Probability in the Engineering and Informational Sciences, 7, 1993, 273–289. Printed in the U.S.A.

STOCHASTIC SCHEDULING IN PRIORITY QUEUES WITH STRICT DEADLINES

DIMITRIOS G. PANDELIS AND DEMOSTHENIS TENEKETZIS
Department of Electrical Engineering and Computer Science
The University of Michigan
Ann Arbor, Michigan 48109

Tasks belonging to N priority classes arrive for processing in a single or multiserver facility. If the processing does not begin by a certain time (deterministic or random), the task is lost and a cost is incurred. We determine properties of dynamic, nonidling, nonpreemptive strategies that minimize an infinite horizon expected cost.

1. INTRODUCTION

In this paper we study the problem of optimally scheduling tasks (e.g., messages to be transmitted, jobs to be processed, customers to be served) that have constraints on their waiting time. Each arriving task has a due date or deadline associated with the beginning of its service. This deadline either is known to the scheduler or has a known probability distribution. If the service of a task does not begin before its deadline expires, the task leaves the system and is considered lost. Tasks may have different priorities, which are modeled by different penalties incurred when a loss occurs (higher priorities correspond to higher penalties). We consider both single-server and multiserver problems. We determine properties of dynamic nonidling, nonpreemptive scheduling strategies that minimize an infinite horizon expected cost due to the tasks lost.

This research was supported in part by the National Science Foundation under grant NCR-9204419.

 0269-9648/93 $5.00 + .00

Work on deadlines was first motivated by job shop applications (for results in this area, see Conway, Maxwell, and Miller [7]). Similar issues are encountered in maintenance and replacement problems (see, e.g., Derman, Lieberman, and Ross [8]). Another class of problems, falling in the content of queueing theory, is that of impatient customers who leave if their waiting time exceeds a certain limit (see Baccelli, Boyer, and Hebuterne [1], Charlot and Pujolle [5], and Stanford [14]). Related problems include limitations on the sojourn time (see Takacs [15]) and system failure if due dates are not met (see Baccelli and Trivedi [2]). The computational complexity of a problem with due dates is studied in Cho and Sahni [6]. Modern applications involving tasks with priorities and deadlines are found in computer and communication networks. Consider, for example, a single channel used for the transmission of different types of messages (voice, video, data files, etc.), with one of them, say voice, having priority over the others.

Until now most of the research on stochastic scheduling with deadlines has concentrated on performance evaluation of ad hoc policies rather than on optimization. The assumption is usually that of first-come first-serve (FCFS) or shortest time to extinction (STE) policy. For results on optimization see Bhattacharya and Ephremides [3] and Panwar, Towsley, and Wolff [11], where the optimality of the STE and STEI (STE when idling is permitted) policies is established for systems with one class of tasks, and Pinedo [12] and Huang and Weiss [10], where simple optimal policies are derived for problems for which there are no arrivals and the deadlines have a known distribution. Recently, Bhattacharya and Ephremides [4] considered a single-server scheduling problem for tasks that belong to two different classes. They considered deadlines with respect to the end of the service and assumed that tasks missing their deadlines are not lost but stay in the system and incur a penalty at a rate depending on the task's class. Our work differs from that of Bhattacharya and Ephremides [3] and Panwar et al. [11] in that it deals with N classes of customers. It differs from that of Pinedo [12] because it considers systems with arrivals. It is also different from that of Bhattacharya and Ephremides [4] because it deals with multiclass multiserver systems where the deadlines are with respect to the beginning of the service and the cost function is different (i.e., tasks that miss their deadlines are lost).

The contributions of this paper are the following. For single-server and multiserver systems and deterministic deadlines we prove that within a priority class it is optimal to process the task with the earliest deadline. In choosing between the tasks with the earliest deadlines in two different classes, it is optimal to process the task that incurs the higher penalty if the deadline of the other task is greater than a threshold that depends on the state of the system. Moreover, for multiserver systems we prove that when the number of available servers is greater than the number of waiting tasks it is optimal to use the fastest servers first. We also show that results similar to the preceding hold for systems with stochastic deadlines.

This paper is organized as follows. In Section 2 we consider the situation in which the scheduler is informed of the deadline of each task upon its arrival. The case in which only the probability distribution of the deadlines is available to the scheduler is studied in Section 3. Finally, in Section 4 we consider variations of the problems studied in the previous sections. We study the problem of routing tasks with deadlines to two similar servers; we also study the effect the inclusion of a switching cost, incurred when a server moves from one type of task to another, has on the optimal policy derived in Section 2.

2. DETERMINISTIC DEADLINES

2.1. The Single-Server Problem

We consider a single-server queueing system where the tasks to be processed arrive according to a Poisson process of rate λ and have exponentially distributed service times with parameter μ. We assume that the arrival and service processes are independent and $\lambda < \mu$. Each arriving task belongs to one of N different classes and has a deadline associated with the beginning of its service that becomes known at the arrival instant. Let d_i be the ith task's deadline. We assume that d_i, $i = 1,2,\ldots$, form a sequence of independent random variables that are also independent of the arrival and service processes. No distribution on deadlines of future arrivals is assumed.

Consider a task of type $i, i = 1,2,\ldots,N$, whose arrival time and deadline are t and d, respectively. If the processing of this task does not begin by its extinction time $e = t + d$, it is lost and incurs a cost C_i, where $C_1 > C_2 > \cdots > C_N$. We assume that $C_N > (\lambda/\lambda + \mu)C_1$. Let Π be the class of nonidling and nonpreemptive policies. Our objective is to find a policy $\pi^* \in \Pi$ such that for any other policy $\pi \in \Pi$ we have

$$\liminf_{t\to\infty}(J_t^{\pi} - J_t^{\pi^*}) > 0, \tag{1}$$

where J_t^{π} is the expected cost incurred under policy π until time t; i.e.,

$$J_t^{\pi} = \sum_{i=1}^{N} C_i E(X_{i,t}^{\pi}),$$

$X_{i,t}^{\pi}$ being the number of type i tasks lost under π until time t.

Note that the expected cost under any policy in Π may go to infinity when $t \to \infty$. Then, according to our optimality criterion, π^* is optimal in the sense that it incurs a cost that, when it goes to infinity, is at a slower rate than the cost incurred by any other policy. Because we consider nonidling and nonpreemptive policies, a decision has to be made when a service is completed and there is more than one task waiting in the queue. Let t_0, $t_0 \geq 0$, be such a decision point and $M^i(t_0)$, $i = 1,2,\ldots,N$, be the set of eligible tasks of type i at time t_0, i.e., the set of tasks of type i whose extinction times are greater than

t_0. The control action is to decide which task belonging to $M(t_0) = \bigcup_{i=1}^{N} M^i(t_0)$ to process.

