A PERFORMANCE ANALYSIS OF MINIMUM LAXITY AND EARLIEST DEADLINE SCHEDULING IN A REAL-TIME SYSTEM

J. Hong, X. Tan, D. Towsley
Department of Computer and Information Science
University of Massachusetts
Amherst, MA 01003

COINS Technical Report 89-71

July 1989

To appear in IEEE Transactions on Computers, December, 1989.

A Performance Analysis of Minimum Laxity and Earliest Deadline Scheduling in a Real-Time System

J. Hong[1], X. Tan, D. Towsley[2]
Department of Computer & Information Science
University of Massachusetts
Amherst, MA 01003

July 1989

Abstract

In this paper we study the performance of a real-time system in which jobs either all have deadlines until the beginning of service or deadlines until the end of service. In the first case we analyze the minimum laxity scheduling policy (ML) when there are c processors and, in the latter case, the preemptive resume earliest deadline scheduling policy (ED) when there is one server. In both cases, the analysis assumes a Poisson arrival process, exponential service times that are not known to the scheduler, and exponential laxities or deadlines. In both cases, we develop families of upper and lower bounds on the fraction of jobs that miss their deadlines. The pessimistic bounds are of special interest because they correspond to a family of implementable policies, $ML(n)$ and $ED(n)$, $n = 1, \cdots$ where the performance approaches that of ML and ED as n increases, but at the cost of increasing overhead. Numerical results show that the bounds are tight, that the difference between the performances of policies that do not use deadline information, (e.g., FIFO) and ML and ED is nonnegligible and that, even for small values of n,(e.g., $n = 3$), $ML(n)$ and $ED(n)$ perform well when compared to ML and ED.

[1]The work of this author was partially supported by the National Science Foundation under grant DCI-87-96236.

[2]The work of the last two authors was partially supported by the Office of Naval Research under contract N0014-87-K-0796 and the National Science Foundation under grant DCR-85-00332.

1 Introduction

There has been increased interest over the last several years in developing models for evaluating the performance of scheduling policies for single and multiprocessor systems which serve jobs with time constraints. Unlike models for traditional computer systems which attempt to predict average job delays, models of real-time systems must accurately estimate the fraction of jobs violating their constraints. In this paper we describe a model for obtaining upper and lower bounds on this performance metric for policies that give priority to jobs with the tightest time constraints. The analysis is conducted under the assumptions of Poisson arrivals, exponential service times, and time constraints that are exponential random variables. We consider two systems, one in which jobs must begin service by a time constraint and another system in which jobs must complete service by a time constraint.

Time constraints on the beginning of service are typically called *laxities* and time constraints on the end of service are called *deadlines*. Panwar and Towsley [?] have shown that the *minimum laxity* policy (ML) maximizes the fraction of jobs beginning service by their laxity out of the class of all non-preemptive work conserving policies for the $G/M/c+G$ system (here the last G denotes that laxities can come from a general distribution) when constraints are to the beginning of service. They also prove that the *earliest deadline* policy maximizes the fraction of jobs meeting their constraints out of the class of work conserving policies that allow preemptions for the $G/M/1+G$ system when jobs have constraints to the end of service and where service times are not known to the scheduler. This policy will preempt a server whenever a job arrives that has a tighter constraint than the job in service. In this paper we develop models that provide bounds on the fraction of jobs meeting their laxities for the $M/M/c+M$ system. We describe a similar model that provides bounds on the performance of the $M/M/1+M$ system when jobs have deadlines. Estimates of the performance of ML and ED are given as the average of the upper and lower bounds. The paper concludes with a comparison of the accuracy of these approximations to simulation.

The pessimistic bounds on the fraction of jobs making their time constraints correspond to the performance of two new policies, $ML(n)$ and $ED(n)$ which, for small values of n, (e.g., $n=3$) incur little overhead, and yet provide performance comparable to that of ML and ED. Briefly, $ML(n)$ uses the ML on the first n jobs in the queue and handles the remaining jobs in a FIFO manner. Hence, if the total number of jobs is $n+m$, then

$ML(n)$ requires $O(n)$ time per job and ML requires $O(n+m)$ time per job.[3] $ED(n)$ behaves in a similar manner. In the case of deadlines until the beginning of service, we compare the performances of $ML(3)$, $ML(2)$, ML and work conserving policies (e.g., FIFO, LCFS) that do not use information regarding time constraints when scheduling jobs. A similar comparison is performed In the case of constraints until the end of service.

Numerous papers address the problem of designing and analyzing the behavior of scheduling policies for real-time systems. In addition to showing the optimality of ML and ED for the systems described above, [9,10] show the optimality of these policies or variants of these policies for a variety of other systems. The analysis of queueing systems handling customers with deadlines has been addressed in numerous papers [2,3,1,4,5,8,6,11,14]. However, except for [14], none of these papers have focussed on scheduling policies that use information on time constraints when scheduling jobs. The last paper provides an approximate analysis of a variant on ML that places the job with the largest laxity at the end of the queue and serves the remaining jobs in FIFO order.

In addition to the above literature that focusses on the exact analysis of simple policies, there exists a growing literature on the design and evaluation (through simulation) of heuristics for scheduling jobs with real-time constraints. The reader is referred to [12] for references to this work.

The paper is organized in the following way. Section 2 contains a definition of ML and the analysis leading to bounds on its performance. A new policy $ML(n)$, $n = 2, 3, \cdots$, which yields a pessimistic bound on the performance of ML is also introduced in Section 2. Section 3 defines ED and describes bounds on its performance. The analysis leading to these bounds is omitted as it is nearly identical to that contained in Section 2. Numerical results are found in section 4. Section 5 summarizes the contributions of the paper.

2 The Minimum Laxity Scheduling Policy

This section begins with a definition of ML and the Markov chain, $\mathcal{M}$, which describes its behavior in a $M/M/c+M$ system. The Markov chain $\mathcal{M}$ is modified to obtain

[3]These times can be reduced to $O(\log n)$ and $O(\log(n+m))$ by use of a data structure such as a heap. However, this may not be more efficient than a list unless m is large.

two new chains, $\mathcal{M}^-(n)$ and $\mathcal{M}^+(n)$ which yield pessimistic and optimistic bounds on the fraction of jobs making their deadlines. The pessimistic bound corresponds to the performance of an implementable policy $ML(n)$ which is also defined. Conditions for the ergodicity of the Markov chains are also established in this section.

2.1 The Minimum Laxity Policy and its Markov Chain Representation

In this subsection, we define ML and the Markov chain representation of its behavior on a c processor system. The proof that the Markov chain presented here actually models ML can be found in [7].

We consider c processors serving jobs from a single queue. Jobs arrive according to a Poisson process with parameter λ. Each job requires a service time that is exponentially distributed with mean $1/\mu$. In addition, there is a deadline associated with each job. A job is thrown away if it cannot begin execution before its deadline; otherwise it is executed. A job that is thrown away is considered to be lost. We denote the interval of time between the arrival of a job and its deadline as its *laxity*. We assume that laxities are exponentially distributed r.v.'s with parameter α. The exponential assumptions are necessary to obtain a tractable continuous time Markov chain.

Any number of different policies can be used to schedule jobs to processors. However, Panwar and Towsley [10] have proven that the minimum laxity scheduling policy (ML) maximizes the fraction of jobs that begin their service by their deadlines from the class of all policies that never allow processors to go idle while there is work in the queue. Here ML is the nonpreemptive policy that always schedules the job with the smallest laxity, i.e., the job closest to its deadline.

