Multi-Dimensional Robustness Optimization in Heterogeneous Distributed Embedded Systems

Arne Hamann, Razvan Racu, Rolf Ernst
Institute of Computer and Communication Network Engineering
Technical University of Braunschweig
{hamann|racu|ernst}@ida.ing.tu-bs.de

Abstract

Embedded system optimization typically considers objectives such as cost, timing, buffer sizes, and power consumption. Robustness criteria, i.e. sensitivity of the system to property variations like execution and transmission delays, input data rates, CPU clock rates, etc., has found less attention despite its practical relevance.

In this paper we present an approach for optimizing multi-dimensional robustness criteria in complex distributed embedded systems. The key novelty of our approach is a scalable stochastic multi-dimensional sensitivity analysis technique approximating the sought-after sensitivity front from two sides, i.e. coming from the space of working and from the space of non-working system property combinations.

We utilize the proposed stochastic sensitivity analysis to derive multi-dimensional robustness metrics, which are capable of bounding the robustness of given system configurations with little computational effort.

The proposed metrics can significantly speed up multi-dimensional robustness optimization by quickly identifying promising system configurations, whose in-depth robustness evaluation can be performed subsequently to the optimization process.

1. Introduction

Design robustness has always been a concern in embedded system design. Robustness is needed to account for estimation errors in early design phases, minor changes of specifications, bug fixes or later extensions or updates of the design to name just a few of the many situations where tolerance of hardware and run time system to modifications is expected. Typically, designers add some slack to the system parameters to reach robustness, e.g. a maximum load limitation for buses. While adding parameter slack used to work reasonably in practice, this heuristic approach is loosing effectiveness. With growing system size and complex networked systems, the effects of modifications on load and timing are more difficult to predict while at the same time, an increasing number of independently developed applications is integrated on the same system possibly leading to unknown coupling effects or limitations. Examples are cars or aircrafts. Results are an increasing design risk and non-extendable systems.

One approach to increase design robustness is parameter adaptation at run-time, such as in adaptive scheduling strategies [9]. Adaptation is very efficient but comes with run-time overhead and makes it more difficult to determine the resulting design robustness. So, adaptation potentially improves the system but does not circumvent the need to determine system robustness.

To systematically determine robustness, we need sensitivity analysis. Sensitivity analysis extends the idea of using a slack metric. Rather than evaluating the slack as the permissible change of a single component parameter until this component reaches 100% utilization, sensitivity analysis takes a global approach and determines the maximum (or minimum) value of a parameter where the whole system is still functional. So, sensitivity analysis takes global effects into account. The sensitivity analyses approaches presented in [10, 11], for instance, can capture the global effects of single system property variations. They are capable of calculating the boundaries representing the transition between working and non-working systems for a large variety of system properties, including worst-case execution times, communication volumes, input data rates, processor and communication link performance, etc.

Once we have a sensitivity analysis available, we can take the next step to include robustness as a regular design goal. For that purpose we need formal robustness metrics that allow to quantify robustness that is then used in system optimization. The approach presented in [7] calculates the robustness to a single parameter change using a one-dimensional sensitivity analysis that employs binary search. The paper proposes robustness metrics for different design scenarios.

However, since the underlying sensitivity analysis is one-dimensional these robustness metrics ignore dependencies between the considered system properties. For instance a small variation of one system property might have drastic effects on the robustness potential of another system property.

There are approaches [5, 1] that consider multi-dimensional robustness optimization problems. However, these approaches are based on very simple application models facilitating the determination of the needed multi-dimensional robustness data. Consequently, these approaches are not applicable to state-of-the-art performance models [8, 3] for complex embedded distributed systems that are able to tightly determine various system performance properties including timing, buffer sizes, and power consumption.

In this paper we present an approach to multi-dimensional robustness optimization, which can be efficiently utilized for robustness optimization of complex embedded distributed systems using state-of-the-art performance models [8, 3].

The key novelty of the presented approach is a stochastic technique approximating the sensitivity front from two sides, i.e. coming from the space of working and coming from the space of non-working system property combinations. Consequently, our ap-

proach can be utilized to define multi-dimensional robustness metrics estimating the lower and the upper robustness bounds of given system configurations.

Additionally, the proposed metrics are parameterizable and allow to trade approximation quality versus computational effort. This represents a huge advantage of the presented approach since even for low approximation qualities, the lower and upper robustness bounds are very good indicators to identify interesting system configurations, whose detailed analysis can be postponed after robustness optimization.

The remainder of this paper is structured as follows. First, we introduce the proposed stochastic multi-dimensional sensitivity analysis (Section 2). We then give a short survey over algorithms for hypervolume calculation, which we utilize to define our robustness metrics (Section 3). Afterwards, we formally define parameterizable multi-dimensional robustness metrics as well as their approximation quality (Section 4), and explain how they can be utilized efficiently for robustness optimization (Section 5). Finally, we introduce a distributed embedded example systems to evaluate the convergence behavior of our robustness metrics and to demonstrate their applicability (Section 6).

2. Stochastic Multi-Dimensional Sensitivity Analysis

In this section we present a stochastic approach for multi-dimensional sensitivity analysis. It is based on the design space exploration framework introduced in [6] using multi-dimensional evolutionary search techniques [2, 14] and state-of-the-art compositional performance verification methods [3, 8].

We first give a short description of how we use the exploration framework to perform multi-dimensional sensitivity analysis (Section 2.1). We then give details on the search space encoding (Section 2.2) and the creation of the initial population used as starting point for exploration (Section 2.3). Afterwards, we discuss the exploration strategy used during variation to bound the search space, and thus improving analysis speed and approximation quality (Section 2.4). Finally, we explain in detail the exploration strategy implemented by the variation operators (Sections 2.5 and 2.6).

2.1 Analysis Idea

Classical applications of exploration frameworks for complex distributed systems assume variation of system parameters like scheduling, mapping, etc. to optimize criteria including timing, power consumption, and buffer sizes.

In this paper we utilize design space exploration in a different way in order to cover multi-dimensional sensitivity analysis. Instead of modifying the system parameter configuration during exploration, we modify system properties subject to sensitivity analysis, i.e. worst-case core execution times, CPU clock rates, input data rates, etc. Thereby, the optimization objectives are, depending on the considered system properties, either the maximization or the minimization of the property values under the restriction that the system must stay functional, i.e. all system constraints, like e.g. end-to-end deadlines, must be satisfied.

For instance, in the case of a three-dimensional $WCET$ sensitivity analysis of three tasks, the search space consists of the $WCET$ assignment for those tasks and the optimization objectives are the simultaneous maximization of the latter.

Note that the utilized exploration framework [6] performs Pareto-optimization, and that the obtained Pareto-front corresponds to the sought-after sensitivity front representing the boundary between feasible and non-feasible system configurations. Furthermore, the algorithms presented in the next sections are applicable to arbitrary system properties, including those which are subject to maximization (e.g. worst-case core execution times) and those which are subject to minimization (e.g. input data rates).

It is important to mention, that the sensitivity front coverage is controlled by problem independent selector algorithms [2]. We currently use SPEA2 [14] as selector, which assures a diversified approximation of the sensitivity front through Pareto-dominance-based selection and density approximation.

2.2 Search Space Encoding

A system property value combination considered during stochastic multi-dimensional sensitivity analysis is encoded as vector containing one real number entry for each considered property. In the following we refer to such a vector as *individual*.

