Sensitivity Analysis for Arbitrary Activation Patterns in Real-time Systems

Moritz Neukirchner, Sophie Quinton, Tobias Michaels, Philip Axer, Rolf Ernst
Institut für Datentechnik und Kommunikationsnetze
Technische Universität Braunschweig
Braunschweig, Germany
neukirchner|quinton|michaels|axer|ernst@ida.ing.tu-bs.de

***Abstract*—Response time analysis, which determines whether timing guarantees are satisfied for a given system, has matured to industrial practice and is able to consider even complex activation patterns modelled through arrival curves or minimum distance functions. On the other side, sensitivity analysis, which determines bounds on parameter variations under which constraints are still satisfied, is largely restricted to variation of single-valued parameters as e.g. task periods.**

In this paper we provide a sensitivity analysis to determine the bounds on the admissible activation pattern of a task, modelled through a minimum distance function. In an evaluation on a set of synthetic testcases we show, that the proposed algorithm provides significantly tighter bounds, than previous exact analyses, that determine allowable parametrizations of activation patterns.

I. Introduction & Related Work

Response time analysis [1], [2] and performance analysis [3], [4] provide helpful techniques to determine whether for a given real-time system all timing constraints are satisfied. Use of such techniques is current industrial practice in the design of real-time systems, as e.g. automotive systems.

However, these analyses only yield whether a system meets all deadlines for a given configuration (i.e. set of parameters). They provide no indication on how much parameters, as e.g. task periods, may be changed without violating constraints. This is the domain of *sensitivity analysis*. Sensitivity analysis becomes relevant, whenever the specification is inaccurate. This may be the case e.g. in early phases of a design process, or for mixed-criticality systems, where the specification of low-criticality components is not fully trusted. Sensitivity analysis determines borderline cases under which a configuration still meets all constraints.

Analytical derivation of sensitivity bounds [5], [6], [7] is hard and was only achieved under restrictions on the system model (e.g. periodic tasks). Other approaches derive sensitivity bounds for certain design parameters through search over the parameter space (e.g. [2], [8], [9], [10]), with schedulability tests or response time analysis as feasibility evaluation. The search-based approaches come at higher computational complexity than the analytical methods. A third approach is taken by [11] which determines sensitivity bounds with an iterative algorithm, which starts from an infeasible configuration and consecutively adapts system parameters to make the configuration feasible. The design of the algorithm avoids the complete search of the parameter space. All of these approaches have in common, that they address sensitivity analysis with respect

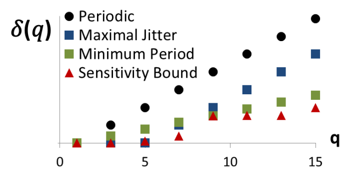


Fig. 1: Example of actual sensitivity bound

to single-valued parameters as e.g. execution times, processor speed (common scaling factor), task periods or activation jitter.

However, when considering the activation pattern of a task through standard event models as e.g. period and jitter, the results may be overly conservative - even for exact analysis. We shortly explain this deficiency. Arbitrary activation patterns can be modelled with *minimum distance functions* $\delta(q)$, which denote the minimum distance between any q consecutive events. A dual representation to minimum distance functions are *arrival curves* [3]. For illustration of the deficiency we refer to figure 1. The original activation pattern specification of a task may be given through an activation period P. The corresponding minimum distance function is shown with black dots in Fig. 1, i.e. $\delta(q) = (q-1)*P$. A periodic plus jitter event model is expressed through $\delta(q) = \max(0, (q-1)*P-J)$. A *jitter* sensitivity analysis would search for the maximum J, i.e. the maximum shift of the function, that may be allowed. This is shown as blue squares. A *period* sensitivity analysis would search for the minimal allowable period, i.e. the minimal slope, shown in green squares. Instead of these parametrizations through standard event models, the actual minimally allowed distances between subsequent events, as imposed through the system constraints, may look as shown in red triangles. They mark the actual sensitivity bound. Thus, although a sensitivity analysis for parametrized activation patterns may be exact, in general it is still conservative.

In this paper we aim to determine an arbitrary activation pattern sensitivity bound, which does not rely on a parametrization, but explicitly yields the values of the minimum distance function. Such arbitrary activation pattern sensitivity bounds can be considered in performance analysis [3], [4] or be used for monitoring as in [12].

The remainder of this paper is structured as follows. In Sec. II we introduce the system model and in Sec. III we explain the response-time analysis which we base our sensitivity analysis upon. In the main part in Sec. IV we introduce our sensitivity analysis. We evaluate this analysis in Sec. V based on a synthetic set of testcases and exact reference algorithms,

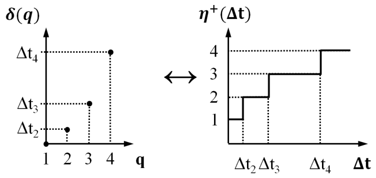


Fig. 2: Example of δ and its corresponding η^+

which work on parametrizations of activation patterns. We conclude the paper in Sec. VI.