For $M^i(t_0) \neq \emptyset$ let $E^i(t_0) = \{e_1^i, \ldots, e_{n_i}^i\}$, $n_i \geq 1$, be the set of extinction times of eligible tasks of type i at time t_0 arranged in increasing order. From now on e_j^i will denote both the time instant e_j^i and the task that has extinction time e_j^i. We introduce the following notation: The basic sample space is denoted by Ω. For a policy π and $S \subset \Omega$, $V_{S,t}^{\pi}$ denotes the expected cost incurred under π along S until time $t \geq t_0$. An optimal policy for the preceding single-server problem possesses the properties given in the following theorem.

THEOREM 1: *Consider a decision instant t_0. Then*

(i) *Within a class of tasks it is optimal to process the one with the shortest extinction time.*

(ii) *For each $i < j$ with $M^i(t_0) \neq \emptyset$, $M^j(t_0) \neq \emptyset$ there exists a time instant (threshold) $t_{ij} \leq e_1^i$ such that it is optimal to process task e_1^i instead of e_1^j if $e_1^j \geq t_{ij}$, and vice versa otherwise.*

PROOF:

(i) Consider a class i of tasks with at least two eligible tasks at time t_0. For every policy $\pi \in \Pi$ that processes task e_k^i, $k \neq 1$, at time t_0, $\tilde{\pi} \in \Pi$ processes e_1^i and is identical to π when π processes tasks other than e_1^i. When π processes (if ever) task e_1^i, $\tilde{\pi}$ processes task e_k^i. We have two cases.

Case 1: Policy π processes task e_1^i. Then task e_k^i is processed under $\tilde{\pi}$ at the same time. Therefore, along $\Omega_1 = \{\omega \in \Omega \mid \pi \text{ serves } e_1^i\}$ we have $V_{\Omega_1,t}^{\pi} = V_{\Omega_1,t}^{\tilde{\pi}}$, $t \geq t_0$.

Case 2: Policy π does not process task e_1^i. Let τ be the end of the busy period[1] under π. If $\tau \geq e_k^i$, $\tilde{\pi}$ loses e_k^i. So along $\Omega_2 = \{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i, \tau(\omega) \geq e_k^i\}$ we have $V_{\Omega_2,t}^{\pi} = V_{\Omega_2,t}^{\tilde{\pi}}$, $t \geq e_k^i$.

If $\tau < e_k^i$, $\tilde{\pi}$ starts serving e_k^i at time τ. If no arrivals occur during the service of e_k^i, π and $\tilde{\pi}$ lose the same tasks and, in addition, π loses e_1^i. So along $\Omega_3 = \{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i, \tau(\omega) < e_k^i, \text{ first event after } \tau \text{ is service}\}$ we have $V_{\Omega_3,t}^{\pi} - V_{\Omega_3,t}^{\tilde{\pi}} = C_i$, $t \geq e_1^i$.

If at least one task, say d, arrives during the service of e_k^i, π starts d. Because the server is exponential, the services of d under π and e_k^i under $\tilde{\pi}$ end at the same time, and from that point on $\tilde{\pi}$ follows π and serves d, if eligible, at the end of the busy cycle under π, in which case the argument is repeated. Therefore, along $\Omega_4 = \{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i, \tau(\omega) < e_k^i, \text{ first event after } \tau \text{ is arrival}\}$ we have $V_{\Omega_4,t}^{\pi} - V_{\Omega_4,t}^{\tilde{\pi}} \geq C_i - C_1$, $t \geq e_1^i$.

From the preceding analysis we obtain

$$J_t^{\pi} - J_t^{\tilde{\pi}} \geq C_i P(\Omega_3) + (C_i - C_1) P(\Omega_4), \qquad t \geq e_k^i. \tag{2}$$

Since the arrival and service processes are independent, we get

$$P(\Omega_3) = P(\{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i, \tau(\omega) < e_k^j\}) \frac{\mu}{\lambda + \mu}, \quad (3)$$

$$P(\Omega_4) = P(\{\omega \in \Omega \mid \pi \text{ does not serve } e_1^j, \tau(\omega) < e_k^i\}) \frac{\lambda}{\lambda + \mu}. \quad (4)$$

From Eqs. (2)–(4) and the assumptions $C_1 > C_2 > \cdots > C_N$, $C_N > (\lambda/\lambda + \mu)C_1$, we get $J_t^{\pi} > J_t^{\tilde{\pi}}$, $t \geq e_k^j$, hence, among all tasks of type i it is optimal to process the one with the shortest extinction time.

(ii) Let e^i and e^j be tasks of type i and j $(i < j)$, respectively, that are eligible at time t_0. Let the deadlines of all eligible tasks except e^j be fixed. Then the statement of part (ii) of the theorem is a direct consequence of the following two lemmas. ■

LEMMA 1: *If $e^j \geq e^i$ it is optimal to serve e^i instead of e^j.*

PROOF: The arguments needed are exactly the same as those used in the proof of part (i) of Theorem 1 and are not repeated here. ■

LEMMA 2: *Suppose that for $e^j = k < e^i$ it is optimal[2] to serve e^i instead of e^j at t_0. Then for $e^j = \ell$, where $\ell > k$, it is still optimal to serve e^i.*

PROOF: To avoid confusion we will attach to each policy a subscript denoting the value of e^j. For example, π_k would be a policy applied to the set of tasks with $e^j = k$.

Consider a policy π_ℓ that processes task e^j at time t_0. Let π_k' be the policy that processes e^j at time t_0 and is identical to π_ℓ afterward. Then clearly $J_t^{\pi_\ell} = J_t^{\pi_k}$, $t \geq t_0$. By assumption there exists a policy $\tilde{\pi}_k'$ that processes task e^i at time t_0 and does better than π_k'; i.e., there exists a time T such that $J_t^{\pi_k} > J_t^{\tilde{\pi}_k}$, $t \geq T$. Consider now a policy $\tilde{\pi}_\ell$ that is identical to $\tilde{\pi}_k'$, except that it processes $e^j = \ell$ when (if ever) $\tilde{\pi}_k'$ processes $e^j = k$. Two cases exhaust all possibilities.

Case 1: $\tilde{\pi}_k'$ serves e^j.

Case 2: $\tilde{\pi}_k'$ does not serve e^j.

