# Complexity of Uniprocessor Scheduling Analysis

Pontus Ekberg and Wang Yi

**Abstract** When designing a real-time system, a *schedulability problem* must be solved in order to show that it will meet all timing constraints at runtime. These are decision problems that, given a model of the system as input, answer whether all timing constraints will be met using a particular combination of scheduling algorithm and computer platform. Creating efficient algorithms for solving schedulability problems is a major focus of real-time systems research, but many of these problems are computationally intractable to varying degrees. Here we will review what is currently known about the computational complexity of schedulability problems for some common task models and scheduling algorithms on uniprocessors.

## Introduction

For a given combination of task model $\mathcal{M}$, scheduling algorithm $\mathcal{A}$ and computer platform $\mathcal{C}$, the corresponding *schedulability problem* is the following decision problem.

**Input:** A task set $\tau$ specified in task model $\mathcal{M}$.

**Output:** Yes, if $\mathcal{A}$ schedules $\tau$ on platform $\mathcal{C}$ in such a way that all timing constraints of $\tau$ are guaranteed to be met. No, otherwise.

For fixed $\mathcal{M}$ and $\mathcal{C}$, the corresponding *feasibility problem* instead asks whether $\tau$ is schedulable by any scheduling algorithm $\mathcal{A}$.

Pontus Ekberg
Department of Information Technology, Uppsala University, Box 337 SE-751 05 Uppsala Sweden
e-mail: pontus.ekberg@it.uu.se

Wang Yi
Department of Information Technology, Uppsala University, Box 337 SE-751 05 Uppsala Sweden
e-mail: yi@it.uu.se

In this chapter we will review what is currently known about the computational complexity of such problems, when $\mathcal{C}$ is a preemptive uniprocessor and $\mathcal{A}$ is either the Earliest Deadline First (EDF) or the Fixed-Priority (FP) scheduling algorithm. EDF and FP are, by far, the most popular and widely-studied scheduling algorithms in this context.

We will consider a number of task models $\mathcal{M}$, but will necessarily limit the scope to a subset of the wealth of task models that have been proposed in the literature. We choose to first take a closer look at the common basic task models: sporadic and periodic tasks. Then we consider more general task models that still generate simple independent jobs, such as the Generalized Multiframe (Baruah et al. 1999) and Digraph Real-Time (Stigge et al. 2011b) task models.

## Sporadic and periodic tasks

A *periodic task* $\tau_i$ is specified by a quadruple of positive integers $\tau_i = (O_i, C_i, D_i, T_i)$, where $O_i$ is its initial offset, $C_i$ its worst-case execution time, $D_i$ its relative deadline, and $T_i$ its period, respectively. Each task generates an unbounded sequence of *jobs* $\langle J_{i,1}, J_{i,2}, \ldots \rangle$, where each job is a basic unit of work given by a triple $(r, c, d)$ of release time $r$, worst-case execution time $c$, and absolute deadline $d$.

For a periodic task $\tau_i$, the $k$'th job $J_{i,k} = (r_{i,k}, c_{i,k}, d_{i,k})$ in its generated job sequence satisfies

- $r_{i,k} = O_i + (k-1)T_i$,
- $c_{i,k} = C_i$,
- $d_{i,k} = r_{i,k} + D_i$.

A periodic *task set* $\tau$ is a collection of periodic tasks $\tau = \{\tau_1, \ldots, \tau_n\}$. We say that $\tau$ is *synchronous* in the special case where $O_i = 0$ for all $\tau_i \in \tau$. To avoid any ambiguity we always say that it is *asynchronous* in the general case. For synchronous task sets we often omit the offset parameter $O$.

Closely related are *sporadic tasks*. Such a task $\tau_i$ is specified by a triple of positive integers $\tau_i = (C_i, D_i, T_i)$. A sporadic task non-deterministically generates any of an infinite number of distinct sequences of jobs, $\langle J_{i,1}, J_{i,2}, \ldots \rangle$, where the $k$'th job $J_{i,k} = (r_{i,k}, c_{i,k}, d_{i,k})$ satisfies

- $r_{i,k} \geq r_{i,k-1} + T_i$, if $k > 1$,
- $c_{i,k} = C_i$,
- $d_{i,k} = r_{i,k} + D_i$.

Further, all the above task models are commonly considered in the following special cases.

- If $D_i = T_i$ for all $\tau_i \in \tau$, then we say that $\tau$ has *implicit deadlines*.
- If $D_i \leq T_i$ for all $\tau_i \in \tau$, then we say that $\tau$ has *constrained deadlines*.

To avoid any ambiguity, we say that a task set has *arbitrary deadlines* when none of the above restrictions are enforced.

For any periodic or sporadic task set $\tau$, its *utilization* $\mathrm{U}(\tau)$ is defined as

$$\mathrm{U}(\tau) \stackrel{\text{def}}{=} \sum_{\tau_i \in \tau} \frac{C_i}{T_i}, \tag{1}$$

and its *hyper-period* $\mathcal{P}(\tau)$ as

$$\mathcal{P}(\tau) \stackrel{\text{def}}{=} \mathrm{lcm}\{T_i \mid \tau_i \in \tau\}. \tag{2}$$

It is well-known that for both EDF and FP on a preemptive uniprocessor, the worst-case job sequence that can be generated by a set of sporadic tasks $\tau$ is exactly the same job sequence that is generated by the synchronous periodic task set $\tau'$ with the same parameters (Baruah et al. 1990a; Lehoczky 1990). It follows that the FP- and EDF-schedulability problems coincide for synchronous periodic and sporadic tasks on preemptive uniprocessors. As EDF is optimal in this setting (Dertouzos 1974), their feasibility problems also coincide. As such, we first consider both these task models jointly, and then asynchronous periodic tasks.

