Uniprocessor Feasibility of Sporadic Tasks Remains coNP-complete Under Bounded Utilization

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

***Abstract*—A central problem in real-time scheduling theory is to decide whether a sporadic task system with constrained deadlines is feasible on a preemptive uniprocessor. It is known that this problem is strongly coNP-complete in the general case, but also that there exists a pseudo-polynomial time solution for instances with utilization bounded from above by any constant c, where $0 < c < 1$. For a long time it has been unknown whether the bounded case also has a polynomial-time solution. We show that for any choice of the constant c, such that $0 < c < 1$, the bounded feasibility problem is (weakly) coNP-complete, and thus that no polynomial-time solution exists for it, unless P = NP.**

I. Introduction

We let a sporadic task system be defined as a finite multiset T of *tasks*, where each task is a triple $(e, d, p) \in \mathbb{N}^3_+$, representing its worst-case execution time, relative deadline and minimum inter-arrival separation (or period), respectively.

A sporadic task *generates* potentially unbounded sequences of *jobs*. A job is an instance of the task's workload, characterized by a release time, an execution time and an absolute deadline. A job from task (e, d, p) has execution time not larger than e time units and absolute deadline exactly d time units after its release time. Release times of two consecutive jobs from (e, d, p) are separated by at least p time units. A sporadic task system T generates any interleaving of job sequences that can be generated by each of the tasks $(e, d, p) \in \mathsf{T}$.

For a sequence of jobs to be successfully scheduled, every job must be executed for a total duration equal to its execution time, between its release time and its absolute deadline. In the following we assume a preemptive uniprocessor, meaning that only one job can be executed at a time, but a job can be paused and resumed at a later time at no additional cost.

Definition I.1 (Feasibility)**.** *A task system T is* feasible *if and only if there is some scheduling algorithm that will successfully schedule all job sequences that can be generated by T.* □

A task system T is said to have *implicit deadlines* if $d = p$ for all $(e, d, p) \in \mathsf{T}$, and said to have *constrained deadlines* if $d \leqslant p$ for all $(e, d, p) \in \mathsf{T}$.

The *utilization* of a task system T is a measure of its asymptotic resource requirements, and is defined as

$$\mathrm{U}(\mathsf{T}) \stackrel{\text{def}}{=} \sum_{(e,d,p)\in\mathsf{T}} \frac{e}{p}. \tag{1}$$

Liu and Layland [1] showed that a task system T with implicit deadlines is feasible if and only if $\mathrm{U}(\mathsf{T}) \leqslant 1$, but this is not a sufficient condition with constrained deadlines.

Dertouzos [2] showed that Earliest Deadline First (EDF) is an optimal scheduling algorithm on preemptive uniprocessors, for all sequences of independent jobs. This means that the terms "feasibility" and "EDF-schedulability" can be used interchangeably here.

Theorem I.2 ([2])**.** *A task system T is feasible if and only if it is EDF-schedulable.* □

A related workload model is that of (strictly) periodic tasks. A periodic task system is a multiset of quadruples $(s, e, d, p) \in \mathbb{N} \times \mathbb{N}^3_+$. The difference to the sporadic task model is that the release times of two consecutive jobs from task (s, e, d, p) must be exactly p time units apart, and the release time of the first job is fixed at time point s. A periodic task system is *synchronous* if $s = 0$ for all tasks (s, e, d, p), and *asynchronous* otherwise. It is known [3] that feasibility testing of sporadic tasks is equally hard as that of synchronous periodic tasks, which means that the terms "sporadic" and "synchronous periodic" can be used interchangeably for the results in this paper as well.

Theorem I.3 ([3])**.** *A sporadic task system T is feasible if and only if the synchronous periodic task system $\{(0, e, d, p) \mid (e, d, p) \in \mathsf{T}\}$ is feasible.* □

A classic result [4], [5] is that the feasibility problem for asynchronous periodic task systems with constrained deadlines is strongly coNP-complete[1], even if restricted to instances where the utilization is bounded from above by a constant c, such that $0 < c < 1$. However, Baruah et al. [5], [3] found a pseudo-polynomial time algorithm for the special case of synchronous periodic (or sporadic) task systems with such a priori bounded utilization.

Except for the existence of this pseudo-polynomial time algorithm, the complexity of the sporadic feasibility problem remained open for many years. It was listed as one of the "most important open algorithmic problems in real-time scheduling" by Baruah and Pruhs [6]. A few years ago, Eisenbrand and Rothvoß [7] showed that the general case (with unbounded utilization) is weakly coNP-complete. This was recently strengthened by Ekberg and Yi [8], who showed that the general case is strongly coNP-complete.

[1]Recall that the problems that are NP- or coNP-complete in the strong sense do not permit even pseudo-polynomial time solutions, unless P = NP, while those that are NP- or coNP-complete in the weak (or ordinary) sense might have such solutions.

Theorem I.4 ([8]). *Deciding whether a sporadic task system with constrained deadlines is feasible on a preemptive uniprocessor is* coNP*-complete in the strong sense.* □

For the case with bounded utilization, it was still unknown whether a polynomial-time solution existed, and Eisenbrand and Rothvoß [7] conjectured that it did. In this paper, however, we show that for any choice of the constant c, such that $0 < c < 1$, the feasibility problem for sporadic task systems restricted to instances with utilization at most c is weakly coNP-complete, and thus that no polynomial-time solutions exist for it, unless P = NP. As it is known that this problem has a pseudo-polynomial time solution, this is a complete classification. A summary of the complexity of the periodic feasibility problem is shown in Figure 1.

	General case	Utilization bounded by some constant c, such that $0 < c < 1$
Asynchronous periodic tasks	coNP-complete in the strong sense. [5]	coNP-complete in the strong sense. [5]
Synchronous periodic tasks (or sporadic)	coNP-complete in the strong sense. [8]	coNP-complete in the weak sense (from this work).

Fig. 1. Complexity classification of the feasibility problem of periodic tasks with constrained deadlines on preemptive uniprocessors.

Let FEASIBILITY denote the general case of the decision problem of whether a given sporadic task system with constrained deadlines is feasible on a preemptive uniprocessor. Let c-FEASIBILITY denote the same problem restricted to instances with utilization at most c. Our hardness proof consists of a polynomial transformation (a polynomial-time many-one reduction) from FEASIBILITY to c-FEASIBILITY for any constant c, such that $0 < c < 1$. The transformation produces some parameter values that grow exponentially in the size of the FEASIBILITY instance. Therefore, c-FEASIBILITY is only shown to be weakly coNP-hard (as expected, since a pseudo-polynomial time solution is known) despite the strong coNP-hardness of FEASIBILITY.