For instance, in the case of a three-dimensional sensitivity analysis for the system properties $\mathcal{P}_1$, $\mathcal{P}_2$ and $\mathcal{P}_3$, an individual A is represented as three dimensional vector, i.e. $A = (a_1, a_2, a_3)$.

2.3 Initial Population

Algorithm 1 describes the creation of the initial population. In the first part (lines 1 to 3) it uses one-dimensional sensitivity analysis as described in [10] to calculate the available slack for each considered system property.

The one-dimensional slack values represent the extreme points of the sought-after sensitivity front, and thus describe the bounding hypercube containing all valid system property assignments. This information is used throughout the whole exploration to considerably limit the generation of invalid individuals.

In the second part of the algorithm (lines 4 to 7) the rest of the initial population is randomly generated. Thereby, the individuals are uniformly distributed within the search space bounded by the hypercube calculated in the first part of the algorithm.

Algorithm 1 Initial Population

INPUT: System properties $\mathcal{P}_1, \ldots, \mathcal{P}_n$, Initial property values $\mathcal{V}^{init}_{\mathcal{P}_1}, \ldots, \mathcal{V}^{init}_{\mathcal{P}_n}$, Initial population size $\alpha > n$
OUTPUT: Initial population $\mathcal{I}$

1: **for all** i such that $1 \leq i \leq n$ **do**
2: $\quad \mathcal{V}^{ext}_{\mathcal{P}_i} = computeSlack(\mathcal{P}_i)$
3: $\quad$ add vector $(\mathcal{V}^{init}_{\mathcal{P}_1}, \ldots, \mathcal{V}^{ext}_{\mathcal{P}_i}, \ldots, \mathcal{V}^{init}_{\mathcal{P}_n})$ to $\mathcal{I}$
4: **while** $(|\mathcal{I}| < \alpha)$ **do**
5: $\quad$ **for all** i such that $1 \leq i \leq n$ **do**
6: $\quad\quad$ Choose random $\mathcal{V}^{rand}_{\mathcal{P}_i} \in \left[\min\left(\mathcal{V}^{init}_{\mathcal{P}_i}, \mathcal{V}^{ext}_{\mathcal{P}_i}\right), \max\left(\mathcal{V}^{init}_{\mathcal{P}_i}, \mathcal{V}^{ext}_{\mathcal{P}_i}\right)\right]$
7: $\quad$ add vector $\left(\mathcal{V}^{rand}_{\mathcal{P}_1}, \ldots, \mathcal{V}^{rand}_{\mathcal{P}_n}\right)$ to $\mathcal{I}$

2.4 Bounding the Search Space

In this section we give a description of the technique used by our approach to efficiently bound the region containing the sought-after sensitivity front during exploration. This information is used by the variation operators presented in the following sections to prevent exploration from generating and evaluating individuals not improving approximation quality.

The algorithm used to bound the search space as described above is based on the notion of Pareto-optimality.

Definition 1 (Pareto-optimal) *Given a set V of n-dimensional vectors in $\mathbb{R}^n$, the vector $v = (v_1, \ldots, v_n) \in V$ dominates the vector $w = (w_1, \ldots, w_n) \in V$ iff for all elements $1 \leq i \leq n$ we have*

1. *minimization problem: $v_i \leq w_i$ and for at least one element l we have $v_l < w_l$.*
2. *maximization problem: $v_i \geq w_i$ and for at least one element l we have $v_l > w_l$.*

A vector is called Pareto-optimal *iff it is not dominated by any other vector in V.*

Given Definition 1 a simple algorithm can be derived checking whether or not a given k-dimensional vector v is Pareto-optimal with respect to a collection of reference vectors V. In the following we refer to such an algorithm as $dominatedBy(v, V, max)$, where $max \in \{true, false\}$ indicates if Pareto-optimality is meant in the sense of a maximization or minimization problem.

In order to achieve an efficient bounding of the search space containing the sensitivity front, our analysis maintains two sets of individuals:

- the set of evaluated Pareto-optimal working individuals $\mathcal{F}^w$ called *bounding working Pareto-front*. Note that for system properties subject to maximization (minimization) Pareto-optimality is meant in the sense of a maximization (minimization) problem.
- the set of evaluated Pareto-optimal non-working individuals $\mathcal{F}^{nw}$ called *bounding non-working Pareto-front*. Note that for system properties subject to maximization (minimization) Pareto-optimality is meant in the sense of a minimization (maximization) problem.

In the following we refer to the region lying between the Pareto-sets $\mathcal{F}^w$ and $\mathcal{F}^{nw}$ as *relevant region*.

It is easy to understand, that the sought-after sensitivity front lies inside the relevant region. Consider for instance the relevant region determined for two system properties subject to maximization visualized in Figure 1.

An individual lying below the bounding working Pareto-front cannot be part of the sensitivity front, since at least one individual of the bounding working Pareto-front has higher, and thus better, values for all considered system properties. Also, an individual lying above the bounding non-working Pareto-front cannot be part of the sensitivity front, since there exist at least one non-working individual on the bounding non-working Pareto-front, which has smaller values for all considered system properties. Consequently, the individual in question is non-working as well.

Algorithm 2 verifies if a given vector v lies in the relevant region. By configuring the input variable *max* the algorithm can be used for system properties subject to minimization ($max = false$) and maximization ($max = true$).

Note that the sets $\mathcal{F}^w$ and $\mathcal{F}^{nw}$ are updated during exploration directly after the evaluation of the offsprings generated during the processing of each generation. More precisely, for each working (non-working) individual i created during variation it is checked whether i is Pareto-optimal with respect to the individuals contained in $\mathcal{F}^w$ ($\mathcal{F}^{nw}$). If this is the case, i is added to $\mathcal{F}^w$ ($\mathcal{F}^{nw}$) and all individuals dominated by i are removed.

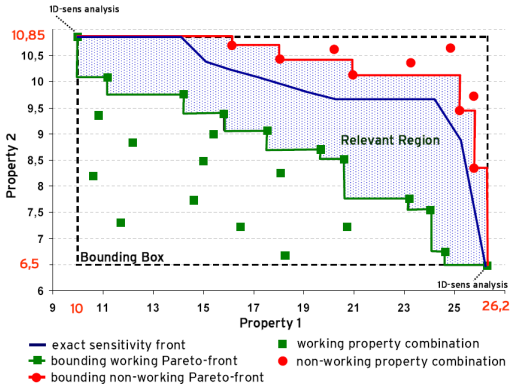


Figure 1. Relevant region for two system properties subject to maximization

Algorithm 2 isInRelevantRegion

INPUT: k-dimensional vector v, bounding working Pareto-front $\mathcal{F}^w$, bounding non-working Pareto-front $\mathcal{F}^{nw}$, type of considered properties $max \in \{true, false\}$

OUTPUT: $true$: if v lies in the relevant region, $false$: otherwise

1: $workingOK = !dominatedBy(v, \mathcal{F}^w, max)$
2: $nonWorkingOK = !dominatedBy(v, \mathcal{F}^{nw}, !max)$
3: **return** $workingOK \;\&\&\; nonWorkingOK$

It is not strictly necessary to remove dominated individuals from the corresponding sets for our approach to work. However, by doing so the maintained sets are kept small and the overall computational effort for checking whether or not a given individual is contained in the relevant region is reduced.

