Static-priority Real-time Scheduling: Response Time Computation is NP-hard

Friedrich Eisenbrand* Thomas Rothvoß

Institute of Mathematics
EPFL, 1015 Lausanne, Switzerland
{friedrich.eisenbrand,thomas.rothvoss}@epfl.ch

August 21, 2008

1 Introduction

This paper is concerned with a classical problem in real-time scheduling. We are given n tasks $(c_1, p_1), \ldots, (c_n, p_n)$, where each task is determined by a running time c_i and a period $p_i \geq c_i$. Each task generates a job of length c_i at each integer multiple of p_i (starting at time 0), which has to be finished before the next job is released; the jobs have *implicit deadlines*, i.e., the deadlines are given by the periods themselves. A *preemptive, static-priority* schedule consists of a priority assignment to all tasks, such that jobs of higher priority always preempt jobs of lower priority.

Liu and Layland [LL73] have shown that the *rate-monotonic* schedule is optimal. Meaning that if there is any feasible priority assignment, then all jobs will meet their deadline if task i has priority $1/p_i$. This means that the computation of an optimal schedule is tractable. Still an important problem remains: Is there a polynomial algorithm which decides whether a task in this schedule is feasible, i.e., whether each job of a task finishes before its implicit deadline?

The *response time* of a task is the maximum amount of time that may elapse between the arrival of a job of this task and its completion. It was already shown in [LL73] that the time required by the first job of each task defines its response time. If r_j denotes the response time of the task (c_j, p_j), then the system is feasible, if and only if $r_j \leq p_j$ holds for each j. What is this number r_j and how can it be computed?

Assume that the periods are ordered, i.e., $p_1 \leq p_2 \leq \ldots \leq p_n$. The first job of task (c_j, p_j) needs time r_j. While it is running, the i-th task, $i < j$, interrupts this job $\lceil r_j/p_i \rceil$ many times and requires time c_i at each interruption. Therefore, r_j is the smallest non-negative value such that

$$r_j = c + \sum_{i<j} \left\lceil \frac{r_j}{p_i} \right\rceil c_i \tag{1}$$

holds, see [JP86, LSD89]. Thus to test feasibility, it suffices to compute all response times and to check, whether $r_i \leq p_i$ holds for $i = 1, \ldots, n$.

Now the question is, whether the smallest r_j satisfying (1) can be found efficiently. Several authors write that *"equations of this form do not lend themselves easily to analytical solution"* [JP86, ABD+95] which raises the suspicion that response-time calculation is difficult.

Our main result is a proof that this is indeed difficult. There does not exist a polynomial algorithm for computing the response time of a task in a given task-system, unless $\mathbf{P} = \mathbf{NP}$. If $\mathbf{P} \neq \mathbf{NP}$, there cannot even exist a constant approximation algorithm for response time computation. This also implies that there cannot be an efficient test for the feasibility of *one task* in a task system unless $\mathbf{P} = \mathbf{NP}$. If such a test would exist, we could apply it, together with binary search, to find the response time efficiently.

This result complements a recent result of Fisher and Baruah [FB05], who can guarantee feasibility of a task, or its infeasibility if the processor has speed $1 + \varepsilon$ for any constant $\varepsilon > 0$ in polynomial time. Our result shows that their algorithm is best possible in the sense that *resource augmentation* is indeed necessary for an efficient feasibility test which is based on response-time calculation.

*Supported by Deutsche Forschungsgemeinschaft (DFG) within Priority Programme 1307 "Algorithm Engineering"

Proof method

The insight which leads to our hardness result is the fact that response-time calculation is related to *simultaneous diophantine approximation*, a classical problem from the geometry of numbers, see, e.g. [NZM91]. Here one is given n rational numbers $\alpha_1, \ldots, \alpha_n$, a natural number $N \in \mathbb{N}$ and a rational error bound $\varepsilon > 0$. The task is to find a natural number $1 \leq Q \leq N$ such that the distance of each $Q \cdot \alpha_i$ to its nearest integer is bounded by ε. In other words, we are searching for a natural number Q such that

$$\forall i \colon |Q \cdot \alpha_i - \lfloor Q \cdot \alpha_i \rceil| \leq \varepsilon$$

where $\lfloor x \rceil$ denotes the nearest integer to x. Lagarias [Lag85] has shown that simultaneous diophantine approximation is **NP**-hard. Equation (1) reminds of diophantine approximation. However there are two main difficulties which prevent the immediate application of the result of Lagarias, apart from several minor adjustments.

i) Due to the *rounding up* in equation (1), we want to consider a variant of diophantine approximation in which we measure the distance of $Q \cdot \alpha_i$ to the nearest integer which is larger that $Q \cdot \alpha_i$, i.e., $\lceil Q \cdot \alpha_i \rceil$.

ii) The error in the classical simultaneous approximation has to be small for each individual $Q \cdot \alpha_i$, whereas equation (1) seems to accumulate the errors.

The following variant of simultaneous diophantine approximation, which we call *directed simultaneous diophantine approximation* incorporates these difficulties. This variant, plays also an important role in integer programming and combinatorial optimization, see, e.g. [HW97, HW02].

DIRECTED DIOPHANTINE APPROXIMATION (DDA)
Given rational numbers $\alpha_1, \ldots \alpha_n \in \mathbb{Q}_+$ and a rational number $\varepsilon > 0$, find the smallest $k \in \mathbb{N}$ such that there exists a $Q \in \{1, \ldots, k\}$ with

$$\sum_{i=1}^{n} (\lceil Q \cdot \alpha_i \rceil - Q \cdot \alpha_i) \leq k \cdot \varepsilon.$$

Here Q denotes the solution of the problem, while k gives its value. In the first part of this paper, we show that response-time computation can be reduced to DDA with an *approximation preserving reduction* with factor 2. To explain this, we first formally describe the response-time problem.