II. PRELIMINARIES

In the hardness proof in the next section we will make use of the following theorem due to Baruah et al. [3]

Theorem II.1 ([3]). *A sporadic task system* T *with constrained deadlines is feasible on a preemptive uniprocessor if and only if* $\mathrm{U}(\mathsf{T}) \leqslant 1$ *and*

$$\forall \ell \in \{0, 1, \ldots, B\}, \quad \mathrm{dbf}(\mathsf{T}, \ell) \leqslant \ell, \tag{2}$$

where $B = \mathcal{P}(\mathsf{T}) + \max\{d \mid (e, d, p) \in \mathsf{T}\}$*, and where*

$$\mathrm{dbf}(\mathsf{T}, \ell) \stackrel{\text{def}}{=} \sum_{(e,d,p)\in\mathsf{T}} \left(\left\lfloor \frac{\ell - d}{p} \right\rfloor + 1 \right) e \tag{3}$$

is the demand bound function *of* T *in interval lengths* ℓ*, and*

$$\mathcal{P}(\mathsf{T}) \stackrel{\text{def}}{=} \mathrm{lcm}\{p \mid (e, d, p) \in \mathsf{T}\} \tag{4}$$

is T*'s* hyper-period. □

For our purposes it will be helpful to reformulate Eq. (2) in the above theorem in terms of *slack* and slightly tighten the upper bound B used for the values of ℓ.[2] First we prove a simple lemma that will also be useful later on.

Lemma II.2. *If* T *is a sporadic task system with constrained deadlines and* $k \in \mathbb{N}$*, then*

$$\mathrm{dbf}(\mathsf{T}, k\mathcal{P}(\mathsf{T}) + \ell) = k\mathcal{P}(\mathsf{T})\mathrm{U}(\mathsf{T}) + \mathrm{dbf}(\mathsf{T}, \ell).$$

Proof: From the fact that each of the periods divides the hyper-period, we get

$$\begin{aligned}
\mathrm{dbf}(\mathsf{T}, k\mathcal{P}(\mathsf{T}) + \ell) &= \sum_{(e,d,p)\in\mathsf{T}} \left(\left\lfloor \frac{k\mathcal{P}(\mathsf{T}) + \ell - d}{p} \right\rfloor + 1 \right) e \\
&= \sum_{(e,d,p)\in\mathsf{T}} \left(\frac{k\mathcal{P}(\mathsf{T})}{p} + \left\lfloor \frac{\ell - d}{p} \right\rfloor + 1 \right) e \\
&= \sum_{(e,d,p)\in\mathsf{T}} \left(\frac{ek\mathcal{P}(\mathsf{T})}{p} + \left(\left\lfloor \frac{\ell - d}{p} \right\rfloor + 1 \right) e \right) \\
&= k\mathcal{P}(\mathsf{T})\mathrm{U}(\mathsf{T}) + \mathrm{dbf}(\mathsf{T}, \ell).
\end{aligned}$$

■

As we reformulate Theorem II.1, we use this lemma to tighten the bound on the values of ℓ that we need to consider.

Corollary II.3. *A sporadic task system* T *with constrained deadlines is feasible on a preemptive uniprocessor if and only if* $\mathrm{U}(\mathsf{T}) \leqslant 1$ *and*

$$\forall \ell \in \{0, 1, \ldots, \mathcal{P}(\mathsf{T}) - 1\}, \quad \mathrm{slack}(\mathsf{T}, \ell) \geqslant 0, \tag{5}$$

where

$$\mathrm{slack}(\mathsf{T}, \ell) \stackrel{\text{def}}{=} \ell - \mathrm{dbf}(\mathsf{T}, \ell). \tag{6}$$

Proof: To see that the smaller range of values for ℓ considered in Eq. (5) is sufficient, note that from Lemma II.2 and $\mathrm{U}(\mathsf{T}) \leqslant 1$ we have, for any $k \in \mathbb{N}$,

$$\begin{aligned}
\mathrm{dbf}(\mathsf{T}, k\mathcal{P}(\mathsf{T}) + \ell) &= k\mathcal{P}(\mathsf{T})\mathrm{U}(\mathsf{T}) + \mathrm{dbf}(\mathsf{T}, \ell) \\
&\leqslant k\mathcal{P}(\mathsf{T}) + \mathrm{dbf}(\mathsf{T}, \ell).
\end{aligned}$$

Therefore,

$$\begin{aligned}
\mathrm{slack}(\mathsf{T}, k\mathcal{P}(\mathsf{T}) + \ell) &= k\mathcal{P}(\mathsf{T}) + \ell - \mathrm{dbf}(\mathsf{T}, k\mathcal{P}(\mathsf{T}) + \ell) \\
&\geqslant k\mathcal{P}(\mathsf{T}) + \ell - k\mathcal{P}(\mathsf{T}) - \mathrm{dbf}(\mathsf{T}, \ell) \\
&= \mathrm{slack}(\mathsf{T}, \ell)
\end{aligned}$$

and so if $\mathrm{slack}(\mathsf{T}, \ell) < 0$ holds for any $\ell \in \mathbb{N}$, then it must be the case that $\mathrm{slack}(\mathsf{T}, \ell \bmod \mathcal{P}(\mathsf{T})) < 0$ also holds. ■

[2] In [3], the considered tasks could have relative deadlines larger than their periods, which is why the larger value B is used there.

III. Reducing Feasibility to c-Feasibility

In this section we describe a polynomial transformation from Feasibility to c-Feasibility, for any given constant c, such that $0 < c < 1$. We start by providing a high-level picture of the transformation in Section III-A, and follow with the details and proofs in Section III-B.

A. An overview

For the transformation we take an arbitrary sporadic task system T_1 with constrained deadlines and compute a new task system T_c, also with constrained deadlines, such that

- $\mathrm{U}(\mathsf{T}_c) \leqslant c$ and
- T_c is feasible if and only if T_1 is feasible.

Note that if $\mathrm{U}(\mathsf{T}_1) > 1$, then T_1 is infeasible and the transformation is trivially computable by producing any infeasible T_c, for example $\mathsf{T}_c = \{(1, 1, \lceil 2/c \rceil), (1, 1, \lceil 2/c \rceil)\}$. In the remainder of this paper we assume $\mathrm{U}(\mathsf{T}_1) \leqslant 1$, without loss of generality.

The construction of T_c follows these four high-level steps.

Step 1: Add a filler task to T_1, resulting in task system T_{fill}, without affecting feasibility. The purpose is to ensure that $\mathrm{slack}(\mathsf{T}_{\text{fill}}, \ell)$ exactly repeats every hyper-period.

Step 2: Scale the deadlines and periods of T_{fill} uniformly with a given constant factor σ, producing task system T_σ. The purpose is to lower utilization while keeping a simple relation between $\mathrm{slack}(\mathsf{T}_{\text{fill}}, \ell)$ and $\mathrm{slack}(\mathsf{T}_\sigma, \ell)$.

Step 3: Construct a "boosting" task system T_{b}, which is designed to fill up the extra slack created by the scaling in step 2 at certain sparsely selected points ℓ. The purpose is to effectively negate the scaling done in step 2 without increasing the utilization too much. The extra slack is filled up at points that are offset by different multiples of T_σ's hyper-period. This is possible thanks to the repetitive nature of $\mathrm{slack}(\mathsf{T}_\sigma, \ell)$ that we ensured in step 1.

Step 4: Take T_c as the union of T_σ and T_{b}. Even though $\mathrm{U}(\mathsf{T}_c) \leqslant c$, we will have that there exists some $\ell \in \mathbb{N}$, such that $\mathrm{slack}(\mathsf{T}_1, \ell) < 0$ if and only if there exists some $\ell' \in \mathbb{N}$, such that $\mathrm{slack}(\mathsf{T}_c, \ell') < 0$.

B. The transformation

Here we describe the transformation in detail and prove a number of lemmas along the way.