The behavior of the system is described by a continuous time Markov chain, $\mathcal{M}$, having state $(I(t), S(t))$ where $I(t)$ denotes the number of jobs in service, $0 \leq I(t) \leq c$, and $S(t)$ is a binary string taken from the set $\mathcal{B} = \{\wedge\} \cup 1(0+1)^*$. In other words, $S(t)$ can be either a null string, $\wedge$, or any binary string beginning with a 1. Whenever $S(t)$ is not the null string, it is useful to visualize it as the concatenation of $g(t) > 0$ strings (groups of bits) where each string consists of one or more 1's followed by one or more 0's (except for the last string which may include no 0's), i.e.,

$$S(t) = 1^{m_1}0^{n_1}1^{m_2}0^{n_2}\cdots 1^{m_{g(t)}}0^{n_{g(t)}},\ m_i > 0, 1 \leq i \leq g(t),\ n_i > 0, 1 \leq i \leq g(t)-1,$$

$$n_{g(t)} \geq 0,\ g(t) \geq 1.$$

The allowable states for $\mathcal{M}$ is $\mathcal{S} = \{(n, \wedge) | 0 \leq n \leq c\} \cup \{(c, s) | s \in \mathcal{B}\}$. The string $S(t)$ can be interpreted as a *partial history* of the system up to time t. Let $n(s)$ denote the number of bits in the binary string $s \in (0+1)^*$. The bits in the string correspond to the $n(S(t))$ most recently arrived jobs whose deadlines have not passed. 0's correspond to jobs that have made their deadlines whereas 1's correspond to jobs still in the queue. Consider the j-th group, $1^{m_j}0^{n_j}$ within $S(t)$. This group corresponds to $m_j + n_j$ jobs of which n_j have made their deadlines (but their deadlines have not expired) and m_j reside still in the queue. $S(t)$ carries the additional semantics that the group of m_j jobs still in the queue have deadlines larger than the n_j jobs that already left the queue. Thus the deadlines of these jobs cannot expire until the deadlines of the n_j jobs do. Furthermore, the group of m_j jobs still in the queue have deadlines larger than the $\sum_{l=j+1}^{g(t)} n_l$ jobs that have made their deadlines but whose deadlines are later than t. These are the jobs represented by zeros to the right of the j-th group in the string $S(t)$.

The Markov process $\mathcal{M}$ exhibits the following behavior.

Arrival An arrival has the effect of either incrementing i by one if $i < c$ or adding a 1 to the right of s if $i = c$. The transitions are illustrated below:

$$\begin{aligned} (i, \wedge) &\xrightarrow{\lambda} (i+1, \wedge), \quad i < c, \\ (c, s) &\xrightarrow{\lambda} (c, s1), \quad s \in \mathcal{B}. \end{aligned}$$

Service completion A departure has the effect of decreasing i by one if $s = \wedge$ or changing the rightmost 1 in s to a 0 otherwise. The transitions are illustrated below:

$$\begin{aligned} (i, \wedge) &\xrightarrow{i\mu} (i-1, \wedge), \quad 1 \leq i \leq c, \\ (c, s10^n) &\xrightarrow{c\mu} (c, s0^{n+1}), \quad s \in \mathcal{B} - \{\wedge\},\ 0 \leq n \\ (c, 10^n) &\xrightarrow{c\mu} (c, \wedge). \end{aligned}$$

Deadline Expiration Transitions due to the expiration of deadlines occur only if $s \neq \wedge$. Consider $s = 1^{m_1}0^{n_1}1^{m_2}0^{n_2}\cdots 1^{m_g}0^{n_g}$. For $1 \leq j \leq g-1$, one of the n_j 0's is deleted if the deadline of any of the $m_j + n_j$ jobs within the j-th group

expires. This occurs with rate $(m_j + n_j)\alpha$. The same happens to the g-th group provided $n_g > 0$. If $n_g = 0$ and the deadline of one of the m_g jobs in the g-th group expires, then a 1 is deleted from that group. In either case this occurs with rate $(m_g + n_g)\alpha$. The transitions are illustrated below:

$$(c, s_1 1^n 0^m s_2) \xrightarrow{(n+m)\alpha} (c, s_1, 1^n 0^{m-1} s_2), \quad s_1 \in \{\wedge\} \cup 1(0,1)^* 0,\ s_2 \in \mathcal{B},\ 1 \leq m, n,$$
$$(c, s1^n) \xrightarrow{n\alpha} (c, s1^{n-1}), \quad s \in \{\wedge\} \cup 1(0,1)^* 0.$$

Later in this section we will show that $\mathcal{M}$ is an irreducible positive recurrent chain. Hence we define the limiting state $(I, S) = \lim_{t\to\infty}(I(t), S(t))$ and let $\pi(i, s) = \Pr[I = i, S = s]$, $1 \leq i \leq c$, $s \in \mathcal{B}$. The equations satisfied by these probabilities can be found in the Appendix.

The fraction of customers lost, Q, can be computed from the stationary distribution by the following expression

$$Q = 1 - \frac{\left(c - \sum_{i=0}^{c-1} \pi(i, \wedge)(c - i)\right)\mu}{\lambda}. \tag{1}$$

We have the following result regarding the correctness of $\mathcal{M}$. The proof can be found in [7].

Theorem 1 *The Markov process $\mathcal{M}$ exactly simulates ML in the sense that the long term fraction of customers lost under ML equals Q given in equation(1).*

We conclude this subsection with the proof of the irreducibility and positive recurrence of $\mathcal{M}$. We find it useful for the proof of the next theorem and the comparison of different systems to introduce the idea of stochastic ordering between two r.v.'s

Definition 1 *Let X and Y be two r.v.'s. The r.v. X is stochastically greater than or equal to Y (written as $X \geq_{st} Y$) iff*

$$\Pr[X \leq x] \leq \Pr[Y \leq x], \quad -\infty < x < \infty.$$

Theorem 2 *The Markov chain $\mathcal{M}$ is an irreducible positive recurrent chain for $\lambda > 0$, $\mu > 0$, $\alpha > 0$.*

Proof. First we show that $\mathcal{M}$ is irreducible. We do this by showing that it is possible to transition from $(0,\wedge)$ to any other state $(i,s) \in \mathcal{S}$ in a finite number of steps and from (i,s) back to $(0,\wedge)$ also in a finite number of steps. We consider $(0,\wedge) \rightarrow (i,s)$ first. If $i < c$, then $s = \wedge$ and the transition occurs with n successive arrivals. If $i = c$ and $s = 1^{m_1}0^{n_1}\cdots 1^{m_g}0^{n_g}$, then the transition occurs with $c + m_1 + n_1$ arrivals followed by n_1 departures followed by $m_2 + n_2$ arrivals followed by n_2 departures followed by $\cdots$ followed by $m_g + n_g$ arrivals followed by n_g departures.
The transition $(i,s) \rightarrow (0,\wedge)$ is shown in a similar manner and is omitted.