RESPONSE TIME COMPUTATION (RTC)
Given tasks $(c_i, p_i) \in \mathbb{Q}_+^2$ for $i = 1, \ldots, n$, find the smallest $r \in \mathbb{Q}_+$, such that

$$c + \sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil c_i \leq r$$

Denote the optimum solutions of DDA and RTC by OPT_{DDA} and OPT_{RTC} respectively. We show that RTC $\leq_2$ DDA holds. More general a reduction $A \leq_\gamma B$ for optimization problems A, B and a constant $\gamma \geq 1$ means, that the existence of a β-approximation algorithm for B implies the existence of a $\gamma \cdot \beta$-approximation for any fixed $\beta \in \mathbb{N}$. Thus a chain of reductions $A_1 \leq_{\gamma_1} \ldots \leq_{\gamma_{m-1}} A_m$ implies that if finding $O(1)$-approximations to A_m is **NP**-hard, then the same holds for A_1.

The second part of this paper deals with a proof that there does not exist a polynomial algorithm which computes a solution to DDA of value k^* with $k^* \leq \gamma \cdot OPT$ for *any* constant $\gamma \geq 1$. This establishes the main result of this paper, namely the fact that response time calculation is **NP**-hard and also that there does not exist a constant approximation algorithm for response-time calculation unless $\mathbf{P} = \mathbf{NP}$.

Apart from yielding a contribution to the theory of real-time scheduling, we think that the proof of inapproximability of DDA is of interest on its own.

2 DDA $\leq_2$ RTC

In this section we show the following result, which is the promised link of response time computation to directed diophantine approximation.

Theorem 1 (DDA $\leq_2$ RTC)**.** *If there exists a β-approximation algorithm for* RTC *for some $\beta \in \mathbb{N}$, then there exists a $2 \cdot \beta$-approximation algorithm for* DDA*.*

The proof is a small sequence of reductions. The first problem that we introduce in this sequence is *earliest idle time*.

EARLIEST IDLE TIME (IDLE)
Given n pairs of rational numbers $(c_i, p_i) \in \mathbb{Q}_+^2$, $i = 1, \ldots, n$ compute the minimum $r > 0$ with

$$\sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil c_i \leq r.$$

This r can be understood as the first time at which the processor is idle.

Lemma 2 (IDLE $\leq_1$ RTC)**.** *For each $\beta \in \mathbb{N}$, if there exists a β-approximation algorithm for* RTC*, then there exists a β-approximation algorithm for* IDLE*.*

Proof. Consider an instance of IDLE given by (c_i, p_i), $i = 1, \ldots, n$. Clearly, we can scale all numbers by the least common multiple D of all denominators to obtain integers. Since r is a solution to the scaled instance if and only if r/D is a solution of the original IDLE-instance, we can assume that $(c_i, p_i) \in \mathbb{N}^2$ for $i = 1, \ldots, n$. Now define $P = \prod_{i=1}^{n} p_i$. Either there is no IDLE solution, then there is nothing to do, or $OPT_{\text{IDLE}} \leq P$. Assume the latter one.

Consider the RTC-instance

$$\min r \quad : \quad \delta + \sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil (c_i - \delta) \leq r,$$

where $\delta = 1/(2 \cdot n \cdot P)$. If r is a solution to IDLE, then r is clearly a solution to RTC.

On the other hand, let $r \leq P$ be a solution to RTC. We have

$$\begin{aligned} \sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil c_i &\leq r + \delta \cdot \left(\sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil - 1 \right) \\ &\leq r + 1/2, \end{aligned}$$

which shows that $\lceil r \rceil$ is a solution to IDLE and that $OPT_{\text{IDLE}} = \lceil OPT_{\text{RTC}} \rceil$ holds. Let r^* be a solution to RTC with $r^* \leq \beta \cdot OPT_{\text{RTC}}$. One has $\lceil r^* \rceil \leq \lceil \beta \cdot OPT_{\text{RTC}} \rceil \leq \beta \cdot OPT_{\text{IDLE}}$ since $\beta \in \mathbb{N}$ is an integer. □

The next problem that we consider is a weighted version of directed diophantine approximation.

WEIGHTED DIOPHANTINE APPROXIMATION (DDA^w)
Given rational numbers $\alpha_1, \ldots, \alpha_n \in \mathbb{Q}_+$, weights $w_1, \ldots, w_n \in \mathbb{Q}_+$ and a value of $\varepsilon > 0$. Find the smallest $Q \in \mathbb{Q}_+$ with

$$\sum_{i=1}^{n} w_i (\lceil Q\alpha_i \rceil - Q\alpha_i) \leq \varepsilon \cdot Q$$

Lemma 3. *One has* DDA$^w \leq_1$ IDLE*.*

Proof. Consider an instance $\alpha_1, \ldots, \alpha_n$, $w_1, \ldots, w_n$ and ε of DDA^w. We construct an instance of IDLE such that any $r \in \mathbb{Q}_+$ is a solution to this instance of IDLE if and only if r is a solution to the DDA^w-instance.