► **Step 1:** *Ensuring the repetitiveness of the* slack *function:* In the first step we simply add a filler task τ_{fill} to T_1, if needed, to create a new task system T_{fill} with a utilization of 1. Figure 2 illustrates this step. Let

$$\mathsf{T}_{\text{fill}} \stackrel{\text{def}}{=} \begin{cases} \mathsf{T}_1 \uplus \{\tau_{\text{fill}}\}, & \text{if } \mathrm{U}(\mathsf{T}_1) < 1, \\ \mathsf{T}_1, & \text{otherwise,} \end{cases} \tag{7}$$

where $\tau_{\text{fill}} \stackrel{\text{def}}{=} (\mathcal{P}(\mathsf{T}_1) - \mathrm{dbf}(\mathsf{T}_1, \mathcal{P}(\mathsf{T}_1)), \mathcal{P}(\mathsf{T}_1), \mathcal{P}(\mathsf{T}_1))$ and where $\uplus$ is the multiset union operator.

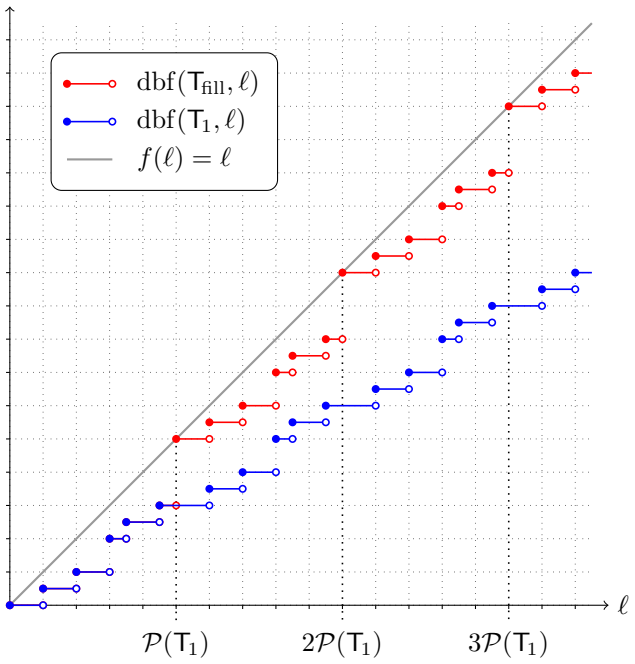

Fig. 2. The demand bound functions of a small example task system T_1 and the corresponding T_{fill}. Note that the difference between the diagonal and $\mathrm{dbf}(\mathsf{T}_{\text{fill}}, \ell)$, which is $\mathrm{slack}(\mathsf{T}_{\text{fill}}, \ell)$, repeats every hyper-period.

Lemma III.1. $\mathrm{U}(\mathsf{T}_{\text{fill}}) = 1$.

Proof: If $\mathrm{U}(\mathsf{T}_1) = 1$, the lemma follows directly. Assume instead $\mathrm{U}(\mathsf{T}_1) < 1$ and note that

$$\begin{aligned} \mathrm{U}(\{\tau_{\text{fill}}\}) &= \frac{\mathcal{P}(\mathsf{T}_1) - \mathrm{dbf}(\mathsf{T}_1, \mathcal{P}(\mathsf{T}_1))}{\mathcal{P}(\mathsf{T}_1)} \\ &= 1 - \frac{\sum_{(e,d,p)\in\mathsf{T}_1} \left(\left\lfloor \frac{\mathcal{P}(\mathsf{T}_1) - d}{p} \right\rfloor + 1 \right) e}{\mathcal{P}(\mathsf{T}_1)} \\ &= 1 - \frac{\sum_{(e,d,p)\in\mathsf{T}_1} \left(\frac{e\mathcal{P}(\mathsf{T}_1)}{p} + \left(\left\lfloor \frac{-d}{p} \right\rfloor + 1 \right) e \right)}{\mathcal{P}(\mathsf{T}_1)} \\ &\stackrel{(*)}{=} 1 - \mathrm{U}(\mathsf{T}_1), \end{aligned}$$

where for $(*)$ we used the fact that $\lfloor -d/p \rfloor = -1$. Clearly, $\mathrm{U}(\mathsf{T}_{\text{fill}}) = \mathrm{U}(\mathsf{T}_1) + \mathrm{U}(\{\tau_{\text{fill}}\}) = 1$. ■

Now, the property of T_{fill} that will be useful for our transformation is captured by the next lemma.

Lemma III.2. *For any* $\ell \in \{0, 1, \ldots, \mathcal{P}(\mathsf{T}_1) - 1\}$ *and* $k \in \mathbb{N}$,

$$\mathrm{slack}(\mathsf{T}_{\text{fill}}, k\mathcal{P}(\mathsf{T}_1) + \ell) = \mathrm{slack}(\mathsf{T}_1, \ell).$$

Proof: Note that $\mathcal{P}(\mathsf{T}_{\text{fill}}) = \mathcal{P}(\mathsf{T}_1)$, and therefore it

follows from Lemmas II.2 and III.1 that

$$\begin{aligned}
&\mathrm{slack}(\mathsf{T}_{\text{fill}}, k\mathcal{P}(\mathsf{T}_1) + \ell) \\
&\qquad = k\mathcal{P}(\mathsf{T}_1) + \ell - \mathrm{dbf}(\mathsf{T}_{\text{fill}}, k\mathcal{P}(\mathsf{T}_1) + \ell) \\
&\qquad = k\mathcal{P}(\mathsf{T}_1) + \ell - (k\mathcal{P}(\mathsf{T}_1)\mathrm{U}(\mathsf{T}_{\text{fill}}) + \mathrm{dbf}(\mathsf{T}_{\text{fill}}, \ell)) \\
&\qquad = \ell - \mathrm{dbf}(\mathsf{T}_{\text{fill}}, \ell) \\
&\qquad \overset{(*)}{=} \ell - \mathrm{dbf}(\mathsf{T}_1, \ell) \\
&\qquad = \mathrm{slack}(\mathsf{T}_1, \ell),
\end{aligned}$$

where for $(*)$ we have $\mathrm{dbf}(\mathsf{T}_{\text{fill}}, \ell) = \mathrm{dbf}(\mathsf{T}_1, \ell)$ because T_{fill} and T_1 at most differ on one task $\tau_{\text{fill}} = (e, d, p)$, for which $d = \mathcal{P}(\mathsf{T}_1) > \ell$ and therefore $\mathrm{dbf}(\{\tau_{\text{fill}}\}, \ell) = 0$. ■

► **Step 2:** *Scaling down the utilization:* In the second step we apply a scaling factor to the relative deadlines and periods of all the tasks in T_{fill}. Figure 3 serves as an illustration. Let

$$\sigma \overset{\text{def}}{=} \left\lceil \frac{2}{c} \right\rceil \tag{8}$$

be the constant scaling factor, where c is the constant defining the set of problems c-FEASIBILITY. Then let

$$\mathsf{T}_\sigma \overset{\text{def}}{=} \{(e, \sigma d, \sigma p) \mid (e, d, p) \in \mathsf{T}_{\text{fill}}\} \tag{9}$$

be the scaled task system. The utilization of T_σ is now bounded by half of c.