II. System Model

We consider a uniprocessor system on which a set of k tasks $\{\tau_1, \dots, \tau_k\}$ is executing. The processor is scheduled according to a preemptive work-conserving scheduling policy. The set $\mathcal{I}_i$ shall denote the set of tasks, which may preempt the scheduling of task τ_i, i.e. $\mathcal{I}_i$ is the set of interferers.

A task is activated by an *activating event* and, when scheduled by the processor, executes for at most its worst-case execution time (WCET) C_i. We refer to each activation of a task as *instance*. As event model we use *minimum distance functions* [13].

Definition 1 (Minimum Distance Function) *The minimum distance function $\delta(q)$ is a pseudo super-additive*[1] *function, which returns a lower bound on the time interval between the first and the last event of any sequence of q event occurrences.*

Thus, a minimum distance function is an event model, which describes the minimal time, in which q events may occur, e.g. $\delta(3) = 10$ denotes, that the first and the last event of any sequence of three events are at least 10 time units apart. We denote the minimum distance function describing the activation of task τ_i by δ_i. For notational simplicity we use bold font to denote tuples, e.g. the tuple of all minimum distance functions $\boldsymbol{\delta} = \langle \delta_1, \dots, \delta_k \rangle$. At points we use a dual representation of minimum distance functions – the *maximum arrival functions*, which are well-known from Real-time Calculus [3].

Definition 2 (Maximum Arrival Function) *The maximum arrival function $\eta^+(\Delta t)$, returns an upper bound on the number of events that can arrive within any half-open time window $[t, t + \Delta t)$.*

An arrival curve returns the maximal number of events that may occur in any time window of size Δt, e.g. $\eta^+(10) = 3$ denotes, that in any time interval of 10 time units at most 3 events may occur. The η^+-function can be derived directly from its corresponding δ-function through

$$\eta^+(\Delta t) = \max_{n \in \mathbb{N}^+} \{ n \ : \ \delta(n) < \Delta t \} \tag{1}$$

An example of a δ function and its corresponding arrival curve is given in figure 2.

To reason about sensitivity, each task τ_i is associated with an arbitrary relative deadline D_i, i.e. a constraint on its worst-case response time (WCRT).

[1]With *pseudo super-additive* we denote the property of a function δ such that $\forall a, b \in \mathbb{N}^+ : \delta(a + b - 1) \geq \delta(a) + \delta(b)$. It corresponds to the property of "good" arrival functions in [14].

III. Response-Time Analysis

The considered class of preemptive, work-conserving schedulers can be analyzed with the *busy-window analysis* (also "busy-period" analysis). It allows to calculate an upper bound on the time interval the processor is busy processing a task τ_i and its interferers $\mathcal{I}_i$ [15]. This level-i busy-period was later extended to arbitrary event models [13] and [16] introduced the level-i q-event busy-time, which allows to reason about the busy-time of a specific instance q of τ_i. Based on busy-times one can calculate an upper bound on a task's WCRT. The sensitivity analysis, which we present in Sec. IV is built on certain properties of the busy-window analysis. Therefore we will review the details of the busy-window analysis.

The *multiple-event busy-window* (or multiple-event busy time) is formally defined by [16] as

Definition 3 (Multiple-Event Busy-Window) *The maximum q-event busy-window $B_i(q, \boldsymbol{\delta})$ is given through the following iterative formula, which is calculated until convergence.*

$$B_i(q, \boldsymbol{\delta}) = q \cdot C_i + \sum_{j \in \mathcal{I}_i} \eta_j^+(B_i(q, \boldsymbol{\delta})) \cdot C_j \tag{2}$$

The maximum q-event busy time $B_i(q, \boldsymbol{\delta})$ describes an upper bound on the amount of time a resource requires to service q activations of task τ_i, assuming that the processor is initially idle. Note, that $B_i(q, \boldsymbol{\delta})$ does not depend on the activation event model η_i^+ (and thus δ_i) of the analyzed task, but only on the activation event models of its interferers η_j^+ and the worst-case execution times of itself C_i and of its interferers C_j, i.e. for any two activation pattern specifications $\boldsymbol{\delta} = \langle \delta_1, \dots, \delta_i, \dots, \delta_n \rangle$, $\boldsymbol{\delta}' = \langle \delta_1, \dots, \delta_i', \dots, \delta_n \rangle$ that differ only in position i

$$B_i(q, \boldsymbol{\delta}) = B_i(q, \boldsymbol{\delta}') \tag{3}$$

We use this property later in Sec. IV.

The calculation of the busy-window assumes that all q activations arrive "sufficiently early", i.e. prior to the completion of its preceding event (the $(q-1)$-event busy-time), i.e.

$$\delta_i(q) \leq B_i(q - 1, \boldsymbol{\delta}) \tag{4}$$

We will denote the number of instances of task τ_i in the maximum busy-window as Q_i, i.e. it is the latest activation that comes sufficiently early

$$Q_i = \max \left(n : \forall q \in \mathbb{N}^+, q \leq n : \delta_i(q) \leq B_i(q - 1, \boldsymbol{\delta}) \right) \tag{5}$$

An example schedule depicting q-event busy-windows is given in Fig. 3 for task τ_3 with interferers τ_1 and τ_2. Two activations come sufficiently early, to fall into the busy-window, as $\delta_3(3) > B_3(2, \boldsymbol{\delta})$, thus $Q_i = 2$. The 1-event and 2-event busy-times are shown in the figure.