The variation operators guiding the exploration process, which are presented in the following sections, utilize Algorithm 2 to highly increase the generation of offsprings directly improving the approximation of the sought-after sensitivity front. This leads to decreased exploration time for the same quality of approximation.

2.5 Crossover Operator

The crossover operator described in Algorithm 3 implements a heuristic strategy to converge towards the sensitivity front, i.e. the boundary between working and non-working systems. Its main function during exploration is to locally refine the approximation of the sensitivity front. It takes as input two parent individuals P_1 and P_2 and generates two offsprings O_1 and O_2 by using the generalized mean function (Definition 2), which maps well to the structure of the solution space.

Definition 2 (Generalized Mean) *For the positive numbers $x_1, \ldots, x_n$ the k-th mean is defined as follows:*

$$M_k(x_1, \ldots, x_n) = \sqrt[k]{\frac{1}{n} \sum_{i=1}^{n} x_i^k}$$

Special cases: $k \to -\infty : \min(x_1, \ldots, x_n)$; $k = -1$: harmonic mean; $k \to 0$: geometric mean; $k = 1$: arithmetic mean; $k = 2$: quadratic mean; $k \to \infty : \max(x_1, \ldots, x_n)$.

For crossover the generalized mean is applied property wise on the property vectors of the parent individuals (line 7). Figure 2 visualizes the behavior of the generalized mean function for the 2-dimensional case. A and B represent two points we want to "crossover". If $k = 1$ is chosen we obtain the arithmetic mean between A and B. This corresponds to a linear characteristic of the sensitivity front, which we observe for instance in the case of load dependent system properties. For the case that the crossover operator chooses $k < 1$ a convex characteristic of the sensitivity front is approximated, whereas $k > 1$ leads to the approximation of a concave characteristic.

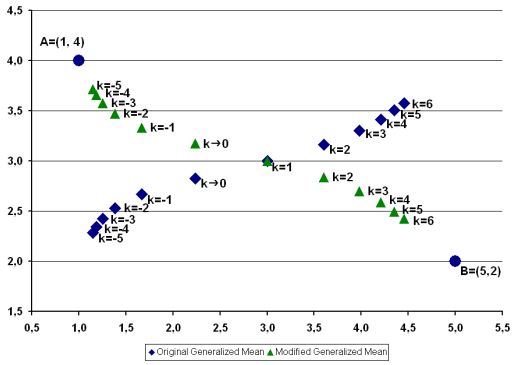


Figure 2. Coordinate-wise generalized mean between A and B for different k

For a better adaptation of the crossover operator to the solution space structure, we modify the values calculated according to the generalized mean formula slightly (lines 8-10). The aim of this modification is to obtain a n-dimensional curve connecting the considered crossover points. Such a connecting curve can be achieved, for instance, by mirroring the calculated general mean values at the bisector defined by the according parent property values in case that the first parent has a higher property value than the second parent. Figure 2 shows the curve we obtain by applying this modification.

The crossover operator utilizes Algorithm 2 presented in Section 2.4 to generate offsprings lying in the relevant region (line 11). However, in some cases the algorithm might fail to find such offsprings. Therefore, the maximum number of attempts is bounded by the constant $attempts_{max}$. One reason for the algorithm not being able to find offsprings lying in the relevant region might be that one of the selected parents for crossover dominates the other one. However, such cases rarely occur during exploration, and are therefore not distinguished in the crossover algorithm.

Note that the crossover operator automatically ensures, that all generated offsprings lie within the bounding hypercube containing all valid system property combinations, which is determined during the creation of the initial population (Section 2.3).

The crossover operator can be used for system properties subject to minimization ($max = false$) and maximization ($max = true$) by configuring the input variable *max*.

2.6 Mutation Operator

The described crossover operator leads to the local convergence of the obtained property values towards the sought-after sensitivity front. In other words, it approximates the sensitivity front "between" individuals considered by the evolutionary algorithm.

Algorithm 3 Crossover operator

INPUT: parents $P_1 = (p_{11}, \ldots, p_{1n})$ and $P_2 = (p_{21}, \ldots, p_{2n})$, k_{min} and k_{max} with $k_{max} \geq k_{min}$, type of considered properties $max \in \{true, false\}$, bounding working Pareto-front $\mathcal{F}^w$, bounding non-working Pareto-front $\mathcal{F}^{nw}$, maximum number of attempts to reach relevant region $attempts_{max}$
OUTPUT: offsprings $O_1 = (o_{11}, \ldots, o_{1n})$ and $O_2 = (o_{21}, \ldots, o_{2n})$

1: **for all** i such that $1 \leq i \leq 2$ **do**
2: Choose random $k \in [k_{min}, k_{max}]$
3: $attempts = 0$
4: **repeat**
5: $attempts = attempts + 1$
6: **for all** j such that $1 \leq j \leq n$ **do**
7: $o_{ij} = M_k\,(p_{1j}, p_{2j})$
8: **if** $(p_{1j} > p_{2j})$ **then**
9: $temp = \min\,(p_{1j}, p_{2j}) + \frac{|p_{1j} - p_{2j}|}{2}$
10: $o_{ij} = temp - (o_{ij} - temp)$
11: **until** $(isInRelevantRegion(O_i, \mathcal{F}^w, \mathcal{F}^{nw}, max)\ ||$ $attempts > attempts_{max})$

Of course, it is possible that the variety of the initial population is insufficient to cover the whole sensitivity front by only using the crossover operator. Additionally, the exploration may get stuck in sub-regions of the front, without the possibility to reach other parts. Therefore, we introduce a mutation operator, enabling the evolutionary search to break out these sub-regions and to cover unexplored parts of the sensitivity front.

The mutation operator is described in Algorithm 4. It takes as input one parent individual from which it creates one offspring by randomly increasing or decreasing each property value by random percentages bounded by $percentage_{max}$. Additionally, the mutation operator takes as input the minimum and the maximum allowed values for each considered system property which are respected during mutation. Note that these values correspond to $\mathcal{V}^{init}_{\mathcal{P}_i}$ and $\mathcal{V}^{ext}_{\mathcal{P}_i}$ calculated during the creation of the initial population (Section 2.3).

Like the crossover operator, the mutation operator utilizes Algorithm 2 described in Section 2.4 to generate offsprings lying in the relevant region (line 11). However, the mutation operator might fail finding such offsprings. The reason for this might be an insufficient Euclidean range, given by $percent_{max}$, for individuals situated far from the relevant region. Therefore, the maximum number of attempts is bounded by the constant $attempts_{max}$.

Note that by configuring the input variable *max* the mutation operator can be used for system properties subject to minimization ($max = false$) and maximization ($max = true$).

3. Hypervolume Calculation

The stochastic multi-dimensional sensitivity analysis approach presented in Section 2 will be utilized in Section 4 to derive expressive robustness metrics for multiple dependent system properties. Thereby, the key algorithm needed to extract robustness properties from the results of the algorithms presented in Section 2 is *hypervolume calculation*.