Using the same arguments as in the proof of part (i) of Theorem 1, we obtain $J_t^{\tilde{\pi}_k} > J_t^{\tilde{\pi}_\ell}$, $t \geq \ell$, but $J_t^{\tilde{\pi}_k} < J_t^{\pi_k} = J_t^{\pi_\ell}$, $t \geq T$. Thus, $J_t^{\pi_\ell} > J_t^{\tilde{\pi}_\ell}$, $t \geq \max\{T, \ell\}$, so it is optimal to serve e^i when $e^j = \ell$. ■

According to parts (i) and (ii) of the theorem, at each decision instant we need only consider the tasks in the set $I(t_0) = \{j \mid e_1^i > e_1^j, \forall i < j\}$. If L is the cardinality of $I(t_0)$, to determine which task is optimal we need to make $L - 1$ pairwise comparisons—in other words, compute $L - 1$ thresholds. In this paper we shall not discuss how to compute the thresholds. Note that a threshold t_{ij} does not only depend on e_1^i, e_1^j, but on the whole set of eligible tasks at time t_0 as well. We illustrate this point by the following example. We have two classes

of tasks with $E^1(0) = \{d\}$, $E^2(0) = \{d_1, d_2\}$, where $d_1 < d < d_2$. The penalties for missed deadlines are C_1 and C_2 for classes 1 and 2, respectively, where $C_1 > C_2$. We have no arrivals ($\lambda = 0$). Let π_1 and π_2 be the policies that start with d and d_1, respectively, and proceed optimally afterward. According to Theorem 1, after the first service completion policies π_1 and π_2 give priority to tasks d_1 and d, respectively. The expected costs due to policies π_1 and π_2 are

$$J^{\pi_1} = C_2 \mu d_1 e^{-\mu d_2} + C_2 e^{-\mu d_1} + C_2 e^{-\mu d_2},$$

and

$$J^{\pi_2} = C_2 \mu d e^{-\mu d_2} + C_1 e^{-\mu d} + C_2 e^{-\mu d_2},$$

so the cost difference of policies π_1 and π_2 is

$$J^{\pi_1} - J^{\pi_2} = C_2 \mu e^{-\mu d_2}(d_1 - d) + C_2 e^{-\mu d_1} - C_1 e^{-\mu d}.$$

The threshold t_{12} is equal to $\max\{0, t_{12}^*\}$, where t_{12}^* is the value of d_1 for which $J^{\pi_1} - J^{\pi_2} = 0$. Clearly, t_{12} depends on d_2.

2.2. The Multiserver Problem

We consider now a queueing system consisting of M parallel servers $S_1, S_2, \ldots, S_M$. The service times at server S_i, $i = 1,2,\ldots,M$, are exponentially distributed with parameter μ_i, where $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_M$. Tasks arrive in a Poisson stream of intensity λ, where $\lambda < \sum_{i=1}^{M} \mu_i$, and have a deadline with respect to the beginning of their service. Similar to the single-server problem, tasks belong to N classes, with tasks of class i incurring a cost C_i when they do not meet their deadlines, where $C_1 > C_2 > \cdots > C_N$ and $C_N > (\lambda/\lambda + \mu_M)C_1$. We assume independence of arrivals, services, and deadlines. Let Π' be the class of nonidling and nonpreemptive list scheduling strategies, i.e., strategies that use the servers in a predetermined order. Our objective is to find a policy in Π' that satisfies the optimality criterion (1).

For decision instants for which the number of eligible tasks is greater than the number of empty servers $S_{i_1}, S_{i_2}, \ldots, S_{i_{M'}}$, $M' \leq M$, the problem is to determine the M' tasks to be processed. It does not matter which task goes to which server because the deadlines are with respect to the beginning of the service and the duration of the service does not affect the cost. For decision instants for which the number of eligible tasks is less than the number of empty servers, the problem is to determine which servers to use. Each policy in Π' selects the servers according to a prespecified priority rule $(S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_M})$, where $(\ell_1, \ell_2, \ldots, \ell_M)$ is a permutation of $(1,2,\ldots,M)$. The following theorem describes the characteristics of the optimal policy for both preceding cases.

THEOREM 2:

(i) *Let t_0 be a decision instant for which the number of eligible tasks is greater than M', the number of available servers. Consider empty server S_{i_ℓ}, $1 \leq \ell \leq M'$. Then*

(ia) *Within a class of tasks it is optimal to assign to S_{i_t} the one with the shortest extinction time.*

(ib) *For each $i < j$ with $M^i(t_0) \neq \emptyset$, $M^j(t_0) \neq \emptyset$ there exists a time instant (threshold) $t_{ij} \leq e_1^i$ such that it is optimal to assign to S_{i_t} task e_1^i instead of e_1^j if $e_1^j \geq t_{ij}$, and vice versa otherwise.*

(ii) *For decision instants for which the number of eligible tasks is less than the number of available servers the optimal policy in Π' (the class of nonidling, nonpreemptive list scheduling strategies) ranks the servers in decreasing order of their service rates, i.e., uses the fastest available servers.*

Discussion: Note that when the number of eligible tasks is larger than the number of available processors the properties of the optimal policy stated in Theorem 2 are essentially identical to those in Theorem 1. Therefore, to select the M' tasks to be processed we have to follow the procedure for the single-server problem M' times. First we determine the "best" task by computing the appropriate thresholds, and then we determine the "best" among the rest of the tasks, and so on.

PROOF:

(i) The proof of parts (ia) and (ib) is very similar to that of Theorem 1, so we shall only emphasize points where the two proofs are different.

(ia) Consider a class i of tasks with at least two eligible tasks at time t_0. For every policy $\pi \in \Pi'$ that assigns to S_{i_t} task e_k^i, $k \neq 1$, at time t_0 we construct a policy $\pi' \in \Pi'$ as follows: At time t_0 π' assigns S_{i_t} to task e_1^i and is identical to π when π processes tasks other than e_1^i. When π processes (if ever) task e_1^i at some server, π' processes task e_k^i at the same server. With τ being the end of the busy period[3] under π (the period for which all servers are busy under π), we have $V^{\pi}_{\Omega_1,t} = V^{\pi'}_{\Omega_1,t}$, $t \geq t_0$, for $\Omega_1 = \{\omega \in \Omega \mid \pi \text{ serves } e_1^i\}$, and $V^{\pi}_{\Omega_2,t} = V^{\pi'}_{\Omega_2,t}$, $t \geq e_k^i$, for $\Omega_2 = \{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i,\ \tau(\omega) \geq e_k^i\}$.

