Improved conditions for bounded tardiness under EPDF Pfair multiprocessor scheduling

UmaMaheswari C. Devi a,⁎, 1, James H. Anderson b

a IBM India Research Laboratory, Bangalore, KA 560071, India
b Department of Computer Science, The University of North Carolina at Chapel Hill, Chapel Hill, NC 27599, USA

Abstract

The earliest-pseudo-deadline-first (EPDF) Pfair algorithm is more efficient than other known Pfair scheduling algorithms, but is not optimal for scheduling recurrent real-time task systems on more than two identical processors. Although not optimal, EPDF may be preferable for real-time systems instantiated on less-powerful platforms, those with soft timing constraints, or those whose task composition can change at run-time. In prior work, Srinivasan and Anderson established a sufficient per-task utilization restriction for ensuring a tardiness of at most \(q\) quanta, where \(q \geq 1\), under EPDF. They also conjectured that under this algorithm, a tardiness bound of one quantum applies to task systems that are not subject to any restriction other than the obvious restrictions, namely, that the total system utilization not exceed the available processing capacity and per-task utilizations not exceed 1.0. In this paper, we present counterexamples that show that their conjecture is false and present sufficient per-task utilization restrictions that are more liberal than theirs. For ensuring a tardiness bound of one quantum, our restriction presents an improvement of 50% over the previous one.

© 2009 Elsevier Inc. All rights reserved.

1. Introduction

We consider the scheduling of recurrent (i.e., periodic, sporadic, or rate-based) real-time task systems on multiprocessor platforms consisting of \(M\) identical, unit-capacity processors. Pfair scheduling, originally introduced by Baruah et al. [4], is the only known way of optimally scheduling such multiprocessor task systems. (A real-time scheduling algorithm is said to be optimal iff it can schedule without deadline misses every task system for which some correct schedule exists.) To ensure optimality, Pfair scheduling imposes a stronger constraint on the timeliness of tasks than that mandated by periodic scheduling. Specifically, under Pfair scheduling, each task must execute at an approximately uniform rate at all times, while respecting a fixed-size allocation quantum. A task's execution rate is defined by its weight (i.e., utilization). Uniform rates are ensured by subdividing each task into quantum-length subtasks that are subject to intermediate deadlines, called pseudo-deadlines, computed based on the task's weight. Under most known Pfair algorithms, subtasks are scheduled...
on an earliest-pseudo-deadline-first basis. However, to avoid deadline misses, ties among subtasks with the same deadline must be resolved carefully. In fact, tie-breaking rules are of crucial importance when devising optimal Pfair scheduling algorithms.

1.1. Motivation

The overheads associated with the tie-breaking rules of the optimal algorithms may be problematic for some applications. The earliest-pseudo-deadline-first (EPDF) algorithm is computationally more efficient than optimal Pfair algorithms in that it does not use any tie-breaking rule to resolve ties among subtasks with the same pseudo-deadline, but disambiguates them arbitrarily. PD\textsuperscript{2}, the most efficient of the known optimal Pfair algorithms, requires two tie-break parameters. Though these tie-break parameters can be computed for each subtask in constant time, there exist some soft and/or dynamic real-time systems in which not using any tie-breaking rule may still be preferable. Eliminating tie-breaking rules may also be preferable in embedded systems with slower processors or limited memory bandwidth.

The viability of EPDF for scheduling soft and/or dynamic real-time systems was first considered by Srinivasan and Anderson in [11], where they provided examples of such applications for which EPDF may be preferable to PD\textsuperscript{2}. Some web-hosting systems, server farms, packet processing in programmable multiprocessor-based routers, and packet transmission on multiple, parallel outgoing router links are among the examples provided by them. In these systems, fair resource allocation is needed, so that quality-of-service guarantees can be provided. However, an extreme notion of fairness that precludes all deadline misses is not required. Moreover, in systems such as routing networks, the inclusion of tie-breaking information in subtask priorities may result in unacceptably high space overhead.

The applications mentioned above may also be dynamic in that the set of tasks and the utilizations of tasks requiring service may change. In [11], Srinivasan and Anderson also noted that the use of tie-breaking rules may be problematic for such dynamic task systems. As they explained, it is possible to reweight each task whenever its utilization changes such that its next subtask deadline is preserved. If no tie-breaking information is maintained, such an approach entails very little computational overhead. However, utilization changes can cause tie-breaking information to change, so if tie-breaking rules are used, then reweighting may necessitate an $O(N)$ cost for $N$ tasks, due to the need to reconstruct the scheduler’s priority queue. This cost may be prohibitive if reallocations are frequent.

Motivated by the above reasons, Srinivasan and Anderson studied EPDF and they succeeded in showing that this algorithm can guarantee a tardiness (i.e., lateness) bound of $q \geq 1$ quanta for every subtask, provided a certain condition holds. Their condition can be ensured by limiting each task’s weight to at most $\frac{2}{qT}$. Unfortunately, Srinivasan and Anderson left open the question of whether such weight restrictions are necessary to guarantee small bounded tardiness. Moreover, they conjectured that EPDF can ensure a tardiness bound of one quantum as long as the weight of each task does not exceed 1.0 (i.e., the capacity of a single processor), and the total system utilization does not exceed the total available processing capacity.

1.2. Contributions

Our contributions in this paper are two-fold. First, we provide counterexamples that show that the above conjecture is false, and that, in general, restrictions on individual task utilizations are necessary to guarantee bounded tardiness under EPDF. Our second contribution is to show that a more liberal per-task weight restriction of $\frac{1+2}{qT}$ is sufficient to ensure a tardiness of $q$ quanta. When $q = 1$, this presents an improvement of 50% over the previous result.

The rest of the paper is organized as follows. Section 2 provides an overview of Pfair scheduling. In Section 3, the results described above are established. Section 4 concludes.

2. Background on Pfair scheduling

This section describes some basic concepts of Pfair scheduling, provides needed background, and summarizes results from prior work reported in [1–4,10,11]. Pfair scheduling [4,10] can be used to schedule a periodic, sporadic, intra-sporadic (IS), or generalized-intra-sporadic (GIS) (see below) task system $\tau$ on $M \geq 1$ identical processors, each of whose processing capacity is taken to be 1.0. As explained later, in this paper, we assume that $M \geq 3$ holds. Each task $T$ of $\tau$ is assigned a rational weight $wt(T) \in (0, 1]$ that denotes the fraction of a single processor it requires. In this paper, for simplicity and to avoid some boundary cases, we assume that $wt(T) < 1$ holds. The sum of the weights of all the tasks in $\tau$, i.e., the total system utilization of $\tau$, is assumed to be at most $M$, which is the total available processing capacity. For a periodic or a sporadic task $T$, $wt(T) = T.e/T.p$, where $T.e$ and $T.p$ are the execution cost and inter-arrival time or period, respectively, of $T$. When scheduled under Pfair algorithms, it is required that $T.e$ and $T.p$ be specified as integers, which are interpreted as integral numbers of quanta.

A periodic or sporadic task $T$ may be invoked zero or more times; $T$ is periodic if any two consecutive invocations of $T$ are separated by exactly $T.p$ time units and is sporadic if $T.p$ is a lower-bound on the inter-arrival separation. Each invocation of $T$ is referred to as a job of $T$. The first job may be invoked or released at any time at or after time zero. Every job of $T$ executes for $T.e$ time units and should complete execution within $T.p$ time units of its release, i.e., every job of $T$ has a relative deadline of $T.p$ time units. (In this paper, for ease of description, we assume that each job executes for exactly
2.1. Periodic, sporadic, and IS task models

In Pfair scheduling, each quantum of execution of each task is referred to as a subtask. Each task gives rise to a potentially infinite sequence of subtasks. The ith subtask of T is denoted Ti, where i ≥ 1. If T is periodic or sporadic, then the kth job of T consists of subtasks Tk,i,e+1, ..., Tk,e, where e = T.e and k ≥ 1.

Each subtask Ti has an associated pseudo-release r(Ti) and pseudo-deadline d(Ti), defined as follows (the prefix “pseudo-” is often omitted for conciseness):

\[ r(T_i) = \Theta(T_i) + \left\lceil \frac{i - 1}{w(T)} \right\rceil \quad \text{and} \quad d(T_i) = \Theta(T_i) + \left\lceil \frac{i}{w(T)} \right\rceil. \]  

(1)

In the above formulas, \( \Theta(T_i) \) denotes the offset of Ti and is used in modeling the late releases of sporadic and IS tasks. It is well known that the sporadic model generalizes the periodic model by allowing jobs to be released “late”; the intra-sporadic model (IS model), proposed by Srinivasan and Anderson in [2,10], is a further generalization that allows subtasks to be released late. The offsets of T’s various subtasks are non-negative and satisfy the following:

\[ k > i \implies \Theta(T_k) \geq \Theta(T_i). \]  

(2)

T is periodic if \( \Theta(T_i) = c \) holds for all i (and is synchronous also if c = 0), and is IS, otherwise. For a sporadic task, all subtasks that belong to the same job will have equal offsets. Examples are given in insets (a) and (b) of Fig. 1. Informally, the restriction in (2) on offsets implies that the separation between any pair of subtask releases is at least the separation between those releases if the task were periodic.

The interval \( [r(T_i), d(T_i)] \) is termed the PF-window of Ti and is denoted \( \omega(T_i) \). The following lemma, concerning PF-window lengths, follows from (1).

**Lemma 1.** (See Anderson and Srinivasan [3].) The length of the PF-window of any subtask Ti of a task T, \( |\omega(T_i)| = d(T_i) - r(T_i) \), is either \( \left\lceil \frac{i}{w(T)} \right\rceil \) or \( \left\lceil \frac{i - 1}{w(T)} \right\rceil + 1 \).

2.2. GIS task model

When proving properties concerning Pfair scheduling algorithms, it is sometimes useful to “eliminate” or “omit” certain subtasks. For example, if a deadline miss does not depend on the existence of a subtask, then ignoring such a subtask makes analysis easier. In [10], Srinivasan and Anderson introduced the generalized intra-sporadic model (GIS model) to facilitate such selective removal of subtasks. A GIS task system is obtained by omitting subtasks from a corresponding IS (or GIS) task system. However, the spacing between subtasks of a task that are not omitted may not be decreased in comparison to how they are spaced in a periodic task. Specifically, subtask \( T_k \) may be followed by subtask \( T_k+i+1 \) if the following holds: \( r(T_k+i+1) \geq r(T_k+i) \). That is, \( r(T_k) \) is not smaller than what it would have been.

![Fig. 1.](image)
if \( T_{i+1}, T_{i+2}, \ldots, T_{k-1} \) were present, and released as early as possible. For the special case where \( T_{i} \) is the first subtask released by \( T \), \( r(T_{i}) \) must be at least \( \frac{i-1}{\text{wt}(T)} \). Fig. 1(c) shows an example. In this example, though subtask \( T_{2} \) is omitted, \( T_{3} \) cannot be released before time 4. If a task \( T \), releases subtask \( T_{k} \) after executing subtask \( T_{i} \), then \( T_{k} \) is called the successor of \( T_{i} \) and \( T_{i} \) is called the predecessor of \( T_{k} \). Note that a periodic task system is an IS task system, which in turn is a GIS task system, so any property established for the GIS task model applies to the other models, as well.

2.3. The early-release task model

The task models described so far are non-work-conserving in that, the second and later subtasks remain ineligible to be scheduled before their release times, even if they are otherwise ready and some processor is idle. The early-release (ER) task model is a work-conserving variant of the other models that allows subtasks to be scheduled before their release times [1]. Early releasing can be applied to subtasks in any of the task models considered so far, and unless otherwise specified, it should be assumed that early releasing is enabled. However, whether subtasks are actually released early is optional. To facilitate this, in this model, each subtask \( T_{i} \) has an eligibility time \( \epsilon(T_{i}) \) that specifies the first time slot in which \( T_{i} \) may be scheduled. The interval \([\epsilon(T_{i}), d(T_{i})] \) is referred to as the IS-window of \( T_{i} \). Fig. 1(d) gives an example. It is required that the following holds:

\[
(\forall i \geq 1 : \epsilon(T_{i}) \leq r(T_{i}) \land \epsilon(T_{i}) \leq \epsilon(T_{i+1}))
\]

(3)

Note that the model is very flexible in that it does not preclude a job from becoming eligible before its release time, but provides mechanisms to restrict such behavior, if so desired. Such flexibility, in conjunction with the sporadic or the IS task model, can be used to schedule rate-based tasks, whose arrival pattern may be jittered, and whose instantaneous workload may deviate from the average or expected workload, such as in distributed multimedia and digital signal processing applications [7,10].

2.4. b-bit

The b-bit or boundary bit is associated with each subtask \( T_{i} \) and is denoted \( b(T_{i}) \). \( b(T_{i}) \) is as defined by (4) below:

\[
b(T_{i}) = \left\lceil \frac{i}{\text{wt}(T)} \right\rceil - \left\lfloor \frac{i}{\text{wt}(T)} \right\rfloor.
\]

(4)

From (1), it can be verified that if \( \Theta(T_{i}) < \Theta(T_{i+1}) \), then \( d(T_{i}) \leq r(T_{i+1}) \). Therefore, the PF-windows of \( T_{i} \) and \( T_{i+1} \) can overlap only if \( \Theta(T_{i}) = \Theta(T_{i+1}) \). It can also be verified that if \( \Theta(T_{i}) = \Theta(T_{i+1}) \), then \( d(T_{i}) - r(T_{i+1}) = b(T_{i}) \). Hence, \( b(T_{i}) \) determines whether the PF-window of \( T_{i} \) can overlap that of \( T_{i+1} \). Observe that \( b(T_{i}) \) is either zero or one. Therefore, the PF-windows of \( T_{i} \) and \( T_{i+1} \) are either disjoint or overlap by at most one slot. In Fig. 1, \( b(T_{2}) = 1 \), while \( b(T_{3}) = 0 \). Therefore, the PF-window of \( T_{2} \) overlaps \( T_{3} \)’s when \( \Theta(T_{3}) = \Theta(T_{2}) \) as in insets (a), (b), and (d). Further, as shown in [6], the following lemma holds.

**Lemma 2. (From [6].)** For all \( i \geq 1, k \geq 1 \), the following holds:

\[
r(T_{i+k}) \geq \begin{cases} 
  d(T_{i}) + k - 1, & b(T_{i}) = 0, \\
  d(T_{i}) + k - 2, & b(T_{i}) = 1.
\end{cases}
\]

2.5. Group deadlines

Like the b-bit, the group deadline is a parameter that is associated with each subtask and is used by some Pfair scheduling algorithms. The group deadline of subtask \( T_{i} \) is denoted \( D(T_{i}) \).

By Lemma 1, all the windows of a heavy task with weights in the range \([1/2, 1]\) are of length two or three. Informally, for such tasks, the group deadline marks the end of a sequence of subtasks whose PF-windows satisfy the following two properties: each window, except possibly of the first subtask in the sequence, is of length two, and every consecutive pair of windows is overlapping. In Fig. 2(a), \( T_{1}, T_{2} \) is one such sequence in which the first window is of length two; \( T_{3}, \ldots, T_{5} \) and \( T_{6}, \ldots, T_{8} \) are other such sequences in which the first window is of length three. In each sequence, if any subtask is not scheduled until its last slot, then all subsequent subtasks will be forced to be scheduled in their last slots as well, and in that sense constitute a “group.” In addition, if the last subtask in the group is followed by a subtask with a window of length three, as in the first two groups considered above, then this subtask will also be precluded from being scheduled in its first slot (when any subtask in the group is scheduled in its last slot). However, no later subtask is directly impacted. Thus, the group deadline of \( T_{i} \) can be thought of as the earliest time \( t \) after \( r(T_{i}) \) such that \( t \) is the release time of some subtask and no subtask released at or after \( t \) is directly influenced by whether \( T_{i} \) is scheduled in its last slot.

Informally, for a heavy periodic task with weight less than one, the end of each slot that is not the first slot of the PF-window of any of its subtasks is a group deadline. In Fig. 2(a), times 4, 8, and 11 are group deadlines for \( T \) in the
whenever it is active. Let received in an ideal schedule. In an ideal schedule, each task executes at a precise rate given by its utilization.

2.6. Concrete and non-concrete task systems

Subtasks are actually released. An illustration for an IS task is provided in Fig. 2(b).

... Ideal schedule, referred to as the interval [0, 11]. Note that no subtask is released at times 3, 7, or 10. Formally, the group deadline of a subtask \( T_i \) is defined as \( D(T_i) = \min\{u : u \geq d(T_i) \wedge u \text{ is a group deadline of } T\} \).

The group deadline of a subtask \( T_i \) of an IS or GIS task is computed assuming that all later subtasks are present and released as early as possible, that is, under the assumption that \( \Theta(T_i) = \Theta(T_j) \) holds for all \( j \geq i \), regardless of how the subtasks are actually released. An illustration for an IS task is provided in Fig. 2(b).

A task system is said to be concrete if release and eligibility times are specified for each subtask of each task, and non-concrete, otherwise. Note that an infinite number of concrete task systems can be specified for every non-concrete task system. The type of the task system is indicated only when necessary.

2.7. Pfair and ERfair schedules

The notion of a Pfair schedule is obtained by comparing the allocations that each task receives in such a schedule to those received in an ideal fluid schedule. In an ideal fluid schedule, each task executes at a precise rate given by its utilization whenever it is active. Let \( A(\text{ideal}, T, t_1, t_2) \) and \( A(S, T, t_1, t_2) \) denote the total allocation to \( T \) in the interval \([t_1, t_2]\) in the ideal schedule and an actual schedule, \( S \), respectively. Then, the “error” in allocation to \( T \) in \( S \) at time \( t \) with respect to the ideal schedule, referred to as the lag of \( T \) at \( t \) in \( S \), is given by \( \text{lag}(T, t, S) = A(\text{ideal}, T, 0, t) - A(S, T, 0, t) \).

If early releases are allowed, then it is not required that the negative lag constraint, \( \text{lag}(T, t) > -1 \), hold. A schedule for which \( \forall T, t : \text{lag}(T, t) < 1 \) holds is said to be ERfair. The release times and deadlines in (1) are assigned such that scheduling each subtask by its deadline is sufficient to generate an ERfair schedule for \( \tau \); a Pfair schedule can be generated if each subtask is scheduled at or after its release time, as well. Further, ensuring that each task is scheduled in a Pfair or an ERfair manner is sufficient to ensure that the deadlines of all jobs are met in a periodic or sporadic task system. A schedule that is Pfair or ERfair exists for a GIS task system \( \tau \) on \( M \) processors iff

\[
\sum_{T \in \tau} \text{wt}(T) \leq M
\]  

holds [2,4]. A GIS task system satisfying (5) and in which the weight of each task is at most 1.0 is said to be feasible on \( M \) processors. (In general, a task system is said to be feasible on \( M \) processors if there exists some way of correctly scheduling that task system on \( M \) processors.)

If \( T \) is synchronous and periodic, then \( A(\text{ideal}, T, 0, t) \) equals \( t \cdot \text{wt}(T) \). However, if \( T \) is GIS, then the allocation it receives in the ideal schedule may be less due to IS separations or omitted subtasks. To facilitate expressing \( A(\text{ideal}, T, 0, t) \) for GIS
Anderson and Srinivasan showed that delayed subtasks. Also, the total allocation that a subtask receives in slot 4. Note that in the ideal schedule, each subtask completes executing by its deadline.

\[ A(t, t, t) = \begin{cases} 
\frac{i-1}{\text{wt}(T_i)} + 1 \times \text{wt}(T) - (i-1), & u = r(T_i), \\
\frac{i - \lceil \frac{i}{\text{wt}(T_i)} \rceil}{\text{wt}(T_i)}, & u = d(T_i) - 1, \\
0, & r(T_i) < u < d(T_i) - 1, \\
\end{cases} \]  \tag{6}

Let \( A(T, t) \) denote the total allocation to task \( T \) in slot \( t \). Then, \( A(T, t) \) is given by \( \sum_l A(l, T, t) \). For example, in Fig. 3, \( A(T, 4) = A(T_2, 4) + A(T_3, 4) = 1/7 + 2/7 = 3/7 \), since no subtasks other than \( T_2 \) and \( T_3 \) receive a non-zero allocation in slot 4. Note that in the ideal schedule, each subtask completes executing by its deadline.