## *Sporadic and synchronous periodic tasks*

The preemptive uniprocessor scheduling of sporadic and synchronous periodic tasks is likely the most well-studied problem in real-time scheduling theory, with classic algorithms for establishing schedulability with both FP and EDF. The table in Figure 1 summarizes the current state-of-the-art in the complexity of these problems.

### Upper bounds

First we briefly summarize the most important results that yield upper bounds on the computational complexity of these problems. For EDF, upper bounds are provided by Liu and Layland (1973) for implicit deadlines and Baruah et al. (1990a) for constrained and arbitrary deadlines.

**Theorem 1 (Liu and Layland (1973)).** *A task set $\tau$ of sporadic or synchronous periodic implicit-deadline tasks is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor if and only if* $\mathrm{U}(\tau) \leq 1$.

**Theorem 2 (Baruah et al. (1990a)).** *A task set $\tau$ of sporadic or synchronous periodic arbitrary-deadline tasks is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor if and only if* $\mathrm{U}(\tau) \leq 1$ *and*

$$\forall \ell \in \{0, 1, \ldots, B\}, \quad \mathrm{dbf}(\tau, \ell) \leq \ell. \tag{3}$$

| | | Implicit deadlines ($D_i = T_i$) | Constrained deadlines ($D_i \leq T_i$) | Arbitrary deadlines ($D_i$, $T_i$ unrelated) |
|---|---|---|---|---|
| FP | Arbitrary utilization | Weakly NP-complete (Pseudo-poly. time algorithm known) | Weakly NP-complete (Pseudo-poly. time algorithm known) | In $\Pi_2^P$ Weakly NP-hard (No pseudo-poly. time algorithm known) |
| FP | Utilization bounded by a constant $c$ | In NP In P for $c \leq \ln 2$ and RM priorities (Pseudo-poly. time algorithm known) | Weakly NP-complete for $0 < c < 1$ (Pseudo-poly. time algorithm known) | In $\Pi_2^P$ Weakly NP-hard for $0 < c < 1$ (No pseudo-poly. time algorithm known) |
| EDF / feasibility | Arbitrary utilization | In P | Strongly coNP-complete | Strongly coNP-complete |
| EDF / feasibility | Utilization bounded by a constant $c$ | In P | Weakly coNP-complete for $0 < c < 1$ (Pseudo-poly. time algorithm known) | Weakly coNP-complete for $0 < c < 1$ (Pseudo-poly. time algorithm known) |

Fig. 1: State-of-the-art in the complexity of preemptive schedulability problems for sporadic or synchronous periodic tasks. Darker cells have open problems.

*where*

$$\mathrm{dbf}(\tau, \ell) \stackrel{def}{=} \sum_{\tau_i \in \tau} \max\left\{0, \left\lfloor \frac{\ell - D_i}{T_i} \right\rfloor + 1\right\} C_i \tag{4}$$

*is the* demand bound function *of* $\tau$ *in time interval lengths* $\ell$ *and where*

$$B \stackrel{def}{=} \min\{B_1, B_2\}, \tag{5}$$

$$B_1 \stackrel{def}{=} \mathcal{P}(\tau) + \max\{D_i \mid \tau_i \in \tau\}, \tag{6}$$

$$B_2 \stackrel{def}{=} \frac{\mathrm{U}(\tau)}{1 - \mathrm{U}(\tau)} \max\{T_i - D_i \mid \tau_i \in \tau\}. \tag{7}$$

From Theorem 1 we clearly have a trivial polynomial time schedulability test for task sets with implicit deadlines.

For constrained or arbitrary deadlines, we note that Theorem 2 yields an exponential time test because $B_1$ is bounded by an exponential function in the size of the

representation of $\tau$ and $\mathrm{dbf}(\tau,\ell)$ can be evaluated in polynomial time for a given value of $\ell$ in the range $\{0,1,\ldots,B\}$. However, when $B_2$ is defined we often have $B_2 \ll B_1$. In particular, if we restrict attention to task sets $\tau$ with $\mathrm{U}(\tau) \leq c$ for some constant $c < 1$, then $B_2$ is bounded by a multi-variable polynomial function in the size of the representation of $\tau$ and its largest numerical parameter. In this special case, Theorem 2 therefore yields a pseudo-polynomial time test. Finally, from Theorem 2 it is clear that the EDF-schedulability problem is in coNP because a single value of $\ell$ for which the inequality in Eq. 3 does not hold serves as a polynomial-time verifiable witness of unschedulability.

For FP, on the other hand, upper bounds are provided by Joseph and Pandya (1986) for implicit and constrained deadlines and by Lehoczky (1990) for arbitrary deadlines. Liu and Layland (1973) provides an upper bound for a special case.

**Theorem 3 (Joseph and Pandya (1986)).** *Let $\tau = \{\tau_1,\ldots,\tau_n\}$ be a synchronous periodic or sporadic task set with implicit or constrained deadlines indexed by decreasing priority, so that $\tau_i$ has higher priority than $\tau_j$ if $i < j$. Then $\tau$ is FP-schedulable on a preemptive uniprocessor if and only if for all $\tau_i \in \tau$*

$$R_i \;\leq\; D_i, \tag{8}$$

*where the worst-case response time $R_i$ of task $\tau_i$ is the smallest positive fixed point to the recurrence relation*

$$R_i \;=\; C_i + \sum_{j=1}^{i-1} \left\lceil \frac{R_i}{T_j} \right\rceil C_j. \tag{9}$$

**Theorem 4 (Lehoczky (1990)).** *Let $\tau = \{\tau_1,\ldots,\tau_n\}$ be a synchronous periodic or sporadic task set with arbitrary deadlines indexed by decreasing priority, so that $\tau_i$ has higher priority than $\tau_j$ if $i < j$. Then $\tau$ is FP-schedulable on a preemptive uniprocessor if and only if for all $\tau_i \in \tau$*

$$\forall k \in \{1,2,\ldots,N_i\}, \quad W_i(k,(k-1)T_i + D_i) \;\leq\; 1, \tag{10}$$

