Fixed-Priority Schedulability of Sporadic Tasks on Uniprocessors is NP-hard

Pontus Ekberg and Wang Yi
Uppsala University, Sweden
Email: {pontus.ekberg | yi}@it.uu.se

***Abstract*—The computational complexity of deciding whether a set of sporadic or synchronous periodic tasks is FP-schedulable on a preemptive uniprocessor has been a longstanding open problem in real-time scheduling theory. We show that the FP-schedulability problem is (weakly) NP-hard, even when restricted to either (i) task sets with implicit deadlines and rate-monotonic priority ordering, or (ii) task sets with constrained deadlines, deadline-monotonic priority ordering and utilization bounded by any constant c, such that $0 < c < 1$.**

I. Introduction

The Fixed-Priority (FP) scheduling algorithm is widely used in real-time systems. We study the computational complexity of the FP-schedulability problem. We consider tasks that are either sporadic or synchronous periodic, and a computer platform consisting of a single preemptive processor. In settling this we close the first problem mentioned by Baruah and Pruhs [1] in their list of "*some of the most important open algorithmic problems in real-time scheduling*".

In this introduction we first describe models and definitions, followed by some technical preliminaries and, last, related work and our contributions.

A. Model and Definitions

Let a sporadic task τ be represented as a triple (e, d, p) of positive integers, where e represents the task's worst-case execution time, d its relative deadline and p its period (or minimum separation time), respectively. A task $\tau = (e, d, p)$ generates an unbounded sequence of jobs, where each job has an execution time of up to e time units and an absolute deadline exactly d time units after its release time. The release times of two consecutive jobs from the task τ are separated by at least p time units. We say that a job is ready when it has been released but has not yet executed to completion.

A task set $\mathcal{T}$ is a (multi-)set of tasks and generates an interleaving of the job sequences generated by each of the tasks in $\mathcal{T}$. We say that $\mathcal{T}$ has

- *implicit deadlines* if $d = p$ for all $(e, d, p) \in \mathcal{T}$,
- *constrained deadlines* if $d \leqslant p$ for all $(e, d, p) \in \mathcal{T}$, and
- *arbitrary deadlines* if no restrictions are placed on the relation between d and p.

We consider scheduling of sporadic tasks on a preemptive uniprocessor. On such machines, the Fixed-Priority (FP) scheduling algorithm takes a fixed (total) priority ordering of the tasks in the task set and then executes jobs strictly according to this priority ordering. In other words, the FP scheduler attaches to each job the priority of the task that generated it, and at any time point chooses among the ready jobs to execute the job with the highest priority, preempting any lower-priority job if needed. If several jobs from the same task are active, these are prioritized in FIFO order.

We establish the results of this paper by relating schedulability with the FP algorithm to schedulability with the Earliest Deadline First (EDF) algorithm, for which hardness results are known. The EDF scheduling algorithm executes jobs in the order of their absolute deadlines. That is, it chooses among the ready jobs to execute the job with the earliest absolute deadline (ties broken arbitrarily), preempting a job with a later deadline if needed.

Both the FP and EDF scheduling algorithms have been extensively studied in the literature. The FP scheduler is the default scheduler in many real-time operating systems, due in part to its easy implementation. While the EDF scheduler is optimal on preemptive uniprocessors [2], it is less common in real-time operating systems.

We say that a task set $\mathcal{T}$ is *FP-schedulable* with a given priority ordering if and only if all jobs generated by $\mathcal{T}$ will always complete execution by their deadlines when using the FP scheduler with that priority ordering on a preemptive uniprocessor. The *FP-schedulability* problem is to determine whether a given task set is FP-schedulable with a given priority ordering. *EDF-schedulable* and *EDF-schedulability* are defined similarly.

The synchronous periodic task is a common alternative workload model to the sporadic task. A synchronous periodic task is also defined by a triple (e, d, p) of positive integers, with the only differences that p instead denotes the exact separation between consecutive job releases, and the first job of each such task are released synchronously at the same time point. It is known that FP- and EDF-schedulability is unaffected by the choice of these two task models.

Theorem I.1 (Lehoczky [3], Baruah et al. [4])**.** *Let $\mathcal{T}$ be a set of sporadic tasks, and let $\mathcal{T}'$ be a set of synchronous periodic tasks with the same parameters as the tasks in $\mathcal{T}$. Then, on a preemptive uniprocessor*

(i) $\mathcal{T}$ is FP-schedulable with a given priority ordering if and only if $\mathcal{T}'$ is FP-schedulable with the same priority ordering, and
(ii) $\mathcal{T}$ is EDF-schedulable if and only if $\mathcal{T}'$ is EDF-schedulable.

The equivalence of FP- and EDF-schedulability for the different types of tasks in this setting means that our results apply to both of them. In the following, when we simply write "task", we interchangeably mean a task that is either sporadic or synchronous periodic.

B. Preliminaries

Here we briefly review some results from real-time scheduling theory that are used in this paper.

In FP scheduling, the priority ordering that assigns higher priority to tasks with shorter relative deadlines (ties broken arbitrarily) is known as the Deadline-Monotonic (DM) priority ordering. DM is an optimal priority ordering for task sets with constrained deadlines.

Theorem I.2 (Leung and Whitehead [5]). *If a task set with constrained deadlines is FP-schedulable on a preemptive uniprocessor with any priority ordering, then it is also FP-schedulable with the DM priority ordering.*

For historical reasons, the DM priority ordering is called the Rate-Monotonic (RM) priority ordering for task sets with implicit deadlines. The seminal work by Liu and Layland [6] established a sufficient condition for FP-schedulability with the RM priority ordering in the form of a utilization bound.

Theorem I.3 (Liu and Layland [6]). *A task set $\mathcal{T}$ of n implicit-deadline tasks is FP-schedulable on a preemptive uniprocessor with the RM priority ordering if*

$$\mathrm{U}(\mathcal{T}) \;\leqslant\; n(2^{\frac{1}{n}} - 1), \tag{1}$$

where

$$\mathrm{U}(\mathcal{T}) \;\overset{\text{def}}{=}\; \sum_{(e,d,p)\,\in\,\mathcal{T}} \frac{e}{p} \tag{2}$$

is the utilization *of $\mathcal{T}$.*

Note that $\lim_{n\to\infty} n(2^{\frac{1}{n}} - 1) = \ln 2 \approx 0.693$, which leads to the simpler sufficient condition

$$\mathrm{U}(\mathcal{T}) \leqslant \ln 2, \tag{3}$$

which we will make use of instead of the one in Eq. (1).

