Information and Computation 281 (2021) 104743

Contents lists available at ScienceDirect

Information and Computation

www.elsevier.com/locate/yinco

Narrowing the speedup factor gap of partitioned EDF

Xingwu Liu [a], Xin Han [b,*], Liang Zhao [c], Zhishan Guo [d,*]

[a] School of Mathematical Sciences, Dalian University of Technology, China SKL Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences, University of Chinese Academy of Sciences, Beijing, China
[b] Key Laboratory for Ubiquitous Network and Service Software of Liaoning Province, Software School, Dalian University of Technology, Dalian, China
[c] Software School, Dalian University of Technology, Dalian, China
[d] Department of Electrical and Computer Engineering, University of Central Florida, Orlando, FL 32766, USA

ARTICLE INFO

Article history:
Received 19 November 2020
Accepted 17 March 2021
Available online 26 March 2021

Keywords:
Real-time sporadic tasks
Resource augmentation bound
Partitioned scheduling
Approximate demand bound function

ABSTRACT

Schedulability is a fundamental problem in analyzing real-time systems, but it often has to be approximated because of the intrinsic computational hardness. Partitioned earliest deadline first (EDF) is one of the most popular polynomial-time and practical scheduler on multiprocessor platforms, and it was shown to have a speedup factor of at most $2.6322 - 1/m$. This paper further improves the factor to $2.5556 - 1/m$ for both the constrained-deadline case and the arbitrary-deadline case, and it is very close to the known (non-tight) lower bound of $2.5 - 1/m$. The key ideas are that we develop a novel method to discretize and regularize sporadic task sets that are schedulable on uniprocessors, and we find that the ratio (ρ) of the approximate demand bound value to the machine capacity is upper-bounded by 1.5556 for the arbitrary-deadline case, which plays an important role in estimating the speed factor of partitioned EDF.

© 2021 Elsevier Inc. All rights reserved.

1. Introduction

Scheduling plays a fundamental role in real-time systems. Basically, given a finite set of tasks, each sequentially releasing infinitely many jobs, the mission of real-time scheduling is to allocate computing resources so that all the jobs are done in a timely manner. Formally, a schedule defines at each time instant which jobs receive the required computing resources (while others must wait). The fundamental question of schedulability naturally arises: is it possible at all to successfully schedule these tasks so as to meet all the deadlines?

Unfortunately, answering this question is often not "easy"; for example, the schedulability of a set of constrained-deadline[1] sporadic tasks, which is the focus of this article, is co-NP-hard even on a uniprocessor platform [2]. For the multiprocessor case, it remains NP-hard for partitioned scheduling, even for implicit-deadline task sets, where the relative deadline of each task equals its period [3]. Here *partitioned* scheduling means that once a task is assigned to a processor, all the jobs released by the task will be scheduled on the dedicated processor. These hardness results imply that it is impossible to exactly decide schedulability in polynomial time, unless P=NP.

* Corresponding authors.
E-mail addresses: hanxin@dlut.edu.cn (X. Han), zsguo@ucf.edu (Z. Guo).

[1] A set of tasks is said to be a constrained-deadline task set if the relative deadline of each task is at most its period (otherwise it is a arbitrary-deadline task set).

https://doi.org/10.1016/j.ic.2021.104743
0890-5401/© 2021 Elsevier Inc. All rights reserved.

Because of the hardness, real-time schedulability problems are usually solved approximately by pessimistic algorithms that always answer "no" unless some sufficient conditions for schedulability are met. To evaluate the performance of such an approximate algorithm (say, $\mathcal{A}$), the concept of the *speedup factor*, also known as the resource augmentation bound, has been proposed. Specifically, algorithm $\mathcal{A}$ has a speedup factor of $s \geq 1$ if whenever a set of tasks is schedulable (by an optimal approach) on a platform with speed 1, $\mathcal{A}$ will return "yes" when the speed of the platform is increased to s. Despite some recent discussion on potential pitfalls [4–6], the speedup factor has been a major metric and standard theoretical tool for assessing scheduling algorithms since the seminal work of Kalyanasundaram and Pruhs [7] in 2000.

Recent years have witnessed impressive progress in finding scheduling algorithms with low speedup factors. For preemptive scheduling (i.e., running jobs might be interrupted by emergent ones), global EDF has a speedup factor of $2 - 1/m$ [8] for scheduling tasks on m identical processors, and there is a polynomial-time algorithm for uniprocessors whose speedup factor is $1 + \epsilon$ [9], where $\epsilon > 0$ is arbitrarily small. For nonpreemptive scheduling, there are also a variety of results [10,11]. In addition to the speedup factor, there are several articles concerning the utilization bound [12–14].

Although the speedup factor on uniprocessors is already known to be tight, the multiprocessor case remains open. Among all schedulers, partitioned scheduling is of particular interest because of its implementation friendliness, simplicity, and capability of extending most uniprocessor results to the multiprocessor scenario directly under naive "partition" heuristics; i.e., once the task-to-core mapping is fixed, the scheduling in the multiprocessor case is reduced to multiple uniprocessor scheduling problems, where classical solutions exist. Since EDF is an optimal preemptive scheduler on a uniprocessor, this article focuses on partitioned EDF.[2] Note that partitioned-deadline-monotonic [15] is also commonly implemented, with a best known speedup factor of 2.8431, while global EDF is not a partitioned paradigm.

A breakthrough in partitioned EDF was made in 2005, when Baruah and Fisher [16] established an upper bound of $3 - 1/m$ for the speedup factor on constrained-deadline task sets and an upper bound of $4 - 2/m$ for the speedup factor on arbitrary-deadline task sets, where m is the number of identical processors. In 2011, Chen and Chakraborty [1] further improved the speedup factor to $2.6322 - 1/m$ for the constrained-deadline case and to $3 - 1/m$ for the arbitrary-deadline case. Also, they established an asymptotical lower bound of 2.5 for the speedup factor for the constrained-deadline case. Since then, the speedup factor bounds have never been improved.

Deriving the upper bound of the speedup factor of partitioned EDF relies heavily on a quantity ρ concerning scheduling on uniprocessors. The quantity ρ, called *the relaxation factor* in this article and formally defined in formula (1) in Section 2, roughly indicates how much the approximate demand bound function (defined in Section 2) deviates from the machine capacity. Baruah and Fisher [16] bridged the relaxation factor and the speedup factor of partitioned EDF by showing that in the case of constrained deadlines, the speedup factor is at most $1 + \rho - 1/m$. As a result, upper-bounding the speedup factor is reduced to upper-bounding of ρ, and it is in this manner that both Baruah and Fisher [16] and Chen and Chakraborty [1] obtained their estimates of the speedup factor. Hence, the relaxation factor itself deserves deep investigation. Baruah and Fisher [16] upper-bounded it by 2, and Chen and Chakraborty [1] narrowed its range to $[1.5, 1.6322]$.

On this ground, we explore a better upper bound of the relaxation factor, and on this basis provide a better estimate of the speedup factor of partitioned EDF for sets of constrained-deadline sporadic tasks. The contributions are summarized as follows:

1. We improve the best existing upper bound of the relaxation factor from 1.6322 to 1.5556 (Theorem 1), which is very close to the lower bound of 1.5 for the uniprocessor case. The result holds for both constrained-deadline tasks and arbitrary-deadline tasks. Accordingly, the speedup factor of partitioned EDF for constrained-deadline tasks decreases from $2.632 - 1/m$ to $2.5556 - 1/m$ (Theorem 2) for the multiprocessor case.
2. We identify a lossless way to discretize and regularize the tasks. As a result, the execution times of the tasks of interest can be fixed to be 1 and the deadlines can be fixed to be $1, 2, \ldots, n$, where n is number of tasks to be scheduled (Lemmas 3, 7, and 8). The only parameter that varies is the period. The transformation is lossless in the sense that the relaxation factor does not change although the parameters are extremely simplified.
3. We invent a method to further transform the tasks so that the period of each task ranges over integers between 1 and $2n$ (Lemma 9). Although this transformation is not guaranteed to be lossless, the loss, if any, is negligible since we prove that the relaxation factor increases by at most 0.0556 (for both constrained-deadline task sets and arbitrary-deadline task sets). These transformation techniques may be further applied to real-time scheduling analysis or other problems.

The rest of this article is organized as follows. Section 2 presents the model and preliminaries. Section 3 focuses on the uniprocessor case, and in it we derive a new upper bound (14/9) of the relaxation factor. Section 4 provides a new upper bound $(23/9 - 1/m)$ of the speedup factor for partitioned EDF. Finally, Section 5 concludes the article and provides some potential future directions.

[2] In partitioned EDF, each task is assigned one and only one processor for the execution of all the jobs this task releases, while on each processor the jobs are executed according to the earliest-deadline-first priority rule.

2. System model and preliminaries

We consider a finite set τ of sporadic tasks. Each task τ_i can be represented by a triple $\tau_i = (e_i, d_i, p_i)$, where e_i is the worst-case execution time, d_i is its relative deadline, and p_i is the minimum interarrival separation length (also known as the period). Such a task releases infinitely many jobs, each of which has an execution time of at most e_i and has to be finished within time d_i after arrival, while the interarrival time of consecutive jobs is at least p_i. The task τ_i is said to be a *constrained-deadline task* if $d_i \le p_i$, and an *arbitrary-deadline task* if no restriction is set between d_i and p_i. When $d_i > p_i$, a job cannot start its execution until its predecessor (released by the same task one period ahead) finishes its execution.