*where*

$$W_i(k,x) \;\stackrel{def}{=}\; \min_{t \leq x} \left( \left( kC_i + \sum_{j=1}^{i-1} \left\lceil \frac{t}{T_j} \right\rceil C_j \right) / t \right) \tag{11}$$

*and*

$$N_i \;\stackrel{def}{=}\; \min\{k \mid W_i(k,kT_i) \leq 1\}. \tag{12}$$

**Theorem 5 (Liu and Layland (1973)).** *A task set $\tau = \{\tau_1,\ldots,\tau_n\}$ of sporadic or synchronous periodic implicit-deadline tasks is FP-schedulable with the rate-monotonic (RM) priority ordering on a preemptive uniprocessor if and only if* $\mathrm{U}(\tau) \leq n(2^{1/n} - 1)$.

From Theorem 3 we have a test for task sets with implicit or constrained deadlines. Because we do not need to consider any $R_i$ where $R_i > D_i$, we can evaluate this test in pseudo-polynomial time. From Theorem 3 it is also clear that the FP-schedulability problem for implicit or constrained deadlines is in NP because a collection of small fixed points for Eq. 9 serves as a polynomial-time verifiable witness of schedulability.

For arbitrary deadlines, there are no pseudo-polynomial time tests known, but Theorem 4 provides a test that can be evaluated in exponential time because $N_i$ will never be larger than the number of jobs from $\tau_i$ in a single hyper-period assuming $\mathrm{U}(\tau) \leq 1$ and for Eq. 11 it is enough to consider integer values of $t$. While not stated by Lehoczky (1990), it is not difficult to see from Theorem 4 that the FP-schedulability problem with arbitrary deadlines must be in the complexity class $\Pi_2^{\mathrm{P}}$ at the second level of the polynomial hierarchy. (To see this, note that a task set $\tau$ is schedulable if *for all* $\tau_i \in \tau$ and *for all* $k \leq N_i$ *there exists* a value for $t$ such that the expression in Eq. 11 is small enough.)

It can be noted that these upper bounds hold both if we ask ask about the FP-schedulability of a task set $\tau$ with a given priority ordering or if we ask whether there exists a priority ordering with which $\tau$ is FP-schedulable. The reason the upper bounds hold in both cases is because it is possible to identify a priority ordering for which a task set is FP-schedulable with little overhead if one exists. In the case with implicit or constrained deadlines, we know that RM and DM are optimal, respectively (Liu and Layland 1973; Leung and Whitehead 1982). With arbitrary deadlines we can use the general method of Audsley (1991) together with the test from Theorem 4 to generate a priority ordering.

Theorem 5 gives a polynomial time test for the narrow special case of task sets with implicit deadlines, RM priority ordering and utilization bounded from above by $\lim_{n\to\infty} n(2^{1/n} - 1) = \ln 2 \approx 0.693$.

### Lower bounds

We will now summarize the lower bounds known on the complexity of these problems. For EDF-schedulability, these lower bounds come from Ekberg and Yi (2015b) and Ekberg and Yi (2015a), where the former deals with the general case and the latter with the special case with bounded utilization. (The first of these subsumes prior results by Eisenbrand and Rothvoß (2010)).

**Theorem 6 (Ekberg and Yi (2015b)).** *The problem of deciding whether a task set of synchronous periodic or sporadic tasks with constrained or arbitrary deadlines is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor is strongly* coNP*-hard.*

**Theorem 7 (Ekberg and Yi (2015a)).** *The problem of deciding whether a task set of synchronous periodic or sporadic tasks with constrained or arbitrary deadlines and utilization bounded by any constant $c$, where $0 < c < 1$, is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor is weakly* coNP*-hard.*

The above result of Ekberg and Yi (2015b) was achieved by relating the EDF-schedulability problem to the Simultaneous Congruences Problem (SCP). SCP is a number theoretic decision problem that was first used by Leung and Whitehead (1982) to show lower bounds on the complexity of FP-schedulability for asynchronous periodic tasks. Leung and Whitehead (1982) showed that SCP is weakly NP-complete by a reduction from CLIQUE, but this result was later improved by Baruah et al. (1990b) who showed that SCP is in fact strongly NP-complete by an alternative reduction from 3-SAT. Ekberg and Yi (2015b) presented a pseudo-polynomial transformation (as defined by Garey and Johnson (1978)) from SCP to the complement of the EDF-schedulability problem, which demonstrated the strong coNP-hardness of EDF-schedulability. This holds even if all tasks have unit execution times (that is, $C_i = 1$ for all $\tau_i \in \tau$). Combined with the upper bound provided by Theorem 2, we can conclude that this problem is strongly coNP-complete. Eisenbrand and Rothvoß (2010) had previously shown weak coNP-hardness for the EDF-schedulability problem by relating it to inapproximability results of Diophantine approximation.

The strong coNP-hardness of the EDF-schedulability problem for constrained and arbitrary deadlines means it cannot have a pseudo-polynomial time test unless $\mathsf{P} = \mathsf{NP}$. However, from Theorem 2 we know that this problem does have a pseudo-polynomial time test in the special case restricted to task sets with utilization bounded by a constant $c$, where $c < 1$. Ekberg and Yi (2015a) showed that this restricted case is weakly coNP-hard for any choice of $c$ such that $0 < c < 1$, and therefore that the pseudo-polynomial time test is in a sense the best possible unless $\mathsf{P} = \mathsf{NP}$. This result was achieved by reducing the general case of the EDF-schedulability problem to the restricted case with bounded utilization. This reduction causes an exponential blowup of numerical task parameters, which is why it shows only weak coNP-hardness for the special case even though the general case is strongly coNP-hard.

