Fair Lateness Scheduling: Reducing Maximum Lateness in G-EDF-like Scheduling

Jeremy P. Erickson and James H. Anderson

University of North Carolina at Chapel Hill

Abstract

Existing research in soft real-time scheduling has focused on determining tardiness bounds given a scheduling algorithm. In this paper, we study lateness bounds, which are related to tardiness bounds, and propose a scheduling algorithm to minimize lateness bounds, namely the global fair lateness (G-FL) algorithm. G-FL is a G-EDF-like scheduler, but has lower maximum lateness bounds than G-EDF. Due to its G-EDF-like nature, it can be used within existing systems that implement arbitrary-deadline G-EDF, and with existing synchronization protocols. Therefore, we argue that G-FL should replace G-EDF for SRT applications.

1 Introduction

Analysis-based soft real-time (SRT) schedulers have been demonstrated to be useful on multiprocessor systems when bounded deadline tardiness is acceptable [1]. Previous work on bounded tardiness, such as [2, 3], has provided tardiness bounds for specific schedulers. For many applications, such as the video processing described in [1], the output of SRT tasks can be stored in a buffer, and the buffer read at the desired rate to simulate HRT completion. The buffer size is determined from tardiness bounds.

If tardiness bounds can be reduced, smaller buffer sizes will be adequate to provide equivalent performance. These smaller buffer sizes can reduce the cost of the system, and may enable more applications to run on resource-constrained devices such as smartphones and tablets.

Past Work. In [4], Leontyev and Anderson provide general analysis for SRT scheduling. They observed that scheduling priorities for most algorithms can be modeled by giving each job a *priority point* (*PP*) in time, with the scheduler always picking the job with the earliest PP (with appropriate tie-breaking). For example, fixed-priority scheduling can be modeled by assigning all jobs of each task a single PP near the beginning of the schedule. *Global earliest deadline first* (*G-EDF*) can be modeled by defining the absolute deadline of a job as its PP. The authors defined a class of *window-constrained* algorithms that provide bounded tardiness with no utilization loss. In [5], response times are analyzed for many schedulers, including a class of *G-EDF-like* (*GEL*) schedulers, in which the PP of each job is defined as some per-task constant after the job's release. Although these papers provide analysis for a large class of scheduling algorithms, they do not provide a method to choose the best scheduling algorithm to meet particular requirements.

One choice of algorithm that has been studied is G-EDF itself. Although G-EDF is known to be suboptimal on multiprocessors, it is attractive for several reasons. Optimal algorithms, such as those described in [6, 7, 8], either cause tasks to experience frequent preemptions and migrations, resulting in prohibitive overheads, or are difficult to implement in practice. In contrast, experimental research has demonstrated that the overheads caused by G-EDF are reasonable when it is used on a moderate number of processors [9]. Furthermore, unlike optimal algorithms, G-EDF has the desirable property that it is *job-level static-priority* (*JLSP*). The JLSP property is required for most known work on real-time synchronization algorithms (see [10]). Several promising non-optimal schedulers such as the earliest-deadline-until-zero-laxity (EDZL) algorithm [11, 12], as well as all optimal schedulers, are not JLSP and thus cannot be used with these synchronization algorithms.

In this work, we modify G-EDF to improve its tardiness bounds. In order to do so, we use the technique of *compliant-vector analysis* (*CVA*), first proposed in [13]. A summary of past work on CVA is provided in Sec. 3.1. Be-

cause some of that past work is applied to arbitrary-deadline sporadic task systems, it can be used to derive response time bounds for arbitrary GEL schedulers. Therefore, for easier reference, in Sec. 3.1 we present the existing analysis as it applies to arbitrary GEL schedulers. Although the same systems can be analyzed using the method described in [5], CVA can provide tighter bounds.

Our Contribution. In this paper, we demonstrate that CVA is useful for determining an appropriate choice of scheduler within the class of GEL schedulers. Any GEL scheduler is JLSP and has the same overheads as G-EDF with arbitrary deadlines. With the CVA method presented here, it is possible that jobs of some tasks can be guaranteed to complete by points *before* their deadlines. Therefore, instead of evaluating tardiness (which would be defined to be zero for such tasks), we evaluate *lateness* instead, defined as the difference between deadline and completion time.

We propose the *global fair lateness* (*G-FL*) scheduler, a GEL scheduler that provides the same lateness bound (under CVA) for all tasks. For each job, G-FL uses a PP that precedes its deadline. We demonstrate that under CVA, G-FL minimizes the maximum lateness bound over all tasks. Therefore, G-FL has several useful properties. If a task system can be proven HRT schedulable by any GEL algorithm under CVA, then it can be proven schedulable with G-FL. If the task system cannot be proven HRT schedulable, then G-FL will still provide a fair allocation of lateness bounds to all tasks. However, our work does not demonstrate that a better GEL scheduler cannot exist given better analysis, and does not imply that no *individual* task will have a lower CVA bound with a technique other than G-FL. Still, we believe that G-FL should be used as a replacement for G-EDF in SRT systems.

Related Work. Because our work involves shortening the deadline used by the scheduler, it superficially resembles the work of Lee et al. [14], in which the deadlines of some tasks are shortened at design time to create "contention-free slots" that allow the priorities of some jobs to be lowered during runtime, increasing system schedulability. However, in their work, the actual deadlines by which jobs must complete are altered, which is not true of our work. Furthermore, their work requires modifying G-EDF in a manner that adds additional runtime overhead and removes the JLSP property, while our work does not.

Organization. In Sec. 2, we describe the task model under consideration and define basic terms. In Sec. 3, we review the CVA provided in [3] and demonstrate for easier reference its applicability to arbitrary GEL schedulers. We also provide several observations about CVA that enable us to prove the desired results about G-FL. In Sec. 4, we present G-FL and prove that it has the desired properties. In Sec. 5, we present experiments comparing tardiness bounds and tardiness in computed schedules for G-FL compared to G-EDF. These experiments show that G-FL can provide significantly lower tardiness bounds and observed tardiness than G-EDF.