Next we show that $(i,\wedge)$ is a positive recurrent state for some $1 \leq i \leq c$. To do this we show that the mean time between visits to $(0,\wedge)$ is finite. Consider a new Markov chain $\mathcal{M}'$ corresponding to a $M/M/\infty$ queue with arrival rate λ and service rate α. Let $L(t)$ denote the number of customers in the infinite server system at time t. It should be clear that $L(t) \geq_{st} n(S(t))$. Hence we have that $\Pr[n(S(t))] = 0] \geq \Pr[L(t) = 0]$. The event $n(S(t)) = 0$ corresponds to the finite set of states $\{(i,\wedge)|1 \leq i \leq c\}$. It follows that there is some i, $1 \leq i \leq c$ such that $\Pr[I(t) = i, S(t) = \wedge] > 0$ for $0 < t$. However, this can only occur if $(i,\wedge)$ is visited infinitely often and the mean time between visits is finite. Therefore $(i,\wedge)$ is a positive recurrent state. Since $\mathcal{M}$ is irreducible, it follows necessarily that every $(i,s) \in \mathcal{S}$ is positive recurrent. □

2.2 The Policy $ML(n)$ and the Markov Chain $\mathcal{M}^-(n)$

We now describe an implementable scheduling policy that is simpler to analyze than ML and provides an upper bound on the loss probability for ML. We divide the queue into two parts, Q_1 and Q_2. Q_1 can hold at most n jobs. If the total number of jobs waiting for service is less than or equal to n, then they are placed into Q_1. Jobs in Q_1 are scheduled using ML. However if there are more than n jobs in the queue, we place new incoming jobs into Q_2. When the scheduler moves a job from Q_1 to a processor, it also moves a job from Q_2 to Q_1 in the order of arrival. In short, Q_1 is a ML queue of size n and Q_2 is a FIFO queue of unbounded size.

Accordingly, we define a new Markov process $\mathcal{M}^-(n)$ with state $(I^-(t,n), S^-(t,n))$ where $I^-(t,n)$ is the number of busy servers and $S^-(t,n)$ is the partial state history at time $t > 0$. The state space is $\mathcal{S}^-(n) = \{(n,\wedge)|1 \leq n \leq c\} \cup \{(c,s)|s \in \mathcal{B}^-(n)\}$ where $\mathcal{B}^-(n) = 1\left(\bigcup_{i=0}^{n-2}(0^*1)^i\right)0^* \bigcup 1(0^*1)^{n-2}0^*1^*$. State transitions obey the following rules.

Arrival The same as in $\mathcal{M}$

Service Completion The behavior is the same as in $\mathcal{M}$ if the total number of 1's in s is less than or equal to n. If the total number of 1's exceeds n then the n-th 1 from the left is converted into a 0.

Deadline Expiration The same as in $\mathcal{M}$

We define $(I^-(n), S^-(n)) = \lim_{t\to\infty}(I^-(t,n), S^-(t,n))$ and let $\pi^-(i,s,n) = \Pr[I^-(n) = i, S^-(n) = s]$ denote the stationary distribution of $\mathcal{M}^-(n)$. These probabilities exist whenever $\lambda > 0$, $\mu > 0$, $\alpha > 0$ and satisfy a set of equations similar to those given for $\pi()$.

The Markov chain can be shown to simulate $ML(n)$ correctly and the loss probability, $Q^-(n)$ is given by the same expression as Q with $\pi^-()$ replacing $\pi()$.

2.3 The Markov Chain $\mathcal{M}^+(n)$

We define another continuous time Markov chain $\mathcal{M}^+(n)$ to have a state $(I^+(t,n), S^+(t,n))$ taken from the set $\mathcal{S}^+(n) = \{(n,\wedge)|1 \le n \le c\} \bigcup \{(c,s)|s \in \mathcal{B}^+(n)\}$ where $\mathcal{B}^+(n) = 1\left(\bigcup_{i=0}^{n-2}(0^*1)^i\right)0^* \bigcup 1(0^*1)^{n-2}0^*1^*0^*$. Transitions between the states are governed by the following three rules.

Arrival If $i < c$ then $s = \wedge$ and i is incremented by 1. If $i = c$ and the number of 1's in s is less than n, then append a 1 to the right end of s. If the number of 1's in s is greater than or equal to n, then insert a 1 after the n-th 1 in s counting from the left.

Service Completion Same as $\mathcal{M}$.

Deadline Expiration Same as $\mathcal{M}$

This chain is irreducible positive recurrent for $\lambda > 0$, $\mu > 0$, $\alpha > 0$ and does not correspond to any implementable policy. Let $\pi^+(i,s,n)$ denote the stationary distribution for $\mathcal{M}^+(n)$. It satisfies a set of equations slightly different from those satisfied by $\pi()$ which are omitted here. Once the stationary distribution is obtained, the fraction of jobs that miss their deadlines, $Q^+(n)$, can be obtained from equation (1) by replacing $\pi()$ with $\pi^+()$.

2.4 Bounds on the Performance of ML

In this section we show the following ordering between the loss probabilities obtained from the three chains, $Q^-(n) \geq Q^-(n+1) \geq Q \geq Q^+(n+1) \geq Q^+(n)$, $1 \leq n$.

In order to do so we introduce a dominance relation among elements in $\mathcal{S}$. Let $n1(s)$ denote the number of 1's in the binary string s. Let $P(i,s)$ denote the position (count from right) of the ith 1 from the left end of s; if $i > n1(s)$ then $P(i,s) = 0$.

Definition 2 *For $x_1, x_2 \in \mathcal{S}$, where $x_1 = (i_1, s_1)$, $x_2 = (i_2, s_2)$, x_1 dominates x_2 ($x_1 \succeq x_2$) if the following three properties are satisfied.*

1. $i_1 \geq i_2$,
2. $n1(s_1) \geq n1(s_2)$,
3. $P(i,s_1) \leq P(i,s_2), i = 1, \cdots, n1(s_2)$ *whenever* $n1(s_2) > 0$.

We introduce three transformations on $\mathcal{S}$,

1. ψ_λ operates as follows

$$\psi_\lambda((i,\wedge)) = (i+1,\wedge), \quad 1 \leq i < c,$$
$$\psi_\lambda((c,s)) = (c,s1), \quad s \in \mathcal{S},$$

2. $\psi_{\mu,j}$ operates as follows, $j = 1, 2, \cdots, c$

$$\psi_{\mu,j}((i,\wedge)) = \begin{cases} (i-1,\wedge), & 1 \leq j \leq i \leq c, \\ (i,\wedge), & 0 \leq i < j \leq c, \end{cases}$$

$$\psi_{\mu,j}((c,10^i)) = (c,\wedge), \quad 0 \leq i,$$
$$\psi_{\mu,j}((c,s10^i)) = (c,s0^{i+1}), \quad 0 \leq i, \ \ s \in 1(0+1)^*.$$

3. $\psi_{\alpha,j}$ operates as follows, $j = 1, 2, \cdots$

$$\psi_{\alpha,j}((i,s)) = (i,s), \quad n(s) < j,$$
$$\psi_{\alpha,j}((i,s_1 1^k 0 s_2)) = (i,s_1 1^k s_2), \quad s_1 \in 1(0+1)^*, \ s_2 \in (0+1)^*, \ n(1^k 0 s_2) = j, \ 0 \leq k,$$
$$\psi_{\alpha,j}((i,s1^j)) = (i,s1^{j-1}).$$

The first step towards proving the ordering between the three Markov chains is the following Lemma.