Liu and Layland's utilization-based condition is only sufficient and only valid for task sets with implicit deadlines. Joseph and Pandya [7] gave an exact condition for FP-schedulability of task sets with constrained deadlines. This condition is based on the calculation of a task's *response time*, which is defined as the maximum amount of time that can pass between the release and finishing time of any job generated by that task.

Theorem I.4 (Joseph and Pandya [7]). *Let $\tau_{\mathrm{low}} = (e_{\mathrm{low}}, d_{\mathrm{low}}, p_{\mathrm{low}})$ be a constrained-deadline task scheduled by the FP scheduler on a preemptive uniprocessor, and let $\mathcal{T}^{\mathrm{hp}}$ be the set of tasks with higher priority than τ_{low} in the given priority ordering. The response time r_{low} of τ_{low} is then the smallest positive fix-point to*

$$r_{\mathrm{low}} \;=\; e_{\mathrm{low}} + \mathrm{rbf}(\mathcal{T}^{\mathrm{hp}}, r_{\mathrm{low}}), \tag{4}$$

where

$$\mathrm{rbf}(\mathcal{T}, r) \;\overset{\text{def}}{=}\; \sum_{(e,d,p)\,\in\,\mathcal{T}} \left\lceil \frac{r}{p} \right\rceil e \tag{5}$$

is the request bound function *of a set of tasks $\mathcal{T}$ in a time interval of size r. All jobs generated by τ_{low} are guaranteed to meet their deadlines with the given priority ordering if and only if $r_{\mathrm{low}} \leqslant d_{\mathrm{low}}$.*

Baruah et al. [4] gave an exact condition for EDF-schedulability of task sets with arbitrary deadlines.

Theorem I.5 (Baruah et al. [4]). *A task set $\mathcal{T}$ is EDF-schedulable on a preemptive uniprocessor if and only if $\mathrm{U}(\mathcal{T}) \leqslant 1$ and*

$$\forall \ell \geqslant 0, \quad \mathrm{dbf}(\mathcal{T}, \ell) \;\leqslant\; \ell, \tag{6}$$

where

$$\mathrm{dbf}(\mathcal{T}, \ell) \;\overset{\text{def}}{=}\; \sum_{(e,d,p)\,\in\,\mathcal{T}} \max\left\{0, \left\lfloor \frac{\ell - d}{p} \right\rfloor + 1\right\} e \tag{7}$$

is the demand bound function *of $\mathcal{T}$ in time interval lengths ℓ.*

Note that for a task set $\mathcal{T}$ with constrained deadlines, we have

$$\mathrm{dbf}(\mathcal{T}, \ell) \;=\; \sum_{(e,d,p)\,\in\,\mathcal{T}} \left(\left\lfloor \frac{\ell - d}{p} \right\rfloor + 1\right) e \tag{8}$$

for $\ell \geqslant 0$. As we consider constrained deadlines in this paper, we will use the simpler form in Eq. (8) for brevity.

We have also the following corollary.[1]

Corollary I.6 (Baruah et al. [4]). *A task set $\mathcal{T}$ with constrained deadlines is EDF-schedulable on a preemptive uniprocessor if and only if $\mathrm{U}(\mathcal{T}) \leqslant 1$ and*

$$\forall \ell \in \{0, 1, \ldots, \mathcal{P}(\mathcal{T}) - 1\}, \quad \mathrm{dbf}(\mathcal{T}, \ell) \;\leqslant\; \ell. \tag{9}$$

where

$$\mathcal{P}(\mathcal{T}) \;\overset{\text{def}}{=}\; \mathrm{lcm}\{p \mid (e,d,p) \in \mathcal{T}\} \tag{10}$$

is the hyper-period *of $\mathcal{T}$.*

The EDF-schedulability problem (or feasibility problem) is known to be strongly coNP-complete in the general case [8]. We will base our new results on the following theorem about the hardness in a more restricted setting with bounded utilization.

Theorem I.7 (Ekberg and Yi [9]). *Deciding whether a task set is EDF-schedulable on a preemptive uniprocessor is* coNP*-complete in the weak sense if restricted to task sets with constrained deadlines and utilization bounded from above by any constant c, such that $0 < c < 1$.*

[1]This is slightly different than the result reported by Baruah et al. [4], where the equivalent is stated for arbitrary deadlines. In this corollary we restrict attention to constrained deadlines and therefore get a slightly smaller range of values for ℓ to consider, which will be a useful property later on. This variant follows directly by the reasoning of Baruah et al. [4]

		Implicit deadlines ($d = p$)	Constrained deadlines ($d \leqslant p$)	Arbitrary deadlines (d, p unrelated)
FP	Arbitrary utilization	Weakly NP-complete (from this work)	Weakly NP-complete (from this work)	Weakly NP-hard (from this work) (*Open*†)
	Utilization bounded by a constant c	Polynomial time for $c \leqslant \ln 2$ and RM priorities [6] (*Open*‡)	Weakly NP-complete for $0 < c < 1$ (from this work)	Weakly NP-hard for $0 < c < 1$ (from this work) (*Open*†)
EDF (Feasibility)	Arbitrary utilization	Polynomial time [6]	Strongly coNP-complete [8]	Strongly coNP-complete [8]
	Utilization bounded by a constant c	Polynomial time [6]	Weakly coNP-complete for $0 < c < 1$ [9]	Weakly coNP-complete for $0 < c < 1$ [9]

Fig. 1. Computational complexity of the schedulability problem for sporadic or synchronous periodic tasks on a preemptive uniprocessor. Except for the entries marked *Open*, upper and lower bounds match, with all weakly NP- or coNP-complete problems also having known pseudo-polynomial time solutions. See the next page for a description of the open problems.