Based on the busy-window length one can derive bounds on the response time of a task. The worst-case response time R_i of a task τ_i is an upper bound on the time interval between any activation of task τ_i and the corresponding completion time. A task τ_i meets its deadline D_i if its worst-case response time does not exceed the deadline, i.e.

$$D_i \geq R_i \tag{6}$$

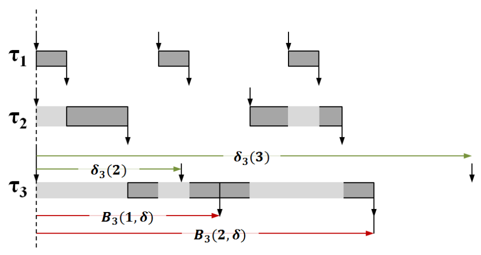


Fig. 3: Example of busy-window

The worst-case response time of the q-th instance in a busy-window is bounded by

$$R_i(q) = B_i(q, \boldsymbol{\delta}) - \delta_i(q) \tag{7}$$

The worst-case response time R_i can be found among the first Q_i busy-windows [15], [17].

$$R_i = \max_{q \in [1, Q_i]} (R_i(q)) \tag{8}$$

With this analysis technique it is possible to derive bounds on the response time of a task, based on a specification of all tasks' activation patterns and their WCETs. In sensitivity analysis we aim to derive the inverse, i.e. based on a specification of constraints we aim to derive a minimum activation pattern specification $\delta_i(q)$, for which all constraints still hold. In contrast to previous approaches we do not attempt, to derive a parametrization of $\delta_i(q)$ (e.g. through period and jitter) but to explicitly derive a lower bound for all individual values of the minimum distance function. Now, we address this problem.

IV. Sensitivity Analysis of Arbitrary Activation Patterns

In the sensitivity analysis of arbitrary activation patterns we want to find minimal values for $\delta_i(q)$ of a given task τ_i, such that the constraints of all tasks are still satisfied.

Our approach to determining minimal values of $\delta_i(q)$ is by posing constraints on the allowed values, while these constraints are based on the above described busy-window analysis technique. We split the derivation of these constraints into two parts, which we first address separately. In the first part, we derive conditions such that a reduction of $\delta_i(q)$ does not violate any constraints of task τ_i. In the second part we consider the influence on other tasks and derive conditions, under which a reduction of $\delta_i(q)$ does not cause violation of constraints of any task $\tau_j \neq \tau_i$. We then compose these conditions to derive an overall sensitivity bound.

A. Self-Influence Conditions

In this section, we solely regard the effects, that changing the activation pattern δ_i of task τ_i has on satisfaction of the deadline D_i of that task, i.e. for now we neglect the influence on other tasks. First, we derive an optimal upper bound on the number of instances of τ_i that may appear in the busy-window, i.e. we determine a sensitivity bound for Q_i, which we will denote with $\bar{Q}_i$. Then, based on this bound, we construct constraints on $\delta_i(q)$, that guarantee that τ_i will meet its deadline. We start with the definition of $\bar{Q}_i$.

Definition 4 (Sensitivity bound on number of instances) *The sensitivity bound $\bar{Q}_i$ on the number of instances of task τ_i that may appear in the busy-window is defined as*

$$\begin{aligned} \bar{Q}_i = \max(n : \forall p \in \mathbb{N}^+, p \leq n \\ D_i \geq B_i(p, \boldsymbol{\delta}) - B_i(p-1, \boldsymbol{\delta})) \end{aligned} \tag{9}$$

Note, that $\bar{Q}_i$ is independent of δ_i, as $B_i(q, \boldsymbol{\delta})$ is independent of δ_i. As a consequence the sensitivity bound on the number of instances can be calculated solely based on the minimum distance functions of τ_i's interferers, the worst-case execution times and τ_i's deadline.

In the following lemma we show, that for a given system there exists an activation event model, such that the deadline of task τ_i is satisfied and that the busy-window contains $\bar{Q}_i$ instances of τ_i. Furthermore, it shows that such an event model does not exist for any value larger than $\bar{Q}_i$.

Lemma 1 (Sensitivity bound on number of instances) *$\bar{Q}_i$ is the maximal value, such that for a given system there exists a minimum distance function δ_i for task τ_i such that its WCRT constraint D_i is satisfied and the maximum busy-window contains $\bar{Q}_i$ instances of τ_i.*

Proof: The proof is split into two parts. First, we show that a minimum distance function δ_i exists, such that the WCRT constraint is satisfied and the busy-window contains $\bar{Q}_i$ instances. Second, we show, that $\bar{Q}_i$ is also the maximal value that satisfies, this condition.