If π does not serve e_1^i and $\tau < e_k^i$, at τ π' starts serving e_k^i at some server S_t (which is empty under π). If no arrivals occur during the service of e_k^i, or all arrivals during the first busy period under π' find some other server empty, π and π' lose the same tasks and π loses in addition task e_1^i. So, for

$$\Omega_3 = \{\omega \in \Omega \mid \pi \text{ does not serve } e_1^i,\ \tau(\omega) < e_k^i, \text{ no arrivals occur during the service of } e_k^i \text{ or all arrivals after } \tau \text{ during the first busy period under } \pi' \text{ find some empty server}\},$$

we have

$$V^{\pi}_{\Omega_3,t} - V^{\pi'}_{\Omega_3,t} = C_i, \qquad t \geq e_1^i.$$

For $\omega \in \Omega_4 = \Omega - (\Omega_1 \cup \Omega_2 \cup \Omega_3)$ it may happen that a task d arrives during the first busy period under π' and finds all servers busy under π' and one server free under π. Task d may or may not be lost under π'; thus, along $\omega \in \Omega_4$ we have

$$V^{\pi}_{\Omega_4,t} - V^{\pi'}_{\Omega_4,t} \geq C_i - C_1, \qquad t \geq e^i_1.$$

Thus, for the cost difference between π and π' we have

$$J^{\pi}_t - J^{\pi'}_t \geq C_i P(\Omega_3) + (C_i - C_1)P(\Omega_4), \qquad t \geq e^i_1. \tag{5}$$

Because the arrival and service processes are independent we get

$$P(\Omega_3) = P(\{\omega \in \Omega \mid \pi \text{ does not serve } e^i_1, \tau(\omega) < e^i_k\}) \times \left(\frac{\mu_i}{\lambda + \mu_i} + B^{\pi'}\right), \tag{6}$$

$$P(\Omega_4) = P(\{\omega \in \Omega \mid \pi \text{ does not serve } e^i_1, \tau(\omega) < e^i_k\}) \times \left(\frac{\lambda}{\lambda + \mu_i} - B^{\pi'}\right), \tag{7}$$

where

$B^{\pi'} \triangleq P$ (all arrivals after τ and during the first busy period under π' find some empty server).

From Eqs. (5)–(7) and the assumptions $C_1 > C_2 > \cdots > C_N$ and $C_N > (\lambda/\lambda + \mu_M)C_1$, which imply that $C_i > ((\lambda/\lambda + \mu_i) - B^{\pi'})C_1$, we get

$$J^{\pi}_t > J^{\pi'}_t, \qquad t \geq e^i_1;$$

hence, among all tasks of type i it is optimal to assign to S_{i_t} the one with the shortest extinction time.

Remark: Note that the result of part (ia) is true if the assumption $C_N > (\lambda/\lambda + \mu_M)C_1$ is replaced by

$$C_N > \left(\frac{\lambda}{\lambda + \mu_M} - \inf_{\pi' \in \Pi'} B^{\pi'}\right) C_1.$$

(ib) The proof is the same as that of Theorem 1 (ii).

So for every policy $\pi \in \Pi'$ that is not characterized by properties (ia) and (ib), we can construct a policy $\pi' \in \Pi'$ that has properties (ia) and (ib) and there exists a time T such that $J^{\pi'}_t < J^{\pi}_t$, $t \geq T$.

(ii) Assume now that the policy $\pi \in \Pi'$, initially considered in part (i) of the proof, is replaced by the policy $\pi' \in \Pi'$ having properties (ia) and (ib). Suppose that π' follows the priority rule $(S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_j, S_i, S_{\ell_{k+1}},$

$\ldots, S_{\ell_{M-2}}$) with $j > i$, and let τ_0 be the first time π' processes a task using server S_j instead of S_i (when S_i is also available). We construct policy π^1 as follows. π^1 is identical to π' until τ_0, uses S_i instead of S_j at time τ_0, and follows the same priority rule as π' afterward.

Let σ_i and σ_j be the first service completion times after τ_0 at servers S_i and S_j, respectively, and τ be the first arrival time after τ_0 for which all servers $S_{\ell_1}, \ldots, S_{\ell_k}$ are occupied. Then

$$E(J_t^{\pi'} - J_t^{\pi^1} \mid \tau < \sigma_i, \sigma_j) = 0, \qquad t \geq 0,$$

because of the memoryless property of the exponential distribution, and

$$E(J_t^{\pi'} - J_t^{\pi^1} \mid \tau \geq \sigma_i, \sigma_j) = 0, \qquad t \geq 0.$$

Therefore, for $t \geq 0$ we have

$$\begin{aligned} J_t^{\pi'} - J_t^{\pi^1} = &\int_{\tau_0}^{\infty} E(J_t^{\pi'} - J_t^{\pi^1} \mid \sigma_i \leq \tau < \sigma_j, \tau = \tau') \\ &\times P(\sigma_i \leq \tau') P(\sigma_j > \tau')\, dF(\tau') \\ &+ \int_{\tau_0}^{\infty} E(J_t^{\pi'} - J_t^{\pi^1} \mid \sigma_j \leq \tau < \sigma_i, \tau = \tau') \\ &\times P(\sigma_j \leq \tau') P(\sigma_i > \tau')\, dF(\tau'), \end{aligned}$$

where F is the distribution function of τ. In the preceding we have used the fact that τ is independent of σ_i, σ_j, because it only depends on the service completions at servers $S_{\ell_1}, \ldots, S_{\ell_k}$ and the arrival process. Because of the construction of policies π' and π^1 we have

$$E(J_t^{\pi'} - J_t^{\pi^1} \mid \sigma_i \leq \tau < \sigma_j, \tau = \tau') = E(J_t^{\pi^1} - J_t^{\pi'} \mid \sigma_j \leq \tau < \sigma_i, \tau = \tau').$$

So the cost difference of π' and π^1 can be written as

$$\begin{aligned} J_t^{\pi'} - J_t^{\pi^1} = &\int_{\tau_0}^{\infty} E(J_t^{\pi'} - J_t^{\pi^1} \mid \sigma_i \leq \tau < \sigma_j, \tau = \tau') \\ &\times [P(\sigma_i \leq \tau') P(\sigma_j > \tau') - P(\sigma_j \leq \tau') P(\sigma_i > \tau')]\, dF(\tau'). \end{aligned}$$

The term in brackets is positive because $\mu_i > \mu_j$. Therefore, if we prove that $E(J_t^{\pi'} - J_t^{\pi^1} \mid \sigma_i \leq \tau < \sigma_j, \tau = \tau')$ is nonnegative for $t \geq 0$, $J_t^{\pi'} - J_t^{\pi^1}$ will be nonnegative for $t \geq 0$.

