On the Optimality of Minimum Laxity and Earliest Deadline Scheduling for Real-Time Multiprocessors

D. Towsley

Dept. of Comp. & Inf. Science
Univ. Massachusetts
Amherst, MA 01003

S.S. Panwar

Dept. of Electr. Engineering
Polytechnic University
Brooklyn, NY 11201

ABSTRACT

In this paper we consider the problem of scheduling jobs having real-time constraints on a multiprocessor. If a job does not complete within a certain time interval of its arrival to such a system, it is considered useless and need not be served. It is therefore desirable to schedule jobs so that the fraction of jobs served within their respective deadlines is maximized. We model such a system as a multiple server queue. We show for a variety of systems that the minimum laxity policy (ML) maximizes the fraction of jobs that successfully complete service when jobs must begin service by their deadline and that the earliest deadline policy (ED) does the same for a wide class of systems when jobs must complete service by their deadlines. Results are given for unreliable multiprocessors and multiprocessors that serve several priority classes of jobs.

INTRODUCTION

Increasing interest has been shown recently in the design and analysis of real-time multiprocessor systems. The workloads served by these systems consist of customers that have real-time constraints, i.e., customers must complete or enter service by specified deadlines. For some systems it is unacceptable for any task to miss its deadline. In these systems task service demands are usually well understood and a substantial literature has focussed on the development and evaluation of scheduling policies for these workloads, [14,15]. Other workloads consist of tasks for which it is not critical that all tasks meet their constraints. Usually, the service requirements and the arrival patterns are not as well understood and the objective is to design policies that will minimize the fraction of tasks that miss their deadlines. The purpose of this paper is to study optimal policies for this second class of workloads.

In this paper we consider as our model for a multiprocessor, a multiple server queue with either finite or infinite capacity, that serves customers with deadlines. We study the effect that different service and buffer overflow policies on the fraction of customers which successfully complete service, i.e., do not miss their deadlines. We show that, out of the class of service-time independent policies, the *minimum laxity* policy (ED) maximizes the fraction of customers that successfully complete service by their deadlines for the preemptive continuous/discrete time G/M/c/K queue when customer deadlines are to the end of service. Here ED is the non-idling policy that schedules the customer closest to his deadline. This result is extended to systems where servers take vacations.

When we restrict ourselves to systems that do not allow preemptions, we show that out of the class of non-idling service time independent policies, ED maximizes the fraction of customers making their deadlines for the continuous/discrete time G/M/c/K queue when deadlines are until the beginning of service and the *minimum laxity* policy (ML) does the same when deadlines are until the end of service.

The ML and ED policies are very similar to the earliest due date (EDD) scheduling policy proposed by Jackson [12]. Consider a set of n tasks $\{T_i, 1 \leq i \leq n\}$ with the corresponding n due dates $\{d_i, 1 \leq i \leq n\}$. Let the finishing times under schedule S be $f_i(S)$. Then the lateness of T is defined as $f_i(S) - d_i$ and the tardiness is defined as $max\{0, f_i(S) - d_i\}$. Jackson showed that the maximum lateness and maximum tardiness are minimized by sequencing the tasks in the order of non-decreasing due dates. As we shall see in Section 3, ML and ED scheduling differs from EDD scheduling in that they never schedules tasks which are already past their due dates. Note that the tasks and their due dates are known a priori under Jackson's model. Similar problems, for models other than queueing systems, have also been studied in [8,16,19,20,23]. In the packet-switching context, variations of the EDD policy for queueing models have been studied in [13,4]. Numerous works consider optimal service disciplines for queues with delay dependent customer behavior [9,24]. In queueing theory literature, queues with impatient customers have been usually analyzed assuming a FCFS scheduling policy [1,6,10]. An analysis of ED for the preemptive M/M/1 queue (deadlines to the end of service) and ML for the nonpreemptive M/M/c queue (deadlines to the beginning of service) can be found in [11].

0-8186-2076-5/90/0000/0017$01.00 © 1990 IEEE

In [18], we have considered the problem of a single server queue with impatient customers under the assumption that deadlines are until customers enter service. We showed that STE (another name for ML and ED) is optimal for a large class of infinite capacity single server queues. The shortest time to extinction with unforced idle times (STEI) class of policies are shown to be optimal for a larger class of queues. Similar results for the continuous time single server infinite capacity queue when the deadlines are to the end of service can be found in [17,2]. Our results generalize these previous results in several ways. First, our results are for multiple server queues, and in some cases, servers are allowed to take vacations. Last, the results in [17,2] are based on interchange arguments which obscure their physical interpretation. The proofs in this paper are based on defining a *state* of the system based on the set of extinction times of the customers in the system and using a forward induction argument to establish dominance of one state over another when operating under different policies. This method provides a clearer understanding of the differences between ML, ED, and other policies. This approach has also been used to develop bounds on the performance of these policies, (see [11]).