C. Contributions and Related Work

The results summarized in the previous section provide some upper bounds on the computational complexity of the FP-schedulability problem. Theorem I.4 immediately yields a pseudo-polynomial time algorithm for task sets with constrained deadlines. It also shows that the same problem is in NP (a set of small fix-points is a witness of schedulability). Theorem I.3 similarly gives a trivial polynomial-time algorithm for in the special case of task sets with implicit-deadlines, RM priorities and utilization bounded by $\ln 2$. (See, e.g., Baruah and Goossens [10] for a more in-depth discussion.) For the case with arbitrary deadlines, there are exponential-time algorithms, as shown by Lehoczky [3], but no pseudo-polynomial time algorithms are known.

Despite the wealth of research related to these problems, lower bounds on their computational complexity have been lacking for decades. To the best of our knowledge, the only previous result related to lower bounds on the FP-schedulability problem was given by Eisenbrand and Rothvoß [11]. They showed that with FP scheduling it is NP-hard to even approximate the response time of the lowest-priority task within a constant factor (i.e., to approximate the smallest fix-point in Eq. (4) for a given task). While this intuitively seems to suggest that the FP-schedulability problem is hard, Eisenbrand and Rothvoß [11] as well as Baruah and Pruhs [1] point out that this does not follow. The reason is that the reductions employed by Eisenbrand and Rothvoß can create some higher-priority tasks that are clearly unschedulable in order to make it hard to calculate the response time of a single lower-priority task. Rothvoß [12] later conjectured that the FP-schedulability problem is NP-hard, even with implicit deadlines and RM priorities.

Our contributions are to provide lower bounds for the FP-schedulability problem that in several cases match the known upper bounds. Our main results are as follows.

(1) The FP-schedulability problem is weakly NP-hard, even when restricted to task sets with implicit deadlines and RM priority ordering.
(2) The FP-schedulability problem is weakly NP-hard, even when restricted to task sets with constrained deadlines, DM priority ordering and utilization bounded by any constant c, such that $0 < c < 1$.

Item (1) settles a stronger version of *Open Problem 1* listed by Baruah and Pruhs [1] and proves the conjecture of Rothvoß [12]. Item (2) shows that with constrained deadlines, the complexity of FP-schedulability remains even when we restrict attention to task sets with arbitrarily low utilization, similar to the case for EDF-schedulability. As the RM and DM priority orderings are known to be optimal in their respective settings, we note that the hardness results of items (1) and (2) remain the same if we were to instead ask if a given task set is FP-schedulable with *any* priority ordering.

Figure 1 combines our lower bounds with the known upper bounds, and contrasts these with the known bounds on EDF-schedulability. Except for those entries marked as *Open*, the lower bounds match the upper bounds, with *weakly* NP- and coNP-complete problems having known pseudo-polynomial time algorithms.

Open†*:* The lower bounds here carry over from the corresponding problems with constrained deadlines. However, while there are exponential-time algorithms for FP-schedulability with arbitrary deadlines (see Lehoczky [3]), there are no known pseudo-polynomial time algorithms. To the best of our knowledge, membership in NP is also open.

Open‡*:* If the FP-schedulability problem is restricted to task sets with utilization bounded by a constant c and RM priority ordering, then Theorem I.3 yields a trivial polynomial-time algorithm for $c \leqslant \ln 2$. It is open if there exists a polynomial-time algorithm for the case when we have RM priority ordering but $\ln 2 < c < 1$, or for the case where an arbitrary priority ordering can be given and $0 < c < 1$.

II. The Hardness of FP-schedulability

In this section we show that the FP-schedulability problem is weakly NP-hard, even when restricted to either of the two special cases mentioned in the previous section. We can prove either case with only a minor variation to the proofs, so the following will target both cases. We split this proof into a chain of two reductions.

(1) The first is a polynomial-time many-one reduction from the EDF-schedulability problem for task sets with constrained deadlines and utilization bounded by any constant c, such that $0 < c < 1$, to a special case of the same problem that is further restricted to task sets where all tasks have pairwise coprime periods.
(2) The second is a polynomial-time many-one reduction from that special case of the EDF-schedulability problem (though here we must have $0 < c \leqslant \ln 2$) to the complement of the FP-schedulability problem. This works by finding a duality between demand bound functions and request bound functions that exists whenever the tasks have pairwise coprime periods.

As the source problem of the first reduction is coNP-hard by Theorem I.7 for any c such that $0 < c < 1$, the NP-hardness of the FP-schedulability problem will follow.

For the first reduction we will use a result from number theory about the Jacobsthal function [13]. The Jacobsthal function $g(n)$ gives the smallest number m, such that all intervals of m consecutive integers contain at least one number coprime to n. The result we will use is an upper bound on the Jacobsthal function due to Iwaniec [14], from which we have

$$g(n) \leqslant \mathcal{K} \log^2(n) \tag{11}$$

for $n \geqslant 2$ and some unknown positive constant $\mathcal{K}$. As a convenience we assume, without loss of generality, that $\mathcal{K} \in \mathbb{N}^+$.

A. Reducing EDF-schedulability with Bounded Utilization to a Special Case with Pairwise Coprime Periods

Let c be any constant such that $0 < c < 1$. Then let $\mathcal{T}_c$ denote an instance of the EDF-schedulability problem restricted to task sets with constrained deadlines and utilization bounded by c. Given any $\mathcal{T}_c$, our reduction produces a task set $\mathcal{T}_c^\perp$ that is further restricted so that all tasks in $\mathcal{T}_c^\perp$ have pairwise coprime periods, while $\mathcal{T}_c^\perp$ is EDF-schedulable if and only if $\mathcal{T}_c$ is.

First, define

$$n \overset{\text{def}}{=} |\mathcal{T}_c|, \tag{12}$$
$$\kappa \overset{\text{def}}{=} (10\mathcal{K}n\mathcal{P}(\mathcal{T}_c))^3, \tag{13}$$

where $\mathcal{K}$ is the constant from Iwaniec's bound on the Jacobsthal function in Eq. (11). Then, let the tasks in $\mathcal{T}_c$ be denoted as $\{(e_1, d_1, p_1), \ldots, (e_n, d_n, p_n)\}$, and let those tasks be indexed by non-decreasing periods, so that $p_j \leqslant p_i$ if $j < i$.