Assume, that task τ_i is activated with

$$\forall q \in [1, \bar{Q}_i] \ : \ \delta_i(q) = B_i(q-1, \boldsymbol{\delta}) \tag{10}$$

$$\forall q > \bar{Q}_i \ : \ \delta_i(q) = \infty \tag{11}$$

i.e. all instances $q \in [1, \bar{Q}_i]$ arrive just at the end of the busy-window of its previous instance. All other instances $q > \bar{Q}_i$ never occur. Then, according to (4) the busy-window contains $\bar{Q}_i$ instances. The response time is given through (7), (8), thus

$$R_i = \max_{q \in [1, \bar{Q}_i]} (B_i(q, \boldsymbol{\delta}) - B_i(q-1, \boldsymbol{\delta})) \tag{12}$$

with (9) we obtain

$$D_i \geq R_i = \max_{q \in [1, \bar{Q}_i]} (B_i(q, \boldsymbol{\delta}) - B_i(q-1, \boldsymbol{\delta})) \tag{13}$$

and thus, the worst-case response time is satisfied.

We now regard the second part of the proof. In contradiction to the initial assumption assume that $\bar{Q}_i + 1$ instances appear in the busy-window and that the WCRT constraint is satisfied.

From the definition of $\bar{Q}_i$ we know that $D_i \geq B_i(j, \boldsymbol{\delta}) - B_i(j-1, \boldsymbol{\delta})$ is not satisfied for any $q > \bar{Q}_i$, thus

$$D_i < B_i(\bar{Q}_i + 1, \boldsymbol{\delta}) - B_i(\bar{Q}_i, \boldsymbol{\delta}) \tag{14}$$

The $(\bar{Q}_i + 1)$-th instance shall satisfy D_i, i.e.

$$B_i(\bar{Q}_i + 1, \boldsymbol{\delta}) - \delta_i(\bar{Q}_i + 1) \leq D_i \tag{15}$$

combining (14) and (15) yields

$$\delta_i(\bar{Q}_i + 1) > B_i(\bar{Q}_i, \boldsymbol{\delta}) \tag{16}$$

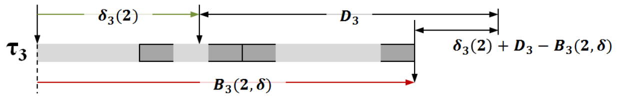


Fig. 4: Example for theorem 1

Thus, instance $\bar{Q}_i + 1$ does not come sufficiently early (4) and does not fall into the busy-window, which violates the initial assumption. ■

Lemma 1 states, that for any system we can find a minimum distance function δ_i of task τ_i such that this task's worst-case response time constraint is satisfied and the maximum busy-window contains $\bar{Q}_i$ instances of τ_i. Furthermore, it states, that this is not possible for any busy-window containing more than $\bar{Q}_i$ instances. Thus, $\bar{Q}_i$ is an optimal upper bound on the number of instances that may appear in a busy-window of τ_i without violating its deadline.

Based on the sensitivity bound on the number of instances of a given task, we now construct a minimal δ_i, such that the constraint D_i is still satisfied. We first give an intuitive example for the following theorem, before formalizing it.

We use the second instance of task τ_3 from the previous example to illustrate the idea. The 2-event busy-window of τ_3 is shown in Fig. 4. The absolute deadline of $q = 2$ is $\delta_3(2) + D_3$. Furthermore, the busy-window length is independent of $\delta_3(2)$ (see (3)) and was calculated to $B_3(2, \boldsymbol{\delta})$. The slack is given through $\delta_3(2) + D_3 - B_3(2, \boldsymbol{\delta})$. Now, $\delta_3(2)$ can be decreased, until the slack is zero. Furthermore, from lemma 1 we know, that at most $\bar{Q}_i$ instances may appear in the busy-window. Thus, this technique may only be applied to the first $\bar{Q}_i$ instances. All other instances may not come sufficiently early. Now, we formalize this approach in theorem 1.

Theorem 1 *For a given activation pattern specification $\boldsymbol{\delta} = \langle \delta_1, \ldots, \delta_n \rangle$ let a given system satisfy task τ_i's constraint on worst-case response time D_i.*

Then, D_i is also satisfied for any $\bar{\boldsymbol{\delta}}_{\boldsymbol{i}} = \langle \delta_1, \ldots, \bar{\delta}_i, \ldots, \delta_n \rangle$ that satisfies the following constraints

$$\forall q \in \mathbb{N}^+, q > \bar{Q}_i \quad : \quad \bar{\delta}_i(q) > B_i(\bar{Q}_i, \boldsymbol{\delta}) \tag{17}$$

$$\forall q \in \mathbb{N}^+, q \leq \bar{Q}_i \quad : \quad \bar{\delta}_i(q) \geq B_i(q, \boldsymbol{\delta}) - D_i \tag{18}$$

Proof: From (3) we have

$$B_i(q, \bar{\boldsymbol{\delta}}_{\boldsymbol{i}}) = B_i(q, \boldsymbol{\delta}) \tag{19}$$