For FP, the best known lower bounds come from Ekberg and Yi (2017).

**Theorem 8 (Ekberg and Yi (2017)).** *The problem of deciding whether a task set of synchronous periodic or sporadic tasks is FP-schedulable on a preemptive uniprocessor is weakly NP-hard, even if restricted to either of the following special cases.*

1. *Implicit deadlines and RM priority ordering.*
2. *Constrained deadlines, DM priority ordering and utilization bounded by constant c, such that* $0 < c < 1$.

This result was found by reducing the special case of the EDF-schedulability problem with bounded utilization to the complement of the FP-schedulability problem, exploiting a duality which exists between the conditions in Eqs. 3 and 8 when tasks have pairwise coprime periods. An intermediate result of Ekberg and Yi (2017) was to show that the EDF-schedulability problem with bounded utilization remains hard when restricted to such periods.

As the RM and DM priority orderings are optimal in these settings, a corollary of the above is that the FP-schedulability problem is NP-hard also if we ask if there exists a priority ordering with which the task set is schedulable.

From Theorem 8 and Theorem 3 we can conclude that the FP-schedulability problem is weakly NP-complete for implicit and constrained deadlines. We can therefore conclude that the pseudo-polynomial time test yielded by Theorem 3 is in a sense the best possible. Membership in NP is not known for arbitrary deadlines, so in that case there is a gap between upper and lower bounds. Theorem 8 also tells us that FP-schedulability remains hard even with bounded utilization as long as deadlines are constrained or arbitrary, mirroring the case for EDF.

For the case with implicit deadlines and utilization bounded by a constant $c$, Theorem 5 gives a trivial polynomial time algorithm for FP-schedulability when $c \leq \ln 2$ and RM priority ordering is used. It remains open if it can also be solved in polynomial time for $\ln 2 < c < 1$ or for other priority orderings.

#### Other results

In addition to the upper and lower bounds described above, there are many other interesting results known. Here we review some of them.

Eisenbrand and Rothvoß (2008) showed that for FP-scheduling, it is not even possible to approximate the worst-case response time (i.e., the smallest fixed point $R_i$ in Eq. 9) of a given task within a constant factor unless $\mathsf{P} = \mathsf{NP}$. However, this does not imply the NP-hardness of FP-schedulability testing (though this was later shown to be NP-hard by Ekberg and Yi (2017)) as the reduction used to show the hardness of such approximation can construct higher-priority tasks that are themselves clearly unschedulable.

For another type of approximation, where the approximated quantity is the speed of the processor, Albers and Slomka (2004) and Fisher and Baruah (2005) gave fully polynomial time approximation schemes for schedulability testing with EDF and FP, respectively. Given a task set of $n$ tasks and a constant $\varepsilon$, where $0 < \varepsilon < 1$, these tests correctly identify in time bounded by a polynomial in $n/\varepsilon$ any task sets that are unschedulable on a unit-speed processor as well as any task sets that are schedulable on a slower processor of speed $1 - \varepsilon$. Task sets that are schedulable on a unit-speed processor but not on a $(1 - \varepsilon)$-speed processor may be misclassified as unschedulable.

Task sets are called harmonic if for each pair of tasks, the period of one task divides the period of the other. Bonifaci et al. (2013) presented polynomial time schedulability tests for both EDF and FP for harmonic task sets with constrained deadlines.

### *Asynchronous periodic tasks*

As asynchronous periodic tasks are a generalization of synchronous periodic tasks, their schedulability problems must be at least as hard, and therefore all the lower bounds from the previous section carry over here.

| | | Implicit deadlines ($D_i = T_i$) | Constrained deadlines ($D_i \leq T_i$) | Arbitrary deadlines ($D_i$, $T_i$ unrelated) |
|---|---|---|---|---|
| FP | Arbitrary utilization | In EXP<br>Weakly NP-hard and strongly coNP-hard | In EXP<br>Weakly NP-hard and strongly coNP-hard | In EXP<br>Weakly NP-hard and strongly coNP-hard |
| FP | Utilization bounded by a constant $c$ | In EXP<br>In P for $c \leq \ln 2$ and RM priorities | In EXP<br>Weakly NP-hard and strongly coNP-hard for $0 < c < 1$ | In EXP<br>Weakly NP-hard and strongly coNP-hard for $0 < c < 1$ |
| EDF / feasibility | Arbitrary utilization | In P | Strongly coNP-complete | Strongly coNP-complete |
| EDF / feasibility | Utilization bounded by a constant $c$ | In P | Strongly coNP-complete | Strongly coNP-complete |

Fig. 2: State-of-the-art in the complexity of preemptive schedulability problems for asynchronous periodic tasks. Darker cells have open problems.

At the same time, we know that the job sequence generated by a synchronous periodic task set is at least as difficult to schedule for both EDF and FP as the job sequence generated by a corresponding asynchronous periodic task set on a preemptive uniprocessor. By a corresponding asynchronous task set we mean a task set with exactly the same parameters, except it may have non-zero offsets. From this is follows that any schedulability test for synchronous periodic (or sporadic) tasks is still sufficient, but not necessarily exact, for asynchronous periodic tasks. In fact, the only test described in the previous section that remains exact for asynchronous periodic tasks is that of Theorem 1 for EDF-schedulability of task sets with implicit deadlines. (We know it must be sufficient, and its condition clearly remains necessary.)

For EDF-schedulability with constrained and arbitrary deadlines, Baruah et al. (1990b) also gave an exact test for the asynchronous case that is very similar to the test of Theorem 2 for synchronous periodic and sporadic tasks.