As shown in Fig. 3, \( A(T, t) \) usually equals \( \text{wt}(T) \), but in certain slots, it may be less than \( \text{wt}(T) \) due to omitted or delayed subtasks. Also, the total allocation that a subtask \( T_i \) receives in the slots that span its window is exactly one in the ideal schedule. These and similar properties have been proved formally in [9]. Later in this paper, we will use Lemma 3, and (7)–(10) given below (examples of which can be seen in Fig. 3):

\[ (\forall T, u \geq 0 : A(T, t, u) \leq \text{wt}(T)), \]  \tag{7}

\[ (\forall T_i : \sum_{u=r(T_i)}^{d(T_i)-1} A(T_i, T, u) = 1), \]  \tag{8}

\[ (\forall T_i, u \geq 0 : A(T_i, t, u) \leq \text{wt}(T_i)), \]  \tag{9}

\[ (\forall T_i, u \in [r(T_i), d(T_i)) : A(T_i, t, u) \geq 1/T_i.p). \]  \tag{10}

**Lemma 3.** If \( b(T_i) = 1 \) and subtask \( T_{i+1} \) exists, then \( A(T_i, d(T_i) - 1) + A(T_i, t+1, r(T_{i+1})) = \text{wt}(T_i) \).

A task \( T \)'s ideal allocation up to time \( t \) is simply

\[ A(T, 0, t) = \sum_{u=0}^{t} A(T, t, u) = \sum_{u=0}^{t} A(T, t, u), \]  and hence

\[ A(T, 0, t) = \sum_{u=0}^{t} A(T, t, u), \]  and hence

**Fig. 3.** Per-slot ideal allocations to subtasks of a task \( T \) with weight \( 3/7 \). These allocations are marked above the subtask windows. (a) \( T \) is synchronous, periodic. \( A(T, t) = 3/7 \) holds for every \( t \). \( A(T_2, 4) = 2/7 \) and \( A(T_3, 4) = 1/7 \). (b) \( T \) is GIS. \( T_2 \)'s release is delayed by one time slot. \( T_4 \) is delayed by an additional time slot and \( T_5 \) is omitted. Here, \( A(T_2, 4) = 2/7 \) and \( A(T_3, 4) = 0 \).
Using (14)–(16),

\[
\text{lag}(T, t + 1) = \sum_{u=0}^{t} (A(\text{ideal}, T, u) - S(T, u))
\]

Similarly, by (12) again, for any \(0 \leq t' \leq t\),

\[
\text{lag}(T, t + 1) = \text{lag}(T, t') + \sum_{u=t'}^{t} (A(\text{ideal}, T, u) - S(T, u))
\]

From (12), \(\text{lag}(T, t + 1)\) (the schedule parameter is omitted in the lag and LAG functions when unambiguous) is given by

\[
\text{lag}(T, t + 1) = \sum_{u=0}^{t} (A(\text{ideal}, T, u) - S(T, u))
\]

Similarly, by (12) again, for any \(0 \leq t' \leq t\),

\[
\text{lag}(T, t + 1) = \text{lag}(T, t') + \sum_{u=t'}^{t} (A(\text{ideal}, T, u) - S(T, u))
\]

Another useful definition, the total lag for a task system \(\tau\) in a schedule \(S\) at time \(t\), LAG\((\tau, t, S)\), or more concisely, LAG\((\tau, t)\), is given by

\[
\text{LAG}(\tau, t) = \sum_{T \in \tau} \text{lag}(T, t).
\]

Using (14)–(16), \(\text{LAG}(\tau, t + 1)\) can be expressed as follows. In (18) below, \(0 \leq t' \leq t\) holds:

\[
\text{LAG}(\tau, t + 1) = \text{LAG}(\tau, t) + \sum_{T \in \tau} (A(\text{ideal}, T, t) - S(T, t))
\]

\[
\text{LAG}(\tau, t + 1) = \text{LAG}(\tau, t') + \sum_{u=t'}^{t} \sum_{T \in \tau} (A(\text{ideal}, T, u) - A(S, T, u))
\]

Eqs. (17) and (18) above can be rewritten as follows using (7):

\[
\text{LAG}(\tau, t + 1) \leq \text{LAG}(\tau, t) + \sum_{T \in \tau} (w(T) - S(T, t))
\]

\[
\text{LAG}(\tau, t + 1) \leq \text{LAG}(\tau, t') + (t + 1 - t') \cdot \sum_{T \in \tau} w(T) - A(S, \tau, t', t + 1)
\]

\[
= \text{LAG}(\tau, t') + (t + 1 - t') \cdot \sum_{u=t'}^{t} \sum_{T \in \tau} S(T, u).
\]

2.8. Soft real-time model

In soft real-time systems, tasks may miss their deadlines. As discussed in the introduction, this paper is concerned with deriving a lateness or tardiness [8] bound for a GIS task system scheduled under EPDF (described below). Formally, the tardiness of a subtask \(T_i\) in schedule \(S\) is defined as tardiness\((T_i, S) = \max(0, t - d(T_i))\), where \(t\) is the time at which \(T_i\) completes executing in \(S\). The tardiness of a task system \(\tau\) under scheduling algorithm \(A\) is defined as the maximum tardiness of any subtask of any task in \(\tau\) in any schedule for any concrete instantiation of \(\tau\) under \(A\). If \(\kappa(M)\) is the maximum tardiness of any task system with \(U_{\text{sum}} \leq M\) under \(A\) on \(M\) processors, then \(A\) is said to ensure a tardiness bound of \(\kappa(M)\) on \(M\) processors. Though tasks in a soft real-time system are allowed to have non-zero tardiness, it is assumed that missed deadlines do not delay future job releases. Hence, guaranteeing a reasonable bound on tardiness that is independent of time is sufficient to ensure that in the long run each task is allocated a processor share that is in accordance with its weight. Because each task is sequential and subtasks of a task have an implicit precedence relationship, a later subtask cannot commence execution until all prior subtasks of the same task have completed execution. Thus, a missed deadline effectively reduces the interval over which the next subtask should be scheduled in order to meet its deadline.
2.9. Algorithm EPDF

Like most Pfair scheduling algorithms, the earliest-pseudo-deadline-first (EPDF) Pfair algorithm functions by choosing subtasks for execution at the beginning of every slot. Under EPDF, higher priority is accorded to subtasks with earlier deadlines; ties among subtasks with equal deadlines are resolved arbitrarily. In prior work, Srinivasan and Anderson have shown that EPDF is optimal on at most two processors [3]. They have also shown that on more than two processors, EPDF can correctly schedule task systems in which the maximum task weight is at most $1/(M-1)$ [12], and that EPDF can ensure a tardiness bound of $q \geq 1$ if the weight of each task is restricted to $\frac{2}{q+1}$ [11]. (Since EPDF is optimal on two processors, in deriving tardiness bounds under this algorithm, we assume that $M \geq 3$ holds.)

The above is a fairly comprehensive summary of basic Pfair scheduling. The rest of this section presents some additional definitions and results that we will use in this paper.

2.10. Active tasks

If subtasks are absent or are released late, then it is possible for a GIS (or IS) task to have no eligible subtasks and an allocation of zero during certain time slots. Tasks with and without subtasks in the interval $[t, t+\ell]$ are distinguished using the following definition of an active task.

Definition 1. A GIS task $U$ is active in slot $t$ if it has one or more subtasks $U_j$ such that $e(U_j) \leq t < d(U_j)$. (A task that is active in $t$ is not necessarily scheduled in that slot.)

2.11. Holes

If fewer than $M$ tasks are scheduled at time $t$ in $S$, then one or more processors are idle in slot $t$. For each slot, each processor that is idle in that slot is referred to as a hole. Hence, if $k$ processors are idle in slot $t$, then there are said to be $k$ holes in $t$. The following lemma is a generalization of one proved in [10], and relates an increase in the total lag of $\tau$, LAG, to the presence of holes.

Lemma 4. (See Srinivasan and Anderson [10].) If $\text{LAG}(\tau, t + \ell, S) > \text{LAG}(\tau, t, S)$, where $\ell \geq 1$, then there is at least one hole in the interval $[t, t + \ell]$.

Intuitively, if there is no idle processor in slots $t, \ldots, t + \ell - 1$, then the total allocation in $S$ in each of those slots to tasks in $\tau$ is equal to $M$. Since $\tau$ is assumed to be feasible, this is at least the total allocation that $\tau$ receives in any slot in the ideal schedule. Therefore, LAG cannot increase.

2.12. Task classification (from [10])

Tasks in $\tau$ may be classified as follows with respect to a schedule $S$ and time interval $[t, t + \ell]$. (For brevity, we let the task system $\tau$ and schedule $S$ be implicit in these definitions.)

$A(t, t + \ell)$: Set of all tasks that are scheduled in one or more slots in $[t, t + \ell]$.

$B(t, t + \ell)$: Set of all tasks that are not scheduled in any slot in $[t, t + \ell]$, but are active in one or more slots in the interval.

$I(t, t + \ell)$: Set of all tasks that are neither active nor scheduled in any slot in $[t, t + \ell]$.

As a shorthand, the notation $A(t)$, $B(t)$, and $I(t)$ is used when $\ell = 1$. $A(t, t + \ell)$, $B(t, t + \ell)$, and $I(t, t + \ell)$ form a partition of $\tau$, i.e., the following holds:

$$\begin{align*}
A(t, t + \ell) & \cup B(t, t + \ell) \cup I(t, t + \ell) = \tau, \\
A(t, t + \ell) & \cap B(t, t + \ell) = B(t, t + \ell) \cap I(t, t + \ell) = I(t, t + \ell) \cap A(t, t + \ell) = \emptyset.
\end{align*}$$

(22)

(23)

This classification of tasks is illustrated in Fig. 4(a) for $\ell = 1$. Using (16), (22), and (23) above, we have the following:

$$\text{LAG}(\tau, t + 1) = \sum_{T \in A(t)} \text{lag}(T, t + 1) + \sum_{T \in B(t)} \text{lag}(T, t + 1) + \sum_{T \in I(t)} \text{lag}(T, t + 1).$$

(24)

The next definition identifies the last-released subtask at $t$ of any task $U$.

Definition 2. Subtask $U_j$ is the critical subtask of $U$ at $t$ iff $e(U_j) \leq t < d(U_j)$ holds, and no other subtask $U_k$ of $U$, where $k > j$, satisfies $e(U_k) \leq t < d(U_k)$. For example, in Fig. 4(a), the critical subtask of $T$ at both $t - 1$ and $t$ is $T_{t+1}$, and that of $U$ at $t + 1$ is $U_{k+1}$.
In what follows, we prove the following theorem.

**Theorem 1.** Tardiness under EPDF can exceed three quanta for feasible GIS task systems. In particular, if EPDF is used to schedule task system $\tau_i$ ($1 \leq i \leq 3$) in Table 1, then a tardiness of $i + 1$ quanta is possible.

**Proof.** Fig. 5 shows a schedule for $\tau_1$, in which a subtask has a tardiness of two at time 50. The schedules for $\tau_2$ and $\tau_3$ are too lengthy to be depicted; we verified them using two independently-coded EPDF simulators.

The sufficient condition for a tardiness of $q > 0$ quanta as given by Srinivasan and Anderson requires that the sum of the weights of the $M - 1$ heaviest tasks be less than $\frac{q+2}{q+3}$. This can be ensured if the weight of each task is restricted to be at most $\frac{q}{q+1}$. We next show that a weight restriction of $\frac{q+2}{q+3}$ per task is sufficient to guarantee a tardiness of $q$ quanta. This restriction is stated below.

(W) The weight of each task is at most $\frac{q+2}{q+3}$.

In what follows, we prove the following theorem.
### Table 1
Counterexamples to show that tardiness under EPDF can exceed three.

<table>
<thead>
<tr>
<th>Task set</th>
<th>Util. (M)</th>
<th>Tardiness (in quanta)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># of tasks</td>
<td>weight</td>
</tr>
<tr>
<td>τ₁</td>
<td>4</td>
<td>1/2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3/4</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>23/24</td>
</tr>
<tr>
<td>τ₂</td>
<td>4</td>
<td>1/2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3/4</td>
</tr>
<tr>
<td></td>
<td>5</td>
<td>23/24</td>
</tr>
<tr>
<td></td>
<td>10</td>
<td>239/240</td>
</tr>
<tr>
<td>τ₃</td>
<td>4</td>
<td>1/2</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>3/4</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>23/24</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>31/32</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>119/120</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>239/240</td>
</tr>
<tr>
<td></td>
<td>6</td>
<td>479/480</td>
</tr>
<tr>
<td></td>
<td>8</td>
<td>959/960</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>1195/1200</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>2399/2400</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>4799/4800</td>
</tr>
</tbody>
</table>

![Fig. 5. Counterexample to prove that tardiness under EPDF can exceed one quantum. 13 periodic tasks with total utilization ten are scheduled on ten processors. In the schedule, tasks of the same weight are shown together as a group. Each column corresponds to a time slot. The PF-window of each subtask is shown as a sequence of dashes that are aligned. An integer value \( n \) in slot \( t \) means that \( n \) tasks in the corresponding group have a subtask scheduled at \( t \). Subtasks that miss deadlines are shown scheduled after their windows. In this schedule, 11 subtasks miss their deadlines at time 48. Hence, tardiness is 2 quanta for at least one subtask.](image)

**Theorem 2.** Tardiness under EPDF is at most \( q \) quanta, where \( q \geq 1 \), for every GIS task system that is feasible on \( M \geq 3 \) processors and satisfies (W).

We use a setup similar to that used by Srinivasan and Anderson in [10] and [11] to prove the above theorem. Though the setup is similar and some fundamental properties are applicable, there are significant differences in the core of the proof.

Our proof is by contradiction, hence, assume Theorem 2 does not hold. This assumption implies that there exists a \( q \geq 1 \), a time \( t_d \), and a concrete task system \( \sigma \) defined as follows.

**Definition 3.** \( t_d \) is the earliest deadline of a subtask with a tardiness of \( q + 1 \) under EPDF in any feasible GIS task system satisfying (W), i.e., there exists some such task system with a subtask with deadline at \( t_d \) and tardiness \( q + 1 \), and there does not exist any such task system with a subtask with deadline prior to \( t_d \) and a tardiness of \( q + 1 \).

**Definition 4.** \( \sigma \) is a feasible concrete GIS task system satisfying (W) with the following properties.

(S1) A subtask in \( \sigma \) with deadline at \( t_d \) has a tardiness of \( q + 1 \) under EPDF.
(S2) No feasible concrete task system satisfying (W) and (S1) releases fewer subtasks in \([0, t_d]\) than \( \sigma \).
In what follows, let $S'$ denote an EPDF schedule for $\sigma$ in which a subtask of $\sigma$ with deadline at $t_d$ has a tardiness of $q + 1$.

By (S1) and (S2), exactly one subtask in $\sigma$ has a tardiness of $q + 1$: if several such subtasks exist, then all but one can be removed and the remaining subtask will still have a tardiness of $q + 1$, contradicting (S2). Similarly, a subtask with deadline later than $t_d$ cannot impact how subtasks with deadlines at or before $t_d$ are scheduled. Therefore, no subtask in $\sigma$ has a deadline after $t_d$. Based on these facts, Lemma 6 below can be shown to hold. In proving Lemma 6, we use the following claim, proved in Appendix A.

Claim 1. There is no hole in any slot in $[t_d - 1, t_d + q)$ in $S'$.

We now show that $LAG$ of $\sigma$ at $t_d$ is exactly $qM + 1$.

Lemma 6. $LAG(\sigma, t_d, S') = qM + 1$.

Proof. By Claim 1, there is no hole in any slot in $[t_d, t_d + q)$ in $S'$. Further, the subtask with a tardiness of $q + 1$ and deadline at $t_d$, as specified in (S1), is not scheduled until time $t_d + q$. (Also, recall that there is exactly one such subtask.) Thus, because every subtask in $\sigma$ is scheduled at most $t_d$, there exist exactly $qM + 1$ subtasks with deadlines at most $t_d$ that are pending at $t_d$ in $S'$. In the ideal schedule, all of these subtasks complete executing by time $t_d$. Therefore, the $LAG$ of $\sigma$ at $t_d$, which is the difference between the ideal allocation and the allocation in $S'$ in $[0, t_d)$, is $qM + 1$. □

By Claim 1, there is no hole in slot $t_d - 1$. Therefore, by the contrapositive of Lemma 4, $LAG(\sigma, t_d - 1, S') \geq LAG(\sigma, t_d, S')$, which, by Lemma 6, is $qM + 1$. Thus, because $LAG(\sigma, 0, S') = 0$, there exists a time $t$, where $0 \leq t < t_d - 1$ such that $LAG(\sigma, t, S') < qM + 1$ and $LAG(\sigma, t + 1, S') \geq qM + 1$. This further implies the existence of a time $0 \leq t_h < t_d - 1$, a concrete task system $\tau$, and an EPDF schedule $S$ for $\tau$ defined as follows.

Definition 5. $t_h$, where $0 \leq t_h < t_d - 1$, is the earliest time such that the $LAG$ in any EPDF schedule for any feasible concrete GIS task system satisfying (W) is at least $qM + 1$ at $t_h + 1$.

Definition 6. $\tau$ is a feasible concrete GIS task system satisfying (W) with the following properties.

(T1) $LAG(\tau, t_h + 1, S) \geq qM + 1$.
(T2) No feasible concrete task system satisfying (W) and (T1) releases fewer subtasks than $\tau$.
(T3) No feasible concrete task system satisfying (W), (T1), and (T2) has a larger rank than $\tau$ where the rank of a task system is the sum of the eligibility times of all its subtasks, i.e., $rank(\tau, t) = \sum_{T_i \in \tau} e(T_i)$.

(T2) can be thought of as identifying a minimal task system in the sense of having $LAG$ exceed $qM + 1$ at the earliest possible time with the fewest number of subtasks, subject to satisfying (W). As already explained, if Theorem 2 does not hold for all task systems satisfying (W), then there exists some task system whose $LAG$ is at least $qM + 1$. Therefore, some task system satisfying (W), (T1), and (T2) necessarily exists. (T3) further restricts the nature of such a task system by requiring subtask eligibility times to be spaced as much apart as possible.

We next prove some properties about the subtasks of $\tau$ scheduled in $S$.

Lemma 7. Let $T_i$ be a subtask in $\tau$. Then, the following properties hold for $T_i$ in $S$.

(a) If $T_i$ is scheduled at $t$, then $e(T_i) \geq \min(t, t_h)$.
(b) If $T_i$ is scheduled before $t_d$, then the tardiness of $T_i$ is at most $q$.

Proof of part (a). Suppose $e(T_i)$ is not equal to $\min(t, t_h)$. Then, by (3) and because $T_i$ is scheduled at $t$, it is before $\min(t, t_h)$. Hence, simply changing $e(T_i)$ so that it equals $\min(t, t_h)$ will not affect how $T_i$ or any other subtask is scheduled. Therefore, the actual allocations in $S$ to every task, and hence, the lag of every task, will remain unchanged. Therefore, the $LAG$ of $\tau$ at $t_h + 1$ will still be at least $qM + 1$. However, changing the eligibility time of $T_i$ increases the rank of the task system, and hence, (T3) is contradicted. □

Proof of part (b). It follows from Definition 3. □

In what follows, we show that if (W) is satisfied, then there does not exist a time $t_h$ as defined in Definition 5, that is, we contradict its existence, and in turn prove Theorem 2. For this, we deduce the $LAG$ of $\tau$ at $t_h + 1$ by determining the lags of the tasks in $\tau$. But first, a brief digression on subtask categorization that will help improve the accuracy with which task lags are bound.
3.1. Categorization of subtasks

As can be seen from (13) and (6), the lag of a task \( T \) at \( t \) depends on the allocations that subtasks of \( T \) receive in each time slot until \( t \) in the ideal schedule. Hence, a tight estimate of such allocations is essential to bounding the lag of \( T \) reasonably accurately. If a subtask’s index is not known, then (6), which can otherwise be used to compute the allocation received by any subtask in any slot exactly, is not of much help. Hence, in this subsection, we define terms that will help in categorizing subtasks, and then derive upper bounds for the allocations that these categories of subtasks receive in certain slots in the ideal schedule.

3.1.1. \( k \)-dependent subtasks

The subtasks of a heavy task with weight in the range \([1/2, 1]\) can be divided into “groups” based on their group deadlines in a straightforward manner: place all subtasks with identical group deadlines in the same group and identify the group using the smallest index of any subtask in that group. For example, in Fig. 2, \( G_1 = \{ T_1, T_2 \} \), \( G_3 = \{ T_3, T_4, T_5 \} \), and \( G_6 = \{ T_6, T_7, T_8 \} \). If there are no IS separations or GIS omissions among the subtasks of a group, then a deadline miss by \( q \) quanta for a subtask \( T_i \) will necessarily result in a deadline miss by at least \( q \) quanta for the subsequent subtasks in \( T_i \)’s group. Hence, a subtask \( T_j \) is dependent on all prior subtasks in its group for not missing its deadline. If \( T \) is heavy, we say that \( T_j \) is \( k \)-dependent, where \( k \) is the \((k + 1)\)st subtask in its group, computed assuming all subtasks are present (that is, as in the determination of group deadlines, even if \( T \) is GIS and some subtasks are omitted, \( k \)-dependency is determined assuming there are no omissions).