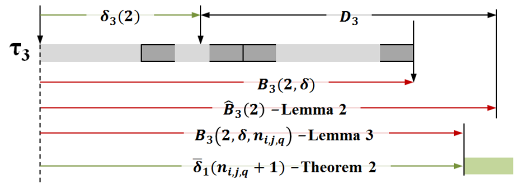
Substituting this in (17) yields

$$\forall q \in \mathbb{N}^+, q > \bar{Q}_i \; : \; \bar{\delta}_i(q) > B_i(\bar{Q}_i, \bar{\boldsymbol{\delta}}_{\boldsymbol{i}}) \tag{20}$$

Thus, taking into account the definition of "sufficiently early" (4), we conclude that for $\bar{\delta}_i(q)$ the maximum busy-window contains no more than $\bar{Q}_i$ instances of task τ_i. Thus, with lemma 1 we know that a minimum distance function δ_i exists, under which the constraint D_i is satisfied.

(18) ensures that all instances $q \in [1, \bar{Q}_i]$ satisfy the deadline D_i. A task's response time is upper bounded by (7), (8). Thus, for the case of $\bar{Q}_i$ instances

$$R_i = \max_{q \in \mathbb{N}^+ \mid q \leq \bar{Q}_i} \left(B_i(q, \bar{\boldsymbol{\delta}}_{\boldsymbol{i}}) - \bar{\delta}_i(q) \right) \tag{21}$$



$\delta_3(2)$
D_3
τ_3
$B_3(2, \delta)$
$\hat{B}_3(2)$ –Lemma 2
$B_3(2, \delta, n_{i,j,q})$ –Lemma 3
$\bar{\delta}_1(n_{i,j,q} + 1)$ –Theorem 2

Fig. 5: Derivation of Cross-Influence Constraints

With (18)

$$R_i \leq \max_{q \in \mathbb{N}^+ \mid q \leq \bar{Q}_i} (D_i) = D_i \tag{22}$$

■

Theorem 1 shows, that we can decrease the activation event model $\delta_i(q)$ of any task τ_i to a value of $\bar{\delta}_i(q)$ without violating the constraints of that task. The bounds can be calculated with knowledge of the worst-case busy-windows of the original specification, i.e. no additional iterative calculation is required. Note however, that influence on other tasks is not considered. We will address this issue in the following section.

B. Cross-Influence Conditions

In this section we analyze the effects, that changing the minimum distance function of a task τ_i has on another task τ_j's response time. Specifically, we aim to derive lower bounds on δ_i such that the worst-case response constraint D_j of task τ_j is still satisfied. For notational clarity we will refer to task τ_i as *analyzed task* and to task τ_j as *influenced task*.

The general reasoning of the construction of the lower bound on $\delta_i(q)$ is depicted in Fig. 5 on the example of task τ_1 as analyzed task, which influences the second instance of task τ_3. In lemma 2, we first derive an implicit constraint on the length of the busy-window $\hat{B}_3(2)$ of the second instance of the influenced task τ_3. Based on this implicit constraint, we derive the maximum number of instances of the analyzed task, that may appear in the 2-event busy-window of τ_3. We will denote this maximal number of instances with $\bar{n}_{i,j,q}$ (here $\bar{n}_{1,3,2}$). In lemma 3 we show, that the busy-window considering this maximal number of instances $B_3(2, \boldsymbol{\delta}, \bar{n}_{1,3,2})$ instead of $\eta_i^+(B_j(q, \boldsymbol{\delta}))$ is smaller than the busy-window constraint $\hat{B}_3(2)$. Finally, based on this number of instances $\bar{n}_{1,3,2}$, we determine a bound on $\bar{\delta}_i(\bar{n}_{i,j,q} + 1)$ (here $\bar{\delta}_1(\bar{n}_{1,3,2} + 1)$) of the analyzed task τ_i. This is given in theorem 2.

We start with the implicit busy-window constraint $\hat{B}_j(q)$.

Lemma 2 *Task τ_j satisfies its worst-case response time constraint D_j, if and only if no instance $q \in [1, Q_j]$ has a busy window larger than $\hat{B}_j(q)$, where $\hat{B}_j(q)$ is given through*

$$\hat{B}_j(q) = D_j + \delta_j(q) \tag{23}$$

Proof: The lemma directly follows from (7) and (8). ■

From this lemma, we now derive a constraint on the number of instances $\bar{n}_{i,j,q}$ of the analyzed task τ_i that may appear in the q-event busy-window of the influenced task τ_j, i.e. $\bar{n}_{i,j,q}$ shall be an upper bound on $\eta_i^+(B_j(q, \boldsymbol{\delta}))$ or $\bar{n}_{i,j,q} \geq \eta_i^+(B_j(q, \boldsymbol{\delta}))$.