2 Task Model

We consider a system τ of *n arbitrary-deadline* sporadic tasks $\tau_i = (T_i, C_i, D_i)$ running on $m \geq 2$ processors, where T_i is the minimum separation time between subsequent releases of jobs of τ_i, $C_i \leq T_i$ is the worst-case execution time of any job of τ_i, and $D_i \geq 0$ is the relative deadline of each job of τ_i. We use $U_i = C_i/T_i$ to denote the *utilization* of τ_i. All quantities are real-valued. We assume that

$$\sum_{\tau_i \in \tau} U_i \leq m, \tag{1}$$

which was demonstrated in [4] to be a necessary condition for SRT schedulability. We assume that $n > m$. If this is not the case, then each task can be assigned its own processor, and no job of each τ_i will have response time exceeding C_i.

If a job has absolute deadline d and completes execution at time t, then its *lateness* is $t - d$, and its *tardiness* is $\max\{0, t - d\}$. If such a job was released at time r, then its *response time* is $t - r$. We bound these quantities on a per-task basis, i.e., for each τ_i, we consider upper bounds on these quantities that apply to all jobs of τ_i. The *max-lateness* bound for τ is the largest lateness bound for any $\tau_i \in \tau$. Similarly, the *max-tardiness* bound for τ is the largest tardiness bound for any $\tau_i \in \tau$.

We use for each τ_i the notation Y_i to refer to its relative PP, R_i to refer to its response time bound, and L_i to refer to its lateness bound.

For all variables subscripted with an i, we also use vector notation to refer to the set of all values for the task system. For example, $\vec{T} = \langle T_1, T_2, \ldots T_n \rangle$.

3 Compliant-Vector Analysis

In Sec. 3.1, we present results from prior work on CVA. In Sec. 3.2, we present new observations about CVA that we use in the proofs in Sec. 4. Our broad goal is to provide the necessary transformations to convert an arbitrary system into a G-FL system with a max-tardiness bound that is no larger.

3.1 Prior Work on CVA

In [3], CVA is presented for G-EDF with arbitrary deadlines. By using $\vec{Y}$, the set of relative PPs, in place of $\vec{D}$,

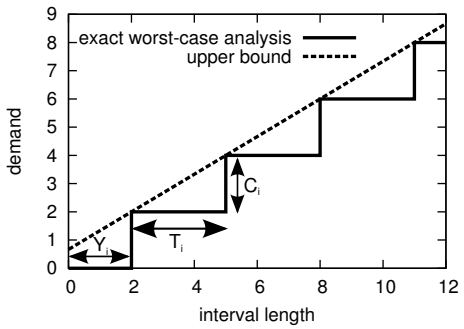


Figure 1: Example of exact analysis and linear upper bound for task maximum demand within an interval.

the relative deadlines, we can use the same existing analysis to analyze arbitrary GEL schedulers. In this subsection, we will make that small change, but otherwise we will simply review the existing work presented in [3]. That paper describes the computation of $\vec{x} = \langle x_1, x_2, \ldots, x_n \rangle$ such that the tardiness of task τ_i is at most $x_i + C_i$. Therefore, the response time bound R_i under G-EDF for any job of task τ_i is $D_i + x_i + C_i$. Substituting Y_i for D_i, we see that under an arbitrary GEL scheduler, each task τ_i has a response time bound

$$R_i = Y_i + x_i + C_i, \tag{2}$$

and a lateness bound

$$L_i = Y_i + x_i + C_i - D_i. \tag{3}$$

Because $D_i \geq 0$ was a necessary condition in [3], $Y_i \geq 0$ is a necessary condition when we apply our analysis. (If this condition is violated, we can increase each Y_i by some constant such that the condition is met, without changing any scheduling decisions.)

We will now briefly review the computation of $\vec{x}$ as provided in [3]. In that analysis, the tardiness of a particular job J of task τ_j is considered, with the inductive assumption that the response time bound in (2) holds for all jobs with higher priority than J. Only jobs with earlier PPs than J need to be considered. At several points in the proof, the amount of work that each task τ_i can demand over an interval of length t (having both release times and PPs within the interval) is upper-bounded by a linear function, as depicted in Fig. 1. We denote the y-intercept of this linear function (for a given Y_i)[1] with

$$S_i(Y_i) = C_i \cdot \max\left\{0, 1 - \frac{Y_i}{T_i}\right\}. \tag{4}$$

$S_i(Y_i)$ upper bounds the extra work demanded by τ_i beyond what would be expected from utilization alone. For the entire system, we define

$$S(\vec{Y}) = \sum_{\tau_i \in \tau} S_i(Y_i). \tag{5}$$

[1] In [3], Y_i was fixed at D_i, so this value was a constant. We have redefined it as a function to facilitate the choice of a GEL scheduling algorithm.

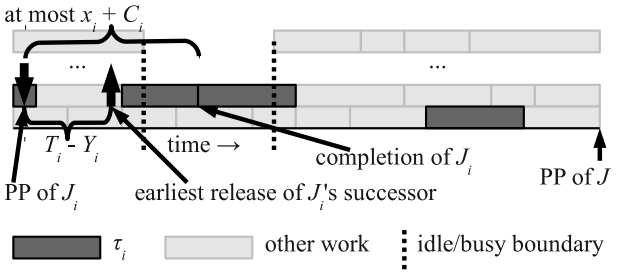


Figure 2: Task running through last idle interval.

As discussed in [3], we consider the work remaining at the PP of J. This work can occur not only due to tasks τ_i for which $Y_i < T_i$ (as accounted for using $S(\vec{Y})$), but also due to each task τ_i that is executing throughout the last idle interval before J's PP, as depicted in Fig. 2. We denote the job of each τ_i running at the beginning of that idle interval as J_i. Intuitively, J_i itself can have at most C_i units of backed-up work, and there can be an additional $x_i U_i - S_i(Y_i)$ units of backed-up work released by τ_i between J_i's release and completion. (This follows because (2) is assumed to inductively hold for higher-priority jobs. A more precise explanation is presented in [3].) Therefore, each such task contributes at most $x_i U_i + C_i - S_i(Y_i)$ units, so an upper bound, denoted as $G(\vec{x}, \vec{Y})$, on the total remaining work from all such tasks is given by

$$G(\vec{x}, \vec{Y}) = \sum_{(m-1) \text{ largest}} (x_i U_i + C_i - S_i(Y_i)). \tag{6}$$