This paper is organized as follows. Section 2 contains a model of the system under study along with definitions of the different scheduling policies of interest to us. The main results of the paper are contained in sections 3, 4, and 5. Section 3 contains the results for systems with deadlines to the end of service that allow preemptions, section 4 contains results for systems for systems with deadlines to the beginning of service, and section 5 contains results for systems with deadlines to the end of service without preemptions. Section 6 discusses some extensions of these optimality results to systems with finite buffers and with deadlines that are not known to the scheduler. We summarize our results in Section 7.

DEFINITIONS AND NOTATION

We consider three different multiple server queues, all of infinite capacity.

- Preemptive queues with deadlines to the end of service.
- Nonpreemptive queues with deadlines to the end of service where a customer that misses its deadline while in service is aborted,
- Nonpreemptive queues with deadlines to the beginning of service,

In all of these systems let T_i denote the arrival time of the i-th customer and A_i denote the time between the arrivals of the $(i-1)$-th and i-th customers. We assume that A_i is a random variable with arbitrary distribution. Let E_i denote the extinction time of the i-th customer (i.e., the time by which it must be served). Here $E_i = T_i + D_i$ where D_i is a random variable with a general distribution. We shall refer to D_i as the real time constraint or the relative deadline for customer i. Last, let $\{B_i\}_{1\leq i}$ be an independent and identically distributed (i.i.d.) sequence of random variables with a general distribution which will be used to assign service times to customers.

We shall use the notation $\boldsymbol{A}_N = \{A_i\}_{1\leq i\leq N}$, $\boldsymbol{D}_N = \{D_i\}_{1\leq i\leq N}$, $\boldsymbol{B}_N = \{B_i\}_{1\leq i\leq N}$, and $\boldsymbol{S}_N = (\boldsymbol{A}_N, \boldsymbol{D}_N, \boldsymbol{B}_N)$, $1 \leq N$. In addition, whenever we focus on a specific sample realization of the above r.v.'s, we shall use lowercase notation (i.e., a_i for A_i, etc ...). Furthermore, we shall let $\boldsymbol{a} = \{a_i\}_{1\leq i}$, $\boldsymbol{b} = \{b_i\}_{1\leq i}$, $\boldsymbol{d} = \{d_i\}_{1\leq i}$, $\boldsymbol{a}_N = \{a_i\}_{1\leq i\leq N}$, $\boldsymbol{b}_N = \{b_i\}_{1\leq i\leq N}$, and $\boldsymbol{d}_N = \{d_i\}_{1\leq i\leq N}$. Last, let $\boldsymbol{s} = (\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{b})$ and $\boldsymbol{s}_N = (\boldsymbol{a}_N, \boldsymbol{d}_N, \boldsymbol{b}_N)$, $N = 1, \ldots$. These last two quantities will be referred to as an input sample and finite input sample respectively.

At this point in the paper we will not specify how service times from the sequence $\{B_i\}$ are assigned to customers. The assignment rule will depend on which system we are interested in and on what property we wish to prove with regard to that system. We use the notation A/C/D/E+ F to denote a queue with customer deadlines where A, C, D, and E have the same meaning as in Kendall's notation while F describes the distribution of the relative deadlines. Last, we make the assumption that $\{B_i | 1 \leq i\}$ is independent of $\{A_i\}$ and $\{D_i\}$.

Let π be a policy that determines what customer in the queue is to be executed (if any) whenever the server is free. This policy makes its decision based on the customers that are eligible for service as well as on the past history of the system. We wish to choose π so that we maximize the fraction of customers beginning service before their respective extinction times. Consider a system in which exactly N customers arrive for service. We define $V_N(\pi)$ to be the number of customers served for this system. We are interested in the fraction, $\overline{V}_N(\pi) = E[V_N(\pi)]/N$, of customers served in this system. We define the fraction of customers served in the system as $N \longrightarrow \infty$ (under policy π) to be

$$\overline{V}(\pi) = \liminf_{N\to\infty} \overline{V}_N(\pi).$$

Finally, let $\overline{V} = \sup_\pi \overline{V}(\pi)$.

Let $\mathcal{C}_\pi(t) = \{c_{j_1}, c_{j_2}, \cdots, c_{j_n}\}$ denote the set of customers in the queue at time t and $\mathcal{R}_\pi(t) = \{c_{j_1}, c_{j_2}, \cdots, c_{j_n}, \cdots, c_{j_m}\}$ denote the set of all customers in the system at time $t, j_i \geq 1$, $1 \leq i \leq m$. Here c_i denotes the i-th customer to arrive to the system. We denote the sets of extinction times associated with these two sets of customers are denoted by $\boldsymbol{E}_\pi(t)$ and $\boldsymbol{R}_\pi(t)$.

Consider the actions that policy π can take at time t. If all the servers are busy, then π takes no action if preemptions are not allowed. If any server is idle at time t or if preemptions are allowed, then π can either schedule no customer or schedule customers from $\mathcal{C}_\pi(t)$. Policy π is allowed to choose one of these actions according to some distribution that depends on $\pi, \mathcal{C}_\pi(t)$ and the previous history $\boldsymbol{H}_t$ (to be defined later in this section). We define $p_j(\pi, t, \boldsymbol{H}_t)$ to be the probability that π schedules customer $c_j \in \mathcal{C}_\pi(t), j = 1, 2, \cdots$ on an idle server and $p_0(\pi, t, H_t)$ to be the probability that π chooses to schedule no customer.