To this end, choose periods $p_i := \frac{1}{\alpha_i}$ and consider an $r \in \mathbb{Q}_+$ which is a solution of DDA^w, i.e., an r satisfying

$$\sum_{i=1}^{n} w_i \left(\left\lceil \frac{r}{p_i} \right\rceil - \frac{r}{p_i} \right) \leq \varepsilon r.$$

Rewriting this equation one obtains

$$\sum_{i=1}^{n} w_i \left\lceil \frac{r}{p_i} \right\rceil \leq r \underbrace{\left(\varepsilon + \sum_{i=1}^{n} \frac{w_i}{p_i} \right)}_{=:\delta}.$$

If we choose $c_i := w_i/\delta$, then the expression is just

$$\sum_{i=1}^{n} \left\lceil \frac{r}{p_i} \right\rceil c_i \leq r$$

which shows that an r is a feasible solution to DDA^w if and only if this r is a feasible solution IDLE. □

To complete the proof of Theorem 1 it remains to show the next lemma.

Lemma 4. *One has* DDA $\leq_2$ DDA^w*.*

Proof. Suppose that there is a β-approximation algorithm for DDA^w and let $\alpha_1, \ldots, \alpha_n$, ε define an instance of DDA. Consider now the instance of DDA^w with some $\varepsilon' > 0$, where we have additionally to the α_i above, an extra $\alpha_0 = 1$ and weights $w_0, \ldots, w_n$ with $w_0 = M$ and $w_i = 1$ for $i = 1, \ldots, n$. Here M is a large number which enforces any β-optimum solution to be an integer. This shows that there is a β-approximation algorithm for the problem

$$\min Q \in \mathbb{N} \colon \sum_{i=1}^{n} (\lceil Q\alpha_i \rceil - Q\alpha_i) \leq Q \cdot \varepsilon'. \qquad (2)$$

Consider now the following problem, which looks very similar to DDA

$$\min Q \in \mathbb{N} \colon \sum_{i=1}^{n} (\lceil Q\alpha_i \rceil - Q\alpha_i) \leq \varepsilon'', \qquad (3)$$

for some $\varepsilon'' > 0$. Let Q^* be an optimal solution of (3). We now show that we can use the β-approximation algorithm for (2) to find a $\widetilde{Q} \leq \beta \cdot Q^*$ which satisfies

$$\sum_{i=1}^{n} (\lceil \widetilde{Q}\alpha_i \rceil - \widetilde{Q}\alpha_i) \leq 2 \cdot \beta \cdot \varepsilon''.$$

By trying out a polynomial number of candidates, we can assume to know a natural number $N \in \mathbb{N}$ with $N \geq Q^* \geq N/2$. Let $\varepsilon' = \varepsilon''/(N/2)$, then

$$\sum_{i=1}^{n}(\lceil Q^*\alpha_i \rceil - Q^*\alpha_i) \leq \varepsilon'' \leq Q^* \cdot \varepsilon',$$

which shows that Q^* is a solution of inequality (2). Now we compute a β-approximation $\widetilde{Q}$ of (2). Clearly $\widetilde{Q} \leq \beta \cdot Q^* \leq \beta \cdot N$ and thus

$$\sum_{i=1}^{n}(\lceil \widetilde{Q}\alpha_i \rceil - \widetilde{Q}\alpha_i) \leq \widetilde{Q} \cdot \varepsilon' \leq 2 \cdot \beta \cdot \varepsilon'', \qquad (4)$$

which is what we need.

Remember that we aim at a $2 \cdot \beta$-approximation algorithm for DDA which is

$$\min k \in \mathbb{N}\colon \exists Q \in \{1,\ldots,k\} : \sum_{i=1}^{n}(\lceil Q\alpha_i \rceil - Q\alpha_i) \leq k \cdot \varepsilon.$$

Let $\widetilde{Q}_k$ be the integer returned by the above described algorithm for approximating (3), where $\varepsilon'' := k \cdot \varepsilon$ and denote the optimum solution of (3) by Q_k^*. With binary search, we can find a k such that $\widetilde{Q}_k > \beta \cdot k$ and $\widetilde{Q}_{k+1} \leq \beta(k+1)$. Since $Q_k^* > k$, there does not exist a $Q \in \{1,\ldots,k\}$ such that

$$\sum_{i=1}^{n}(\lceil Q\alpha_i \rceil - Q\alpha_i) \leq k \cdot \varepsilon$$

holds. On the other hand, we have

$$\sum_{i=1}^{n}(\lceil \widetilde{Q}_{k+1}\alpha_i \rceil - \widetilde{Q}_{k+1}\alpha_i) \leq 2 \cdot \beta \cdot (k+1) \cdot \varepsilon$$

and $\widetilde{Q}_{k+1} \leq \beta \cdot (k+1)$ which shows that $\widetilde{Q}_{k+1}$ is a $2 \cdot \beta$-approximate solution of DDA. □

Summarizing, we have already shown the reductions in Figure 1, implying that if computing an $O(1)$-approximation to DDA is **NP**-hard to obtain, then the same holds for response time computation. What now follows is the proof, that DDA is indeed hard to solve.