Lemma 1 *If $x_1 \preceq x_2$, $x_1, x_2 \in S$, then*

$$\psi_\lambda(x_1) \preceq \psi_\lambda(x_2), \tag{2}$$

$$\psi_{\mu,j}(x_1) \preceq \psi_{\mu,j}(x_2), \quad 1 \le j \le c, \tag{3}$$

$$\psi_{\alpha,j}(x_1) \preceq \psi_{\alpha,j}(x_2), \quad 1 \le j. \tag{4}$$

Proof. Throughout the proof we will use the notation $x_1 = (i_1, s_1)$, $x_2 = (i_2, s_2)$, $\psi(x'_j) = (i'_j, s'_j)$, $j = 1, 2$, $d = n1(s_1) - n1(s_2)$, and $d' = n1(s'_1) - n1(s'_2)$. We will establish the three properties of dominance in all three cases.

Equation (2). Consider ψ_λ.

1. $i'_1 = \min(c, i_1 + 1) \le \min(c, i_2 + 1) = i'_2$

2. $d' = d + \chi(i_1 = c) - \chi(i_2 = c) \ge d \ge 0$.[4] The first inequality is due to property 1 of the dominance relation between x_1 and x_2.

3. If $n1(s_2) = 0$, then we are done. If $n1(s_2) > 0$, then $P(i, s'_2) = P(i, s_2) + 1 \le P(i, s_1) + 1 = P(i, s'_1)$.

Equation (3). Consider $\psi_{\mu,j}$, $1 \le j \le c$.

1. $i'_1 = i_1 - \chi(s_1 = \wedge \text{ and } j \le i_1) \ge i_2 - \chi(s_1 = \wedge \text{ and } j \le i_1) \ge i_2 - \chi(s_2 = \wedge \text{ and } j \le i_2) = i'_2$.

2. There are two cases
a) $d > 0$: $d' \ge d - 1 \ge 0$.
b) $d = 0$: $d' = d \ge 0$.

3. If $n1(s_2) = 0$, then we are done. If $n1(s_2) > 0$, then

$$P(i, s'_2) = P(i, s_2) - 1 \le P(i, s_1) - 1 \le P(i, s'_1).$$

Equation(4). Consider $\psi_{\alpha,j}$, $j = 1, 2, \cdots$.

[4] Here $\chi(P)$ takes value 1 if the predicate P is true and 0 otherwise.

1. $i'_1 = i_1 \geq i_2 = i'_2$.

2. There are two cases.
a) If $n1(s_1) > n1(s_2)$ then it follows that $n1(s'_1) \geq n1(s'_2)$.
b) If $n1(s_1) = n1(s_2)$ then because of property 3 of dominance, it follows that $n1(s'_1) \geq n1(s'_2)$.

3. Notice that in any case, we have $P(i,s) \geq P(i,s') \geq P(i,s) - 1$. There are two cases.
a) If a 1 is deleted from s_2, then we can assume that this is the right most bit in s_1 and therefore we have

$$P(i,s'_1) \geq P(i,s_1) - 1 = P(i,s_2) - 1 = P(i,s'_2).$$

b) If a 0 is deleted from s_2, then because $P(1,s_1) \geq P(1,s_2)$ and the first bit is always a 1, some bit is also deleted from s_1. There are two subcases.
i) A 1 is deleted from s_1. In this case there is no 0 to the right of this bit in s_1 and we can assume that the 0 and the 1 are both deleted from the k-th position from the right in s_2 and s_1 respectively. Then when $P(i,s_2) > k$ we have $P(i,s'_1) = P(i,s_1) - 1 \geq P(i,s_2) - 1 = P(i,s'_2)$. When $P(i,s_2) < k$, we have that $P(i,s'_1) = P(i,s_1) \geq P(i,s_2) = P(i,s'_2)$.
ii) A 0 is deleted from s_1. Assume that $P(i,s'_1) < P(i,s'_2)$ for some i, and this i is the smallest index for which the inequality holds. This can only occur if $P(i,s'_1) + 1 = P(i,s_1) = P(i,s_2) = P(i,s'_2)$, a 0 is deleted at position $k_1 = P(i,s_1) - 1$ in s_1, a 0 is deleted at position $k_2 \geq P(i,s_1)$ in s_2 and there is a 1 at position k_1 in s_2. This leads to $P(i+1,s_2) = k_1 > P(i+1,s_1)$ which is a contradiction.
c) If nothing is deleted from s_2, it is trivial to establish condition 3.

This completes the proof of the lemma. □

We define two additional transformations $\psi_{\mu_n,j}$, $1 \leq j \leq c$, and ψ_{λ_n} as follows:

1. $\psi_{\mu_n,j}$ operates as follows,

$$\psi_{\mu_n,j}((i,s)) = \psi_{\mu,j}((i,s)), \quad n1(s) \leq n,$$

$$\psi_{\mu_n,j}((c,s_1 1 s_2)) = (c,s_1 0 s_2), \quad s_1 \in 1(0+1)^*,\ s_2 \in (0+1)^*,\ N1(s_1) = n-1.$$

2. ψ_{λ_n} operates as follows,

$$\psi_{\lambda_n}((i,s)) = \psi_{\lambda}((i,s)), \quad n1(s) \leq n,$$

$$\psi_{\lambda_n}((c,s_1 1 s_2)) = (c,s_1 11 s_2), \quad s_1 \in 1(0+1)^*,\ s_2 \in (0+1)^*,\ n1(s_1) = n-1.$$

These transformations correspond to a completion event in $\mathcal{M}^-(n)$ and an arrival event in $\mathcal{M}^+(n)$ respectively. From this definition we have the following Lemma,

Lemma 2 *For any $x \in S$, we have*

$$\psi_{\mu_n,j}(x) \preceq \psi_{\mu_{n+1},j}(x) \preceq \psi_{\mu,j}(x), \quad n \geq 1$$

and

$$\psi_{\lambda}(x) \preceq \psi_{\lambda_{n+1}}(x) \preceq \psi_{\lambda_n}(x), \quad n \geq 1.$$

Proof: The proof follows from the definitions of $\psi_{\mu_n,j}$ and ψ_{λ_n} using similar arguments to those used to prove lemma 1. □

A consequence of the two preceding Lemmas is

Theorem 3 *The following orderings exist between $\mathcal{M}$, $\mathcal{M}^-(n)$, and $\mathcal{M}^+(n)$, $n = 1, 2, \cdots$,*

$$I^+(t,n) \geq_{st} I^+(t,n+1), \qquad n = 1, 2, \cdots,$$

$$I^+(t,n) \geq_{st} I(t) \geq_{st} I^-(t,n) \qquad n = 1, 2, \cdots,$$

$$I^-(t,n+1) \geq_{st} I^-(t,n), \qquad n = 1, 2, \cdots,$$

for $t > 0$ provided that the above orderings hold at $t = 0$, and

$$I^+(n) \geq_{st} I^+(n+1), \qquad n = 1, 2, \cdots,$$

$$I^+(n) \geq_{st} I \geq_{st} I^-(n) \qquad n = 1, 2, \cdots,$$

$$I^-(n+1) \geq_{st} I^-(n), \qquad n = 1, 2, \cdots.$$

Proof. The proof of the first three relations is by induction on the times at which arrivals, service completions or deadline misses occur using the results of lemmas 1 and 2. Once the first three relations are established, the second set of relations between the limiting values of I, $I^-(n)$, and $I^+(n)$ follow from Stoyan [13, Proposition 1.2.3, p. 6]. □

