AT&T Technical Journal
Vol. 64, No. 2, February 1985
Printed in U.S.A.

# Analysis of Clocked Schedules—High-Priority Tasks

By B. T. DOSHI*

(Manuscript received December 8, 1983)

Clocked schedules are used in a variety of real-time systems to perform tasks in accordance with their delay requirements. A comprehensive overview of various clocked schedules can be found in the accompanying paper by Fredericks et al. For one model of a clocked schedule we can see approximate and exact analysis of the relevant performance measures in the papers by Fredericks et al. and by Ackroyd in this issue of the *Journal*. These results can also be used to evaluate the long-term delays in the other models. For extremely time-critical (high-priority) tasks, the probability of tasks not getting served in the scheduled interval and the short-term delay distribution are important performance measures that are sensitive to the detail structure of the clocked schedule. In this paper, we show that these performance measures can be obtained in terms of steady-state distribution of an embedded Markov chain. This steady-state distribution is calculated, exactly or approximately, for a number of models, and the results are used to compare, numerically, various scheduling mechanisms.

## I. INTRODUCTION

Clocked schedules are used in a variety of real-time systems to perform tasks in accordance with their timing requirements. These real-time systems (switching, monitoring, etc.) are characterized by having to perform some tasks with extremely stringent timing requirements. We call these high-priority tasks. Besides these, real-time systems also perform tasks that can tolerate somewhat longer delays

* AT&T Bell Laboratories.

without affecting performance. These will be called low-priority tasks. Finally, these systems perform a variety of background tasks (audits, maintenance, etc.), which have very liberal timing requirements but which require a specified minimum fraction of the processor time over a reasonably long period of time. These tasks are usually performed when no high- or low-priority work is present in the system. We call these the fill tasks. We emphasize that the boundaries among these categories of tasks are not clear-cut and that within each category the tasks may have significantly differing time scales. There are various mechanisms for allocating the processor time to meet the varying requirements of the individual tasks. These mechanisms and their relative merits are discussed in Ref. 1. Clocked schedules provide a very effective and reliable mechanism to achieve the desired timing objectives. Of course, to use them effectively, it is necessary to understand their performance as it applies to the various categories of tasks. Considerable progress has been made in this direction (see Refs. 1, 2, and 5 through 8). For low-priority and fill tasks approximate analysis of queueing models is described in Ref. 1. Exact numerical procedures for these models are developed in Ref. 2. For many systems the analyses in Refs. 1 and 2 can also be used to study high-priority tasks. However, for other systems the modeling assumptions may not adequately capture the working of the schedule to accurately analyze the short-term delay of high-priority tasks. For such systems it is necessary to incorporate additional features into the clocked schedule model and understand the implications of these features. Some of these features are the gating mechanism, the strategies used under overrun, and the dependence introduced by autonomous work generation. In this paper we consider increasingly complex models of clocked schedule by introducing these features one at a time, and we develop methodology to study the performance measures for the high-priority tasks in these situations. To analyze these models in a relatively simple way, we make assumptions that restrict our analysis only to high-priority tasks.

This paper is organized as follows: In Section II we briefly define the clocked schedule and the performance measures that we will use. We also introduce various gating mechanisms and overrun strategies in that section. In Section III we analyze a simple clocked schedule with respect to the performance measures for the high-priority tasks. We also use this analysis to numerically compare various scheduling and priority mechanisms. Section IV deals with various generalizations and complex clocked schedules.

## II. MODEL OF A CLOCKED SCHEDULE

The mechanism underlying a clocked schedule has been discussed in detail in Ref. 1, so we will be very brief here.

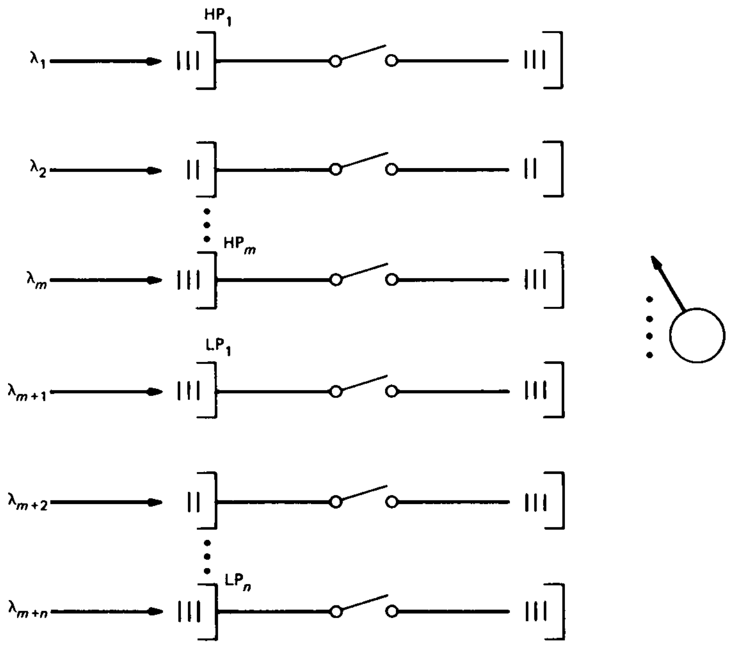

Fig. 1—Multiclass queueing structure in clocked schedules.

Assume that there are $m$ high-priority tasks and $n$ low-priority tasks numbered $\{1, 2, \ldots, m, m+1, \ldots, m+n\}$. Each task has two queues, the external queue and the internal queue (see Fig. 1). Jobs arrive at the external queues and, at specified times, are transferred to the corresponding internal queues. The internal queues are served according to a priority scheme, with lower-numbered queues having higher priority. The priority of the high-priority queue over the low-priority queue and the fill work is preemptive. The priority between two high-priority queues is determined by the overrun strategy discussed below.

The time is divided into intervals of length $T$ ms. During each of these time slots some high-priority tasks and some low-priority tasks are scheduled (for an example see Fig. 2). The jobs in the queues for the tasks scheduled in a given slot are typically moved from the external queues to the internal queues in that time interval. The gating mechanism determines when this transfer happens. Three gating mechanisms are used in practice, the actual choice depending on the application and hardware limitations. The first gating mechanism (G1) transfers the jobs in the external queues to the internal queues for all tasks scheduled in a given interval at the beginning of that interval. No more transfer takes place during that interval. In the second gating mechanism (G2) the processor starts with task 1 at the

| TASK NUMBER | 0 | $T$ | $2T$ | $3T$ |
|---|---|---|---|---|
| 1 | 1 | 1 | 1 | 1 |
| 2 | 0 | 1 | 0 | 1 |
| 3 | 1 | 0 | 0 | 0 |
| 4 | 0 | 0 | 0 | 1 |
|  | $T$ | $2T$ | $3T$ | $4T$ |

1 = TASK SCHEDULED

0 = TASK NOT SCHEDULED

Fig. 2—An example of a clocked schedule for $m = 2$, $n = 2$.

beginning of each interval. If it is scheduled for that interval the transfer takes place and the processor serves the internal queue for task 1 to completion. During this period no additional transfer takes place for queue 1. After exhausting the first queue the processor moves to queue 2. If the corresponding task is scheduled, the transfer takes place at this time, and so on. Note that the order of service for the internal queues is always $(1, 2, \ldots, m + n, \text{fill})$. The schedule table only indicates when the transfer takes place for each task. The third gating mechanism (G3) is essentially similar to G2, the difference being that when the processor is serving an internal queue, the new arrivals to the corresponding external queue move immediately to the internal queue.

If all the internal queues become empty before any interval ends, then the processor does fill work until the end of that interval. If the interval ends while the processor is working on a low-priority job, then that work is preempted and the processor moves to queue 1. If the processor is working on a high-priority queue when the interval ends, then we say an overrun has occurred. We consider three different overrun strategies, S1, S2, and S3. In strategy S1 (Fig. 3) the remaining high-priority work in the internal queues is expelled. This is done in some systems either because the necessary bookkeeping ability is not available or because serving a high-priority task after a reasonably long delay is most likely to result in unsuccessful service and waste of the processor time. More importantly, as we will see in Section IV, this strategy is the starting point of our analysis for more common strategies S2 and S3, which are discussed below. The strategy S2 (Fig.

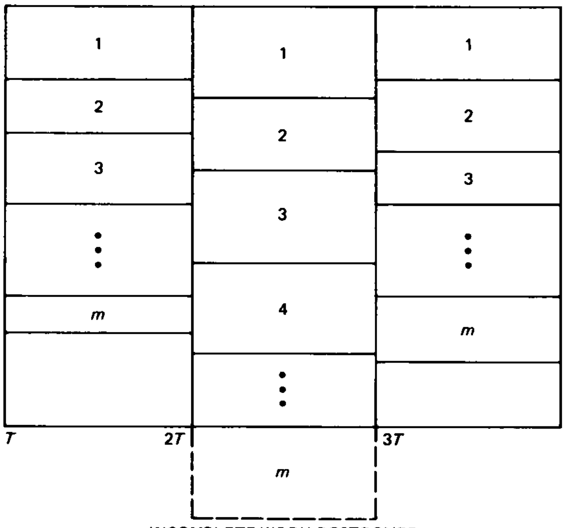

Fig. 3—Strategy S1, in which incomplete work is destroyed.