Now, the task set $\mathcal{T}_c^\perp$ produced by the reduction is defined as

$$\mathcal{T}_c^\perp \overset{\text{def}}{=} \{(e'_1, d'_1, p'_1), \ldots, (e'_n, d'_n, p'_n)\}, \tag{14}$$

where

$$e'_i \overset{\text{def}}{=} \kappa e_i, \tag{15}$$
$$d'_i \overset{\text{def}}{=} \kappa d_i, \tag{16}$$
$$p'_i \overset{\text{def}}{=} \min\{m \geqslant \kappa p_i \mid m \text{ is coprime to } p'_j, \forall j < i\}. \tag{17}$$

Note that by this construction, $\mathcal{T}_c^\perp$ is a copy of $\mathcal{T}_c$ where task parameters have been scaled by κ, except that the periods might be even larger to ensure that they are pairwise coprime.

We begin by establishing a lemma about how increasing the relative magnitudes of the periods by a moderate amount does not affect EDF-schedulability. Then we show in another lemma that the conditions of the first lemma applies to the task set $\mathcal{T}_c^\perp$, relative to $\mathcal{T}_c$, and that they therefore have the same EDF-schedulability. Last we show how to compute $\mathcal{T}_c^\perp$ in polynomial time.

Lemma II.1. *Let $\mathcal{T}$ be a task set with constrained deadlines and let $\tilde{\mathcal{T}}$ be another task set satisfying*

$$\tilde{\mathcal{T}} = \{(ke_i, kd_i, kp_i + \delta_i) \mid (e_i, d_i, p_i) \in \mathcal{T}\},$$

where $k \in \mathbb{N}^+$ and

$$0 \leqslant \delta_i < \frac{kp_i}{\mathcal{P}(\mathcal{T})}.$$

Then, $\tilde{\mathcal{T}}$ is EDF-schedulable if and only if $\mathcal{T}$ is EDF-schedulable.

Proof: We show the two directions separately.

$\mathcal{T}$ is EDF-schedulable $\Longrightarrow \tilde{\mathcal{T}}$ is EDF-schedulable:

Assume that $\mathcal{T}$ is EDF-schedulable. Then, for all $\ell \geqslant 0$ we have

$$\begin{aligned}
\mathrm{dbf}(\tilde{\mathcal{T}}, \ell) &= \sum_{(\tilde{e}_i, \tilde{d}_i, \tilde{p}_i) \in \tilde{\mathcal{T}}} \left(\left\lfloor \frac{\ell - \tilde{d}_i}{\tilde{p}_i} \right\rfloor + 1 \right) \tilde{e}_i \\
&= \sum_{(e_i, d_i, p_i) \in \mathcal{T}} \left(\left\lfloor \frac{\ell - k d_i}{k p_i + \delta_i} \right\rfloor + 1 \right) k e_i \\
&\leqslant \sum_{(e_i, d_i, p_i) \in \mathcal{T}} \left(\left\lfloor \frac{\frac{\ell}{k} - d_i}{p_i} \right\rfloor + 1 \right) k e_i \\
&= k \, \mathrm{dbf}(\mathcal{T}, \ell / k) \\
&\overset{(*)}{\leqslant} \ell,
\end{aligned}$$

where $(*)$ follows from Theorem I.5 and the EDF-schedulability of $\mathcal{T}$. By the same theorem, $\tilde{\mathcal{T}}$ must also be EDF-schedulable.

$\tilde{\mathcal{T}}$ is EDF-schedulable $\Longrightarrow \mathcal{T}$ is EDF-schedulable:

Assume that $\tilde{\mathcal{T}}$ is EDF-schedulable. Then, for all $\ell \geqslant 0$ we have

$$\begin{aligned}
\mathrm{dbf}(\mathcal{T}, \ell) &= \sum_{(e_i, d_i, p_i) \in \mathcal{T}} \left(\left\lfloor \frac{\ell - d_i}{p_i} \right\rfloor + 1 \right) e_i \\
&= \sum_{(e_i, d_i, p_i) \in \mathcal{T}} \left(\left\lfloor \frac{\left(k + \frac{k}{\mathcal{P}(\mathcal{T})}\right)(\ell - d_i)}{\left(k + \frac{k}{\mathcal{P}(\mathcal{T})}\right) p_i} \right\rfloor + 1 \right) e_i \\
&= \sum_{(e_i, d_i, p_i) \in \mathcal{T}} \left(\left\lfloor \frac{k\ell + \frac{k\ell}{\mathcal{P}(\mathcal{T})} - k d_i - \frac{k d_i}{\mathcal{P}(\mathcal{T})}}{k p_i + \frac{k p_i}{\mathcal{P}(\mathcal{T})}} \right\rfloor + 1 \right) e_i \\
&\leqslant \sum_{(\tilde{e}_i, \tilde{d}_i, \tilde{p}_i) \in \tilde{\mathcal{T}}} \left(\left\lfloor \frac{k\ell + \frac{k\ell}{\mathcal{P}(\mathcal{T})} - \tilde{d}_i}{\tilde{p}_i} \right\rfloor + 1 \right) \frac{\tilde{e}_i}{k} \\
&= \frac{\mathrm{dbf}\left(\tilde{\mathcal{T}}, k\ell + \frac{k\ell}{\mathcal{P}(\mathcal{T})}\right)}{k} \\
&\overset{(*)}{\leqslant} \ell + \frac{\ell}{\mathcal{P}(\mathcal{T})},
\end{aligned}$$

where $(*)$ follows from Theorem I.5 and the EDF-schedulability of $\tilde{\mathcal{T}}$. Now, because $\mathrm{dbf}(\mathcal{T}, \ell)$ is integer-valued, we must have

$$\mathrm{dbf}(\mathcal{T}, \ell) \leqslant \left\lfloor \ell + \frac{\ell}{\mathcal{P}(\mathcal{T})} \right\rfloor,$$

but for all $\ell \in \{0, 1, \ldots, \mathcal{P}(\mathcal{T}) - 1\}$ we then have

$$\mathrm{dbf}(\mathcal{T}, \ell) \leqslant \left\lfloor \ell + \frac{\ell}{\mathcal{P}(\mathcal{T})} \right\rfloor = \ell,$$

and it follows that $\mathcal{T}$ is EDF-schedulable by Corollary I.6. ■

Now we show that task set $\mathcal{T}_c^{\perp}$ meets the conditions of Lemma II.1, relative to $\mathcal{T}_c$, and therefore preserves the EDF-schedulability of $\mathcal{T}_c$.