Hypervolume is usually used as a measure to compare efficiency and to ensure diversity in evolutionary multi-objective algorithms. Especially for the second point it is required that the

Algorithm 4 Mutation operator

INPUT: parent $A = (a_1, \ldots, a_n)$, minimum property values $a_1^{min}, \ldots, a_n^{min}$, maximum property values $a_1^{max}, \ldots, a_n^{max}$, maximum property variation percentage $percent_{max} < 1$, type of considered properties $max \in \{true, false\}$ bounding working Pareto-front $\mathcal{F}^w$, bounding non-working Pareto-front $\mathcal{F}^{nw}$, maximum number of attempts to reach relevant region $attempts_{max}$
OUTPUT: offspring $B = (b_1, \ldots, b_n)$

1: $attempts = 0$
2: **repeat**
3: $attempts = attempts + 1$
4: **for all** i such that $1 \leq i \leq n$ **do**
5: Choose random percentage $p \in]0, percent_{max}]$
6: Choose random boolean $bool$
7: **if** $(bool)$ **then**
8: $b_i = \min\left(a_i \times (1+p), a_i^{max}\right)$
9: **else**
10: $b_i = \max\left(a_i \times (1-p), a_i^{min}\right)$
11: **until** $(isInRelevantRegion(B, \mathcal{F}^w, \mathcal{F}^{nw}, max)$ $||$ $attempts > attempts_{max})$

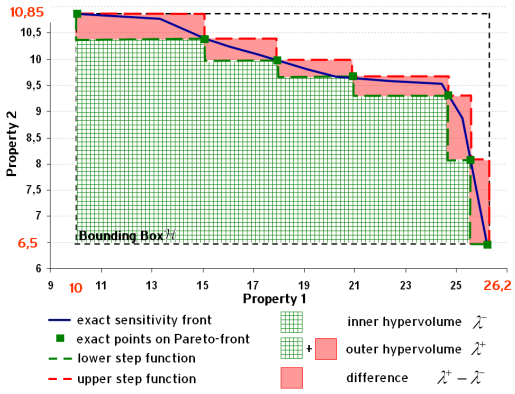


Figure 3. Inner and outer hypervolume

hypervolume can be calculated efficiently. Therefore, several efficient algorithms for hypervolume calculation were proposed in the last years [4, 12, 13].

In this paper we distinguish two different notions of hypervolume: the *inner hypervolume* λ^- and the *outer hypervolume* λ^+.

$\lambda^-(\mathcal{V})$ corresponds to the dominated space enclosed by a set of given Pareto-optimal vectors $\mathcal{V}$ within the bounding hypercube $\mathcal{H}$. Note that in the two-dimensional case $\lambda^-(\mathcal{V})$ corresponds to the area covered by the lower step function connecting the points in $\mathcal{V}$. In this paper, we utilize the algorithm given in [13] to calculate $\lambda^-(\mathcal{V})$.

$\lambda^+(\mathcal{V})$ is defined as the difference between the volume of the bounding hypercube $\mathcal{H}$ and the space of vectors in $\mathcal{H}$ dominating at least one vector in $\mathcal{V}$. In the two-dimensional case $\lambda^+(\mathcal{V})$ corresponds to the area covered by the upper step function connecting the points in $\mathcal{V}$.

Given the algorithm for calculating the *inner hypervolume* $\lambda^-(\mathcal{V})$, the *outer hypervolume* $\lambda^+(\mathcal{V})$ can be calculated according to Algorithm 5.

First, the hypercube bounding $\mathcal{V}$ is calculated (lines $1-6$). Afterwards, the origin of the vectors in $\mathcal{V}$ is translated to the extreme

Algorithm 5 $\lambda^+(\mathcal{V})$

INPUT: Set of n dimensional Pareto-optimal vectors $\mathcal{V}$
OUTPUT: Outer hypervolume λ^+ of $\mathcal{V}$

1: **for all** i such that $1 \leq i \leq n$ **do**
2: $min[i] = \infty$
3: $max[i] = -\infty$
4: **for all** $v \in \mathcal{V}$ **do**
5: $min[i] = \min\left(min[i], v[i]\right)$
6: $max[i] = \max\left(max[i], v[i]\right)$
7: **for all** $v \in \mathcal{V}$ **do**
8: **for all** i such that $1 \leq i \leq n$ **do**
9: $v[i] = max[i] - v[i]$
10: $\lambda^+_{tmp} = 1$
11: **for all** i such that $1 \leq i \leq n$ **do**
12: $\lambda^+_{tmp} = \lambda^+_{tmp} \times (max[i] - min[i])$
13: $\lambda^+(\mathcal{V}) = \lambda^+_{tmp} - \lambda^-(\mathcal{V})$

point of the bounding hypercube (lines $7-9$). Note that the inner hypervolume of the translated vector set corresponds to the space containing all vectors dominating at least one vector of the initial set $\mathcal{V}$. Finally, λ^+ is calculated by substracting the inner hypervolume of the translated vector set from the hypervolume of the bounding hypercube (lines $10-13$).

Figure 3 visualizes an example Pareto-front as well as the inner and outer hypervolumes of several points situated on the Pareto-front.

4. Multi-dimensional Robustness Metrics

In this section we utilize the bounding working Pareto-front $\mathcal{F}^w$ and the bounding non-working Pareto-front $\mathcal{F}^{nw}$ determined during the stochastic multi-dimensional sensitivity analysis (Section 2) to define the robustness of a given system configuration and the approximation quality.

We introduce two different notions of robustness: the *minimum guaranteed robustness* (Definition 3) and the *maximum possible robustness* (Definition 4).

Definition 3 (Minimum Guaranteed Robustness $\mathcal{R}^-$) *We consider a system $\mathcal{S}$ with parameter configuration c and a set of system properties $\mathcal{P} = \{p_1, \ldots, p_n\}$. Given the bounding working Pareto-front $\mathcal{F}^w_{\mathcal{P},c}$ for the parameters in $\mathcal{P}$ the minimum guaranteed robustness is defined as follows:*

$$\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c}) = \lambda^-(\mathcal{F}^w_{\mathcal{P},c})$$

Definition 4 (Maximum Possible Robustness $\mathcal{R}^+$) *We consider a system $\mathcal{S}$ with parameter configuration c and a set of system properties $\mathcal{P} = \{p_1, \ldots, p_n\}$. Given the bounding non-working Pareto-front $\mathcal{F}^{nw}_{\mathcal{P},c}$ for the parameters in $\mathcal{P}$ the maximum possible robustness is defined as follows:*

$$\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c}) = \lambda^+(\mathcal{F}^{nw}_{\mathcal{P},c})$$

$\mathcal{R}^-$ and $\mathcal{R}^+$ are taylored for the underlying stochastic approach to multi-dimensional sensitivity analyis approximating the sensitivity front from two sides, i.e. coming from the space of working and from the space of non-working system property combinations. As we will see in Section 5 this property of our approach

enables its efficient integration into design space exploration for multi-dimensional robustness optimization.

Note that $\mathcal{R}^-$ and $\mathcal{R}^+$ depend on the bounding fronts $\mathcal{F}^w_{\mathcal{P},c}$ and $\mathcal{F}^{nw}_{\mathcal{P},c}$ determined by the underlying stochastic multi-dimensional sensitivity analysis, and thus do not represent the exact robustness of a given system configuration. Consequently, different evaluations of $\mathcal{R}^-$ and $\mathcal{R}^+$ might yield different results. However, it is guaranteed that the real robustness $\mathcal{R}^{real}$ lies somewhere in the interval defined by $\mathcal{R}^-$ and $\mathcal{R}^+$:

$$\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c}) < \mathcal{R}^{real} < \mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c})$$

It is important to mention that $\mathcal{R}^-$ and $\mathcal{R}^+$ are always conservative and do not converge to $\mathcal{R}^{real}$ even if we invest infinite computational effort in the underlying stochastic multi-dimensional sensitivity analysis. This is due to the nature of λ^- and λ^+.