The task that arrives at time τ' is processed by S_i under π' and by S_j under π^1. Therefore, at time τ' servers $S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_j, S_i$ are busy under π', while servers $S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_j$, are busy under π^1. Let τ'' be the first arrival time after τ' and σ_i' and σ_j' the first service completion times after τ' at servers S_i and S_j, respectively. Note that $\sigma_j' = \sigma_j$ because of the exponentiality of the server. If $\tau'' \geq \sigma_i', \sigma_j'$ or $\sigma_i' \leq \tau'' < \sigma_j'$, policies π' and π^1 are coupled at time τ''. If $\sigma_j' \leq \tau'' < \sigma_i'$,

at time τ'' some servers among $S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}$ are busy under both π' and π^1 and, moreover, server S_i is busy under π', but not under π^1. Finally, if $\tau'' < \sigma_i', \sigma_j'$, then along some servers being busy under both π' and π^1, S_i is busy under π' but not under π^1, or, if no service has occurred before τ'', $S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_j, S_i, S_{\ell_{k+1}}$ are busy under π' and $S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_j, S_i$ are busy under π^1. Repeating the argument we see that for any sample path we either have policies π' and π^1 coupled at some time instant or one more server is occupied under π'. Thus, there is no way a task can be processed under π' and lost under π^1, although it is possible that a task will be processed under π^1 but lost under π'. If the latter occurs, then at the time this task is lost all servers are full and the same tasks (if any) are waiting to be served under both π' and π^1, so the policies are coupled from that point on. Therefore, $J_t^{\pi^1} \leq J_t^{\pi'}$, $t \geq 0$.

We can now construct a modification π^2 of π^1 in the same way as π^1 modifies π', i.e., the first time π^1 prefers S_j from S_i, π^2 uses S_i instead of S_j and follows the same priority rule as π^1 afterward. Therefore, $J_t^{\pi^2} \leq J_t^{\pi^1}$, $t \geq 0$. Continuing the construction of such modified policies we conclude that $J_t^{\tilde{\pi}} \leq J_t^{\pi'}$, $t \geq 0$, where $\tilde{\pi}$ follows the priority rule $(S_{\ell_1}, S_{\ell_2}, \ldots, S_{\ell_k}, S_i, S_j, S_{\ell_{k+1}}, \ldots, S_{\ell_{M-2}})$.

If we keep improving policy $\tilde{\pi}$ by interchanging the order in each pair of consecutive servers for which the slowest server has the highest priority, we will eventually get policy π^* that follows the priority rule $(S_1, S_2, \ldots, S_M)$, has properties (ia) and (ib), and $J_t^{\pi^*} \leq J_t^{\pi'}$, $t \geq 0$. But because $J_t^{\pi'} < J_t^{\pi}$, $t \geq T$, we get $J_t^{\pi^*} < J_t^{\pi}$, $t \geq T$, and the proof is complete. ■

Theorems 1 and 2 state that in both the single-server and multiserver problems the optimal policy selects the tasks to be processed according to a threshold rule. Note that this threshold rule depends not only on the set of eligible tasks, but also on the number of servers as well. We illustrate this point by the following example. We have two classes of tasks with $E^1(0) = \{d\}$, $E^2(0) = \{d_1, d_2\}$, where $d_1 < d_2 < d$. The penalties for missed deadlines are C_1 and C_2 for classes 1 and 2, respectively, where $C_1 > C_2$. We assume

$$\frac{1}{2\mu} \log \frac{C_1}{C_2} < d - d_2 < \frac{1}{\mu} \log \frac{C_1}{C_2}. \tag{8}$$

We have no arrivals ($\lambda = 0$). Let π_1 and π_2 be the policies that start with d and d_1, respectively, and proceed optimally afterward.

Case 1: 1 Server: According to Theorem 1, after the first service completion policy π_1 gives priority to task d_1. The expected cost due to policy π_1 is

$$J^{\pi_1} = C_2 \mu d_1 e^{-\mu d_2} + C_2 e^{-\mu d_1} + C_2 e^{-\mu d_2}.$$

After the first service completion policy π_2 gives priority to task d. This is a direct consequence of the second inequality in Eq. (8). The expected cost due to policy π_2 is

$$J^{\pi_2} = C_2 e^{-\mu d_2} + C_2 \mu d_2 e^{-\mu d_2} + C_1 e^{-\mu d}.$$

The cost difference of policies π_1 and π_2 is

$$J^{\pi_1} - J^{\pi_2} = C_2 \mu e^{-\mu d_2}(d_1 - d_2) + C_2 e^{-\mu d_1} - C_1 e^{-\mu d}.$$

As $d_1 \to d_2$, $J^{\pi_1} - J^{\pi_2} \to C_2 e^{-\mu d_2} - C_1 e^{-\mu d} < 0$, because of the second inequality in Eq. (8). Therefore when d_1 is sufficiently close to d_2, π_1 does better than π_2.

Case 2: 2 Servers: According to Theorem 2, after policy π_1 assigns task d to server 1, it is optimal to assign task d_1 to server 2. The expected cost due to policy π_1 is

$$J^{\pi_1} = C_2 e^{-\mu d_2}.$$

After policy π_2 assigns task d_1 to server 1, it is optimal to assign task d_2 to server 2. This is a direct consequence of the first inequality in Eq. (8). The expected cost due to policy π_2 is

$$J^{\pi_2} = C_1 e^{-2\mu d}.$$

The first inequality in Eq. (8) implies that $J^{\pi_2} < J^{\pi_1}$, so π_2 does better than π_1. We see from this example that the threshold is different in the two cases, since when we have one server policy π_1 does better than π_2 for some $d_1 < d_2$, while in the case of two servers policy π_2 is better than π_1 for all $d_1 < d_2$.