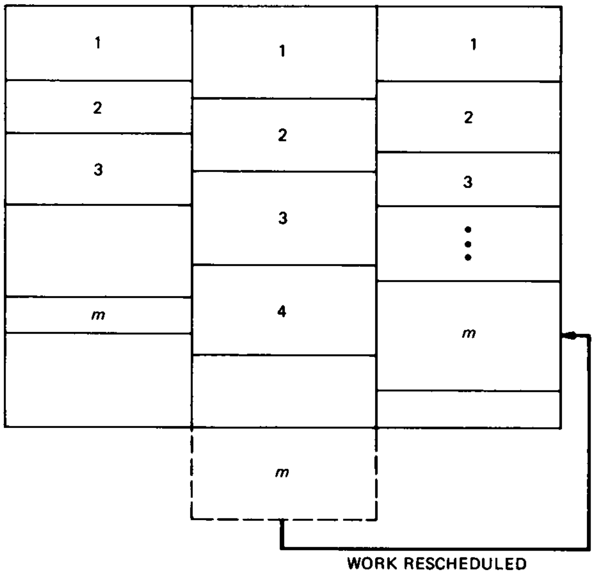

Fig. 4—Strategy S2, in which incomplete work is rescheduled.

4) under overrun is the same as that used for interrupting a low-priority task, namely, the incomplete work is rescheduled and the processor moves back to queue 1. In strategy S3 (Fig. 5) the end of interval is ignored until all the high-priority internal queues are empty.

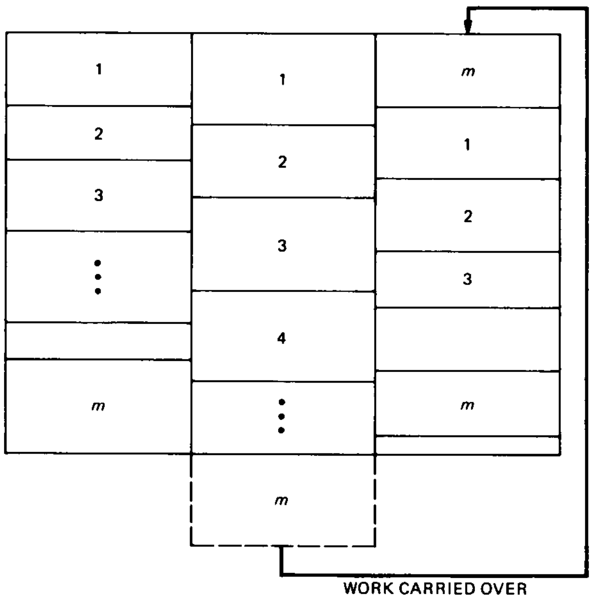

Fig. 5—Strategy S3, in which incomplete work is carried over.

Only then does the processor move to queue 1. Note, however, that the end of interval is usually marked by an interrupt from an asynchronous clock. Thus in S3 an overrun reduces the time available in the next interval.

As mentioned earlier, the high-priority tasks are typically very time critical. In this paper we consider high-priority tasks as those which we require to be completed in the scheduled interval with very high probability. This dictates which performance measures capture our interest.

We are interested in the probability of an overrun. Assuming that a high-priority task is scheduled once every $p$ time slots, we are interested in the delay distribution in the range $[pT, (p + 1)T]$. Once we address these two questions successfully, we can answer the other important questions: What is the largest $T$ for which the delay criteria for all high-priority tasks are satisfied? This frequently determines the best value of $T$. Which is the best priority order for the high-priority tasks?

Since we are interested in the delay distribution over a short time range, the effects of gating mechanism and overrun may be quite significant here. In this paper we develop a methodology to address the issues discussed above in a manner suitable for relatively easy computations. We begin with a rather simple special case of the general model and then introduce complexity in the schedule gradually.

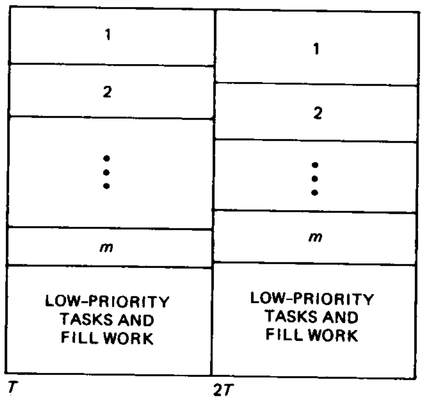
1
2
m
LOW-PRIORITY TASKS AND FILL WORK
1
2
m
LOW-PRIORITY TASKS AND FILL WORK
T
2T

Fig. 6—A simple clocked schedule.

## III. ANALYSIS OF A SIMPLE CLOCKED SCHEDULE

Recall that high-priority tasks have preemptive priority over low-priority and fill tasks and so are not affected by them. Thus we can simplify the description of the schedule by replacing all the low-priority and the fill tasks with a single task, $m + 1$.

The special case we study here (see Fig. 6) is a clocked schedule in which each of the $m$ high-priority tasks is scheduled in each interval. Thus the entire schedule has a period of one time interval ($T$). In addition we assume that the jobs in the external queue $i$ arrive according to a Poisson process at rate $\lambda_i$, and that the arrival processes in different queues are independent. When the processor starts serving queue $i$, it incurs an overhead $OH_i$ with distribution function $H_i$. Each job in queue $i$ has service time $X_i$ with distribution function $F_i$, $i = 1, \ldots, m$. For $i = 1, 2, \ldots, m$, let

$$a_i = \int_0^\infty x dH_i(x),$$

$$\alpha_i = \int_0^\infty (x - a_i)^2 dH_i(x),$$

$$b_i = \int_0^\infty x dF_i(x),$$

$$\beta_i = \int_0^\infty x^2 dF_i(x),$$

$$\rho_i = \lambda_i b_i,$$

and

$$\gamma_i = \lambda_i \beta_i.$$

We analyze this clocked schedule with overrun strategy S1 and gating mechanisms G1 and G2. Note that we require the high-priority queues to exhaust in every interval with high probability, so the difference between overrun strategies S1, S2, and S3 should be small.

## *3.1 Analysis for gating mechanism G1*

The analysis for the gating mechanism G1 is simple. First consider an arbitrary clock interval of length $T$. Suppose we let this interval continue until all the internal high-priority queues are empty. Let $t_i$ denote the time from the beginning of this interval to the moment the processor exhausts internal queue $i$. Then

$$t_i = \sum_{j=1}^{i} OH_j + \sum_{j=1}^{i} \sum_{k=1}^{N_j} X_{jk}, \tag{1}$$

where $X_{jk}$ is the service time of the $k$th job in the $j$th queue and $N_j$ is the number of jobs served by the processor from the $j$th queue. $N_j$ has Poisson distribution with mean $\lambda_j T$. Thus, if $\tilde{G}_i$, $\tilde{H}_i$ and $\tilde{F}_i$ denote the Laplace transforms of $t_i$, $OH_i$ and $X_i$, respectively, then

$$\tilde{G}_i(s) = \prod_{j=1}^{i} [\tilde{H}_j(s) e^{-\lambda_j T(1-\tilde{F}_j(s))}]. \tag{2}$$

The above equation can be inverted to obtain the distribution of $t_i$. This can be done, for example, by using the transform inversion technique discussed in Ref. 3. Alternatively, the convolutions involved in eq. (1) can be carried out directly using fast Fourier transforms after suitably discretizing the random variables.[2] Obviously, an overrun occurs when $t_m > T$. Thus,

$$P\{\text{overrun}\} = P\{t_m > T\} = 1 - G_m(T).$$

Also,

$$\int_{t=0}^{\infty} e^{-st}(1 - G_m(t))dt = \frac{1 - \tilde{G}_m(s)}{s}. \tag{3}$$

Thus, we can get $P\{\text{overrun}\}$ by inverting the right-hand side of eq. (3) at $T$.