Figure 3 visualizes λ^- and λ^+ for several points lying on the sensitivity front. We observe that even if the bounding Pareto-fronts $\mathcal{F}^w$ and $\mathcal{F}^{nw}$ completely converge during multi-dimensional sensitivity analysis, and thus lie on the sought-after sensitivity front, the robustness metrics $\mathcal{R}^-$ and $\mathcal{R}^+$ are still imprecise.

Accordingly, we define the approximation imprecision of the two defined robustness metrics, which is due to the utilized hypervolume functions λ^- and λ^+.

Definition 5 (Approximation Imprecisions $\mathcal{I}_{\mathcal{R}^-}$ and $\mathcal{I}_{\mathcal{R}^+}$) *We consider a system $\mathcal{S}$ with parameter configuration c and a set of system properties $\mathcal{P} = \{p_1, \ldots, p_n\}$. Given the bounding working Pareto-front $\mathcal{F}^w_{\mathcal{P},c}$ and the bounding non-working Pareto-front $\mathcal{F}^{nw}_{\mathcal{P},c}$ for the parameters in $\mathcal{P}$ the approximation imprecision of $\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})$ and $\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c})$ are defined as follows:*

$$\mathcal{I}_{\mathcal{R}^-}(\mathcal{F}^w_{\mathcal{P},c}) = \frac{\lambda^+(\mathcal{F}^w_{\mathcal{P},c}) - \lambda^-(\mathcal{F}^w_{\mathcal{P},c})}{\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})}$$

$$\mathcal{I}_{\mathcal{R}^+}(\mathcal{F}^{nw}_{\mathcal{P},c}) = \frac{\lambda^+(\mathcal{F}^{nw}_{\mathcal{P},c}) - \lambda^-(\mathcal{F}^{nw}_{\mathcal{P},c})}{\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c})}$$

It is intuitive that the approximation imprecisions $\mathcal{I}_{\mathcal{R}^-}$ and $\mathcal{I}_{\mathcal{R}^+}$ decrease during stochastic multi-dimensional sensitivity analysis as the number of points on the bounding Pareto-fronts increases.

However, $\mathcal{I}_{\mathcal{R}^-}$ and $\mathcal{I}_{\mathcal{R}^+}$ are not related to the difference between the miniumum guaranteed robustness $\mathcal{R}^-$ and the maximum possible robustness $\mathcal{R}^+$. Consequently, they are rather an indicator for the granularity and distribution of points on the bounding Pareto-fronts than metrics for the approximation quality.

Based on the previous definitions, we therefore define the approximation quality of the minimum guaranteed robustness $\mathcal{R}^-$ and the maximum possible robustness $\mathcal{R}^+$ as follows:

Definition 6 (Approximation Quality $\mathcal{Q}$) *We consider a system $\mathcal{S}$ with parameter configuration c and a set of system properties $\mathcal{P} = \{p_1, \ldots, p_n\}$. Given the bounding working Pareto-front $\mathcal{F}^w_{\mathcal{P},c}$ and the bounding non-working Pareto-front $\mathcal{F}^{nw}_{\mathcal{P},c}$ for the parameters in $\mathcal{P}$ the approximation quality of the robustness metrics $\mathcal{R}^-$ and $\mathcal{R}^+$ is defined as follows:*

$$\mathcal{Q}(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c}) = \frac{1}{2}\left(\mathcal{Q}^-(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c}) + \mathcal{Q}^+(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c})\right)$$

where

$$\mathcal{Q}^-(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c}) = \frac{\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c}) - \mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})}{\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})} - \mathcal{I}_{\mathcal{R}^-}(\mathcal{F}^w_{\mathcal{P},c})$$

$$\mathcal{Q}^+(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c}) = \frac{\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c}) - \mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})}{\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c})} - \mathcal{I}_{\mathcal{R}^+}(\mathcal{F}^{nw}_{\mathcal{P},c})$$

Consequently,

$$\mathcal{Q}(\mathcal{F}^w_{\mathcal{P},c}, \mathcal{F}^{nw}_{\mathcal{P},c}) = \frac{1}{2}\left(\frac{\lambda^+(\mathcal{F}^{nw}_{\mathcal{P},c}) - \lambda^+(\mathcal{F}^w_{\mathcal{P},c})}{\lambda^-(\mathcal{F}^w_{\mathcal{P},c})} + \frac{\lambda^-(\mathcal{F}^{nw}_{\mathcal{P},c}) - \lambda^-(\mathcal{F}^w_{\mathcal{P},c})}{\lambda^+(\mathcal{F}^{nw}_{\mathcal{P},c})}\right)$$

Basically, $\mathcal{Q}^-$ and $\mathcal{Q}^+$ are defined as the percentual share of the difference between the maximum possible robustness $\mathcal{R}^+$ and the minimum guaranteed robustness $\mathcal{R}^-$ to $\mathcal{R}^-$ and $\mathcal{R}^+$, respectively.

Since the bounding Pareto-fronts are refined during multi-dimensional sensitivity analysis the difference between $\mathcal{R}^+$ and $\mathcal{R}^-$ converges towards zero. However, due to the approximation imprecision of the robustness metrics, zero is never reached even for completely converged bounding Pareto-fronts. As explained above, this is due to the utilized hypervolume functions λ^- and λ^+.

In order to obtain a quality metric, which is independent of effects due to the characteristics of the underlying hypervolume functions, we subtract the approximation imprecisions $\mathcal{I}_{\mathcal{R}^-}$ and $\mathcal{I}_{\mathcal{R}^+}$ from $\mathcal{Q}^-$ and $\mathcal{Q}^+$, respectively. By this means the resulting quality metric $\mathcal{Q}$ reflects very well the real approximation quality of the presented approach.

Note that lower values for $\mathcal{Q}$ correspond to higher approximation quality.

In Section 6.2 the approximation quality metric $\mathcal{Q}$ will be taken as criterion for the convergence of the robustness metrics in the two- and three-dimensional cases.

5. Using the Metrics for Robustness Optimization

The robustness metrics proposed in Section 4 can easily be integrated into design space exploration to realize a multi-dimensional robustness optimization approach.

During multi-dimensional robustness optimization of a system properties set $\mathcal{P}$, stochastic multi-dimensional sensitivity analysis as described in Section 2 is performed for each considered system configuration c. This analysis delivers the bounding working Pareto-front $\mathcal{F}^w_{\mathcal{P},c}$ and the bounding non-working Pareto-front $\mathcal{F}^{nw}_{\mathcal{P},c}$, which in turn are used to derive the minimum guaranteed robustness $\mathcal{R}^-(\mathcal{F}^w_{\mathcal{P},c})$ and the maximum possible robustness $\mathcal{R}^+(\mathcal{F}^{nw}_{\mathcal{P},c})$.

In Section 4 we already discussed that $\mathcal{R}^-$ and $\mathcal{R}^+$ depend on the stochastically determined bounding Pareto-fronts, and are thus not deterministic. Also, the approximation quality of the robustness metrics depends on the computational effort spent for the underlying stochastic multi-dimensional sensitivity analysis.