We follow the widely adopted identical multiprocessor model, which consists of $m \ge 1$ processors of speed s (unless explicitly mentioned, $s = 1$ by default). For any task (e, d, p), its jobs can be executed on any of the processors, and the execution of any job takes at most $\frac{e}{s}$ time units. The aim of schedulability testing is to decide weather a set of sporadic tasks is schedulable on a platform. Here *schedulable* means that there exists a schedule for the set of tasks such that each job can cumulatively receive enough execution time between its release and its deadline.

Given a set of tasks, a schedulability test is a set of conditions to check—it returns success when all the deadlines can be guaranteed to be met. A schedulability test has a *speedup factor* (also known as a resource augmentation factor) of s (≥ 1) if any task set that is schedulable on a unit-speed platform will successfully pass this test on a platform with speed s. Informally, the speedup factor measures how "far away" a given schedulability test is from an optimal one—it reflects the effectiveness of a schedulability test. A smaller speedup factor indicates a better schedulability test, while a speedup factor of 1 indicates an optimal test. Our objective is to estimate the speedup factor of partitioned EDF on multiprocessor platforms.

Before continuing, we introduce some notation. Given a task τ_i, the demand bound function $dbf(\tau_i, t)$ [17] and its approximation $dbf^*(\tau_i, t)$ [9] are defined to be

$$dbf(\tau_i, t) = \begin{cases} 0 & \text{if } t < d_i, \\ \left(\left\lfloor \frac{t-d_i}{p_i} \right\rfloor + 1 \right) \cdot e_i & \text{otherwise,} \end{cases}$$

$$dbf^*(\tau_i, t) = \begin{cases} 0 & \text{if } t < d_i, \\ \left(\frac{t-d_i}{p_i} + 1 \right) \cdot e_i & \text{otherwise.} \end{cases}$$

Roughly speaking, $dbf(\tau_i, t)$ represents the total workload of task τ_i that has to be finished by time t, and dbf^* is a linear approximation of dbf.

These functions can be extended to task sets. For any set τ of tasks, we define

$$dbf(\tau, t) = \sum_{\tau_i \in \tau} dbf(\tau_i, t), \quad dbf^*(\tau, t) = \sum_{\tau_i \in \tau} dbf^*(\tau_i, t).$$

It is well known that the demand bound function fully determines the schedulability on uniprocessors, according to the following lemma.

Lemma 1 *([17]). A set τ of tasks is schedulable on uniprocessors if and only if $dbf(\tau, t) \le t$ for any $t \ge 0$.*

We are now ready to define the relaxation factor ρ, which plays a critical role in fulfilling our objective in this article:

$$\rho = \sup_{\tau \in \Gamma} \frac{dbf^*(\tau, d)}{d}, \tag{1}$$

where Γ is the family of sporadic task sets that are schedulable on uniprocessors, and d is the largest relative deadline in τ. Roughly speaking, ρ approximately stands for the growth rate of the demand over $[0, d)$ of schedulable task sets. Such a growth rate will have larger values at some deadline points, and thus elaboration of all the deadlines (d) will suffice.

We will see that the relaxation factor ρ is the optimum value of the following mathematical program MP_0:

$$\sup \quad \frac{dbf^*(\tau, d_n)}{d_n} \qquad (MP_0) \tag{2}$$

$$\text{subject to} \quad dbf(\tau, t) \le t, \quad \forall t > 0, \tag{3}$$

$$d_i + p_i > d_n, \quad 1 \le i \le n-1, \tag{4}$$

$$d_1 \le d_2 \le \cdots \le d_n, \tag{5}$$

$$n \in \mathbb{Z}^+, e_i, d_i, p_i \in \mathbb{R}^+, \quad 1 \le i \le n, \tag{6}$$

where $\mathbb{Z}^+$ is the set of positive integers, while $\mathbb{R}^+$ stands for the set of positive real numbers (the superscript + in this article excludes 0). Condition (3) means τ is schedulable because of Lemma 1, and condition (4) means each task releases exactly one job during the period $[0, d_n)$.

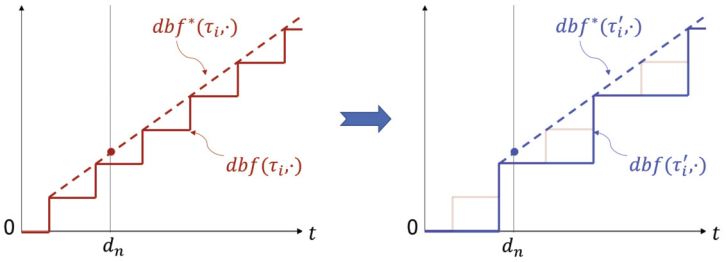

Fig. 1. Illustration of the task transformation in Lemma 2.

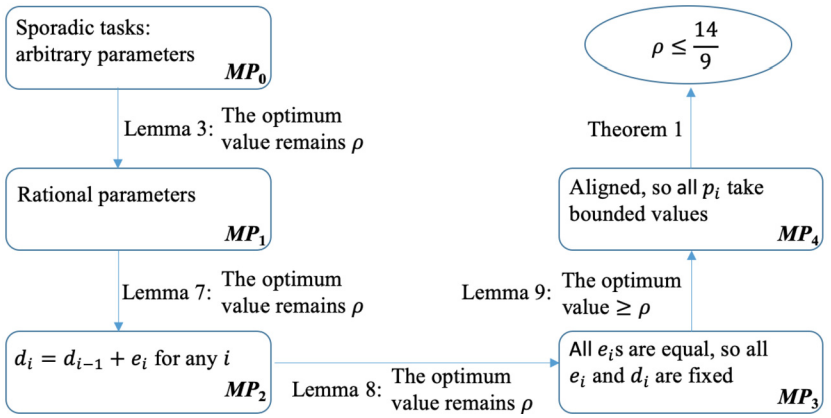

Fig. 2. The flow of the proofs in Section 3. The constraints are added incrementally, so each box presents only the new constraint. The overall constraints in each box are formulated into a mathematical program whose name MP_* is given at the lower-right corner of the box.

Lemma 2. *The relaxation factor is the optimum value of* MP_0.

Proof. Let $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \leq i \leq n\}$ be an arbitrary set of sporadic tasks that is schedulable on a uniprocessor with speed 1. Assume that $d_1 \leq d_2 \leq \cdots \leq d_n$. Apply the transformation proposed in [1]:

$$e'_i = \left(\left\lfloor \frac{d_n - d_i}{p_i} \right\rfloor + 1\right) \cdot e_i, \tag{7}$$

$$p'_i = \left(\left\lfloor \frac{d_n - d_i}{p_i} \right\rfloor + 1\right) \cdot p_i, \tag{8}$$

$$d'_i = \left(\left\lfloor \frac{d_n - d_i}{p_i} \right\rfloor\right) \cdot p_i + d_i. \tag{9}$$

Let $\tau' = \{\tau'_1, \tau'_2, \ldots, \tau'_n\}$, with $\tau'_i = (e'_i, d'_i, p'_i)$ for any $1 \leq i \leq n$. The transformation is illustrated in Fig. 1. The underlying idea is to increase the values of the parameters e_i, d_i, and p_i such that each task releases exactly one job before d_n while the system is as busy as before.

It was proven in [1] that the following results hold simultaneously:

1. $dbf^*(\tau, t) = dbf^*(\tau', t)$ for any $t \geq d_n$.
2. $dbf(\tau, t) \geq dbf(\tau', t)$ for $t > 0$.
3. $d'_n < d'_i + p'_i$ for $1 \leq i \leq n$.
4. $d'_n = d_n$.

This immediately leads to our lemma. □

3. Improved upper bound of the relaxation factor

To estimate the speedup factor for multiprocessor partitioned scheduling, we first analyze the relaxation factor and hence focus on uniprocessors. The main result of this section is Theorem 1, which establishes 14/9 as an upper bound of the relaxation factor for sporadic tasks.

The basic idea of our proof is to discretize any given task set into a regular form, thus reducing the problem to an optimization one on bounded integers with several constraints (MP_4). Roughly speaking, Lemma 3 ensures that the optimum value remains ρ if the parameters of the tasks are restricted to be rational numbers. Lemma 7 claims that further requiring $d_i = e_i + d_{i-1}$ for all i keeps the optimum value unchanged. The trend continues in Lemma 8 even if all the tasks are required to have the same worst-case execution time. Finally, Lemma 9 enables us to consider only tasks with bounded periods. These transformations reduce the estimation of ρ to a simpler optimization problem that is solved approximately in Lemma 11. These results immediately lead to Theorem 1. The overall proof flow is illustrated in Fig. 2.

3.1. Rationalizing the parameters

We first observe that the optimum value of MP_0 remains unchanged even if the domain $\mathbb{R}^+$ is replaced by $\mathbb{Q}^+$, the set of positive rational numbers.

$$\sup \quad \frac{dbf^*(\tau, d_n)}{d_n} \qquad (MP_1) \tag{10}$$

$$\text{subject to} \quad dbf(\tau, t) \le t, \quad \forall t > 0, \tag{11}$$

$$d_i + p_i > d_n, \quad 1 \le i \le n-1, \tag{12}$$

$$d_1 \le d_2 \le \cdots \le d_n, \tag{13}$$

$$n \in \mathbb{Z}^+, e_i, d_i, p_i \in \mathbb{Q}^+, \quad 1 \le i \le n. \tag{14}$$

Lemma 3. *MP_0 and MP_1 have the same optimum value.*

Proof. The lemma immediately holds if the following two claims are true:

1. The objective functions of MP_0 and MP_1 are the same and continuous.
2. The domain of MP_1 is a dense subset of that of MP_0. *Dense* means that for any $\epsilon > 0$ and any feasible solution $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ to MP_0, there is a feasible solution $\tau' = \{\tau_i' = (e_i', d_i', p_i') : 1 \le i \le n\}$ to MP_1 such that for any $1 \le i \le n$,

$$|e_i' - e_i| < \epsilon, |d_i' - d_i| < \epsilon, |p_i' - p_i| < \epsilon. \tag{15}$$

It suffices to prove claim 2 since claim 1 obviously holds.