Next, we obtain the sojourn time distribution for a job arriving at queue $i$ assuming that it does get served in the following time slot. Let $S_i$ denote the sojourn time for an arbitrary job arriving at queue $i$, that is, $S_i$ is the time between the arrival of a job at queue $i$ and the completion of its service. Suppose that a job arrives at queue $i$ when $R$ time units are left to the end of a time slot and that there are $N_i$ jobs waiting in that external queue at that time. Then $R$ is uniformly distributed on $(0, T)$ and, conditioned on $R$, $N_i$ has Poisson distribution with mean $\lambda_i(T - R)$. The sojourn time, $S_i$, is then given by

$$S_i = R + t_{i-1} + \sum_{j=1}^{N_i+1} X_{ij} + OH_i.$$

Thus, if $\tilde{S}_i$ denotes the Laplace transform of $S_i$, then

$$\tilde{S}_i(s) = \frac{\tilde{H}_i(s)\tilde{G}_{i-1}(s)\tilde{F}_i(s)[e^{-\lambda_i T(1-\tilde{F}_i(s))} - e^{-sT}]}{T(s - \lambda_i(1 - \tilde{F}_i(s))}. \tag{4}$$

Once again, eq. (4) can be inverted to obtain the distribution of $S_i$.

We next obtain the mean and variance of $t_i$ and the mean of $S_i$. From eq. (2) we easily get

$$E[t_i] = \sum_{j=1}^{i} (a_j + \rho_j T), \tag{5}$$

and

$$\mathrm{Var}[t_i] = \sum_{j=1}^{i} (\alpha_j + \gamma_j T). \tag{6}$$

Also, by differentiating eq. (4) and setting $s = 0$, we get

$$\bar{S}_i = E[S_i] = \sum_{j=1}^{i-1} (a_j + \rho_j T) + b_i + \frac{T}{2}(1 + \rho_i) + a_i. \tag{7}$$

### *3.2 Analysis for gating mechanism G2*

We next analyze the simple clocked schedule under gating mechanism G2. The overrun strategy is still assumed to be S1. Our general approach is to define an embedded Markov chain and show that the relevant performance measures can be obtained easily in terms of the steady-state distribution of this Markov chain. We then find expression for this steady-state distribution.

Let $N_{n,i}$ denote the number of jobs in the external queue $i$ when the $n$th time slot of length $T$ begins. Then $\{(N_{n1}, N_{n2}, \cdots N_{nm}): n \geq 1\}$ is an ergodic Markov chain. Let $(N_1, N_2, \cdots N_m)$ denote the steady-state random vector corresponding to this Markov chain and let $\tilde{P}(z_1, z_2, \cdots, z_m)$ denote the generating function of the random vector $(N_1 \cdots N_m)$, that is,

$$\tilde{P}(z_1, z_2, \cdots, z_m) = E[z_1^{N_1} z_2^{N_2} \cdots z_m^{N_m}].$$

We first show that the probability of overrun and the sojourn time distribution for any queue can be obtained easily in terms of $\tilde{P}$. Consider an arbitrary time slot and, as before, allow all high-priority internal queues to be completely served in this slot by letting the available interval overrun, if necessary. Let $t_i$ denote the time from the beginning of this time slot until the completion of service to the internal queue $i$. Let $(N_1, N_2, \cdots N_m)$ denote the number in the external queues at the beginning of this slot, and for $i = 1, 2, \cdots m$, let $N_i'$ denote the number transferred from the $i$th external queue to the internal queue when the service to that queue begins. Let $t_o = 0$. Then for $i = 1, 2, \cdots, m$,

$$t_i = t_{i-1} + OH_i + \sum_{j=1}^{N_i'} X_{ij}, \tag{8}$$

$$N_j' = N_i + M_i(t_{i-1}), \tag{9}$$

where

$$M_i(x) \sim \text{Poisson } (\lambda_i x). \tag{10}$$

Equations (8) through (10) allow us to calculate the distributions of $t_i$ and $N_i'$, $i = 1, 2, \cdots m$, recursively. In terms of transforms, let

$$\tilde{G}(s_1, s_2, \cdots s_m) = E[e^{-(s_1t_1+s_2t_2+\cdots+s_mt_m)}].$$

Then, $\tilde{G}$ can be obtained recursively from eqs. (8) through (10) as follows: Let $\underline{s} = (s_1, s_2, \cdots, s_m)$. Define $\sigma_i(\underline{s})$ and $\omega_i(\underline{s})$, $i = 1, \cdots, m$, recursively by

$$\sigma_m(\underline{s}) = s_m,$$

$$\omega_m(\underline{s}) = \tilde{F}_m(\sigma_m(\underline{s})),$$

$$\sigma_i(\underline{s}) = \lambda_{i+1}(1 - \omega_{i+1}(\underline{s})) + \sigma_{i+1}(\underline{s}) + s_i, \qquad 1 \leqslant i \leqslant m - 1, \tag{11}$$

and

$$\omega_i(\underline{s}) = \tilde{F}_i(\sigma_i(\underline{s})), \qquad 1 \leqslant i \leqslant m - 1.$$

Then

$$\tilde{G}(\underline{s}) = \tilde{P}[\omega_1(\underline{s}), \omega_2(\underline{s}), \omega_3(\underline{s}), \cdots \omega_m(\underline{s})] \prod_{i=1}^{m} \tilde{H}_i[\sigma_i(\underline{s})]. \tag{12}$$

The marginal Laplace transform of $t_m$ now follows by putting $\underline{s} = \underline{s}^{(m)} = (0, 0, \cdots, \underline{s}_m)$. Once again, this can be inverted to get the distribution function of $t_m$. In particular,

$$P\{\text{overrun}\} = 1 - P\{t_m < T\},$$

and

$$\int_0^\infty e^{-st} P\{t_m > t\} dt = \frac{1 - \tilde{G}(\underline{s}^{(m)})}{s}. \tag{13}$$

Thus $P\{\text{overrun}\}$ can be obtained by inverting the right-hand side in eq. (13) at $T$.

We next derive the distribution function of the sojourn time in a given queue in terms of $\tilde{P}$. Suppose we are considering queue $i$. We assume that the probability that the task $i$ overruns the interval is essentially zero. Let $S_i$ denote sojourn time of an arbitrary job in queue $i$ and $\tilde{S}_i$ its Laplace transform. Let $N_{i1}''$ and $N_{i2}''$ denote, respectively, the numbers in the internal queue $i$ and the external queue $i$ when an arbitrary job in queue $i$ completes service. Let $N_i'' = N_{i1}'' + N_{i2}''$. Finally, let $N_i'$ be the number of jobs in the external queue $i$ just before they are transferred to the internal queue and the processor starts working on them. Let $\tilde{r}_{i1}$, $\tilde{r}_{i2}$, $\tilde{r}_i$, and $\tilde{q}_i$ denote the generating functions of $N_{i1}''$, $N_{i2}''$, $N_i''$, and $N_i'$, respectively. We express $\tilde{S}_i$ in terms of $\tilde{r}_i$, $\tilde{r}_i$ in terms of $\tilde{q}_i$, and $\tilde{q}_i$ in terms of $\tilde{P}$. Combining these relations we will get $\tilde{S}_i$ in terms of $\tilde{P}$.

First note that, since the service within a given queue is First-In First-Out (FIFO), $N_i''$denotes the number of arrivals to queue $i$ during a time interval of length $S_i$. Thus, from Ref. 4 we get

$$\tilde{S}_i(s) = \tilde{r}_i(1 - s/\lambda_i). \tag{14}$$

Next,

$$\begin{aligned} \tilde{r}_i(z) &= E[z^{N_i''}] \\ &= E[E[z^{N_i''} | N_i']] \\ &= E[E[E[z^{N_{i1}''+N_{i2}''} | K_i] | N_i']], \end{aligned} \tag{15}$$

where $K_i$ is the position of the job that completed service in the batch of size $N_i'$ transferred to the $i$th internal queue when the processor last arrived there. From eq. (15) we get

$$\begin{aligned} \tilde{r}_i(z) &= E[E[z^{N_i'-K_i}\tilde{F}_i(\lambda_i(1-z))^{K_i}\tilde{H}_i(\lambda_i(1-z)) | N_i']] \\ &= \frac{\tilde{H}_i(\lambda_i(1-z))(\tilde{F}_i(1-z))[\tilde{q}_i(z) - \tilde{q}_i(\tilde{F}_i(\lambda_i(1-z)))]}{\bar{n}_i(z - \tilde{F}_i(\lambda_i(1-z)))}, \end{aligned} \tag{16}$$

where $\bar{n}_i = E[N_i']$. Finally, by the arguments used to derive eq. (12) we obtain the following relation between $\tilde{q}_i$ and $\tilde{P}$: Let

$$\sigma_{i-1}'(z) = \lambda_i(1 - z),$$

$$\omega_{i-1}'(z) = \tilde{F}_{i-1}[\sigma_{i-1}(z)],$$

$$\sigma_j'(z) = \lambda_j(1 - \omega_{j+1}'(z)) + \sigma_{j+1}'(x), \qquad 1 \leqslant j \leqslant i - 1,$$

and

$$\omega_j'(z) = \tilde{F}_j(\sigma_j'(z)) \quad 1 \leqslant j \leqslant i - 2.$$

Then

$$\begin{aligned} \tilde{q}_i(z) &= E[z^{N_i'}] \\ &= \prod_{j=1}^{i-1} \tilde{H}_j(\sigma_j'(z))\tilde{P}(\omega_1'(z), \omega_2'(z), \cdots, \omega_{i-1}'(z), z). \end{aligned} \tag{17}$$

Combining eqs. (14), (16), and (17) we obtain $\tilde{S}_i$ in terms of $\tilde{P}$.