Recall that by Lemma 1, all subtasks of a heavy task with weight less than one are of length two or three. Further, note that in each group, each subtask except possibly the first is of length two. This implies that for a periodic task the deadlines of any two successive subtasks that belong to the same group differ by exactly one. Also, in each group, each subtask except possibly the final subtask has a \( b \)-bit of one. Finally, if the final subtask of a group has a \( b \)-bit of one, then the first subtask of the group that follows is of length three. These properties are summarized in the following lemma.

Lemma 8. The following properties hold.

(a) Let \( T \) be a heavy task with \( wt(T) < 1 \) and let \( T_1 \) be a 0-dependent subtask of \( T \). Then, one of the following holds: (i) \( i = 1 \); (ii) \( b(T_{1-1}) = 0 \); (iii) \( |\omega(T_1)| = 3 \).
(b) If \( T_j \) is a \( k \)-dependent subtask of a periodic task \( T \), where \( i \geq 2 \) and \( k \geq 1 \), then \( d(T_j) = d(T_{i-1}) + 1 \) and \( r(T_j) = d(T_{i-1}) - 1 \).
(c) Let \( T_i \), where \( i > 1 \), be a \( k \)-dependent subtask of \( T \) with \( wt(T) < 1 \). If \( k \geq 1 \), then \( |\omega(T_i)| = 2 \) and \( b(T_{i-1}) = 1 \).

If a task \( T \) is light, then we simply define all of its subtasks to be 0-dependent. In this case, each subtask is in its own group.

3.1.2. Miss initiators

A subtask scheduled at \( t \) and missing its deadline by \( c \) quanta, where \( c \geq 1 \), is referred to as a \( \text{miss initiator by } c \) (or a \( c\)-MI, for short) for its group, if no subtask of the same task is scheduled at \( t - 1 \). A miss initiator by \( q \), i.e., a \( q\)-MI, will simply be referred to as an MI. Thus, a subtask is a \( c\)-MI if it misses its deadline by \( c \) quanta and is either the first subtask in its group to do so or separated from its predecessor by an IS or GIS separation, and its predecessor is not scheduled in the immediately preceding slot. Such a subtask is termed a miss initiator by \( c \) because in the absence of future separations, it causes all subsequent subtasks in its group to miss their deadlines by \( c \) quanta as well. Several examples of MIs for \( q = 1 \) are shown in Fig. 6.

3.1.3. Successors of miss initiators

The immediate successor \( T_{i+1} \) of a \( c\)-MI \( T_i \) is called a successor of a \( c\)-MI (or \( c\)-SMI, for short) if \( Tardiness(T_{i+1}) = \text{tardiness}(T_i) = c \), \( S(T_{i+1}, t) = 1 \) \( \Rightarrow \) \( S(T_i, t - 1) = 1 \), and \( T_i \) is a \( c\)-MI. (A successor of a miss initiator by \( q \), i.e., a \( q\)-SMI, will simply be referred to as an SMI.) Fig. 6 shows several examples for \( q = 1 \). Note that for \( T_{i+1} \) to be a \( c\)-SMI, its predecessor in \( S \) must be \( T_i \), rather than some lower-indexed subtask of \( T \). Also note that a \( c\)-SMI is at least 1-dependent.

The lemma below follows immediately from Lemma 8(a), which by (1) implies that the deadline of the first subtask of a group is greater than that of the final subtask of the preceding group by at least two.

Lemma 9. Let \( T_j \) be a subtask that is scheduled at \( t \) and let \( T_j \)’s tardiness be \( c > 0 \) quanta. If \( T_j \), where \( j < i \), is scheduled at \( t - 1 \) and \( T_j \) does not belong to the same dependency group as \( T_i \), then the tardiness of \( T_j \) is at least \( c + 1 \).

The next lemma bounds the allocation received by a \( k \)-dependent subtask in the first slot of its window in the ideal schedule, and is proved in Appendix A.
From the above, we have the following:

3.2. Subclassification of tasks in $T$ at time $t$ by an X. Solid (dotted) lines indicate slots that lie within (outside) the window of a subtask. A subtask scheduled in a dotted slot misses its deadline. In Fig. 6, possible schedules for the second job of (a) a periodic and (b) a GIS task of weight 7/9 under EPDF. Subtasks are scheduled in the slots marked.

Lemma 10. The allocation received by a $k$-dependent subtask in its first slot in the ideal schedule are as follows.

(a) The allocation $A(\text{ideal}, T_i, r(T_i))$ received in the ideal schedule by a $k$-dependent subtask $T_i$ of a periodic task $T$ with $wt(T) < 1$ in the first slot of its window is at most $k \cdot \frac{t}{r_p} - (k - 1) - \frac{t}{r_p}$, for all $k \geq 0$.

(b) The allocation $A(\text{ideal}, T_i, r(T_i))$ received in the ideal schedule by a $k$-dependent subtask $T_i$ of a GIS task $T$ in the first slot of its window is at most $k \cdot \frac{t}{r_p} - (k - 1) - \frac{t}{r_p}$, for all $k \geq 0$.

(c) Let $T_i$, where $i \geq k + 1$ and $k \geq 1$, be a subtask of $T$ with $wt(T) < 1$ such that $|\omega(T_i)| \geq 3$ and $b(T_{i-1}) = 1$. Let the number of subtasks in $T_{i-1}$'s dependency group be at least $k$. Then, $A(\text{ideal}, T_i, r(T_i)) \leq k \cdot \frac{t}{r_p} - (k - 1) - \frac{t}{r_p}$.

The next lemma bounds the lag of a task at time $t$, based on the $k$-dependency of its last-scheduled subtask. This is also proved in Appendix A.

Lemma 11. Let $T_i$ be a $k$-dependent subtask of a task $T$ for $k \geq 0$, and let the tardiness of $T_i$ be $s$ for some $s \geq 1$ (that is, $T_i$ is scheduled at time $d(T_i) + s - 1$). Then $\text{lag}(T, d(T_i) + s) < (k + s - 1) \cdot wt(T) - k$.

3.2. Subclassification of tasks in $A(t)$

Recall from Section 2 that a task in $A(t)$ is scheduled in slot $t$. We further classify tasks in $A(t)$, based on the tardiness of their subtasks scheduled at $t$, as follows.

$A_0(t)$: Includes $T$ in $A(t)$ iff its subtask scheduled at $t$ has zero tardiness.

$A_q(t)$: Includes $T$ in $A(t)$ iff its subtask scheduled at $t$ has a tardiness of $q$.

$A_{q-1}(t), q > 1$: Includes $T$ in $A(t)$ iff its subtask scheduled at $t$ has a tardiness greater than 0 but less than $q$.

$A_q(t)$ is further partitioned into $A^0_q(t)$, $A^1_q(t)$, and $A^2_q(t)$.

$A^0_q(t)$: Includes $T$ in $A_q(t)$ iff its subtask scheduled at $t$ is an MI.

$A^1_q(t)$: Includes $T$ in $A_q(t)$ iff its subtask scheduled at $t$ is an SMI.

$A^2_q(t)$: Includes $T$ in $A_q(t)$ iff its subtask scheduled at $t$ is neither an MI nor an SMI.

$A^0_{q-1}(t), q > 1$: Included in $A_{q-1}(t)$ iff its subtask scheduled at $t$ is a $c$-MI, where $0 < c < q$.

From the above, we have the following:

$$A_0(t) \cup A_q(t) \cup A_{q-1}(t) = A(t) \quad \text{and} \quad A^0_q(t) \cup A^1_q(t) \cup A^2_q(t) = A_q(t).$$

$$A_0(t) \cap A_q(t) = A_q(t) \cap A_{q-1}(t) = A_{q-1}(t) \cap A_0(t) = \emptyset.$$  

$$A^0_q(t) \cap A^1_q(t) = A^1_q(t) \cap A^2_q(t) = A^2_q(t) \cap A^0_q(t) = \emptyset.$$  

$$A_0(t) \cup A^0_q(t) = A^0_q(t) \cup A^0_{q-1}(t) = A^0_{q-1}(t) \cup A_0(t) = \emptyset.$$  

$$A^1_q(t) \cup A^1_{q-1}(t) = A^1_{q-1}(t) \cup A^1_q(t) = A^1_q(t) \cup A^1_{q-1}(t) = \emptyset.$$  

$$A^2_q(t) \cup A^2_{q-1}(t) = A^2_{q-1}(t) \cup A^2_q(t) = A^2_q(t) \cup A^2_{q-1}(t) = \emptyset.$$  

$$A^0_q(t) \cup A^1_q(t) = A^1_q(t) \cup A^2_q(t) = A^2_q(t) \cup A^0_q(t) = \emptyset.$$
3.3. Task lags by task classes and subclasses

By the definition there of $t_h$ in Definition 5, $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$. Hence, by Lemma 4, the following holds.

\( \square \)

(H) There is at least one hole in slot $t_h$.

The next lemma gives bounds on the lags of tasks in $A(t)$, $B(t)$, and $I(t)$ at $t + 1$, where $t \leq t_h$ is a slot with a hole, and hence, the lemma holds for $t = t_h$, as well.

**Lemma 12.** Let $t \leq t_h$ be a slot with a hole. Then, the following bounds hold for lag at $t + 1$ of a task $T$ depending on whether it is scheduled at $t$ and the type of its subtask scheduled at that time.

(a) (From [11].) For $T \in I(t)$, $\text{lag}(T, t + 1) = 0$.
(b) (From [11].) For $T \in B(t)$, $\text{lag}(B, t + 1) \leq 0$.
(c) (From [6].) For $T \in A_0(t)$, $\text{lag}(T, t + 1) < \text{wt}(T)$.
(d) For $T \in A_0^1(t)$, $\text{lag}(T, t + 1) < (q + 1) \cdot \text{wt}(T)$.
(e) For $T \in A_0^2(t)$, $\text{lag}(T, t + 1) < (q + 2) \cdot \text{wt}(T) - 1$.
(f) For $T \in A_1^2(t)$, $\text{lag}(T, t + 1) < (q + 3) \cdot \text{wt}(T) - 2$.
(g) For $T \in A_{q-1}^1(t)$, $\text{lag}(T, t + 1) < q \cdot \text{wt}(T)$.

**Proof.** Parts (a) and (b) are proved in [11]. To see why they hold, note that no task in $B(t)$ or $I(t)$ is scheduled at $t$. Because there is a hole in $t$, the critical subtask of a task in $B(t)$ is scheduled before $t$; similarly, the latest subtask of a task in $I(t)$ with release time at or before $t$ should have completed execution by $t$. Hence, such tasks cannot be behind with respect to the ideal schedule. Part (c) is proved in [6]. The remaining parts are proved below. \( \square \)

**Proof of part (d).** If $T \in A_0^1(t)$, then the subtask $T_i$ of $T$ scheduled at $t$ is an MIL, and $d(T_i) = t - q + 1$. Further $T_i$ is $k$-dependent, where $k \geq 0$. Hence, by Lemma 11, $\text{lag}(T, t + 1)$ is less than $(k + q + 1) \cdot \text{wt}(T) - k$, which (because $\text{wt}(T) \leq 1$) is at most $(q + 1) \cdot \text{wt}(T)$, for all $k \geq 0$. \( \square \)

**Proof of part (e).** If $T \in A_0^2(t)$, then the subtask $T_i$ of $T$ scheduled at $t$ is an SMI, and is $k$-dependent for $k \geq 1$. Also, $d(T_i) = t - q + 1$. Thus, by Lemma 11, $\text{lag}(T, t + 1) < (k + q + 1) \cdot \text{wt}(T) - k \leq (q + 2) \cdot \text{wt}(T) - 1$ for all $k \geq 1$ (because $\text{wt}(T) \leq 1$). \( \square \)

**Proof of part (f).** Similar to that of part (e). \( \square \)

**Proof of part (g).** Let $T_i$ be $T$’s subtask scheduled at $t$ and let $s$ denote the tardiness of $T_i$. Then, $t + 1 = d(T_i) + s$. Let $T_i$ be $k$-dependent, where $k \geq 0$. By the definition of $A_{q-1}$, $0 < s < q$ holds, and by Lemma 11, $\text{lag}(T, d(T_i) + s) = \text{lag}(T, t + 1) < (k + s + 1) \cdot \text{wt}(T) - k \leq (k + q) \cdot \text{wt}(T) - k \leq q \cdot \text{wt}(T)$, for all $k \geq 0$. \( \square \)

3.4. Some auxiliary lemmas

In proving Theorem 2, we also make use of the following three lemmas, established in prior work by Srinivasan and Anderson.

**Lemma 13.** (See Srinivasan and Anderson [10].) If $\text{LAG}(\tau, t + 1) > \text{LAG}(\tau, t)$, then $B(t) \neq \emptyset$.

The following is an intuitive explanation for why Lemma 13 holds. Recall from Section 2 that $B(t)$ is the set of all tasks that are active but not scheduled at $t$. Because $o(T_i) \leq r(T_i)$ holds, by Definition 1 and (6), only tasks that are active at $t$ may receive positive allocations in slot $t$ in the ideal schedule. Therefore, if every task that is active at $t$ is scheduled at $t$, then the total allocation in $S$ cannot be less than the total allocation in the ideal schedule, and hence, by (17), LAG cannot increase across slot $t$.

**Lemma 14.** (See Srinivasan and Anderson [10].) Let $t < t_d$ be a slot with holes and let $T \in B(t)$. Then, the critical subtask at $t$ of $T$ is scheduled before $t$.

To see that the above lemma holds, let $T_i$ be the critical subtask of $T$ at $t$. By its definition, the IS-window of $T_i$ overlaps slot $t$, but $T$ is not scheduled at $t$. Also, there is at least a hole in $t$. Because EPDF does not idle a processor while there is a task with an outstanding execution request, $T_i$ is thus scheduled before $t$. \( \square \)
Lemma 15. (See Srinivasan and Anderson [10].) Let $U_j$ be a subtask that is scheduled in slot $t'$, where $t' \leq t \leq t_h$ and let there be a hole in $t$. Then, $d(U_j) = t + 1$.

This lemma is true because it can be shown that if $d(U_j) > t + 1$ holds, then $U_j$ has no impact on how subtasks are scheduled after $t$. In particular, it can be shown that even if $U_j$ is removed, no subtask scheduled after $t$ can be scheduled at or before $t$. Therefore, it can be shown that if the lemma does not hold, then the GIS task system obtained from $t$ by removing $U_j$ also has a LAG at least $qM + 1$ at $t_h + 1$, which is a contradiction to (T2).

Arguments similar to those used in proving the above lemma can be used to show the following. This lemma is proved in [6].

Lemma 16. (From [6].) Let $t < t_d$ be a slot with holes. Let $U_j$ be a subtask that is scheduled at $t$ and let the tardiness of $U_j$ be zero. Then, $d(U_j) = t + 1$ and $b(U_j) = 1$.

In the rest of this subsection, we will establish three more lemmas for later use. But first, a couple of definitions.

Definition 7. $t_h$ denotes the latest time before $t_h$ at which the subtask that is critical at $t_h$ of any task in $B(t_h)$ is scheduled.

$U$ and $U_j$ are henceforth to be taken as defined below.

Definition 8 ($U$ and $U_j$). $U$ denotes a task in $B(t_h)$ with a subtask $U_j$ that is critical at $t_h$ scheduled at $t_h$.

The lemma below shows that the deadline of the critical subtask at $t_h$ of every task in $B(t_h)$ is at $t_h + 1$.

Lemma 17. Let $T$ be a task in $B(t_h)$ and let $T_i$ be $T$'s critical subtask at $t_h$. Then, $d(T_i) = t_h + 1$.

Proof. Because $T$ is in $B(t_h)$, $T$ is active at $t_h$, but is not scheduled at $t_h$. Hence, $T_i$, which is critical at $t_h$, should have been scheduled earlier. In this case, by Lemma 15, $d(T_i) \leq t_h + 1$ holds. However, since $T_i$ is $T$’s critical subtask at $t_h$, by Definition 2, $d(T_i) = t_h + 1$ holds. Therefore, $d(T_i) = t_h + 1$ follows. $\square$

The following lemma shows that at least one subtask scheduled in $t_h$ has a tardiness of zero, i.e., $|A_0(t_h)| \geq 1$. It is proved in Appendix A.

Lemma 18. There exists a subtask $W_e$ scheduled at $t_h$ with $e(W_e) \leq t_h$, $d(W_e) = t_h + 1$, and $S(W_e, t) = 0$, for all $t \in [t_h, t_h]$. Also, there is no hole in any slot in $[t_h, t_h]$. (Note that, by this lemma, $A_0(t_h) \neq \emptyset$.)

The next lemma establishes some properties with respect to a slot in which at least one MI is scheduled. It is also proved in Appendix A.

Lemma 19. Let $t_m \leq t_h$ be a slot in which an MI is scheduled. Then, the following hold.

(a) For all $t$, where $t_m = (q + 2) < t < t_m$, there is no hole in slot $t$, and for each subtask $V_k$ that is scheduled in $t$, $d(V_k) = t_m = q + 1$.

(b) Let $W$ be a task in $B(t_m)$ and let the critical subtask $W_e$ of $W$ at $t_m$ be scheduled before $t_m$. Then, $W_e$ is scheduled at or before $t_m = (q + 2)$.

3.5. Core of the proof

Having classified the tasks at $t_h$ and determined their lags at $t_h + 1$, we next show that if (W) holds, then LAG$(\tau, t_h + 1) < M + 1$ in each of the following cases.

For conciseness, in what follows, we denote subsets $A(t_h)$, $B(t_h)$, and $I(t_h)$ as $A$, $B$, and $I$, respectively. Subsets $A_q(t_h)$ and $A_q(t_h)$ and their subsets are similarly denoted without the time parameter.

Case A: $A_q = \emptyset$.

Case B: $A_q^0 \neq \emptyset$ or $(A_q^1 \neq \emptyset$ and $A_q^{0-1} \neq \emptyset)$.

Case C: $A_q^0 = \emptyset$ and $A_q^1 \neq \emptyset$ and $A_q^{0-1} = \emptyset$.

Case D: $A_q^0 = A_q^1 = \emptyset$. 
The following notation is used to denote subset cardinality:

\[
\begin{align*}
  a_0 &= |A_0|; & a_q &= |A_q|; & a_0^0 &= |A_0^0|; & a_1^1 &= |A_1^1|; & a_2^2 &= |A_2^2|; \\
  a_{q-1}^0 &= |A_{q-1}^0|; & a_{q-1} &= |A_{q-1}|.
\end{align*}
\]

\(h\) is defined as follows (there is no correspondence between \(h\) as defined here and the subscript \(h\) in \(t_h\). The subscript \(h\) in \(t_h\) is just an indication that \(t_h\) is a slot with holes):

\[h \overset{\text{def}}{=} \text{number of holes in } t_h.\]

Because there is at least one hole in \(t_h\)

\[h > 0. \tag{28}\]

In the remainder of this paper, let \(W_{\text{max}}\) denote the maximum weight of any task in \(\tau\). That is,

\[W_{\text{max}} = \max_{T \in \tau} \{\text{wt}(T)\}. \tag{29}\]

In each of the above cases, \(\text{LAG}(\tau, t_h + 1)\) can be expressed as follows:

\[
\text{LAG}(\tau, t_h + 1) = \sum_{T \in \tau} \text{lag}(T, t_h + 1)
\]

\[
\leq \sum_{T \in A_0} \text{lag}(T, t_h + 1) + \sum_{T \in A_{q-1}} \text{lag}(T, t_h + 1) + \sum_{T \in A_q^0} \text{lag}(T, t_h + 1)
\]

\[
+ \sum_{T \in A_q^1} \text{lag}(T, t_h + 1) + \sum_{T \in A_q^2} \text{lag}(T, t_h + 1) \quad (\text{by (22), (23), (25), and Lemmas } 12(a) \text{ and (b)})
\]

\[
< \sum_{T \in A_0} \text{wt}(T) + \sum_{T \in A_{q-1}} q \cdot \text{wt}(T) + \sum_{T \in A_q^0} (q + 1) \cdot \text{wt}(T)
\]

\[
+ \sum_{T \in A_q^1} ((q + 2) \cdot \text{wt}(T) - 1) + \sum_{T \in A_q^2} ((q + 3) \cdot \text{wt}(T) - 2) \quad (\text{by Lemmas } 12(c)-(g)).
\]

Using (29), \(\text{LAG}(\tau, t_h + 1)\) can be bounded as

\[
\text{LAG}(\tau, t_h + 1)
\]

\[
< a_0 \cdot W_{\text{max}} + a_{q-1} \cdot q \cdot W_{\text{max}} + a_q^0 (q + 1) W_{\text{max}} + a_q^1 (q + 2) W_{\text{max}} - 1 + a_q^2 (q + 3) W_{\text{max}} - 2 \tag{30}
\]

\[
\leq \begin{cases}
  a_0 \cdot W_{\text{max}} + a_q^0 (q + 1) W_{\text{max}} + a_q^1 (q + 2) W_{\text{max}} - 1 + (a_{q-1} + a_q^2) (q + 3) W_{\text{max}} - 2, & W_{\text{max}} \geq \frac{2}{3}, \\
  a_0 \cdot (2/3) + a_q^0 (q + 1) (2/3) + a_q^1 (q + 2) (2/3) - 1 + (a_{q-1} + a_q^2) (q \cdot (2/3)), & W_{\text{max}} < \frac{2}{3},
\end{cases}
\]

\[(\text{because } (q + 3) W_{\text{max}} - 2 \geq q \cdot W_{\text{max}} \text{ for } W_{\text{max}} \geq 2/3). \tag{31}\]

Note that though \((q + 3) W_{\text{max}} - 2 < q \cdot W_{\text{max}}\) holds for \(W_{\text{max}} < 2/3, (q + 3) \cdot (2/3) - 2 = (2/3) \cdot q > q \cdot W_{\text{max}}\) holds for all \(W_{\text{max}} < 2/3\). Therefore, if the values of \(a_0, a_{q-1}, a_q\) are not dependent on whether \(W_{\text{max}} \geq 2/3\) or \(W_{\text{max}} < 2/3\), determining a bound on \(\text{LAG}(\tau, t_h + 1)\) using the expression corresponding to \(W_{\text{max}} \geq 2/3\) in (31) (of course, assuming that \(W_{\text{max}} \geq 2/3\) serves as an upper bound for \(\text{LAG}\) when \(W_{\text{max}} < 2/3\). Hence, later in the paper, when \(a_0, a_{q-1}, a_q\) are not dependent on \(W_{\text{max}}\), we bound \(\text{LAG}(\tau, t_h + 1)\) in this way.