Corollary 1 *The following ordering exists between $\mathcal{M}$, $\mathcal{M}^-(n)$, and $\mathcal{M}^+(n)$, $n = 1, \cdots$,*

$$Q^+(n) \leq Q^+(n+1), \quad n = 1, 2, \cdots,$$

$$Q^+(n) \leq Q \leq Q^-(n), \quad n = 1, 2, \cdots,$$

$$Q^-(n+1) \leq Q^-(n), \quad n = 1, 2, \cdots.$$

Proof. All three chains are ergodic; consequently stationary distributions exist for each of them. The probabilities that a job misses its deadline in the three systems are expressed as

$$Q = 1 - \frac{\sum_{i=1}^{c} \Pr[I = i] i\mu}{\lambda},$$

$$Q^-(n) = 1 - \frac{\sum_{i=1}^{c} \Pr[I^-(n) = i] i\mu}{\lambda},$$

$$Q^+(n) = 1 - \frac{\sum_{i=1}^{c} \Pr[I^+(n) = i] i\mu}{\lambda}.$$

As a consequence of Theorem 3 we have

$$\sum_{i=1}^{c} \Pr[I^+(n) = i] \geq \sum_{i=1}^{c} \Pr[I = i] \geq \sum_{i=1}^{c} \Pr[I^-(n) = i], \qquad n = 1, 2, \cdots.$$

This establishes the corollary. □

Remark. With a little bit more effort, it is possible to establish *strict inequalities* among the miss probabilities in the Corrolary 1.

3 The Earliest Deadline Scheduling Policy

In this section we define the policy ED and its Markov chain $\mathcal{M}_{ED}$. We then modify this Markov chain to obtain two new MC's $\mathcal{M}^-_{ED}(n)$ and $\mathcal{M}^+_{ED}(n)$ that provide pessimistic and optimistic bounds on the fraction of jobs that miss their deadlines under ED. For the case that ED operates on system with a single processor, its behavior is remarkably similar to that of ML operating on a single processor. Hence, all of the development in the previous section applies with little modification to ED. This is unfortunately not true when the number of processors is greater than one.

3.1 The Earliest Deadline Policy and its Markov Chain Representation

We consider one processor serving jobs from a single queue. Jobs arrive according to a Poisson process with parameter λ. Each job requires a service time that is exponentially distributed with mean $1/\mu$. In addition, there is a deadline associated with each job. A job is thrown away if it does not complete execution before its deadline. This can occur while the job is in the queue or while it is in service. If the job is in the queue at the time that the deadline is reached, the job is thrown out. If the job is in service at the time that the job reaches its deadline, it is aborted and then thrown out. In either case the job is considered lost from the system. We denote the interval of time between the arrival of a job and its deadline as its *relative deadline* and we assume that they are exponentially distributed r.v.'s with parameter α.

Panwar and Towsley [10] have proven that the earliest deadline scheduling policy (ED) maximizes the fraction of jobs that complete service by their deadlines from the class of all policies that use no service time information, allow preemptions, and never allow processors to go idle while there is work in the queue. ED is a preemptive policy that always executes the job closest to its deadline.

The behavior of the system is described by a continuous time Markov chain, $\mathcal{M}_{ED}$, having state $S_{ED}(t) \in \mathcal{B}$. Here $S_{ED}(t)$ has the same interpretation as it does in $\mathcal{M}$ except that it now includes the job *in service*.

The Markov process $\mathcal{M}$ exhibits the following behavior.

Arrival An arrival has the effect of adding a 1 to the right of $S_{ED}(t)$. The transitions are illustrated below:

$$s \xrightarrow{\lambda} s1, \quad s \in \mathcal{B}.$$

Service completion A departure has the effect of changing the rightmost 1 in $S_{ED}(t)$ to a 0. The transitions are illustrated below:

$$s10^n \xrightarrow{\mu} s0^{n+1}, \quad s \in \mathcal{B} - \{\wedge\},\ 0 \leq n.$$
$$10^n \xrightarrow{\mu} \wedge.$$

Deadline Expiration Transitions due to the expiration of deadlines occur only when $s \neq \wedge$. Consider $s = 1^{m_1}0^{n_1}1^{m_2}0^{n_2}\cdots 1^{m_g}0^{n_g}$. For $1 \leq j \leq g-1$, one of the n_j

0's is deleted if the deadline of any of the $m_j + n_j$ jobs within the j-th group expires. This occurs with rate $(m_j + n_j)\alpha$. This also occurs to the g-th group provided $n_g > 0$. If $n_g = 0$ and the deadline of one of the m_g jobs in the g-th group expires, then a 1 is deleted from that group. In either case this occurs with rate $(m_g + n_g)\alpha$. The transitions are illustrated below:

$$s_1 1^n 0^m s_2 \xrightarrow{(n+m)\alpha} s_1, 1^n 0^{m-1} s_2, \quad s_1 \in \{\wedge\} \cup 1(0,1)^* 0,\ s_2 \in \mathcal{B},\ 1 \leq m, n,$$
$$s 1^n \xrightarrow{n\alpha} s 1^{n-1}, \quad s \in \{\wedge\} \cup 1(0,1)^* 0.$$

We define the limiting state $S_{ED} = \lim_{t\to\infty} S_{ED}(t)$ and let $\pi_{ED}(s) = \Pr[S_{ED} = s]$, $s \in \mathcal{B}$. These probabilities exist whenever $\lambda > 0$, $\mu > 0$, $\alpha > 0$. The equations are similar to those for ML and are not included.

The Markov chain $\mathcal{M}$ can be shown to simulate ED correctly, [7], and the fraction of customers lost, Q_{ED}, can be computed from the stationary distribution by the following expression

$$Q_{ED} = 1 - \frac{(1 - \pi_{ED}(\wedge))\mu}{\lambda}. \tag{5}$$

3.2 The Policy $ED(n)$ and the Markov Chain $\mathcal{M}^-_{ED}(n)$

We now describe an implementable scheduling policy, $ED(n)$, that is simpler to analyze than ED and provides an upper bound on the loss probability for ED.

We divide the queue into two parts, Q_1 and Q_2. Q_1 can hold at most n jobs including the job in service. If the total number of jobs waiting for service is less than or equal to n, then they are placed into Q_1. Jobs in Q_1 are scheduled using ED. However if there are more than n jobs in the queue, we place new incoming jobs into Q_2. When a job completes service and leaves Q_1, we move a job from Q_2 to Q_1 in the order of arrival. In short, Q_1 is a ED queue of size n and Q_2 is a FIFO queue of unbounded size.

Accordingly, we define a new Markov process $\mathcal{M}^-_{ED}(n)$ with state $S^-_{ED}(t,n)$ where $S^-_{ED}(t,n)$ is the partial state history at time $t > 0$. The state space is $\mathcal{B}^-(n)$ (as defined in section 2.2). State transitions obey the following rules.

Arrival The same as in $\mathcal{M}_{ED}$

Service Completion The behavior is the same as in $\mathcal{M}_{ED}$ if the total number of 1's in s is less than n. If the total number of 1's is greater than or equal to n then the n-th 1 from the left is converted into a 0.

Deadline Expiration The same as in $\mathcal{M}_{ED}$

These rules generate a Markov chain which can be solved to obtain the loss probability under $ED(n)$.