Let $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ be an arbitrary set of tasks that is a feasible solution to MP_0, and let ϵ be an arbitrary positive real number. Without loss of generality, assume that $\epsilon < \min_{1 \le i \le n} e_i$. For any $1 \le i \le n$, arbitrarily choose

$$p_i' \in \left(p_i + \frac{\epsilon}{2}, p_i + \epsilon\right) \cap \mathbb{Q}^+,$$

$$d_i' \in \left(d_i + \frac{(i-1)\epsilon}{2n}, d_i + \frac{i\epsilon}{2n}\right) \cap \mathbb{Q}^+,$$

$$e_i' \in (e_i - \epsilon, e_i) \cap \mathbb{Q}^+.$$

Obviously, we have $p_i' > p_i, d_i' > d_i, e_i' < e_i$. Let τ' denote the set of tasks $\{\tau_i' = (e_i', d_i', p_i') : 1 \le i \le n\}$.

We now show that τ' is a feasible solution to MP_1. Since τ' meets conditions (14) and (15) by definition, it is enough to check conditions (11)–(13).

To continue, arbitrarily fix an integer $1 \le i \le n$.

Observe that

$$d_i' > d_i + \frac{(i-1)\epsilon}{2n} \ge d_{i-1} + \frac{(i-1)\epsilon}{2n} > d_{i-1}'.$$

Hence, τ' satisfies condition (13) of MP_1.

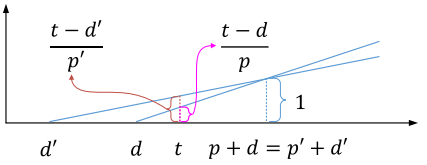

Fig. 3. Illustration of the proof of Lemma 4.

The task set τ' satisfies condition (12) because

$$\begin{aligned} d'_i + p'_i &> d_i + \frac{(i-1)\epsilon}{2n} + p_i + \frac{\epsilon}{2} \\ &\geq d_i + p_i + \frac{\epsilon}{2} \\ &> d_n + \frac{\epsilon}{2} \quad \text{(since } \tau \text{ satisfies condition (4))} \\ &> d'_n. \end{aligned}$$

Regarding condition (11), arbitrarily fix $t > 0$. When $t < d'_i$, we have

$$dbf(\tau'_i, t) = 0 \leq dbf(\tau_i, t).$$

When $t \geq d'_i$, because $p'_i > p_i, d'_i > d_i, e'_i < e_i$, we have

$$\begin{aligned} dbf(\tau'_i, t) &= \left(\left\lfloor \frac{t-d'_i}{p'_i} \right\rfloor + 1 \right) \cdot e'_i \\ &< \left(\left\lfloor \frac{t-d_i}{p_i} \right\rfloor + 1 \right) \cdot e_i = dbf(\tau_i, t). \end{aligned}$$

As a result, we always have $dbf(\tau', t) \leq dbf(\tau, t)$. Since $dbf(\tau, t) \leq t$ by condition (3), we also have $dbf(\tau', t) \leq t$, so τ' satisfies condition (11).

Altogether, τ' is a feasible solution to MP_1. □

3.2. Tightening the deadlines

Hereafter, let $d_0 = d'_0 = 0$. The objective of this subsection is to prove that the optimum value of MP_1 remains unchanged even if the deadlines are *tight*. Here "tightness" requires that $d_i = d_{i-1} + e_i$ for all $1 \leq i \leq n$, intuitively meaning that the system remains busy in the early phase. The proof consists mainly of two steps: Lemma 5 justifies tightening the first $n-1$ deadlines, while Lemma 6 enables us to handle the last deadline. This immediately leads to the equivalence between MP_1 and the following mathematical program:

$$\sup \quad \frac{dbf^*(\tau, d_n)}{d_n} \qquad (MP_2) \tag{16}$$

$$\text{subject to} \quad dbf(\tau, t) \leq t, \quad \forall t > 0, \tag{17}$$

$$d_i + p_i > d_n, \quad 1 \leq i \leq n-1, \tag{18}$$

$$d_i = e_i + d_{i-1}, \quad 1 \leq i \leq n, \tag{19}$$

$$n \in \mathbb{Z}^+, e_i, d_i, p_i \in \mathbb{Q}^+, \quad 1 \leq i \leq n. \tag{20}$$

We now present a technical lemma that will be frequently used.

Lemma 4. *Suppose $d, p, d', p' \in \mathbb{R}^+$ are such that $d + p = d' + p'$ and $d > d'$. For any real number t,*

$$\frac{t-d'}{p'} > \frac{t-d}{p}$$

if and only if $t < d + p$.

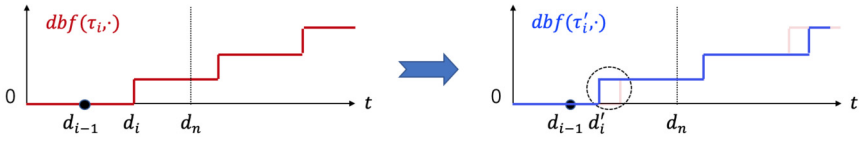

Fig. 4. Task transformation in Lemma 5.

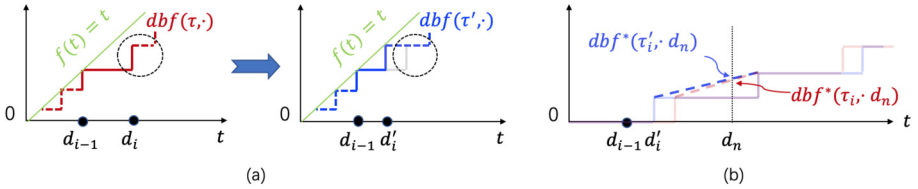

Fig. 5. (a) τ' remains feasible. (b) The objective value of τ' is at least that of τ.

Proof. The basic idea is illustrated in Fig. 3. Let $\delta = d - d' = p' - p$.

Then

$$\begin{aligned}\frac{t-d'}{p'} > \frac{t-d}{p} &\Leftrightarrow p\cdot(t-d') > p'\cdot(t-d)\\ &\Leftrightarrow p\cdot(t-d+\delta) > (p+\delta)\cdot(t-d)\\ &\Leftrightarrow p\cdot\delta > \delta\cdot(t-d)\\ &\Leftrightarrow p > t-d. \quad \square\end{aligned}$$

The following definition $M(\tau)$ will be used in Lemmas 5 and 6. For any feasible solution $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ to MP_1, let $S(\tau) = \{i : 1 \le i \le n, d_i \ne e_i + d_{i-1}\}$. Define

$$M(\tau) = \begin{cases} n - \min S(\tau) & \text{if } S(\tau) \ne \emptyset \\ -1 & \text{otherwise.}\end{cases}$$

We further prove a property of MP_1.

Lemma 5. *For any feasible solution τ to MP_1 with $M(\tau) \ge 1$, there is another feasible solution τ' to MP_1 such that $M(\tau') < M(\tau)$ and $\frac{dbf^*(\tau',d')}{d'} \ge \frac{dbf^*(\tau,d)}{d}$, where d and d' are the maximum relative deadlines in τ and τ', respectively.*

Proof. Arbitrarily fix a feasible solution τ to MP_1 with $M(\tau) \ge 1$. Suppose $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$. Let $k = n - M(\tau) = \min S(\tau) < n$.

Basically, we will modify d_k to be $e_k + d_{k-1}$, and prove that the new task set remains feasible and that the objective value does not decrease. Figs. 4 and 5 demonstrate such a transformation and relationship.