3. STOCHASTIC DEADLINES

3.1. The Single-Server Problem

Tasks arrive according to a Poisson process of rate λ and are served by a single exponential server of rate μ. The arrival and service processes are independent and $\lambda < \mu$. Each arriving task has a deadline associated with the beginning of its service. Contrary to the problem examined in Section 2.1 the deadlines of arriving tasks are not known to the scheduler. Let d_i be the ith task's deadline. We assume that d_i, $i = 1,2,\ldots$, form a sequence of i.i.d. random variables that are independent of the arrival and service processes and are finite a.s. We also assume that the common distribution of $\{d_i\}$ has increasing likelihood ratio.[4] The interpretation of this assumption is that a task that has arrived earlier is more likely to have an earlier extinction time than a task that has arrived later. Tasks belong to N different classes, with tasks of class i incurring a cost C_i when they do not meet their deadlines, where $C_1 > C_2 > \cdots > C_N$ and $C_N > (\lambda/\lambda + \mu)C_1$. Our objective is to find a policy that is optimal within the set of nonidling and nonpreemptive policies according to optimality criterion (1).

For a decision instant t_0 we denote by $M^i(t_0)$ the set of eligible tasks of type i. If $M^i(t_0) \neq \emptyset$, we define $A^i(t_0) = \{a_1^i, \ldots, a_{n_i}^i\}$, $n_i \geq 1$, to be the set of arrival times of eligible tasks of type i at time t_0 arranged in increasing order. With a_j^i denoting both the time instant a_j^i and the task with arrival time a_j^i, the properties of the optimal policy are given by the following theorem.

THEOREM 3: *Consider a decision instant t_0. Then*

(i) *Within a class of tasks it is optimal to process the one with the earliest arrival time (i.e., the policy* FCFS *is optimal).*

(ii) *For each $i < j$ with $M^i(t_0) \neq \emptyset$, $M^j(t_0) \neq \emptyset$ there exists a time instant (threshold) $t_{ij} \geq a_1^j$ such that it is optimal to process task a_1^i instead of a_1^j if $a_1^i \leq t_{ij}$, and vice versa otherwise.*

Discussion: By parts (i) and (ii) of the theorem we conclude that the search for the optimal task is confined to the set $I'(t_0) = \{j \mid a_1^i > a_1^j, \forall i < j\}$. With L being the cardinality of $I'(t_0)$, to determine the optimal task we need to make $L - 1$ pairwise comparisons, i.e., compute $L - 1$ thresholds. The way to compute these thresholds will not be discussed in this paper.

PROOF:

(i) For a class i with at least two eligible tasks at time t_0, let π and $\tilde{\pi}$ be the policies that at time t_0 process tasks a_m^i and a_1^i, respectively, and afterward proceed optimally. With e_m^i, e_1^i denoting the extinction times of a_m^i, a_1^i and f_m^i, f_1^i their respective probability densities, we get

$$J_t^\pi - J_t^{\tilde{\pi}} = \int_{k=t_0}^{\infty} \int_{\ell=k}^{\infty} E\left(J_t^\pi - J_t^{\tilde{\pi}} \mid e_1^i = k, e_m^i = \ell\right) f_1^i(k) f_m^i(\ell)\, dk\, d\ell$$
$$+ \int_{k=t_0}^{\infty} \int_{\ell=k}^{\infty} E\left(J_t^\pi - J_t^{\tilde{\pi}} \mid e_m^i = k, e_1^i = \ell\right) f_m^i(k) f_1^i(\ell)\, dk\, d\ell.$$

Because of the construction of policies π and $\tilde{\pi}$, we have

$$E\left(J_t^\pi - J_t^{\tilde{\pi}} \mid e_m^i = k, e_1^i = \ell\right) = E\left(J_t^{\tilde{\pi}} - J_t^\pi \mid e_m^i = \ell, e_1^i = k\right)$$
$$= -E\left(J_t^\pi - J_t^{\tilde{\pi}} \mid e_1^i = k, e_m^i = \ell\right).$$

So the cost difference of π and $\tilde{\pi}$ can be written as

$$J_t^\pi - J_t^{\tilde{\pi}} = \int_{k=t_0}^{\infty} \int_{\ell=k}^{\infty} E\left(J_t^\pi - J_t^{\tilde{\pi}} \mid e_1^i = k, e_m^i = \ell\right)$$
$$\times \left(f_1^i(k) f_m^i(\ell) - f_m^i(k) f_1^i(\ell)\right) dk\, d\ell. \tag{9}$$

The assumption that the distribution of the deadlines has increasing likelihood ratio implies that $e_1^i \underset{LR}{\leq} e_m^i$. Then, because $k \leq \ell$, we have

$$\frac{f_m^i(k)}{f_1^i(k)} \leq \frac{f_m^i(\ell)}{f_1^i(\ell)} \Rightarrow f_1^i(k) f_m^i(\ell) - f_m^i(k) f_1^i(\ell) \geq 0. \tag{10}$$

Construct now a policy π' as follows: π' processes a_1^i at time t_0 and is identical to π when π processes tasks other than a_1^i. When π processes (if ever) task a_1^i, π' processes task a_m^i, if it is still eligible, and the task with the earliest arrival time, if task a_m^i is not eligible. Then for $k \le \ell$ it can be shown that

$$E(J_t^{\pi} - J_t^{\pi'} \mid e_1^i = k, e_m^i = \ell, k \le \ell) > 0, \qquad t \ge \ell \tag{11}$$

(the proof of Eq. (11) is identical to that of part (i) of Theorem 1). Because both policies π' and $\tilde{\pi}$ process task a_1^i at time t_0 and $\tilde{\pi}$ acts optimally afterward, while π' does not necessarily do so, there exists a time $T_{k\ell}$ such that

$$E(J_t^{\pi'} - J_t^{\tilde{\pi}} \mid e_1^i = k, e_m^i = \ell, k \le \ell) \ge 0, \qquad t \ge T_{k\ell}. \tag{12}$$

From Eqs. (9)–(12) and the assumption that the deadlines are finite a.s. we conclude that there exists a time T such that $J_t^{\pi} > J_t^{\tilde{\pi}}$, $t \ge T$; hence, among tasks of type i it is optimal to process the one with the earliest arrival time.

(ii) Let a^i and a^j be tasks of type i and j $(i < j)$, respectively, that are eligible at time t_0. Let the arrival times of all eligible tasks except a^i be fixed. Then part (ii) of the theorem follows from the next two lemmas.