Lemma III.3. $\mathrm{U}(\mathsf{T}_\sigma) \leqslant c/2$.

Proof: Directly from Lemma III.1, Eq. (8) and (9). ■

Because of the uniform way in which T_σ was scaled, we have the following lemma.

Lemma III.4. *For any* $\ell \in \mathbb{N}$,

$$\mathrm{slack}(\mathsf{T}_\sigma, \sigma\ell) \;=\; \mathrm{slack}(\mathsf{T}_{\text{fill}}, \ell) + (\sigma - 1)\ell.$$

Proof: From the definition of T_σ we get

$$\begin{aligned}
\mathrm{slack}(\mathsf{T}_\sigma, \sigma\ell) &= \sigma\ell - \mathrm{dbf}(\mathsf{T}_\sigma, \sigma\ell) \\
&= \sigma\ell - \sum_{(e,d,p)\in\mathsf{T}_\sigma} \left(\left\lfloor \frac{\sigma\ell - d}{p} \right\rfloor + 1 \right) e \\
&= \sigma\ell - \sum_{(e,d,p)\in\mathsf{T}_{\text{fill}}} \left(\left\lfloor \frac{\sigma\ell - \sigma d}{\sigma p} \right\rfloor + 1 \right) e \\
&= \sigma\ell - \sum_{(e,d,p)\in\mathsf{T}_{\text{fill}}} \left(\left\lfloor \frac{\ell - d}{p} \right\rfloor + 1 \right) e \\
&= \sigma\ell - \mathrm{dbf}(\mathsf{T}_{\text{fill}}, \ell) \\
&= \mathrm{slack}(\mathsf{T}_{\text{fill}}, \ell) + (\sigma - 1)\ell.
\end{aligned}$$ ■

► **Step 3:** *Generating a boosting task system:* The task system T_σ created in the last step clearly does not preserve feasibility with regards to T_{fill} because of the extra slack introduced by the scaling of parameters. In this step we will craft a task system T_{b}, designed to effectively negate this extra slack. The main challenge is in doing so while using only polynomially many tasks and keeping the utilization low enough. This step is more involved than the previous steps and also a bit harder to visualize, but Figure 4 illustrates some of the key concepts. The boosting task system will contain β different tasks that are referred to using their indices,

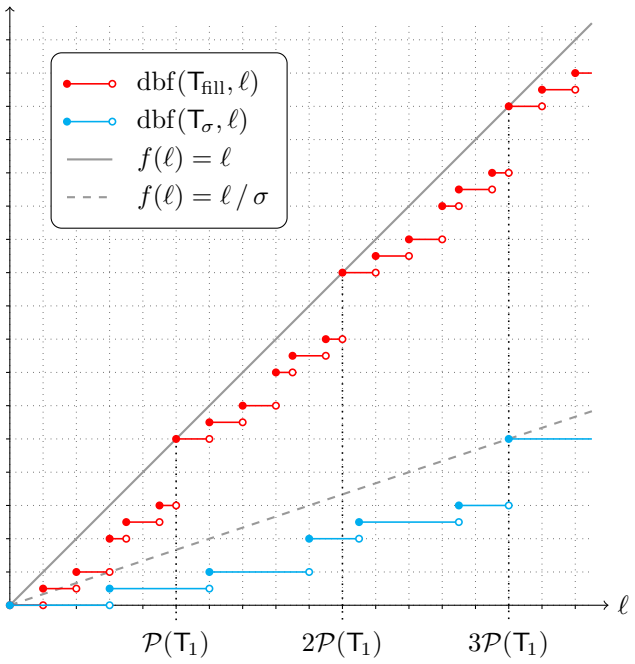

Fig. 3. The scaled task system T_σ has essentially the same pattern as T_{fill}, but with a lower utilization. Here we have $\sigma = 3$.

$$\mathsf{T}_{\text{b}} \overset{\text{def}}{=} \{\tau_0, \ldots, \tau_{\beta-1}\}, \tag{10}$$

where

$$\beta \overset{\text{def}}{=} \lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil . \tag{11}$$

For each boosting task τ_i, let $\tau_i \overset{\text{def}}{=} (e_i, d_i, p_i)$, where

$$d_i \overset{\text{def}}{=} (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^i \sigma, \tag{12}$$

$$p_i \overset{\text{def}}{=} (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^{i+1} \sigma, \tag{13}$$

$$e_i \overset{\text{def}}{=} \left(\frac{\sigma - 1}{\sigma} - \sum_{j=0}^{i-1} \frac{e_j}{p_j} \right) d_i. \tag{14}$$

The first thing we need to show is that the boosting task system, as defined above, consists of valid constrained-deadline sporadic tasks. The next lemma establishes this and some slightly stronger constraints on the parameters.

Lemma III.5. *For all* $\tau_i \in \mathsf{T}_{\text{b}}$, *we have*

$$(e_i, d_i, p_i) \in \mathbb{N}_+^3 \text{ and } e_i < d_i < p_i.$$

Proof: It follows directly from Eq. (12) and (13) that for any $\tau_i \in \mathsf{T}_{\text{b}}$, we have $d_i, p_i \in \mathbb{N}_+$ and $d_i < p_i$. What remains to be shown is that

$$e_i \in \mathbb{N}_+ \text{ and } e_i < d_i.$$

We use strong induction over i to prove this statement for all $\tau_i \in \mathsf{T}_\mathrm{b}$. As there is nothing to show for $i \geqslant \beta$, we assume $i < \beta$ in the following.

For the base case, note that

$$\begin{aligned} e_0 &= \left(\frac{\sigma-1}{\sigma}\right) d_0 \\ &= \left(\frac{\sigma-1}{\sigma}\right) \sigma \\ &= \sigma - 1. \end{aligned}$$

From Eq. (8) it is clear that $\sigma \geqslant 3$, and therefore $\sigma - 1 \in \mathbb{N}_+$ and $e_0 = \sigma - 1 < \sigma = d_0$.

We split the induction step into three parts, showing that $e_i \in \mathbb{Z}$, that $e_i > 0$ and that $e_i < d_i$, respectively.

Part 1 ($e_i \in \mathbb{Z}$): First, we rewrite e_i in a form without fractions.

$$\begin{aligned} e_i &= \left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{i-1} \frac{e_j}{p_j}\right) d_i \\ &= (\sigma-1)(\sigma\mathcal{P}(\mathsf{T}_1)+2)^i - \sum_{j=0}^{i-1} e_j \frac{(\sigma\mathcal{P}(\mathsf{T}_1)+2)^i}{(\sigma\mathcal{P}(\mathsf{T}_1)+2)^{j+1}} \\ &= (\sigma-1)(\sigma\mathcal{P}(\mathsf{T}_1)+2)^i - \sum_{j=0}^{i-1} e_j (\sigma\mathcal{P}(\mathsf{T}_1)+2)^{i-j-1} \end{aligned}$$

Now, from the induction hypothesis we have $e_j \in \mathbb{N}_+$ for $j < i$, and therefore the rewritten form of e_i consists of only additions, subtractions and (repeated) multiplications of integers. As $\mathbb{Z}$ are closed under these operations, it follows that $e_i \in \mathbb{Z}$.