The total number of processors, \(M\), expressed in terms of the number of subtasks in each subset of \(A\) scheduled at \(t_h\), and the number of holes in \(t_h\), is as follows:

\[M = a_0 + a_{q-1} + a_q^0 + a_q^1 + a_q^2 + h. \tag{32}\]

3.6. Case A: \(A_q = \emptyset\)

Case A is dealt with as follows.

**Lemma 20.** If \(A_q = \emptyset\), then \(\text{LAG}(\tau, t_h + 1) < qM + 1\).
Proof. If $A_q = \emptyset$, then
\[
\text{LAG}(\tau, t_h + 1) < a_0 \cdot W_{\text{max}} + a_{q-1} \cdot q \cdot W_{\text{max}} \quad (\text{by (30)} \text{ and } a_0^0 = a_1^1 = a_2^2 = 0)
\]
\[
\leq a_0 \cdot q \cdot W_{\text{max}} + a_{q-1} \cdot q \cdot W_{\text{max}}
\]
\[
< (M - h) \cdot q \cdot W_{\text{max}} \quad (\text{by (32)}, a_0 + a_{q-1} = M - h \text{ for this case})
\]
\[
< qM + 1. \quad \Box
\]

Hence, if no subtask with a tardiness of $q$ is scheduled in $t_h$, then (T1) is contradicted.

3.7. Case B: $A_q^0 \neq \emptyset$ or $(A_1^1 \neq \emptyset$ and $A_{q-1}^0 \neq \emptyset$)

By Lemma 12(d), $\text{lag}(T, t_h + 1)$ could be as high as $(q + 1) \cdot \text{wt}(T)$, if the subtask $T_i$ of $T$ scheduled at $t_h$ is an MI, i.e., is in $A_q^0$. Therefore, if $A_q^0$ is large, then LAG at $t_h + 1$ could exceed $qM + 1$. However, as we show below, if the number of MIs and SMIs scheduled at $t_h$ is large, then the number of tasks that are inactive at $t_h$ is also large, which in turn be used to show that LAG does not increase across $t_h$. Specifically, we show that if $a_0^0 + a_1^1 > (q + 1)(h - 1)$, then LAG($\tau, t_h + 1$) $\leq$ LAG($\tau, t_h$) $< qM + 1$, contradicting (T1). (Otherwise, the number of MIs and SMIs is not large enough for LAG to equal or exceed $qM + 1$.)

We begin by giving a lemma concerning the sum of the weights of tasks in $I$.

Lemma 21. If $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$, then $\sum_{V \in I} \text{wt}(V) < h$.

Proof. By (17),
\[
\text{LAG}(\tau, t_h + 1) = \text{LAG}(\tau, t_h) + \sum_{T \in \tau} (A(\text{ideal}, T, t_h) - S(T, t_h))
\]
\[
= \text{LAG}(\tau, t_h) + \sum_{T \in A \cup B} (A(\text{ideal}, T, t_h)) - (M - h)
\]
\[
= \text{LAG}(\tau, t_h) + \sum_{T \in A \cup B} \text{wt}(T) - (M - h) \quad (\text{by (7)}).
\]

If $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$, then by the derivation above,
\[
\sum_{T \in A \cup B} \text{wt}(T) > M - h. \quad (33)
\]

By (5), (22), and (23), $\sum_{T \in I} \text{wt}(T) \leq M - \sum_{T \in A \cup B} \text{wt}(T)$, which by (33) implies that $\sum_{T \in I} \text{wt}(T) < h$. \Box

We next determine the largest number of MIs and SMIs that may be scheduled at $t_h$, for $\sum_{T \in I} \text{wt}(T) < h$ to hold. We begin with a lemma that gives the latest time that a subtask of a task in $B$ may be scheduled if $a_0^0 > 0$ or ($a_1^1 > 0$ and $a_{q-1}^0 > 0$).

Lemma 22. If $a_0^0 > 0$ (that is, an MI is scheduled at $t_h$), or ($a_1^1 > 0$ and $a_{q-1}^0 > 0$) (that is, an SMI, and a c-MI, where $0 < c < q$, is scheduled at $t_h$), then subtask $U_j$ defined by Definition 8 is scheduled no later than $t_h - (q + 2)$, i.e., $t_h \leq t_h - (q + 2)$.

Proof. If $a_0^0 > 0$ holds, then this lemma is immediate from Definitions 8, 7, and Lemma 19(b). (Note that Definitions 8 and 7 imply that $U_j$ is scheduled before $t_h$.)

If $a_{q-1}^0 > 0$ holds, then a c-MI, where $0 < c < q$, say $T_j$, is scheduled at $t_h$. Hence, $d(T_j) = t_h + 1 - c \leq t_h$ holds. By the definition of c-MI, the predecessor of $T_j$ is not scheduled at $t_h - 1$. Hence, the deadline of every subtask scheduled at $t_h - 1$ is at most $t_h$. By Definition 2, $d(U_j) \geq t_h + 1$. Therefore, $U_j$ is not scheduled at $t_h - 1$.

If $a_1^1 > 0$ holds, then an SMI is scheduled at $t_h$, and its predecessor, which is an MI, is scheduled at $t_h - 1$. Therefore, by Lemma 19(b), $U_j$ is not scheduled in $[t_h - 1 - (q + 1), t_h - 1) = [t_h - (q + 2), t_h - 1)$.

Thus, if both $a_{q-1}^0 > 0$ and $a_1^1 > 0$ hold, $U_j$ is not scheduled later than $t_h - (q + 3)$. \Box

The lemma that follows is used to identify tasks that are inactive at $t_h$.

Lemma 23. Let $T$ be a task that is not scheduled at $t_h$. If $T$ is scheduled in any of the slots in $[t_h + 1, t_h)$, then $T$ is in $I$. 
**Proof.** $T$ clearly is not in $A$. Because $T$ is scheduled in $[t_b + 1, t_h)$, $T$ is also not in $B$, by Definition 7. □

In the rest of this subsection, we let $s$ denote the number of slots in $[t_b + 1, t_h)$. That is,

$$s \equiv t_h - t_b - 1 \geq q + 1 \quad \text{(by Lemma 22).} \tag{34}$$

We now determine a lower bound on the number of subtasks of tasks in $I$ that may be scheduled in $[t_b + 1, t_h)$ as a function of $a_0^h, a_1^h, h,$ and $s$. For this purpose, we assign subtasks scheduled in $[t_b + 1, t_h)$ to processors in a systematic way. This assignment is only for accounting purposes; subtasks need not be bound to processors in the actual schedule.

### 3.7.1 Processor groups

The assignment of subtasks to processors is based on the tasks scheduled at $t_h$. The $M$ processors are partitioned into four disjoint sets, $P_1$, $P_2$, $P_3$, and $P_4$, based on the tasks scheduled at $t_h$, as follows.

**$P_1$:** By Lemma 18, there is at least one subtask $W_i$ scheduled at $t_h$ such that $e(W_i) \leq t_b$ and $S(W, t) = 0$, for $t$ in $[t_b, t_h)$. We assign one such subtask to the lone processor in this group. Hence, $|P_1| = 1$.

**$P_2$:** The $h$ processors that are idle at $t_h$ comprise this group. Thus, $|P_2| = h$.

**$P_3$:** This group consists of the $a_0^q + a_1^q$ processors on which the $a_0^q$ MIs and $a_1^q$ SMIs are scheduled. Because either $a_0^q > 1$ or $a_1^q > 1$ holds, $|P_3| > 1$. $\tau^3$ denotes the subset of all tasks scheduled on processors in $P_3$ at $t_h$.

**$P_4$:** Processors not assigned to $P_1$, $P_2$, or $P_3$ belong to this group. $\tau^4$ denotes the subset of all tasks scheduled on processors in $P_4$ at $t_h$.

### 3.7.2 Subtask assignment in $[t_b + 1, t_h)$

Subtasks scheduled in $[t_b + 1, t_h)$ are assigned to processors by the following rules. Tasks in $\tau^3$ and $\tau^4$ are assigned to the same processor that they are assigned to in $t_h$, in every slot in which they are scheduled in $[t_b + 1, t_h)$. (It is trivial that such an assignment is possible since by the processor groups defined above, $|\tau^3| + |\tau^4| = P_3 + P_4 \leq M - h - 1 < M$.) Subtasks of tasks not in $\tau^3$ or $\tau^4$ may be assigned to any processor.

The next three lemmas bound the number of subtasks of tasks in $T$ scheduled in $[t_b + 1, t_h)$. These lemmas assume that the assignment of subtasks to processors in $[t_b + 1, t_h)$ follows the rules described above. In these lemmas we assume that either $a_0^q > 1$ or $(a_0^q \geq 1$ and $a_1^q \geq 1$) holds.

**Lemma 24.** The number of subtasks of tasks in $I$ that are scheduled in $[t_b + 1, t_h)$ is at least $s \cdot (h + 1) + (a_0^q + a_1^q)$.

**Proof.** We first make the following two claims.

**Claim 2.** Let $T_i$ be a subtask assigned to a processor in $P_1$ or $P_2$ in $[t_b + 1, t_h)$. Then, $T$ is in $I$.

**Proof.** By our assignment of subtasks to processors, tasks assigned to processors in $P_1$ or $P_2$ in $[t_b + 1, t_h)$ are not scheduled at $t_h$. Therefore, $T$ is not scheduled at $t_h$. Hence, by Lemma 23, $T$ is inactive at $t_h$, i.e., is in $I$. □

**Claim 3.** At least one of the subtasks assigned to each processor in $P_3$ in $[t_b + 1, t_h)$ is a subtask of a task in $I$.

**Proof.** Let $P_3$ be any processor in $P_3$, and let $T_i$ be the subtask scheduled on $P_3$ at $t_h$. Then, $T_i$ is either an MI or an SMI. In the former case, by the definition of an MI, $S(T, t_h - 1) = 0$, and in the latter, by the definition of an SMI, $S(T, t_h - 2) = 0$. By Lemma 22, $t_h \leq t_h - (q + 2)$. Thus, since $q \geq 1$, and by Lemma 18, there is no hole in any slot in $[t_h, t_h)$, there is no hole in the slot $t_h - 2$ or $t_h - 1$. Thus, a subtask of a task $T$ other than $T$ is assigned to $P_3$ in one of these two slots. By our subtask assignment, $T$ is not scheduled at $t_h$; thus, by Lemma 23, $T \in I$. □

The lemma follows from the definition of $s$ in (34), and Claims 2 and 3 above. □

**Lemma 25.** The sum of the weights of the tasks in $I$ is at least

$$\frac{(h+1)q}{2 + q + T} + \frac{a_0^q + a_1^q}{2 + q + T}.$$ 

**Proof.** Let $V_k$ be a subtask of a task $V$ in $I$ that is scheduled in $[t_b + 1, t_h)$. Then, by Definition 1, $d(V_k) \leq t_h$. By Definition 8, $U_j$ is scheduled at $t_b$ and by Definition 2, $d(U_j) \geq t_b + 1$. Because $V_k$ with an earlier deadline than $U_j$ is scheduled later than $t_b$, either $r(V_k) > t_b + 1$ or $V_k$’s predecessor $V_j$, where $j < k$, is scheduled at $t_b$. In the latter case, by Lemma 7(b), tardiness($V_j$) $\leq q$, and hence, $d(V_j) \geq t_b - q + 1$, which, by Lemma 2, implies $r(V_k) \geq t_b - q$. Thus, we have the following:

$$\forall V_k: V \in I \iff ((u \in [t_b + 1, t_h) \land S(V_k, u) = 1) \implies (r(V_k) \geq t_b - q \land d(V_k) \leq t_h)). \tag{35}$$
We next show that $\text{wt}(V) \geq \frac{V.n}{s+q+1}$, where $V.n$ is the number of subtasks of $V$ scheduled in $[t_b + 1, t_h]$. Let $V_k$ and $V_\ell$ denote the first and final subtasks of $V$ scheduled in $[t_b + 1, t_h]$. Then, by (35), $r(V_k) \geq t_b - q$ and $d(V_\ell) \leq t_h$. Hence,

$$d(V_\ell) - r(V_k) \leq t_h - t_b + q = s + q + 1 \quad \text{(by the definition of $s$ in (34)).} \tag{36}$$

By (1),

$$d(V_\ell) - r(V_k) = \left[ \frac{\ell}{\text{wt}(V)} \right] - \left[ \frac{k - 1}{\text{wt}(V)} \right] + \Theta(V_\ell) - \Theta(V_k) \geq \left[ \frac{\ell}{\text{wt}(V)} \right] - \left[ \frac{k - 1}{\text{wt}(V)} \right] \quad \text{(by } \ell > k \text{ and (2)).} \tag{37}$$

By (36) and (37), we have $\left[ \frac{\ell}{\text{wt}(V)} \right] - \left[ \frac{k - 1}{\text{wt}(V)} \right] \leq s + q + 1$, which implies $\frac{\ell}{\text{wt}(V)} - \frac{k - 1}{\text{wt}(V)} \leq s + q + 1$, i.e.,

$$\text{wt}(V) \geq \frac{\ell - k + 1}{s + q + 1} \geq \frac{V.n}{s + q + 1} \quad \text{(because } V.n = \ell - k + 1 \text{ if } V \text{ is periodic and } V.n \leq \ell - k + 1 \text{ if } V \text{ is IS or GIS).}$$

Therefore, we have $\sum_{\forall \ell} \text{wt}(V) \geq \frac{V.n}{s + q + 1} \geq \frac{(h + 1)s}{s + q + 1} + \frac{a_0^0 + a_0^1}{s + q + 1}$, where the last inequality is by Lemma 24. □

**Lemma 26.** If $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$ and either $a_q^0 \geq 1$ or $(a_q^1 \geq 1$ and $a_{q-1}^0 \geq 1$), then $a_q^0 + a_q^1 \leq \min((h - 1)(q + 1) - 1, M - h - 1)$.

**Proof.** By Lemma 21, if $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$, then $\sum_{\forall \ell} \text{wt}(V) < h$. By Lemma 25, $\frac{(h + 1)s}{s + q + 1} + \frac{a_0^0 + a_0^1}{s + q + 1} \leq \sum_{\forall \ell} \text{wt}(V)$. Therefore, $\frac{(h + 1)s}{s + q + 1} + \frac{a_0^0 + a_0^1}{s + q + 1} < h$, which implies that

$$a_q^0 + a_q^1 < h(q + 1) - s \leq h(q + 1) - (q + 1) \quad \text{(by (34))} \tag{38}$$

$$= (h - 1)(q + 1). \tag{39}$$

Also, there are $h$ holes in $t_h$, and by Lemma 18, $a_0 \geq 1$. Therefore, by (32),

$$a_q^0 + a_q^1 \leq M - h - 1. \tag{39}$$

Eqs. (38) and (39) imply that $a_q^0 + a_q^1 \leq \min((h - 1)(q + 1) - 1, M - h - 1)$. □

We now conclude Case B by establishing the following.

**Lemma 27.** If $a_q^0 > 0$ or $(a_q^1 > 0$ and $a_{q-1}^0 > 0$), then $\text{LAG}(\tau, t_h + 1) < qM + 1$.

**Proof.** Because $W_{\text{max}}^\tau < 1$, assuming $W_{\text{max}} \geq 2/3$ (because, as discussed earlier, $a_0$, $a_{q-1}$, and $a_q^i$ are not dependent on $W_{\text{max}}$, by (31), we have

$$\text{LAG}(\tau, t_h + 1) < a_0 \cdot W_{\text{max}} + ((q + 1) \cdot W_{\text{max}}) \cdot (a_q^0 + a_q^1) + (a_{q-1}^0 + a_{q-1}^2) \cdot ((q + 3) \cdot W_{\text{max}} - 2). \tag{40}$$

By Lemma 26, if $\text{LAG}(\tau, t_h + 1) > \text{LAG}(\tau, t_h)$, then $a_q^0 + a_q^1 \leq \min(h - 1)(q + 1) - 1, M - h - 1)$. By Lemmas 12(a)-(g) (and as can be seen from the coefficients of the $a_i$ terms in (40)), the lag bounds for tasks in $A_q^0 \cup A_q^1$ are higher than those for the other tasks. Hence, $\text{LAG}(\tau, t_h + 1)$ is maximized when $a_q^0 + a_q^1 = \min((h - 1)(q + 1) - 1, M - h - 1)$. We assume this is the case. Note that

$$\min((h - 1)(q + 1) - 1, M - h - 1) = \begin{cases} (h - 1)(q + 1) - 1, & h \leq \frac{M + 1 + q}{q + 2}, \\ M - h - 1, & \text{otherwise}. \end{cases} \tag{41}$$

Based on (41), we consider two cases.

**Case 1:** $h > \frac{M + 1 + q}{q + 2}$. For this case, $\text{LAG}$ is maximized when $a_q^0 + a_q^1 = M - h - 1$, and hence, by (32), $a_q + a_{q-1} + a_q^2 = M - h - (a_q^0 + a_q^1) = 1$. Because, by Lemma 18, $a_0 > 0$, we have $a_0 = 1$, and hence, $a_{q-1} = a_q^2 = 0$. Substituting $a_0 = 1$, $a_q^2 = a_{q-1} = 0$, and $a_q^0 + a_q^1 = M - h - 1$ in (40), we have $\text{LAG}(\tau, t_h + 1) < W_{\text{max}} + (q + 1) \cdot W_{\text{max}} \cdot (a_q^0 + a_q^1) = W_{\text{max}} + (q + 1) \cdot W_{\text{max}} \cdot (M - h - 1) < W_{\text{max}} + (q + 1) \cdot W_{\text{max}} \cdot (M - \frac{M + 1 + q}{q + 2} - 1)$ (where the last inequality is by the condition of
Case 1, namely, $h > \frac{M+1+b}{q+2}$. If $qM + 1 \leq \text{LAG}(\tau, t_h + 1)$, then $W_{\text{max}}(q + 1) \cdot W_{\text{max}}(M - \frac{M+1+b}{q+2} - 1) > qM + 1$, which implies that $W_{\text{max}} > \frac{q+2}{q+1}$ for all $q \geq 1$ and $M \geq 2$. This contradicts (W), and hence, $\text{LAG}(\tau, t_h + 1) < qM + 1$.