Lemma II.2. *$\mathcal{T}_c^{\perp}$ is EDF-schedulable if and only if $\mathcal{T}_c$ is EDF-schedulable.*

Proof: The lemma holds, by construction and Lemma II.1, if

$$p_i' - \kappa p_i < \frac{\kappa p_i}{\mathcal{P}(\mathcal{T}_c)}, \tag{18}$$

for all $1 \leqslant i \leqslant n$. We show that Eq. (18) holds by strong induction over i. The base case ($i = 1$) trivially holds as $p_1' = \kappa p_1$. For $1 < i \leqslant n$, note that p_i' is coprime to p_j' for all $j < i$ if and only if p_i' is coprime to N, where $N = \prod_{j=1}^{i-1} p_j'$. We know that all intervals of $g(N)$ consecutive integers contain a number coprime to N, where g is the Jacobsthal function. Also note that we have

$$\begin{aligned}
N &= \prod_{j=1}^{i-1} p_j' \\
&\overset{(*)}{<} \prod_{j=1}^{i-1} \left(\kappa p_j + \frac{\kappa p_j}{\mathcal{P}(\mathcal{T}_c)} \right) \\
&\leqslant \prod_{j=1}^{i-1} 2\kappa p_j \\
&\overset{(**)}{<} (2\kappa p_i)^n,
\end{aligned}$$

where $(*)$ follows from the induction hypothesis and $(**)$ from the ordering of task indices by non-decreasing periods. Using Iwaniec's bound on g from Eq. (11), we then have

$$\begin{aligned}
g(N) &\leqslant \mathcal{K} \log^2(N) \\
&< \mathcal{K} \log^2((2\kappa p_i)^n) \\
&= \mathcal{K} n^2 \log^2(2\kappa p_i) \\
&= 36\mathcal{K} n^2 \log^2\left(\sqrt[6]{2\kappa p_i}\right) \\
&< 36\mathcal{K} n^2 \sqrt[3]{2\kappa p_i} \\
&< 36\mathcal{K} n^2 (2\sqrt[3]{\kappa} p_i) \\
&= 720\mathcal{K}^2 n^3 \mathcal{P}(\mathcal{T}_c) p_i \\
&< \frac{\kappa p_i}{\mathcal{P}(\mathcal{T}_c)}.
\end{aligned} \tag{19}$$

It follows that there exists a number coprime to N in the interval $\left[\kappa p_i, \kappa p_i + \frac{\kappa p_i}{\mathcal{P}(\mathcal{T}_c)}\right)$. From the definition of p_i', it must therefore be the case that

$$p_i' - \kappa p_i < \frac{\kappa p_i}{\mathcal{P}(\mathcal{T}_c)}.$$

There is nothing to show for $i > n$, so this concludes the induction step and the proof. ■

Finally, the reduction can be shown to be valid.

Lemma II.3. *Deciding whether a task set of sporadic or synchronous periodic tasks is EDF-schedulable on a preemptive uniprocessor is* coNP*-complete in the weak sense if restricted to task sets with constrained deadlines, utilization bounded from above by any constant c, such that $0 < c < 1$, and tasks with pairwise coprime periods.*

Proof: We know from Theorem I.7 that this decision problem without the restriction to pairwise coprime periods is coNP-complete for any constant c such that $0 < c < 1$. We have shown that there exists a reduction[2] to the special case with pairwise coprime periods that by Lemma II.2 is a valid many-one reduction. What remains to be shown is that the reduction can be computed in polynomial time.

For this, the only challenge is to compute the values of the periods p'_i. To see that this can be done in polynomial time, we again use Iwaniec's bound on the Jacobsthal function g. By definition, p'_i is the smallest integer not less than κp_i and coprime to $N = \prod_{j=1}^{i-1} p'_j$. This number must be in the interval $[\kappa p_i, \kappa p_i + g(N))$, but from Eq. (19) we have

$$g(N) \; < \; \mathcal{K} n^2 \log^2(2\kappa p_i).$$

As $\mathcal{K} n^2 \log^2(2\kappa p_i)$ is bounded by some polynomial in the size of the representation of $\mathcal{T}_c$, we know that p'_i is contained in an interval of polynomial length. We can search this interval for p'_i by looking at all integers $\kappa p_i, \kappa p_i + 1, \dots$ until we find a number that is coprime to N. Using standard algorithms for computing greatest common divisors, this can be done in polynomial time. ■

B. Reducing EDF-schedulability to the Complement of FP-schedulability

Now we describe a polynomial-time many-one reduction from the special case of the EDF-schedulability problem that was shown to be coNP-complete in Lemma II.3 to the complement of the FP-schedulability problem.

Let c be any constant such that $0 < c \leqslant \ln 2$. Given an instance $\mathcal{T}_c^{\perp}$ of the EDF-schedulability problem with constrained deadlines, utilization bounded by c, and pairwise coprime periods, let $\hat{\ell}$ be the smallest number such that

$$\hat{\ell} \;\equiv\; d \pmod{p}, \qquad \forall\, (e,d,p) \in \mathcal{T}_c^{\perp} \tag{20}$$

and

$$\hat{\ell} \;>\; \max\{p \mid (e,d,p) \in \mathcal{T}_c^{\perp}\}. \tag{21}$$

Because the tasks in $\mathcal{T}_c^{\perp}$ have pairwise coprime periods, we know by the Chinese remainder theorem that $\hat{\ell}$ exists and that $\hat{\ell} \leqslant 2\mathcal{P}(\mathcal{T}_c^{\perp})$. Note that if $\mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell}) > \hat{\ell}$, then $\mathcal{T}_c^{\perp}$ is infeasible by Theorem I.5 and the reduction is trivial. In the following we assume, without loss of generality, that

$$\mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell}) \;\leqslant\; \hat{\ell}. \tag{22}$$

The reduction then produces a task set $\mathcal{T}_{\text{FP}}$ as

$$\mathcal{T}_{\text{FP}} \;\overset{\text{def}}{=}\; \mathcal{T}_{\text{FP}}^{\text{hp}} \,\cup\, \{\tau_{\text{low}}\}, \tag{23}$$