3.3 The Markov Chain $\mathcal{M}^{+}_{ED}$

We define another continuous time Markov chain $\mathcal{M}^{+}_{ED}$ to have a state $S^{+}_{ED}(t,n)$ taken from the set $\mathcal{B}^{+}(n+1)$. Transitions between the states are governed by the following three rules.

Arrival If the number of 1's in s is less than or equal to n, then append a 1 to the right end of s. If the number of 1's in s exceeds n, then insert a 1 after the n-th 1 in s counting from the left.

Service Completion Same as $\mathcal{M}_{ED}$.

Deadline Expiration Same as $\mathcal{M}_{ED}$

This does not correspond to any implementable policy but leads to a lowerbound, $Q^{+}_{ED}(n)$, on the fraction of jobs that miss their deadlines.

We state without proof the following result regarding the ordering between the Markov chains $\mathcal{M}_{ED}$, $\mathcal{M}^{-}_{ED}(n)$, and $\mathcal{M}^{+}_{ED}(n)$.

Theorem 4 *The following ordering exists between $\mathcal{M}_{ED}$, $\mathcal{M}^{-}_{ED}(n)$, and $\mathcal{M}^{+}_{ED}(n)$, $n = 1,\cdots,$*

$$Q^{+}_{ED}(n) \leq Q^{+}_{ED}(n+1), \quad n = 1,2,\cdots,$$

$$Q^{+}_{ED}(n) \leq Q \leq Q^{-}_{ED}(n), \quad n = 1,2,\cdots,$$

$$Q^{-}_{ED}(n+1) \leq Q^{-}_{ED}(n), \quad n = 1,2,\cdots.$$

4 Numerical Results

In this section we study the tightness of the bounds on Q generated by $ML(2)$, $ML(3)$ and $\mathcal{M}^+(2)$ and the bounds on Q_{ED} generated by $ED(3)$ and $\mathcal{M}^+_{ED}(2)$. We also examine the efficiency of FCFS, $ML(2)$, and $ML(3)$ for scheduling customers with laxities and FCFS and $ED(3)$ for scheduling customers with deadlines to end of service. However, first we wish to comment on the computational requirements for obtaining the bounds.

Consider the Markov chain $\mathcal{M}^-(n)$. The state of this chain can be represented by the $n-$tuple $(u_1, \cdots, u_n)$ where u_i represents the number of 0's between the i-th and $(i+1)$-th 1 in $S^-(n)$, $i = 1, \cdots n-1$ and u_n represents the number of jobs in the FIFO portion of the queue. These variables can take on any nonnegative integer value. Hence, we truncated the state space in order to obtain estimates for $Q^-(n)$. In the case of $ML(2)$ and $ML(3)$ we are left with two and three dimensional finite state Markov chains respectively. In all cases, it sufficed to delete all states with values of u_i greater than 20. A similar procedure was used in order to obtain optimistic bounds from $\mathcal{M}^+(2)$ and for obtaining the bounds for the performance of ED.

4.1 The Performance of ML

In general, we find the bounds to be the closest when deadlines are tight and to widen as deadlines increase in length. In addition, they are also close for small arrival rates and tend to widen as the job arrival rate increases. Table 1 illustrates this behavior by comparing the pessimistic bounds on Q generated by $ML(1)$ (FIFO), $ML(2)$, and $ML(3)$, the upper bound on Q generated by $\mathcal{M}^+(2)$, and Q obtained from simulation for the case of a single processor. Results are given for different values of the arrival rate λ and mean laxity $1/\alpha$. In all cases, the times are normalized with respect to the mean service time $1/\mu$.

We observe that the natural estimate for Q, $(Q^-(3)+Q^+(2))/2$ never deviates from the estimate obtained by simulation by more than 3.1%. Typically, the error is considerably less than that when either $1/\alpha = 2, 4$ or $\lambda \leq .6$. We have also observed that the bounds become closer as the number of processors increases (data not shown here).

Figures 1 - 3 illustrate the performance of FIFO, $ML(3)$ and ML for different numbers of processors $c = 1, 4, 8$, two different mean laxities, $1/\alpha = 2, 8$ as the traffic load

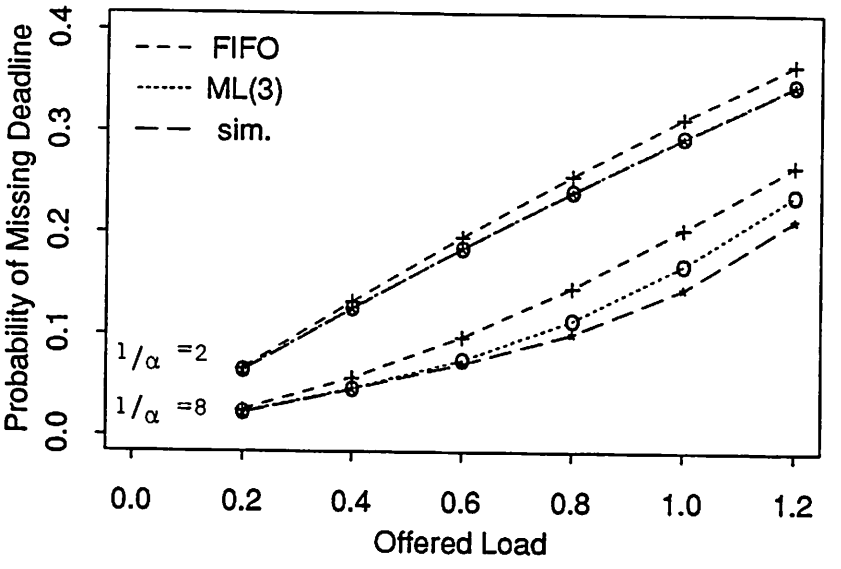

Figure 1: Comparison of ML, $ML(3)$, and FIFO on One Processor.

increases. In these figures, all times are normalized with respect to c/μ.

We make the following observations. When the laxities are small, the difference between the performance of ML and $ML(3)$ is negligible. However, there can be up to a 5.7% increase in the fraction of jobs lost when FIFO is used. When the laxities increase a difference appears between $ML(3)$ and ML. However, the difference between $ML(3)$ and FIFO also increases (11% difference in fraction of jobs missing deadline in the case of $1/\alpha = 8$, $\lambda = 1.2$ and $c = 1$). Hence by using a little bit of information, namely, the laxities of the first three jobs in the queue, $ML(3)$ provides a performance improvement over policies such as FIFO that use no laxity information without incurring the overhead of ML. Briefly, the cost of inserting/removing elements from the queue under ML is either $O(m)$ (using a queue) or $O(\log m)$ (using a heap) where m is the number of jobs in the queue whereas the cost of inserting/removing from the queue under $ML(3)$ is $O(1)$.