As demonstrated in [3], the work remaining at J's priority point from all jobs is at most $G(\vec{x}, \vec{Y}) + S(\vec{Y})$, and the work from J itself is at most C_j. We pessimistically assume that J does not run at all before its PP. Because J will be allowed to execute at any moment after its PP when fewer than m processors are in use by competing work, the response time for J is at most

$$\frac{G(\vec{x}, \vec{Y}) + S(\vec{Y}) - C_j}{m} + C_j + Y_j. \tag{7}$$

(In this expression, the first term is due to the competing work after the PP of J, the C_j term indicates the maximum execution required by J, and the Y_j term appears because we assume that execution begins no earlier than the PP.) Therefore, in light of (2), to find the response time bounds, we compute $\vec{x}$ such that

$$\forall i, x_i = \frac{G(\vec{x}, \vec{Y}) + S(\vec{Y}) - C_i}{m}. \tag{8}$$

The vector satisfying (8) is called the *minimum compliant vector*. In [3], (8) is justified by the following theorem. We will not replicate the proof here.

Theorem 1. *There exists a unique minimum compliant vector $\vec{x}$ such that (8) holds and response time bounds are as in (2) iff*

$$\forall i, Y_i \geq 0. \tag{9}$$

Furthermore, no smaller response time bound is achievable using CVA for any task.

As [3] shows, the minimum compliant vector can be computed in polynomial time. The method to do so is based on several general principles that are also directly relevant to the present work, so we will call them "laws." The first follows immediately from (8).

Shape Law. *$\vec{x}$ is the minimum compliant vector iff there exists s such that*

$$\forall i, x_i = \frac{s - C_i}{m} \tag{10}$$

and

$$s = G(\vec{x}, \vec{Y}) + S(\vec{Y}). \tag{11}$$

Observe that s is independent of the task index i.

In [3], $\vec{Y}$ is fixed, and by (2), $\vec{x}$ must be computed in order to compute $\vec{R}$. In light of the Shape Law, we simply need to compute the necessary s. Towards this end, the next law is simply the Shape Law restated with additional notation. We use $\vec{v}(s)$ in place of $\vec{x}$. (We include "First" in the name of this law because a similar law will appear in the next subsection.)

First Restated Shape Law. *Suppose $\vec{Y}$ is fixed. Let $\vec{v}(s)$ be defined as*

$$\forall i, v_i(s) = \frac{s - C_i}{m}. \tag{12}$$

Let

$$M^Y(\vec{Y}, s) = G(\vec{v}(s), \vec{Y}) + S(\vec{Y}) - s. \tag{13}$$

Then, $\vec{v}(s)$ is the minimum compliant vector iff

$$M^Y(\vec{Y}, s) = 0. \tag{14}$$

Observe that, when using the notation of the First Restated Shape Law, (2) becomes

$$R_i = Y_i + v_i(s) + C_i. \tag{15}$$

Finally, because zeros of piecewise linear functions can be computed in polynomial time, our next law will allow the zero of $M^Y(\vec{Y}, s)$ to be computed in polynomial time.

M^Y Piecewise Linear Law. *$M^Y(\vec{Y}, s)$ is piecewise linear with respect to s.*

Proof.

$$\begin{aligned} M^Y(\vec{Y}, s) &= \{\text{By (13)}\} \\ &\quad G(\vec{v}(s), \vec{Y}) + S(\vec{Y}) - s \\ &= \{\text{By (5)–(6) and (12)}\} \\ &\quad \sum_{m-1 \text{ largest}} \left(U_i \cdot \frac{s - C_i}{m} + C_i - S_i(Y_i) \right) \\ &\quad + \sum_{\tau_i \in \tau} S_i(Y_i) - s \end{aligned}$$

□

These laws are sufficient to understand the computation of $\vec{x}$, the minimum compliant vector.

3.2 New CVA Laws

In this subsection, we prove some new laws that will enable us to reason about the behavior of CVA when certain parameters are modified. At several points, our basic methodology will be similar to that in prior work. We first create a function $f(s)$ such that $f(s) = 0$ iff $\vec{v}(s)$ satisfies (10)–(11) with appropriate substitutions. We then use the Intermediate Value Theorem to show that such a zero exists. The Intermediate Value Theorem requires that we have points a and b such that $f(a) \geq 0$ and $f(b) \leq 0$ (or vice versa). The following lemma will allow us to show that $f(0) > 0$ for each function we will use. For example, observe that Lem. 1 implies $M^Y(\vec{Y}, 0) > 0$, where $M^Y(\vec{Y}, s)$ is defined as in the First Restated Shape Law. Our result was proven for that case in [3], but here we have generalized it.

Lemma 1. *For arbitrary $\vec{Y}$,*

$$G(\vec{v}(0), \vec{Y}) + S(\vec{Y}) > 0. \tag{16}$$

Proof.

$$\begin{aligned} & G(\vec{v}(0), \vec{Y}) + S(\vec{Y}) \\ &= \{\text{By (5)–(6) and (12)}\} \\ &\quad \sum_{m-1 \text{ largest}} \left(-\frac{U_i C_i}{m} + C_i - S_i(Y_i) \right) \\ &\quad + \sum_{\tau_i \in \tau} S_i(Y_i) \\ &= \{\text{Rearranging}\} \\ &\quad \sum_{m-1 \text{ largest}} \left(\frac{C_i(m - U_i)}{m} - S_i(Y_i) \right) \\ &\quad + \sum_{\tau_i \in \tau} S_i(Y_i) \\ &\geq \{\text{Because each } S_i(Y_i) \geq 0 \text{ by (4); observe that} \\ &\quad \text{each } S_i(Y_i) \text{ in the first summation also appears} \\ &\quad \text{in the second}\} \end{aligned}$$

$$\sum_{m-1 \text{ largest}} \frac{C_i(m-U_i)}{m}$$
$$> \{\text{Because } U_i \leq 1 < m\}$$
$$0$$
□

For our next three laws, we will assume that $\vec{R}$ and $\vec{Y}$ have already been given, and that they comply with Thm. 1. That is, they are related as in (2) with minimum compliant vector $\vec{x}$. We will then show that when some component of $\vec{R}$ is increased, the values of $\vec{Y}$ and $\vec{x}$ can be modified to yield *precisely* the new bounds, while $\vec{x}$ remains the minimum compliant vector.