[2]This proof is actually somewhat non-constructive. As we do not know the value of $\mathcal{K}$, we have not described a concrete reduction, but only demonstrated that one exists.

where

$$\mathcal{T}_{\text{FP}}^{\text{hp}} \;\overset{\text{def}}{=}\; \{(e,p,p) \mid (e,d,p) \in \mathcal{T}_c^{\perp}\} \tag{24}$$

$$\tau_{\text{low}} \;\overset{\text{def}}{=}\; (e_{\text{low}}, d_{\text{low}}, p_{\text{low}}) \tag{25}$$

$$e_{\text{low}} \;\overset{\text{def}}{=}\; \hat{\ell} - \mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell}) + 1 \tag{26}$$

$$d_{\text{low}} \;\overset{\text{def}}{=}\; \hat{\ell} \tag{27}$$

$$p_{\text{low}} \;\overset{\text{def}}{=}\; \varphi \hat{\ell} \tag{28}$$

for some $\varphi \in \mathbb{N}^+$. We will set the value of φ later in Theorem II.8, to target either of the two special cases of the FP-schedulability problem that was mentioned in Section I-C.

First we establish two lemmas, starting with a lemma about the sizes of counterexamples to the EDF-schedulability of $\mathcal{T}_c^{\perp}$.

Lemma II.4. *$\mathcal{T}_c^{\perp}$ is EDF-schedulable if and only if*

$$\forall \ell \in \{0, 1, \dots, \hat{\ell}\}, \quad \mathrm{dbf}(\mathcal{T}_c^{\perp}, \ell) \;\leqslant\; \ell. \tag{29}$$

Proof: The necessity of the condition in Eq. (29) follows directly from Theorem I.5. We show the sufficiency by contradiction. Assume for this purpose that the condition in Eq. (29) holds but $\mathcal{T}_c^{\perp}$ is not EDF-schedulable. Then, by assumption and Theorem I.5 there must exist a $\lambda \in \mathbb{N}^+$ such that $\mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell} + \lambda) > \hat{\ell} + \lambda$, but

$$\begin{aligned}
\mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell} + \lambda) \;&=\; \sum_{(e,d,p) \in \mathcal{T}_c^{\perp}} \left(\left\lfloor \frac{\hat{\ell} + \lambda - d}{p} \right\rfloor + 1 \right) e \\
&\overset{(*)}{=} \sum_{(e,d,p) \in \mathcal{T}_c^{\perp}} \left(\frac{\hat{\ell} - d}{p} + \left\lfloor \frac{\lambda}{p} \right\rfloor + 1 \right) e \\
&\overset{(*)}{=} \sum_{(e,d,p) \in \mathcal{T}_c^{\perp}} \left(\left\lfloor \frac{\hat{\ell} - d}{p} \right\rfloor + 1 \right) e \;+ \sum_{(e,d,p) \in \mathcal{T}_c^{\perp}} \left\lfloor \frac{\lambda}{p} \right\rfloor e \\
&\leqslant\; \mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell}) + \lambda \mathrm{U}(\mathcal{T}_c^{\perp}) \\
&\overset{(**)}{\leqslant}\; \hat{\ell} + \lambda \mathrm{U}(\mathcal{T}_c^{\perp}) \\
&\leqslant\; \hat{\ell} + \lambda c \\
&<\; \hat{\ell} + \lambda,
\end{aligned}$$

where the equalities marked $(*)$ hold because $(\hat{\ell} - d)/p$ is an integer for all $(e,d,p) \in \mathcal{T}_c^{\perp}$ by the definition of $\hat{\ell}$, and $(**)$ follows by assumption. The sufficiency of the condition in Eq. (29) follows from this contradiction. ■

Next we demonstrate a connection between demand bound functions and request bound functions. We know from Theorems I.4 and I.5 that these can provide conditions for EDF- and FP-schedulability, respectively, which makes this a key step for the reduction.

Lemma II.5. *For all $\ell \in \{0, 1, \dots, \hat{\ell}\}$, we have*

$$\mathrm{dbf}(\mathcal{T}_c^{\perp}, \hat{\ell}) - \mathrm{dbf}(\mathcal{T}_c^{\perp}, \ell) \;=\; \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, \hat{\ell} - \ell).$$

Proof: Take any $\ell \in \{0, 1, \ldots, \hat{\ell}\}$ and we have

$$\begin{aligned}
&\mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{dbf}(\mathcal{T}_c^\perp, \ell) \\
&\quad= \sum_{(e,d,p)\in\mathcal{T}_c^\perp} \left(\left\lfloor \frac{\hat{\ell}-d}{p} \right\rfloor + 1\right)e - \left(\left\lfloor \frac{\ell-d}{p} \right\rfloor + 1\right)e \\
&\quad= \sum_{(e,d,p)\in\mathcal{T}_c^\perp} \left(\left\lfloor \frac{\hat{\ell}-d}{p} \right\rfloor - \left\lfloor \frac{\ell-d}{p} \right\rfloor\right)e \\
&\quad\overset{(*)}{=} \sum_{(e,d,p)\in\mathcal{T}_c^\perp} \left(\frac{\hat{\ell}-d}{p} - \left\lfloor \frac{\ell-d}{p} \right\rfloor\right)e \\
&\quad\overset{(*)}{=} \sum_{(e,d,p)\in\mathcal{T}_c^\perp} \left(\left\lceil \frac{\hat{\ell}-d-(\ell-d)}{p} \right\rceil\right)e \\
&\quad= \sum_{(e,d,p)\in\mathcal{T}_c^\perp} \left\lceil \frac{\hat{\ell}-\ell}{p} \right\rceil e \\
&\quad\overset{(**)}{=} \sum_{(e,d,p)\in\mathcal{T}_{\text{FP}}^{\text{hp}}} \left\lceil \frac{\hat{\ell}-\ell}{p} \right\rceil e \\
&\quad= \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, \hat{\ell}-\ell),
\end{aligned}$$

where equalities marked $(*)$ follow because $(\hat{\ell}-d)/p$ is an integer and $(**)$ follows from the definition of $\mathcal{T}_{\text{FP}}^{\text{hp}}$. ■

Recall that we restricted the source problem of our reduction to instances $\mathcal{T}_c^\perp$ with $\mathrm{U}(\mathcal{T}_c^\perp) \leqslant c \leqslant \ln 2$. As is demonstrated by the following lemma, this means that the FP-schedulability of the constructed task set $\mathcal{T}_{\text{FP}}$ hinges only on the schedulability of its lowest priority task τ_{low} for appropriate priority orderings.