Part 2 ($e_i > 0$): We have

$$\begin{aligned} e_i &= \left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{i-1} \frac{e_j}{p_j}\right) d_i \\ &\overset{(*)}{\geqslant} \left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{i-1} \frac{d_j}{p_j}\right) d_i \\ &= \left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{i-1} \frac{1}{(\sigma\mathcal{P}(\mathsf{T}_1)+2)}\right) d_i \\ &= \left(\frac{\sigma-1}{\sigma} - \frac{i}{(\sigma\mathcal{P}(\mathsf{T}_1)+2)}\right) d_i \\ &\overset{(**)}{>} \left(\frac{\sigma-1}{\sigma} - \frac{\mathcal{P}(\mathsf{T}_1)}{(\sigma\mathcal{P}(\mathsf{T}_1)+2)}\right) d_i \\ &> \left(\frac{\sigma-1}{\sigma} - \frac{1}{\sigma}\right) d_i \\ &\geqslant \left(\frac{1}{\sigma}\right) d_i \\ &> 0, \end{aligned}$$

where for $(*)$ we used $e_j < d_j$ from the induction hypothesis and for $(**)$ we used $i < \beta = \lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil < \mathcal{P}(\mathsf{T}_1)$.

Part 3 ($e_i < d_i$): Lastly, note that

$$\begin{aligned} e_i &= \left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{i-1} \frac{e_j}{p_j}\right) d_i \\ &< \left(1 - \sum_{j=0}^{i-1} \frac{e_j}{p_j}\right) d_i \\ &\overset{(*)}{\leqslant} d_i, \end{aligned}$$

where for $(*)$ we used $e_j \in \mathbb{N}_+$ for $j < i$ from the induction hypothesis.

Taken together, the three parts imply $e_i \in \mathbb{N}_+$ and $e_i < d_i$, which concludes the proof. ∎

Having shown that T_b is a valid sporadic task system, we can show that it has the properties we seek. We start by showing that the utilization of T_b is low, namely bounded by half of c.

Lemma III.6. $\mathrm{U}(\mathsf{T}_\mathrm{b}) < c/2$.

Proof: Consider any $\tau_i \in \mathsf{T}_\mathrm{b}$ and note that

$$\begin{aligned} \mathrm{U}(\{\tau_i\}) &= e_i \,/\, p_i \\ &< d_i \,/\, p_i \\ &< 1 \,/\, (\sigma\mathcal{P}(\mathsf{T}_1)). \end{aligned}$$

Now, because $\mathcal{P}(\mathsf{T}_1) \in \mathbb{N}_+$, we have $\lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil < \mathcal{P}(\mathsf{T}_1)$ and therefore

$$\begin{aligned} \mathrm{U}(\mathsf{T}_\mathrm{b}) &< \beta \,/\, (\sigma\mathcal{P}(\mathsf{T}_1)) \\ &< 1 \,/\, \sigma \\ &\leqslant c \,/\, 2. \end{aligned}$$

∎

Now we study the demand of T_b to see that it provides the right amount of it to selectively fill up the extra slack that was created for T_σ by the scaling in step 2. With the next lemma we ensure that the demand of T_b never fills up more than this extra slack.

Lemma III.7. *For all $\ell \in \mathbb{N}$,*

$$\mathrm{dbf}(\mathsf{T}_\mathrm{b}, \ell) \leqslant \frac{\sigma-1}{\sigma}\ell.$$

Proof: First, let T_b^m denote the subset consisting of the smallest $\min(m, \beta)$ tasks in T_b,

$$\mathsf{T}_\mathrm{b}^m \overset{\text{def}}{=} \{\tau_i \in \mathsf{T}_\mathrm{b} \mid i < m\}.$$

Then we use induction over m to prove that

$$\forall \ell \in \mathbb{N},\ \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m, \ell) \leqslant \frac{\sigma-1}{\sigma}\ell,$$

for all $m \in \mathbb{N}_+$. Because $\mathrm{dbf}(\mathsf{T}_\mathrm{b}^m, \ell) = \mathrm{dbf}(\mathsf{T}_\mathrm{b}, \ell)$ for $m \geqslant \beta$, the lemma will follow.

The base case holds as we have $\mathsf{T}_\mathrm{b}^1 = \{\tau_0\}$ and

$$\begin{aligned}\mathrm{dbf}(\{\tau_0\},\ell) &= \left(\left\lfloor\frac{\ell - d_0}{p_0}\right\rfloor + 1\right) e_0 \\ &\leqslant \left(\left\lfloor\frac{\ell - d_0}{d_0}\right\rfloor + 1\right) e_0 \\ &= \left\lfloor\frac{\ell}{d_0}\right\rfloor e_0 \\ &\leqslant \frac{e_0}{d_0}\ell \\ &= \frac{\sigma-1}{\sigma}\ell.\end{aligned}$$

For the induction step we need to show for all $m \in \mathbb{N}_+$ that

$$\begin{aligned}&\forall \ell \in \mathbb{N},\, \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m,\ell) \leqslant \frac{\sigma-1}{\sigma}\ell \implies \\ &\forall \ell \in \mathbb{N},\, \mathrm{dbf}(\mathsf{T}_\mathrm{b}^{m+1},\ell) \leqslant \frac{\sigma-1}{\sigma}\ell.\end{aligned}$$

The induction step trivially holds for $m \geqslant \beta$. Assume instead $m < \beta$ and note that $\mathsf{T}_\mathrm{b}^{m+1} \setminus \mathsf{T}_\mathrm{b}^m = \{\tau_m\}$. Take any $\ell \in \mathbb{N}$ and write $\ell = kd_m + \ell'$ for some $k \in \mathbb{N}$ and $\ell' < d_m$. Then,

$$\begin{aligned}\mathrm{dbf}(\{\tau_m\},\ell) &= \mathrm{dbf}(\{\tau_m\}, kd_m + \ell') \\ &= \left(\left\lfloor\frac{kd_m + \ell' - d_m}{p_m}\right\rfloor + 1\right) e_m \\ &\overset{(*)}{\leqslant} \left(\left\lfloor\frac{kd_m}{p_m}\right\rfloor + 1\right) e_m \\ &\overset{(**)}{\leqslant} ke_m \\ &= k\left(\frac{\sigma-1}{\sigma} - \sum_{j=0}^{m-1}\frac{e_j}{p_j}\right) d_m \\ &= kd_m\frac{\sigma-1}{\sigma} - kd_m\mathrm{U}(\mathsf{T}_\mathrm{b}^m),\end{aligned}$$

where for $(*)$ we used $\ell' < d_m$ and for $(**)$ we used $d_m < p_m$. Now, from Eq. (12) and (13) it is clear that $d_m = p_{m-1} = \mathcal{P}(\mathsf{T}_\mathrm{b}^m)$. Using Lemma II.2 and the induction hypothesis we then get

$$\begin{aligned}\mathrm{dbf}(\mathsf{T}_\mathrm{b}^m,\ell) &= \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m, kd_m + \ell') \\ &= kd_m\mathrm{U}(\mathsf{T}_\mathrm{b}^m) + \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m,\ell') \\ &\leqslant kd_m\mathrm{U}(\mathsf{T}_\mathrm{b}^m) + \frac{\sigma-1}{\sigma}\ell'.\end{aligned}$$