The next law is analogous to the First Restated Shape Law, and also follows directly from the Shape Law by definition. We again use $\vec{v}(s)$ in place of $\vec{x}$, and we now use $\vec{I}(s)$ in place of $\vec{Y}$, because we must determine $\vec{Y}$.

Second Restated Shape Law. *Denote $\vec{v}(s)$ as in (12). By (15), denote $\vec{I}(s)$ with*

$$\forall i, I_i(s) = R_i - v_i(s) - C_i. \tag{17}$$

Also denote

$$M^R(s) = G(\vec{v}(s), \vec{I}(s)) + S(\vec{I}(s)) - s \tag{18}$$

Then, $\vec{v}(s)$ is the minimum compliant vector iff

$$M^R(s) = 0. \tag{19}$$

The next law is required to use the Intermediate Value Theorem. To make the proof less tedious, we make use of the fact that the set of continuous functions is closed under addition and composition.

M^R Continuity Law. *$M^R(s)$ is continuous with respect to s.*

Proof. Observe from (12) that each $v_i(s)$ is continuous with respect to s. By (17), each $I_i(s)$ is therefore also continuous with respect to s. Therefore, by (4), $S_i(I_i(s))$ is also continuous with respect to s. By (5) and (6), we know that $G(\vec{v}(s), \vec{I}(s))$ and $S(\vec{I}(s))$ are continuous with respect to s. Therefore, by (18), $M^R(s)$ is continuous with respect to s. □

We now prove the desired modification property.

R_j Increase Law. *Suppose $\vec{Y}$ yields precisely response time bounds $\vec{R}$ using Thm. 1. Then, if a single component R_j is increased, there exists a new value of $\vec{Y}$ such that Thm. 1 yields precisely the new response time bounds $\vec{R}$.*

Proof. Assume that $\vec{Y}$ yields precisely response time bounds $\vec{R}$ using Thm. 1, where $\vec{x}$ is the unique minimum compliant vector mentioned in that theorem. By the Second Restated Shape Law, there exists an s_0 such that $M^R(s_0) = 0$, where by (10) and (12), $\vec{v}(s_0) = \vec{x}$, and by (15) and (17), $\vec{I}(s_0) = \vec{Y}$. By (17), when R_j is increased, $I_j(s_0)$ will also increase. By (4), $S_j(I_j(s_0))$ will not increase. Because R_j is the only value modified, no other $v_i(s_0)$ or $S_i(I_i(s_0))$ will change. If $S_j(I_j(s_0))$ decreases, then $S(\vec{I}(s_0))$ (see (5)) will decrease, and $G(\vec{v}(s_0), \vec{I}(s_0))$ (see (6)) will increase by no more than the decrease in $S(\vec{I}(s_0))$.

Therefore, by (18), after the increase,

$$M^R(s_0) \leq 0. \tag{20}$$

By the Intermediate Value Theorem, Lem. 1, and the M^R Continuity Law, there is an

$$s_1 \leq s_0 \tag{21}$$

such that

$$M^R(s_1) = 0 \tag{22}$$

under the new analysis.

By our assumption that $\vec{Y}$ initially yielded precisely bounds $\vec{R}$, (9) held initially. As noted above, $\vec{I}(s_0) = \vec{Y}$, so by (9),

$$\forall i, \vec{I}(s_0) \geq 0. \tag{23}$$

By (12), (17), and (21)–(23),

$$\forall i, \vec{I}(s_1) \geq \vec{I}(s_0) \geq 0. \tag{24}$$

Therefore, (9) also continues to hold when $\vec{Y}$ is assigned $\vec{I}(s_1)$ instead of $\vec{I}(s_0)$. Thus, by Thm. 1, the Second Restated Shape Law, and (22), the PP assignment $\vec{I}(s_1)$ does indeed yield the desired response time bounds with minimum compliant vector $\vec{v}(s_1)$. □

We now demonstrate that we can achieve smaller analytical response time bounds by reducing the PPs we use in the analysis. As the PP Uniform Modification Law will demonstrate, we can perform this PP reduction for the analysis without modifying the PPs used by the scheduler at runtime.

PP Uniform Modification Law. *Suppose there are two GEL schedulers with PP assignments $\vec{Y}$ and $\vec{Y}'$ such that each Y_i and Y_i' differ by a system-wide constant. The response time bounds derived by analyzing $\vec{Y}$ will also apply to a system scheduled using $\vec{Y}'$, and vice versa.*

Proof. Using either $\vec{Y}$ or $\vec{Y}'$ will result in the same scheduler decisions, because each absolute PP has been increased or decreased by the same constant. □

This observation will allow us to state the PP Uniform Reduction Law, which allows us to determine the *smallest* response time bounds available using Thm. 1. We will examine the effect on bounds when $\vec{Y}$ is altered in a manner

consistent with the PP Uniform Modification Law. In order to do so, we first analyze the behavior of each single $S_i(Y_i)$ when Y_i is reduced by a non-negative constant Δ.

Lemma 2. *If $\Delta \geq 0$, then for every i, $S_i(Y_i - \Delta) \leq U_i\Delta + S_i(Y_i)$.*

Proof.

$$\begin{aligned}
S_i(Y_i-\Delta) &= \{\text{By (4)}\} \\
&\quad \max\left\{0, C_i\left(1-\frac{Y_i-\Delta}{T_i}\right)\right\} \\
&= \{\text{Rearranging}\} \\
&\quad \max\left\{0, U_i\Delta + C_i\left(1-\frac{Y_i}{T_i}\right)\right\} \\
&\leq \{\text{By definition of max}\} \\
&\quad U_i\Delta + \max\left\{0, C_i\left(1-\frac{Y_i}{T_i}\right)\right\} \\
&= \{\text{By (4)}\} \\
&\quad U_i\Delta + S_i(Y_i)
\end{aligned}$$

□

With this analysis completed, we now characterize the effect of altering $\vec{Y}$ on response time bounds.