**Theorem 9 (Baruah et al. (1990b)).** *A task set $\tau$ of asynchronous periodic arbitrary-deadline tasks is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor if and only if* $\mathrm{U}(\tau) \leq 1$ *and*

$$\forall t_1, t_2 \in \{0, 1, \ldots, B\} \text{ such that } t_1 < t_2, \quad \mathrm{dbf}(\tau, t_1, t_2) \leq t_2 - t_1. \tag{13}$$

*where*

$$\mathrm{dbf}(\tau, t_1, t_2) \stackrel{def}{=} \sum_{\tau_i \in \tau} \max\left\{0, \left\lfloor \frac{t_2 - O_i - D_i}{T_i} \right\rfloor - \max\left\{0, \left\lceil \frac{t_1 - O_i}{T_i} \right\rceil\right\} + 1\right\} C_i \tag{14}$$

*is the* demand bound function *of $\tau$ in the time interval $[t_1, t_2]$ and where*

$$B \stackrel{def}{=} \max\{O_i \mid \tau_i \in \tau\} + 2\mathcal{P}(\tau). \tag{15}$$

This theorem clearly demonstrates that the EDF-schedulability problem for asynchronous periodic tasks is also in coNP. A major difference between Theorem 9 and Theorem 2 is that we here have to consider all possible time intervals contained in $[0, B]$, while for Theorem 2 we implicitly only considered time intervals starting at time point zero. In terms of our complexity classification, however, the important difference is that the value of $B$ here is not bounded by any polynomial function in the size of the representation of $\tau$ and its largest numerical parameter, which is the case in Theorem 2 when restricted to the special case of task sets with utilization bounded by a constant $c < 1$. Theorem 9 therefore does not provide any pseudo-polynomial time test, even in the case with bounded utilization. Indeed, Baruah et al. (1990b) showed that even the bounded case of this problem is strongly coNP-hard. Leung and Merrill (1980) had already shown that it was weakly coNP-hard by reducing the Simultaneous Congruences Problem (SCP) to the complement of it. At the time, SCP was only known to be weakly NP-hard, but this was improved to strong NP-hardness by Baruah et al. (1990b). In combination with the previous reduction of Leung and Merrill (1980), which do not cause exponential blowup of numerical parameters, the strong coNP-hardness of the EDF-schedulability problem followed.

**Theorem 10 (Leung and Merrill (1980); Baruah et al. (1990b)).** *The problem of deciding whether a task set of asynchronous periodic tasks with constrained or arbitrary deadlines and utilization bounded by any constant c, where $0 < c < 1$, is EDF-schedulable (or, equivalently, feasible) on a preemptive uniprocessor is strongly coNP-complete.*

For FP, the complexity of the schedulability problems in all the cases in Figure 2 are still open, though we have some bounds already. Because Theorem 5 must still provide a sufficient condition, the case with implicit deadlines and utilization bounded by a constant $c$ remains easy as long as we have RM priorities and $c \leq \ln 2$. All other cases in Figure 2 must be NP-hard as the lower bounds of Ekberg and Yi (2017) are carried over from the synchronous periodic case.

In addition, these same problems are also coNP-hard, as was shown by Leung and Whitehead (1982). Similar to the case with EDF, Leung and Whitehead (1982) first demonstrated weak coNP-hardness by a reduction from SCP. Their result is easily improved to strong coNP-hardness given that Baruah et al. (1990b) have since shown SCP to be strongly NP-hard. While not stated by Leung and Whitehead (1982), it is not difficult to see from their proofs that their results apply also to the case with bounded utilization if deadlines are constrained or arbitrary.

**Theorem 11 (Leung and Whitehead (1982); Baruah et al. (1990b)).** *The problem of deciding whether a task set of asynchronous periodic tasks is FP-schedulable with a given priority ordering on a preemptive uniprocessor is strongly coNP-hard, even if restricted to (i) implicit deadlines or (ii) constrained deadlines and utilization bounded by a constant c, such that* $0 < c < 1$.

As seen above, most of the FP-schedulability problems for asynchronous periodic tasks are both NP-hard and coNP-hard. It therefore seems unlikely any of them are NP- or coNP-complete as that would imply $\mathsf{NP} = \mathsf{coNP}$.[1] The FP-schedulability of asynchronous periodic tasks can be tested in exponential time, even for arbitrary deadlines as shown by Goossens (1999). For a task set $\tau$, this essentially amounts to computing the response times of all jobs up to time point $\max\{O_i \mid \tau_i \in \tau\} + 2\mathcal{P}(\tau)$.

## Task models with complex job-release patterns

Here we consider task models where tasks can generate more complex job sequences than those generated by sporadic tasks, but where each job $J$ is still an independent unit of work expressed by a release time $r$, a worst-case execution time $c$, and an absolute deadline $d$.

The task models considered in this section form of hierarchy with respect to their expressiveness. We say that a task model $\mathcal{M}_A$ generalizes a task model $\mathcal{M}_B$ if there exists a (total) function $f$ mapping tasks from model $\mathcal{M}_B$ to model $\mathcal{M}_A$, such that for any task $\tau$ of model $\mathcal{M}_B$, the set of possible job (sub-)sequences generated by $\tau$ and $f(\tau)$ are exactly the same, modulo dummy jobs with zero execution time. Intuitively, this means that anything that can be modeled in $\mathcal{M}_B$ can also be modeled in $\mathcal{M}_A$, making the latter at least as expressive. Such a generalization relation must be transitive.

For any pair of the task models considered in this section where such a mapping $f$ is known, $f$ can easily be computed in polynomial time. Consequently, if $\mathcal{M}_A$ generalizes $\mathcal{M}_B$, any upper bounds on the computational complexity of schedulability problems for $\mathcal{M}_A$ can also be applied to $\mathcal{M}_B$, while any lower bound for $\mathcal{M}_B$ can be applied to $\mathcal{M}_A$.