However, independent of the reached approximation quality, it is always guaranteed that the real robustness of the analyzed system configuration is contained in the interval defined by $\mathcal{R}^-$ and $\mathcal{R}^+$.

Consequently, the precision of the approach can be safely scaled. This represents a huge advantage of the presented approach since even for low approximation qualities, the minimum guaranteed

and the maximum possible robustness metrics are very good indicators for identifying interesting system worth to be analyzed in detail.

We therefore propose to perform a two-dimensional Pareto-optimization of the minimum guaranteed robustness $\mathcal{R}^-$ and the maximum possible robustness $\mathcal{R}^+$. In so doing, the multi-dimensional robustness optimization yields on the one hand system configurations with large guaranteed robustness and on the other hand system configurations with possibly large robustness potential, which needs to be confirmed or disapproved by further analysis.

It is obvious that using this approach it is sufficient to invest comparatively little computational effort in the robustness evaluation during exploration to identify interesting system configurations, and to postpone their detailed robustness analysis after optimization.

6. Experiments

In this section we first introduce a complex distributed example system (Section 6.1). In Section 6.2 we then perform several experiments to determine the convergence behavior of the robustness metrics presented in Section 4. Afterwards, we use these metrics to evaluate the robustness of the initial configuration of the given distributed example system (Section 6.3). Finally, we optimize its robustness by integrating the robustness metrics into design space exploration (Section 6.4).

6.1 Example System

We consider the example setup shown in Figure 4. It contains four different computational units connected by a bus. Three applications are mapped on the architecture. A video application (solid chain) gathers data from a camera controlled by the micro controller uC, performs preprocessing on the DSP and post-processing on the PPC core. The second application (dashed chain) reads data from a sensor, which is first processed on the ARM core and then forwarded to the PPC core, which, in turn, controls an actor. The third application (dotted line) is a streaming application that runs on the ARM processor core and uses the DSP for data processing.

All three applications have constrained end-to-end latencies which need to be satisfied for the system to function correctly (Table 1(c)).

The computational resources (PPC, uC, DSP, and ARM) are all scheduled according to the static priority preemptive policy, and the interconnecting bus is arbitrated by the CAN protocol. Core execution and core communication times as well as priorities of all tasks and exchanged messages are given in Tables 1(a) and 1(b), respectively. The periods of incoming data at the system inputs (Cam, $Sens$, and S_{in}) are specified in Table 1(d).

6.2 Convergence of the Robustness Metrics

In this section we discuss the convergence behavior of the proposed robustness metrics and their approximation quality. More precisely, we investigate for how long we need to run the underlying stochastic multi-dimensional sensitivity analysis presented in Section 2 to get sufficiently accurate results for the robustness properties of the system configurations considered during robustness optimization.

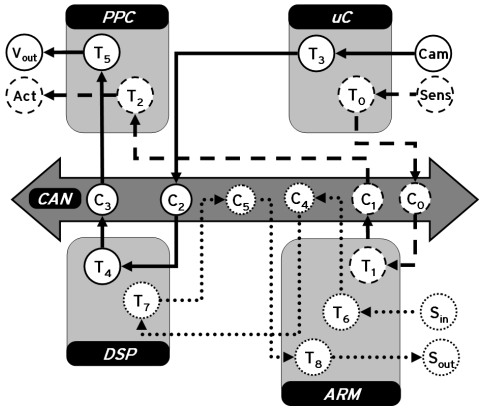


Figure 4. Example system

Channel	*CCT*	*Priority*
C0	[10.4, 12.4]	5
C1	[12, 14.4]	3
C2	[20, 26.4]	2
C3	[15.2, 22.4]	6
C4	[10.4, 14.4]	1
C5	[15.2, 26.4]	4

(a) Core Communication Times

Task	*CET*	*Priority*
T5	[30.9, 43.1]	2
T2	[15.8, 27.4]	1
T3	[36.9, 46.3]	2
T0	[20.1, 48.5]	1
T4	[13, 86]	2
T7	[40.3, 44.5]	1
T1	[27.1, 154.9]	2
T6	[14.2, 63.6]	1
T8	[11.5, 291.2]	3

(b) Core Execution Times

Path	*Deadline*
$Sens \rightarrow Act$	850
$Cam \rightarrow V_{out}$	1100
$S_{in} \rightarrow S_{out}$	1000

(c) End-to-End Constraints

System Input	*Period* $\mathcal{P}$
$Sens$	500
Cam	100
S_{in}	1000

(d) Input Event Models

Table 1. System Parameters

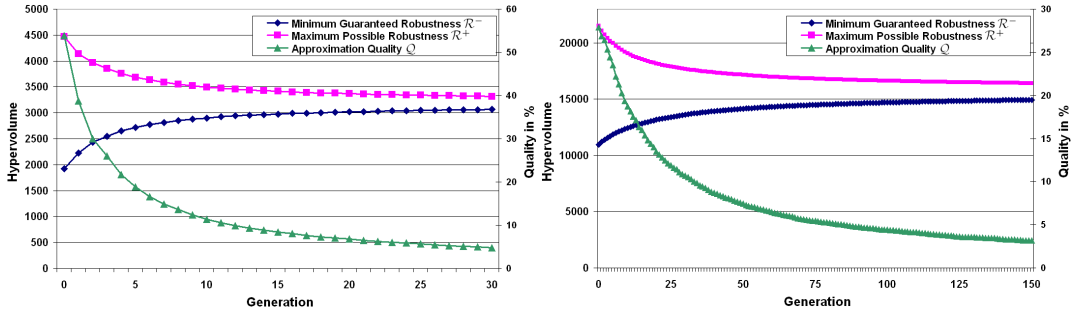
A first impression of the increasing precision of the underlying stochastic multi-dimensional sensitivity analysis is given in Figure 6.

We observe that the approximation quality of the bounding Pareto-fronts is rather rough after 10 generations (Figure 6(a)). The approximation quality is greatly improved after 20 generation (Figure 6(b)), and after 30 generations the bounding working and non-working Pareto-fronts have nearly converged and approximate the sought-after sensitivity front very precisely (Figure 6(c)).

Note that the run-time of the above performed stochastic two-dimensional sensitivity analyses considering 10, 20, and 30 generations are 20, 40, and 60 seconds on a standard P4 at $2.4Ghz$, respectively.

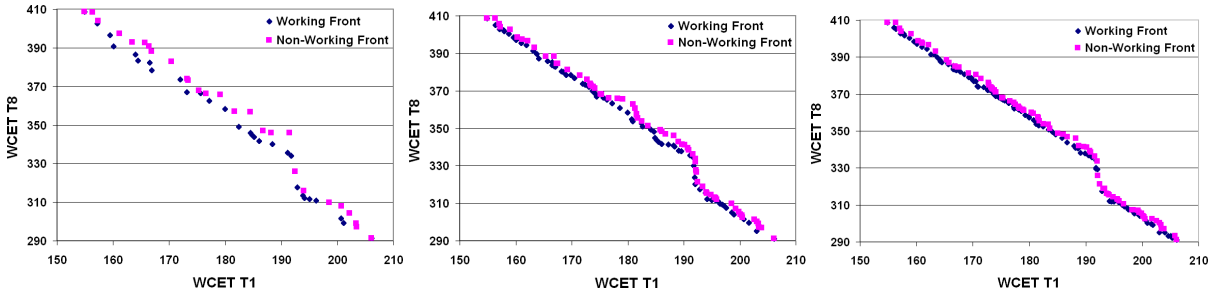
Figure 5(a) and 5(b) visualize the average values for the minimum guaranteed robustness $\mathcal{R}^-$, the maximum possible robustness $\mathcal{R}^+$, and the approximation quality $\mathcal{Q}$ obtained by 100 two- and three-dimensional stochastic sensitivity analyses, respectively. Thereby, the stochastic sensitivity analyses considering 30 and 150 generations of each 10 individual in the two- and three-dimensional case, respectively.