Thus, all performance measures of interest can be expressed in terms of the steady-state distribution of the vector $(N_1, N_2, \cdots, N_m)$. We next evaluate this distribution $P(l_1, l_2, \cdots, l_m)$ or, equivalently, its generating function $\tilde{P}(z_1, z_2, \cdots, z_m)$. Let

$$\begin{aligned} &Q(K_1, K_2, \cdots, K_m \mid l_1, l_2, \cdots, l_m) \\ &\quad = P\{N_{n+1,1} = k_1, N_{n+1,2} = k_2, \cdots, N_{n+1,m} \\ &\quad = k_m \mid N_{n,1} = l_1, \cdots, N_{n,m} = 1_m\} \end{aligned}$$

be the transition probability function for the Markov chain $\{(N_{n,1}, N_{n,2}, \cdots, N_{n,m}): n \geqslant 1\}$. Then $P$ is the unique nonnegative solution of

$$\begin{aligned} &P(k_1, k_2, \cdots, k_m) \\ &\quad = \sum_{l_i=0}^{\infty} \sum_{l_2=0}^{\infty} \sum_{l_m=0}^{\infty} P(l_1, l_2, \cdots, l_m) Q(k_1, k_2, \cdots, k_m \mid l_1, \cdots, l_m), \end{aligned} \tag{18}$$

for $k_i = 0, 1 \cdots, i = 1, 2 \cdots m$, and

$$\sum_{k_1=0}^{\infty} \sum_{k_2=0}^{\infty} \cdots \sum_{k_m=0}^{\infty} P(k_1 \cdots k_m) = 1. \tag{19}$$

$Q$ can be readily expressed in terms of $l_1, l_2, \cdots, l_m, k_1, k_2, \cdots, k_m$ and the arrival and service time parameters. In particular, it is clear that $Q(k_1, \cdots k_m \mid l_1, l_2, \cdots l_m)$ does not depend on $l_m$. Thus, if we define

$$P_i(k_1, k_2, \cdots, k_i) = P\{N_1 = k_1, \cdots, N_i = k_i\} \tag{20}$$

and

$$\begin{aligned} &Q_i(k_1 \cdots k_i \mid l_1, l_2, \cdots, l_{i-1}) \\ &\quad = P\{N_{n+1,1} = k_1, \cdots, N_{n+1,i} = k_i \mid N_{n,1} \\ &\quad = l_1, \cdots, N_{n,i-1} = l_{i-1}\}, \end{aligned} \tag{21}$$

for $1 \leqslant i \leqslant m$, then

$$\begin{aligned} P_m(k_1, \cdots, k_m) &= \sum_{l_1=0}^{\infty} \sum_{l_2=0}^{\infty} \cdots \sum_{l_m=0}^{\infty} P_m(l_1 \cdots l_m) \\ &\qquad \cdot Q_m(k_1, \cdots k_m \mid l_1 \cdots l_{m-1}) \\ &= \sum_{l_1=0}^{\infty} \cdots \sum_{l_{m-1}=0}^{\infty} P_{m-1}(l_1 \cdots l_{m-1}) \\ &\qquad \cdot Q_m(k_1 \cdots k_m \mid l_1 \cdots l_{m-1}), \end{aligned} \tag{22}$$

and, in general, for $i = 2, \cdots m$,

$$\begin{aligned} P_i(k_1, \cdots k_i) = \sum_{l_1=0}^{\infty} \cdots \sum_{l_{i-1}=0}^{\infty} P_{i-1}(l_1, \cdots, l_{i-1}) \\ \cdot Q_i(k_1 \cdots, k_i \mid l_1 \cdots, l_{i-1}). \end{aligned} \tag{23}$$

Thus $P_1, P_2, \cdots, P_m$ can be calculated recursively using eq. (23). We start these recursions with

$$P_1(k_1) = \frac{e^{-\lambda_1 T}(\lambda_1 T)^{K_1}}{k_1!} k_1 = 0, \cdots .$$

These recursions use the functions $Q_i$ for $i = 1, 2, \cdots m$. Of course, some form of discretization is necessary to carry out these recursions. For large $m$ it is convenient to have a closed form approximation instead. Such an approximation is possible if we assume that the probability of an overrun occurring by queue $i$, $i = 1, \cdots m - 1$ is negligible. We illustrate this approximation below for the case $m = 2$ and then write down the approximation for general $m$. For $m = 2$, the exact generating function is given by

$$\begin{aligned}\tilde{P}(z_1, z_2) &= E[z_1^{N_1} z_2^{N_2}] \\ &= E[E[z_1^{N_1} z_2^{N_2} \mid t_1']] \\ &= E[e^{-\lambda_1 T(1-z_1)-\lambda_2 T(1-z_2)+\lambda_2 t_1'(1-z_2)}],\end{aligned}$$

where $t_1'$ represents the time from the beginning of the previous s ot until the completion of all work in the internal queue 1 in that sl t. Thus,

$$\tilde{P}(z_1, z_2) = e^{-\lambda_1 T(1-z_1)-\lambda_2 T(1-z_2)} \cdot \sum_{l_1=0}^{\infty} \frac{e^{-\lambda_1 T}(\lambda_1 T)^{l_1}}{l_1!} \cdot \left[\begin{array}{l} e^{\lambda_2 T(1-z_2)} \displaystyle\int_{s_1=T}^{\infty} \int_{x_1=0^-}^{s_1} dH_1(x_1) dF_1^{(l_1)}(s_1 - z_1) \\ + \displaystyle\int_{s_1=0^-}^{T} \int_{x_1=0}^{s_1} dH_1(x_1) dF_1^{(l_1)}(s_1 - x_1) e^{\lambda_2 s_1(1-z_2)} \end{array}\right]. \tag{24}$$

Our approximation is equivalent to replacing the first term in the squared bracket in eq. (24) by

$$\int_{s_1=T}^{\infty} \int_{x_1=0^-}^{s_1^\dagger} dH_1(x_1) dF_1^{(l_1)}(s_1 - z_1) e^{\lambda_2 s_1(1-z_2)}.$$

If we denote the resulting approximation for $\tilde{P}$ by $\bar{P}$, then

$$\bar{P}(z_1, z_2) = \tilde{H}_i(-\lambda_2(1 - z_2)) e^{-\lambda_1 T(1-z_1)-\lambda_2 T(1-z_2)-\lambda_1 T(1-\tilde{F}_1(-\lambda_2(1-z_2)))}. \tag{25}$$

How good is $\bar{P}$ an approximation for $\tilde{P}$? If we put $z_1 = 1$ in eqs. (24) and (25) and then invert the resulting generating functions for $N_2$, we can compare the exact and approximate distributions for $N_2$. Tables I through III give such distributions for three sets of parameters. The approximation seems remarkably good in these cases. These tables

Table I—Distribution of the number in queue 2 at the beginning of an interval with $\lambda_1 = 0.05, \lambda_2 = 0.10, b_1 = 1.0, b_2 = 1.0, T = 20.0$

| $k$ | $P(N_2 = k)$ Exact | $P(N_2 = k)$ Approximation Using $\bar{P}$ |
|---|---|---|
| 0 | 0.055308 | 0.055308 |
| 1 | 0.104504 | 0.104504 |
| 2 | 0.099035 | 0.099035 |
| 3 | 0.062750 | 0.062750 |
| 4 | 0.029901 | 0.029901 |
| 5 | 0.003648 | 0.003648 |
| 6 | 0.001001 | 0.001001 |
| 7 | 0.000241 | 0.000241 |
| 8 | 0.000052 | 0.000052 |

Table II—Distribution of the number in queue 2 at the beginning of an interval with $\lambda_1 = 0.05, \lambda_2 = 0.10, b_1 = 2.0, b_2 = 1.0, T = 20.0$

| $k$ | $P(N_2 = k)$ Exact | $P(N_2 = k)$ Approximation Using $\bar{P}$ |
|---|---|---|
| 0 | 0.062126 | 0.062126 |
| 1 | 0.109075 | 0.109075 |
| 2 | 0.097270 | 0.097270 |
| 3 | 0.058602 | 0.058602 |
| 4 | 0.026782 | 0.026782 |
| 5 | 0.009889 | 0.009889 |
| 6 | 0.003069 | 0.003069 |
| 7 | 0.000823 | 0.000823 |
| 8 | 0.000194 | 0.000194 |

Table III—Distribution of the number in queue 2 at the beginning of an interval with $\lambda_1 = 0.05, \lambda_2 = 0.10, b_1 = 3.0, b_2 = 1.0, T = 20.0$

| $k$ | $P(N_2 = k)$ Exact | $P(N_2 = k)$ Approximation Using $\bar{P}$ |
|---|---|---|
| 0 | 0.070641 | 0.070636 |
| 1 | 0.112676 | 0.112681 |
| 2 | 0.094152 | 0.094152 |
| 3 | 0.054193 | 0.054193 |
| 4 | 0.023988 | 0.023988 |
| 5 | 0.008665 | 0.008665 |
| 6 | 0.002651 | 0.002651 |
| 7 | 0.000704 | 0.000704 |
| 8 | 0.000166 | 0.000166 |

also suggest the region in which the approximation is likely to work well.

