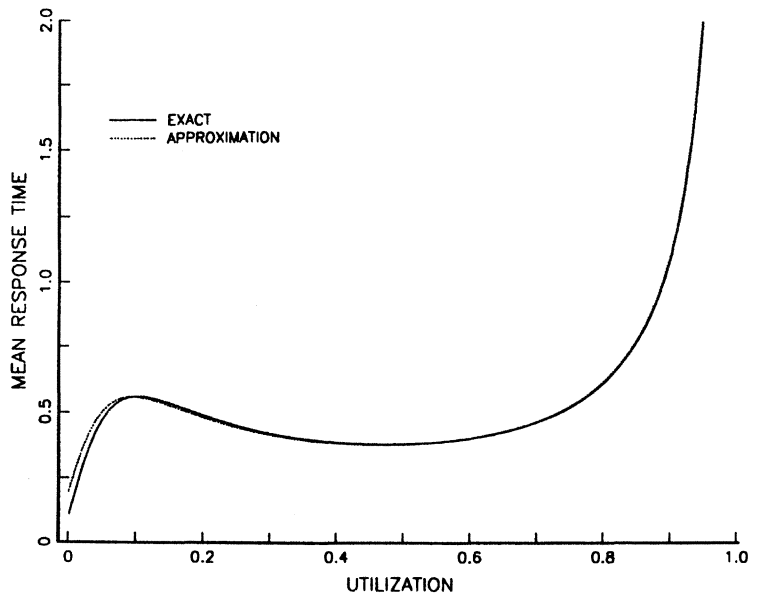

Figure 12. Comparison of approximation and exact analysis for service rates of 10 and 0.1 with both thresholds zero.

which the mean time in the system has a local maximum at low utilizations. This situation occurs because of the wide variation in service rates. Since both thresholds are zero, the slower server is scheduled even when it would be better to have customers wait until the faster server became available. The incremental delay due to scheduling the slower server is offset, for higher utilizations, by the increased number of customers processed by the faster server, and thus the mean time in system decreases, over a region, as the utilization increases. It is clear from the graph that the approximation mimics this anomalous behavior.

4. Conclusions

We have presented an approximation for the expected response time for a queueing system having multiple servers scheduled by a threshold discipline. The main method used in the approximation is to decompose the system into a set of related transient processes that are analyzed separately to obtain parameters used in a simplified birth-death process. We validated the approximation against simulation results and found it very accurate for low and high utilizations, and tended to overestimate the actual response time for moderate utilizations.

The complexity of calculating the approximation increases exponentially with the number of thresholds that are the same. The growth is linear only in the case in which equal thresholds occur for servers whose rates are also the same. Nevertheless, the approximation is probably not practical for systems having many sets of equal thresholds. Determining a method to reduce the complexity for cases of this type is an interesting research problem.

Acknowledgment

The authors would like to thank the anonymous reviewers for their detailed comments and careful reading of the manuscript.

This work was performed in part while Don Towsley was a visiting scientist at the IBM Thomas J. Watson Research Center. The work was also supported in part by the National Science Foundation under grant ECS-8310771.

References

AGRAWALA, A., E. COFFMAN, M. GAREY, AND S. TRIPATHI. 1984. A Stochastic Optimization Algorithm Minimizing Expected Flow Times on Uniform Processors. *IEEE Trans. Comput.* **C-33**, 351–356.

HEIDELBERGER, P., AND P. WELCH. 1981. A Spectral Method for Confidence Interval Generation and Run Length Control in Simulations. *Comm. Assoc. Comput. Mach.* **24**, 233–245.

IBE, O. 1982. An Approximate Analysis of a Multi-Server Queueing System with a Fixed Order of Access. *IBM Res. Rep. RC 9346.*

LIN, W., AND P. KUMAR. 1984. Optimal Control of a Queueing System with Two Heterogeneous Servers. *IEEE Trans. Automat. Control* **AC-29**, 696–703.

NELSON, R., AND D. TOWSLEY. 1985. On Maximizing the Number of Departures before a Deadline on Multiple Processors. *IBM Res. Rep. RC 11255.*

NEUTS, M., AND K. MEIER. 1981. On the Use of Phase Type Distributions in Reliability Modelling of Systems with a Small Number of Components. *OR Spektrum.* **2**, 227–234.

SAUER, C. H., E. A. MACNAIR AND S. SALZA. 1980. A Language for Extended Queueing Network Models. *IBM. J. Res. Develop.* **24**, 747–755.