We observe that $\mathcal{R}^-$ and $\mathcal{R}^+$ converge with increasing generation count. Thereby, the convergence behavior is of logarithmic characteristic. We also observe that the approximation quality in-



(a) Two-dimensional case: WCETs T1 and T8 (b) Three-dimensional case: WCETs C0, C3, and T1

Figure 5. Evolution of robustness metrics and approximation quality



(a) Result after 10 generations (100 Individuals) (b) Result after 20 generations (200 Individuals) (c) Result after 30 generations (300 Individuals)

Figure 6. Increasing approximation quality in the two-dimensional case: WCETs T1 and T8

creases with increasing generation count.

In the two-dimensional case (Figure 5(a)) the stochastic sensitivity analysis achieves an average approximation quality of 11.4% after 10 generations. As we can observe in Figure 6(a) this corresponds to a rather rough approximation of the sought-after sensitivity front. After 20 generations the approximation quality increases to 6.8%, which corresponds to a quite precise approximation of the sensitivity front as can be seen in Figure 6(b). Finally, after 30 generations, the stochastic two dimensional sensitivity analysis reaches an approximation quality of 4.8%, which corresponds to a very precise approximation of the sensitivity front as visualized in Figure 6(c).

In the three-dimensional case the analysis time needed to reach comparable approximation qualities as in the two-dimensional case is approximately 3 times higher as can be observed in Figure 5(b).

The above results indicate that in the two-dimensional case each system configuration should be evaluated by the underlying stochastic sensitivity analysis using between 10 and 30 generation of 10 individuals each. In the three-dimensional case the effort spent for the stochastic sensitivity analysis should be 3 times higher, and thus between 30 and 90 generations of 10 individuals.

Consequently, the evaluation time for a single system configuration during robustness optimization lies between 20 and 60 seconds in the two-dimensional case, and between 60 and 180 seconds in the three-dimensional case.

6.3 Robustness of the Initial Configuration

Scheduling analysis of the example system presented in Section 6.1 with performance verification techniques [3, 8] reveals that all end-to-end constraints are satisfied: $Sens \rightarrow Act$ 551.2, $S_{in} \rightarrow S_{out}$ 882.6, and $Cam \rightarrow V_{out}$852.2.

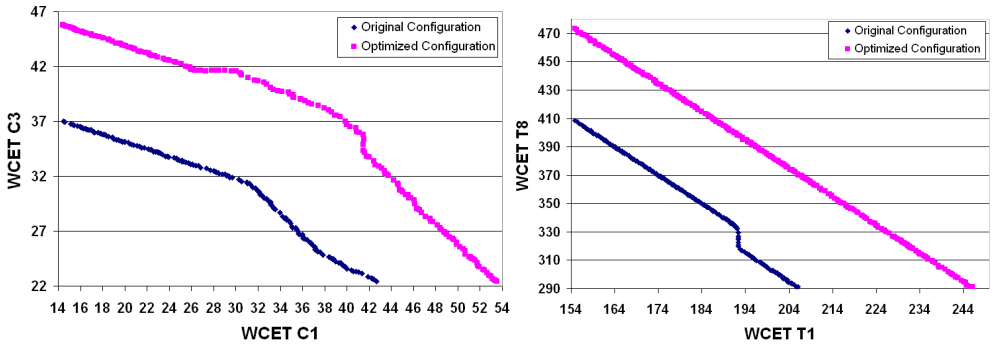
In the following we are interested in the two-dimensional robustness properties of the communication channels $C1$ and $C3$ as well as the tasks $T1$ and $T8$ mapped on the ARM processor. Additionally, we are interested in three-dimensional robustness properties of the three tasks mapped on the ARM processor $T1$, $T6$, and $T8$ as well as of the task $T1$ in conjunction with the communication channels $C0$ and $C3$.

Figures 7(a) and 7(b) visualize the sensitivity fronts for the two-dimensional cases. The sensitivity fronts were calculated using 50 generations of 10 individuals each, which took approximately 100 seconds on a standard P4 at $2.4Ghz$. The determined sensitivity fronts correspond to a robustness of 241 and 3141 for $C1$-$C3$ and $T1$-$T8$, respectively.

The sensitivity fronts for the three-dimensional cases are visualized in Figures 8(a) and 8(c). The sensitivity fronts were calculated using 150 generations of 10 individuals each, which took approximately 300 seconds on a standard P4 at $2.4Ghz$. The determined sensitivity fronts correspond to a robustness of 14942 and 64289 for $T1 - C3 - C0$ and $T1 - T6 - T8$, respectively.

6.4 Optimizing Robustness

In order to optimize the robustness of the given system we include the robustness metrics presented in Section 4 into design space exploration as described in Section 5. Thereby, the search space of the robustness optimization consists of the priority assignments on all processors and on the interconnecting bus.



(a) WCETs C1 and C3 (b) WCETs T1 and T8

Figure 7. Two-dimensional WCET sensitivity fronts for original and optimized system configurations

In all experiments we run the outer exploration optimization loop, i.e. the loop modifying the priority assignments, for 10 generations consisting of 25 individuals. The underlying multi-dimensional sensitivity analysis used to calculate the robustness metrics considers 15 generation in the two-dimensional and 50 generations in the three-dimensional cases. Thereby, 10 individuals are evaluated in each generation. Note that with this setup a complete robustness optimization takes approximately 900 and 3000 seconds in the two- and three-dimensional cases, respectively.

The two-dimensional robustness optimization of the communication channels $C1$ and $C3$ yielded the following optimal priority assignment: $C4 > C5 > C2 > C0 > C1 > C3$ (CAN), $T7 > T4$ (DSP), $T3 > T0$ (uC), $T2 > T5$ (PPC), and $T6 > T1 > T8$ (ARM). The evaluated minimum guaranteed and maximum possible robustness of this configuration are 551.77 and 625.4, respectively. A more detailed analysis determined the real robustness of this system configuration to be 583.2. Compared to the original system configuration this corresponds to a robustness increase of more than 140%. The corresponding sensitivity front is visualized in Figure 7(a).

The priority assignment with optimal two-dimensional robustness properties for the tasks $T1$ and $T8$ determined during robustness optimization is the following: $C4 > C5 > C0 > C2 > C1 > C3$ (CAN), $T7 > T4$ (DSP), $T0 > T3$ (uC), $T2 > T5$ (PPC), and $T6 > T1 > T8$ (ARM). The minimum guaranteed and the maximum possible robustness calculated during optimization for this configuration are 7854.34 and 9032.3, respectively. A detailed analysis revealed that the robustness of the optimized system configuration is equal to 8236.2, which corresponds to an robustness increase of more than 160% compared to the original configuration. The corresponding sensitivity front is visualized in Figure 7(b).