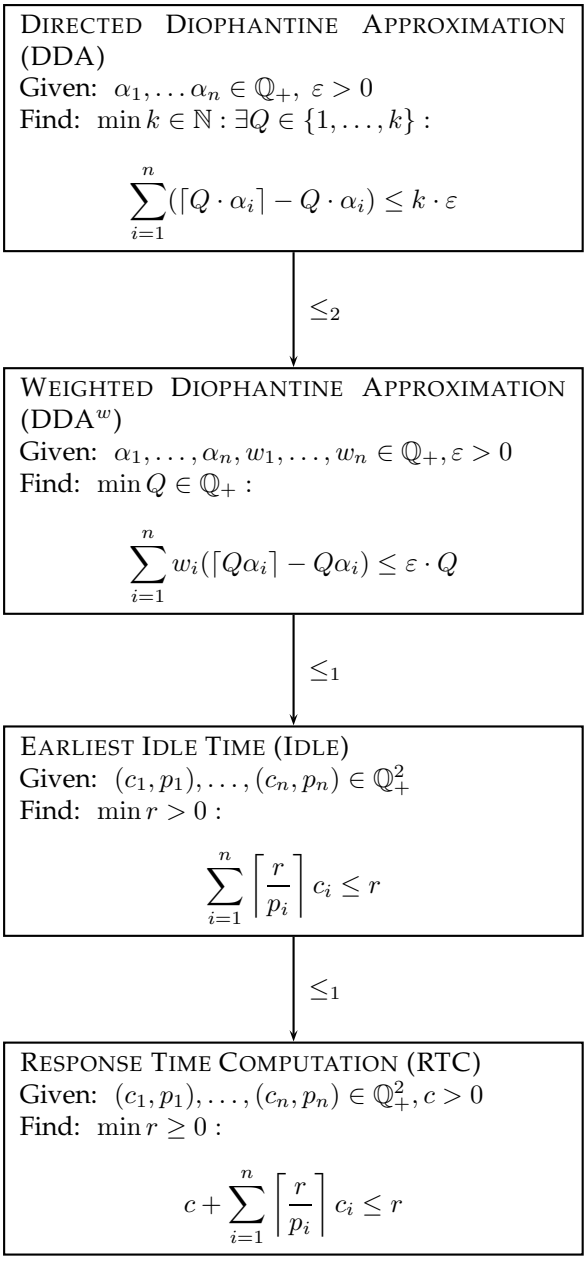

Figure 1: Overview over reductions, leading to DDA $\leq_2$ RTC

3 PIR $\leq_2$ DDA

In this section we show that DDA can be used to find the shortest, non-negative, integer vector in a hyperplane through the origin. More formally, we consider the following problem.

POSITIVE INTEGER RELATION (PIR)
Given a hyperplane $\sum_{i=1}^{n} a_i x_i = 0$, find that vector $x \in \mathbb{Z}_+^n \backslash \{\mathbf{0}\}$ on the hyperplane, which minimizes $\|x\|_1 = \sum_{i=1}^{n} x_i$.

The following proof is a modification of a proof of Lagarias [Lag85] used for giving a reduction from *shortest integer relation* w.r.t. ℓ_∞-norm to simultaneous diophantine approximation w.r.t. ℓ_∞-norm. Lagarias reduced shortest integer relation without the positiveness constraint to the classical diophantine approximation problem, where the rounding operation is the replacement with the nearest integer. Our problem PIR is an adaption of shortest integer relation which takes care of the fact that the rounding operation in DDA is the nearest larger integer. In fact we further adapt the proof of Lagarias such that it works for the accumulation of errors, i.e., for the ℓ_1-norm. The proof of Lagarias was also used by [RS96] to show that simultaneous diophantine approximation is intractable w.r.t. approximations.

Let $[x] := \lceil x \rceil - x$ be the distance of x to the next larger integer. In case that $x \in \mathbb{Q}^n$ is a vector, we define $[x] := \sum_{i=1}^{n} [x_i]$ to be the accumulated distance of the entries to the next larger integer.

Define OPT_{PIR} to be the length $\|x^*\|_1$ of an optimal solution x^* to PIR. For given ε the number OPT_{DDA} denotes the smallest integer k such that there is a $Q \in \{1, ..., k\}$ with $[Q\alpha] \leq k\varepsilon$. Using linear programming [Kha79] we can compute a fractional solution x'/D in the hyperplane $a^T x = 0$ with $x' \in \mathbb{Z}_+^n$ and $D \in \mathbb{N}$ (both of polynomial encoding size). Then x' is an (in general extremely bad) integer solution. However from now on, we need to consider only PIR solutions whose values are upperbounded by $\rho := \|x'\|$. The precise claim that we are going to show is as follows

Theorem 5. *(*PIR $\leq_2$ DDA*). Given a* PIR *instance, there is a* DDA *instance such that a* PIR *solution of value $k \in \mathbb{N}$ implies the existence of a* DDA *solution of value at most $k \cdot N$, while a* DDA *solution of value kN can be turned efficiently into a* PIR *solution of value $2k$. Here, N is a number, depending on the instance and $k \leq \rho$ with ρ as defined above.*

Clearly this theorem implies that $OPT_{\text{PIR}} \leq OPT_{\text{DDA}}/N \leq 2 \cdot OPT_{\text{PIR}}$ as well as that a β-approximation algorithm for DDA can be used to construct a 2β-approximation algorithm for PIR.

Denote $A := \rho \sum |a_j|$. Choose different primes $p, q_1, ..., q_n$, such that $q_1, \ldots, q_n$ are sufficiently close to each other. More precisely we demand that

1. $A < p^R < q_1^T < q_2^T < ... < q_n^T < 2 \cdot q_1^T$
2. p and all q_i are co-prime to all a_j
3. $q_1^T > 2\rho n \cdot p^R$

for suitable choices of $R, T \in \mathbb{N}$.

It is shown in [Lag85, RS96] that such prime numbers (having even stronger properties) exist and can be computed in polynomial time. Furthermore the values of R and T are both bounded by a polynomial in the input size. For the sake of completeness the proof can be found in the appendix.

The following system of congruences appears already in [MA78] and is crucial for the reduction.

$$r_j \equiv_{q_i^T} 0 \ \forall i \neq j \tag{5}$$
$$r_j \equiv_{p^R} a_j \tag{6}$$
$$r_j \not\equiv_{q_j} 0 \tag{7}$$

Since the moduli q_i^T, p^R are co-prime there are solutions for r_j, by the *Chinese Remainder Theorem*, see e.g. [NZM91]. Choose the smallest possible solution for r_j.