Specifically, by the definition of k, we have $d_k \ne e_k + d_{k-1}$, $d_i = e_i + d_{i-1}$ for all $i < k$, and

$$\sum_{i=1}^{k-1} e_i = d_{k-1}. \tag{21}$$

Since $d_k \ge d_i$ for any $i < k$, we have

$$\begin{aligned}\sum_{i=1}^{k} e_i &\le \sum_{i=1}^{k} dbf(\tau_i, d_k) \quad \text{(by the definition of } dbf)\\ &\le dbf(\tau, d_k) \le d_k,\end{aligned}$$

where the last inequality holds because τ satisfies condition (11). This, together with formula (21), leads to $e_k \le d_k - d_{k-1}$. By the assumption that $e_k \ne d_k - d_{k-1}$, we get

$$e_k < d_k - d_{k-1}. \tag{22}$$

Construct $\tau' = \{\tau_i' = (e_i', d_i', p_i') : 1 \le i \le n\}$, where

$$d_i' = d_i,\ p_i' = p_i,\ e_i' = e_i \text{ for any } i \ne k$$

and

$$e'_k = e_k, d'_k = d_{k-1} + e_k, p'_k = d_k + p_k - d'_k.$$

By formula (22), $d'_k < d_k$. By definition, $d'_i = e'_i + d'_{i-1}$ for all $i \le k$, so

$$M(\tau') \le n - (k+1) < n - k = M(\tau).$$

We now prove that τ' is a feasible solution to MP_1.
First of all, τ' satisfies condition (14) by definition.
Then, note that $\tau'_i = \tau_i$ for any $i \neq k$. Since τ satisfies condition (12), τ' satisfies condition (12) for $i \neq k$. Furthermore,

$$\begin{aligned} p'_k + d'_k &= d_k + p_k \quad \text{(by the definition of } p'_k\text{)} \\ &> d_n \quad \text{(because } \tau \text{ satisfies condition (12))} \\ &= d'_n \quad \text{(by the definition of } d'_n\text{)}, \end{aligned}$$

so τ' also satisfies condition (12) for $i = k$. Likewise, considering that $d'_i = d_i$ for $i \neq k$ and $d_{k-1} < d'_k < d_k \le d_{k+1}$, τ' satisfies condition (13) because so does τ.
To show that condition (11) is satisfied by τ', we arbitrarily choose $t > 0$ and proceed case by case.
Case 1: if $t < d'_k$. Then

$$\begin{aligned} dbf(\tau', t) &= \sum_{1 \le i \le n} dbf(\tau'_i, t) \\ &= \sum_{1 \le i < k} dbf(\tau'_i, t) \quad \text{(because } t < d'_j \text{ for } j \ge k\text{)} \\ &= \sum_{1 \le i < k} dbf(\tau_i, t) \quad \text{(because } \tau'_i = \tau_i \text{ for } i < k\text{)} \\ &\le dbf(\tau, t) \\ &\le t \quad \text{(because } \tau \text{ satisfies condition (11))}. \end{aligned}$$

Case 2: if $d'_k \le t < d_k$. Then

$$\begin{aligned} dbf(\tau', t) &= \sum_{1 \le i \le n} dbf(\tau'_i, t) \\ &= \sum_{1 \le i \le k} \left(\left\lfloor \frac{t - d'_i}{p'_i} \right\rfloor + 1 \right) \cdot e'_i \\ &= \sum_{1 \le i \le k} e'_i \quad \text{(because } d'_i + p'_i > d'_n = d_n \ge d_k > t \text{ for any } i\text{)} \\ &= \sum_{1 \le i \le k} e_i = d'_k \le t. \end{aligned}$$

Case 3: if $d_k \le t < d'_k + p'_k$. Then

$$\begin{aligned} dbf(\tau'_k, t) &= \left(\left\lfloor \frac{t - d'_k}{p'_k} \right\rfloor + 1 \right) \cdot e_k \\ &= e_k \quad \text{(because } d'_k < d_k \le t < d'_k + p'_k\text{)} \\ &= \left(\left\lfloor \frac{t - d_k}{p_k} \right\rfloor + 1 \right) \cdot e_k, \end{aligned}$$

where the last equality is due to $d_k \le t < d'_k + p'_k = d_k + p_k$.
For any $i \neq k$, $dbf(\tau'_i, t) = dbf(\tau_i, t)$ since $\tau'_i = \tau_i$.
As a result, $dbf(\tau', t) = dbf(\tau, t) \le t$ because τ satisfies condition (11).
Case 4: if $t \ge d'_k + p'_k$. Because

$$d'_k < d_k \text{ and } p'_k + d'_k = d_k + p_k,$$

by Lemma 4, we have

$$\frac{t-d'_k}{p'_k} \le \frac{t-d_k}{p_k}.$$

Then

$$\begin{aligned} dbf(\tau', t) &= \sum_{1\le i\le n} dbf(\tau'_i, t) \\ &= \sum_{i\ne k} dbf(\tau'_i, t) + \left(\left\lfloor \frac{t-d'_k}{p'_k} \right\rfloor + 1\right)\cdot e_k \\ &\le \sum_{i\ne k} dbf(\tau'_i, t) + \left(\left\lfloor \frac{t-d_k}{p_k} \right\rfloor + 1\right)\cdot e_k \\ &= \sum_{i\ne k} dbf(\tau_i, t) + dbf(\tau_k, t) \quad (\text{since } \tau'_i = \tau_i \text{ for } i \ne k) \\ &= dbf(\tau, t) \le t \quad (\text{since } \tau \text{ satisfies condition (11)}). \end{aligned}$$

Altogether, τ' satisfies condition (11), so it is a feasible solution to MP_1.
Finally, we show that

$$\frac{dbf^*(\tau, d_n)}{d_n} \le \frac{dbf^*(\tau', d'_n)}{d'_n}.$$

Since $k < n$, we have $d'_n = d_n$, so it suffices to show $dbf^*(\tau, d_n) \le dbf^*(\tau', d'_n)$.
By the definition of τ', for any $i \ne k$,

$$dbf^*(\tau_i, d_n) = dbf^*(\tau'_i, d'_n).$$

Furthermore, note three facts:

1. $p'_k + d'_k = d_k + p_k$.
2. $d'_k < d_k$.
3. $d_n < d_k + p_k$ because of condition (12).

By Lemma 4, these facts mean

$$\frac{d_n - d_k}{p_k} < \frac{d'_n - d'_k}{p'_k},$$

and then

$$dbf^*(\tau_k, d_n) = \left(\frac{d_n - d_k}{p_k} + 1\right)e_k \le \left(\frac{d'_n - d'_k}{p'_k} + 1\right)e_k = dbf^*(\tau'_k, d_n).$$

As a result, $dbf^*(\tau, d_n) \le dbf^*(\tau', d'_n)$. □

When $M(\tau) = 0$, i.e., $d_i = e_i + d_{i-1}$ for all $i < n$, and $d_n > e_n + d_{n-1}$, the proof above does not work. Hence, we need the following lemma, which plays a key role in proving Lemma 7.

Lemma 6. *For any feasible solution τ to MP_1 with $M(\tau) = 0$, there is a feasible solution τ' to MP_2 such that $\frac{dbf^*(\tau', d')}{d'} \ge \frac{dbf^*(\tau, d)}{d}$, where d and d' are the maximum relative deadlines in τ and τ', respectively.*

Proof. Arbitrarily fix a feasible solution τ to MP_1 with $M(\tau) = 0$. We prove the lemma by induction on $|\tau|$, the number of tasks in τ.

Base: $|\tau| = 1$. τ consists of one task (e, d, p). By straightforward calculation, $dbf(\tau, d) = dbf^*(\tau, d) = e$. Applying condition (11) with $t = d$, we have $dbf(\tau, d) \le d$, so $\frac{dbf^*(\tau, d)}{d} \le 1$. Consider the singleton task set $\tau' = \{(d, d, d)\}$. It is a solution to MP_2, and $\frac{dbf^*(\tau', d)}{d} = 1 \ge \frac{dbf^*(\tau, d)}{d}$. Hence τ' satisfies the requirement.

Hypothesis: The lemma holds when $|\tau| < n$.

Induction: Suppose $|\tau| = n$. Let $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$. Since $M(\tau) = 0$, we have $d_n \ne e_n + d_{n-1}$ and

$$d_j = e_j + d_{j-1}, \text{ for } 1 \le j \le n-1. \tag{23}$$

Like inequality (22) in the proof of Lemma 5, we also have $d_n > e_n + d_{n-1}$.

Basically, we increase e_n to be $d_n - d_{n-1}$, but this might overload the system. For adjustment, we accordingly offload the task τ_i whose job arrives earliest after time d_n, and modify the period p_n to be sufficiently large.

Formally, let $i = \arg\min_{1\le j\le n} d_j + p_j$. There are three cases.

Case 1: $i < n$ and $d_n - d_{n-1} - e_n < e_i$. Let $\theta = d_n - d_{n-1} - e_n$. Note that $\theta > 0$ since $d_n > e_n + d_{n-1}$. We construct a new task set $\tau' = \{\tau'_j = (e'_j, d'_j, p'_j) : 1 \le j \le n\}$ as follows:

- $e'_n = e_n + 2\theta, d'_n = d_n, p'_n = \left\lceil \frac{2e'_n}{e_n} \right\rceil p_n$.
- $e'_i = e_i - \theta, d'_i = d_i - \theta, p'_i = p_i + \theta$.
- For $i < j < n$, $e'_j = e_j, d'_j = d_j - \theta, p'_j = p_j + \theta$.
- For $j < i$, $\tau'_j = \tau_j$.