The three-dimensional robustness optimization of $T1$, $C3$, and $C0$ yielded the same optimal priority assignment as the two-dimensional case considering the communication channels $C1$ and $C3$. Robustness optimization determined a minimum guaranteed robustness of 31520.31 and maximum possible robustness of 44371.32 for this configuration. The real robustness of the system configuration is equal to 37006.52, which corresponds to a robustness increase of more than 145% compared to the original system configuration. The corresponding sensitivity front is visualized in Figure 8(b).

The optimal priority assignment for the three-dimensional robustness of the tasks $T1$, $T6$, and $T8$ is the same as in the two-dimensional case considering only $T1$ and $T8$. Thereby, the determined values for the minimum guaranteed and the maximum possible robustness are 243392.81 and 418181.17, respectively. The robustness subsequently determined by detailed robustness evaluation is equal to 294862.49, representing a robustness increase of more than 350% compared to the original system configuration. The corresponding sensitivity front is visualized in Figure 8(d).

7. Conclusion

In this paper we presented an efficient approach to multi-dimensional robustness optimization in complex distributed embedded systems. Our approach is based on a scaleable stochastic multi-dimensional sensitivity analysis, which we utilized to derive upper and lower robustness bounds for given system configurations with reasonable computational effort.

We have shown by means of extensive experiments that the application of the presented techniques to robustness optimization allows the detection of promising system configurations without performing computationally expensive in-depth robustness evaluations.

8. References

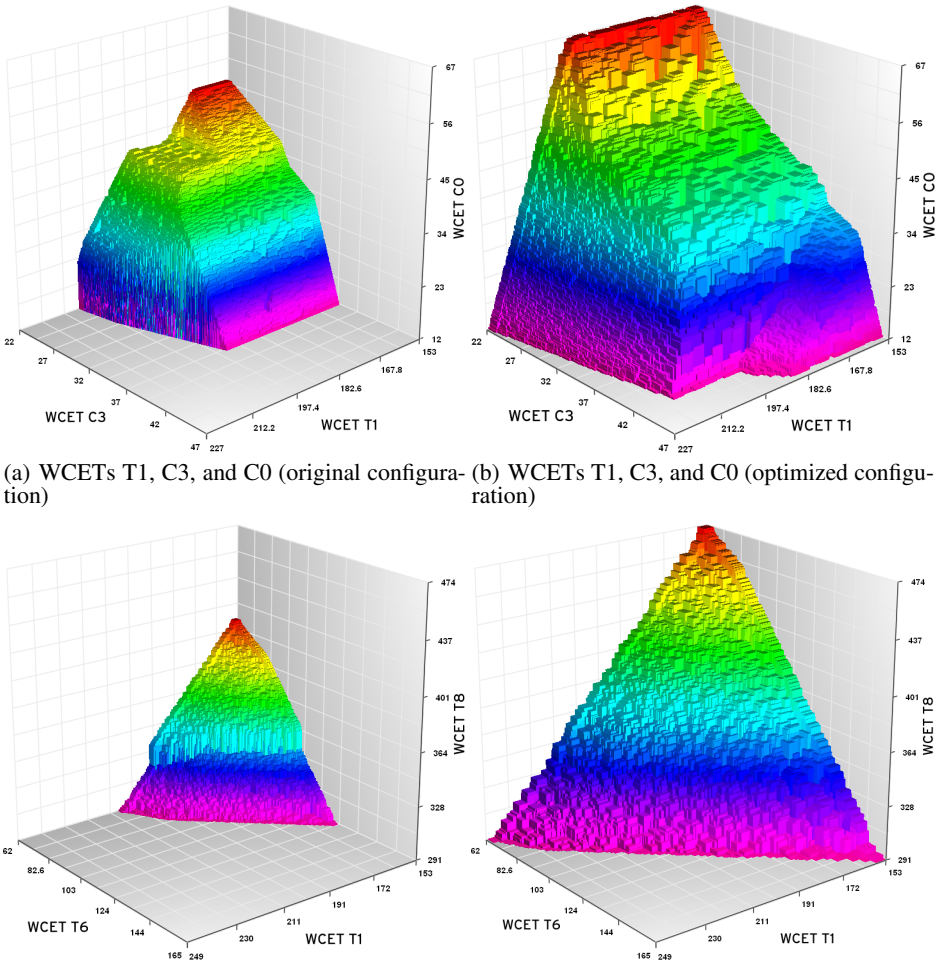
[1] S. Ali, A.A. Maciejewski, H.J. Siegel, and J. Kim. Measuring the robustness of a resource allocation. *IEEE Transactions on Parallel and Distributed Systems*, 15(7):630–641, July 2004.

[2] S. Bleuler, M. Laumanns, L. Thiele, and E. Zitzler. PISA — a platform and programming language independent interface for search algorithms.

[3] S. Chakraborty, S. Künzli, and L. Thiele. A general framework for analysing system properties in platform-based embedded system designs. In *Proc. of the IEEE/ACM Design, Automation and Test in Europe Conference (DATE)*, Munich, Germany, 2003.

[4] M. Fleischer. The measure of pareto optima: Applications to multi-objective metaheuristics. *Lecture Notes in Computer Science*, 2632:519–533, 2003.

[5] D. Gu, F. Drews, and L. Welch. Robust task allocation for dynamic distributed real-time systems subject to multiple



(a) WCETs T1, C3, and C0 (original configuration)

(b) WCETs T1, C3, and C0 (optimized configuration)

(c) WCETs T1, T6, and T8 (original configuration)

(d) WCETs T1, T6, and T8 (optimized configuration)

Figure 8. Three-dimensional WCET sensitivity fronts for original and optimized system configuration

environmental parameters. In *Proc. of the 25th IEEE International Conference on Distributed Computing Systems (ICDCS)*, Columbus, Ohio, USA, June 2005.

[6] A. Hamann, M. Jersak, K. Richter, and R. Ernst. A framework for modular analysis and exploration of heterogeneous embedded systems. *Real-Time Systems Journal*, 33(1-3):101–137, July 2006.

[7] A. Hamann, R. Racu, and R. Ernst. A formal approach to robustness maximization of complex heterogeneous embedded systems. In *Proc. of the IEEE/ACM/IFIP International Conference on HW/SW Codesign and System Synthesis (CODES-ISSS)*, Seoul, South Korea, 2006.

[8] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst. System level performance analysis - the SymTA/S approach. *IEE Proceedings Computers and Digital Techniques*, 152(2):148–166, March 2005.

[9] C. Lu, J.A. Stankovic, S.H. Son, and G. Tao. Feedback control real-time scheduling: framework, modeling, and algorithms. *Real-Time Systems Journal*, 23(1-2):85–126, 2002.

[10] R. Racu, M. Jersak, and R. Ernst. Applying sensitivity analysis in real-time distributed systems. In *Proc. of the 11th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, San Francisco, California, March 2005.

[11] Steve Vestal. Fixed-priority sensitivity analysis for linear compute time models. *IEEE Transactions on Software Engineering*, 20(4), april 1994.

[12] L. While, P. Hingston, L. Barone, and S. Huband. A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1):29–38, February 2006.

[13] E. Zitzler. Hypervolume metric calculation:. *ftp://ftp.tik.ee.ethz.ch/pub/people/zitzler/hypervol.c*, 2001.

[14] E. Zitzler, M. Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm for multiobjective optimization. In *Proc. Evolutionary Methods for Design, Optimisation, and Control*, pages 95–100, Barcelona, Spain, 2002.