There is also an efficient way to compute r_j. Define $B := \prod_{j=1}^{n} q_j^T$, then the Chinese Remainder Theorem allows to compute some $r'_j \leq p^R B/q_j^T$, which simultaneously solve (5) and (6). Then there are two possibilities: Either we have $r'_j \not\equiv_{q_j} 0$, then $r_j := r'_j$ is a suitable choice. Otherwise we have $r_j := r'_j + p^R B/q_j^T \not\equiv_{q_j} 0$, while (5) and (6) still hold. In any case $r_j \leq 2p^R B/q_j^T$.

We need the following observation

Lemma 6. *The systems*

$$\underbrace{\sum_{j=1}^{n} x_j a_j = 0 \quad x \in \mathbb{Z}_+^n \quad 1 \leq \|x\|_1 \leq \rho}_{(I)} \quad \text{and} \quad \underbrace{\sum_{j=1}^{n} x_j r_j \equiv_{p^R} 0 \quad x \in \mathbb{Z}_+^n \quad 1 \leq \|x\|_1 \leq \rho}_{(II)}$$

have the same set of solutions.

Proof. Since $a_j \equiv_{p^R} r_j$, each solution x for (I) is a solution for (II). Vice versa, let x be a solution for (II), thus $\sum_{j=1}^{n} x_j r_j \equiv_{p^R} 0$. Due to $a_j \equiv_{p^R} r_j$ congruence $\sum_{j=1}^{n} x_j a_j \equiv_{p^R} 0$ holds. But we have

$$|\sum_{j=1}^{n} x_j a_j| \leq \underbrace{\underbrace{\|x\|_1}_{\leq \rho} \cdot \sum_{j=1}^{n} |a_j|}_{\leq A} < p^R$$

thus $\sum_{j=1}^{n} x_j a_j = 0$. We conclude that x solves (I). □

Basically this lemma allows us, to replace each a_j by a value r_j, having additional properties. This procedure will pay off later.

By $r_j^* \in \mathbb{Z}_{q_j^T}$ we denote the unique value s.t. $r_j \cdot r_j^* \equiv_{q_j^T} -1$ (this must exist since $r_j \not\equiv_{q_j} 0$ implies that $\gcd(r_j, q_j^T) = 1$). Define $N := \sum_{j=1}^n r_j$, then the DDA-instance for the reduction is

$$\begin{aligned} \alpha_0 &:= \frac{1}{p^R} \\ \alpha_j &:= \frac{r_j^*}{q_j^T} \quad \forall j = 1, ..., n \\ \varepsilon &:= \frac{1}{N q_1^T}. \end{aligned}$$

To give some intuition behind this system: Since all q_j^T are co-prime, there is a one-to-one correspondence between solutions x and good diophantine approximations Q. We will see that x lies on the hyperplane $a^T x = 0$ if and only if the corresponding Q is a multiple of p^R. Moreover the distance of $Q\alpha_j$ to the next larger integer will be proportional to x_j.

Theorem 7. *If there exists an $x \in \mathbb{Z}_+^n \backslash \{\mathbf{0}\}$ with $a^T x = 0$ and $\|x\|_1 = k$, then one has $OPT_{\text{DDA}} \leq k \cdot N$.*

Proof. Let $x \in \mathbb{Z}_+^n \backslash \{\mathbf{0}\}$ be that PIR solution with $k := \|x\|_1 = OPT_{\text{PIR}}$. It suffices to prove the existence of a $Q \in \{1, ..., k \cdot N\}$ with $[Q\alpha] \leq k/q_1^T = kN \cdot \varepsilon$. We choose $Q := \sum_{j=1}^n \underbrace{x_j}_{\geq 0} \underbrace{r_j}_{>0} > 0$. Clearly one has

$$Q = \sum_{j=1}^n x_j \cdot r_j \leq \|x\|_1 \cdot N = k \cdot N.$$

thus Q is within the feasible bounds. It remains to show that $Q\alpha$ gives a good approximation. Note that

$$[Q\alpha_0] = \left[\frac{\sum_{j=1}^n x_j r_j}{p^R} \right] = 0,$$

due to the reason that $a^T x = 0$ and therefore $\sum_{j=1}^n r_j x_j \equiv_{p^R} 0$ (see Lemma 6). Furthermore we derive that

$$\begin{aligned} [Q\alpha] &= \underbrace{[Q\alpha_0]}_{=0} + \sum_{i=1}^n [Q\alpha_i] = \sum_{i=1}^n \left[r_i^* \frac{\sum_{j=1}^n x_j \cdot r_j}{q_i^T} \right] \\ &= \sum_{i=1}^n \left[\frac{x_i r_i r_i^*}{q_i^T} \right] \overset{0 \leq x_i \ll q_i^T}{=} \sum_{i=1}^n \frac{x_i}{q_i^T} \leq \frac{k}{q_1^T} \end{aligned}$$

using that $r_j \equiv_{q_i^T} 0$ for $i \neq j$ and $r_i \cdot r_i^* \equiv_{q_i^T} -1$. The claim then follows. □

Next, we show the reverse direction.

Theorem 8. *Given a* DDA *solution Q of value kN for $k \in \mathbb{N}$, one can efficiently derive a* PIR *solution of cost at most $2k$.*

Proof. Suppose there is a number $Q \in \{1, ..., kN\}$ with $[Q\alpha] \leq \varepsilon kN = \frac{k}{q_1^T}$, then we have to show the existence of an integer vector $x \geq 0$ with $a^T x = 0$ and $1 \leq \|x\|_1 \leq 2k$. Since we already know a PIR solution of cost ρ, we may suppose that $k \leq \rho/2$. Assume for contradiction, that Q is not a multiple of p^R, then

$$[Q\alpha] \geq [Q\alpha_0] = \left[\frac{Q}{p^R} \right] \geq \frac{1}{p^R} \overset{q_1^T > \rho \cdot p^R \geq k p^R}{>} \frac{k}{q_1^T}.$$

Thus it follows that $Q \equiv_{p^R} 0$.