LEMMA 3: *If $a^i \le a^j$ it is optimal to serve a^i instead of a^j.*

PROOF: The proof is similar to that of part (i) and is omitted. ■

LEMMA 4: *Suppose that for $a^i = k > a^j$ it is optimal to serve a^j instead of a^i. Then for $a^i = \ell$, where $\ell > k$, it is still optimal to serve a^j.*

PROOF: As in Lemma 2 we will attach to each policy a subscript denoting the value of a^i. For a policy π_ℓ that processes task a^i at time t_0, let π_k' be the policy that processes a^i at time t_0 and is identical to π_ℓ afterward. Then $J_t^{\pi_\ell} = J_t^{\pi_k'}$, $t \ge t_0$. By assumption we know that there exists a policy $\tilde{\pi}_k'$ that processes task a^j at time t_0 and is better than π_k'; i.e., there exists a time T_1 such that $J_t^{\pi_k'} > J_t^{\tilde{\pi}_k'}$, $t \ge T_1$. Construct now policy $\tilde{\pi}_\ell$ as follows. $\tilde{\pi}_\ell$ is identical to $\tilde{\pi}_k'$, except that it processes $a^i = \ell$ when (if ever) $\tilde{\pi}_k'$ processes $a^i = k$. The construction of such a policy is possible because, if a^i is served under $\tilde{\pi}_k'$, it is also eligible under $\tilde{\pi}_\ell$, since, if d is its deadline, its extinction time under $\tilde{\pi}_k'$ is $k + d$, which is less than $\ell + d$, its extinction time under $\tilde{\pi}_\ell$. It can be shown (see proof of Lemma 2) that $E(J_t^{\tilde{\pi}_k'} - J_t^{\tilde{\pi}_\ell} \mid d = d') > 0$, $t \ge \ell + d'$. Since d is finite a.s. the preceding relation implies that there exists a time T_2 such that $J_t^{\tilde{\pi}_k'} - J_t^{\tilde{\pi}_\ell} > 0$, $t \ge T_2$. But $J_t^{\tilde{\pi}_k'} < J_t^{\pi_k'} = J_t^{\pi_\ell}$, $t \ge T_1$. Therefore, $J_t^{\pi_\ell} > J_t^{\tilde{\pi}_\ell}$, $t \ge \max\{T_1, T_2\}$, so it is optimal to serve a^j when $a^i = \ell$. ■

3.2. The Multiserver Problem

In this section we turn to the problem of scheduling tasks in a queueing system consisting of M parallel exponential servers $S_1, S_2, \ldots, S_M$, with server S_i hav-

ing rate μ_i, where $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_M$. The interarrival times are also exponential with parameter λ, where $\lambda < \sum_{i=1}^{M} \mu_i$. Each arriving task has a deadline with respect to the beginning of its service. The deadline is not known to the scheduler, but instead its probability distribution is available. We assume that the deadlines form an i.i.d. sequence of random variables that are independent of the arrival and service processes, are finite a.s., and have increasing likelihood ratio. Tasks belong to one of N classes, with tasks of class i incurring a penalty C_i when they miss their deadlines, where $C_1 > C_2 > \cdots > C_N$ and $C_N > (\lambda/\lambda + \mu_M)C_1$. Our objective is to find a nonidling and nonpreemptive list policy that is optimal in the sense of criterion (1).

We again have to consider decision instants where the number of eligible tasks is greater than the number of empty servers $S_{i_1}, S_{i_2}, \ldots, S_{i_{M'}}$, $M' \leq M$, and decision instants where the number of eligible tasks is less than the number of empty servers. The optimal policy is characterized by the properties given in the following theorem.

THEOREM 4:

(i) *Let t_0 be a decision instant such that the number of eligible tasks is greater than M', the number of available servers. Consider empty server S_{i_ℓ}, $1 \leq \ell \leq M'$. Then*

(ia) *Within a class of tasks it is optimal to assign to S_{i_ℓ} the one with the earliest arrival time.*

(ib) *For each $i < j$ with $M^j(t_0) \neq \emptyset$, $M^i(t_0) \neq \emptyset$ there exists a time instant (threshold) $t_{ij} \geq a_1^j$ such that it is optimal to assign to S_{i_ℓ} task a_1^i instead of a_1^j if $a_1^i \leq t_{ij}$, and vice versa otherwise.*

(ii) *For decision instants for which the number of eligible tasks is less than the number of available servers the optimal policy ranks the servers in decreasing order of their service rates, i.e., uses the fastest available servers.*

Discussion: When the number of eligible tasks is greater than the number M' of empty servers, we select the M' tasks to be processed one by one by following the procedure for the single-server problem M' times. First we compute the appropriate thresholds to determine the "best" task, then we repeat for the "best" among the remaining tasks, and so on.

PROOF: The proof of parts (ia) and (ib) is the same as that of Theorem 3 and is omitted. For decision instants for which the number of eligible tasks is less than the number of available servers the problem is not different from the one where the deadlines are known to the scheduler, because the extinction times of the waiting tasks do not affect the cost of a given policy, since the deadlines are with respect to the beginning of the service and all the tasks are served. So for the proof we refer the reader to the proof of Theorem 2. ■

4. COUNTEREXAMPLES TO EXTENSIONS

In this section we briefly discuss two possible extensions of the problems presented in Section 2: (i) a problem that includes a routing decision in addition to scheduling and (ii) a scheduling problem with switching cost. We find out that these problems lack a nice structure, the first one because of the highly nonlinear cost function, and the second one because of the inclusion of the switching cost.

We discuss each problem separately starting with the problem that includes a routing decision in addition to scheduling. Arriving tasks have to be routed to one of two similar exponential servers. Each task belongs to one of N different classes and has a deadline that becomes known at the arrival instant. Let d_i be the ith task's deadline. We assume that d_i, $i = 1,2,\ldots$, form a sequence of independent random variables that are also independent of the arrival and service processes. No distribution on the deadlines of future arrivals is assumed. If the service of a task i does not begin by its extinction time, the task is lost and incurs a cost C_i, where $C_1 > C_2 > \cdots > C_N$. The objective is to find a routing strategy γ^* as well as a nonidling, nonpreemptive policy $\pi^* \in \Pi'$ such that for any other routing strategy γ and any other policy $\pi \in \Pi'$

$$\liminf_{t\to\infty}(J_t^{\gamma,\pi} - J_t^{\gamma^*,\pi^*}) > 0, \tag{13}$$

where $J_t^{\gamma,\pi}$ is the expected cost under policy (γ,π) until time t, i.e.,