Putting the above together, we get

$$\begin{aligned}\mathrm{dbf}(\mathsf{T}_\mathrm{b}^{m+1},\ell) &= \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m,\ell) + \mathrm{dbf}(\{\tau_m\},\ell) \\ &\leqslant \frac{\sigma-1}{\sigma}(kd_m + \ell') \\ &= \frac{\sigma-1}{\sigma}\ell.\end{aligned}$$

To conclude, we have shown that for all $m \in \mathbb{N}_+$,

$$\forall \ell \in \mathbb{N},\, \mathrm{dbf}(\mathsf{T}_\mathrm{b}^m,\ell) \leqslant \frac{\sigma-1}{\sigma}\ell.$$

In particular, it holds for $m = \beta$ and the lemma follows. ∎

We now consider the points at which T_b provides exactly the right amount of demand to negate the scaling done for T_σ in step 2. We call these the *boost points* of T_b and denote them $\mathcal{B}(\mathsf{T}_\mathrm{b})$.

$$\mathcal{B}(\mathsf{T}_\mathrm{b}) \overset{\text{def}}{=} \left\{\ell \in \mathbb{N} \mid \mathrm{dbf}(\mathsf{T}_\mathrm{b},\ell) = \frac{\sigma-1}{\sigma}\ell\right\} \tag{15}$$

The following lemma lets us identify boost points.

Lemma III.8. *If* $\dot{\mathsf{T}} \subseteq \mathsf{T}_\mathrm{b}$*, then*

$$\sum_{\tau_i \in \dot{\mathsf{T}}} d_i \;\in\; \mathcal{B}(\mathsf{T}_\mathrm{b}).$$

Proof: Consider the relation $(2^{\mathsf{T}_\mathrm{b}}, \subset)$, i.e., the power set of T_b ordered by strict inclusion, and note that it is well-founded. We prove the lemma using well-founded induction on this relation. To do so we must show that the following implication holds for any $\dot{\mathsf{T}}$, such that $\dot{\mathsf{T}} \subseteq \mathsf{T}_\mathrm{b}$.

$$\left(\forall \ddot{\mathsf{T}},\ \text{s.t. } \ddot{\mathsf{T}} \subset \dot{\mathsf{T}},\ \sum_{\tau_i \in \ddot{\mathsf{T}}} d_i \in \mathcal{B}(\mathsf{T}_\mathrm{b})\right) \implies \sum_{\tau_i \in \dot{\mathsf{T}}} d_i \in \mathcal{B}(\mathsf{T}_\mathrm{b})$$

We split the proof into two cases.

Case 1 ($\dot{\mathsf{T}} = \emptyset$): Clearly, $\sum_{\tau_i \in \emptyset} d_i = 0$ and from Eq. (15) we have $0 \in \mathcal{B}(\mathsf{T}_\mathrm{b})$.

Case 2 ($\dot{\mathsf{T}} \neq \emptyset$): Now, $\dot{\mathsf{T}}$ must contain a "largest" task τ_α, where

$$\alpha \overset{\text{def}}{=} \max\{i \mid \tau_i \in \dot{\mathsf{T}}\}$$

is the largest index of the tasks in $\dot{\mathsf{T}}$. Let

$$\begin{aligned}\ddot{\mathsf{T}} &\overset{\text{def}}{=} \dot{\mathsf{T}} \setminus \{\tau_\alpha\}, \\ \ddot{\ell} &\overset{\text{def}}{=} \sum_{\tau_i \in \ddot{\mathsf{T}}} d_i, \\ \dot{\ell} &\overset{\text{def}}{=} d_\alpha + \ddot{\ell}.\end{aligned}$$

Since $\ddot{\mathsf{T}} \subset \dot{\mathsf{T}}$, we know that $\ddot{\ell} \in \mathcal{B}(\mathsf{T}_\mathrm{b})$ holds from the induction hypothesis. To conclude the proof we must show that $\dot{\ell} \in \mathcal{B}(\mathsf{T}_\mathrm{b})$ holds as well.

Consider now the sets of boosting tasks with indices smaller than α and at least as large as α, respectively.

$$\begin{aligned}\mathsf{T}_{\alpha\downarrow} &\overset{\text{def}}{=} \{\tau_i \in \mathsf{T}_\mathrm{b} \mid i < \alpha\} \\ \mathsf{T}_{\alpha\uparrow} &\overset{\text{def}}{=} \{\tau_i \in \mathsf{T}_\mathrm{b} \mid i \geqslant \alpha\}\end{aligned}$$

Because $\ddot{\mathsf{T}} \subseteq \mathsf{T}_{\alpha\downarrow}$ we have

$$\begin{aligned}\ddot{\ell} &\leqslant \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} d_i \\ &= (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^0\sigma \;+\; \cdots \;+\; (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^{\alpha-1}\sigma \\ &\overset{(*)}{=} \sigma\left(\frac{(\sigma\mathcal{P}(\mathsf{T}_1) + 2)^\alpha - 1}{\sigma\mathcal{P}(\mathsf{T}_1) + 2 - 1}\right) \\ &< (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^\alpha\sigma \\ &= d_\alpha,\end{aligned}$$

where for $(*)$ we used the closed form for the sum of a finite geometric series. It follows that for all $\tau_i \in \mathsf{T}_{\alpha\uparrow}$ we have $d_i \geqslant d_\alpha > \ddot{\ell}$ and therefore

$$\mathrm{dbf}(\mathsf{T}_{\alpha\uparrow}, \ddot{\ell}) \;=\; 0.$$

Using this and the induction hypothesis we get

$$\begin{aligned}\mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \ddot{\ell}) \;&=\; \mathrm{dbf}(\mathsf{T}_{\mathrm{b}}, \ddot{\ell}) - \mathrm{dbf}(\mathsf{T}_{\alpha\uparrow}, \ddot{\ell}) \\ &=\; \mathrm{dbf}(\mathsf{T}_{\mathrm{b}}, \ddot{\ell}) \\ &=\; \frac{\sigma-1}{\sigma}\ddot{\ell}.\end{aligned}$$

Now, from Eq. (12) and (13) we know that p_i divides d_α for all $\tau_i \in \mathsf{T}_{\alpha\downarrow}$. Therefore,

$$\begin{aligned}\mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \;&=\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, d_\alpha + \ddot{\ell}) \\ &=\; \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} \left(\left\lfloor \frac{d_\alpha + \ddot{\ell} - d_i}{p_i} \right\rfloor + 1 \right) e_i \\ &=\; \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} \left(\frac{d_\alpha}{p_i} + \left\lfloor \frac{\ddot{\ell} - d_i}{p_i} \right\rfloor + 1 \right) e_i \\ &=\; \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} \frac{e_i}{p_i} d_\alpha + \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \ddot{\ell}) \\ &=\; \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} \frac{e_i}{p_i} d_\alpha + \frac{\sigma-1}{\sigma}\ddot{\ell}\end{aligned}$$