Case 2: $h \leq \frac{M+1+b}{q+2}$. For this case, $\text{LAG}$ is maximized when $a_0 + a_1 = (h-1)(q+1) - 1$. By (32), we have $a_{q-1} + a_q = M - h - (a_0 + a_0) - a_1) = M - h - a_0 - (hq + h - q - 2)$. Therefore, by (40),

$$\text{LAG}(\tau, t_h + 1) < a_0 \cdot W_{\text{max}}(q + 1) \cdot W_{\text{max}}(a_0 + a_1) + ((q + 3) \cdot W_{\text{max}} - 2) - (a_{q-1} + a_q)^1$$

$$= a_0 \cdot W_{\text{max}}(q + 1) \cdot W_{\text{max}}((hq + h - q - 2)$$

$$+ ((q + 3) \cdot W_{\text{max}} - 2)(M - 2h - a_0 - hq + q + 2).$$

(42)

If $qM + 1 \leq \text{LAG}(\tau, t_h + 1)$, then the expression on the right-hand side of (42) exceeds $qM + 1$, which implies that $W_{\text{max}} > \frac{q+2}{q+1}$. Let $f = \frac{q+2}{q+1}$. Then $f = (q+2) + (M+2a_0 + h - 2h - 2a_0 - 2qh)$, and let $Y$ denote the denominator, $(q + 3)M + 2q + 4 - 5h - (2 + q)a_0 + 2q - 3h$, of $f$. To show that the lemma holds for this case, we show that unless $W_{\text{max}}$ exceeds $\frac{q+2}{q+1}$, $qM + 1 > \text{LAG}(\tau, t_h + 1)$. For this purpose, we determine a lower bound to the value of $f$. Note that for a given number of processors, $M$, and tardiness, $q$, $f$ varies with $a_0$ and $h$. Because $a_0 + a_1 = (h-1)(q+1) - 1$, we have $h > \frac{q+2}{q+1}$, hence, because $h$ is integral, $h \geq 2$ holds. The first derivative of $f$ with respect to $h$ is $\frac{M(q^2+q-2)+a_0(q^2+q-2)+2q^2+5q+9}{(q+2)(q+1)}$, which is non-negative for all $a_0 \geq 0$, and that with respect to $a_0$ is $\frac{M(q^2+q-2)+4(q^2+q-2)+2q^2+5q+9}{(q+2)(q+1)}$, which is non-negative for $h \leq \frac{M(q^2+q-2)+4(q^2+q-2)+2q^2+5q+9}{(q+2)(q+1)}$. Thus, $f$ is minimized when $h = 2$, and because $\frac{M(q^2+q-2)+4(q^2+q-2)+2q^2+5q+9}{(q+2)(q+1)} \geq 1$ (where $M+1+q > h$ holds for this case), when $a_0 = 1$. When $h = 2$ and $a_0 = 1$, hold, $f = \frac{M+2a_0 + h - 2h - 2a_0 - 2qh}{(q+2)(q+1)} > \frac{q+2}{q+1}$ for all $M$ (since when $h = 2$ and $a_0 = 1$, we have $M \geq 4$). Hence, $W_{\text{max}} > \frac{q+2}{q+1}$, which is a violation of (W), and the lemma follows for this case.

Thus, if an MI or an SMI and a $c$-MI are scheduled in $t_h$, then (T1) is contradicted.

3.8. Case C: ($A^0_0 = \emptyset$ and $A^1\emptyset \neq \emptyset$ and $A^0_{q-1} = \emptyset$)

For this case, we show that if $\text{LAG}(\tau, t_h + 1, S) > qM + 1$, then there exists another concrete task system $\tau'$, obtained from $\tau$ by removing certain subtasks, such that $\text{LAG}(\tau') = t_h - 1$ in an EPDF schedule $S'$ is greater than $qM + 1$ contradicting the minimality of $t_h$ (in Definition 5). Our approach is to identify task subsets, determine the lag for tasks in each subset in $S'$ at $t_h - 1$, and use task lags to determine the LAG of $\tau'$ at $t_h - 1$. We begin by defining needed subsets of subtasks and tasks.

In this case, since no MI is scheduled in slot $t_h$, $t_h$ (in Definition 7) can be as late as $t_h - 1$. This is stated below:

$$t_h \leq t_h - 1.$$ 

(43)

Let $t'_h$ be defined as follows.

Definition 9. $t'_h$ denotes the latest time, if any, before $t_h - 1$ that a subtask with deadline at or after $t_h$ is scheduled.

Since at least one SMI is scheduled at $t_h$, at least one MI is scheduled at $t_h - 1$. Therefore, by Lemma 19(a), the following holds.

(C) The deadline of every subtask scheduled in any slot in $[t_h - (q + 2), t_h - 1)$ is at or before $t_h - q$.

Since $q \geq 1$ holds, (C) implies the following:

when it exists, $t'_h \leq t_h - (q + 3)$.

(44)

Let $t^1_s$ through $t^3_s$ be subsets of subtasks defined as follows. In the definitions that follow, when we say that $T_i$ is ready at $t'_s$, we mean that $o(T_i) \leq t'_s$, and $T_i$’s predecessor, if any, is scheduled before $t'_s$.

$$t^1_s \overset{\text{def}}{=} \{T_i \mid T_i \text{ is either the critical subtask at } t_h \text{ of a task in } B(t_h) \text{ or the critical subtask at } t_h - 1 \text{ of a task in } B(t_h - 1), t'_s \text{ exists, } T_i \text{ is scheduled at or before } t'_s, \text{ and } T \text{ is not scheduled at } t_h\},$$

$$t^2_s \overset{\text{def}}{=} \{T_i \mid d(T_i) \equiv t_h, T_i \text{ is scheduled at } t_h - 1, \text{ and } T \text{ is not scheduled at } t_h\},$$

$$t^3_s \overset{\text{def}}{=} \{T_i \mid T \in A_0(t_h), T_i \text{ is scheduled at } t_h, \text{ and } T_i \text{ is ready at or before } t_h - (q + 3) \text{ in } S\}.$$
Lemma 33. Let $\tau$ be a task with a subtask in $\tau^i$, for all $1 \leq i \leq 8$. Note that $\tau^7 \subseteq \tau^1$ and $\tau^8 \subseteq \tau^2$ hold.

The following lemma establishes some properties concerning the subsets of subtasks and tasks defined above. A proof can be found in Appendix A.

Lemma 28. The following properties hold for subsets $\tau^i$ and $\tau^j$ defined above, where $1 \leq i \leq 8$.

(a) For every task $T$, there is at most one subtask in $(\tau^1 \cup \tau^2 \cup \tau^5)$.
(b) Let $T$ be scheduled at $t_h$ be the subtask of a task $T$ in $A_q(t_h)$ or $A_{q-1}(t_h)$. Then, $T$ is in $\tau^5$.
(c) $\tau^2 \subseteq \tau^1$ and $\tau^8 \subseteq \tau^2$.
(d) Subsets $\tau^i$, where $1 \leq i \leq 6$, are pairwise disjoint.

Let $\tau^R$ define $\{T_i \mid T_i \in A_q(t_h), T_i$ is scheduled at $t_h, T_i$ is not ready at or before $t_h - (q + 3)$ in $S\}$.

and let $\tau'$ be a concrete GIS task system obtained from $\tau$ by removing all the subtasks in $\tau^R$. Let $S'$ be an EPDF schedule for $\tau'$ such that ties among subtasks with equal deadlines are resolved in the same way as they are resolved in $S$. Our goal is to show that $\text{LAG}(\tau', t_h - 1, S') \geq qM + 1$, and derive a contradiction to the minimality of $t_h$ in Definition 5. For this purpose, in the next few lemmas (proved in Appendix A), we establish lag bounds for $S'$ for tasks with subtasks in the subsets defined above. We will denote the ideal schedule for $\tau$ as $\text{ideal}^\tau$ and that for $\tau'$ as $\text{ideal}^{\tau'}$.

Lemma 29. Let $T$ be a task with a subtask in $\tau^1$ or $\tau^2$. Then, $\text{lag}(T, t_h - 1, S') = \text{lag}(T, t_h + 1, S)$.

Lemma 30. Let $T$ be a task with a subtask in $\tau^3$. Then, $\text{lag}(T, t_h - 1, S') > \text{lag}(T, t_h + 1, S) - 1/(q + 2)$.

Lemma 31. Let $T$ be a task with a subtask in $\tau^4$. Then, $\text{lag}(T, t_h - 1, S') \geq \text{lag}(T, t_h + 1, S) - 2 \cdot W_{\text{max}} + 1$.

Lemma 32. Let $T$ be a task with a subtask in $\tau^5$. Then, $\text{lag}(T, t_h - 1, S') \geq \text{lag}(T, t_h + 1, S) + 2 - 2 \cdot W_{\text{max}}$.

Lemma 33. Let $T$ be a task with a subtask in $\tau^6$. Then, $\text{lag}(T, t_h - 1, S') = \text{lag}(T, t_h + 1, S)$.

Let $\tau^c = \tau' \setminus (\bigcup_{i=1}^{6} \tau_i)$. Because $\tau$ and $\tau'$ are concrete instantiations of the same non-concrete task system, they both contain the same tasks, and hence, $\tau^c = \tau \setminus (\bigcup_{i=1}^{6} \tau_i)$. We show the following concerning the lag of a task in $\tau^c$ at $t_h - 1$ in $S'$. (This lemma is also proved in Appendix A.)

Lemma 34. Let $T$ be a task in $\tau^c$. Then, $\text{lag}(T, t_h - 1, S') = \text{lag}(T, t_h + 1, S)$.

Having determined bounds for the lags of tasks at $t_h - 1$ in $S'$, we now determine a lower bound for the LAG of $\tau'$ at $t_h - 1$ in $S'$, and show that if $W$ holds, then $\text{LAG}(\tau', t_h - 1, S') \geq qM + 1$.

Lemma 35. If either $W_{\text{max}} \leq \frac{q+3}{2q+4}$ and $a_0 \leq \frac{(M-h)(q+1)}{q+2}$ or $(W_{\text{max}} > \frac{q+3}{2q+4}$ and $a_0 \leq 2(M - h)(1 - W_{\text{max}})$), then $\text{LAG}(\tau', t_h - 1, S') \geq qM + 1$.

Proof. By (16),

$$\text{LAG}(\tau', t_h - 1, S') = \sum_{T \in \tau'} \text{lag}(T, t_h - 1, S')$$

(by the construction of $\tau'$)
\[ \begin{align*}
&= \sum_{i=1}^{6} \sum_{\tau \in T^i} \log(T, t_h - 1, S^i) + \sum_{\tau \in T^i} \log(T, t_h - 1, S^i) \\
&\quad \quad \text{(by Lemmas 28(c) and (d), and because } \tau^i \tau = \tau \setminus \bigcup_{i=1}^{6} T^i) \\
&\geq \sum_{\tau \in \tau^{1 \cap \tau^{2 \cap \cdots \cap \tau^{6}}} T^i} \log(T, t_h + 1, S) + \sum_{i=3}^{5} \sum_{\tau \in T^i} \log(T, t_h - 1, S') \\
&\quad \quad \text{(by Lemmas 29, 33, and 34)} \\
&\geq \sum_{i=1}^{6} \sum_{\tau \in T^i} \log(T, t_h + 1, S) + \sum_{\tau \in T^i} \log(T, t_h + 1, S) - |\tau^3| \cdot \frac{1}{q + 2} \\
&\quad + |\tau^4| \cdot (2W_{\max} - 1) + |\tau^5| \cdot (2 - 2W_{\max}) \quad \text{(by Lemmas 30–32)} \\
&= LAG(\tau, t_h + 1, S) - |\tau^3| \cdot \frac{1}{q + 2} - |\tau^4| \cdot (2W_{\max} - 1) + |\tau^5| \cdot (2 - 2W_{\max}) \\
&\quad \quad \text{(by the definitions of sets } \tau^i, \text{ where } 1 \leq i \leq 6, \text{ and } \tau^i). \quad \text{(46)}
\end{align*} \]

Note that
\[ |\tau^3| + |\tau^4| = a_0. \quad \text{(47)} \]

By Lemma 28(b), \(|\tau^5| = |A_q| + |A_{q-1}| = a_q + a_{q-1} - 1\). By the definitions of \(A_q, A_q^0, A_q^1, A_q^2\), and by (25)–(27), \(a_q = a_q^0 + a_q^1 + a_q^2\). However, because no MI is scheduled at \(t_h\) by the conditions of Case C, \(a_q^0 = 0\), and hence,
\[ |\tau^5| = a_q^1 + a_q^2 + a_{q-1} = M - h - a_0 \quad \text{(by (32))}. \quad \text{(48)} \]

We now consider the following two cases based on the statement of the lemma.

**Case 1:** \(W_{\max} > \frac{q + 3}{2q + 4}\) and \(a_0 \leq 2(M - h)(1 - W_{\max})\). Since \(W_{\max} > \frac{q + 3}{2q + 4}, 2W_{\max} - 1 > \frac{1}{q + 2}\) holds. By (46),
\[ LAG(\tau', t_h - 1, S') \geq LAG(\tau, t_h + 1, S) - |\tau^3| \cdot \frac{1}{q + 2} - |\tau^4| \cdot (2W_{\max} - 1) + |\tau^5| \cdot (2 - 2W_{\max}) \]
\[ \quad \quad \text{(because as mentioned above, } 2W_{\max} - 1 > \frac{1}{q + 2}) \]
\[ = LAG(\tau, t_h + 1, S) - a_0 \cdot (2W_{\max} - 1) + (M - h - a_0) \cdot (2 - 2W_{\max}) \quad \text{(by (47) and (48))} \]
\[ = LAG(\tau, t_h + 1, S) - a_0 + (M - h) \cdot (2 - 2W_{\max}) \quad \text{(by (47))} \]
\[ \geq LAG(\tau, t_h + 1, S) \quad \text{(because } 2(M - h) \cdot (1 - W_{\max}) \geq a_0 \text{ for this case}) \]
\[ \geq qM + 1 \quad \text{(by (T1))}. \quad \text{(49)} \]

**Case 2:** \(W_{\max} < \frac{q + 3}{2q + 4}\) and \(a_0 \leq \frac{(M - h)(q + 1)}{q + 2}\). Since \(W_{\max} \leq \frac{q + 3}{2q + 4}, 2W_{\max} - 1 \leq \frac{1}{q + 2}\) holds. As with Case 1, by (46),
\[ LAG(\tau', t_h - 1, S') \geq LAG(\tau, t_h + 1, S) - |\tau^3| \cdot \frac{1}{q + 2} - |\tau^4| \cdot (2W_{\max} - 1) + |\tau^5| \cdot (2 - 2W_{\max}) \]
\[ \quad \quad \text{(because } 2W_{\max} \leq 1 \leq \frac{1}{q + 2}) \]
\[ \geq LAG(\tau, t_h + 1, S) - |\tau^3| \cdot \frac{1}{q + 2} - |\tau^4| \cdot \frac{1}{q + 2} + |\tau^5| \cdot (2 - 2W_{\max}) \]
\[ \quad \quad \text{(because } W_{\max} \leq \frac{q + 3}{2(q + 2)}) \]
\[ = LAG(\tau, t_h + 1, S) - a_0 \cdot \frac{1}{q + 2} + (M - h - a_0) \cdot \frac{q + 1}{q + 2} \quad \text{(by (47) and (48))} \]
\[ = LAG(\tau, t_h + 1, S) - a_0 + (M - h) \cdot \frac{q + 1}{q + 2}. \]
\[ \geq \text{LAG}(\tau, t_h + 1, S) \quad \text{(because } a_0 \leq \frac{(M - h) \cdot (q + 1)}{q + 2} \text{ for this case)} \]
\[ \geq qM + 1 \quad \text{(by (T1))}. \]  

(50)

The lemma follows from (49) and (50), and by the conditions of Cases 1 and 2, respectively. \(\square\)

In completing Case C, we make use of this auxiliary algebraic lemma, proved in Appendix A.

**Lemma 36.** The roots of \( f(W_{\text{max}}) \) defined as
\[ f(W_{\text{max}}) = 2(M - h)(q + 1)W_{\text{max}}^2 - (q + 2)(M - h)W_{\text{max}} - ((q - 1)M + 1 + h) = 0 \]
are
\[ W_{\text{max}} = \frac{(q + 2)(M - h) + \sqrt{9q^2(M - h)^2 + \Delta}}{4(M - h)(q + 1)}, \]
where \( \Delta = 4(M - h)(M(q - 1) + h(2q^2 + q + 1) + 2q + 2). \)

We conclude this case by establishing the following lemma.

**Lemma 37.** If either \( W_{\text{max}} < \frac{q + 3}{2q + 4} \) and \( a_0 > \frac{(M - h)(q + 1)}{q + 2} \) or \( W_{\text{max}} > \frac{q + 3}{2q + 4} \) and \( a_0 > 2(M - h)(1 - W_{\text{max}}) \), then \( \text{LAG}(\tau, t_h + 1, S) < qM + 1 \).

**Proof.** We consider two cases based on the statement of the lemma.

**Case 1:** \( W_{\text{max}} > \frac{q + 3}{2q + 4} \) and \( a_0 > 2(M - h)(1 - W_{\text{max}}) \). By (30),
\[ \text{LAG}(\tau, t_h + 1, S) < a_0 \cdot W_{\text{max}} + a_0^2 (q + 1)W_{\text{max}} + a_0 - q \cdot W_{\text{max}} + a_0^2 ((q + 2)W_{\text{max}} - 1) + a_0^2 ((q + 3)W_{\text{max}} - 2) \]
\[ < a_0 \cdot W_{\text{max}} + a_0^2 (q + 1)W_{\text{max}} + a_0 - q \cdot W_{\text{max}} + a_0^2 ((q + 2)W_{\text{max}} - 1) + a_0^2 ((q + 3)W_{\text{max}} - 2) \]
\[ = a_0 \cdot (1 - (q + 1)W_{\text{max}}) + (M - h) \cdot ((q + 2)W_{\text{max}} - 1) \]
\[ < 2(M - h)(1 - W_{\text{max}}) \cdot (1 - (q + 1)W_{\text{max}}) + (M - h) \cdot ((q + 2)W_{\text{max}} - 1) \]
\[ \quad \text{(because } W_{\text{max}} > \frac{q + 3}{2q + 4} \text{ for all } q \geq 1, 1 - (q + 1)W_{\text{max}} < 0; \text{ also, by the conditions of Case 1, } a_0 > 2(M - h)(1 - W_{\text{max}}) \text{)} \]
\[ = 2(M - h)(q + 1)W_{\text{max}}^2 - (q + 2)(M - h)W_{\text{max}} - (q - 1)M + 1 + h = 0. \]

We next show that \( \text{LAG}(\tau, t_h + 1, S) < qM + 1 \) holds (if \( W \) holds). Suppose to the contrary that \( \text{LAG}(\tau, t_h + 1, S) \geq qM + 1 \); then by the derivation above
\[ 2(M - h)(q + 1)W_{\text{max}}^2 - (q + 2)(M - h)W_{\text{max}} - ((q - 1)M + 1 + h) > 0. \]

By Lemma 36, the roots of \( f(W_{\text{max}}) = 2(M - h)(q + 1)W_{\text{max}}^2 - (q + 2)(M - h)W_{\text{max}} - ((q - 1)M + 1 + h) = 0 \) are
\[ W_{\text{max}} = \frac{(q + 2)(M - h) + \sqrt{9q^2(M - h)^2 + \Delta}}{4(M - h)(q + 1)}, \]
and \( W_{\text{max}} = \frac{(q + 2)(M - h) - \sqrt{9q^2(M - h)^2 + \Delta}}{4(M - h)(q + 1)}. \)
Since \( 0 < h < M \) and \( q \geq 1 \) hold, \( \Delta > 0 \) holds, and hence, \( \sqrt{9q^2(M - h)^2 + \Delta} \) is greater than \( 3q(M - h) \). Note that \( W_{\text{max}} > \frac{(q + 2)(M - h) + 3(M - h)q}{4(M - h)(q + 1)} = \frac{q + 2}{4q + 4} > 0. \) Also, because \( h < M \), \( 3q(M - h) \geq (q + 2)(M - h) \) for all \( q \geq 1 \). Therefore, \( W_{\text{max}} < 0. \) The first derivative of \( f(W_{\text{max}}) \) with respect to \( W_{\text{max}} \) is given by \( f'(W_{\text{max}}) = 4(M - h)(q + 1)W_{\text{max}} - (q + 2)(M - h) \), which is positive for \( W_{\text{max}} > \frac{q + 2}{4q + 4} \). Hence, \( f(W_{\text{max}}) \) is an increasing function of \( W_{\text{max}} \) for \( W_{\text{max}} \geq \frac{q + 2}{4q + 4}; \) further, the following hold: \( W_{\text{max}} > \frac{q + 2}{4q + 4} \). Therefore, we have \( f(W_{\text{max}}) < 0 \) for \( W_{\text{max}} < \frac{q + 2}{4q + 4} \). Because as mentioned earlier, \( W_{\text{max}} = \frac{(q + 2)(M - h) + \sqrt{9q^2(M - h)^2 + \Delta}}{4(M - h)(q + 1)} = \frac{q + 2}{4q + 4} \), it follows that, for all \( 0 < W_{\text{max}} \leq \frac{q + 2}{4q + 4}, f(W_{\text{max}}) < 0. \) By (W), \( W_{\text{max}} \leq \frac{q + 2}{4q + 4} \) holds, and hence, (51) does not hold, implying that \( \text{LAG}(\tau, t_h + 1) < qM + 1 \). Thus, by the conditions of Case 1, if \( W_{\text{max}} > \frac{q + 3}{2q + 4} \) and \( a_0 > 2(M - h)(1 - W_{\text{max}}) \), then \( \text{LAG}(\tau, t_h + 1, S) < qM + 1 \) follows.
Case 2: $W_{\text{max}} \leq \frac{q+3}{2q+4}$ and $a_0 > \frac{(M-h)\cdot(q+1)}{q+2}$. Because $\frac{q+3}{2q+4} < \frac{1}{2}$, for all $q \geq 1$, $q \cdot W_{\text{max}} \geq (q+3)W_{\text{max}} - 2$ holds. Hence, by (30), we have

$$\text{LAG}(\tau, t_h + 1, S) < a_0 \cdot W_{\text{max}} + (a_{q-1} + a_q^1)q \cdot W_{\text{max}} + a_q^4(q + 1)W_{\text{max}} + a_q^1((q + 2)W_{\text{max}} - 1)$$

$$= a_0 \cdot W_{\text{max}} + (a_{q-1} + a_q^1)q \cdot W_{\text{max}} + a_q^1((q + 2)W_{\text{max}} - 1)$$

(because $a_q^0 = 0$ by the conditions of Case C).