We now show that τ' is a feasible solution to MP_2. Since conditions (18)–(20) hold by definition, we prove that τ' satisfies condition (17) for any $t > 0$:

1. Suppose $t < d_n$. Let $0 \le k < n$ be such that $d'_k \le t < d'_{k+1}$. Then
$$\begin{aligned} dbf(\tau', t) &= \sum_{j=1}^{k} e'_j \quad \text{(since } \tau' \text{ satisfies condition (18))} \\ &= d'_k \quad \text{(since } \tau' \text{ satisfies condition (19))} \\ &\le t \quad \text{(by the definition of } k\text{)}. \end{aligned}$$
2. Suppose $d_n \le t < d_i + p_i$. We have $dbf(\tau', t) = \sum_{j=1}^{n} e'_j = d_n \le t$.
3. Consider $t \ge d_i + p_i$. We first prove that $dbf(\tau_i, t)$ decreases at by least 2θ, and then $dbf(\tau_n, t)$ increases by at most 2θ, and for any other j the value of $dbf(\tau_j, t)$ does not increase.
First,
$$\begin{aligned} dbf(\tau'_i, t) &= \left(\left\lfloor \frac{t - d'_i}{p'_i} \right\rfloor + 1 \right) \cdot e'_i \\ &\le \left(\left\lfloor \frac{t - d_i}{p_i} \right\rfloor + 1 \right) \cdot e'_i \quad \text{(by Lemma 4)} \\ &= \left(\left\lfloor \frac{t - d_i}{p_i} \right\rfloor + 1 \right) \cdot (e_i - \theta) \\ &\le dbf(\tau_i, t) - 2\theta. \end{aligned}$$
Then, when $t < p'_n + d_n$, $dbf(\tau'_n, t) = e'_n = e_n + 2\theta \le dbf(\tau_n, t) + 2\theta$. When $t \ge p'_n + d_n$, let $k \ge 1$ be the integer such that $kp'_n + d_n \le t < (k+1)p'_n + d_n$, and we have
$$\begin{aligned} dbf(\tau'_n, t) =& (k+1)e'_n \\ \le& 2ke'_n \\ \le& \left(k \left\lceil \frac{2e'_n}{e_n} \right\rceil + 1 \right) \cdot e_n \\ =& dbf(\tau_n, kp'_n + d_n) \\ \le& dbf(\tau_n, t) \\ <& dbf(\tau_n, t) + 2\theta. \end{aligned}$$
Finally, for any $j \notin \{i, n\}$, $dbf(\tau'_j, t) \le dbf(\tau_j, t)$ for two reasons. On the one hand, when $t < d_j + p_j$, $dbf(\tau'_j, t) = e'_j = e_j = dbf(\tau_j, t)$. On the other hand, when $t \ge d_j + p_j$, by Lemma 4, $\frac{t-d'_j}{p'_j} \le \frac{t-d_j}{p_j}$, which means
$$\begin{aligned} dbf(\tau'_j, t) &= \left(\left\lfloor \frac{t - d'_j}{p'_j} \right\rfloor + 1 \right) \cdot e'_j \\ &\le \left(\left\lfloor \frac{t - d_j}{p_j} \right\rfloor + 1 \right) \cdot e_j \end{aligned}$$

$$= dbf(\tau_j, t).$$

As a result, $dbf(\tau', t) = \sum_{j=1}^{n} dbf(\tau'_j, t) \le \sum_{j=1}^{n} dbf(\tau_j, t) = dbf(\tau, t)$. Hence, $dbf(\tau', t) \le t$ because τ is a feasible solution to MP_1.

Altogether, we have proven that τ' satisfies condition (17).

Next we prove $\frac{dbf^*(\tau', d'_n)}{d'_n} \ge \frac{dbf^*(\tau, d_n)}{d_n}$. Since $d'_n = d_n$, it is equivalent to show $dbf^*(\tau', d_n) \ge dbf^*(\tau, d_n)$. This follows from

- $dbf^*(\tau'_n, d_n) - dbf^*(\tau_n, d_n) = e'_n - e_n = 2\theta$;
- $dbf^*(\tau'_i, d_n) - dbf^*(\tau_i, d_n) \ge -2\theta$ because

$$\begin{aligned} dbf^*(\tau'_i, d_n) &= \left(\frac{d_n - d'_i}{p'_i} + 1\right) \cdot e'_i \\ &\ge \left(\frac{d_n - d_i}{p_i} + 1\right) \cdot (e_i - \theta) \quad \text{(by Lemma 4)} \\ &= dbf^*(\tau_i, d_n) - \left(\frac{d_n - d_i}{p_i} + 1\right)\theta \\ &\ge dbf^*(\tau_i, d_n) - 2\theta; \end{aligned}$$

- for $i < j < n$, $dbf^*(\tau'_j, d_n) \ge dbf^*(\tau_j, d_n)$ because

$$\begin{aligned} dbf^*(\tau'_j, d_n) &= \left(\frac{d_n - d'_j}{p'_j} + 1\right) \cdot e'_j \\ &\ge \left(\frac{d_n - d_j}{p_j} + 1\right) \cdot e_j \quad \text{(by Lemma 4)} \\ &= dbf^*(\tau_j, d_n); \end{aligned}$$

- for $j < i$, $dbf^*(\tau'_j, d_n) - dbf^*(\tau_j, d_n) = 0$ since $\tau'_j = \tau_j$.

Hence, the proof of case 1 is finished.

Case 2: $i < n$ and $d_n - d_{n-1} - e_n \ge e_i$. Let $\theta = e_i$. We construct a new task set $\tau' = \{\tau'_j = (e'_j, d'_j, p'_j) : 1 \le j \le n-1\}$ as follows:

- $e'_{n-1} = e_n + 2\theta, \quad d'_{n-1} = d_n, \quad p'_{n-1} = \left\lceil \frac{2e'_{n-1}}{e_n} \right\rceil p_n$.
- For $i \le j \le (n-2)$, $e'_j = e_{j+1}, \quad d'_j = d_{j+1} - \theta, \quad p'_j = p_{j+1} + \theta$.
- For $j < i$, $\tau'_j = \tau_j$.

Next we prove that τ' is a feasible solution to MP_1. Since conditions (12)–(14) hold by definition, we prove that condition (11) holds for any $t > 0$:

1. Suppose $t < d_n$. Let $0 \le k < (n-1)$ be such that $d'_k \le t < d'_{k+1}$. If $k < i$, then we have $dbf(\tau', t) = \sum_{j=1}^{k} e_j = d_k = d'_k \le t$. Otherwise

$$\begin{aligned} dbf(\tau', t) &= \sum_{j=1}^{k} e'_j \quad \text{(since } \tau' \text{ satisfies condition (12))} \\ &= \sum_{j=1}^{i-1} e_j + \sum_{j=i+1}^{k+1} e_j = d_{k+1} - e_i \\ &= d_{k+1} - \theta = d'_k \quad \text{(by the definition of } \tau') \\ &\le t \quad \text{(by the definition of } k). \end{aligned}$$

2. Suppose $d_n \le t < d_i + p_i$. We have $dbf(\tau', t) = \sum_{j=1}^{n-1} e'_j \le d_n \le t$.
3. Consider $t \ge d_i + p_i$. First of all, we have $dbf(\tau_i, t) \ge 2e_i = 2\theta$.
   Then, when $t < p'_{n-1} + d_n$, $dbf(\tau'_{n-1}, t) = e'_{n-1} = e_n + 2\theta \le dbf(\tau_n, t) + 2\theta$. When $t \ge p'_{n-1} + d_n$, let $k \ge 1$ be the integer such that $kp'_{n-1} + d_n \le t < (k+1)p'_{n-1} + d_n$, and we have

$$\begin{aligned} dbf(\tau'_{n-1}, t) =& (k+1)e'_{n-1} \\ \leq& 2ke'_{n-1} \\ \leq& \left(k\left\lceil \frac{2e'_{n-1}}{e_n} \right\rceil + 1\right) \cdot e_n \\ =& \left(k\frac{p'_{n-1}}{p_n} + 1\right) \cdot e_n \\ =& dbf(\tau_n, kp'_{n-1} + d_n) \\ \leq& dbf(\tau_n, t) \\ <& dbf(\tau_n, t) + 2\theta. \end{aligned}$$

In addition, for any $j < i$, $dbf(\tau'_j, t) \leq dbf(\tau_j, t)$ by definition.

Finally, for any $i \geq j \leq (n-2)$, $dbf(\tau'_j, t) \leq dbf(\tau_{j+1}, t)$ for two reasons. On the one hand, when $t < d_{j+1} + p_{j+1}$, $dbf(\tau'_j, t) = e'_j = e_{j+1} = dbf(\tau_{j+1}, t)$. On the other hand, when $t \geq d_{j+1} + p_{j+1}$, by Lemma 4, $\frac{t-d'_j}{p'_j} \leq \frac{t-d_{j+1}}{p_{j+1}}$, which means

$$\begin{aligned} dbf(\tau'_j, t) =& \left(\left\lfloor \frac{t-d'_j}{p'_j} \right\rfloor + 1\right) \cdot e'_j \\ \leq& \left(\left\lfloor \frac{t-d_{j+1}}{p_{j+1}} \right\rfloor + 1\right) \cdot e_{j+1} \\ =& dbf(\tau_{j+1}, t). \end{aligned}$$

As a result, $dbf(\tau', t) = \sum_{j=1}^{n-1} dbf(\tau'_j, t) \leq \sum_{j=1}^{n} dbf(\tau_j, t) = dbf(\tau, t)$. Hence, $dbf(\tau', t) \leq t$ because τ is a feasible solution to MP_1.

Altogether, we have proven that τ' satisfies condition (11).

We now prove $\frac{dbf^*(\tau', d'_{n-1})}{d'_{n-1}} \geq \frac{dbf^*(\tau, d_n)}{d_n}$. Since $d'_{n-1} = d_n$, it is equivalent to show $dbf^*(\tau', d_n) \geq dbf^*(\tau, d_n)$. This follows from

- $dbf^*(\tau'_{n-1}, d_n) - dbf^*(\tau_n, d_n) = e'_{n-1} - e_n = 2\theta$;
- $dbf^*(\tau_i, d_n) = \left(\frac{d_n - d_i}{p_i} + 1\right) \cdot e_i < 2e_i = 2\theta$;
- for $i \leq j \leq n-2$, $dbf^*(\tau'_j, d_n) \geq dbf^*(\tau_{j+1}, d_n)$ because

$$\begin{aligned} dbf^*(\tau'_j, d_n) =& \left(\frac{d_n - d'_j}{p'_j} + 1\right) \cdot e'_j \\ \geq& \left(\frac{d_n - d_{j+1}}{p_{j+1}} + 1\right) \cdot e_{j+1} \quad \text{(by Lemma 4)} \\ =& dbf^*(\tau_{j+1}, d_n); \end{aligned}$$

- for $j < i$, $dbf^*(\tau'_j, d_n) - dbf^*(\tau_j, d_n) = 0$ since $\tau'_j = \tau_j$.