4.2 Performance of ED

In this subsection we examine the accuracy of the bounds for the ED policy operating on a single processor. Figure 4 illustrates the behavior of $ED(1)$ (FIFO), $ED(3)$ and the optimistic bound based on $\mathcal{M}^{+}_{ED}(2)$, as a function of the offered traffic for three

λ	$1/\alpha$	$ML(1)$ (FCFS)	$ML(2)$	$ML(3)$	Simulation	Approx. $(Q^-(3)+Q^+(2))/2$	$\mathcal{M}^+(2)$
0.2	2	.0665	.0648	.0646	.0639	.0646	.0646
0.4	2	.1319	.1271	.1257	.1253	.1256	.1256
0.6	2	.1954	.1877	.1840	.1838	.1838	.1837
0.8	2	.2559	.2465	.2401	.2394	.2397	.2394
1.0	2	.3130	.3032	.2942	.2935	.2937	.2932
1.2	2	.3662	.3569	.3463	.3453	.3455	.3448
0.2	4	.0427	.0405	.0399	.0397	.0399	.0399
0.4	4	.0905	.0840	.0805	.0800	.0802	.0800
0.6	4	.1429	.1327	.1235	.1224	.1224	.1213
0.8	4	.1988	.1869	.1707	.1675	.1680	.1654
1.0	4	.2564	.2450	.2238	.2178	.2193	.2148
1.2	4	.3141	.3046	.2822	.2747	.2767	.2713
0.2	8	.0251	.0231	.0225	.0222	.0224	.0224
0.4	8	.0570	.0510	.0463	.0453	.0456	.0450
0.6	8	.0971	.0877	.0748	.0707	.0711	.0675
0.8	8	.1460	.1355	.1136	.1006	.1030	.0925
1.0	8	.2033	.1943	.1679	.1445	.1503	.1327
1.2	8	.2665	.2604	.2374	.2122	.2188	.2002

Table 1: Comparison of Bounds and Simulation for one Processor.

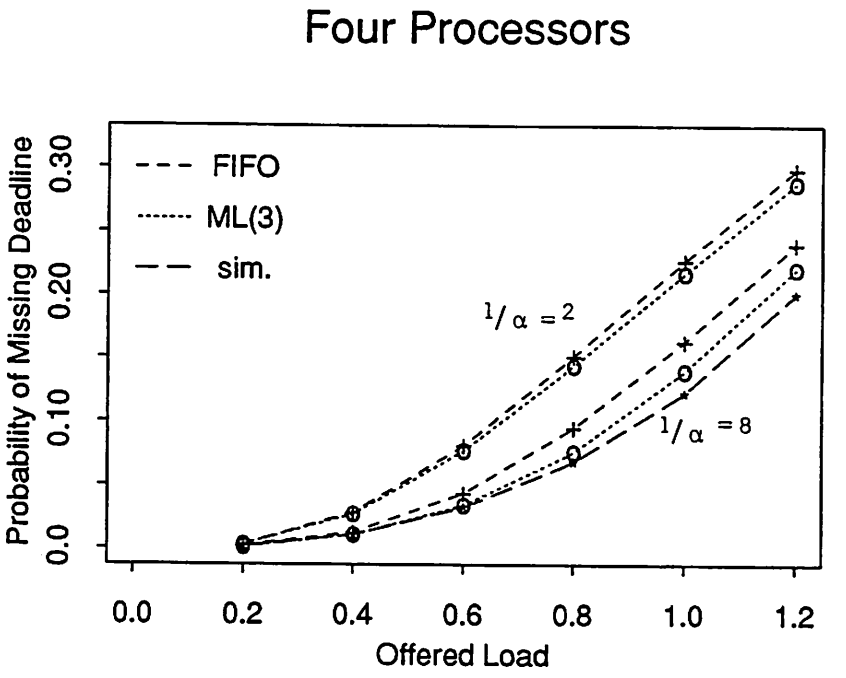

Figure 2: Comparison of ML, $ML(3)$, and FIFO on Four Processors.

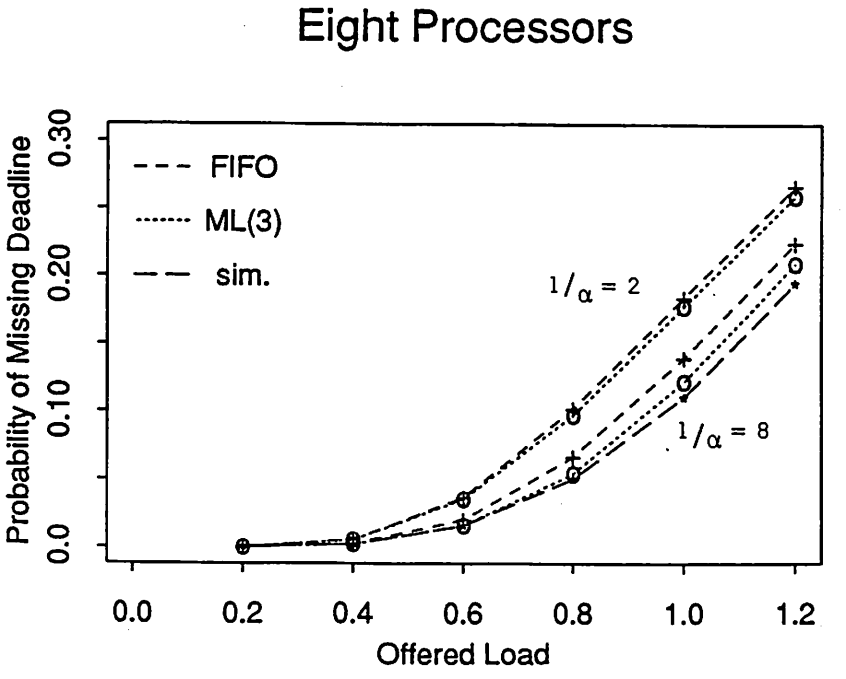

Figure 3: Comparison of ML, $ML(3)$, and FIFO on Eight Processors.

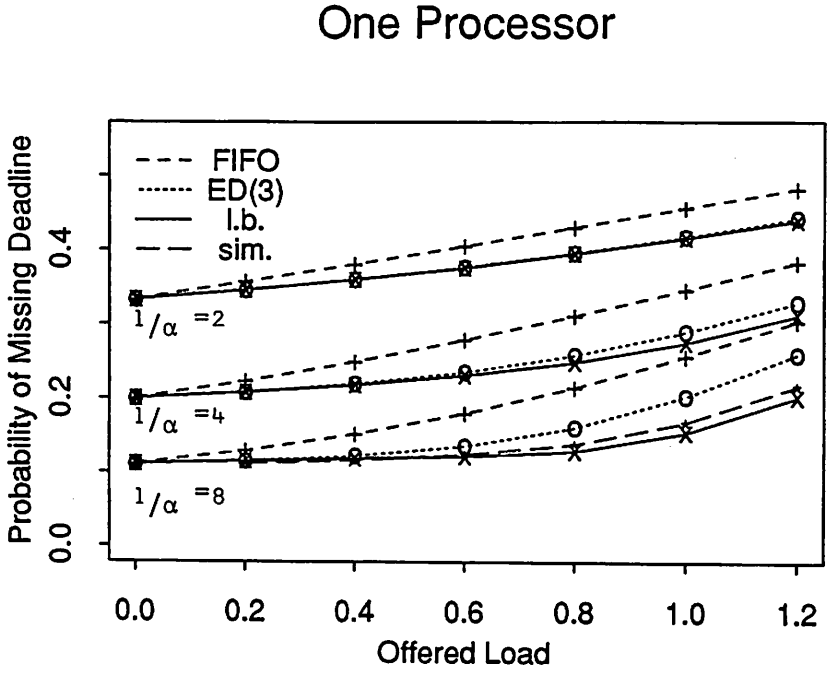

Figure 4: Accuracy of Bounds and Comparison of FIFO to $ED(3)$.

different mean deadline intervals, $1/\alpha = 2, 4, 8$. Also included are loss probabilities obtained from simulation for the case $1/\alpha = 8$ and high traffic loads. We first observe that for low traffic loads, the fraction of jobs that miss their deadlines can be expressed as $\alpha/(\alpha + 1)$. We also observe that the bounds are not as close to each other for this system as they were for ML. For example, using the average of the bounds as an estimate for Q_{ED} may result in an error of as much as 15%. when $1/\alpha = 8$ and the arrival rate exceeds 0.8. The errors are significantly less for other reported parameter values.