(52)

We consider two subcases based on the value of $W_{\text{max}}$.

Subcase 2(a): $\frac{1}{2} < W_{\text{max}} \leq \frac{q+3}{2q+4}$. For this case, $(q + 2)W_{\text{max}} - 1 > q \cdot W_{\text{max}}$ holds. Hence, by (52), we have

$$\text{LAG}(\tau, t_h + 1, S) < a_0 \cdot W_{\text{max}} + (a_{q-1} + a_q^1)(q + 2)W_{\text{max}} - 1)$$

$$\leq a_0 \cdot W_{\text{max}} + (M - h - a_0) \cdot (q + 2)W_{\text{max}} - 1) \quad \text{(by (32) because } a_q^0 = 0)$$

$$= a_0 \cdot (1 - (q + 1)W_{\text{max}}) + (M - h) \cdot ((q + 2)W_{\text{max}} - 1).$$

(53)

Let $f(a_0, W_{\text{max}}) \overset{\text{def}}{=} a_0 \cdot (1 - (q + 1)W_{\text{max}}) + (M - h) \cdot ((q + 2)W_{\text{max}} - 1)$, the right-hand side of the above inequality. Our goal is to determine an upper bound for $f(a_0, W_{\text{max}})$. We first show that $f(a_0, W_{\text{max}})$ is an increasing function of $W_{\text{max}}$ for all $a_0 \geq 0$, and a decreasing function of $a_0$ for any $W_{\text{max}} \geq \frac{1}{q+1}$. (In the description that follows, we assume $a_0$ and $W_{\text{max}}$ are non-negative.) The first derivative of $f(a_0, W_{\text{max}})$ with respect to $W_{\text{max}}$ is $(M-h)(q+2)-a_0(q+1)$. Therefore, since $a_0 \leq M - h$ and $M - h > 0$, it follows that $(M-h)(q+2)-a_0(q+1)$ is positive for all $q \geq 0$. Hence, $f(a_0, W_{\text{max}})$ is an increasing function of $W_{\text{max}}$ for all valid $a_0$. Further, $f(a_0, W_{\text{max}})$ is a non-decreasing function of $a_0$ for all $W_{\text{max}} \leq \frac{1}{q+1}$, and is a decreasing function of $a_0$ for all $W_{\text{max}} > \frac{1}{q+1}$. Therefore, since $W_{\text{max}} \leq \frac{q+3}{2q+4}$, $a_0 \leq M - h$, and $a_0 \geq 0$ (by Lemma 18), $f(a_0, W_{\text{max}})$ is maximized when either $W_{\text{max}} = \frac{q+3}{2q+4}$ and $a_0 = 0$ or $W_{\text{max}} = \frac{1}{q+1}$ and $a_0 = M - h$. It can easily be verified that $f(a_0, \frac{q+3}{2q+4}) = a_0 \cdot \left(\frac{-q^2 - 2q + 1}{2} + h \cdot \frac{q + 1}{2}\right) < Q + 1$ for all $a_0 \geq 1$. It can also be verified that $f(a_0, \frac{1}{q+1}) = \frac{M-h}{q+1} < Q + 1$ for all $a_0$. Hence, $f(a_0, W_{\text{max}}) < Q + 1$, and therefore, $\text{LAG}(\tau, t_h + 1, S) < Q + 1$ holds.

Subcase 2(b): $W_{\text{max}} \leq \frac{1}{2}$. For this case, $(q + 2)W_{\text{max}} - 1 \leq q \cdot W_{\text{max}}$ holds. Hence, by (52), we have

$$\text{LAG}(\tau, t_h + 1, S) < a_0 \cdot W_{\text{max}} + (a_{q-1} + a_q^1 + a_q^2) \cdot q \cdot W_{\text{max}}$$

$$\leq a_0 \cdot W_{\text{max}} + (M - h - a_0) \cdot q \cdot W_{\text{max}} \quad \text{(by (32) because } a_q^0 = 0)$$

$$= a_0 \cdot W_{\text{max}} + (M - h) \cdot q \cdot W_{\text{max}}$$

$$< qM + 1.$$

By the reasoning in Subcases 2(a) and 2(b), it follows that if $W_{\text{max}} \leq \frac{q+1}{2q+4}$ and $a_0 \geq \frac{(M-h)\cdot(q+1)}{q+2}$, then $\text{LAG}(\tau, t_h + 1, S) < qM + 1$.

Finally, the lemma holds by the conclusions drawn in Cases 1 and 2.

By Lemmas 35 and 37, for any $a_0$ and $W_{\text{max}}$, either $\text{LAG}(\tau, t_h + 1, S) < Q + 1$ or $\text{LAG}(\tau', t_h - 1, S') \geq Q + 1$ holds. Thus, either (T1) or Definition 5 is contradicted.

3.9. Case D: $(A_q^0 = A_q^1 = \emptyset)$

Lemma 38. If $A_q^0 = A_q^1 = \emptyset$, then $\text{LAG}(\tau, t_h + 1) < qM + 1$.

Proof. Because $A_q^0 = A_q^1 = \emptyset$, and $a_0, a_{q-1}$, and $a_q^2$ are independent of $W_{\text{max}}$, as explained earlier (when (31) was established), we bound $\text{LAG}(\tau, t_h + 1)$ assuming $W_{\text{max}} \geq 2/3$. Hence, by (31), and $A_q^0 = A_q^1 = \emptyset$, we have $\text{LAG}(\tau, t_h + 1) < a_0 \cdot W_{\text{max}} + ((q + 3)W_{\text{max}} - 2) \cdot (a_q^2 - a_q^0)$, which, by (32), equals $a_0 \cdot W_{\text{max}} + ((q + 3)W_{\text{max}} - 2) \cdot (M - h - a_0)$.

Contrary to the statement of the lemma, assume $\text{LAG}(\tau, t_h + 1) \geq qM + 1$. This assumption implies that $a_0 \cdot W_{\text{max}} + ((q + 3)W_{\text{max}} - 2) \cdot (M - h - a_0) > qM + 1$, which, in turn, implies that $W_{\text{max}} > f(\frac{(q+2)M-2h-2a_0}{(1+3)M-(q+3)h-a_0})$. We now determine a lower bound for $f$ and show that $f$ lies outside the range of values assumed for $W_{\text{max}}$ and arrive at a contradiction. Let $Y$ denote the denominator of $f$. The first derivative of $f$ with respect to $h$ is given by $\frac{(q+2)M-2h-2a_0}{Y^2}$, which is non-negative.
for all $M \geq 1$, $a_0 \geq 1$, and $q \geq 1$. The first derivative of $f$ with respect to $a_0$ is given by $\frac{M(q^2 + q - 2) + 2h + q + 2}{(q+2)M-3}$, which is also non-negative for all $M \geq 1$, $q \geq 1$, and $h \geq 0$. Hence, since $h$ and $a_0$ are greater than zero, $f$ is minimized when $h = a_0 = 1$, for which $f = \frac{q^2 + 2}{(q+2)M-3} > \frac{q+2}{q+3}$ holds, for all $q \geq 1$, $M > 1$. This violates (W), and hence, our assumption is false, and the lemma follows. □

By Lemmas 20, 27, 35, 37, and 38, if (W) is satisfied, then either $\text{LAG}(t, t_0 + 1) < qM + 1$ or there exists another task system with LAG under EPDF at least $qM + 1$ at $t_0 - 1$. Thus, either (T1) or the minimality of $t_0$ is contradicted. So, task system $\tau$ as defined in Definition 6 does not exist, and Theorem 2 holds.

Theorem 2 implies that if each task weight is at most $W_{\text{max}}$, then tardiness under EPDF is at most $[\frac{3W_{\text{max}}^2 - 2}{W_{\text{max}}}]$, and we have the following corollary.

**Corollary 1.** If the weight of each task in a feasible GIS task system $\tau$ is at most $W_{\text{max}}$, then EPDF ensures a tardiness bound of $\max(1, [\frac{3W_{\text{max}}^2 - 2}{W_{\text{max}}}]$) for $\tau$.

**Proof.** Assume to the contrary that the tardiness for some subtask in $\tau$ is $q$, where $q > \max(1, \frac{3W_{\text{max}}^2 - 2}{W_{\text{max}}})$. Then, $q > \max(1, \frac{3W_{\text{max}}^2 - 2}{W_{\text{max}}})$ holds, which implies that $q > 1$ and $W_{\text{max}} < \frac{q+2}{q+3}$. This contradicts Theorem 2. □

4. Conclusion

We have presented counterexamples that show that, in general, tardiness under the EPDF $P_f$ algorithm can exceed a small constant number of quanta for feasible recurrent real-time task systems. Thus, the conjecture that EPDF ensures a tardiness bound of one quantum for all feasible task systems is proved false. We have also presented sufficient per-task utilization restrictions that are more liberal than those previously known for ensuring a tardiness of one quantum for all feasible task systems. Thus, the conjecture that a small constant number of quanta for feasible recurrent real-time task systems. Thus, the conjecture that

**Acknowledgments**

The first author is grateful to Anand Srinivasan for his EPDF simulator and for sharing the details of a task system that is close to missing a deadline by two quanta (specifically, whose LAG can be $M-1$). Examination of that task system served as a starting point for the construction of several task systems that can miss their deadlines by two or more quanta, including those reported in this paper.

**Appendix A. Proofs omitted in the main text**

In this appendix, we present all proofs omitted in the main paper. We begin with Claim 1.

A.1. Proof of Claim 1

**Claim 1.** There is no hole in any slot in $[t_d - 1, t_d + q]$ in $S'$.

**Proof.** By Definition 3, (S1), and (S2), exactly one subtask in $\sigma$ has a tardiness of $q + 1$. Let $T_i$ denote that subtask. By (S1) again, the deadline of $T_i$ is at $t_d$, and hence, $T_i$ is scheduled at time $t_d + q$.

The proof of the claim is by induction on decreasing time $t$. We start by showing that there is no hole in slot $t_d + q - 1$.

**Base case: $t = t_d + q - 1$.** Let $T_h$ denote the predecessor, if any, of $T_i$. Because the deadlines of any two successive subtasks of the same task differ by at least one time unit, $d(T_h) \leq t_d - 1$ holds. Therefore, by Definition 3, the tardiness of $T_h$ is at most $q$, and $T_h$ completes executing by $t_d + q - 1$. Hence, no subtask of $T$ is scheduled in slot $t_d + q - 1$. Thus, there is no hole in slot $t_d + q - 1$; otherwise, EPDF would schedule $T_i$ there.

**Induction hypothesis.** Assume that there is no hole in any slot in $[t', t_d + q]$, where $t_d - 1 < t' < t_d + q$.

**Induction step: $t = t' - 1$.** We show that there is no hole in slot $t' - 1$. The deadline of every subtask scheduled in $t'$ is at most $t_d$. Hence, the release time and the eligibility time of every such subtask is at or before $t_d - 1$. Since $t_d - 1 \leq t' - 1$, every subtask scheduled at $t'$ can be scheduled at $t' - 1$ unless its predecessor is scheduled there. By the induction hypothesis,
there is no hole in slot $t'$. Hence, if there is a hole in $t' - 1$, then at most $M - 1$ of the $M$ subtasks scheduled at $t'$ can have their predecessors scheduled at $t' - 1$, implying that at least one of the subtasks scheduled at $t'$ should have been scheduled at $t' - 1$, which is a contradiction. Therefore, there can be no hole in $t' - 1$. □

A.2. Proofs from Section 3.1

Lemma 10. The allocation received by a $k$-dependent subtask in its first slot in the ideal schedule are as follows.

(a) The allocation $A(\text{ideal}, T_i, r(T_i))$ received in the ideal schedule by a $k$-dependent subtask $T_i$ of a periodic task with $\text{wt}(T) < 1$ in the first slot of its window is at most $k \cdot \frac{T_i}{T_p} - (k - 1) - \frac{1}{T_p}$, for all $k \geq 0$.

(b) The allocation $A(\text{ideal}, T_i, r(T_i))$ received in the ideal schedule by a $k$-dependent subtask $T_i$ of a GIS task in the first slot of its window is at most $k \cdot \frac{T_i}{T_p} - (k - 1) - \frac{1}{T_p}$, for all $k \geq 0$.

(c) Let $T_i$, where $i \geq k + 1$ and $k \geq 1$, be a subtask of $T$ with $\text{wt}(T) < 1$ such that $|\omega(T_i)| \geq 3$ and $b(T_{i-1}) = 1$. Let the number of subtasks in $T_{i-1}$’s dependency group be at least $k$. Then, $A(\text{ideal}, T_i, r(T_i)) \leq k \cdot \frac{T_i}{T_p} - (k - 1) - \frac{1}{T_p}$.

Proof. Each part is proved below in turn.

Proof of part (a). The proof is by induction on $k$.

Base case: $k = 0$. Because $\text{wt}(T) < 1$, and $T.e$ and $T.p$ are integral, $T.e \leq T.p - 1$. Thus, by (9), $A(\text{ideal}, T_i, r(T_i)) \leq \text{wt}(T) = T.e/T.p \leq (T.p - 1)/T.p = 1 - 1/T.p$, and the lemma holds for the base case.

Induction step. Assuming that the lemma holds for $(k - 1)$-dependent subtasks, we show that it holds for $k$-dependent subtasks, where $k \geq 1$. Because $k \geq 1$, by the definition of $k$-dependency, $i > 1$ and $T$ is heavy. Hence, by Lemma 1, $|\omega(T_{i-1})|$ is either two or three. We consider two cases.

Case 1: $|\omega(T_{i-1})| = 2$. Since $k \geq 1$, $T_{i-1}$ is $(k - 1)$-dependent. Therefore, by the induction hypothesis,

\[
A(\text{ideal}, T_{i-1}, r(T_{i-1})) \leq (k - 1) \cdot (T.e/T.p) - (k - 2) - (1/T.p).
\]

Because $|\omega(T_{i-1})| = 2$, by (8), $A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) = 1 - A(\text{ideal}, T_{i-1}, r(T_{i-1}))$. Hence, by (54), $A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \geq (k - 1) + (1/T.p) - (k - 1) \cdot (T.e/T.p)$. Because $T_i$ is $k$-dependent, where $k \geq 1$, by Lemma 8(c), $b(T_{i-1}) = 1$, and by Lemma 3, $A(\text{ideal}, T_i, r(T_i)) = (T.e/T.p) - A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \leq k \cdot (T.e/T.p) - (k - 1) - (1/T.p)$.

Case 2: $|\omega(T_{i-1})| = 3$. By the contra-positive of Lemma 8(c), $T_{i-1}$ is 0-dependent; hence, $T_i$ is 1-dependent, i.e., $k = 1$. By Lemma 8(c), $b(T_{i-1}) = 1$, and hence, by Lemma 3,

\[
A(\text{ideal}, T_i, r(T_i)) = \frac{T.e}{T.p} - A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \leq \frac{T.e}{T.p} - \frac{1}{T.p} \quad \text{(by (10)).}
\]

Proof of part (b). It follows from part (a) and the definition of GIS tasks. (The allocation that $T_i$ receives in each slot of its window is identical to the allocation that it would receive if $T$ were periodic.) □

Proof of part (c). Since $|\omega(T_i)| \geq 3$, by Lemma 8(c), $T_i$ is 0-dependent and is the first subtask in its group. Hence, $T_{i-1}$ is the final subtask in its dependency group, and since there are at least $k$ subtasks in $T_{i-1}$’s group, $T_{i-1}$ is at least $(k - 1)$-dependent. Hence, by Lemma 10(b), $A(\text{ideal}, T_{i-1}, r(T_{i-1})) \leq (k - 1) \cdot (T.e/T.p) - (k - 2) - (1/T.p)$. (What follows is similar to the reasoning used in the induction step in the proof of Lemma 10(a).) If $|\omega(T_{i-1})| = 2$, then, by (8), $A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \geq 1 - (k - 1) \cdot \frac{T.e}{T.p} - (k - 2) - \frac{1}{T.p} = (k - 1) - (k - 1) \cdot \frac{T.e}{T.p} + \frac{1}{T.p}$. By the statement of the lemma, $b(T_{i-1}) = 1$, and hence, by Lemma 3, $A(\text{ideal}, T_i, r(T_i)) = \text{wt}(T) - A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \leq k \cdot \frac{T.e}{T.p} - (k - 1) - \frac{1}{T.p}$. Thus, the lemma holds when $|\omega(T_{i-1})| = 2$.

On the other hand, if $|\omega(T_{i-1})| \geq 3$, then by Lemma 8(c), $T_{i-1}$ is 0-dependent. By (10), $A(\text{ideal}, T_{i-1}, d(T_{i-1}) - 1) \geq \frac{T.e}{T.p}$, and hence, because $b(T_{i-1}) = 1$, by Lemma 3, $A(\text{ideal}, T_i, r(T_i)) = \text{wt}(T) - A(\text{ideal}, T_{i-1}, d(T_{i-1})) \leq \frac{T.e}{T.p} - \frac{1}{T.p}$. By the statement of the lemma, $|\omega(T_i)| = 3$, and hence, $T_i$ is also 0-dependent. Thus, $T_{i-1}$ is the only subtask in its group, and hence, $k = 1$. (Note that $k$ here denotes the number of subtasks that are in the same dependency group as $T_{i-1}$.) Therefore, the lemma holds for this case too. □

Lemma 11. Let $T_i$ be a $k$-dependent subtask of a task $T$ for $k \geq 0$, and let the tardiness of $T_i$ be $s$ for some $s \geq 1$ (that is, $T_i$ is scheduled at time $d(T_i) + s - 1$). Then $\text{lag}(T, d(T_i) + s) < (k + s + 1) \cdot \text{wt}(T) - k$. 

Proof. By the statement of the lemma, $T_i$ and all prior subtasks of $T$ are scheduled in $[0, d(T_i)+s)$. Hence, lag($T$, d($T_i$)+s) depends on the number of subtasks of $T$ after $T_i$ (that is, successors of $T_i$) released prior to d($T_i$)+s, the allocations they receive in the ideal schedule, and when they are scheduled in $S$. It can be verified from (1) and (2) that at most $s+1$ successors of $T_i$ - $T_{i+1}, \ldots, T_{i+s+1}$ - are released without any IS separations and $S$ is maximized if all those subtasks are present and are released without any IS separations and $S$ has not scheduled any of them by time d($T_i$)+s. We will assume that this is the case. (The statement of the lemma implies that none of those subtasks is scheduled by d($T_i$)+s.) By Lemma 2, at most one successor of $T_i$, namely $T_{i+1}$, can have a release time that is before d($T_i$). Further, r($T_{i+1}$) $\geq$ d($T_i$) - 1 holds. Hence, lag($T$, d($T_i$)+s) $\leq$ A(ideal, $T_{i+1}$, d($T_i$)+1) + A(ideal, $T$, d($T_i$)+s). If r($T_{i+1}$) > d($T_i$) - 1 holds, then A(ideal, $T_{i+1}$, d($T_i$)-1) = 0. On the other hand, if r($T_{i+1}$) = d($T_i$) - 1, then by (4), b($T_i$) = 1. Further, either $\left|\omega(T_{i+1})\right| = 2$ or $\left|\omega(T_{i+1})\right| > 2$. In the former case, $T$ is heavy, and because b($T_i$) = 1, by the definition of k-dependency (and given by Lemma 8(a)), $T_{i+1}$ belongs to the same dependency group as $T_i$ and is (k+1)-dependent. Hence, by Lemma 10(b), A(ideal, $T_{i+1}$, r($T_{i+1}$)) $\leq$ (k+1) \cdot wt($T$) - k - \frac{1}{\rho p}. If the latter holds, i.e., $\left|\omega(T_{i+1})\right| > 2$, we reason as follows. Since $T_i$ is k-dependent, the number of subtasks in $T_i$'s group is at least $k+1$. Therefore, since b($T_i$) = 1, Lemma 10(c) applies for $T_{i+1}$ and it follows that A(ideal, $T_{i+1}$, r($T_{i+1}$)) $\leq$ (k+1) \cdot wt($T$) - k - \frac{1}{\rho p}. Thus, in either case, A(ideal, $T_{i+1}$, d($T_i$) - 1) = A(ideal, $T_{i+1}$, r($T_{i+1}$)) $\leq$ (k+1) \cdot wt($T$) - k - \frac{1}{\rho p}.