If π chooses not to schedule a customer at time t and $\mathcal{C}_\pi(t) \neq 0$, then it delays making a new scheduling decision by a random amount of time τ with some arbitrary distribution function $F_\tau(x|H_t)$ (τ takes on discrete values in the case of a discrete time queue). The policy does not perform a scheduling decision until either τ time units elapse or an arrival occurs. Without loss of generality, we may impose one last constraint on π, namely, π is prohibited from scheduling two successive idle times on the same server when the queue is nonempty unless they are separated by the arrival of one or more customers.

In the case that π is allowed to preempt customers, we introduce some additional parameters. If π decides to schedule a customer at time t, then $q(\pi, t, \boldsymbol{H}_t)$ is the probability that the customer will not be preempted in the absence of customer arrivals and service completions. The customer is scheduled for preemption with probability $1 - q(\pi, t, \boldsymbol{H}_t)$ and is provided with τ units of service where τ has cumulative distribution function $H_\tau(x|\boldsymbol{H}_t)$. The customer is preempted after τ units of time provided it has not completed by that time and there have been no arrivals or service completions of other customers. If an arrival or a service completion occurs, then π is allowed to reschedule the customer if it so desires.

The history of the system up to time t may be defined by $\boldsymbol{H}_t = (\boldsymbol{a}_t, \boldsymbol{d}_t, \boldsymbol{r}_t, \boldsymbol{f}_t, \boldsymbol{e}_t, \boldsymbol{u}_t)$ where $\boldsymbol{a}_t$ is an ordered set of arrival times of all customers that arrive prior to $t, \boldsymbol{d}_t$ is an ordered set of relative deadlines corresponding to the customers that arrive prior to $t, \boldsymbol{r}_t$, $\boldsymbol{f}_t$, $\boldsymbol{e}_t$, are ordered sets containing the times of all scheduling decisions prior to time t, the identities of the customers and the servers to which they were scheduled respectively. In addition, $\boldsymbol{u}_t$ is an ordered set of the service times for customers completed prior to time t.

We end this section by formally defining the policies of interest to us.

Definition 1 *In the case of deadlines until the beginning of service, policy π is the minimum laxity policy (ML) if at time $t'_k, (1 \leq k)$, it always schedules the eligible customer with the smallest deadline on any one of the available servers. In addition, the server is always busy as long as eligible customers are available which have not yet been served, i.e $p_0(\pi, t) = 0$ whenever the server is available and $\mathcal{C}_\pi(t) \neq \phi$.*

Definition 2 *In the case of deadlines until the beginning of service, policy π is the earliest deadline policy (ED) if at time $t'_k, (1 \leq k)$, it always schedules the eligible customer with the smallest deadline on any one of the available servers. In addition, the server is always busy as long as eligible customers are available which have not yet been served, i.e $p_0(\pi, t) = 0$ whenever the server is available and $\mathcal{C}_\pi(t) \neq \phi$.*

PREEMPTIVE SYSTEMS WITH DEADLINES TO THE END OF SERVICE

In this section we show that ED is the best service time independent policy for the preemptive continuous time and discrete time G/M/c+G queue when deadlines are to the end of service. This result is shown to apply to queues where servers take vacations.

The basis of our proofs of most of these results and the results for nonpreemptive policies is the comparison of sets of extinction times. We will show that the set of extinction times for eligible customers under ED *dominates* the set of extinction times under any other policy. Consequently, we turn our attention to the definition of dominance and the derivation of properties that it satisfies.

Consider two sets of nonnegative real numbers $R = \{x_1, x_2, \cdots, x_n\}$ and $S = \{y_1, y_2, \cdots, y_m\}$ each ordered so that $x_i \geq x_{i+1}$, $i = 1, \cdots n$ and $y_i \geq y_{i+1}$, $i = 1, \cdots m$.

Definition 3 *We say that R dominates S ($R \succ S$) if $n \geq m$ and $x_i \geq y_i$, $i = 1, 2, \cdots m$.*

We define the following three operations

- $Large(R, k) = \{x_1, x_2, \cdots, x_k\}, \quad 0 \leq k \leq n.$
- $Small(R, k) = \{x_{n-k+1}, \cdots, x_n\}, \quad 0 \leq k \leq n.$
- $Shift(R, x) = \{x_i - x \mid x_i \geq x\}.$

The following lemma gives conditions under which dominance is preserved when set operations, the *Large* operation, and the *Shift* operation are performed on R and S.