PP Uniform Reduction Law. *Consider a GEL scheduler with relative PP assignment $\vec{Y}$. By subtracting from each Y_i the value $\min_{\tau_j \in \tau} Y_j$, an equivalent scheduler can be created for which a minimum compliant vector exists. Furthermore, if the original $\vec{Y}$ satisfied (9), then the analysis using the modified $\vec{Y}$ produces response time bounds that are no larger.*

Proof. For brevity, let

$$\Delta = \min_{\tau_j \in \tau} Y_j \tag{25}$$

By the PP Uniform Modification Law, the analysis produced by subtracting Δ from each Y_i also applies to the original $\vec{Y}$. Furthermore, by definition of "min," $Y_i - \Delta \geq 0$ holds for each i, so the new assignment satisfies (9). Therefore, the preconditions of Thm. 1 are met, so a minimum compliant vector exists.

Suppose the original $\vec{Y}$ satisfies (9). It immediately follows from (25) that

$$\Delta \geq 0. \tag{26}$$

For notational convenience, we will use $\vec{Y} - \Delta$ to denote the vector obtained by subtracting Δ from each component of $\vec{Y}$. In light of the First Restated Shape Law, we define s_0 so that

$$M^Y(\vec{Y}, s_0) = 0. \tag{27}$$

We now want to show that s_1 exists so that $M^Y(\vec{Y} - \Delta, s_1) = 0$.

We will first analyze $M^Y(\vec{Y} - \Delta, s_0 + m\Delta)$, which will allow us to use the Intermediate Value Theorem. Observe from (5)–(6) and (13) that each task τ_i contributes to $S(\vec{Y} - \Delta)$, and some tasks contribute to $G(\vec{v}(s_0 + m\Delta), \vec{Y} - \Delta)$ as well. We will analyze the final value of $M^Y(\vec{Y} - \Delta, s_0 + m\Delta)$ by analyzing the contribution from both the tasks that contribute to $G(\vec{v}(s_0 + m\Delta), \vec{Y} - \Delta)$ and those that do not. For τ_i that does contribute to $G(\vec{v}(s_0 + m\Delta))$, we have

$$\begin{aligned}
&\left(\left(\frac{s_0 + m\Delta - C_i}{m}\right)U_i + C_i - S_i(Y_i - \Delta)\right) \\
&\quad + S_i(Y_i - \Delta) \\
&= U_i\Delta + \left(\left(\frac{s_0 - C_i}{m}\right)U_i + C_i - S_i(Y_i)\right) + S_i(Y_i).
\end{aligned} \tag{28}$$

For all other τ_i, by Lem. 2 and (26) we have

$$S_i(Y_i - \Delta) \leq U_i\Delta + S_i(Y_i). \tag{29}$$

Therefore,

$$\begin{aligned}
&M^Y(\vec{Y} - \Delta, s_0 + m\Delta) \\
&= \{\text{By (13)}\} \\
&\quad G(\vec{v}(s_0 + m\Delta), \vec{Y} - \Delta) + S(\vec{Y} - \Delta) - s_0 - m\Delta \\
&= \{\text{By (5)–(6) and (12)}\} \\
&\quad \sum_{m-1 \text{ largest}} \left(\left(\frac{s_0 + m\Delta - C_i}{m}\right)U_i + C_i - S_i(Y_i - \Delta)\right) \\
&\quad + \sum_{\tau_i \in \tau} S_i(Y_i - \Delta) - s_0 - m\Delta \\
&\leq \{\text{By (28)–(29); although the set of tasks in the first} \\
&\quad \text{summation below may differ from the previous} \\
&\quad \text{expression, that can only produce a larger result}\} \\
&\quad \sum_{m-1 \text{ largest}} \left(\left(\frac{s_0 - C_i}{m}\right)U_i + C_i - S_i(Y_i)\right) \\
&\quad + \sum_{\tau_i \in \tau} U_i\Delta + \sum_{\tau_i \in \tau} S_i(Y_i) - s_0 - m\Delta \\
&= \{\text{Rearranging}\} \\
&\quad \sum_{m-1 \text{ largest}} \left(\left(\frac{s_0 - C_i}{m}\right)U_i + C_i - S(Y_i)\right) \\
&\quad + \sum_{\tau_i \in \tau} S_i(Y_i) - s_0 - \left(m - \sum_{\tau_i \in \tau} U_i\right)\Delta \\
&\leq \{\text{By (1)}\} \\
&\quad \sum_{m-1 \text{ largest}} \left(\left(\frac{s_0 - C_i}{m}\right)U_i + C_i - S(Y_i)\right) \\
&\quad + \sum_{\tau_i \in \tau} S_i(Y_i) - s_0 \\
&= \{\text{By (5)–(6) and (12)}\} \\
&\quad G(\vec{v}(s_0), \vec{Y}) + S(\vec{Y}) - s_0
\end{aligned}$$

$$
\begin{aligned}
&= \{\text{By (13)}\} \\
&\quad M^Y(\vec{Y}, s_0).
\end{aligned}
$$

By (27), $M^Y(\vec{Y}, s_0) = 0$. Therefore, by the above analysis, $M^Y(\vec{Y} - \Delta, s_0 + \Delta) \leq 0$. By the Intermediate Value Theorem, the M^Y Piecewise Linear Law, and Lem. 1, there exists a

$$s_1 \leq s_0 + m\Delta \tag{30}$$

such that $M^Y(\vec{Y} - \Delta, s_1) = 0$, as desired. Therefore, by the First Restated Shape Law, $\vec{v}(s_1)$ is the new minimum compliant vector. Denote the original response time bounds as $\vec{R}$ and the new response time bounds as $\vec{R}'$. For each i,

$$
\begin{aligned}
R_i' &= \{\text{By (12) and (15)}\} \\
&\quad Y_i - \Delta + \frac{s_1 - C_i}{m} + C_i \\
&\leq \{\text{By (30)}\} \\
&\quad Y_i + \frac{s_0 - C_i}{m} + C_i \\
&= \{\text{By (12) and (15)}\} \\
&\quad R_i. \qquad \square
\end{aligned}
$$

4 Global Fair Lateness Scheduling

In this section, we describe the G-FL scheduler and prove that it minimizes the max-lateness bound. First, we define the scheduler.