We proceed in two subcases.

Case 2.1: $d_n - d_{n-1} - e_n > e_i$. Then τ' is a feasible solution to MP_1 with $M(\tau') = 0$. The lemma follows from the induction hypothesis.

Case 2.2: $d_n - d_{n-1} - e_n = e_i$. By the definition of τ', one can see that condition (19) also holds, so τ' is a desired feasible solution to MP_2. The lemma thus holds.

Hence, the proof of case 2 is finished.

Case 3: $i = n$. Choose k such that $d_n + kp_n > d_1 + p_1$. Define tasks $\tau'_j = \tau_j$ for $1 \leq j < n$ and $\tau'_n = (e_n, d_n, kp_n)$. Let $\tau' = \{\tau'_j : 1 \leq j \leq n\}$. We observe three facts:

1. By the construction, $\frac{dbf^*(\tau', d_n)}{d_n} = \frac{dbf^*(\tau, d_n)}{d_n}$.
2. For any $t > 0$, $dbf(\tau'_n, t) \leq dbf(\tau_n, t)$ and $dbf(\tau'_j, t) = dbf(\tau_j, t)$ for any $1 \leq j < n$, meaning that $dbf(\tau', t) \leq dbf(\tau, t) \leq t$. Hence, τ' is a feasible solution to MP_1.

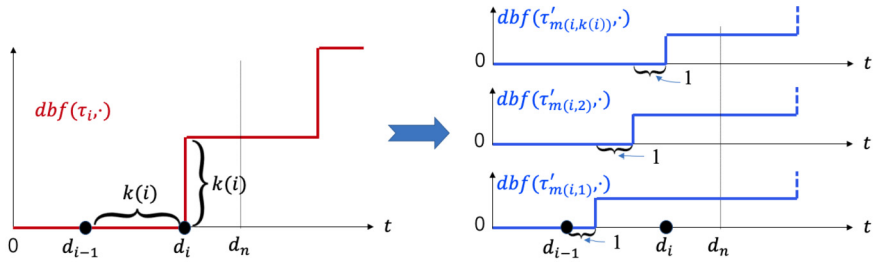

Fig. 6. Splitting each τ_i into $\tau'(i) = \{\tau'_{m(i,j)} : 1 \le j \le k(i)\}$.

3. $M(\tau') = 0$ and $n > \arg\min_{1\le j\le n} d'_j + p'_j$, where d'_j and p'_j are the relative deadline and the period of task τ'_j, respectively. The proof is thus reduced to case 1 or case 2.

Altogether, we have finished the proof. □

Applying Lemmas 5 and 6, we immediately get the following result.

Lemma 7. *MP_1 and MP_2 have the same optimum value.*

3.3. Unifying execution times

In this subsection, a further constraint is imposed on MP_2; namely, all the tasks have identical execution times (into MP_3). We show that this modification does not change the optimum value.

$$\sup \quad \frac{dbf^*(\tau, d_n)}{d_n} \qquad (MP_3) \tag{24}$$

$$\text{subject to} \quad dbf(\tau, t) \le t, \quad \forall t > 0, \tag{25}$$

$$d_i + p_i > d_n, \quad 1 \le i \le n-1, \tag{26}$$

$$d_i = e_i + d_{i-1}, \quad 1 \le i \le n, \tag{27}$$

$$e_i = d_n/n, \quad 1 \le i \le n, \tag{28}$$

$$n \in \mathbb{Z}^+, e_i, d_i, p_i \in \mathbb{Q}^+, \quad 1 \le i \le n. \tag{29}$$

Lemma 8. *MP_2 and MP_3 have the same optimum value.*

The basic idea of the proof is that for any feasible solution to MP_2, we will construct a feasible solution to MP_3 whose objective value is no smaller. This leads to the lemma since the feasible domain of MP_3 is included in that of MP_2 and the two mathematical programs have the same objective function.

Roughly speaking, the construction is to split each task into a set of *smaller* subtasks with identical execution times, as demonstrated in Fig. 6. The fact that the splitting keeps the feasibility and does not reduce the dbf^* value is intuitively shown in Fig. 7.

Proof. Let $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ be an arbitrary feasible solution to MP_2. Because of condition (20), we can choose $\delta \in \mathbb{Q}^+$ such that

$$k(i) \triangleq \frac{e_i}{\delta}$$

is an integer for any $1 \le i \le n$. Let $n' = \sum_{i=1}^{n} k(i)$.

For any $1 \le l \le n'$, define task $\tau'_l = (e'_l, d'_l, p'_l)$ as below, where $1 \le i \le n$ and $1 \le j \le k(i)$ are such that $l = m(i,j) \triangleq j + \sum_{1\le h<i} k(h)$:

$$e'_l = \delta,$$

$$d'_l = d_{i-1} + \frac{j}{k(i)}(d_i - d_{i-1}) = d_{i-1} + j\delta,$$

$$p'_l = p_i + d_i - d'_l.$$

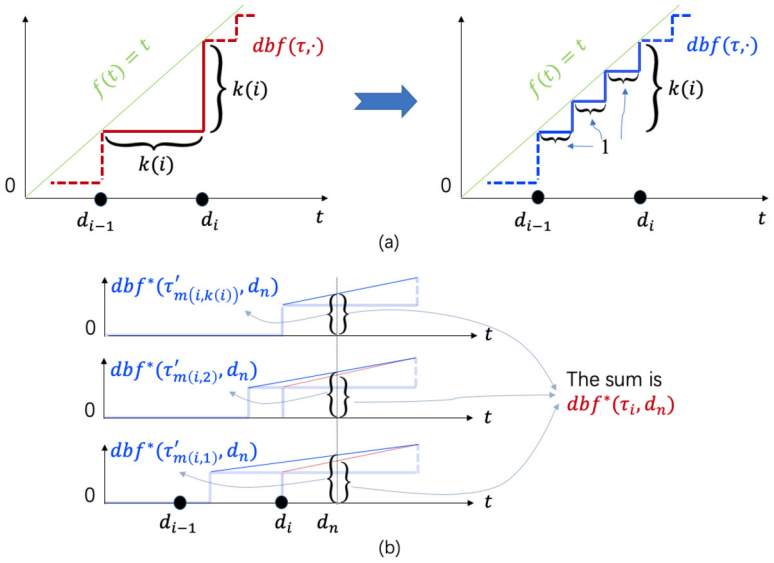

Fig. 7. (a) The splitting keeps the feasibility. (b) The dbf^* value is not reduced.

Let $\tau'(i) = \{\tau'_{m(i,j)} : 1 \le j \le k(i)\}$ for any $1 \le i \le n$, and $\tau' = \cup_{i=1}^{n} \tau'(i)$. Let $d'_0 = 0$. Next we prove that τ' is a feasible solution to MP_3.

Since τ' satisfies conditions (26)–(29) by definition, we now investigate condition (25) by arbitrarily fixing $t > 0$ and proceeding case by case.

Case 1: $t < d'_{n'}$. Let integer $h \ge 0$ be such that $d'_h \le t < d'_{h+1}$. Then

$$\begin{aligned} dbf(\tau', t) &= \sum_{1 \le r \le n'} dbf(\tau'_r, t) \\ &= \sum_{1 \le r \le h} dbf(\tau'_r, t) \quad (\text{because } t < d'_{h+1}) \\ &= \sum_{1 \le r \le h} \left(\left\lfloor \frac{t - d'_r}{p'_r} \right\rfloor + 1 \right) \cdot e'_r \\ &= \sum_{1 \le r \le h} e'_r = d'_h \le t, \end{aligned}$$

where the fourth equality holds because of the inequality $p'_r > t - d'_r$, which in turn follows from three facts:

1. For any $1 \le i \le n$ and $1 \le j \le k(i)$, we have

   $$p'_{m(i,j)} = p_i + d_i - d'_{m(i,j)} \text{ by definition.}$$

2. From condition (18), $p_i + d_i > d_n$ holds for all i, $1 \le i \le n$.
3. $d_n = d'_{n'} > t$.

Case 2: $t \ge d'_{n'}$. It suffices to prove that for any $1 \le i \le n$,

$$dbf(\tau'(i), t) \le dbf(\tau_i, t).$$

Suppose $t < d_i + p_i$. We observe that

$$\begin{aligned} dbf(\tau'(i), t) &= \sum_{j=1}^{k(i)} dbf(\tau'_{m(i,j)}, t) \\ &= \sum_{j=1}^{k(i)} \left(\left\lfloor \frac{t - d'_{m(i,j)}}{p'_{m(i,j)}} \right\rfloor + 1 \right) \delta \end{aligned}$$

$$
\begin{aligned}
&= k(i)\delta \quad (\text{because } t < d_i + p_i = d'_{m(i,j)} + p'_{m(i,j)}) \\
&= e_i \quad (\text{by the definition of } k(i)) \\
&= dbf(\tau_i, t) \quad (\text{because } d_i \le t < d_i + p_i).
\end{aligned}
$$

Then consider $t \ge d_i + p_i$. For any $1 \le j \le k(i)$, since $d_i \ge d'_{m(i,j)}$ and $d_i + p_i = d'_{m(i,j)} + p'_{m(i,j)}$, Lemma 4 implies

$$
\frac{t - d'_{m(i,j)}}{p'_{m(i,j)}} \le \frac{t - d_i}{p_i},
$$

which further leads to

$$
\begin{aligned}
dbf(\tau'(i), t) &= \sum_{j=1}^{k(i)} \left(\left\lfloor \frac{t - d'_{m(i,j)}}{p'_{m(i,j)}} \right\rfloor + 1 \right) \delta \\
&\le \sum_{j=1}^{k(i)} \left(\left\lfloor \frac{t - d_i}{p_i} \right\rfloor + 1 \right) \delta \\
&= \left(\left\lfloor \frac{t - d_i}{p_i} \right\rfloor + 1 \right) e_i \\
&= dbf(\tau_i, t).
\end{aligned}
$$

Altogether, condition (25) is satisfied in both cases, so τ' is a feasible solution to MP_3.