Lemma 1 *If $R \succ S$, then:*

1. *$R + \{x\} \succ S + \{x\}$, for $x > 0$,*
2. *$R - \{x_n\} \succ S$, when $n > m$,*
3. *$R \succ S - \{y\}$, where $y \in S$,*
4. *$R - \{x\} \succ S - \{y\}$, where $x \in R$, $y \in S$, and $x \leq y$,*

5. *Assume that* $R = \{x_1, \cdots, x_n\}$ *where* $x_i \geq x_{i+1}$, $1 \geq i < n$ *and* $S = \{y_1, \cdots, y_m\}$ *where* $y_i \geq y_{i+1}$, $1 \leq i < m$. *Then* $R - \{x_k\} \succ S - \{y_j\}$ *for* $k \geq j$,

6. $Shift(R, x) \succ Shift(S, x)$.

7. $Large(R, |S|) \succ S$.

8. *Assume that* R *and* S *can be expressed as* $R = R_1 + R_2$ *and* $S = S_1 + S_2$ *such that* $R_1 \succ S_1$, $|R_2| = n$, $|S_2| = n'$ *with* $n \geq n'$. *Let* $R_2 = (x_1, x_2, \cdots, x_n)$ *and* $S_2 = (y_1, y_2, \cdots, y_{n'})$ *where* $x_i \geq x_{i+1}$ *and* $y_i \geq y_{i+1}$, $i = 1, \cdots n' - 1$, *then* $R - \{x_i\} \succ S - \{y_i\}$ *for* $i = 1, \cdots, n'$.

Proof: The proof of 1, 2, 3, and 6 may be found in [18]. Properties 4, 5 and 7 follow from the operations performed on R and S and the definition of "$\succ$". The proof of 8 may be found in [25]. ∎

In order to proceed with our treatment of preemptive systems, we introduce the notation $\boldsymbol{X}_\pi(t) = (n_\pi(t), \boldsymbol{E}_\pi(t))$ where $n_\pi(t)$ is the number of customers that have made their deadlines by time t. We refer to this as the state of the system at time t under policy π. We introduce the following notion of dominance between states.

Definition 4 *We say that* $\boldsymbol{X}_{\pi_1}(t)$ *dominates* $\boldsymbol{X}_{\pi_2}(t)$ *(*$\boldsymbol{X}_{\pi_1}(t) \succ \boldsymbol{X}_{\pi_2}(t)$*) iff*

1. $n_{\pi_1}(t) \geq n_{\pi_2}(t)$,

2. $\boldsymbol{E}_{\pi_1}(t) \succ Small(\boldsymbol{E}_{\pi_2}(t), |\boldsymbol{E}_{\pi_2}(t)| + n_{\pi_2}(t) - n_{\pi_1}(t))$.

Before we prove the main result of this section we describe some guidelines used to assign customers to servers and service times to customers. Without loss of generality we restrict ourselves to policies that satisfy the following rules.

- If the number of customers being served at some point in time is $i < c$, then the first i servers are busy.

- If servers i and j are occupied where $i < j$, then the deadlines of the customers assigned to these servers must be in non-decreasing order.

If policy π does not satisfy the above rules, we can always construct a policy π^* that satisfies these rules so that $\overline{V}_N(\pi) = \overline{V}_N(\pi^*)$ for all N and $\overline{V}(\pi) = \overline{V}(\pi^*)$. There also exists an ML policy that satisfies the above rules.

We now discuss the method by which we assign service times to jobs. Divide $\boldsymbol{B}$ into $c+1$ sequences, $\boldsymbol{B}^{(j)} = \{B_{i,j}\}_{i=1,\cdots}$, $j = 1, 2, \cdots, c+1$. Consider the i-th customer. Let m_i' denote the number of times it is scheduled. Let $s_{i,1}, s_{i,2}, \cdots, s_{i,m_i'}$ be the times at which it is scheduled, $q_{i,1}, q_{i,2}, \cdots, q_{i,m_i'-1}$ be the times at which it is preempted, k_i the identity of the server at which it completes, and $m_i = \min\{j \,|\, \sum_{l=1}^{j} > s_{i,m_i'}\}$. If the i-th customer misses its deadline, then $k_i = 0$. The service time, X_i of the i-th customer is

$$X_i = \begin{cases} \sum_{l=1}^{m_i'-1}(q_{i,l} - s_{i,l}) + \sum_{l=1}^{m_i} B_{l,k} - s_{i,m_i'}, & k_i \neq 0, \\ \sum_{l=1}^{m_i'-1}(q_{i,l} - s_{i,l}) + B_{i,c+1}, & k_i = 0. \end{cases} \quad (1)$$

We claim that the service times received by customers according to this assignment rule are i.i.d. exponential r.v.'s with parameter μ.

Theorem 1 *ML maximizes the fraction of customers that complete service before their deadlines out of the class of service time independent policies for the G/M/c/K+G queue when the deadlines are to the end of service, i.e.,* $\overline{V}_N(ML) \geq \overline{V}_N(\pi)$, $N > 0$, $\overline{V}(ML) \geq \overline{V}(\pi)$ *where* π *is any service time independent policy.*

Proof: The proof by forward induction on the times that the following events can occur,

- $\mathcal{E}_0$ - arrival to both systems,

- $\mathcal{E}_1$ - completion of a job in either or both systems,

- $\mathcal{E}_2$ - job missing deadline under one or both policies,

Let $(t_0, \sigma_0), (t_1, \sigma_1), \cdots$ be the sequence of times and events that occur at those times, i.e., event σ_i occurs at time t_i where $\sigma_i \in \{\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2\}$.