Lemma II.6. *Task set $\mathcal{T}_{\text{FP}}$ is FP-schedulable with either RM or DM priority ordering if and only if under this scheduling all jobs generated by τ_{low} are guaranteed to meet their deadlines.*

Proof: Assume either RM or DM priority ordering. Then τ_{low} is the lowest-priority task in $\mathcal{T}_{\text{FP}}$ as by Eqs. (21), (27) and (28) we have both $p_{\text{low}} > \max\{p \mid (e,d,p) \in \mathcal{T}_{\text{FP}}^{\text{hp}}\}$ and $d_{\text{low}} > \max\{d \mid (e,d,p) \in \mathcal{T}_{\text{FP}}^{\text{hp}}\}$. As the tasks in $\mathcal{T}_{\text{FP}}^{\text{hp}}$ all have implicit deadlines, the RM and DM priority orderings are in fact the same for $\mathcal{T}_{\text{FP}}$. Note also that by construction we have

$$\mathrm{U}(\mathcal{T}_{\text{FP}}^{\text{hp}}) \;=\; \mathrm{U}(\mathcal{T}_c^\perp) \;\leqslant\; c \;\leqslant\; \ln 2.$$

As τ_{low} can not affect the scheduling of the higher-priority tasks in $\mathcal{T}_{\text{FP}}^{\text{hp}}$, we know by Theorem I.3 (see especially the simplified condition in Eq. (3)) that all jobs generated by tasks in $\mathcal{T}_{\text{FP}}^{\text{hp}}$ will meet their deadlines. Task set $\mathcal{T}_{\text{FP}}$ is therefore FP-schedulable if and only if all jobs generated by τ_{low} will meet their deadlines. ■

With a last lemma we show that the schedulability of the jobs generated by τ_{low}, and thus the whole task set $\mathcal{T}_{\text{FP}}$, is inversely related to the EDF-schedulability of $\mathcal{T}_c^\perp$.

Lemma II.7. *Task set $\mathcal{T}_{\text{FP}}$ is FP-schedulable with either RM or DM priority ordering if and only if $\mathcal{T}_c^\perp$ is not EDF-schedulable.*

Proof: First note that by Lemma II.6 we know that $\mathcal{T}_{\text{FP}}$ is FP-schedulable under RM or DM priority ordering if and only if all jobs generated by task τ_{low} are guaranteed to meet their deadlines. By Theorem I.4, this is the case if and only if there exists a fix-point r_{low} to the equation

$$r_{\text{low}} \;=\; e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r_{\text{low}}),$$

such that $0 < r_{\text{low}} \leqslant d_{\text{low}}$. Now we separately prove the two directions of the lemma.

$\mathcal{T}_{\text{FP}}$ *is FP-schedulable* $\Longrightarrow$ $\mathcal{T}_c^\perp$ *is not EDF-schedulable:*

Assume that $\mathcal{T}_{\text{FP}}$ is FP-schedulable, and hence that there exists some r_{low} such that $0 < r_{\text{low}} \leqslant d_{\text{low}}$ and $r_{\text{low}} = e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r_{\text{low}})$. Then,

$$\begin{aligned}
&\mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell} - r_{\text{low}}) \\
&\quad= \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell} - r_{\text{low}}) + \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) \\
&\quad\overset{(*)}{=} \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, \hat{\ell} - (\hat{\ell} - r_{\text{low}})) \\
&\quad= \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r_{\text{low}})) \\
&\quad= \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) + e_{\text{low}} - r_{\text{low}} \\
&\quad\overset{(**)}{=} \hat{\ell} - r_{\text{low}} + 1,
\end{aligned}$$

where $(*)$ follows from Lemma II.5 and $(**)$ from the definition of e_{low}. By Theorem I.5 it follows that $\mathcal{T}_c^\perp$ is not EDF-schedulable.

$\mathcal{T}_c^\perp$ *is not EDF-schedulable* $\Longrightarrow$ $\mathcal{T}_{\text{FP}}$ *is FP-schedulable:*

Assume that $\mathcal{T}_c^\perp$ is not EDF-schedulable. Then, by Lemma II.4 there exists some $\ell \in \{0, 1, \ldots, \hat{\ell}\}$ such that $\mathrm{dbf}(\mathcal{T}_c^\perp, \ell) > \ell$. Let $r' = d_{\text{low}} - \ell$ and note that

$$\begin{aligned}
r' &= d_{\text{low}} - \ell \\
&= \hat{\ell} - \ell \\
&\overset{(*)}{>} \hat{\ell} - \mathrm{dbf}(\mathcal{T}_c^\perp, \ell) \\
&= \hat{\ell} - \mathrm{dbf}(\mathcal{T}_c^\perp, \ell) + \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) \\
&= e_{\text{low}} + \mathrm{dbf}(\mathcal{T}_c^\perp, \hat{\ell}) - \mathrm{dbf}(\mathcal{T}_c^\perp, \ell) - 1 \\
&\overset{(**)}{=} e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, \hat{\ell} - \ell) - 1 \\
&= e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r') - 1,
\end{aligned}$$

where $(*)$ follows by assumption and $(**)$ follows from Lemma II.5. Because both r' and $e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r') - 1$ are integers, we must then have

$$r' \;\geqslant\; e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r'), \tag{30}$$

but by the definition of request bound functions (see Eq. (5)) we also have

$$0 \;<\; e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, 0). \tag{31}$$

Combining Eqs. (30) and (31) with the observation that $\mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r)$ is a non-decreasing function in r, we can see that there must exist some r_{low} such that $0 < r_{\text{low}} \leqslant r' \leqslant d_{\text{low}}$ and $r_{\text{low}} = e_{\text{low}} + \mathrm{rbf}(\mathcal{T}_{\text{FP}}^{\text{hp}}, r_{\text{low}})$. By Theorem I.4, it follows that all jobs from τ_{low} are guaranteed to meet their deadlines, and thus by Lemma II.6 that $\mathcal{T}_{\text{FP}}$ is FP-schedulable. ■

Finally we can show the NP-hardness of the FP-schedulability problem.