[1] It is unknown if $\mathsf{NP} = \mathsf{coNP}$, but it is generally conjectured that $\mathsf{NP} \neq \mathsf{coNP}$.

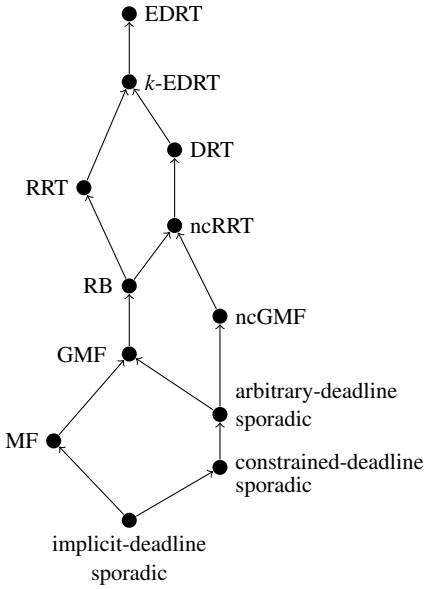


Fig. 3: A hierarchy of task models. Here we have lifted some restrictions on the relation between different task parameter values, and also assume that those parameters are natural numbers. Figure adapted from Stigge (2014).

The following is a list of the task models that we fit into a hierarchy, with the abbreviations that we will use for them. The list does not try to be exhaustive with respect to the task models than could be included.

| | |
|---|---|
| MF | The Multiframe task model (Mok and Chen 1997) |
| GMF | The Generalized Multiframe task model (Baruah et al. 1999) |
| ncGMF | The Non-Cyclic GMF task model (Tchidjo Moyo et al. 2010) |
| RB | The Recurring Branching task model (Baruah 1998) |
| RRT | The Recurring Real-Time task model (Baruah 2003) |
| ncRRT | The Non-Cyclic RRT task model (Baruah 2010) |
| DRT | The Digraph Real-Time task model (Stigge et al. 2011b) |
| EDRT | The Extended DRT task model (Stigge et al. 2011a) |

We will not describe the syntax and semantics of each of these task models. Most of them can be readily understood as graph-based task models restricted to particular classes of graphs. We refer the reader to the survey of Stigge and Yi (2015) for more details.

In addition, we include sporadic tasks with implicit, constrained and arbitrary deadlines. We also include a restricted case of EDRT, called $k$-EDRT, which only allows tasks with at most $k$ "global" constraints, for a constant $k$. The $k$-EDRT task model is relevant because it is the most general in this hierarchy that has efficient

schedulability tests in some settings (Stigge et al. 2011a). Another interesting property of $k$-EDRT is that it generalizes RRT already for $k = 1$.

The hierarchy of task models is captured by Figure 3, where an arrow from task model $\mathcal{M}_B$ to $\mathcal{M}_A$ means that $\mathcal{M}_A$ generalizes $\mathcal{M}_B$. For the sake of brevity, we have here lifted restrictions on the relation of task parameter values that were assumed in some of the original formulations of these task models, so that most of them here generalize arbitrary-deadline sporadic tasks. We assume that the numerical task parameters in all models are natural numbers. Some were originally specified with real numbers as parameters, but this is not a good choice if we want to reason about their related computational problems.

## *EDF-schedulability*

Because tasks in all these task models generate independent jobs in a way that is unaffected by scheduling decisions, EDF is still an optimal scheduling algorithm (Dertouzos 1974). Many of the papers in which these task models are presented also give EDF-schedulability tests, but thanks to the hierarchy in Figure 3, a few of those subsume the rest in terms of complexity classification. Most of the EDF-schedulability tests use some variant of the test in Theorem 2 based on demand bound functions, with a bound on the time intervals to test that is basically an extension of $B_2$ in Eq. 7. This often provides pseudo-polynomial time tests when restricted to task sets $\tau$ with $\mathrm{U}(\tau) \leq c$, for a constant $c < 1$. However, for many of the task models there is no alternative presented to bound $B_1$ in Theorem 2. As a consequence, these tests only target the case with bounded utilization, and may not work at all if $\mathrm{U}(\tau) = 1$ where a bound like $B_2$ is not defined. In the following, for the complexity of EDF-schedulability we only consider the case that is restricted to task sets with utilization bounded by a constant $c$, where $0 < c < 1$.

Figure 4 shows the best known upper and lower bounds on the complexity of the EDF-schedulability problem with bounded utilization for task models in the hierarchy. The upper bounds can be found in three places (subsuming previous bounds). First, there is the (trivial) polynomial time test of Liu and Layland (1973) for implicit deadline sporadic tasks. Second, there is the pseudo-polynomial time test for $k$-EDRT task sets for any constant $k$ by Stigge et al. (2011a), which yields tests of the same complexity for all task sets generalized by $k$-EDRT. Last, the EDF-schedulability problem for RB task sets is in coNP. This was not stated by Baruah (1998), but follows easily from the test described. It is an open problem whether the EDF-schedulability problems for more general task models are also in coNP. The bounds for P and coNP in Figure 4 would hold also without the restriction to bounded utilization, but the bound for pseudo-polynomial time would not hold assuming $\mathsf{P} \neq \mathsf{NP}$.