In general let $\bar{P}_m$ represent the approximation for $\hat{P}_m$ and for $1 \leq i \leq m-1$, let

$$\bar{P}_i(z_1, z_2, \cdots z_i) = \bar{P}_m(z_1, z_2, \cdots, z_i, 1, 1, \cdots 1). \qquad (26)$$

Then we can express $\bar{P}_i$ in terms of $\bar{P}_{i-1}$ as follows, thus enabling a simple recursive calculation of $\bar{P}_m$: For $\underline{z} = (z_1, z_2, \cdots, z_m)$, let

$$\phi_i(\underline{z}) = \lambda_i(1 - z_i),$$

$$\psi_m(\underline{z}) = \phi_m(\underline{z}),$$

and for $1 \leq i \leq m-1$,

$$\psi_i(\underline{z}) = \phi_i(\underline{z}) + \psi_{i+1}(\underline{z}) - \lambda_i(1 - \tilde{F}_i(-\psi_{i+1}(\underline{z}))). \qquad (27)$$

Then

$$\bar{P}_m(z_1, \cdots, z_m)$$
$$= e^{-T\sum_{i=1}^{m}\phi_i(\underline{z})} \prod_{i=1}^{m-1} \tilde{H}_i[-\psi_{i+1}(\underline{z})]$$
$$\cdot \bar{P}_m(\tilde{F}_1(-\psi_2(\underline{z})), \tilde{F}_2(-\psi_3(\underline{z})), \cdots, \tilde{F}_{m-1}(-\psi_m(\underline{z})), 1). \qquad (28)$$

This represents $P_m$ in terms of $P_{m-1}$. Applying this equation successively $m$ times we get $P_m$ in terms of $P_{m-1}, P_{m-2}, \cdots, P_1$ and $P_0 = P_m(1, 1, \cdots, 1) = 1$. Thus $P_m$ can be obtained recursively from eq. (28).

### 3.3 Moments for gating mechanism G2

We again look at the case $m = 2$ and obtain approximate expressions for the first two moments of $t_1$ and $t_2$, where $t_1$ and $t_2$ are as defined in Section 3.1. We also obtain the mean values of the sojourn times, $S_i$, $i = 1, 2$, assuming that the probability of an overrun is negligible. We compare these expressions with those for gating mechanism $G_1$ and also with the gating mechanism G2 but with indices 1 and 2 reversed, that is, with queue 2 served at higher priority than queue 1. We will use our approximation throughout this analysis. First, from eqs. (12) and (28) we get

$$\tilde{G}(s_1, s_2) = E[e^{-t_1s_1 - t_2s_2}]$$
$$= \tilde{H}_2(s_2)\tilde{H}_1(s_1 + s_2 + \lambda_2(1 - \tilde{F}_2(s_2))\tilde{H}_1(-\lambda_2(1 - \tilde{F}_2(s_2))))$$
$$\cdot e^{-T[\lambda_2(1 - \tilde{F}_2(s_2)) + \lambda_1[2 - \tilde{F}_1(s_1 + s_2}$$
$$+ \lambda_2(1 - \tilde{F}_2(s_2)) - \tilde{F}_1(-\lambda_2(1 - \tilde{F}_2(s_2))))]]. \qquad (29)$$

Putting $s_2 = 0 (s_1 = 0)$ in eq. (29) we get the approximate Laplace transform of $t_1(t_2)$. Let

$$\tilde{g}_1(s) = \text{Log } \tilde{G}(s, 0) \tag{30}$$

and

$$\tilde{g}_2(s) = \text{Log } \tilde{G}(0, s). \tag{31}$$

Then

$$E[t_i] = -\tilde{g}_i'(0^+) \tag{32}$$

and

$$\text{var}[t_i] = \tilde{g}_i''(0^+) \tag{33}$$

for $i = 1, 2$. Substituting from eq. (29) into eqs. (30) through (33), we get

$$E[t_1] = \alpha_1 + \rho_1 T, \tag{34a}$$

$$E[t_2] = \alpha_1 + \alpha_2 + (\rho_1 + \rho_2)T, \tag{34b}$$

$$\text{Var}[t_1] = \alpha_1 + \gamma_1 T, \tag{34c}$$

and

$$\text{Var}[t_2] = \alpha_1 + \alpha_2 + T(\gamma_1 + \gamma_2) + 2\rho_2(1 + \rho_2)(\alpha_1 + \gamma_1 T). \tag{35}$$

Comparing these with the expressions derived in Section 3.1 for the gating mechanism G1, we conclude that the mean values of $t_1$ and $t_2$ do not depend on the gating mechanism. The variance of $t_1$ also remains the same. However, the variance of $t_2$ for the gating $G_2$ is larger than that for gating $G_1$. The difference is

$$2\rho_2(1 + \rho_2)(\alpha_1 + \gamma_1 T) > 0. \tag{36}$$

This implies that the probability of an overrun is likely to be larger for gating G2 than for gating G1. The difference depends on the average load in queue 2 ($\rho_2$) and the variability of the load in queue 1 ($\alpha_1 + \gamma_1 T$). Since, in general, it is much easier to analyze G1 than to analyze G2, we frequently use the gating G1 as an approximation for gating G2. The left-hand side in (36) then gives a measure of this approximation. If this quantity is small the approximation is likely to be good. We will come back to this point in Section 3.4.

Next, we see what happens if we interchange queues 1 and 2. In other words, the parameters of the arrival and service time for these two queues are interchanged and we compare the moments of $t_1$ and $t_2$ for these two orders. We will use suffix 0 for the original order and $R$ for the reversed order. Then,

$$E_R[t_1] - E_0[t_1] = 0,$$

$$E_R[t_2] - E_0[t_2] = 0,$$

$$\mathrm{Var}_R[t_1] - \mathrm{Var}_0[t_1] = 0,$$

and

$$\mathrm{Var}_R[t_2] - \mathrm{Var}_0[t_2] = 2\rho_1(1 + \rho_1)(\alpha_2 + \gamma_2 T) - 2\rho_2(1 + \rho_2)(\alpha_1 + \gamma_1 T).$$

If we associated smaller variance with better performance, then the order (1, 2) is better than the order (2, 1) if

$$\frac{\rho_1(1 + \rho_1)}{\alpha_1 + \gamma_1 T} > \frac{\rho_2(1 + \rho_2)}{\alpha_2 + \gamma_2 T}. \qquad (37)$$

This suggests that if other things are equal the queue with smaller variability should be served first if we are concerned about the overruns.

We next evaluate the mean sojourn time, $\bar{S}_i$, in queue $i$, $i = 1, 2$, assuming that it gets served completely in the scheduled slot. We use eqs. (14), (16), (17), and (28) and some straightforward but tedious algebra to obtain

$$\bar{S}_1 = a_1 + b_1 + \frac{T}{2}(1 + \rho_1), \qquad (38)$$

and

$$\bar{S}_2 = a_2 + b_2 + \frac{T}{2}(1 + \rho_2) + \frac{\alpha_1 + \gamma_1 T}{T}(1 + \rho_2). \qquad (39)$$

Comparing eqs. (38) and (39) with eq. (7), we see that the mean value, $\bar{S}_1$, of $S_1$ is not affected by the gating mechanism. However, $\bar{S}_2$ does depend on the gating. If we use the suffixes 1 and 2 to denote gatings G1 and G2, respectively, then

$$\bar{S}_{22} - \bar{S}_{21} = \frac{\alpha_1 + \gamma_1 T}{T}(1 + \rho_2) - a_1 - \rho_1 T. \qquad (40)$$

Under the assumption that the probability of overrun is very small, the above difference is negative. Thus the mean sojourn time under gating G2 is smaller than under G1.