Definition 1. *The **Global Fair Lateness (G-FL)** scheduler is the GEL scheduler with relative PP assignment*

$$\forall i, Y_i = D_i - \frac{m-1}{m} C_i.$$

Here we demonstrate Thm. 2, which shows that G-FL minimizes the max-lateness bound achievable for any task system τ. This proof is based on a series of transitions, as depicted in Fig. 3.

Theorem 2. *Suppose that there is a PP assignment $\vec{Y}$ yielding a max-lateness bound of $L_{\max}$ under Thm. 1. Then, using G-FL each lateness bound is at most $L_{\max}$ using the analysis provided in the PP Uniform Reduction Law.*

Proof. We begin with an arbitrary system, as in Fig. 3(a). By definition of lateness,

$$\forall i, R_i = D_i + L_i. \tag{31}$$

Therefore, increasing the lateness bound of a task also increases its response time bound. Using the R_j Increase Law repeatedly, we can transform the arbitrary system into one where each task has lateness bound $L_{\max}$, as depicted in Fig. 3(b).

Using (15) and (31), by the First Restated Shape Law we see that for some s,

$$
\begin{aligned}
\forall i, Y_i &= D_i + L_{\max} - v(s) - C_i \\
&= \{\text{By (12)}\} \\
&\quad D_i + L_{\max} - \frac{s}{m} - \frac{m-1}{m} C_i
\end{aligned}
$$

By the PP Uniform Reduction Law, we can instead consider assignment $\vec{Y}'$ such that,

$$
\begin{aligned}
\forall i, Y_i' &= D_i + L_{\max} - \frac{s}{m} - \frac{m-1}{m} C_i \qquad (32) \\
&\quad - \min_{\tau_j \in \tau} \left(D_j + L_{\max} - \frac{s}{m} - \frac{m-1}{m} C_j \right) \\
&= \{\text{Because } L_{\max} \text{ and } \tfrac{s}{m} \text{ do not depend on } j\} \\
&\quad D_i - \frac{m-1}{m} C_i - \min_{\tau_j \in \tau} \left(D_j - \frac{m-1}{m} C_j \right). \qquad (33)
\end{aligned}
$$

Also by the PP Uniform Reduction Law, no task will have a larger response time bound when using $\vec{Y}'$ than when using $\vec{Y}$, so no task will have a lateness bound exceeding $L_{\max}$. This choice of PPs and the resulting lateness bounds are depicted in Fig. 3(c).

We now consider the behavior of G-FL. (Recall that the PPs of G-FL are specified in Def. 1.) We again use the PP Uniform Reduction Law considering this assignment of PPs, defining the new set of PPs as $\vec{Y}'$. We obtain

$$Y_i' = D_i - \frac{m-1}{m} C_i - \min_{\tau_j \in \tau} \left(D_j - \frac{m-1}{m} C_j \right). \tag{34}$$

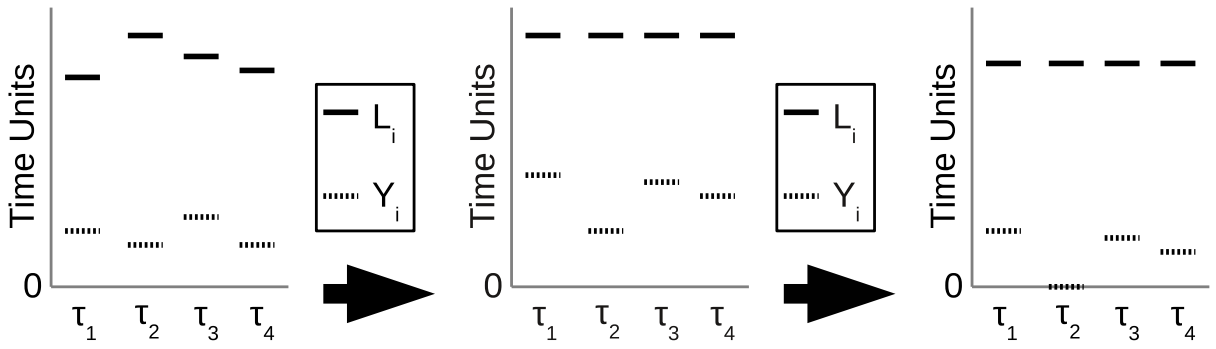
Observe that (33) and (34) are identical. Therefore, under G-FL each task will have a lateness bound no larger than $L_{\max}$. $\square$

Finally, we demonstrate that G-FL has "fair lateness" in the sense that each task will have the same analytical lateness bound under the PP Uniform Reduction Law.

Theorem 3. *Under G-FL, all lateness bounds computed using the PP Uniform Reduction Law are identical.*

Proof. Recall that $\vec{L}$ denotes the lateness bounds. We consider lateness bounds computed using the PP Uniform Reduction Law. By the Shape Law, there is some s such that

$$
\begin{aligned}
L_i &= \{\text{By (3) and (10)}\} \\
&\quad Y_i + \frac{s - C_i}{m} + C_i - D_i \\
&= \{\text{Applying the PP Uniform Reduction Law to} \\
&\quad \text{Def. 1}\}
\end{aligned}
$$



(a) Arbitrary assignment of $\vec{Y}$.

(b) All L_i assigned $L_{\max}$.

(c) Best analysis matches best G-FL analysis.