Compute $\hat{x}_j := Q \cdot (-r_j^*) \bmod q_j^T$ (such that $0 \leq \hat{x}_j < q_j^T$). We will show that since Q yields a good approximation to α, the vector $\hat{x}$ is a short vector on the hyperplane $a^T x = 0$. Using $q_i^T < 2q_1^T$ for $i = 1, \ldots, n$ we obtain

$$\begin{aligned} \frac{\|\hat{x}\|_1}{q_1^T} &\leq 2 \sum_{j=1}^n \frac{\hat{x}_j}{q_j^T} \\ &= 2 \sum_{j=1}^n \left[\frac{-\hat{x}_j}{q_j^T} \right] \\ &= 2 \sum_{j=1}^n \left[\frac{Q \cdot r_j^*}{q_j^T} \right] \\ &= 2[Q\alpha] \\ &\leq \frac{2k}{q_1^T} \end{aligned}$$

thus in fact the candidate solution $\hat{x}$ is short: $\|\hat{x}\|_1 \leq 2k$.

It remains to show, that $\hat{x}$ lies on the hyperplane $a^T x = 0$. Multiplying equation $Q \cdot (-r_j^*) \equiv_{q_j^T} \hat{x}_j$ with r_j yields

$$Q \equiv_{q_j^T} Q(-r_j^*) r_j \equiv_{q_j^T} \hat{x}_j r_j$$

Recall that $B = \prod_{j=1}^n q_j^T$. See the following implication

$$\left[\begin{array}{lll} Q & \equiv_{q_j^T} & \hat{x}_j r_j \quad \forall j \\ 0 & \equiv_{q_i^T} & \hat{x}_j r_j \quad \forall i \neq j \end{array} \right] \Rightarrow \left[Q \equiv_B \sum_{j=1}^n \hat{x}_j r_j \right]$$

Next, we need to compare Q and B. Plugging in the bound on r_j we derive

$$N = \sum_{j=1}^n r_j \leq \sum_{j=1}^n \frac{2p^R B}{q_j^T} \leq \frac{2np^R B}{q_1^T} < \frac{B}{\rho} \leq \frac{B}{k},$$

thus $Q \leq k \cdot N < B$. Furthermore

$$\sum_{j=1}^{n} \hat{x}_j r_j \leq \underbrace{\max_{j=1,\ldots,n} \hat{x}_j}_{\leq k} \cdot \underbrace{\sum_{j=1}^{n} r_j}_{=N} < B.$$

But then $Q = \sum_{j=1}^{n} \hat{x}_j r_j$ holds (not only $\equiv_B$). Clearly

$$0 \equiv_{p^R} Q \equiv_{p^R} \sum_{j=1}^{n} \hat{x}_j r_j,$$

thus $\hat{x}$ is a solution of (II) (recall that $\|\hat{x}\|_1 \leq k \leq \rho$) and due to Lemma 6 also for (I). To see that $\hat{x} \neq \mathbf{0}$, note that otherwise $Q = \sum_{j=1}^{n} 0 \cdot r_j = 0$, contradicting $Q > 0$. □

4 k-SET COVER $\leq_2$ PIR

Recall that k-SET COVER is the well-known SET COVER problem with the additional constraint, that the cardinality of all sets is bounded by a constant k.

> k-SET COVER
> Given sets $S_1, \ldots, S_n$ with cardinality $|S_i| \leq k$ for $i = 1, \ldots, n$ over a ground set $U = \bigcup_{i=1}^{n} S_i$, find $\min\{|I| \mid \bigcup_{i \in I} S_i = U\}$

In this section we convey the known inapproximability results for k-SET COVER to PIR. Trevisan [Tre01] observed that

> Unless **NP** = **P**, there is a universal constant c, such that each fixed k, k-SET COVER cannot be approximated within a factor of $\ln k - c \ln \ln k$. This is an implicit result of the proof in [Fei98].

Theorem 9. *For any fixed $k \in \mathbb{N}$ one has k-SET COVER $\leq_2$ PIR.*

Proof. Kannan [Kan83] showed that given a subspace $Ax = \mathbf{0}$, one can easily find a vector $a \in \mathbb{Z}^n$ whose encoding size is polynomial (in $\log \mu$ and the encoding size of the matrix A), such that

$$B_\mu \cap \{x \in \mathbb{Z}^n \mid Ax = \mathbf{0}\} = B_\mu \cap \{x \in \mathbb{Z}^n \mid a^T x = 0\}$$

(here B_μ denotes the ball of radius μ around the origin). Thus we may assume to have some β-approximation algorithm for finding short non-negative integer vectors in a subspace (which is not a hyperplane). In fact a choice of $\mu = \beta n$ suffices for our reduction from k-SET COVER. Given a constant k, a k-SET COVER instance consists of sets $S_1, ..., S_n \subseteq U$ with $\bigcup_{i=1}^{n} S_i = U$ and $|S_i| \leq k$, $|U| = m$. We call a family of sets *complete*, if for sets S_i all subsets $S \subseteq S_i$ are also contained in the instance. After adding at most $2^k n = O(n)$ sets we may assume that the instance is complete. Of course this does not change the minimal number OPT_{SC} of sets, needed to cover U. Moreover any solution for the complete instance can be turned into a solution for the original instance which has at most of the same cost. This can be done by replacing each "artificial" set in the solution by the corresponding superset.