$$J_t^{\gamma,\pi} = \sum_{i=1}^{N} C_i E(X_{i,t}^{\gamma,\pi}),$$

$X_{i,t}^{\gamma,\pi}$ being the number of type i tasks lost under policy (γ,π) until time t. After a routing strategy γ is fixed this problem reduces (for each server) to that of Section 2. Thus, we restrict attention to the routing problem. For the routing problem it is well known (see Ephremides, Varaiya, and Walrand [9]) that when the tasks incur a fixed holding cost per unit time during their waiting time, the optimal routing strategy sends tasks to the server with the shortest queue length. This result no longer holds when the tasks have a deadline with respect to the beginning of their service. The optimal decision does not depend on the queue lengths at the two servers, but on the deadlines of the tasks present in the system. We illustrate this by the following example. We have deterministic deadlines, one class of tasks incurring a cost of one unit when they miss their deadlines, the two servers have rate $\mu = 1$, and there are no arrivals ($\lambda = 0$). The sets of tasks waiting in servers 1 and 2 at time 0 are $E_1(0) = \{d_1\}$ and $E_2(0) = \{d_2, d_3\}$, respectively, where $d_2 < d_3$. A task with extinction time d_4, where $d_4 < d_1, d_2$, is to be assigned to one of the two servers. Let $J_{ik}, i,k = 1,2$, be the expected cost incurred at server k by the policy that assigns task d_4 to server i and then, according to Theorem 1, serves the tasks in increasing order of their extinction times. Then

$$J_{11} = e^{-d_1}, \qquad J_{12} = e^{-d_3}, \qquad J_{21} = 0,$$
$$J_{22} = 2e^{-d_3} + e^{-d_2} - e^{-d_3} + d_2 e^{-d_3}$$

and the cost difference between the policy that routes d_4 to server 1 and the policy that prefers server 2 is

$$J_{11} + J_{12} - (J_{21} + J_{22}) = e^{-d_1} - e^{-d_2} - d_2 e^{-d_3}.$$

Routing to server 1 is optimal when $d_1 > d_2$, but for $d_1 < d_2$ and d_3 large enough it is optimal to route to server 2, the server with the largest queue length.

Next we consider the problem of Section 2.1 and, in addition, we assume that a cost K is incurred whenever the server switches from one class of tasks to another. The existence of such a switching cost usually complicates even problems with linear costs, so we would expect the same to happen to our problem. Indeed, the following example shows that Theorem 1 is not valid when a switching cost is included. We have two classes of tasks with $E^1(0) = \{d_1\}$, $E^2(0) = \{d_2\}$, where $d_1 < d_2$. The penalties for missed deadlines are C_1 and C_2 for classes 1 and 2, respectively, where $C_1 > C_2$. We have no arrivals ($\lambda = 0$). Then, according to Theorem 1, when no switching cost is present the optimal policy processes task d_1 first. Let π_1 and π_2 be the policies that start with d_1 and d_2, respectively. The expected costs incurred by these policies when the switching cost K is included are

$$J^{\pi_1} = K(1 - e^{-\mu d_2}) + C_2 e^{-\mu d_2}, \qquad J^{\pi_2} = K(1 - e^{-\mu d_1}) + C_1 e^{-\mu d_1},$$

so the cost difference of policies π_1 and π_2 is

$$J^{\pi_1} - J^{\pi_2} = K(e^{-\mu d_1} - e^{-\mu d_2}) + C_2 e^{-\mu d_2} - C_1 e^{-\mu d_1}.$$

With K sufficiently large this difference becomes positive, so it is optimal to start with d_2, a result that does not agree with Theorem 1.

5. CONCLUSIONS

We have considered the problem of optimally scheduling tasks with priorities and deadlines (deterministic and stochastic) in both a single-server and a multi-server queueing system. Using interchange arguments we showed that threshold policies are optimal. The computation of the thresholds describing the optimal policies is difficult, because the thresholds depend on the deadlines of all the tasks present in the system. Finally, simple counterexamples were given to show that the properties of the optimal policy are not preserved when a switching penalty is included and that optimal routing strategies under the assumption of linear costs are no longer optimal when strict deadlines are considered.

Notes

1. We define the end of the busy period to be the first time when the server becomes idle.
2. The problem of the existence of an optimal policy will not be discussed in this work. We assume that an optimal policy exists.

3. We define the end of the busy period to be the first time when at least one server becomes idle.

4. Let X and Y be nonnegative random variables with densities f and g, respectively. We say that X is larger than Y in likelihood ratio and write $X \underset{LR}{\geq} Y$ if the ratio of their respective densities $f(x)/g(x)$ is nondecreasing in x. For a nonnegative random variable X let X_t denote its residual life after t units. We say that X has increasing likelihood ratio if X_t decreases in likelihood ratio as t increases.

References

1. Baccelli, F., Boyer, P., & Hebuterne, G. (1984). Single server queues with impatient customers. *Advances in Applied Probability* 16: 887–905.
2. Baccelli, F. & Trivedi, K.S. (1985). A single server queue in a hard real time environment. *Operations Research Letters* 4(4): 161–168.
3. Bhattacharya, P.P. & Ephremides, A. (1989). Optimal scheduling with strict deadlines. *IEEE Transactions on Automatic Control* 34: 721–728.
4. Bhattacharya, P.P. & Ephremides, A. (1991). Optimal allocation of a server between two queues with due times. *IEEE Transactions on Automatic Control* 36: 1417–1423.
5. Charlot, F. & Pujolle, G. (1978). Recurrence in single server queues with impatient customers. *Annales de l'Institut Henri Poincaré Sec. B* XIV: 399–410.
6. Cho, Y. & Sahni, S. (1981). Preemptive scheduling of independent jobs with release and due dates on open, flow and job shops. *Operations Research* 29: 511–522.
7. Conway, R.W., Maxwell, W.L., & Miller, L.W. (1967). *Theory of scheduling.* Reading, MA: Addison-Wesley.
8. Derman, C., Lieberman, G.J., & Ross, S.M. (1978). A renewal decision problem. *Management Science* 24: 554–563.
9. Ephremides, A., Varaiya, P., & Walrand, J.C. (1980). A simple dynamic routing problem. *IEEE Transactions on Automatic Control* AC-25: 690–693.
10. Huang, C.-C. & Weiss, G. (1992). Scheduling jobs with stochastic processing times and due dates to minimize total tardiness. Preprint.
11. Panwar, S.S., Towsley, D., & Wolff, J.K. (1988). Optimal scheduling policies for a class of queues with customer deadlines to the beginning of service. *Journal of Association for Computing Machinery* 35(4): 832–844.
12. Pinedo, M. (1983). Stochastic scheduling with release dates and due dates. *Operations Research* 31: 559–572.
13. Ross, S. (1983). *Stochastic processes.* New York: Wiley.
14. Stanford, R.E. (1979). Reneging phenomena in single channel queues. *Mathematics of Operations Research* 4: 162–178.
15. Takacs, L. (1974). A single server queue with limited virtual waiting time. *Journal of Applied Probability* 11: 612–617.