Finally,

$$\begin{aligned}\mathrm{dbf}(\mathsf{T}_{\mathrm{b}}, \dot{\ell}) \;&=\; \mathrm{dbf}(\mathsf{T}_{\alpha\uparrow}, \dot{\ell}) \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &\geqslant\; \mathrm{dbf}(\{\tau_\alpha\}, \dot{\ell}) \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &\geqslant\; \mathrm{dbf}(\{\tau_\alpha\}, d_\alpha) \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &=\; e_\alpha \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &=\; \frac{\sigma-1}{\sigma} d_\alpha - \sum_{j=0}^{\alpha-1} \frac{e_j}{p_j} d_\alpha \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &=\; \frac{\sigma-1}{\sigma} d_\alpha - \sum_{\tau_i \in \mathsf{T}_{\alpha\downarrow}} \frac{e_i}{p_i} d_\alpha \;+\; \mathrm{dbf}(\mathsf{T}_{\alpha\downarrow}, \dot{\ell}) \\ &=\; \frac{\sigma-1}{\sigma}(d_\alpha + \ddot{\ell}) \\ &=\; \frac{\sigma-1}{\sigma}\dot{\ell}\end{aligned}$$

The inequalities above must be strict equalities based on Lemma III.7, and we conclude that $\dot{\ell} \in \mathcal{B}(\mathsf{T}_{\mathrm{b}})$. ∎

Now that we have a lemma for identifying boost points, we can show that $\mathcal{B}(\mathsf{T}_{\mathrm{b}})$ cover enough relevant points in $\mathbb{N}$.

Lemma III.9. *For all $\ell \in \{0, 1, \ldots, \mathcal{P}(\mathsf{T}_1) - 1\}$, there exists some $k \in \mathbb{N}$, such that*

$$\sigma(k\mathcal{P}(\mathsf{T}_1) + \ell) \;\in\; \mathcal{B}(\mathsf{T}_{\mathrm{b}}).$$

Proof: Take any $\ell \in \{0, 1, \ldots, \mathcal{P}(\mathsf{T}_1) - 1\}$ and note that ℓ can be represented in binary using $\beta = \lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil$ bits. Let

$$\mathsf{b}_{\beta-1} \cdots \mathsf{b}_1 \mathsf{b}_0$$

be this binary representation of ℓ, where b_0 is the least significant bit. Consider the set of boosting tasks with indices matching the indices of the positive bits in the binary representation of ℓ

$$\dot{\mathsf{T}} \;\stackrel{\text{def}}{=}\; \{\tau_i \in \mathsf{T}_{\mathrm{b}} \mid \mathsf{b}_i = 1\}.$$

For any $\tau_i \in \dot{\mathsf{T}}$ we can rewrite

$$\begin{aligned} d_i \;&=\; (\sigma\mathcal{P}(\mathsf{T}_1) + 2)^i \sigma \\ &\stackrel{(*)}{=}\; \left(\sum_{m=0}^{i} \binom{i}{m} \sigma^m \mathcal{P}(\mathsf{T}_1)^m 2^{i-m} \right) \sigma \\ &=\; \left(\sum_{m=1}^{i} \binom{i}{m} \sigma^m \mathcal{P}(\mathsf{T}_1)^m 2^{i-m} + 2^i \right) \sigma \\ &=\; \left(k_i \mathcal{P}(\mathsf{T}_1) + 2^i\right) \sigma, \end{aligned}$$

where

$$k_i = \sum_{m=1}^{i} \binom{i}{m} \sigma^m \mathcal{P}(\mathsf{T}_1)^{m-1} 2^{i-m}$$

and where for $(*)$ we used the binomial theorem. Let

$$k \;\stackrel{\text{def}}{=}\; \sum_{\tau_i \in \dot{\mathsf{T}}} k_i$$

and note that $k \in \mathbb{N}$. We then have

$$\begin{aligned} \sum_{\tau_i \in \dot{\mathsf{T}}} d_i \;&=\; \sum_{\tau_i \in \dot{\mathsf{T}}} (k_i \mathcal{P}(\mathsf{T}_1) + 2^i)\sigma \\ &=\; \sigma \left(\sum_{\tau_i \in \dot{\mathsf{T}}} k_i \mathcal{P}(\mathsf{T}_1) + \sum_{\tau_i \in \dot{\mathsf{T}}} 2^i \right) \\ &=\; \sigma \left(k\mathcal{P}(\mathsf{T}_1) + \sum \{2^i \mid \mathsf{b}_i = 1\} \right) \\ &=\; \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell). \end{aligned}$$

To conclude, we note that because $\dot{\mathsf{T}} \subseteq \mathsf{T}_{\mathrm{b}}$, Lemma III.8 gives us

$$\sum_{\tau_i \in \dot{\mathsf{T}}} d_i \;=\; \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell) \;\in\; \mathcal{B}(\mathsf{T}_{\mathrm{b}})$$

and the lemma follows. ∎

► **Step 4:** *Assembling the finished task system:* In the last step we put everything together. Let

$$\mathsf{T}_c \;\stackrel{\text{def}}{=}\; \mathsf{T}_\sigma \uplus \mathsf{T}_{\mathrm{b}}, \tag{16}$$

and note that T_c is an instance of c-FEASIBILITY.

Lemma III.10. $\mathrm{U}(\mathsf{T}_c) \;<\; c.$

Proof: Directly from Lemmas III.3 and III.6. ∎

We can now show that the transformation correctly preserves the feasibility of the original task system. We start by showing that T_c is feasible if T_1 is.

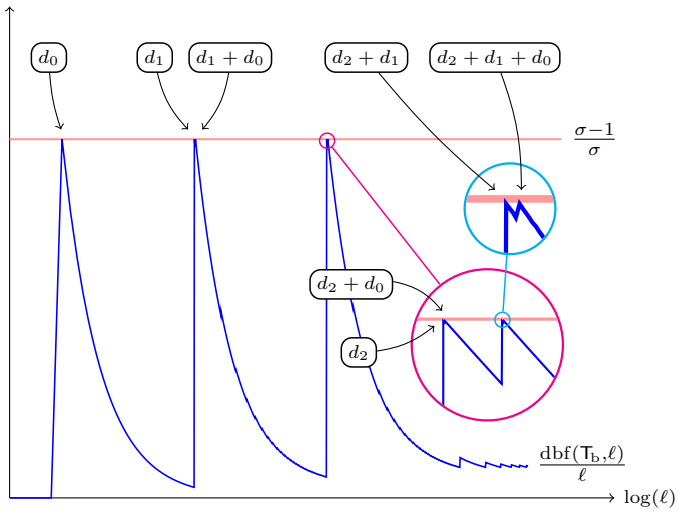

Fig. 4. A plot of $\mathrm{dbf}(\mathsf{T}_\mathrm{b}, \ell) \,/\, \ell$ for a small example T_b with $\beta = 3$. Note the logarithmic scale on the horizontal axis. The boost points are those points at which this function touches the line $(\sigma - 1) \,/\, \sigma$. Every task $\tau_i \in \mathsf{T}_\mathrm{b}$ adds a new "peak" to the plot, touching the line $(\sigma - 1) \,/\, \sigma$ at $\ell = d_i$. Immediately after, all previous peaks for tasks τ_j, with $j < i$, will repeat, as can be seen in the zoomed in portions of the plot. The number of boost points therefore grows exponentially in β. The labels show the value of ℓ at the peaks. Note also how the function converges towards $\mathrm{U}(\mathsf{T}_\mathrm{b})$ to the right.