Figure 3: Proof structure for Thm. 2.

$$
\begin{aligned}
& D_i - \frac{m-1}{m}C_i - \min_{\tau_j \in \tau}\left(D_j - \frac{m-1}{m}C_j\right) + \frac{s-C_i}{m} \\
& \quad + C_i - D_i \\
&= \{\text{Rearranging}\} \\
& \quad D_i - D_i - \frac{m-1}{m}C_i + \frac{m-1}{m}C_i + \frac{s}{m} \\
& \quad - \min_{\tau_j \in \tau}\left(D_j - \frac{m-1}{m}C_j\right) \\
&= \{\text{Cancelling}\} \\
& \quad \frac{s}{m} - \min_{\tau_j \in \tau}\left(D_j - \frac{m-1}{m}C_j\right). \qquad (35)
\end{aligned}
$$

Observe that (35) does not depend on the task index i. Therefore, the lateness bound is the same for all tasks. □

5 Experimental Analysis

To compare G-FL and G-EDF, we generated implicit-deadline task sets based on the experimental design from prior studies (e.g., [15]). We generated task utilizations using either a uniform or a bimodal distribution. For task sets with uniformly distributed utilizations, we used either a *light* distribution with values randomly chosen from $[0.001, 0.1]$, a *medium* distribution with values randomly chosen from $[0.1, 0.4]$, or a *heavy* distribution with values randomly chosen from $[0.5, 0.9]$. For tasks sets with bimodally distributed utilizations, values were chosen uniformly from either $[0.001, 0.5]$ or $[0.5, 0.9]$, with respective probabilities of $8/9$ and $1/9$ for *light* distributions, $6/9$ and $3/9$ for *medium* distributions, and $4/9$ and $5/9$ for *heavy* distributions. We generated integral task periods using a uniform distribution from $[3ms, 33ms]$ for *short* periods, $[10ms, 100ms]$ for *moderate* periods, or $[50ms, 250ms]$ for *long* periods. Each experiment was run with either $m = 2$, $m = 4$, or $m = 6$ CPUs. By [9] we do not need to consider significantly larger numbers of CPUs, as clustered scheduling will outperform global scheduling for such systems in practice.

Because negative lateness is likely not beneficial to a system designer, we compared tardiness rather than lateness. All tardiness bound computations were done with respect to the PP Uniform Reduction Law. With either PP assignment (G-EDF or G-FL), the computation algorithm from [3] can be used.

We ran two experiments for each possible combination of utilization distribution, period distribution, and processor count: one to compare the tardiness bounds for G-FL and G-EDF, and one to compare the observed tardiness for G-FL and G-EDF. For each experiment, we generated 1000 task sets. For each task set τ, we generated tasks until generating one that would cause $\sum_{\tau_i \in \tau} U_i$ to exceed m. For observed tardiness experiments, we restricted task worst-case execution times to be integers.

For each task set, we computed "maximum tardiness" metrics g for G-EDF and h for G-FP. In the tardiness bound experiments, g and h were defined as the maximum tardiness bounds computed using the PP Uniform Reduction Law. In the observed tardiness experiments, we defined g and h as the maximum tardiness experienced by any job during the first 100 seconds of execution, when all job releases are periodic and all jobs run for their full worst-case execution times. For each experiment, we computed $\bar{g}$, the mean g value over all task sets, and $\bar{h}$, the mean h value over all task sets. The *relative improvement* for an experiment is defined as $(\bar{g} - \bar{h})/\bar{g}$.

Results of our experiments are shown in Figs. 4 and 5. Tardiness bound experiments often showed an improvement of about 30%, and observed tardiness experiments some-

times showed an improvement exceeding 99%. These results show that G-FL is significantly better than G-EDF both in terms of analytical bounds and in terms of observed tardiness.

Although this work focuses on SRT scheduling, we did explore the use of G-FL for HRT scheduling by computing the percentage of task sets with no observed deadline misses. These results are depicted in Fig. 6 for G-EDF and Fig. 7 for G-FL. Particularly for tasks with bimodal utilization distributions, G-FL scheduled many more task sets without observed deadline misses.

6 Conclusion

We have demonstrated that G-FL provides max-lateness bounds no larger than those currently available for G-EDF and have provided experimental evidence that G-FL is superior to G-EDF both in terms of max-tardiness bounds and maximum observed tardiness. Furthermore, G-FL provides equal tardiness bounds for all tasks in the system, and therefore provides a closer relationship between deadlines and response time bounds than G-EDF currently does. The implementation of G-FL is identical to that of G-EDF with arbitrary deadlines, and G-FL maintains the desirable JLSP property (enabling known synchronization techniques.) Therefore, G-FL is a better choice than G-EDF for SRT systems.

Although this paper does not focus on HRT analysis for G-FL, experimental evidence indicates that G-FL may be a better HRT scheduler than G-EDF. Future work could examine this possibility.

References

[1] C. Kenna, J. Herman, B. B. Brandenburg, A. F. Mills, and J. H. Anderson, "Soft real-time on multiprocessors: Are analysis-based schedulers really worth it?" in *RTSS*, 2011, pp. 99–103.

[2] U. C. Devi and J. H. Anderson, "Tardiness bounds under global EDF scheduling on a multiprocessor," *Real-Time Sys.*, vol. 38, no. 2, pp. 133–189, 2008.

[3] J. P. Erickson, N. Guan, and S. K. Baruah, "Tardiness bounds for global EDF with deadlines different from periods," in *OPODIS*, 2010, pp. 286–301, revised version at http://cs.unc.edu/~jerickso/opodis2010-tardiness.pdf.

[4] H. Leontyev and J. H. Anderson, "Generalized tardiness bounds for global multiprocessor scheduling," *Real-Time Sys.*, vol. 44, no. 1, pp. 26–71, 2010.

[5] H. Leontyev, S. Chakraborty, and J. H. Anderson, "Multiprocessor extensions to real-time calculus," in *RTSS*, 2009, pp. 410–421.

[6] S. K. Baruah, N. K. Cohen, C. G. Plaxton, and D. A. Varvel, "Proportionate progress: A notion of fairness in resource allocation," *Algorithmica*, vol. 15, pp. 600–625, 1996.

[7] T. Megel, R. Sirdey, and V. David, "Minimizing task preemptions and migrations in multiprocessor optimal real-time schedules," in *RTSS*, 2010, pp. 37–46.

[8] P. Regnier, G. Lima, E. Massa, G. Levin, and S. Brandt, "RUN: Optimal multiprocessor real-time scheduling via reduction to uniprocessor," in *RTSS*, 2011, pp. 104–115.

[9] A. Bastoni, B. B. Brandenburg, and J. H. Anderson, "An empirical comparison of global, partitioned, and clustered multiprocessor EDF schedulers," in *RTSS*, 2010, pp. 14–24.

[10] B. B. Brandenburg, "Scheduling and locking in multiprocessor real-time operating systems," Ph.D. dissertation, The University of North Carolina at Chapel Hill, 2011.