By (7), A(ideal, $T$, d($T_i$), d($T_i$)+s) $\leq$ s \cdot wt($T$). Hence, lag($T$, d($T_i$)+s) $\leq$ A(ideal, $T_{i+1}$, d($T_i$)-1) + A(ideal, $T$, d($T_i$), d($T_i$)+s) $\leq$ (k+1) \cdot wt($T$) - k - \frac{1}{\rho p} < (k+s+1) \cdot wt($T$) - k. □

A.3. Proofs from Section 3.4

Lemma 18. There exists a subtask $W_t$ scheduled at $t_b$ with o($W_t$) $\leq$ $t_b$, d($W_t$) = $t_b$ + 1, and $S(W, t)$ = 0, for all $t$ $\in$ [tb, $t_b$]. Also, there is no hole in any slot in [tb, $t_b$]. (Note that, by this lemma, A0(tb) $\neq \emptyset$.)

Proof. We first show that the first subtask to be displaced upon $U_j$'s removal (where $U_j$ is as defined in Definition 8) has properties as stated for $W_t$, i.e., is eligible at or before $t_b$ and has its deadline at $t_b$ + 1.

Let $\tau'$ be the task system obtained by removing $U_j$ from $\tau$, and let $S'$ be the EPDF schedule for $\tau'$. Let $\Delta_1 = \langle X^{(1)}$, $t_1$, $X^{(2)}$, $t_2$ $\rangle$ be the first valid displacement, if any, that results due to the removal of $U_j$. Then, $X^{(1)}$ = $U_j$, $t_1$ = $t_b$, and by Lemma 5,

$$t_2 > t_1 = t_b.$$  

We first show that $t_2 \geq t_b$.

Assume to the contrary that $t_2 < t_b$. Then, by (55) and Definition 7, $T$ is not in $B(t_b)$. Therefore, $T$ is in $I(t_b)$ or in $A(t_b)$.

In either case,

$$d(X^{(2)}) \leq t_b.$$  

To see this, note that if $T$ $\in$ $I(t_b)$, then because $T$ is not active at $t_b$, by Definition 1, d($X^{(2)}$) $\leq$ $t_b$. On the other hand, if $T$ $\in$ A($t_b$), then consider $T$'s subtask, say $Tk$, scheduled at $t_b$. By Lemma 16, d($Tk$) $\leq$ $t_b$ + 1. Because $X^{(2)}$ is scheduled at $t_2 < t_b$, $X^{(2)}$ is an earlier subtask of $T$ than $Tk$, and hence, by (1) and (2), d($X^{(2)}$) $\leq$ $t_b$. Because $U_j$ is $U_j$'s subtask at $t_b$ and $U$ is in $B(t_b)$, by Lemma 17, we have

$$d(U_j) = t_b + 1.$$  

By (56) and (57), (d($U_j$)) > d($X^{(2)}$). However, since EPDF selects $U_j$ over $X^{(2)}$ at time $t_b$ (which follows because the displacement under consideration is valid), this is a contradiction. Thus, our assumption that $t_2 < t_b$ holds is false.

Having shown that $t_2 \geq t_b$, we next show $t_2 = t_b$. Assume, to the contrary, that $t_2 > t_b$. Since displacement $\Delta_1 = \langle U_j$, $t_b$, $t_1$, $t_2$ $\rangle$ is valid, o($X^{(2)}$) $\leq$ $t_b$. This implies that $X^{(2)}$ is eligible to be scheduled at $t_b$ (i.e., $T$ is not scheduled at $t_b$), and because there is a hole in $t_b$, it should have been scheduled there in $S$, and not later at $t_2$. It follows that $t_2 = t_b$.

Finally, because $U_j$ is scheduled at $t_b$ in preference to $X^{(2)}$, d($T_i$) $\geq$ d($U_j$) = $t_b$ + 1 (from (57)), which by Lemma 15 (since $X^{(2)}$ is scheduled in slot $t_b$) implies that

$$d(X^{(2)}) = t_b + 1.$$  

Thus, the first subtask, if any, to be displaced upon $U_j$'s removal satisfies the properties specified for $W_t$ in the statement of the lemma. Hence, if a subtask with such properties does not exist, then $U_j$'s removal will not lead to any displacements.

Next, we show that unless the other two conditions specified in the lemma also hold, no subtask will be displaced upon $U_j$'s removal. For this, first note that by (57) and (58) $X^{(2)}$ and $U_j$ have equal deadlines, and hence, $X^{(2)}$ is not $U_j$'s successor. Next, note that because (U_j, $t_b$, $X^{(2)}$, $t_b$) is valid, no subtask of $T$ prior to $X^{(2)}$ is scheduled in [tb, $t_b$], and also if there is a hole in any slot in [tb, $t_b$], then EPDF would have scheduled $X^{(2)}$ at $t_b$.

Thus, if the lemma is false, then removing $U_j$ does not result in any displacements. We now show that, in such a case, $\text{LAG}(\tau', t_b+1, S') \geq qM + 1$. $\text{LAG}(\tau', t_b+1, S') = A(\text{ideal}, \tau', 0, t_b+1) + A(S', \tau', 0, t_b+1)$. $\tau'$ contains every subtask that is in $\tau$ except $U_j$. $U_j$ is scheduled before $t_b$ in $S$, and by (57), d($U_j$) = $t_b$ + 1. Therefore, $U_j$ receives an allocation of one quantum
by time \( t_h + 1 \) in the ideal schedule for \( \tau \), and hence, \( A(\text{ideal}, \tau', 0, t_h + 1) = A(\text{ideal}, \tau, 0, t_h + 1) - 1 \). Similarly, since no subtask other than \( U_j \) of \( \tau \) is displaced or removed in \( S' \), the total allocation to \( \tau' \) in \( S' \) up to time \( t_h + 1 \), \( A(S', \tau', 0, t_h + 1) \), is \( A(S, \tau, 0, t_h + 1) - 1 \). Therefore, \( \text{LAG}(\tau', t_h + 1, S') = A(\text{ideal}, \tau, 0, t_h + 1) - A(S, \tau, 0, t_h + 1) = \text{LAG}(\tau, t_h + 1, S) \geq qM + 1 \) (by (T1)). To conclude, we have shown that, \( \tau' \) with one fewer subtask than \( \tau \) also has a \( \text{LAG} \) of at least \( qM + 1 \) at \( t_h + 1 \), which contradicts (T2). \( \square \)

**Lemma 19.** Let \( t_m \leq t_h \) be a slot in which an MI is scheduled. Then, the following hold.

(a) For all \( t \), where \( t_m - (q + 2) < t < t_m \), there is no hole in slot \( t \), and for each subtask \( V_k \) that is scheduled in \( t \), \( d(V_k) \leq t_m - q + 1 \).

(b) Let \( W \) be a task in \( B(t_m) \) and let the critical subtask \( W' \) of \( W \) at \( t_m \) be scheduled before \( t_m \). Then, \( W' \) is scheduled at or before \( t_m - (q + 2) \).

**Proof of part (a).** The proof is by induction on decreasing \( t \). We start with \( t = t_m - 1 \).

**Base case:** \( t = t_m - 1 \). Let \( T_i \) be an MI scheduled at \( t_m \). (By the statement of the lemma, at least one MI is scheduled in \( t_m \).) Then, \( d(T_i) = t_m - q + 1 \), and \( S(T, t_m - 1) = 0 \), from the definition of an MI. Hence, \( T_i \) is eligible at \( t_m - 1 \). Because \( T_i \) is not scheduled at \( t_m - 1 \), it follows that there is no hole in \( t_m - 1 \) and that the priority of every subtask \( V_k \) scheduled at \( t_m - 1 \) is at least that of \( T_i \), i.e., \( d(V_k) \leq d(T_i) = t_m - q + 1 \).

**Induction hypothesis.** Assume that the claim in part (a) holds for all \( t \), where \( t' + 1 < t \leq t_m - 1 \) and \( t_m - (q + 1) < t' + 1 < t_m \).

**Induction step.** We now show that the claim holds for \( t = t' \). By the induction hypothesis, there is no hole in \( t' + 1 \) and \( d(T_i) < t_m - q + 1 \) holds for every subtask \( T_i \) scheduled in \( t' + 1 \). Therefore, since \( w(t(T)) < 1 \), by (1), \( r(T_i) < t_m - q - 1 \). Thus, there are \( M \) subtasks with a release time at or before \( t_m - q - 1 \) and deadline at or before \( t_m - q + 1 \) scheduled in \( t' + 1 \). If there is a hole in \( t' \) or a subtask with deadline later than \( t_m - q + 1 \) scheduled in \( t' \), then there is at least one subtask scheduled in \( t' + 1 \) whose predecessor is not scheduled in \( t' \). Such a subtask is eligible at \( t' \), since its release time is at or before \( t_m - q - 1 \). Hence, if there is a hole in \( t' \), then the work-conserving behavior of EPDF is contradicted. Otherwise, the pseudo-deadline-based scheduling of EPDF is contradicted. Hence, the claim holds for \( t = t' \). \( \square \)

**Proof of part (b).** By Definition 2, \( d(W') \geq t_m + 1 \). Hence, since \( q \geq 1 \), this part easily follows from part (a). \( \square \)

**A.4.** Proofs from Section 3.8

**Lemma 28.** The following properties hold for subsets \( \tau'_1 \) and \( \tau' \) defined in Section 3.8, where \( 1 \leq i \leq 8 \).

(a) For every task \( T \), there is at most one subtask in \( (\tau'_1 \cup \tau'_2 \cup \tau'_6) \).

(b) Let \( T_i \) be the subtask of a task \( T \) in \( A_q(t_h) \) or \( A_{q-1}(t_h) \). Then, \( T_i \) is in \( \tau'_6 \).

(c) \( \tau'_7 \subseteq \tau'_8 \) and \( \tau'_8 \subseteq \tau'_4 \).

(d) Subsets \( \tau'_i \), where \( 1 \leq i \leq 6 \), are pairwise disjoint.

**Proof.** Each of the above properties is proved below.

**Proof of part (a).** We first show that each task \( T \) has at most one subtask in \( \tau'_1 \). Let \( T_i \) in \( \tau'_1 \) be the critical subtask at \( t_h \) of \( T \), which is in \( B(t_h) \). Then, by Lemma 17, \( d(T_j) = t_h + 1 \) holds. Because \( w(t(T)) < 1 \), by (1), \( r(T_i) \leq d(T_j) - 2 = t_h - 1 \) holds. Hence, by the definition of a critical subtask in Definition 2, \( T_i \) is critical at \( t_h - 1 \) also. Thus, if \( T \) has a critical subtask \( T_j \) at \( t_h \), then \( T \) cannot have a subtask that is different from \( T_i \) that is critical at \( t_h - 1 \). Hence, it follows that each task has at most one subtask in \( \tau'_1 \).

We next show that each task can have at most one subtask in \( \tau'_2 \cup \tau'_6 \). Note that a subtask is in \( \tau'_2 \) or \( \tau'_6 \) only if it is scheduled at \( t_h - 1 \). Further, each task \( T \) can have at most one subtask scheduled at \( t_h - 1 \). Hence, if \( T \)’s subtask \( T_i \) scheduled at \( t_h - 1 \) has its deadline at or after \( t_h \), then \( T_i \) is in \( \tau'_2 \); else, in \( \tau'_6 \).

Finally, we show that if \( T \) has a subtask \( T_j \) in \( \tau'_1 \), then it does not have a subtask in \( \tau'_2 \cup \tau'_6 \), and vice versa. If \( T_i \) is in \( B(t_h - 1) \), then \( T \) cannot have a subtask scheduled at \( t_h - 1 \), and hence, cannot have a subtask in \( \tau'_2 \cup \tau'_6 \) (because every subtask in these sets is scheduled at \( t_h - 1 \)). On the other hand, if \( T_i \) is in \( B(t_h) \) and is \( T \)’s critical subtask at \( t_h \), then note the following. (i) \( \tau'_1 \) is non-empty only if \( \tau'_6 \) exists; (ii) by Lemma 17, \( d(T_j) = t_h + 1 \) holds; and (iii) \( T_j \) is scheduled at or before \( t'_q \), whereas a subtask in \( \tau'_2 \cup \tau'_6 \) is scheduled at \( t_h - 1 \). By (iv), \( \tau'_6 \leq t_h - (q + 3) \). Thus, by (ii) and (iii), no subtask of \( T \) with a deadline at or before \( t_h \) can be scheduled at \( t_h - 1 \), and hence, can be in \( \tau'_2 \cup \tau'_6 \). On the other hand, if a subtask of \( T \) with a deadline after \( t_h \) is scheduled at \( t_h - 1 \), then it contradicts the fact that \( T_i \) is \( T \)’s critical subtask at \( t_h \). So, no such subtask can be in \( \tau'_2 \cup \tau'_6 \) either. \( \square \)
Proof of part (b). By the conditions of Case C, no c-MI, where \( c > 0 \), is scheduled at \( t_h \). Further, because \( T \) is in \( A_q(t_h) \) or \( A_{q-1}(t_h) \), tardiness of \( T_i \) is greater than zero. Hence, by the definition of c-MI and because \( T \) is not a c-MI, \( T \) is also scheduled at \( t_h - 1 \). Therefore, \( T_i \) is in \( \tau^s \).

Proof of part (c). Immediate from the definitions.

Proof of part (d). By part (a), every task \( T \) has at most one subtask in \( \tau^1 \cup \tau^2 \cup \tau^6 \). Therefore, \( \tau^1 \), \( \tau^2 \), and \( \tau^6 \) are pairwise disjoint. By (25) and (26), \( A_0 \), \( A_q \), and \( A_{q-1} \) are pairwise disjoint, and hence, by their definitions, \( \tau^3 \), \( \tau^4 \), and \( \tau^5 \) are pairwise disjoint, and subtasks in them are scheduled at \( t_h \). However, by the definitions of \( \tau^3 \), \( \tau^4 \), and \( \tau^5 \), no task of a subtask in any of these subsets is scheduled at \( t_h \). Therefore, a task in \( \tau^1 \), \( \tau^2 \), or \( \tau^6 \) is not in \( \bigcup_{i=3}^{5} \tau^i \), that is \( \tau^1 \cup \tau^2 \cup \tau^6 \) is disjoint from \( \bigcup_{i=3}^{5} \tau^i \). Since \( \tau^1 \), \( \tau^2 \), and \( \tau^6 \) are pairwise disjoint, as are \( \tau^3 \), \( \tau^4 \), and \( \tau^5 \), all six subsets are pairwise disjoint.

We make the following two claims before proving the remaining lemmas.

Claim 4. No subtask with deadline at or before \( t_h - 1 \) is removed or displaced in \( S' \).

Proof. It follows from the fact that the deadline of every subtask removed, that is, the deadline of every subtask in \( \tau^R \) (refer 45), is at or after \( t_h \). Hence, because ties in \( S \) and \( S' \) are resolved identically, the removed subtasks cannot impact how subtasks with earlier deadlines are scheduled, and hence, cannot cause such subtasks to be displaced. (Subtasks in \( \tau^i \) are critical subtasks at \( t_h \) or at \( t_h - 1 \), and hence their deadlines are at or after \( t_h \). Similarly, subtasks in \( \tau^5 \) are scheduled at \( t_h \) and have a tardiness of zero, implying that their deadlines are at or after \( t_h + 1 \).)

Claim 5. The release time of every subtask in \( \tau \) is at or before \( t_h \).

Proof. Because there is a hole in \( t_h \) (by (H)), by Lemma 15, no subtask scheduled at or before \( t_h \) can have a deadline after \( t_h + 1 \), implying that the release time of every such subtask is at or before \( t_h \). Hence, a subtask with release time after \( t_h \) is scheduled after \( t_h \) in \( S' \). For every such subtask, allocations in both the ideal schedule and \( S \) are zero in \( [0, t_h + 1] \). Therefore, the LAG of \( \tau \) at \( t_h + 1 \) does not depend on such a subtask. Further, if such a subtask is removed, the schedule before \( t_h + 1 \) is not impacted and no subtask scheduled at or after \( t_h + 1 \) can shift to \( t_h \) or earlier. Hence, the LAG of \( \tau \) at \( t_h + 1 \) is not altered. Thus, the presence of subtasks released after \( t_h \) contradicts (T2).

Lemma 29. Let \( T \) be a task with a subtask in \( \tau^1 \) or \( \tau^2 \). Then, \( \text{lag}(T, t_h - 1, S') = \text{lag}(T, t_h + 1, S) \).

Proof. By (11),
\[
\text{lag}(T, t_h - 1, S') = A(\text{ideal}_f^l, T, 0, t_h - 1) - A(S', T, 0, t_h - 1). \tag{59}
\]
To prove this lemma, we will express the allocation to \( T \) in \( \text{ideal}_f^l \) and \( S' \) in terms of its allocations in \( \text{ideal}_f \) and \( S \), respectively. We will establish some properties needed for this purpose.

By Lemma 28(a), \( T \) has exactly one subtask in \( \tau^1 \cup \tau^2 \). Let \( T_i \) denote the distinct subtask of \( T \) that is in \( \tau^1 \) or \( \tau^2 \), and \( T_j \), its predecessor in \( \tau^3 \) or \( \tau^5 \), respectively. If any. Note that \( T_j \) does not exist if \( d(T_i) = t_h \), and need not necessarily exist otherwise.

Regardless of whether \( T_i \) is in \( \tau^1 \) or \( \tau^2 \), \( T_i \) is scheduled at or before \( t_h' \) in \( S \), which by (44), is before \( t_h - 1 \). Hence, because there is a hole in \( t_h \), by Lemma 15, \( d(T_i) \leq t_h + 1 \) holds. We next show that the following holds.

(D) No subtask of \( T \) has its deadline after \( t_h + 1 \).

Since \( T \) is not scheduled in \( t_h \) and there is a hole in \( t_h \), \( T_i \)'s successor, if any, cannot have its eligibility time at or before \( t_h \) and deadline after \( t_h + 1 \). By Claim 5, no subtask in \( \tau \) has a release time at or after \( t_h + 1 \). Thus, (D) holds.

We next claim that of \( T \)'s subtasks, only \( T_i \) and/or \( T_j \) may receive non-zero allocations in the ideal schedule for \( \tau \) in slots \( t_h - 1 \) and/or \( t_h \). For this, note that the following hold: (i) since \( d(T_j) = t_h \) (by the definitions of \( \tau^3 \) and \( \tau^5 \)), no subtask of \( T \) prior to \( T_j \) has its deadline after \( t_h - 1 \); (ii) because there is a hole in \( t_h \), and \( T \) is not scheduled at \( t_h \) in \( S \) (by the definitions of \( \tau^1 \) and \( \tau^2 \)), no subtask of \( T \) released after \( T_i \) has its eligibility time, and hence, release time at or before \( t_h \). Hence, by (6), no subtask of \( T \) other than \( T_i \) and \( T_j \) receives any allocation in \( t_h - 1 \) and/or \( t_h \). By (i) and (ii) above and because \( \tau \) contains every subtask of \( T \) that is in \( \tau \) except \( T_i \) and \( T_j \), we have \( A(\text{ideal}_f^l, T, 0, t_h - 1) = A(\text{ideal}_f^l, T, 0, t_h + 1) - A(T_i, 0, t_h + 1) - A(T_j, 0, t_h + 1) \). Because the deadlines of \( T_i \) and \( T_j \) are at most \( t_h + 1 \), both these subtasks receive ideal allocations of one quantum each by \( t_h + 1 \). Hence,
\[
A(\text{ideal}_f^l, T, 0, t_h - 1) = \begin{cases} A(\text{ideal}_f^l, T, 0, t_h + 1) - 2, & \text{if } T_j \text{ exists;} \\ A(\text{ideal}_f^l, T, 0, t_h + 1) - 1, & \text{if } T_j \text{ does not exist.} \end{cases} \tag{60}
\]
We now express the allocation to $T$ in $S'$ in terms of its allocation in $S$. If $T_i$ is in $\tau_1^S$, then, in $S$, $T_i$ is scheduled at or before $t_{i}^* - (q + 3) \leq t_h - 1$ (refer (44)); if it is in $\tau_2^S$, then $T_i$ is scheduled at $t_h - 1$. Thus, in either case $T_i$ is scheduled at or before $t_{i}^* - 1$ in $S$. Hence, $T_j$, if it exists, is scheduled at or before $t_h - 1$ in $S$. As for where other subtasks of $T$ are scheduled in $S$, there is a hole in $t_h$, and (by the definitions of $\tau_1^S$ and $\tau_2^S$) $T$ is not scheduled at $t_h$. Therefore, if some subtask of $T$ is scheduled after $t_h$, then its eligibility time is at or after $t_h + 1$, and hence its deadline is after $t_h + 1$. However, by (D), no subtask of $T$ has a deadline after $t_h + 1$. Hence, there does not exist a subtask of $T$ that is scheduled after $t_h$ in $S$, which implies that there does not exist a subtask of $T$ that is scheduled after $t_h$ in $S'$ and before $t_h - 1$ in $S'$. Further, because no subtask can displace to the right, there does not exist a subtask of $T$ that is scheduled before $t_h - 1$ in $S'$, and at or after $t_h - 1$ in $S'$. As already mentioned, every subtask of $T$ except $T_i$ and $T_j$ is present in $\tau'$. Therefore,

$$A(S', T, 0, t_h, 1) = \begin{cases} A(S, T, 0, t_h + 1) - 2, & \text{if } T_j \text{ exists}, \\ A(S, T, 0, t_h + 1) - 1, & \text{if } T_j \text{ does not exist}. \end{cases} \tag{61}$$

By (59)-(61), regardless of whether $T_j$ exists, $\ell(T, t_h - 1, S') = A(\{T_i\}, T, 0, t_h + 1) - A(S, T, 0, t_h + 1, S) = \ell(T, t_h + 1, S) - 1/(q + 2)$.