The busy-window of the q-th instance of task τ_j for a given activation pattern $\boldsymbol{\delta}$ but considering $\bar{n}_{i,j,q}$ instances of task τ_i instead of $\eta_i^+(B_j(q, \boldsymbol{\delta}))$ is given through the following iterative formula, which is directly deduced from the busy-window equation (2).

$$B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q}) = q \cdot C_j + \sum_{l \in \mathcal{I}_j \setminus i} \eta_l^+(B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q})) \cdot C_l + \bar{n}_{i,j,q} \cdot C_i \quad (24)$$

Thus, we need to find the maximum value of $\bar{n}_{i,j,q}$ such that the busy-window constraint is satisfied, i.e. $B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q}) \leq \hat{B}_j(q)$. We formalize this in the following lemma.

Lemma 3 *For a given activation pattern specification $\boldsymbol{\delta}$ let a given system satisfy task τ_j's constraint on worst-case response time D_j.*

$\bar{n}_{i,j,q}$ is the maximum number of instances of task τ_i that may appear in the q-event busy-window of task τ_j instead of $\eta_i^+(B_j(q, \boldsymbol{\delta}))$, such that the worst-case response time constraint D_j is still satisfied for the q-th instance of τ_j. $\bar{n}_{i,j,q}$ is given through

$$\bar{n}_{i,j,q} = \max_{n \in \mathbb{N}^+} \left(n : B_j(q, \boldsymbol{\delta}, n) \leq \hat{B}_j(q) \right) \quad (25)$$

Proof: First we show, that the constraint D_j is still satisfied for the q-th instance of τ_j if at most $\bar{n}_{i,j,q}$ instances of τ_i appear in the busy-window $B_j(q, \boldsymbol{\delta})$. Then, we show that $\bar{n}_{i,j,q}$ is maximal.

By definition $B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q})$ is the q-event busy-window of task τ_j considering $\bar{n}_{i,j,q}$ instances of task τ_i instead of $\eta_i^+(B_j(q, \boldsymbol{\delta}))$. By the relation within the max term of (25) this busy-window has to satisfy the busy-window constraint $\hat{B}_j(q)$ of the q-th instance of τ_j. With lemma 2 we conclude, that the D_j is satisfied for the q-th instance of τ_j.

Maximality of $\bar{n}_{i,j,q}$ follows from (25) and lemma 2. ■

To determine the complexity of the calculation of $\bar{n}_{i,j,q}$, note, that n is in natural numbers. Furthermore, $B_j(q, \boldsymbol{\delta}, n)$ can always be lower bounded without iteration to $q \cdot C_j + n \cdot C_i$ and is upper bounded by the constraint $\hat{B}_j(q)$. Consequently, determination of $\bar{n}_{i,j,q}$ requires a limited number of candidates.

With lemma 3, we can determine the maximum number of instances of the analyzed task, that may appear in any given q-event busy-window of the influenced task. Note, that $\hat{B}_j(q)$ and also $\bar{n}_{i,j,q}$ are independent of η_i^+. Thus, we can reason about adaptation of δ_i under consideration of other tasks' constraints. We express this in the following theorem.

Theorem 2 *For a given activation pattern specification $\boldsymbol{\delta} = \langle \delta_1, \ldots, \delta_n \rangle$ let a given system satisfy task τ_j's constraint on worst-case response time D_j.*

Then, D_j is also satisfied for any $\bar{\boldsymbol{\delta}}_{\boldsymbol{i}} = \langle \delta_1, \ldots, \bar{\delta}_i, \ldots, \delta_n \rangle$ that satisfies the following constraints

$$\forall q \in \mathbb{N}^+ : \bar{\delta}_i(\bar{n}_{i,j,q} + 1) > B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q}) \quad (26)$$

Proof: Applying $\bar{\eta}_i^+$ to both sides of (26) yields

$$\bar{\eta}_i^+(\bar{\delta}_i(\bar{n}_{i,j,q} + 1)) > \bar{\eta}_i^+(B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q})) \quad (27)$$

According to (1) we obtain

$$\bar{n}_{i,j,q} \geq \bar{\eta}_i^+(B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q})) \quad (28)$$

Algorithm 1 Sensitivity analysis for task τ_i

```
Input: $C_l$, $B_l(q, \boldsymbol{\delta})$ for all tasks $\tau_l \in \{\tau_p : \tau_i \in \mathcal{I}_p, \tau_i\}$ and activations
 1: $\forall q : \delta_i(q) = 0$
 2: // self-influence
 3: calculate $\bar{Q}_i - eq.\ 9$
 4: for $q$ do
 5:     if $q \leq \bar{Q}_i$ then
 6:         $\delta_i(q) = \max(\delta_i(q), B_i(q, \boldsymbol{\delta}) - D_i)$
 7:     else
 8:         $\delta_i(q) = \max(\delta_i(q), B_i(\bar{Q}_i, \boldsymbol{\delta}) + 1)$
 9: // cross-influence
10: for $\tau_j : \tau_i \in \mathcal{I}_j$ do
11:     for $q$ do
12:         calculate $\bar{n}_{i,j,q}$ - eq. 25
13:         $\delta_i(\bar{n}_{i,j,q} + 1) = \max(\delta_i(\bar{n}_{i,j,q} + 1), B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q}) + 1)$
14: // make $\delta_i$ super-additive
15: $\forall a, b : \delta(a + b - 1) = \max(\delta(a + b), \delta(a) + \delta(b))$
```