Lemma III.11. *If T_1 is feasible, then T_c is also feasible.*

Proof: We prove the lemma by contradiction. Assume for this purpose the negation of the lemma's claim—that T_1 is feasible, but T_c is not. By Corollary II.3, there must then exist some $\ell \in \mathbb{N}$ such that $\mathrm{slack}(\mathsf{T}_c, \ell) < 0$. Since all $(e, d, p) \in \mathsf{T}_c$ have d and p that are multiples of σ, it is evident that there must also exist some $\ell' \in \mathbb{N}$ such that

$$\mathrm{slack}(\mathsf{T}_c, \sigma\ell') \;<\; 0.$$

Note that

$$\begin{aligned}\mathrm{slack}(\mathsf{T}_c, \sigma\ell') \;&=\; \sigma\ell' - \mathrm{dbf}(\mathsf{T}_c, \sigma\ell') \\ &=\; \sigma\ell' - \mathrm{dbf}(\mathsf{T}_\sigma, \sigma\ell') - \mathrm{dbf}(\mathsf{T}_\mathrm{b}, \sigma\ell') \\ &=\; \mathrm{slack}(\mathsf{T}_\sigma, \sigma\ell') - \mathrm{dbf}(\mathsf{T}_\mathrm{b}, \sigma\ell'). \qquad (17)\end{aligned}$$

From Lemma III.4 we have

$$\mathrm{slack}(\mathsf{T}_\sigma, \sigma\ell') \;=\; \mathrm{slack}(\mathsf{T}_\mathrm{fill}, \ell') + (\sigma - 1)\ell' \qquad (18)$$

and from Lemma III.7 we have

$$\mathrm{dbf}(\mathsf{T}_\mathrm{b}, \sigma\ell') \;\leqslant\; (\sigma - 1)\ell'. \qquad (19)$$

By combining Eq. (17), (18) and (19) and using Lemma III.2, we conclude that

$$\begin{aligned}\mathrm{slack}(\mathsf{T}_c, \sigma\ell') \;&\geqslant\; \mathrm{slack}(\mathsf{T}_\mathrm{fill}, \ell') \\ &=\; \mathrm{slack}(\mathsf{T}_1, \ell' \bmod \mathcal{P}(\mathsf{T}_1)).\end{aligned}$$

By assumption, however, T_1 is feasible and therefore

$$\mathrm{slack}(\mathsf{T}_c, \sigma\ell') \;\geqslant\; \mathrm{slack}(\mathsf{T}_1, \ell' \bmod \mathcal{P}(\mathsf{T}_1)) \;\geqslant\; 0.$$

The lemma follows from this contradiction. ∎

We now show the other direction.

Lemma III.12. *If T_1 is infeasible, then T_c is also infeasible.*

Proof: Assume that T_1 is infeasible, then by Corollary II.3 there exists an $\ell \in \{0, 1, \ldots, \mathcal{P}(\mathsf{T}_1) - 1\}$ such that

$$\mathrm{slack}(\mathsf{T}_1, \ell) \;<\; 0.$$

By Lemma III.9, there also exists some $k \in \mathbb{N}$ such that

$$\sigma(k\mathcal{P}(\mathsf{T}_1) + \ell) \;\in\; \mathcal{B}(\mathsf{T}_\mathrm{b}).$$

By the definition of $\mathcal{B}(\mathsf{T}_\mathrm{b})$, we therefore have

$$\mathrm{dbf}(\mathsf{T}_\mathrm{b}, \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell)) \;=\; (\sigma - 1)(k\mathcal{P}(\mathsf{T}_1) + \ell). \qquad (20)$$

Note also that from Lemma III.4 we have

$$\begin{aligned}&\mathrm{slack}(\mathsf{T}_\sigma, \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell)) \\ &\quad=\; \mathrm{slack}(\mathsf{T}_\mathrm{fill}, k\mathcal{P}(\mathsf{T}_1) + \ell) + (\sigma - 1)(k\mathcal{P}(\mathsf{T}_1) + \ell). \qquad (21)\end{aligned}$$

Using Eq. (20) and (21) and Lemma III.2 we conclude that

$$\begin{aligned}&\mathrm{slack}(\mathsf{T}_c, \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell)) \\ &\quad=\; \mathrm{slack}(\mathsf{T}_\sigma, \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell)) - \mathrm{dbf}(\mathsf{T}_\mathrm{b}, \sigma(k\mathcal{P}(\mathsf{T}_1) + \ell)) \\ &\quad=\; \mathrm{slack}(\mathsf{T}_\mathrm{fill}, k\mathcal{P}(\mathsf{T}_1) + \ell) \\ &\quad=\; \mathrm{slack}(\mathsf{T}_1, \ell) \\ &\quad<\; 0,\end{aligned}$$

and hence that T_c is infeasible by Corollary II.3. ∎

Finally, our main Theorem follows.

Theorem III.13. *The c-FEASIBILITY problem is (weakly) coNP-complete for any constant c such that $0 < c < 1$.*

Proof: First, note that we have described a transformation from FEASIBILITY to c-FEASIBILITY, which produces correct outputs according to Lemmas III.10, III.11 and III.12. Secondly, note that the constructed task system T_c has only polynomially many tasks,

$$|\mathsf{T}_1| + 1 + \lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil$$

at most. Also note that the largest parameter in T_c is $p_{\beta-1}$, which has a value of

$$(\sigma\mathcal{P}(\mathsf{T}_1) + 2)^{\lceil \log_2(\mathcal{P}(\mathsf{T}_1)) \rceil}\sigma,$$

which requires only polynomial space. The transformation is then trivially computable in polynomial time.

From Theorem I.4 we know that FEASIBILITY is coNP-complete. The polynomial transformation therefore proves the coNP-hardness of c-FEASIBILITY. Because c-FEASIBILITY is a special case of FEASIBILITY, we know that c-FEASIBILITY is also in coNP. The coNP-completeness of c-FEASIBILITY follows. ∎

IV. CONCLUSIONS

The sporadic task model is one of the basic formalisms for specifying real-time workload. The problem of deciding whether a given sporadic task system is schedulable on a

preemptive uniprocessor is of fundamental importance for real-time scheduling theory. The special case that is limited to task systems with utilization bounded by some constant is widely encountered in the literature, thanks to the existence of an algorithm that solves that problem in pseudo-polynomial time. A long-standing open question has been whether this special case also has a polynomial-time solution. We have shown that for any reasonable choice of the constant, the problem is coNP-complete and therefore cannot be solved in polynomial time, assuming $\mathsf{P} \neq \mathsf{NP}$.

ACKNOWLEDGMENT

We would like to thank the anonymous reviewers for their detailed and helpful comments.

REFERENCES

[1] 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.

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

[4] J. Y.-T. Leung and M. Merrill, "A note on preemptive scheduling of periodic, real-time tasks," *Information Processing Letters*, vol. 11, no. 3, pp. 115–118, 1980.

[5] S. K. Baruah, L. E. Rosier, and R. R. Howell, "Algorithms and complexity concerning the preemptive scheduling of periodic, real-time tasks on one processor," *Real-Time Systems*, vol. 2, no. 4, pp. 301–324, 1990.

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

[7] F. Eisenbrand and T. Rothvoß, "EDF-schedulability of synchronous periodic task systems is coNP-hard," in *Proceedings of the ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 2010, pp. 1029–1034.

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

$\boxtimes$