### *3.4 Numerical results*

In this section we use the analysis of Sections 3.1 through 3.3 to compute the probability of overrun and the sojourn time distributions for some special cases of the simple clocked schedule described earlier. For these numerical results we used the algorithm in Ref. 3 for inverting Laplace transforms. When the underlying function is not

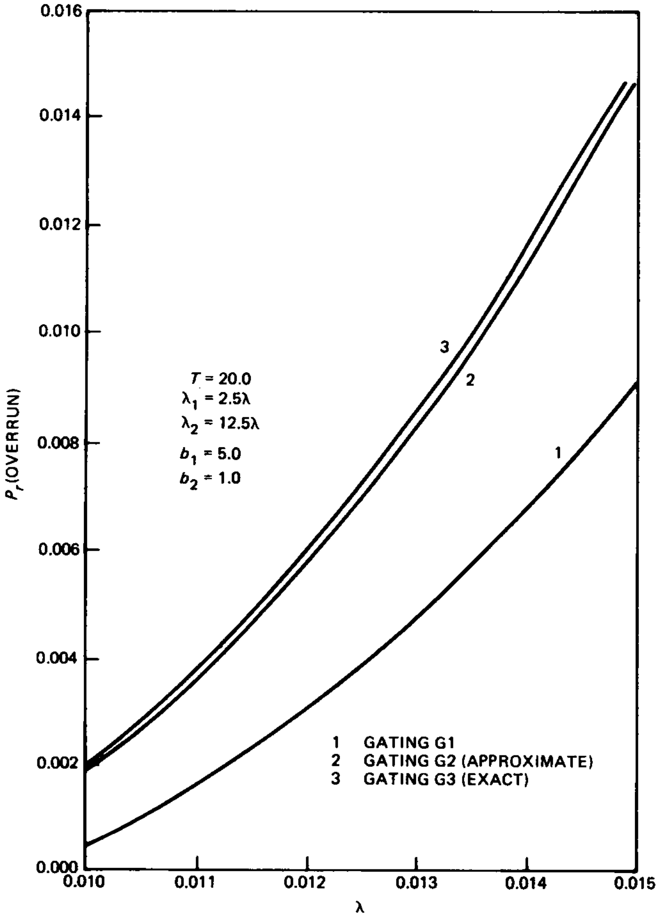

Fig. 7—$P_r${OVERRUN} as a function of load.

smooth enough, it may be more appropriate to discretize the problem and then use one of the routines for inverting the generating functions.

Suppose $m = 2$ and the overhead is zero. The service times are deterministic, 5 ms for queue 1 and 1 ms for queue 2. The length of a time slot is $T = 20$ ms. Finally, $\lambda_1 = 2.5\lambda$ and $\lambda_2 = 12.5\lambda$. Figure 7 shows the probability of overrun as a function of $\lambda$ for gating $G_1$, gating $G_2$ (exact), and gating $G_2$ (approximate). For gating G2 the exact and approximate results agree very well. For the selected values of the parameters the results for gating G1 and G2 differ significantly. The probability of overrun using gating G2 is higher than that using G1. This can be explained by the higher variance, as shown in Section 3.3. Alternatively, consider the work in queue 1 and queue 2 to be done in a time slot (see Fig. 8). For queue 1 the number of jobs to be served is

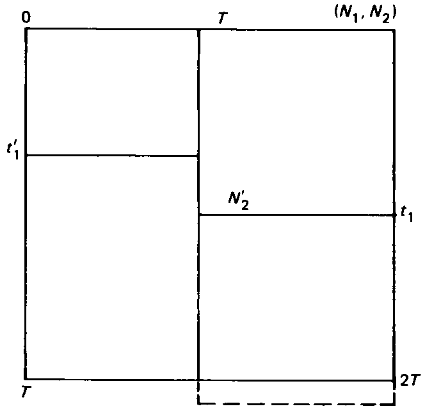

Fig. 8—Work content in queue 2.

the number that arrived in time $T$ for both gating mechanisms. The number of jobs to be served in queue 2 is the number that arrived in time $T$ for gating G1 and in time $T - t'_1 + t_1$, where $t'_1$ and $t_1$ are, respectively, times to complete all work in queue 1 in the preceding and current intervals. The mean value of $T - t'_1 + t_1$ is $T$, but because of the randomness in $T - t'_1 + t_1$, the amount of work arriving at queue 2 in that time interval has a larger tail than that of the work arriving in time $T$. This explains the larger probability of overrun. This also indicates that the difference in the two gating mechanisms is related to the variability in $t'_1$ and $t_1$, which corresponds to the variability in the work arriving at queue 1.

Next we reverse the order of service within a time slot. For gating G1 the probability of an overrun is not affected by this. However, for gating G2 this does change the probability of an overrun. Figure 9 shows the probability of an overrun as a function of $\lambda$ for gating G1, gating G2 with order (1, 2), and gating G2 with order (2, 1). We see considerable reduction in the probability of an overrun with order (2, 1). This is expected because queue 2 has less variable work load than queue 1 has. Also, as expected, the difference between gating G1 and G2 is much smaller with order (2, 1) than with order (1, 2).

We further investigate the effects of the order of service within a time slot by considering an example with three queues ($m = 3$). The service times and overheads are deterministic. The parameters are as shown in Table IV. Queues 2 and 3 have the same parameters, while queue 1 has more variable work load than either queue 2 or 3. The probability of an overrun with gatings G1 and G2 and orders (1, 2, 3), (2, 1, 3), and (2, 3, 1) are shown in Table IV. Once again, we observe that the probability of an overrun decreases as we push the more variable queue down in the priority order.

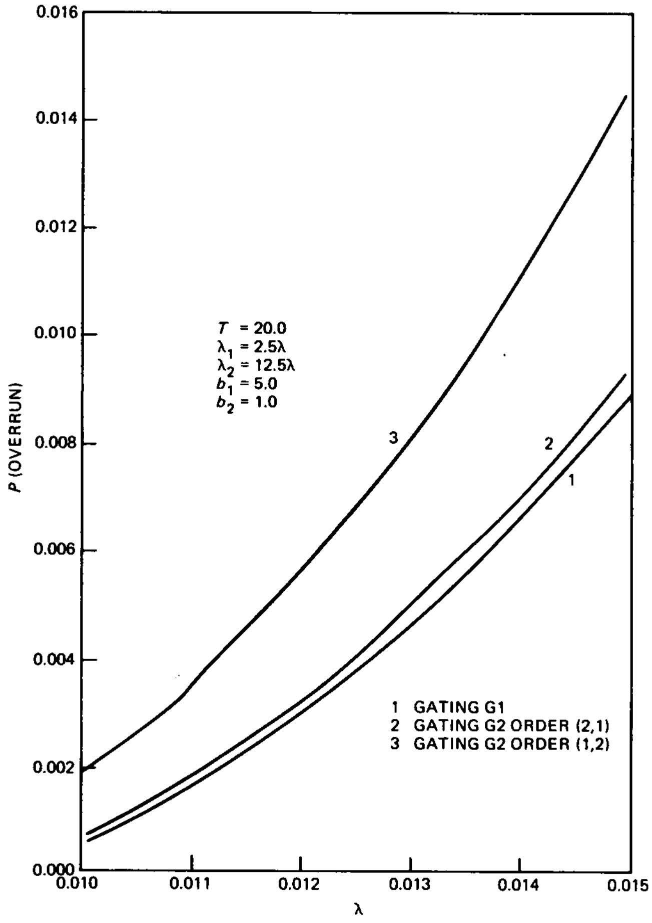

Fig. 9—$P$\{OVERRUN\} as a function of load.

Thus, from the moments considerations and from numerical results we have the following guidelines when the probability of an overrun is of concern:

1. The difference between the two gating mechanisms is greater when the queues with more variable work load are served earlier in a time slot.

2. The probability of an overrun is smaller if queues with more variable work load are pushed down in the priority order.

Of course, in most clocked schedules the differences in the time criticality of various tasks determine the order of service. The above guidelines, however, are useful when deciding order of service among tasks of comparable time criticality.

We next evaluate the sojourn time distributions numerically. The

Table IV—Probability of overrun

Number of Queues = 3, $T = 20.0$ Parameters

| Queue | Arrival Rate | Service Time | Overhead |
|---|---|---|---|
| 1 | 1.0 | 0.2 | 1.0 |
| 2 | 2.0 | 0.1 | 1.0 |
| 3 | 2.0 | 0.1 | 1.0 |

$P$\{OVERRUN\}

| Schedule | Gating G2 | Gating G1 | Gating G2 Approximation (First Order) |
|---|---|---|---|
| (1, 2, 3) | 0.027 | 0.015 | 0.025 |
| (2, 1, 3) | 0.023 | 0.015 | 0.022 |
| (2, 3, 1) | 0.021 | 0.015 | 0.020 |

case considered here is $m = 2$, no overhead, and deterministic service times. The parameters are $\lambda_2 = 0.05$, $b_2 = 2.0$, $\lambda_1 = 0.05$, and $b_1 = 1.0$. In Fig. 10 we have complementary sojourn time distribution, $1 - S_2(t)$, for queue 2 under gatings G1 and G2. As indicated earlier the delay for queue 2 with gating G1 is larger than with gating G2.