With this and (2) and (24) we conclude that $B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q})$ is an upper bound for $B_j(q, \bar{\boldsymbol{\delta}}_{\boldsymbol{i}})$, i.e.

$$B_j(q, \boldsymbol{\delta}, \bar{n}_{i,j,q}) \geq B_j(q, \bar{\boldsymbol{\delta}}_{\boldsymbol{i}}) \quad (29)$$

With (26) and the definition of sufficiently early (4) the theorem follows. ■

Theorem 2 allows to derive constraints on the minimum distance function δ_i of a task τ_i based on the deadline D_j of another task τ_j. Thus, in combination with theorem 1 we have the means to calculate minimally allowed values for $\delta_i(q)$ under consideration of all constraints. In the following section we embed these constraints into a sensitivity analysis.

C. Overall analysis

In this section we consider the constraints on minimum distance functions from the previous two sections and embed them into a sensitivity analysis scheme.

The overall analysis is given in algorithm 1. As input, the algorithm requires for the task under analysis τ_i and for all influenced tasks $\tau_l \in \{\tau_p : \tau_i \in \mathcal{I}_p\}$, the worst-case execution times and deadlines, and the busy-windows for an initial activation pattern description, under which all constraints are satisfied.

First all $\delta_i(q)$ are initialized to 0 (line 1) as starting point. These values are increased throughout the algorithm according to the constraints from theorems 1 and 2. Furthermore, the maximum number of instances of τ_i that may occur in a busy-window $\bar{Q}_i$ is calculated (line 3). To account for the self-influence for all considered instances, the constraint according to theorem 1 is calculated, depending on, whether the instance may occur within the busy-window (line 6) or may not fall into the busy-window (line 8). To consider the cross-influence, we iterate over all tasks and calculate for all instances, the number of allowed instances of τ_i (line 12) and assign the constraint on $\delta_i(\bar{n}_{i,j,q} + 1)$ according to theorem 2. In the end, we make the function pseudo super-additive again (line 15), as required by def. 1. Note, that although the algorithm iterates over all q, minimum distance functions are practically used only on a limited interval. Iterations over q (lines 4, 11) can be limited to such relevant intervals to bound runtimes.

This completes the approach to arbitrary activation pattern sensitivity under consideration of worst-case response time constraints. We have first derived conditions for $\delta_i(q)$ under which the analyzed task itself does not violate its constraint

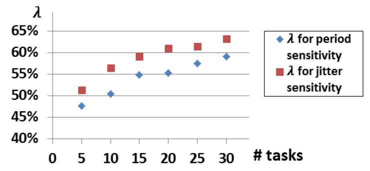


Fig. 6: Quality metric w.r.t. the two reference algorithms

and then considered the influence on other tasks. Finally, we have combined the constraints to an overall sensitivity analysis. In the next section we evaluate the presented analysis with synthetic testcases and two exact parametrized analyses.

V. Evaluation

In order to evaluate the performance of the proposed sensitivity analysis, we compare our algorithm based on a synthetic set of testcases with two exact sensitivity analyses, that operate on parametrized activation patterns rather, than arbitrary activation patterns.

The first reference algorithm performs a binary search over activation jitter, i.e. the maximum allowed activation jitter of the analyzed task is determined. The second reference algorithm performs the same binary search over the activation period of the analyzed task.

The testcases, consist of a set of 5, 10, 15, 20, 25 or 30 tasks on a processor under static priority preemptive scheduling. All tasks are activated periodically with period from a uniform distribution in the interval [100, 5000] and an activation jitter from a uniform distribution in [0, 5*period]. WCETs are generated with UUniFast [18] to achieve a utilization of 0.7. Deadlines are derived from the WCRTs of the initial system specification such that the constraint is chosen from uniform distribution in the interval [1.3*WCRT, 2.5*WCRT]. The sensitivity analysis is performed for the highest priority task to obtain a maximum number of constraints. We have determined the sensitivity bound on δ_i for $q \in [1, 20]$. For each number of tasks we have analyzed 100 different systems.