We will demonstrate that $\boldsymbol{X}_{ML}(t) \succ \boldsymbol{X}_\pi(t)$ for every sample $\boldsymbol{S} = \boldsymbol{s}$ and $t \geq 0$ provided that $\boldsymbol{X}_{ML}(t_0) \succ \boldsymbol{X}_\pi(t_0)$.

According to property 6 of Lemma 1, if $\boldsymbol{X}_{ML}(t_i) \succ \boldsymbol{X}_\pi(t_i)$, and $t_i < t_{i+1}$, then $\boldsymbol{X}_{ML}(t) \succ \boldsymbol{X}_\pi(t)$ for $t_i \leq t < t_{i+1}$.

We proceed with our inductive argument.

Basis Step: The hypothesis is trivially true for $t = t_0$.

Inductive Step: Assume that $\boldsymbol{X}_{ML}(t_l) \succ \boldsymbol{X}_\pi(t_l)$ for $l \leq i$. We now show that it also holds for $i+1$. There are three cases according to the type of event.

Case 1 ($\sigma_{i+1} = \mathcal{E}_0$): In this case, neither n_π nor n_{ML} are affected and property 1 of Lemma 1 guarantees that $\boldsymbol{X}_{ML}(t_{i+1}) \succ \boldsymbol{X}_\pi(t_{i+1})$.

Case 2 ($\sigma_{i+1} = \mathcal{E}_1$): There are three subcases according to whether the completion is under π, ML, or both policies. If the completion is under π only, then it occurs on server j where $j > |\boldsymbol{E}_{ML}(t_{i+1}^-)|$. This implies that $|\boldsymbol{E}_\pi(t_{i+1}^-)| > |\boldsymbol{E}_{ML}(t_{i+1}^-)|$ which further implies that $n_{ML}(t_{i+1}^-) > n_\pi(t_{i+1}^-)$. Consequently $n_{ML}(t_{i+1}) = n_{ML}(t_{i+1}^-) \geq n_\pi(t_{i+1}^-) = n_\pi(t_{i+1}^-) +$

1 and $\boldsymbol{E}_{ML}(t_{i+1}) = \boldsymbol{E}_{ML}(t_{i+1}^-) \succ Small(\boldsymbol{E}_\pi(t_{i+1}^-), |\boldsymbol{E}_\pi(t_{i+1}^-)+ n_\pi(t_{i+1}^-) - n_{ML}(t_{i+1}^-)) = Small(\boldsymbol{E}_\pi(t_{i+1}), |\boldsymbol{E}_\pi(t_{i+1})+n_\pi(t_{i+1}) - n_{ML}(t_{i+1}))$. Hence $\boldsymbol{X}_{ML}(t_{i+1}) \succ \boldsymbol{X}_\pi(t_{i+1})$.
If the completion is under ML only, then a similar calculation yields $\boldsymbol{X}_{ML}(t_{i+1}) \succ \boldsymbol{X}_\pi(t_{i+1})$.
If the completion is under both policies, then property 5 of Lemma 1 ensures that $\boldsymbol{X}_{ML}(t_{i+1}) \succ \boldsymbol{X}_\pi(t_{i+1})$.

Case 3 ($\sigma_{i+1} = \mathcal{E}_2$): Again there are three subcases according to whether the customer misses his deadline under π, ML, or both policies. If under π, property 3 of Lemma 1 is applicable. If under ML, property 2 of Lemma 1 is applicable. Last, property 4 of Lemma 1 is applicable when the losses occur under both policies.

It follows that $E[V_N(ML)|\boldsymbol{S} = \boldsymbol{s}] \geq E[V_N(\pi)|\boldsymbol{S} = \boldsymbol{s}]$ and $\overline{V}_N(ML) \geq \overline{V}_N(\pi)$ for $N = 1, 2, \cdots$ and $\overline{V}(ML) \geq \overline{V}(\pi)$. ∎

Remark. Similar results can also be proven for the discrete time bulk arrival G/M/c/K+G queue. Here the service time consists of an integer number of time units that is given by a geometric r.v. This model is of particular use in data communications in the case that the service time is always a single time unit. It forms the basis of many models of statistical multiplexers. In the case that customers require a single time unit of service, there is no distinction between preemptive and non-preemptive systems. Furthermore, there is no distinction between systems in which customers must meet their deadlines either by the time service begins or by the time service completes.