The lower bounds on the complexity of the EDF-schedulability problem with utilization bounded by $c$ come from two sources. First, Stigge et al. (2011a) show that it is strongly coNP-hard for the EDRT task model, for any $c$ where $0 < c < 1$. It

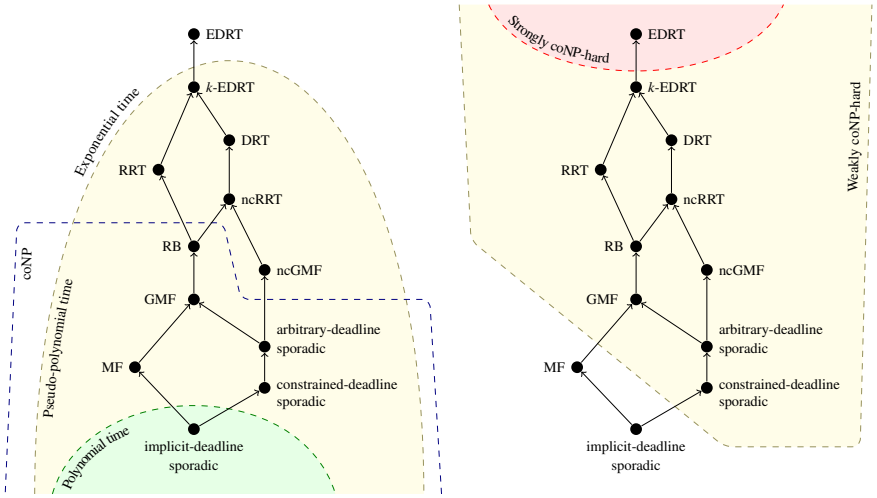


Fig. 4: Currently best-known upper bounds (left) and lower bounds (right) on the complexity of the EDF-schedulability problem (or, equivalently, the feasibility problem) when restricted to task sets with utilization bounded by a constant $c < 1$.

can be noted that this result also holds when the EDRT task model is restricted to the equivalent of constrained deadlines. Second, from Ekberg and Yi (2015a) we know that the problem is weakly coNP-hard already for constrained-deadline sporadic task sets and for any $0 < c < 1$ (see Theorem 7). From Ekberg and Yi (2015b) we also know that the EDF-schedulability problem is strongly coNP-hard already for constrained-deadline sporadic task sets and up if we do not bound the utilization by a constant.

## *FP-schedulability*

Schedulability testing for FP is generally harder than for EDF because exact tests seemingly need to consider a large number of combinations of concrete job sequences. In contrast to EDF, where the local worst case for each task is easily combined to a global worst case for the task set (e.g., by the summation in Eq. 4), exact FP-schedulability tests that have been presented for more general task models must try different combinations of per-task behaviors. This generally leads to tests with very high worst-case complexity.

The best known upper and lower bounds for FP-schedulability is shown in Figure 5. For upper bounds, we have mainly the results from Theorem 3, which show

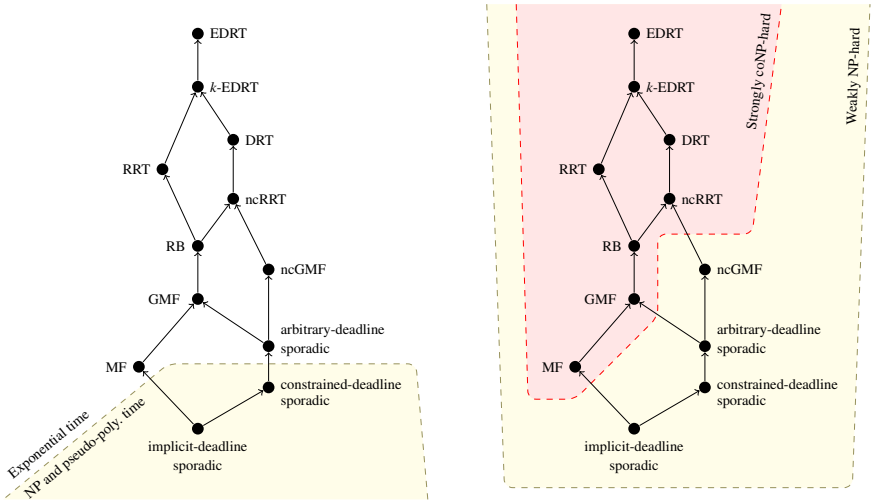


Fig. 5: Currently best-known upper bounds (left) and lower bounds (right) on the complexity of the FP-schedulability problem.

that the problems for implicit- and constrained-deadline sporadic tasks are in NP and can be solved in pseudo-polynomial time. For the remaining task models, their FP-schedulability problems can be seen to be in EXP by arguments of exhaustive simulation.

The lower bounds come from two sources. Ekberg and Yi (2017) show weak NP-hardness already for implicit-deadline sporadic tasks (see Theorem 8). Stigge (2014) show strong coNP-hardness for the MF task model by a reduction from the complement of the strongly NP-complete 3-PARTITION problem. As can be seen in Figure 5, the FP-schedulability problem for many task models are both NP- and coNP-hard, and therefore cannot be in NP or coNP unless $\mathsf{NP} = \mathsf{coNP}$.

Despite the high complexity of some of these task models, Stigge and Yi (2013) presented an FP-schedulability test for constrained-deadline DRT task sets based on the technique of *iterated abstraction refinement*. While this test has a high worst-case complexity, empirical evaluations by Stigge and Yi (2013) show that it in practice outperforms the pseudo-polynomial time test for EDF-schedulability for the same task model.

## Conclusions

In this chapter we have reviewed the current state-of-the-art in classifying the complexity of schedulability problems on preemptive uniprocessors. For the sake of brevity, the focus was on task models that generate simple independent jobs. This

only represents a part of the task models that have been considered in the literature, however. Task models with locked mutually exclusive resources or with self-suspensions are examples of models not considered here, although some results regarding their complexity are also known.

A take-away from this review is the observation that almost all real-time schedulability problems—even very basic ones—are computationally intractable in some sense. Even so, many of these are routinely solved exactly. For example, the FP-schedulability problem for implicit-deadline sporadic tasks is (weakly) NP-complete, but despite this, very few seem to think that the response-time analysis of Theorem 3 is impractically slow, at least for offline analysis. Also EDF-schedulability of constrained-deadline sporadic tasks—a strongly coNP-complete problem—is often solved without any problems by practitioners who have never considered restricting themselves to task sets with utilization bounded by a constant.