Consider the set of all solutions $(x, y) \in (\mathbb{Z}_+^n \times \mathbb{Z}_+) \backslash \{\mathbf{0}\}$ in the subspace

$$x_1 \cdot \chi(S_1) + ... + x_n \cdot \chi(S_n) = y \cdot \mathbf{1}$$

where $\chi(S_i)$ denotes the characteristic vector of S_i. We need to show two claims

(1) If there is a solution of cost α for k-SET COVER, then there is a solution of cost at most 2α for PIR.

(2) From a PIR solution of cost α one can efficiently derive a cheaper k-SET COVER solution.

We begin by showing (1). Let $I \subseteq \{1, \ldots, n\}$ be a k-SET COVER solution. Since the instance is complete, we may assume that each element in U is covered exactly once in $\bigcup_{i \in I} S_i$. Denote $\alpha := |I|$. Then

$$x_i = \begin{cases} 1 & \text{if } i \in I \\ 0 & \text{otherwise} \end{cases} \quad \text{and} \quad y = 1$$

is a feasible PIR solution of cost $\alpha + 1 \leq 2\alpha$.

For (2), let (x, y) be any PIR instance of cost $\alpha = y + \sum_{i=1}^{n} x_i$. Since we have a nontrivial solution and all characteristic vectors are non-negative, we must have $y \geq 1$. Then $I := \{i \mid x_i \geq 1\}$ is clearly a k-SET COVER solution of cost $|I| \leq \sum_{i=1}^{n} x_i \leq \alpha$. This shows the claim. □

To keep polynomiality one can choose $k = \log n$, deriving that PIR cannot be approximated within a factor of $\log \log n - c \log \log \log n$.

Note that for all integer linear programs with only a constant number of equations an optimum solution can be found in pseudo-polynomial time, thus the same holds for PIR. In that sense the last result is remarkable, since such problems often admit an FPTAS.

The above proof yields the last building block for proving inapproximability of DDA, see Figure 2 for an overview.

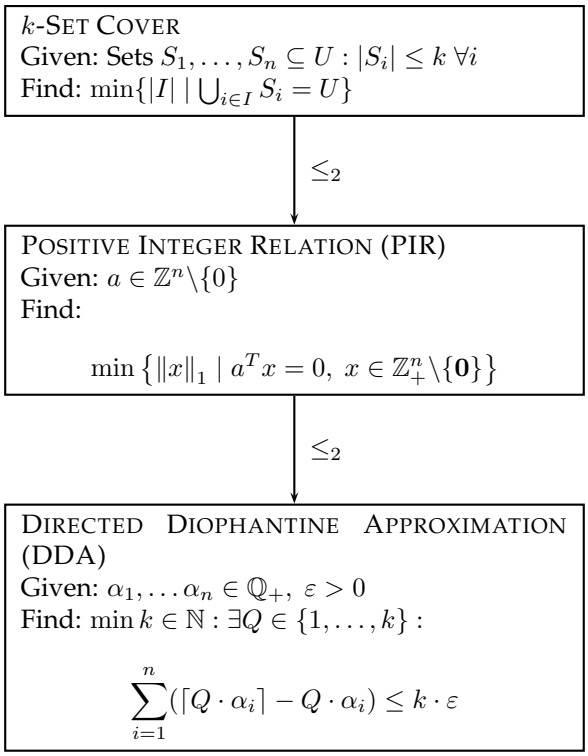

Figure 2: Overview over 2nd part of reductions

Note that the related shortest integer relation cannot be approximated within a factor of $2^{\log^{0.5-\gamma} n}$ in the ℓ_∞-norm for any $\gamma > 0$, unless $\mathbf{NP} \subseteq \mathbf{DTIME}(n^{\text{polylog}(n)})$ [RS98]. On the other hand shortest integer relation can be approximated within a factor of $\sqrt{n} \cdot 2^{n/2}$ using the famous LLL-algorithm [LLL82]. No such result is known for PIR, thus the following question arises.

> Is there a $2^{p(n)}$-approximation algorithm for PIR for some polynomial p?

5 Conclusions and open question

We showed that response time computation for Rate-monotonic, preemptive scheduling is **NP**-hard. However, what we did not show is that the feasibility test itself is **NP**-hard. The problem is that although it is **NP**-hard to decide, whether all jobs of a given task meet its deadlines, it might be the case for some of the constructed instances, that prior tasks are obviously infeasible. In fact, what one has to do is to design a suitable instance, for which all but the last task are clearly feasible.

Moreover the utilization of the instances designed in the reduction have utilizations, very close to one. The question arises, whether the response time can be computed in polynomial time, if the utilization is upper bounded by $1 - \varepsilon$ for any constant $\varepsilon > 0$.

References

[ABD+95] Neil C. Audsley, Alan Burns, Robert I. Davis, Ken Tindell, and Andy J. Wellings. Fixed priority pre-emptive scheduling: An historical perspective. *Real-Time Systems*, 8(2-3):173–198, 1995.

[FB05] Nathan Fisher and Sanjoy Baruah. A fully polynomial-time approximation scheme for feasibility analysis in static-priority systems with arbitrary relative deadlines. In *ECRTS '05: Proceedings of the 17th Euromicro Conference on Real-Time Systems (ECRTS'05)*, pages 117–126, Washington, DC, USA, 2005. IEEE Computer Society.

[Fei98] Uriel Feige. A threshold of ln n for approximating set cover. *Journal of the ACM*, 45(4):634–652, 1998.

[HB88] D. R. Heath-Brown. The number of primes in a short interval. *J. Reine Angew. Math.*, 389:22–63, 1988.