The rest of the proof is to show that

$$
dbf^*(\tau', d'_{n'}) \ge dbf^*(\tau, d_n).
$$

Note that for any $1 \le i \le n, 1 \le j \le k(i)$,

$$
d'_{n'} = d_n < p_i + d_i = d'_{m(i,j)} + p'_{m(i,j)} \text{ and } d'_{m(i,j)} \le d_i.
$$

Lemma 4 implies that

$$
\frac{d'_{n'} - d'_{m(i,j)}}{p'_{m(i,j)}} \ge \frac{d_n - d_i}{p_i}.
$$

Then for any $1 \le i \le n$, we have

$$
\begin{aligned}
dbf^*(\tau'(i), d'_{n'}) &= \sum_{j=1}^{k(i)} \left(\frac{d'_{n'} - d'_{m(i,j)}}{p'_{m(i,j)}} + 1 \right) \delta \\
&\ge \left(\frac{d_n - d_i}{p_i} + 1 \right) e_i \\
&= dbf^*(\tau_i, d_n).
\end{aligned}
$$

Therefore, $dbf^*(\tau', d'_{n'}) \ge dbf^*(\tau, d_n)$. □

3.4. Aligning the periods

It is still difficult to estimate the optimum value of MP_3, partly because condition (25) is hard to handle. Thus, instead of conditions (25) and (26), we require that the task set be aligned, as defined next.

Definition 1. Given a task set $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$, a permutation π over $\{1, 2, \ldots, n\}$ is called an *aligning permutation* of τ if

$$
d_{\pi(i)} + p_{\pi(i)} = d_n + d_i
$$

for any $1 \le i \le n$. τ is said to be *aligned* if it has an aligning permutation.

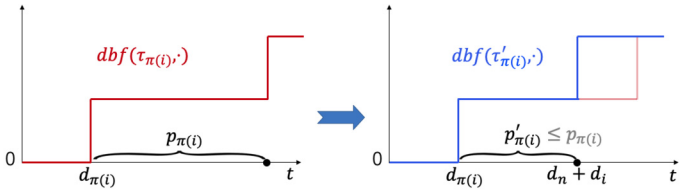

Fig. 8. Illustration of the proof of Lemma 9.

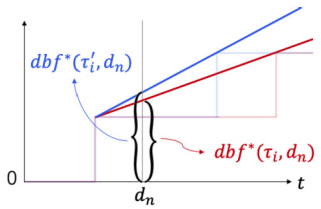

Fig. 9. The transformation in Lemma 9 does not reduce the objective value.

Remark 1. We will consider aligned task sets in the context of conditions (32) and (33) as in the following MP_4. Then being aligned means that every d_1 time during period $[0, 2d_n]$, there is a job (the first or second job released by some task) that reaches its deadline. Since any job needs execution time $\frac{d_n}{n}$, the system has to execute the jobs one after another, having no idle time during $[0, 2d_n]$ at all. Hence, neither the periods nor the deadlines of the tasks can be further shrunk to keep the task set schedulable. Intuitively, aligned task sets make the system as busy as possible during $[0, 2d_n]$, so they might lead to an upper bound of ρ.

Being aligned implies condition (26), and in some sense "relaxes" condition (25) for ease of analysis. Hence, we replace these conditions in MP_3 with "aligned," and show that the optimum value of MP_3 does not decrease after the modification. Specifically, we define a new mathematical program:

$$\sup \quad \frac{dbf^*(\tau, d_n)}{d_n} \qquad (MP_4) \tag{30}$$

$$\text{subject to} \quad \tau \text{ is aligned}, \tag{31}$$

$$d_i = e_i + d_{i-1}, \quad 1 \le i \le n, \tag{32}$$

$$e_i = d_n/n, \quad 1 \le i \le n, \tag{33}$$

$$n \in \mathbb{Z}^+, e_i, d_i, p_i \in \mathbb{Q}^+, \quad 1 \le i \le n. \tag{34}$$

Lemma 9. *The optimum value of MP_3 is not more than that of MP_4.*

The basic idea of the proof is that given any feasible solution to MP_3, we sort the tasks increasingly according to their second deadlines; namely, $p_i + d_i$. Then we adjust the periods of the tasks so that for any ith task (order in the sorting), its second deadline is $d_n + d_i$. This transformation trivially guarantees alignment.

Proof. Arbitrarily choose a feasible solution $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ to MP_3. Let π be a permutation over $\{1, 2, \ldots, n\}$ such that

$$d_{\pi(1)} + p_{\pi(1)} \le d_{\pi(2)} + p_{\pi(2)} \le \cdots \le d_{\pi(n)} + p_{\pi(n)}. \tag{35}$$

For any $1 \le i \le n$, construct a task $\tau'_{\pi(i)} = (e'_{\pi(i)}, d'_{\pi(i)}, p'_{\pi(i)})$, where

$$e'_{\pi(i)} = e_{\pi(i)}, d'_{\pi(i)} = d_{\pi(i)}, p'_{\pi(i)} = d_n + d_i - d'_{\pi(i)}.$$

Let $\tau' = \{\tau'_i : 1 \le i \le n\}$. The construction is demonstrated in Fig. 8.

We will show that τ' is a feasible solution to MP_4. Since conditions (32)–(34) are satisfied by definition, we need only to prove condition (31). Because $p'_{\pi(i)} + d'_{\pi(i)} = d_n + d_i = d'_n + d'_i$ for any $1 \le i \le n$, π is an aligning permutation of τ', and condition (31) is satisfied.

We now prove $dbf^*(\tau', d'_n) \ge dbf^*(\tau, d_n)$, as illustrated in Fig. 9. We first derive an inequality as a tool.

For any $1 \le i \le n$, let $j = \pi^{-1}(i)$, i.e., $\pi(j) = i$, and we have

$$\begin{aligned} d_i + p_i \ge & dbf(\tau, d_i + p_i) \quad \text{(since } \tau \text{ satisfies condition (25))} \\ = & \sum_{1 \le l \le j} dbf(\tau_{\pi(l)}, d_i + p_i) + \sum_{j < l \le n} dbf(\tau_{\pi(l)}, d_i + p_i) \\ \ge & \sum_{1 \le l \le j} 2e_{\pi(l)} + \sum_{j < l \le n} e_{\pi(l)} \\ = & \frac{2jd_n}{n} + \frac{(n-j)d_n}{n} \\ = & d_n + d_j \quad \text{(because of conditions (27) and (28))}, \end{aligned}$$

where the second inequality is because

$$d_i + p_i = d_{\pi(j)} + p_{\pi(j)} \ge d_{\pi(l)} + p_{\pi(l)} \text{ for any } l \le j$$

and $d_i + p_i > d_n \ge d_{\pi(l)}$ for any $l > j$. Hence, by the definition of τ', we have

$$d_i + p_i \ge d_n + d_j = d_n + d_{\pi^{-1}(i)} = d'_i + p'_i. \tag{36}$$

For any $1 \le i \le n$, by (36) and $d_i = d'_i$, we have $p'_i \le p_i$. This, together with $e'_i = e_i, d'_i = d_i$ for any $1 \le i \le n$, implies $dbf^*(\tau', d'_n) \ge dbf^*(\tau, d_n)$. As a result,

$$\frac{dbf^*(\tau, d_n)}{d_n} \le \frac{dbf^*(\tau', d'_n)}{d'_n}.$$

The lemma thus holds. □

We present a technical lemma before moving on.

Lemma 10. *For any $x_1, x_2, \ldots, x_n \in \mathbb{R}^+$ such that*

$$\sum_{i=1}^{n} x_i = n^2,$$

we have

$$\sum_{i=1}^{n} \frac{i}{x_i} \ge \frac{4n}{9}.$$

Proof. By Cauchy's inequality,

$$\left(\sum_{i=1}^{n} \frac{i}{x_i}\right)\left(\sum_{i=1}^{n} x_i\right) \ge \left(\sum_{i=1}^{n} \sqrt{i}\right)^2.$$

Note that

$$\begin{aligned} \sum_{i=1}^{n} \sqrt{i} \ge & \sum_{i=1}^{n} \int_{i-1}^{i} \sqrt{x} dx \quad \text{(by the monotonicity of } \sqrt{x}) \\ = & \int_{0}^{n} \sqrt{x} dx \\ = & \frac{2}{3} n^{\frac{3}{2}}. \end{aligned}$$

Therefore,

$$\sum_{i=1}^{n} \frac{i}{x_i} \ge \frac{\frac{4}{9}n^3}{n^2} = \frac{4n}{9}.$$

Hence, we have proved the lemma. □

Lemma 11. *The optimum value of MP_4 is at most $\frac{14}{9}$.*

Proof. Arbitrarily choose a feasible solution $\tau = \{\tau_i = (e_i, d_i, p_i) : 1 \le i \le n\}$ to MP_4. Let $\delta = \frac{d_n}{n}$. By conditions (32) and (33),

$$e_i = \delta \text{ and } d_i = i\delta$$

for any $1 \le i \le n$.