Theorem 1 can be generalized to include systems in which servers take vacations. This is of interest for at least two reasons. First, processors in any multiprocessor system are prone to failures. Second, systems in which servers take vacations can be used to model real-time systems with two classes of customers. For example, one class of tasks may be unable to tolerate missed deadlines. The second class of jobs may be able to tolerate some missed deadlines. If the tasks in the first class are well understood (i.e., known service times, arrival times), they can be given higher priority than the second class of tasks and scheduled independently of the second class. The second class of tasks are like the customers that we have considered in our model for which the object is to develop policies that will minimize the fraction of tasks that miss their deadlines. Thus tasks in the second class see a system where servers take vacations.

Let $\{U_{i,j}, W_{i,j}\}_{i=1,\cdots}$, $j = 1, 2, \cdots, c$ be families of r.v.'s such that $U_{i,j}$ is the length of the i-th time interval during which the j-th server is available for service and $W_{i,j}$ is the length of the i-th time interval during which the j-th server is on vacation (unavailable for service). We allow these sequences of r.v.'s to have arbitrary statistics so long as they are independent of $\boldsymbol{A}$, $\boldsymbol{B}$, $\boldsymbol{D}$. In this case we state the following result.

Theorem 2 *ML maximizes the fraction of customers that make their deadlines in the preemptive continuous and discrete time G/M/c/K+G queue with vacations when the deadlines are to the end of service, i.e.,* $\overline{V}_N(ml) \geq \overline{V}_N(\pi)$, $N > 0$, $\overline{V}(ML) \geq \overline{V}(\pi)$ *for any service time independent policy* π.

Proof. The proof is similar to that given for Theorem 1 and is omitted here. ∎

NON-PREEMPTIVE SYSTEMS WITH DEADLINES TO END OF SERVICE

In this section we show that ED is the best policy from the class of non-idling policies for the non-preemptive G/M/c/K +G queue when deadlines are to the end of service.

Consider a policy π that is allowed to preempt a customer solely to move him to another server. We refer to this as a *limited preemption* policy and claim that the performance of this policy does not differ from a policy that uses the same scheduling rules except that it does not allow preemptions. We will find it easier to work with these limited preemption policies. Specifically, we consider limited preemption policies that enforce the following rules:

- If the number of customers in service, n is less than the number of servers, then they are placed on the first n servers.
- Customers are placed on servers such that the deadline associated with the customer on the i-th server is greater than or equal to that associated with the customer on the $(i+1)$-th server.

Customers are assigned service times according to the same rule used in analyzing the system that allows preemptions.

Theorem 3 *ED provides the best performance of all non-idling service time independent policies for the non-preemptive G/M/c/K+G queue when the deadlines are to the end of service, i.e.,* $\overline{V}_N(ED) \geq \overline{V}_N(\pi)$, $N > 0$, $\overline{V}(ED) \geq \overline{V}(\pi)$ *for any non-idling service time independent policy* π.

Proof: Though similar to the proof of Theorem 1, the proof of this theorem is more intricate because of the fact that the set of deadlines of the customers in service under ED may not dominate the set of deadlines of the customers under an arbitrary policy π. We show instead that both $\boldsymbol{E}_{ED}(t) \succ \boldsymbol{E}_\pi(t)$ and $\boldsymbol{R}_{ED}(t) \succ \boldsymbol{R}_\pi(t)$ for every sample path $\boldsymbol{S} = \boldsymbol{s}$ using a forward induction argument on the times of events. These events are the same as defined in Theorem 1. Let $(t_0, \sigma_0), (t_1, \sigma_1), \cdots$ be the sequence of times and events that occur at those times, i.e., event σ_i occurs at time t_i where $\sigma_i \in \{\mathcal{E}_0, \mathcal{E}_1, \mathcal{E}_2\}$.

We note as in Theorem 1 that if $\boldsymbol{E}_{ED}(t_i) \succ \boldsymbol{E}_\pi(t_i)$ and $\boldsymbol{R}_{ED}(t_i) \succ \boldsymbol{R}_\pi(t_i)$ and $t_i < t_{i+1}$, then $\boldsymbol{E}_{ED}(t) \succ \boldsymbol{E}_\pi(t)$ and $\boldsymbol{R}_{ED}(t) \succ \boldsymbol{R}_\pi(t)$ for $t_i \le t < t_{i+1}$.

We proceed with our inductive argument.

Basis Step: The hypothesis is trivially true for $t = t_0$.

Inductive Step: Assume that $\boldsymbol{E}_{ED}(t_l) \succ \boldsymbol{E}_\pi(t_l)$ and $\boldsymbol{R}_{ED}(t_l) \succ \boldsymbol{R}_\pi(t_l)$ for $l \le i$. We now show that it also holds for $i+1$. There are three cases according to the type of event.

Case 1 ($\sigma_{i+1} = \mathcal{E}_0$): This case is similar to case 1 in Theorem 1 and the details are omitted.