**Lemma 30.** Let $T$ be a task with a subtask in $\tau_3^S$. Then, $\ell(T, t_h - 1, S') > \ell(T, t_h + 1, S) - 1/(q + 2)$.

**Proof.** Let $T_i$ be $T$’s subtask in $\tau_3^S$. In $S$, $T_i$ is scheduled at $t_h$ and is ready at or before $t_h - (q + 3)$. Therefore, by Lemma 7(a), $r(T_i) \leq t_h - (q + 3)$ holds. Since $T$ is in $A_0(t_h)$, and $T_i$ is scheduled at $t_h$ in $S$, the tardiness of $T_i$ is zero in $S$. Therefore, $d(T_i) \geq t_h + 1$ holds, which by (H) and Lemma 16 implies that

$$d(T_i) = t_h + 1. \quad \tag{62}$$

Hence, $|\omega(T_i)| = d(T_i) - r(T_i) \geq q + 4$ holds, and using Lemma 1, it can be shown that $\ell(T, t_h + 1, S) - 1 < (q + 2)$ holds. By Lemma 12(c), $\ell(T, t_h + 1, S) < \ell(T, t_h + S) < \ell(T, t_h + 1, S) - 1/(q + 2)$, and hence, because $\ell(T, t_h + S) < 1/(q + 2)$, it follows that

$$\ell(T, t_h + 1, S') < 1/(q + 2). \quad \tag{63}$$

We next show that $\ell(T, t_h - 1, S') = 0$. For this, we need to show that the total allocation to $T$ in $[0, t_h - 1)$ is equal in ideal $\tau'$ and $S'$. We first show that the total allocation in $[0, t_h - 1)$ to subtasks of $T$ released after $T_i$ is zero in both $S'$ and ideal $\tau'$. By (62) and Lemma 2, the release time of the successor, $T_j$, if any, of $T_i$ is at or after $t_h$. Hence, the allocation to every subtask of $T$ released after $T_i$ is zero in $[0, t_h - 1)$ in the ideal schedule for $\tau'$. Also, because $T_i$ is scheduled at $t_h$ in $S$, $T_j$ is scheduled at or after $t_h + 1$ in $S$. Hence, by Lemma 7(a), $e(T_j) \geq t_h$ holds. Therefore, every subtask of $T$ released after $T_i$ is scheduled at or after $t_h$ in $S$, that is, receives zero allocation in $[0, t_h - 1)$ in $S'$.

We now show that subtasks of $T$ released before $T_i$ receive equal allocations in $[0, t_h - 1)$ in both ideal $\tau'$ and $S'$. Since $T_i$ is ready at or before $t_h - (q + 3)$, $T_i$’s predecessor, if any, and all prior subtasks of $T$, if any, complete executing at or before $t_h - (q + 3)$ in $S$, and hence, in $S'$, as well (because no subtask can displace to the right). Furthermore, as discussed above, $r(T_i) \leq t_h - (q + 3)$ holds, and hence, by Lemma 2, the deadline of $T_i$’s predecessor is at or before $t_h - (q + 2)$. Hence, all subtasks released before $T_i$ complete executing by $t_h + (q + 2)$ in ideal $\tau'$ as well.

Therefore, because $T_i$ is not present in $\tau'$, the total allocation to all the subtasks of $T$ in $\tau'$ in $[0, t_h - 1)$ is equal in $S'$ and ideal $\tau'$. Hence, $\ell(T, t_h - 1, S') = 0$, and because (63) holds, the lemma follows. □

**Lemma 31.** Let $T$ be a task with a subtask in $\tau_4^S$. Then, $\ell(T, t_h - 1, S') \geq \ell(T, t_h + 1, S) - 2 \cdot W_{max} + 1$.

**Proof.** First, we show that (R) below holds.

(R) No subtask of $T$ is removed.

For this, note that because $T$ is in $\tau_4^S$, by Lemma 28(d), $T$ is not in $\tau_i^S$, where $1 \leq i \leq 6$ and $i \neq 4$. Hence, by Lemma 28(c), $T$ is also not in $\tau_i^S$ or in $\tau_i^S$. Thus, $T$ does not have a subtask in $\tau_i^S$, and hence, (R) holds.

Let $T_i$ be $T$’s subtask in $\tau_4^S$ and let $t_c = t_h - (q + 3)$. Then, $T_i$ is not ready at $t_c$ in $S$. We show that $T_i$ is not ready at $t_c$ in $S'$ also. Let $T_j$ denote $T_i$’s predecessor, if any, in $\tau$.

We now show that no subtask of $T$ that is scheduled at or after $t_h - 1$ in $S$ is scheduled before $t_h - 1$ in $S'$. Note that $T_i$ is scheduled at $t_h$ in $S$. Hence, it suffices to show that $T_i$ is not scheduled before $t_h - 1$ in $S'$ (which would imply that no later subtask is scheduled before $t_h - 1$), and if $T_j$ is scheduled at $t_h - 1$ in $S$, then it is not scheduled earlier in $S'$.

Because $T_i$ is scheduled at $t_h$ in $S$ and $T_j$ is not ready at $t_c$ in $S$, Lemma 7(a) implies that either $r(T_j) > t_c$, or $r(T_j) \leq t_c$ and $T_j$ exists and does not complete executing by $t_c$. If the former holds, then because $r(T_j) > t_c$ and $T_i$ is scheduled at $t_h > t_h - (q + 3) = t_c$ in $S$, by Lemma 7(a), $e(T_j) > t_c$ holds, and hence, $T_i$ is not eligible, and hence, not ready, at $t_c$ in $S'$ either. If the latter holds, then by Lemma 2, $d(T_j) < t_c + 1 \leq t_h - (q + 2)$ holds, and hence, by Claim 4, $T_j$ is not displaced, and does not complete executing by $t_c$ in $S'$ also. Therefore, $T_i$ is not ready at $t_c$ in this case too.
Given that $T_i$ is not ready at $t_e$ in $S'$, it is easy to show that $T_i$ is not scheduled before $t_h - 1$ in $S'$. For this, note that by Claim 4, no subtask with deadline at or before $t_h - 1$ is displaced or removed. Hence, since (C) holds, no subtask scheduled in $[t_h - (q + 2), t_h - 1)$ is displaced or removed. Therefore, because $T_i$ is not ready at or before $t_e = t_h - (q + 3)$, $T_i$ cannot be scheduled before $t_h - 1$ in $S'$.

We next show that if $T_i$'s predecessor $T_j$ exists and is scheduled at $t_h - 1$ in $S$, then it is not scheduled earlier in $S'$. Because $T_i$ is scheduled at $t_h$ and $T$ is in $A_0(t_h)$, $T_i$'s tardiness is zero, and hence, by Lemma 16, $d(T_i) = t_h + 1$. Hence, $d(T_j) = t_h$ holds. If $d(T_j) < t_h$ holds, then, by Claim 4, $T_j$ is not displaced. In the other case, namely, $d(T_j) = t_h$, by Lemma 2, $r(T_j) = t_h - 1$, and hence, $|o(T_j)| = d(T_j) - r(T_j) = (t_h - t_h) = -2$ holds. Therefore, by Lemma 1, $|o(T_j)| < 3$, and hence, $r(T_j) = d(T_j) - 2 = t_h - 3$. If $T_j$ is scheduled at $t_h - 1$ in $S$, then by Lemma 7(a), $e(T_j) > t_h - 3$. However, because $q > 1$, by (C), the deadline of every subtask scheduled in $[t_h - 3, t_h - 1)$ is at or before $t_h - q$, and hence, by Claim 4, no such subtask is displaced or removed. Therefore, in this case too, if $T_j$ is scheduled at $t_h - 1$ in $S$, it is not scheduled earlier in $S'$. Thus, no subtask of $T$ that is scheduled at or after $t_h - 1$ in $S$ is scheduled before $t_h - 1$ in $S'$.

We are now ready to establish the lag of $T$ at $t_h - 1$ in $S'$. By (11), we have

$$\text{lag}(\tau', t_h - 1, S') = A(\text{id}eal_{\tau'}, T, 0, t_h - 1) - A(S', T, 0, t_h - 1)$$

$$= A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - A(\text{id}eal_{\tau'}, T, t_h - 1, t_h + 1) - (A(S, T, 0, t_h + 1) - A(S, T, t_h - 1, t_h + 1))$$

(because, by (R), no subtask of $T$ is removed, and no subtask of $T$ scheduled at or after $t_h - 1$ in $S$ is scheduled before $t_h - 1$ in $S'$)

$$\geq A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - 2 \cdot W_{\max} - (A(S, T, 0, t_h + 1) - A(S, T, t_h - 1, t_h + 1))$$

(by (7) and (29))

$$\geq A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - 2 \cdot W_{\max} - A(S, T, 0, t_h + 1) + 1$$

(because at least subtask $T_i$ of $T$ is scheduled in $[t_h - 1, t_h + 1)$ in $S'$)

$$= \text{lag}(T, t_h + 1, S') - 2 \cdot W_{\max} + 1. \quad \Box$$

**Lemma 32.** Let $T$ be a task with a subtask in $\tau^2_5$. Then, $\text{lag}(T, t_h - 1, S') \geq \text{lag}(T, t_h + 1, S') + 2 - 2 \cdot W_{\max}$.

**Proof.** As with Lemma 31, we first show that no subtask of $T$ is removed. Because $T$ is in $\tau^5$, by Lemma 28(d), $T$ is not in $\tau^i$, where $1 \leq i \leq 6$ and $i \neq 5$. Hence, by Lemma 28(c), $T$ is also not in $\tau^7$ or in $\tau^8$. Thus, $T$ does not have a subtask in $\tau^5$, and hence, no subtask of $T$ is removed.

We next show that the subtasks of $T$ scheduled at $t_h$ or $t_h - 1$ are not displaced.

Let $T_j$ be $T$'s subtask scheduled at $t_h$. By the definition of $A_q$ and $A_{q - 1}$, the tardiness of $T_j$ is greater than zero, and hence, $d(T_j) = t_h$. Let $T_j$ be $T$'s predecessor. By the definition of $\tau^5$, $T_j$ exists. Further, $d(T_j) = t_h$ holds and $T_j$ is scheduled at $t_h - 1$.

We now show that $T_j$ and $T_j$ are not displaced. For this, observe that because $d(T_j) = t_h - 1$ holds, $T_j$ is not displaced by Claim 4. Therefore, because $T_j$ is $T$'s successor, $T_j$ is not ready to be scheduled until $t_h$, and hence, is not displaced either.

The above facts can be used to determine the lag of $T$ at $t_h - 1$ in $S'$ as follows. By (11), we have

$$\text{lag}(\tau', t_h - 1, S') = A(\text{id}eal_{\tau'}, T, 0, t_h - 1) - A(S', T, 0, t_h - 1)$$

$$= A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - A(\text{id}eal_{\tau'}, T, t_h - 1, t_h + 1) - (A(S, T, 0, t_h + 1) - A(S, T, t_h - 1, t_h + 1))$$

(because no subtask of $T$ is removed, and because neither $T_i$ nor $T_j$ is displaced, no subtask of $T$ scheduled at or after $t_h - 1$ in $S$ is scheduled before $t_h - 1$ in $S'$)

$$\geq A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - 2 \cdot W_{\max} - (A(S, T, 0, t_h + 1) - A(S, T, t_h - 1, t_h + 1))$$

(by (7) and (29))

$$= A(\text{id}eal_{\tau'}, T, 0, t_h + 1) - 2 \cdot W_{\max} - A(S, T, 0, t_h + 1) + 2$$

(because exactly two subtasks of $T$, $T_i$ and $T_j$, are scheduled in $[t_h - 1, t_h + 1)$)

$$= \text{lag}(T, t_h + 1, S') - 2 \cdot W_{\max} + 2. \quad \Box$$

**Lemma 33.** Let $T$ be a task with a subtask in $\tau^5_5$. Then, $\text{lag}(T, t_h - 1, S') > \text{lag}(T, t_h + 1, S)$.

**Proof.** Let $T_i$ denote $T$'s subtask in $\tau^5_5$. Because there is a hole in $t_h$ (by (H)) and $T$ is not scheduled at $t_h$, the eligibility time, and hence, the release time of $T_i$'s successor is at least $t_h + 1$. However, by Claim 5, the release time of every subtask in $\tau$ is at most $t_h$. Therefore, $T_i$ does not have a successor.

Since $T_i$ is not in $\tau^2_5$, $d(T_i) \leq t_h - 1$ holds. Thus, all subtasks of $T$ have their deadlines by $t_h - 1$ and complete executing by $t_h$ in both $\text{id}eal_{\tau'}$ and $S$. Therefore, $T$'s lag at $t_h + 1$ in $S$ is zero.
Because \( d(T_i) \leq t_h - 1 \) and \( T_i \) does not have a successor, by Claim 4, no subtask of \( T \) is displaced. Thus, in the ideal schedule for \( T' \), subtasks of \( T \) complete executing by \( t_h - 1 \), whereas \( T_i \) is not complete until \( t_h - 1 \) in \( S' \). Thus, \( \text{lag}(T, t_h - 1, S') > 0 \), from which the lemma follows. \( \square \)

**Lemma 34.** Let \( T \) be a task in \( T \). Then, \( \text{lag}(T, t_h - 1, S') = \text{lag}(T, t_h + 1, S') \).

**Proof.** Because \( T \) is in \( T \), \( T \) does not contain a subtask in sets \( \tau_i \), where \( 1 \leq i \leq 8 \). Hence, \( T \) does not have a subtask that is removed. We next show that \( T \) does not have a subtask that is scheduled at \( t_h \) or \( t_h - 1 \).

If \( T \) has a subtask \( T_i \) that is scheduled at \( t_h \), then \( T \) is in \( A \). By the condition of this case (Case C), \( A^0_0 = \emptyset \) and \( A^0_{q-1} = \emptyset \). Hence, by (25), \( T \) is in one of \( A_0(t_h), A_1^0(t_h), A_2^0(t_h), \) and \( A_3^0(t_h) \), where \( i \geq 1 \). However, if \( T \) is in \( A_0(t_h) \), then \( T_i \) is in \( \tau_3^3 \) or \( \tau_4^4 \). On the other hand, if \( T \) is in one of the remaining sets, then \( T_i \) has a tardiness greater than zero, but is not a c-ML, and hence, \( T \) is scheduled at \( t_h - 1 \); therefore, \( T_i \) is in \( \tau_3^3 \). Thus, \( T_i \) is in one of \( \tau_3^3 \), \( \tau_4^4 \), and hence, \( T \) is in one of \( \tau_3^3 \), \( \tau_4^4 \), and \( \tau_5^5 \). This contradicts the fact that \( T \) is in \( T \). Therefore, \( T \) cannot have a subtask scheduled at \( t_h \).

We now show that \( T \) does not have a subtask scheduled at \( t_h - 1 \). By the definitions of \( \tau_3^3 \) and \( \tau_5^5 \), any subtask that is scheduled at \( t_h - 1 \), but does not have a later subtask of its task scheduled at \( t_h \), is in one of these two subsets. Therefore, if \( T \) has a subtask \( T_i \) scheduled at \( t_h - 1 \), then \( T \) is in \( T \) (and hence not in \( T \) or \( T \)), \( T \) is scheduled at \( t_h \) also. But as was shown above, \( T \) is not scheduled at \( t_h \), and hence, is not scheduled at \( t_h - 1 \) either. Thus, \( T \) is not scheduled in either \( t_h \) or \( t_h - 1 \).

By Claim 5, no subtask of \( T \) is released at or after \( t_h - 1 \). Therefore, because there is a hole in \( t_h \), and \( T \) is not scheduled in either \( t_h \) or \( t_h - 1 \), every subtask of \( T \) is scheduled before \( t_h - 1 \), and completes executing by \( t_h - 1 \) in \( S \). Hence, because there is a hole in \( t_h \), by Lemma 15, the deadline of every subtask of \( T \) is at or before \( t_h - 1 \).

To complete the proof, we show that the deadline of every subtask of \( T \) is at most \( t_h - 1 \). Suppose to the contrary some subtask of \( T \) has its deadline after \( t_h - 1 \). Let \( T_i \) be such a subtask with the largest index. Then, \( T_i \) is the critical subtask of \( T \) at either \( t_h \) or \( t_h - 1 \) or at both times. Because \( T \) is not scheduled at either \( t_h \) or \( t_h - 1 \), \( T_i \) is scheduled before \( t_h - 1 \). Hence, \( T \) is in \( B(t_h - 1) \) or \( B(t_h) \) or both. Also, because \( d(T_i) \geq t_h \), by Definition 9, \( t_h ^{\prime} \) exists and \( T \) is scheduled at or before \( t_h ^{\prime} \). But then, by the definition of \( t_h ^{\prime} \), \( T_i \) is in \( \tau_3^3 \), which contradicts the fact that \( T_i \) is in \( T \). Therefore, our assumption that \( T \) has a subtask with deadline after \( t_h - 1 \) is incorrect.

Thus, all subtasks of \( T \) complete executing by \( t_h - 1 \) in both the ideal schedules. Hence, the \( \text{lag} \) of \( T \) in \( S \) at \( t_h - 1 \) is zero.

Because no subtask of \( T \) is removed or displaced, and every subtask of \( T \) is scheduled before \( t_h - 1 \) in \( S \), therefore, all subtasks of \( T \) complete executing by \( t_h - 1 \) in \( S' \) also. Thus, \( \text{lag}(T, t_h - 1, S') = 0 \). The lemma follows. \( \square \)

**Lemma 35.** The roots of \( f(W_{\text{max}}) = 2(M - h)(q + 1)W_{\text{max}}^2 - (q + 2)(M - h)W_{\text{max}} - ((q - 1)M + 1 + h) = 0 \) are \( W_{\text{max}} = \frac{(q + 2)(M - h) - \sqrt{9q^2(M - h)^2 + 8(M - h)q + 4(q + 1)(q - 1)M + 1 + h}}{4(M - h)(q + 1)} \), where \( \Delta = 4(M - h)(M(q - 1) + h)(2q^2 + 2q + 1) + 2q + 2 \).

**Proof.** The roots of \( f(W_{\text{max}}) \) are given by \( W_{\text{max}} = \frac{(q + 2)(M - h) - \sqrt{(q + 2)^2(M - h)^2 + 8(M - h)(q + 1)(q - 1)M + 1 + h}}{4(M - h)(q + 1)} \). Let \( I = (q + 2)^2(M - h)^2 + 8(M - h)(q + 1)(q - 1)M + 1 + h \) (the term within the square root). Then,

\[
I = (q + 2)^2(M - h)^2 + 8(M - h)(q + 1)(q - 1)M + 1 + h
\]

\[
= q^2(M - h)^2 + (4q + 4)(M - h)^2 + 8q^2(M - h)^2 + 8(M - h)(q + 1)(q - 1)M + 1 + h
\]

\[
= 9q^2(M - h)^2 + 4(M - h)(q + 1)(q - 1)M + 1 + h + 2q + 2
\]

\[
= 9q^2(M - h)^2 + \Delta. \quad \square
\]

(64)

**References**