It seems therefore as if we should not automatically settle for approximate solutions to problems that we know are NP- or coNP-hard out of a concern for efficiency. Some of these problems are likely genuinely difficult to solve also in practice, but others may well allow practically efficient exact solutions.

Another take-away is that complexity does not end at NP or coNP. Many of the problems seen in this chapter, such as FP-schedulability for asynchronous periodic tasks or for task models that generalize the Multiframe model, are known to be both NP- and coNP-hard at the same time. Assuming a widely held conjecture in complexity theory (i.e., $\mathsf{NP} \neq \mathsf{coNP}$) these problems are not in NP or coNP. They may be hard for larger complexity classes, such as PSPACE or higher levels of the polynomial hierarchy. Actually pinpointing the complexity of these problems should be considered a worthwhile effort.

## References

K. Albers and 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 (ECRTS)*, pages 187–195, June 2004.

N. Audsley. Optimal priority assignment and feasibility of static priority tasks with arbitrary start times. Technical report, University of York, England, 1991.

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)*, pages 182–190, 1990a.

S. 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*, 2(4):301–324, 1990b.

S. Baruah, D. Chen, S. Gorinsky, and A. K. Mok. Generalized multiframe tasks. *Real-Time Systems*, 17:5–22, 1999.

S. K. Baruah. Feasibility analysis of recurring branching tasks. In *Proceedings of the 10th Euromicro Workshop on Real-Time Systems (EWRTS)*, pages 138–145, 1998.

S. K. Baruah. Dynamic- and static-priority scheduling of recurring real-time tasks. *Real-Time Systems*, 24(1):93–128, 2003.

S. K. Baruah. The non-cyclic recurring real-time task model. In *Proceedings of the 31st Real-Time Systems Symposium (RTSS)*, pages 173–182, 2010.

V. Bonifaci, A. Marchetti-Spaccamela, N. Megow, and A. Wiese. Polynomial-time exact schedulability tests for harmonic real-time tasks. In *Proceedings of the 34th Real-Time Systems Symposium (RTSS)*, pages 236–245, Dec 2013.

M. L. Dertouzos. Control robotics: The procedural control of physical processes. In *Proceedings of the IFIP congress*, volume 74, pages 807–813, 1974.

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)*, pages 397–406. IEEE Computer Society, 2008.

F. Eisenbrand and 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 (SODA)*, pages 1029–1034, 2010.

P. Ekberg and W. Yi. Uniprocessor feasibility of sporadic tasks remains coNP-complete under bounded utilization. In *Proceedings of the 36th Real-Time Systems Symposium (RTSS)*, pages 87–95, 2015a. doi: 10.1109/RTSS.2015.16.

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)*, pages 281 – 286, 2015b.

P. Ekberg and W. Yi. Fixed-priority schedulability of sporadic tasks on uniprocessors is NP-hard. In *Proceedings of the 38th Real-Time Systems Symposium (RTSS)*, pages 139–146, 2017. doi: 10.1109/RTSS.2017.00020.

N. Fisher and S. Baruah. A fully polynomial-time approximation scheme for feasibility analysis in static-priority systems with arbitrary relative deadlines. In *Proceedings of the 17th Euromicro Conference on Real-Time Systems (ECRTS)*, pages 117–126, July 2005.

M. R. Garey and D. S. Johnson. Strong NP-completeness results: Motivation, examples, and implications. *Journal of the ACM*, 25(3):499–508, 1978.

J. Goossens. *Scheduling of Hard Real-Time Periodic Systems with Various Kinds of Deadline and Offset Constraints*. PhD thesis, Université libre de Bruxelles, 1999.

M. Joseph and P. Pandya. Finding response times in a real-time system. *The Computer Journal*, 29(5):390–395, 1986.

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

J. Y.-T. Leung and M. Merrill. A note on preemptive scheduling of periodic, real-time tasks. *Information Processing Letters*, 11(3):115–118, 1980.

J. Y.-T. Leung and J. Whitehead. On the complexity of fixed-priority scheduling of periodic, real-time tasks. *Performance Evaluation*, 2(4):237–250, 1982.

C.-L. Liu and J. W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *Journal of the ACM*, 20(1):46–61, 1973.

A. K. Mok and D. Chen. A multiframe model for real-time tasks. *IEEE Transactions on Software Engineering*, 23(10):635–645, 1997.

M. Stigge. *Real-Time Workload Models: Expressiveness vs. Analysis Efficiency*. PhD thesis, Uppsala University, Department of Information Technology, 2014.

M. Stigge and W. Yi. Combinatorial abstraction refinement for feasibility analysis. In *Proceedings of the 34th Real-Time Systems Symposium (RTSS)*, pages 340–349, 2013.

M. Stigge and W. Yi. Graph-based models for real-time workload: a survey. *Real-Time Systems*, 51(5):602–636, sep 2015.

M. Stigge, P. Ekberg, N. Guan, and W. Yi. On the tractability of digraph-based task models. In *Proceedings of the 23rd Euromicro Conference on Real-Time Systems (ECRTS)*, pages 162–171, July 2011a.

M. Stigge, P. Ekberg, N. Guan, and W. Yi. The digraph real-time task model. In *Proceedings of the 17th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, pages 71–80, 2011b.

N. Tchidjo Moyo, E. Nicollet, F. Lafaye, and C. Moy. On schedulability analysis of non-cyclic generalized multiframe tasks. In *Proceedings of the 22nd Euromicro Conference on Real-Time Systems (ECRTS)*, pages 271–278, 2010.