Let π be an aligning permutation of τ. Then we have

$$\sum_{i=1}^{n} p_{\pi(i)} = \sum_{i=1}^{n} (d_n + d_i - d_{\pi(i)}) = nd_n = n^2\delta,$$

which implies $\sum_{i=1}^{n} \frac{p_{\pi(i)}}{\delta} = n^2$. By Lemma 10,

$$\sum_{i=1}^{n} \frac{i\delta}{p_{\pi(i)}} \ge \frac{4n}{9}.$$

Hence,

$$\begin{aligned}
\sum_{j=1}^{n} \frac{d_j + p_j - d_n}{p_j} &= \sum_{i=1}^{n} \frac{d_{\pi(i)} + p_{\pi(i)} - d_n}{p_{\pi(i)}} \\
&= \sum_{i=1}^{n} \frac{d_i}{p_{\pi(i)}} \quad \text{(since } \tau \text{ is aligned)} \\
&= \sum_{i=1}^{n} \frac{i\delta}{p_{\pi(i)}} \ge \frac{4n}{9}.
\end{aligned}$$

As a result,

$$\begin{aligned}
dbf^*(\tau, d_n) &= \sum_{j=1}^{n} dbf^*(\tau_j, d_n) \\
&= \sum_{j=1}^{n} \left(2 - \frac{p_j + d_j - d_n}{p_j}\right) e_j \\
&= \sum_{j=1}^{n} \left(2 - \frac{p_j + d_j - d_n}{p_j}\right) \delta \\
&= 2n\delta - \delta \sum_{j=1}^{n} \frac{p_j + d_j - d_n}{p_j} \\
&\le 2n\delta - \frac{4n}{9}\delta = \frac{14}{9} d_n.
\end{aligned}$$

The lemma holds. □

3.5. Upper-bounding the relaxation factor

We are ready to present one of the main results of this article, which claims that the relaxation factor is at most 1.5556.

Theorem 1. *The relaxation factor ρ is at most $\frac{14}{9}$.*

Proof. It follows from Lemmas 3, 7, 8, 9, and 11. □

4. Partitioned scheduling on multiprocessors

This section is devoted to partitioning sporadic tasks on multiprocessors, where the tasks are assumed to have constrained deadlines. Note that although Theorem 1 holds for arbitrary deadlines, the extension to multiprocessors applies only for the constrained-deadline case. We focus on the algorithm of deadline-monotonic partitioned EDF; namely, the algorithm PARTITION in [16].

Basically, the algorithm PARTITION assigns tasks sequentially in nondecreasing order of relative deadlines to processors that are numbered by distinct integers. Suppose the $(i-1)$th task has just been assigned. Let $\tau(k)$ be the set of tasks that have been assigned to processor k, for any k. Then the ith task is assigned to the least-numbered processor k that can safely serve the task, i.e., $e_i + dbf^*(\tau(k), d_i) \leq d_i$.

Remember that we have upper-bounded the relaxation factor ρ. The following lemma bridges ρ and the speedup factor of PARTITION.

Lemma 12 *([16,1]). The speedup factor of the algorithm PARTITION for constrained-deadline tasks is* $1 + \rho - 1/m$*, where* m *is the number of processors.*

We now present the other main result of this article.

Theorem 2. *The speedup factor of the algorithm PARTITION for constrained-deadline tasks is at most* $2.5556 - 1/m$.

Proof. The theorem immediately follows from Theorem 1 and Lemma 12. □

5. Conclusion and future work

In this article, we improved the upper bound of the speedup factor of (polynomial-time) partitioned EDF from $2.6322 - 1/m$ to $2.5556 - 1/m$ for constrained-deadline sporadic tasks on m identical processors, narrowing the gap between the upper and lower bounds from 0.1322 to 0.0556. This is an immediate corollary of our improvement of the upper bound of the relaxation factor from 1.6322 to 1.5556, which holds for both the constrained-deadline scenario and the arbitrary-deadline scenario.

Technically, our improvements are rooted in a novel discretization that transforms the tasks into regular form. The discretization essentially restricts attention to the tasks with fixed execution times and deadlines. Only the period parameter remains flexible to some extent—ranging over the set $\{1, 2, \ldots, 2n\}$, where n is the number of tasks to be scheduled. With such transformation, the estimation of the relaxation factor is reduced to a much simpler optimization problem. However, we have not yet proved that the last-step transformation (for periods) is lossless. This means that the discretization might increase the relaxation factor. The good news is that the incurred loss, if not zero at all, is guaranteed to be no more than 0.0556.

Regarding future directions, we conjecture that a upper bound of 1.5 for the relaxation factor can be derived, thus closing the gap between the upper and lower bounds. If this is the case, the speedup factor of partitioned EDF becomes fully determined, at least in the case of constrained deadlines.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors thank Prof. Sanjoy Baruah from Washington University at St. Louis and Prof. Yungang Bao from the Institute of Computing Technology, Chinese Academy of Sciences, for fruitful discussions. This work was partially supported by the National Key Research and Development Program of China (Grant No. 2016YFB1000201), the Key-Area Research and Development Program of Guangdong Province (2020B010164003), the National Natural Science Foundation of China (11971091, 62072433, and 62090020), Liaoning Natural Science Foundation (2019-MS-062), the Youth Innovation Promotion Association of the Chinese Academy of Sciences (2013073), the Strategic Priority Research Program of the Chinese Academy of Sciences (XDC05030200), and the US National Science Foundation (CNS-1850851).

References

[1] J.-J. Chen, S. Chakraborty, Resource augmentation bounds for approximate demand bound functions, in: 2011 IEEE 32nd Real-Time Systems Symposium (RTSS), IEEE, 2011, pp. 272–281.

[2] F. Eisenbrand, T. Rothvoß, EDF-schedulability of synchronous periodic task systems is conp-hard, in: Proceedings of the Twenty-First Annual ACM-SIAM Symposium on Discrete Algorithms, SIAM, 2010, pp. 1029–1034.

[3] A.K.-L. Mok, Fundamental design problems of distributed systems for the hard-real-time environment, Ph.D. thesis, Massachusetts Institute of Technology, 1983.
[4] J.-J. Chen, G. von der Brüggen, W.-H. Huang, R.I. Davis, On the pitfalls of resource augmentation factors and utilization bounds in real-time scheduling, in: 29th Euromicro Conference on Real-Time Systems (ECRTS 2017), Leibniz International Proceedings in Informatics (LIPIcs), 2017, pp. 9:1–9:25.
[5] Z. Guo, Regarding the optimality of speedup bounds of mixed-criticality schedulability tests, in: Mixed Criticality on Multicore/Manycore Platforms (Dagstuhl Seminar Reports) 17131, 2017.
[6] K. Agrawal, S. Baruah, Intractability issues in mixed-criticality scheduling, in: Proceedings of the 30th Euromicro Conference on Real-Time Systems (ECRTS'18), Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.
[7] B. Kalyanasundaram, K. Pruhs, Speed is as powerful as clairvoyance, J. ACM 47 (2000) 617–643.
[8] C.A. Phillips, C. Stein, E. Torng, J. Wein, Optimal time-critical scheduling via resource augmentation, Algorithmica 32 (2002) 163–200.
[9] K. Albers, F. Slomka, An event stream driven approximation for the analysis of real-time systems, in: Proceedings of the 16th Euromicro Conference on Real-Time Systems, 2004, ECRTS 2004, IEEE, 2004, pp. 187–195.
[10] R.R. Devillers, J. Goossens, Liu and Layland's schedulability test revisited, Inf. Process. Lett. 73 (2000) 157–161.
[11] R.I. Davis, A. Thekkilakattil, O. Gettings, R. Dobrin, S. Punnekkat, J. Chen, Exact speedup factors and sub-optimality for non-preemptive scheduling, Real-Time Syst. 54 (2018) 208–246.
[12] E. Bini, G.C. Buttazzo, Measuring the performance of schedulability tests, Real-Time Syst. 30 (2005) 129–154.
[13] E. Bini, The quadratic utilization upper bound for arbitrary deadline real-time tasks, IEEE Trans. Comput. 64 (2015) 593–599.
[14] J. Theis, G. Fohler, Transformation of sporadic tasks for off-line scheduling with utilization and response time trade-offs, in: Proceedings of the 19th International Conference on Real-Time and Network Systems, RTNS '11, Nantes, France, September 29–30, 2011, 2011, pp. 119–128.
[15] J. Chen, Partitioned multiprocessor fixed-priority scheduling of sporadic real-time tasks, in: 2016 28th Euromicro Conference on Real-Time Systems (ECRTS), 2016, pp. 251–261.
[16] S. Baruah, N. Fisher, The partitioned multiprocessor scheduling of sporadic task systems, in: 26th IEEE International Real-Time Systems Symposium, 2005, RTSS 2005, IEEE, 2005, pp. 321–329.
[17] S.K. Baruah, A.K. Mok, L.E. Rosier, Preemptively scheduling hard-real-time sporadic tasks on one processor, in: Proceedings of the 11th Real-Time Systems Symposium, 1990, IEEE, 1990, pp. 182–190.