## IV. GENERALIZATIONS AND COMPLEX SCHEDULES

This section contains various generalizations of the basic methodology described in Section III. Some of these generalizations are aimed at relaxing underlying assumptions, while the others look at more complex schedules. To avoid presenting a lot of tedious and repetitive algebra we will only outline the derivations in this section. The details are similar to those in Section III.

### *4.1 The effects of queues feeding downward*

In Section III we assumed that the arrivals to different queues form independent Poisson processes. In many applications an arrival (job) may complete one task and immediately request another. Thus some of the task queues may feed the others. In typical situations the high-priority queues feed downward within an interval. That is, when a job is served in a high-priority queue, it may create an arrival directly to the internal queue of another task still to be served in that interval. Here, we indicate how to extend the results of Section III to include such feed-downs.

The basic clocked schedule is still the same as in Section III, but the arrival processes are now more general. Queue $i$ still has an *exogenous* Poisson arrival process at rate $\lambda_i$, $i = 1 \cdots m$. However, when a job is served in queue $i$, it creates an arrival to the internal

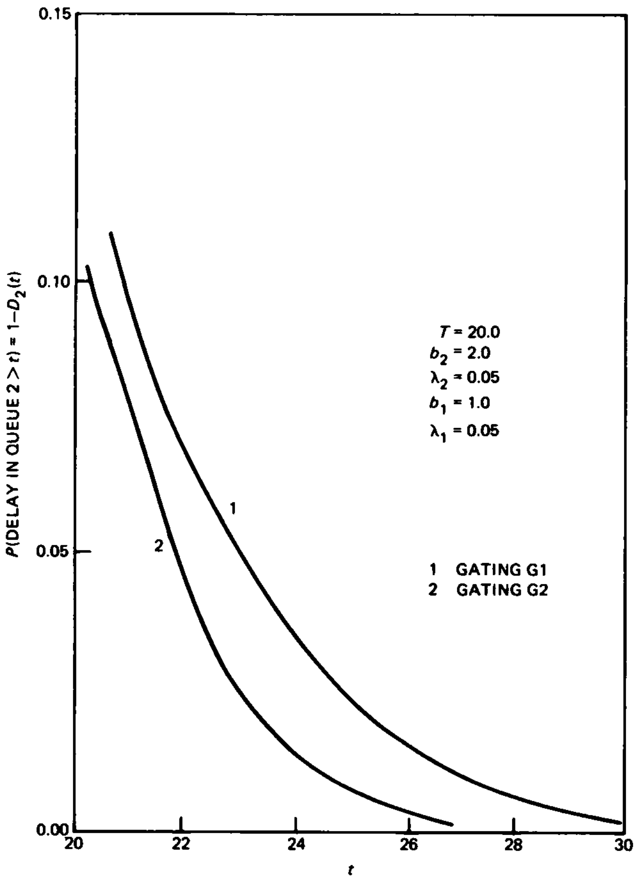

Fig. 10—Delay distribution in queue 2.

queue $j$ with probability $p_{ij}$. We assume

$$p_{ij} = 0 \quad j \leq i, \tag{41}$$

and

$$\sum_{j>i} p_{ij} \leq 1. \tag{42}$$

The Poisson arrivals may represent either genuinely exogenous arrivals or those generated by service completions in other queues in previous intervals. The Poisson approximation seems reasonable when the number of intervals between the completion of a task and the generation of a new one is large.

We consider the overrun strategy S1 and gating G2 (gating G1 is easier to analyze) and obtain the joint transform $\tilde{G}(s_1, s_2, \cdots s_m)$ of

$t_1, \cdots, t_m$. This can be done recursively as follows: We first obtain $\tilde{G}$ in terms of $\tilde{P}$ where

$$\tilde{P}(z_1, \cdots, z_m) = E[z_1^{N_1} z_2^{N_2} \cdots z_m^{N_m}].$$

We then express $\tilde{P}$ in terms of $\tilde{G}$ with the last argument being zero. Since $\tilde{G}(0, 0, \cdots 0) = 1$, we get both $\tilde{G}$ and $\tilde{P}$ by iterating on these two relations. We implicitly make the assumptions of the type made in Section III. Our expressions below are approximations that work when the probability that queue $i$ overruns is very small, $i = 1, \cdots, m - 1$.

For $\underline{z} = (z_1, z_2, \cdots, z_m)$, let $\phi_i(\underline{z}) = -\lambda_i(1 - z_i)$, $i = 1, \cdots m$. For $\underline{s} = (s_1, s_2, \cdots s_m)$, let

$$A_{i,m}(\underline{s}) = 1, \qquad i = 1, \cdots m,$$

$$B_m(\underline{s}) = s_m,$$

$$C_m(\underline{s}) = \lambda_m(1 - \tilde{F}_m(B_m(\underline{s}))A_{m,m}(\underline{s})),$$

and, for $i \leq j \leq m - 1$,

$$A_{i,j}(\underline{s}) = A_{i,j+1}(\underline{s})[1 - p_{i,j+1} + p_{i,j+1}\tilde{F}_{j+1}(B_{j+1}(\underline{s}))],$$

$$B_j(\underline{s}) = s_j + C_{j+1}(\underline{s}) + B_{j+1}(\underline{s}),$$

and

$$C_j(\underline{s}) = \lambda_j[1 - \tilde{F}_j(B_j(\underline{s}))A_{j,j+1}(\underline{s})].$$

Then

$$\tilde{G}(\underline{s}) = \prod_{i=1}^{m} \tilde{H}_i(B_i(\underline{s})) \cdot \tilde{P}(\tilde{F}_1(B_1(\underline{s}))A_{1,1}(\underline{s}), \cdots, \tilde{F}_m(B_m(\underline{s}))A_{m,m}(\underline{s})), \quad (43)$$

and

$$\tilde{P}(\underline{z}) = e^{T\sum_{i=1}^{m}\phi_i(\underline{z})}\tilde{G}(\phi_2(\underline{z}), \cdots, \phi_m(\underline{z}), 0). \quad (44)$$

We next study the effect of these feed-downs numerically. We consider the case $m = 2$, no overhead, and deterministic service times with $b_1 = b_2 = 1.0$. Let $p_{12} = p$. We consider six cases with $p$ varying from 0 to 1 in steps of 0.2. The exogenous arrival rate $\lambda_2$ in queue 2 is adjusted so that

$$\lambda_2' = \lambda_2 + p\lambda_1 = \lambda_1,$$

that is,

$$\lambda_2 = \lambda_1(1 - p)$$

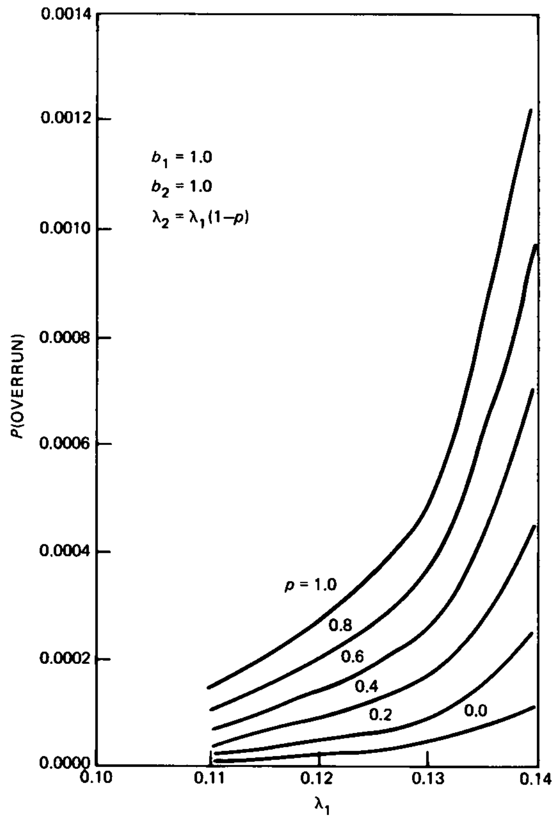

Fig. 11—$P$\{OVERRUN\} with feed-down.

for all six cases. $\lambda_1$ is varied from 0.1 to 0.15. The case $p = 0$ corresponds to no feed-down. Increasing $p$ corresponds to increasing feed-down and hence increasing correlation between the queues in an interval. The probability of overrun for each of these six cases is presented in Fig. 11. At large $p$, the effect of feed-down is clearly very significant.

## *4.2 Complex schedules*

In Section III we analyzed a simple clocked schedule in which each high-priority queue is scheduled in each interval. Although such schedules occur in practice, in most applications a subset of the set of high-priority tasks is scheduled in each interval. This subset changes from one time interval to the next. The entire schedule may repeat after $N_{\text{period}}$ time slots. Such a schedule is a clocked schedule with time slot length $T$ and schedule period $N_{\text{period}}$. Figure 12 illustrates a schedule with period 2.