Theorem II.8. *Deciding whether a set of sporadic or synchronous periodic tasks is FP-schedulable with a given priority ordering on a preemptive uniprocessor is* NP*-hard in the weak sense. This holds even if restricted in either of the following ways.*

(i) All tasks have implicit deadlines and the priority ordering is RM.

(ii) All tasks have constrained deadlines, the priority ordering is DM and the utilization of the task set is bounded from above by any constant $\tilde{c}$, such that $0 < \tilde{c} < 1$.

Proof: We have described a reduction from the EDF-schedulability problem restricted to task sets with constrained deadlines, pairwise coprime periods and utilization bounded by any constant c, such that $0 < c \leqslant \ln 2$, to the complement of the FP-schedulability problem. By Lemma II.7, this is a valid many-one reduction with both the RM and DM priority orderings. Note that the value of $\hat{\ell}$ can be computed in polynomial time using standard algorithms for the Chinese remainder theorem, and the rest of the reduction can trivially be computed in polynomial time as well. Because the above EDF-schedulabilty problem is coNP-complete by Lemma II.3, the FP-schedulability problem is (weakly) NP-hard.

We can get either of the two restrictions (i) and (ii) by adapting the value of the constant φ in Eq. (28). Note that all lemmas shown so far are independent of the value of φ. If we set $\varphi = 1$, then $\mathcal{T}_{\text{FP}}$ has implicit deadlines and the NP-hardness of the FP-schedulability problem with restriction (i) follows.

Consider instead restriction (ii) with any constant $\tilde{c}$, such that $0 < \tilde{c} < 1$. We choose the constant c for bounding the utilization for the EDF-schedulability problem that is the source of the reduction as $c = \tilde{c}/2$. Note that we still have $c \leqslant \ln 2$ and that our EDF-schedulability problem is coNP-complete also with this c by Lemma II.3. Then we set $\varphi = \lceil 2/\tilde{c} \rceil$. Now, $\mathcal{T}_{\text{FP}}$ has constrained deadlines and we also have

$$\begin{aligned} \mathrm{U}(\mathcal{T}_{\text{FP}}) &= \mathrm{U}(\mathcal{T}_{\text{FP}}^{\text{hp}}) + \mathrm{U}(\{\tau_{\text{low}}\}) \\ &= \mathrm{U}(\mathcal{T}_{c}^{\perp}) + \frac{e_{\text{low}}}{p_{\text{low}}} \\ &\leqslant c + \frac{1}{\varphi} \\ &\leqslant \frac{\tilde{c}}{2} + \frac{\tilde{c}}{2} \\ &= \tilde{c}. \end{aligned}$$

The NP-hardness of the FP-schedulability problem with restriction (ii) follows. ∎

III. Conclusions

The computational complexity of deciding whether a set of sporadic or synchronous periodic tasks is FP-schedulable on a preemptive uniprocessor has been a longstanding open problem, arguably going back to the seminal work of Liu and Layland [6]. We have provided lower bounds on the complexity of this problem, which in several important settings match upper bounds provided by classic results in real-time scheduling theory.

An important outstanding question is the exact complexity of FP-schedulability with arbitrary deadlines. This problem has a well-known exponential-time algorithm, but there remains a gap between this and our lower bound, which shows that the problem is weakly NP-hard.

References

[1] S. Baruah and K. Pruhs, "Open problems in real-time scheduling," *Journal of Scheduling*, vol. 13, no. 6, pp. 577–582, 2010.

[2] M. L. Dertouzos, "Control robotics: The procedural control of physical processes," in *Proceedings of the IFIP congress*, vol. 74, 1974, pp. 807–813.

[3] J. P. Lehoczky, "Fixed priority scheduling of periodic task sets with arbitrary deadlines," in *Proceedings of the 11th Real-Time Systems Symposium (RTSS)*, Dec 1990, pp. 201–209.

[4] S. Baruah, A. K. Mok, and L. E. Rosier, "Preemptively scheduling hard-real-time sporadic tasks on one processor," in *Proceedings of the 11th Real-Time Systems Symposium (RTSS)*, 1990, pp. 182–190.

[5] J. Y.-T. Leung and J. Whitehead, "On the complexity of fixed-priority scheduling of periodic, real-time tasks," *Performance Evaluation*, vol. 2, no. 4, pp. 237–250, 1982.

[6] C.-L. Liu and J. W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," *Journal of the ACM*, vol. 20, no. 1, pp. 46–61, 1973.

[7] M. Joseph and P. Pandya, "Finding response times in a real-time system," *The Computer Journal*, vol. 29, no. 5, pp. 390–395, 1986.

[8] P. Ekberg and W. Yi, "Uniprocessor feasibility of sporadic tasks with constrained deadlines is strongly coNP-complete," in *Proceedings of the 27th Euromicro Conference on Real-Time Systems (ECRTS)*, 2015, pp. 281 – 286.

[9] ——, "Uniprocessor feasibility of sporadic tasks remains coNP-complete under bounded utilization," in *Proceedings of the 36th Real-Time Systems Symposium (RTSS)*, 2015, pp. 87–95.

[10] S. Baruah and J. Goossens, "Scheduling real-time tasks: Algorithms and complexity," 2004.

[11] F. Eisenbrand and T. Rothvoß, "Static-priority real-time scheduling: Response time computation is NP-hard," in *Proceedings of the 29th Real-Time Systems Symposium (RTSS)*. IEEE Computer Society, 2008, pp. 397–406.

[12] T. Rothvoss, "On the computational complexity of periodic scheduling," Ph.D. dissertation, SB, Lausanne, 2009.

[13] E. E. Jacobsthal, "Über sequenzen ganzer zahlen: Von denen keine zu n teilerfremd ist, 1-3," *Norske Vid. Selsk. Forh. (Trondheim)*, vol. 33, pp. 117–139, 1960.

[14] H. Iwaniec, "On the problem of Jacobsthal," *Demonstratio Math.*, vol. 11, pp. 225–231, 1978.

⊠