[11] H. Cho, B. Ravindran, and E. D. Jensen, "Efficient real-time scheduling algorithms for multiprocessor systems," *IEICE Trans. Comm.*, vol. 85, pp. 807–813, 2002.

[12] T. Baker, M. Cirinei, and M. Bertogna, "EDZL scheduling analysis," *Real-Time Systems*, vol. 40, pp. 264–289, 2008.

[13] J. P. Erickson, U. Devi, and S. K. Baruah, "Improved tardiness bounds for global EDF," in *ECRTS*, 2010, pp. 14–23.

[14] J. Lee, A. Easwaran, and I. Shin, "Maximizing contention-free executions in multiprocessor scheduling," in *RTAS*, 2011, pp. 235–244.

[15] A. Bastoni, B. B. Brandenburg, and J. H. Anderson, "Is semi-partitioned scheduling practical?" in *ECRTS*, 2011, pp. 125–135.

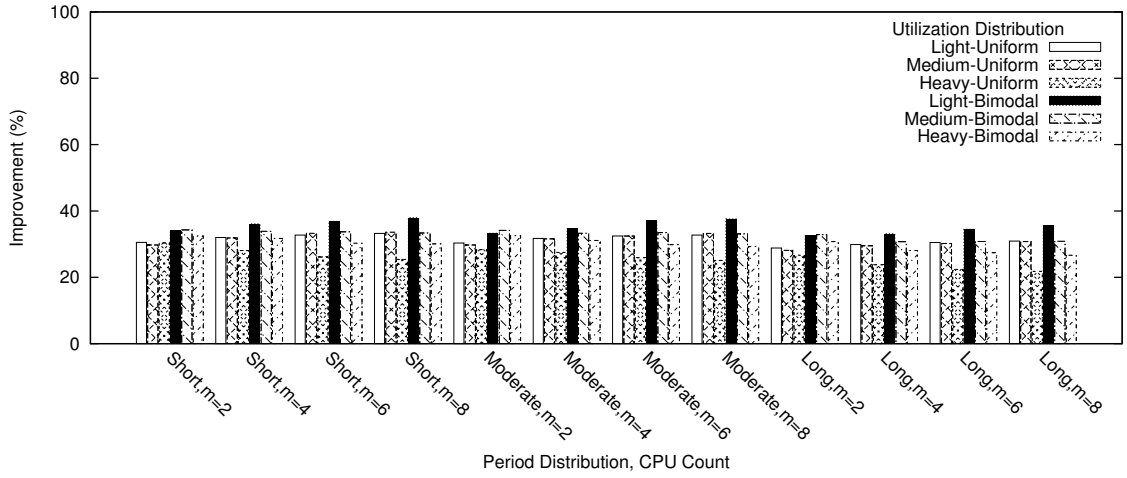


Figure 4: Improvement of G-FL over G-EDF for max-tardiness bound.

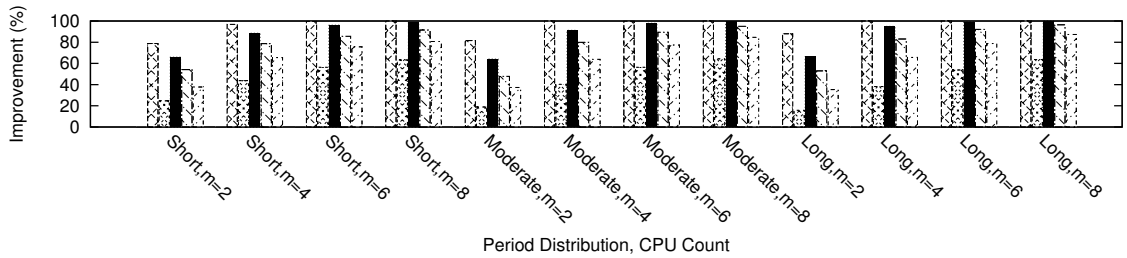


Figure 5: Improvement of G-FL over G-EDF for max observed tardiness. (No tardiness was observed for Light-Uniform in any schedule.)

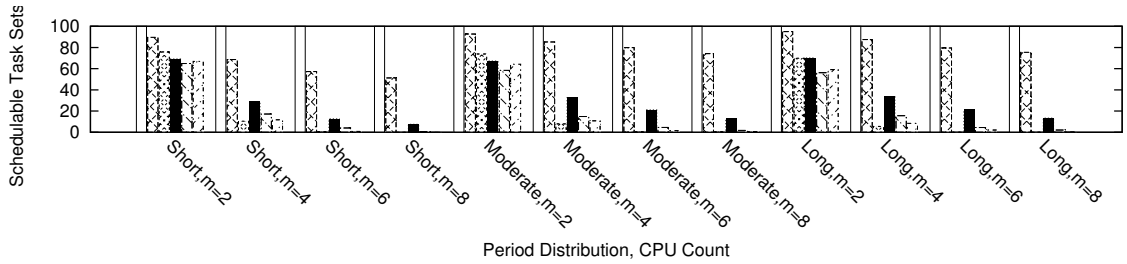


Figure 6: Ratio of task sets with no observed deadline misses for G-EDF.

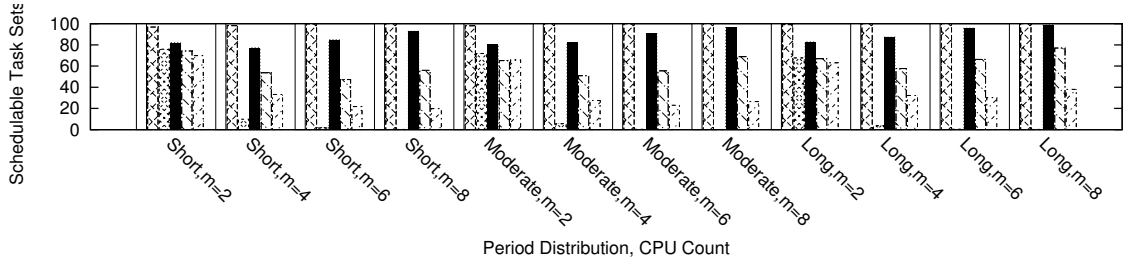


Figure 7: Ratio of task sets with no observed deadline misses for G-FL.