In theory we can extend the results of Section III to analyze a

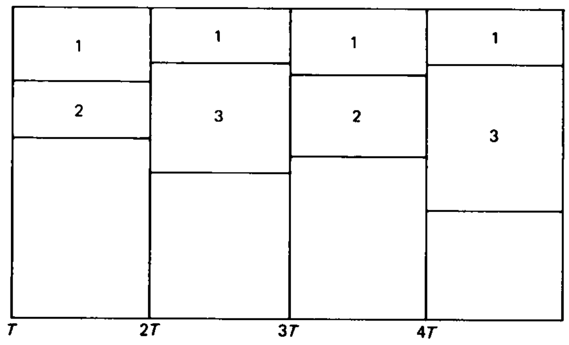

Fig. 12—A clocked schedule with period $2T$.

schedule with period $N_{\text{period}}$. For overrun strategy S1 and gating mechanism G1, the analysis is straightforward and can be carried out individually for each time slot within a period. For gating G2 with overrun strategy S1, note that

$$\{(N_{nN_{\text{period}}+1,1}, N_{nN_{\text{period}}+1,2}, \cdots, N_{nN_{\text{period}}+1}, m{:}n \geq 1)\}$$

is a time-homogeneous Markov chain and its balance equations can be written down as in Section III. We can also use the techniques of Section III to obtain transforms $\tilde{G}$ for each time slot in a period in terms of $\tilde{P}$, the joint generating function of the steady-state numbers in $m$ queues at the beginning of a period.

However, for large $N_{\text{period}}$ both the analysis and numerical evaluations may be difficult because of the dimensionality. Moreover, we need individual analysis for each schedule. When such difficulties arise, the approximation discussed below may be an appropriate alternative. This approximation applies uniformly to a wide variety of complex schedules and, for each slot, requires the same computational effort as for the simple schedule of Section III.

In order to understand this approximation consider an interval in which $m_1$ of the $m$ high-priority queues are scheduled. For simplicity we number them from 1 through $m_1$, the order in which they are scheduled for service. Let $T$ denote, as before, the length of the interval and, for $i = 1, 2 \cdots m_1$, let $T_i$ denote the nominal time between service initiations for queue $i$ (that is, if all queues complete service at their expected times, then the time between two successive starts of service to queue $i$ is $T_i$). For the simple schedule of Section III, $T_i = T$ *for* $i = 1, \cdots m$. For the schedule in Fig. 12, $T_1 = T$, $T_2 = T_3 = 2T$. Let

$$T_i' = T_i - \sum_{j=1}^{i-1} (a_j + \rho_j T_j). \tag{45}$$

Then the approximation for $\tilde{P}$ is given by

$$\tilde{P}(z_1, z_2, \cdots z_{m_1}) = e^{-\sum_{i=1}^{m_1} \lambda_i T_i'(1 - z_i)}. \tag{46}$$

The relations expressing $\tilde{G}(s_1, s_2, \cdots s_{m_1})$ and $\tilde{S}_i(s)$ in terms of $\tilde{P}$ remain the same as in Section III.

Recall that in Section III we calculated the probability of overrun for $m = 3$. The parameter values and the probability of overrun are given in Table IV. For the same parameter values we calculated the probability of overrun for gating G2 using the approximation described above. The results given in the last column indicate that the approximation works well in this case. A little reflection suggests that for schedules with longer periods, the approximation should be even better.

### *4.3 Other overrun strategies*

In Section III we analyzed clocked schedules with overrun strategy S1. In theory, it is possible to extend these results to the schedules using overrun strategies S2 and S3. For strategy S2, $\{(N_{n1}, N_{n2}, \cdots N_{nm}):n \geqslant 1\}$ is still a time-homogeneous Markov chain. Its balance equations can be written down easily, but they do not have the triangular structure of those for strategy S1. Thus, these equations cannot be solved recursively. They can, however, be solved as a system of linear equations after suitably discretizing the underlying distributions. For strategy S3 an overrun in one time slot will reduce the time available in the following slot. In this case $\{N_{n1}, N_{n2}, \cdots N_{nm}, U_n):n \geqslant 1\}$ is a Markov chain, where $N_{ni}$ is the number in the external queue $i$ when the server visits queue 1 for the $n$th time and $U_n$ is the time remaining in that interval. Again, the steady-state distribution of this Markov chain can be obtained by solving a system of linear equations obtained by suitably discretizing the underlying distributions.

The numerical complexity involved in the above procedures may preclude their use for all but small values of $m$. For large $m$, some approximation or relatively easy to obtain bounds are needed. Note that for strategies S2 and S3 the probability of an overrun is bounded below by the probability of an overrun for strategy S1, because strategy S1 ignores the work carried over from an overrun interval into the next. If the amount of overrun is probabilistically small this bound must be quite close.

For strategy S3 we can also get an approximation if we assume that overrun does not extend beyond one clock interval. This approximation is obtained by adding a dummy queue, say queue 0, at the beginning of each schedule interval. The overhead in this queue represents the work carried over from the preceding interval. The

Table V—$P\{\text{OVERRUN}\}$ for strategy S3 with $T = 20.0$, $\lambda_1 = 0.1$, $b_1 = 1.0$, and $b_2 = 5.0$

| | $P\{\text{OVERRUN}\}$ | |
|---|---|---|
| $\lambda_2$ | Approximation $S_1$ | Approximation Using Dummy Queue |
| 0.025 | 0.003278 | 0.003427 |
| 0.05 | 0.023526 | 0.026999 |

arrival rate to this queue is assumed to be zero. We then use the following iterative procedure:

Step 1—Assume that the overhead in queue 0 is 0.

Step 2—Using the results of Section III for strategy S1, obtain the distribution of the carried over work, $(t_m - T)^+$.

Step 3—Assume that the overhead in queue 0 has the distribution of $(t_m - T)^+$ obtained in Step 2.

Repeat Steps 2 and 3 until two successive iterations give "close" values for the distribution of $(t_m - T)^+$.

Numerical values of $P\{\text{OVERRUN}\}$ calculated using strategy S1 and the approximation for strategy S3 are given in Table V. The parameter values are $m = 2$, no overhead, deterministic service times, $b_1 = 1.0$, $b_2 = 5.0$, $\lambda_1 = 0.1$, and $\lambda_2 = 0.025$ and 0.05.

## V. CONCLUSIONS

We presented a methodology to analyze the performance measures for the high-priority tasks in clocked schedules. The analysis was exact for a simple clocked schedule. For more complex clocked schedules this analysis formed the basis for easy to use approximations.

## REFERENCES

1. A. A. Fredericks, B. L. Farrell, and D. F. DeMaio, "Approximate Analysis of a Generalized Clocked Schedule," AT&T Tech. J., this issue.
2. M. H. Ackroyd, "Numerical Computation of Delays in Clocked Schedules," AT&T Tech. J., this issue.
3. D. L. Jagerman, "An Inversion Technique for the Laplace Transform With Application to Approximation," B.S.T.J., *57* (March 1978), pp. 669–710.
4. L. Kleinrock, *Queueing Systems*, New York: Wiley, 1975.
5. A. Fredericks, "Analysis and Design of Processor Schedules for Real Time Applications," *Applied Probability-Computer Science, The Interface I*, edited by R. Disney and T. Ott, Boston, MA: Birkhäuser, 1982, pp. 433–50.
6. D. Manfield and P. Trans-Gia, "Queueing Analysis of Scheduled Communications Phases in Distributed Processing Systems," Performance 81, Proc. of 8th Int. Symp. on Comp. Perf. Mod., Meas. and Eval., Amsterdam, November 4–6, 1981, edited by F. J. Kylstra, Amsterdam: North-Holland, pp. 233–50.
7. A. Fredericks, "Analysis of a Class of Schedules for Computer Systems With Real Time Applications," *Performance of Computer Systems*, edited by M. Aroto, A. Butrimenko, and E. Gelenbe, Amsterdam: North-Holland, 1979, pp. 201–16.

8. H. Jans, "On Queueing Systems With Clocked Operation and Priorities," Proc. 10th ITC, Montreal, 1983, Section 4.4a, Paper No. 4.

## AUTHOR

**Bharat T. Doshi,** B. Tech. (Mechanical Engineering), 1970, I.T.T. Bombay; Ph.D. (Operations Research), 1974, Cornell University; AT&T Bell Laboratories, 1979—. Before joining AT&T Bell Laboratories Mr. Doshi was an Assistant Professor at Rutgers University. At AT&T Bell Laboratories his technical work includes modeling and analysis of processor schedules, overload controls, and capacity estimation. His research interests are queueing theory and scheduling theory applied to the performance analysis of computer, communication, and production systems. Member, IEEE, ORSA; Associate Editor, OR Letters.