[HBI79] D. R. Heath-Brown and H. Iwaniec. On the difference between consecutive primes. *Bull. Amer. Math. Soc. (N.S.)*, 1(5):758–760, 1979.

[HW97] Henk and Weismantel. Test sets of the knapsack problem and simultaneous diophantine approximation. In *ESA: Annual European Symposium on Algorithms*, 1997.

[HW02] Martin Henk and Robert Weismantel. Diophantine approximations and integer points of cones. *Combinatorica*, 22(3):401–407, 2002.

[JP86] Mathai Joseph and Paritosh K. Pandya. Finding response times in a real-time system. *Computer Journal*, 29(5):390–395, 1986.

[Kan83] Ravindran Kannan. Polynomial-time aggregation of integer programming problems. *Journal of the ACM*, 30(1):133–145, January 1983.

[Kha79] Khachiyan, L. G. A polynomial algorithm for linear programming. *Soviet Math. Doklady*, 20:191–194, 1979. (Russian original in Doklady Akademiia Nauk SSSR, 244:1093–1096).

[Lag85] Lagarias. The computational complexity of simultaneous diophantine approximation problems. *SICOMP: SIAM Journal on Computing*, 14, 1985.

[LL73] C. L. Liu and James W. Layland. Scheduling algorithms for multiprogramming in a hard-real-time environment. *J. ACM*, 20(1):46–61, 1973.

[LLL82] A. K. Lenstra, H. W. Lenstra Jr., and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:515–534, 1982.

[LSD89] John P. Lehoczky, Lui Sha, and Y. Ding. The rate monotonic scheduling algorithm: Exact characterization and average case behavior. In *IEEE Real-Time Systems Symposium*, pages 166–171, 1989.

[MA78] Kenneth L. Manders and Leonard Adleman. *NP*-complete decision problems for binary quadratics. *Journal of Computer and System Sciences*, 16(2):168–184, April 1978.

[NZM91] Ivan Niven, Herbert S. Zuckerman, and Hugh L. Montgomery. *An Introduction to the Theory of Numbers, fifth edition*. Wiley, 1991.

[RS96] Carsten Rössner and Jean-Pierre Seifert. Approximating good simultaneous Diophantine approximations is almost NP-hard. In *Mathematical foundations of computer science 1996 (Cracow)*, volume 1113 of *Lecture Notes in Comput. Sci.*, pages 494–505. Springer, Berlin, 1996.

[RS98] C. Rössner and J.P. Seifert. On the hardness of approximating shortest integer relations among rational numbers. *TCS: Theoretical Computer Science*, 209, 1998.

[Tre01] Trevisan. Non-approximability results for optimization problems on bounded degree instances. In *STOC: ACM Symposium on Theory of Computing (STOC)*, 2001.

Appendix

We still have to show that suitable prime numbers for the reduction PIR $\leq_2$ DDA exist and that they can be found in polynomial time. The proof follows closely [RS96].

Lemma 10. *Let m be the encoding size of the* PIR *instance and let ρ be a value, whose encoding length is bounded by a polynomial in m. One can find different primes $p, q_1, ..., q_n$ as well as natural numbers R and T in polynomial time, such that*

1. $A := \rho \sum |a_j| < p^R < q_1^T < q_2^T < ... < q_n^T < 2 \cdot q_1^T$
2. *p and all q_i are co-prime to all a_j*
3. $q_1^T > 2\rho n \cdot p^R$
4. *$T, R, p, q_1, \ldots, q_n$ are bounded by a polynomial in m.*

Proof. The number of different prime factors appearing in some a_j is clearly bounded by m. Due to the prime number theorem, see, e.g. [NZM91], the first $m+1$ prime numbers can be computed in polynomial time by testing the first $O(m \log(m))$ natural numbers. Choose p among these primes, s.t. $\gcd(p, a_j) = 1$ for $i = 1, \ldots, n$. Compute the smallest R and T (for example using binary search) such that $p^R > A$ as well as $2^T > 2\rho n \cdot p^R$ and $T \geq m$. Clearly both, R and T are polynomially bounded in m. It remains to find prime numbers $q_1, \ldots, q_n$, which are sufficiently close to each other. Fortunately, we may use a very deep result in number theory at this point.

Theorem 11. *[HBI79, HB88] For each $\delta > \frac{11}{20}$ there exists a constant c_δ such that for all $z > c_\delta$ the interval $[z, z^\delta]$ contains a prime.*

Consider an arbitrary $i \in \{0, \ldots, T^2 - 1\}$. Choose $\delta = \frac{3}{5}$ then for sufficiently large m, there is a prime between each $z := T^{20} + i(2T)^{12}$ and $z + z^\delta$ with

$$z^\delta \leq (T^{20} + i(2T)^{12})^{3/5} < ((2T)^{20})^{3/5} = (2T)^{12}.$$

Thus there must be a prime in each interval $[T^{20} + i(2T)^{12}, T^{20} + (i+1)(2T)^{12}[$. Since T is polynomially bounded, we can compute $m + n + 1 \leq T^2$ distinct primes from $[T^{20}, T^{20} + T^2(2T)^{12}] \subseteq [T^{20}, T^{20} + T^{15}]$. Select n of these primes, which are co-prime to p and all a_j and denote them by $q_1, \ldots, q_n$. Using $1 + x \leq e^x$ for $x \in \mathbb{R}$ we obtain

$$\frac{q_n^T}{q_1^T} \leq \left(\frac{T^{20} + T^{15}}{T^{20}}\right)^T \leq \left(1 + \frac{1}{T^5}\right)^T \leq e^{T/T^5} < 2$$

The claim then follows. □