Case 2 ($\sigma_{i+1} = \mathcal{E}_1$): There are two subcases according to whether the completion is under ED or both policies. (Note: according to the inductive hypothesis and the server assignment rule, a completion under π implies a completion under ED.) If the completion is under ED only, then $\boldsymbol{E}_\pi(t_{i+1}) = \emptyset$ which implies that $\boldsymbol{E}_{ED}(t_{i+1}) \succ \boldsymbol{E}_\pi(t_{i+1})$. Because of the way that customers are assigned to servers, the deadline of the completed customer cannot reside in $Large(\boldsymbol{R}_{ED}(t_i^-), |\boldsymbol{R}_\pi(t_i^-)|)$. Consequently $\boldsymbol{R}_{ED}(t_{i+1}) \succ \boldsymbol{R}_\pi(t_{i+1})$. If the completion is under both policies, then property 8 of Lemma 1 and the inductive hypothesis ensure that $\boldsymbol{R}_{ED}(t_{i+1}) \succ \boldsymbol{R}_\pi(t_{i+1})$. The inductive hypothesis and the fact that ED will schedule the customer with the smallest deadline from $\mathcal{C}_{ED}(t_i^-)$ ensures that property 5 of Lemma 1 can be applied to show that $\boldsymbol{E}_{ED}(t_{i+1}) \succ \boldsymbol{E}_\pi(t_{i+1})$.

Case 3 ($\sigma_{i+1} = \mathcal{E}_2$): Again there are three subcases according to whether the customer misses his deadline under π, ED, or both policies. If under π, property 3 of Lemma 1 is applicable. If under ED, property 2 of Lemma 1 is applicable. If under both ED and π, then we have further subcases according to whether the customers were in service or in the queue. In all of these cases, the result is obtained by using property 4 from Lemma 1.

It follows that $\overline{V}_N(ED) \ge \overline{V}_N(\pi)$ for $N = 1, 2, \cdots$ and $\overline{V}(ED) \ge \overline{V}(\pi)$. ∎

Remark. Similar results can be proven for discrete time queues.

NON-PREEMPTIVE SYSTEMS WITH DEADLINES TO BEGINNING OF SERVICE

In this section we show that ML is the best policy out of the class of *non-idling policies* for these queues when service times are restricted to be independent and identically distributed exponential random variables.

Theorem 4 *ED is the best non-idling, service time independent policy for the non-preemptive G/G/c/K+G system with deadlines until the beginning of service, i.e.,* $\overline{V}_N(ED) \ge \overline{V}_N(\pi)$, $N > 0$, $\overline{V}(ED) \ge \overline{V}(\pi)$ *for any non-idling, service time independent policy* π.

Proof: Define $\boldsymbol{T}_\pi(t) = (t_\pi^{(1)}(t), \cdots, t_\pi^{(c)}(t))$ where $t_\pi^{(j)}(t) = 1$ if server j is busy under π at time t and 0 otherwise.

As before, the proof is by forward induction. Using the properties of the dominance relation "$\succ$" We show that $\boldsymbol{E}_{ED}(t) \succ \boldsymbol{E}_\pi(t)$ and $\boldsymbol{T}_{ED}(t) \ge \boldsymbol{T}_\pi(t)$ at each possible event (i.e., arrival, departure, and deadline miss) for every sample path $\boldsymbol{S}$. The arguments are similar to those found in Theorems 1 and 3. It follows that $\overline{V}_N(ED) \ge \overline{V}_N(\pi)$ for $N = 1, 2, \cdots$ and $\overline{V}(ED) \ge \overline{V}(\pi)$. ∎

Remark. Analogous results hold also for bulk arrival discrete time G/M/c/K+G queues.

EXTENSIONS

These optimality results can be extended in a number of different ways. For example, if there is storage capacity for only a finite number of jobs, then we have shown that the ML and ED scheduling policies should be coupled with buffer management policies that operate in a similar way. Specifically the optimal buffer management policy should operate in the following manner

- Whenever a job arrives and there is sufficient space in the buffer for it, admit it.
- Whenever a job arrives to a full buffer, throw away the job closest to its deadline from among all of the jobs present, including the newly arrived job.

Variations of this along with the ML or ED policies maximize the fraction of customers making their deadlines under the assumptions stated in the earlier sections. Details of these results are found in [25].

These results can also be extended to the class of idling and non-idling policies for the case of non-preemptive systems. It is possible to show that the best policies belong to the class of minimum laxity with inserted idle time (MLI) policies and earliest deadline with inserted idle time (EDI) policies for similar assumptions as in preceding sections. Members of these classes policies either chooses to idle a server when there is work in the queue or uses the ML and ED rules for scheduling jobs. Details of these results can also be found in [25].

Last, results have been obtained for systems where deadlines are not known to the scheduler *except when jobs miss their* *deadlines.* For example, if the deadline distribution has an em increasing failure rate (IFR) [21], then the optimal policy is first-come first-serve (FCFS). Similarly, if the deadline distribution has an em decreasing failure rate (DFR) [21], then the optimal policy is lastst-come first-serve (LCFS).