We also observe a significant difference between FIFO and $ED(3)$. $ED(3)$ can decrease the fraction of jobs that miss their deadlines by as much as 15% at very little increase in overhead.

5 Summary

In this paper we considered the problem of analyzing ML and ED for systems in which jobs have real-time constraints. This is important because these two policies are known to be optimal under a large class of workload assumptions. We observed that the Markov chains associated with these policies are not amenable to exact analysis. Hence we modified the Markov chains so as to develop two families of Markov chains which

can be shown to provide bounds on the performance of ML and ED. The resulting Markov chains were easier to solve numerically.

In addition, the Markov chains that produce the pessimistic bounds correspond to implementable families of policies $ML(n)$ and $ED(n)$ where n corresponds to the maximum number of jobs at the head of the queue that are scheduled according to the ML and ED rules. The remaining jobs are treated in a FIFO manner. We observed that $ML(3)$ and $ED(3)$ can provide a 5% to 15% improvement in performance over a policy such as FIFO that does not use any laxity or deadline information.

Several generalizations of this work appear straightforward. For example, it should be possible to conduct a similar analysis of ML for the M/G/1+M system by studying the system behavior at departure instances and for the G/M/1+M system by studying the behavior at arrival instances.

The work reported in this paper does not completely solve the problem of estimating the performance of ML and ED through analytic means. As we have observed, the distance between the bounds on the fraction of jobs missing their deadlines increases as the mean laxity and mean relative deadline increases. Hence work remains to be done to develop tighter bounds for this region of the workload parameter space.

Appendix

We list the equations that must be solved by the stationary distribution, $\pi(i,s)$, $1 \leq c$, $s \in \mathcal{B}$, for ML.

$$\lambda\pi(0,\wedge) = \mu\pi(1,\wedge), \tag{6}$$

$$(\lambda + i\mu)\pi(i,\wedge) = \lambda\pi(i-1,\wedge) + (i+1)\mu\pi(i+1,\wedge), \quad 1 \leq i < c, \tag{7}$$

$$(\lambda + c\mu)\pi(c,\wedge) = \lambda\pi(c-1,\wedge) + c\mu\sum_{j=0}^{\infty}\pi(c,10^j) + \alpha\pi(c,1), \tag{8}$$

$$\begin{aligned}
&\left(\lambda + c\mu + \alpha\sum_{j=1}^{g} m_j + \sum_{j=1}^{g-1} n_j\right)\pi(c,1^{m_1}0^{n_1}\cdots 1^{m_{g-1}}0^{n_{g-1}}1^{m_g}) = \\
&\lambda\pi(c,1^{m_1}0^{n_1}\cdots 1^{m_{g-1}}0^{n_{g-1}}1^{m_g-1}) \\
&+\alpha\sum_{k=1}^{g-1}(m_k+n_k+1)\pi(1^{m_1}0^{n_1}\cdots 1^{m_k}0^{n_k+1}\cdots 1^{m_{g-1}}0^{n_{g-1}}1^{m_g}) \\
&+\alpha\sum_{k,m_k>1}\sum_{l=1}^{m_k-1}(l+1)\pi(1^{m_1}0^{n_1}\cdots 0^{n_{k-1}}1^l01^{m_k-l}0^{n_k}\cdots 1^{m_{g-1}}0^{n_{g-1}}1^{m_g}), \\
&\quad m_j, n_j > 0, 1 \leq j \leq g-1,\ m_g > 0,\ 1 \leq g,
\end{aligned} \tag{9}$$

$$\begin{aligned}
&\left(\lambda + c\mu + \alpha\sum_{j=1}^{g}(m_j+n_j)\right)\pi(c,1^{m_1}0^{n_1}\cdots 1^{m_g}0^{n_g}) = \\
&+ c\mu\pi(1^{m_1}0^{n_1}\cdots 1^{m_g}0^{n_g-1}1) + c\mu\pi(1^{m_1}0^{n_1}\cdots 1^{m_g+1}0^{n_g-1}) \\
&+\alpha\sum_{k=1}^{g}(m_k+n_k+1)\pi(1^{m_1}0^{n_1}\cdots 1^{m_k}0^{n_k+1}\cdots 1^{m_g}0^{n_g}) \\
&+\alpha\sum_{k,m_k>1}\sum_{l=1}^{m_k-1}(l+1)\pi(1^{m_1}0^{n_1}\cdots 0^{n_{k-1}}1^l01^{m_k-l}0^{n_k}\cdots 1^{m_g}0^{n_g}), \\
&\quad m_j, n_l > 0, 1 \leq j \leq g,\ 1 \leq g.
\end{aligned} \tag{10}$$

References

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

[2] D.Y. Barrer, "Queueing with Impatient Customers and Indifferent Clerks", *Operations Research 5*, pp. 644-649, 1957.

[3] D.Y. Barrer, "Queueing with Impatient Customers and Ordered Service", *Operations Research 5*, pp. 650-656, 1957.

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

[5] B. Gavish, P. Schweitzer, "The Markovian Queue with Bounded Waiting Time", *Management Sci. 23*, 12, pp. 1349-1357, Aug. 1977.

[6] R.B. Haugen, E. Skogan, "Queueing Systems with Stochastic Timeout", *IEEE Trans. on Communications 28*, 12, pp. 1984-1989, Dec. 1980.

[7] J. Hong, X. Tan, and D. Towsley, "The Binary Simulation of the Minimum Laxity and Earliest Deadline Scheduling Policies for Real-Time Systems", COINS Technical Report 89-70, Dept. of Computer and Information Science, University of Massachusetts, July 1989.

[8] J.F. Kurose, R. Chipalkatti, "Load Sharing in Soft Real-Time Distributed Computer Systems", *IEEE Trans. Comp. 36*, 8, Aug. 1987.

[9] S.S. Panwar, D. Towsley, 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, October 1988.

[10] S.S. Panwar, D. Towsley, "On the Optimality of the STE Rule for Multiple Server Queues that Serve Customers with Deadlines", COINS Technical Report 88-81, Dept. of Computer and Information Science, University of Massachusetts, July 1988.

[11] R.E. Stanford, "Reneging Phenomena in Single Channel Queues", *Mathematics of Operations Research 4*, 2, pp. 162-178, May 1979.

[12] J.A. Stankovic, K. Ramamritham, eds., "Tutorial: Hard Real-Time Systems", Computer Society Press of the IEEE, 1988.

[13] D. Stoyan, *Comparison Methods for Queues and Other Stochastic Models*, John Wiley & Sons, 1983.

[14] W. Zhao, J.A. Stankovic, "Performance Analysis of FCFS and Improved FCFS Scheduling Algorithms for Dynamic Real-Time Computer Systems", submitted to the 1989 Real-Time Systems Symposium.