Results are as anticipated in figure 1. The determined arbitrary activation pattern sensitivity bounds are below the parametrized bounds for all testcases and values of q. Instead of showing such sensitivity bounds for individual testcases, we quantify the improvement over all testcases. For quantification of the improvement we introduce a quality metric, which we base on the relative difference to the reference sensitivity bound, i.e. let $\bar{\delta}_{ref}$ be the activation pattern determined by the reference algorithm and $\bar{\delta}_{arb}$ be the sensitivity bound determined by the proposed algorithm. The relative difference is $\Delta(q) = \frac{\bar{\delta}_{ref}(q) - \bar{\delta}_{arb}(q)}{\bar{\delta}_{ref}(q)}$. Then, the quality metric is the mean relative difference over the interval $[1, q_{max}]$, i.e.

$$\lambda = \text{mean}_{q \in [1, q_{max}]} \Delta(q) \tag{30}$$

Figure 6 shows the mean quality metric for the testcases from the different parameter sets, i.e. the different numbers of tasks. The values were obtained w.r.t. each of the two reference algorithms. We see, that the proposed algorithm outperforms either reference algorithm significantly, although the initial system specification is strictly periodic with jitter. On average, the determined values of the minimum distance functions are $\sim 55\%$ below those, obtained by the reference algorithms, despite the fact that the reference algorithms are exact. This improvement originates solely from the fact, that the proposed algorithm does not rely on a parametrization of the activation pattern, but determines the allowed values of $\delta_i(q)$ for each q individually. Runtimes of the proposed algorithm are tractable. For the testcases the analysis required between 0.03s and 356s on an Intel Core i3@2,4GHz processor with 3GB of RAM.

VI. Conclusion

In this paper we have presented a novel approach to activation pattern sensitivity analysis under consideration of worst-case response time constraints. Instead of searching for a maximally allowed parametrization of an activation pattern, as previous approaches, our approach determines an arbitrary minimum distance function as sensitivity bound. The correctness of these bounds is formally proven and in an extensive set of testcases, we have shown, that the obtained sensitivity bounds are significantly more accurate than previous exact approaches, that work on parametrizations of activation patterns.

References

[1] K. Tindell and J. Clark, “Holistic schedulability analysis for distributed hard real-time systems,” *Microprocessing and Microprogramming*, vol. 40, pp. 117 – 134, 1994.

[2] R. Davis and A. Burns, “Robust priority assignment for fixed priority real-time systems,” in *Real-Time Systems Symp. (RTSS)*, 2007.

[3] L. Thiele, S. Chakraborty, and M. Naedele, “Real-time calculus for scheduling hard real-time systems,” in *Symp. on Circuits and Systems (ISCAS)*, 2000.

[4] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, “System level performance analysis - the symta/s approach,” *Computers and Digital Techniques, IEE Proc. -*, vol. 152, pp. 148–166, 2005.

[5] J. Lehoczky, L. Sha, and Y. Ding, “The rate monotonic scheduling algorithm: exact characterization and average case behavior,” *Proc. of Real Time Systems Symp. (RTSS)*, pp. 166 –171, 1989.

[6] T.-W. Kuo and A. Mok, “Load adjustment in adaptive real-time systems,” in *Real-Time Systems Symp. (RTSS)*, 1991.

[7] E. Bini, M. D. Natale, and G. Buttazzo, “Sensitivity analysis for fixed-priority real-time systems,” *Real-Time Syst.*, vol. 39, pp. 5–30, 2007.

[8] A. Hamann, R. Racu, and R. Ernst, “A formal approach to robustness maximization of complex heterogeneous embedded systems,” in *Int’l. Conf. on Hardware/Software Codesing and System Synthesis (CODES+ISSS)*, 2006.

[9] S. Punnekkat, R. Davis, and A. Burns, “Sensitivity analysis of real-time task sets,” *Lecture Notes In Computer Science*, vol. 1345, pp. 72–82, 1997.

[10] R. Racu, A. Hamann, and R. Ernst, “Sensitivity analysis of complex embedded real-time systems,” *Real-Time Systems*, vol. 39, pp. 31–72, 2008.

[11] F. Zhang, A. Burns, and S. Baruah, “Sensitivity analysis of arbitrary deadline real-time systems with edf scheduling,” *Real-Time Systems*, vol. 47, pp. 224–252, 2011.

[12] M. Neukirchner, T. Michaels, P. Axer, S. Quinton, and R. Ernst, “Monitoring arbitrary activation patterns in real-time systems,” in *Proc. of IEEE Real-Time Systems Symposium (RTSS)*, Dec 2012.

[13] K. Richter, “Compositional scheduling analysis using standard event models,” Ph.D. dissertation, Technical University of Braunschweig, Department of Electrical Engineering and Information Technology, 2004.

[14] J. Y. L. Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*. Springer Verlag, 2001.

[15] J. Lehoczky, “Fixed priority scheduling of periodic task sets with arbitrary deadlines,” in *Real-Time Systems Symposium, 1990. Proceedings., 11th*. IEEE, 1990, pp. 201–209.

[16] S. Schliecker, J. Rox, M. Ivers, and R. Ernst, “Providing accurate event models for the analysis of heterogeneous multiprocessor systems,” in *Proc. 6th Int’l. Conf. on Hardware Software Codesign and System Synthesis (CODES-ISSS)*, 2008.

[17] S. Schliecker, “Performance analysis of multiprocessor real-time systems with shared resources,” Ph.D. dissertation, Technische Universität Braunschweig, Braunschweig, Germany, 2011.

[18] E. Bini and G. Buttazzo, “Measuring the performance of schedulability tests,” *Real-Time Syst.*, vol. 30, pp. 129–154, 2005.