SUMMARY

We have shown, out of the class of non-idling service time independent policies, ML maximizes fraction of customers that begin service by their deadlines for the nonpreemptive G/M/c/K+G queue when deadlines are until the beginning of service and that ED maximizes fraction of customers that complete service by their deadlines for the nonpreemptive G/M/c/K+G queue when deadlines are until the end of service. Last, if deadlines are to the end of service, then the best policy that does not use service time information for the preemptive G/M/c/K+G queue is ED. These results hold for systems in which servers take vacations.

REFERENCES

[1] F. Baccelli, P. Boyer, and G. Hebuterne, "Single-Server Queues with Impatient Customers" *Adv. Appl. Prob.*, Vol. 16, pp. 887-905, 1984.

[2] P. Bhattacharya and A. Ephremides, "Scheduling of Time-Critical Messages," *Proc. 1988 Conf. Inf. Sciences Systems* pp. 623-628, 1988.

[3] P. Bremaud, *Point Processes and Queues: Martingale Dynamics*, Springer-Verlag, New York, 1981.

[4] T.M. Chen, J. Walrand, and D.G. Messerschmitt, "Dynamic Priority Protocols for Packet Voice", *IEEE Journal on Sel. Areas in Comm.*, **7**, 5, pp. 632-643, June 1989.

[5] E.G. Coffman (Ed.), *Computer and Job-Shop Scheduling Theory*, John Wiley, New York, 1976.

[6] J.W. Cohen, "Single Server Queues with Restricted Accessibility," *J. Eng. Math.*, Vol. 3, 4, pp. 265-284, Oct. 1969.

[7] R.W. Conway, W.L. Maxwell and L.W. Miller, *Theory of Scheduling*, Addison-Wesley, Reading, Mass., 1967.

[8] M. Dertouzos, "Control Robotics: The Procedural Control of Physical Processes," *Proceedings of the IFIP Congress*, pp. 807-813, 1974.

[9] B.T. Doshi and E.H. Lipper, "Comparisons of Service in a Queueing System with Delay Dependent Customer Behavior", *Applied Probability - Computer Science: The Interface, Vol. II*, R.L. Disney. T.J. Ott, Eds., Cambridge, MA, Birkhauser, pp. 269-301, 1982.

[10] B. Gavish and P.J. Schweitzer, "The Markovian Queue with Bounded Waiting Time," *Management Sci.*, Vo. 21, 12, pp. 1349-1357, Aug. 1972.

[11] J. Hong, X. Tan, D. Towsley, "A Performance Analysis of Minimum Laxity and Earliest Deadline Scheduling in a Real-Time System," to appear in *IEEE Trans. Computers*, Dec. 1989.

[12] J.R. Jackson, "Scheduling a Production Line to Minimize Maximum Tardiness," Research Report 43, Management Science Research Report, UCLA, 1955.

[13] Y. Lim and J. Kobza, "Analysis of a Delay-Dependent Priority Discipline in a Multi-Class Traffic Packet Switching Node", *Proc. IEEE INFOCOM-88*, New Orleans, LA, 1988.

[14] C. Liu and J. Layland, "Scheduling Algorithms for Multi-Programming in a Hard-Real-Time Environment," *J. ACM*, Vol. 20, pp. 46-61, 1973.

[15] A. Mok and M. Dertouzos, "Multiprocessor Scheduling in a Hard Real-Time Environment," *Proc. 7th Texas Conf. on Comp. Syst.*, Nov. 1978.

[16] J.M. Moore, "An n Job, One Machine Sequencing Algorithm for Minimizing the Number of Late Jobs," *Management Science*, vol. 15, pp. 102-109, 1968.

[17] S.S. Panwar, *Time Constrained and Multi-access Communications*, Ph.D. Thesis, Department of Electrical and Computer Engineering, University of Massachusetts, Amherst, Feb. 1986.

[18] S.S. Panwar, D. Towsley, and J.K. Wolf, "Optimal Scheduling Policies for a Class of Queues with Customer Deadlines to the Beginning of Service," *J. ACM*, **35**, 4, pp. 832-844, Oct. 1988.

[19] W.P. Pierskalla and C. Roach, "Optimal Issuing Policies for Perishable Inventory," *Management Science*, vol. 18, pp. 603-614, 1972.

[20] M. Pinedo, "Stochastic Scheduling with Release Dates and Due Dates," *Operations Research*, vol. 31, pp. 559-572, 1983.

[21] S.M. Ross, *Stochastic Processes*, John Wiley & Sons, New York, 1983.

[22] M. Schwartz, *Telecommunication Networks: Protocols, Modeling and Analysis*, Addison Wesley, 1987.

[23] Zaw-Sing Su and Kenneth C. Sevcik, "A Combinatorial Approach to Scheduling Problems," *Operations Research*, vol. 26, pp. 836-844, 1978.

[24] D. Towsley, F. Baccelli, "Comparisons of service disciplines in a tandem queueing network with real-time constraints", to appear in *Operations Research Letters*, November 1990.

[25] D. Towsley, S.S. Panwar, "Comparison of Service and Buffer Overflow Policies for Multiple Server Queues that Serve Customers with Deadlines", COINS Technical Report, Dept. of Computer & Information Science, Univ. Massachusetts.