

Robust Design of Embedded Systems

Martin Lukasiewicz, Michael Glaß, and Jürgen Teich
University of Erlangen-Nuremberg, Germany
{martin.lukasiewicz,glass,teich}@cs.fau.de

Abstract—This paper presents a methodology to evaluate and optimize the robustness of an embedded system in terms of invariability in case of design revisions. Early decisions in embedded system design may be revised in later stages resulting in additional costs. A method that quantifies the expected additional costs as the robustness value is proposed. Since the determination of the robustness based on arbitrary revisions is computationally expensive, an efficient set-based approach that uses a symbolic encoding as Binary Decision Diagrams is presented. Moreover, a methodology for the integration of the optimization of the robustness into a design space exploration is proposed. Based on an external archive that accepts also near-optimal solutions, this robustness-aware optimization is efficient since it does not require additional function evaluations as previous approaches. Two realistic case studies give evidence of the benefits of the proposed approach.

I. INTRODUCTION

A large number of design decisions are made early in the design process of embedded systems. If one or more of these design decision have to be revised later in the design process, additional costs arise. In a worst-case scenario, a single revised design decision might lead to a string of additional design changes in order to keep the solution feasible in terms of both the design and objectives. A viable approach to avoid these late changes might be the exclusion of all solutions that contain any critical design decision. However, in general this approach is too restrictive, leading either to a lack of feasible designs or unacceptable objectives.

In order to overcome these shortcomings in state-of-the-art embedded system design, this paper introduces the robustness related to design revisions. In this context, a solution is termed robust if the expected additional costs due to design revisions are minor. The determined robustness value shall support a designer in the decision making by additional information about the design alternatives. On the other hand, the robustness might be used in a design space exploration to enable the search for robust solutions.

Contributions: The evaluation of the robustness in embedded system design is a challenging task. Known approaches that consider the direct neighborhood in the solution space are not applicable due to numerous constraints and the resulting low number of feasible solutions. In contrast, an approach based on a given reference set of acceptable solutions is proposed. The focus of this paper are possible design revisions in subsequent design stages that might cause additional costs. An approach that takes arbitrary design revisions into account is presented. Since this approach is computationally expensive, an efficient algorithm based on a symbolic encoding is proposed to overcome this drawback.

In order to optimize the robustness value within a design space exploration, a methodology that incorporates the robustness evaluation in the optimization process is presented. Using an external archive as the reference set to determine the robustness avoids additional computationally expensive objective function evaluations. The external archive is based on an adapted ϵ -archive that also accepts near-optimal solutions to increase the number of acceptable solutions and, thus, also the significance of the determined robustness value.

Two realistic case studies show the benefits of the presented approach: (1) The robustness of several solutions obtained from an exploration of a *Motion-JPEG decoder* is determined to support the decision making for the final implementation. (2) The robustness optimization is applied to an *automotive subnetwork* exploration. Finally, the scalability of the presented methodology is demonstrated by a large-scale synthetic testcase.

Organization of the paper: The remainder of the paper is outlined as follows: Section II discusses related work. Section III presents methods to evaluate the robustness of a solution. The proposed robustness-aware optimization is presented in Section IV. Experimental results are given in Section V before the paper is concluded in Section VI.

II. RELATED WORK

The related work [1], [2], [3], [4], [5] on the evaluation of robustness assumes a dense neighborhood: The problems are either continuous or unconstrained. This prohibits the applicability for a large set of embedded system design problems where the inherent constraints imply a low number of feasible solutions and, thus, a sparse neighborhood. In contrast, the presented approach is generally applicable to embedded system design using knowledge about critical design decisions.

A survey on the optimization of the robustness is given in [6]. Due to the non-linearity of the robustness value, the optimization of the robustness has been studied mostly in a *meta-heuristic* context. Known approaches either approximate the objective functions [3], [4] or use an integration of the continuous solution space [5]. As mentioned in the previous paragraph, these assumptions do not hold for the embedded system design. In [7], the authors suggest a sampling method of several neighborhood solutions and determine the variation of the objectives. However, this approach requires additional function evaluations, forming an obstacle in embedded system design where the evaluation is highly time-consuming. In [2], the robustness is determined by the comparison to the population, i.e., the current solution of the optimization process. Though this approach requires no additional objective function evaluations, this reference set might be too small to be significant. In contrast, the work at hand presents an approach based on an external archive that also accepts near-optimal solutions to increase the size of the reference set of acceptable solutions. Previous work on robustness optimization in the area of embedded systems has been studied in [8], [9], but is restricted to the influence on real-time constraints.

The optimization under *uncertainty* [10] is an orthogonal approach to the presented robustness optimization that deals with inaccuracy of the objectives of the solutions. On the other hand, *sensitivity analysis* [11] like presented in [12], [13] only considers the influence of single design variables on the objectives.

III. ROBUSTNESS EVALUATION

This section presents approaches to determine the robustness of a solution $x \in X_f$. The set X_f represents all feasible solutions, i.e., solutions without any constraint violation. In

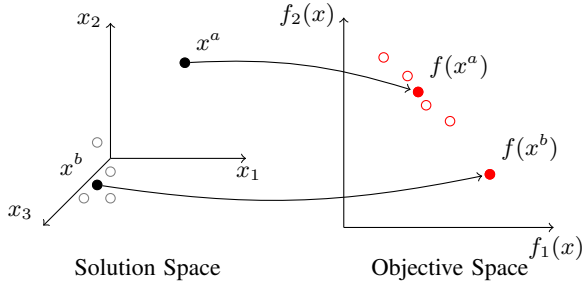


Fig. 1. Based on the density of the neighborhood, solution x^a has a low robustness in the solution space and x^b has a low robustness in the objective space.

embedded system design, many problems are constrained such that the set of feasible solutions is significantly smaller than the complete solution space. The function $f : X_f \rightarrow \mathbb{R}^m$ determines the m objectives of a solution, e.g., monetary costs, energy consumption, etc.

In the following, the robustness function $r : X_f \rightarrow \mathbb{R}^+$ is defined such that a small value indicates a high robustness. All following approaches determine the robustness based on a given reference set $X_A \subseteq X_f$. The reference set X_A represents a set of *acceptable solutions* that may also contain suboptimal solutions in terms of their objectives since minor deteriorations in the objectives are acceptable if a major cost increase due to design revisions can be avoided.

The revision cost function $c : X_f \times X_f \rightarrow \mathbb{R}^+$ determines the arising costs when a solution x is changed to \tilde{x} :

$$c(x, \tilde{x}) = c_{sol}(x, \tilde{x}) + c_{obj}(x, \tilde{x}) \quad (1)$$

This function is composed by the costs in the solution space $c_{sol}(x, \tilde{x})$ (modification costs) and in the objective space $c_{obj}(x, \tilde{x})$ (quality deviation). Figure 1 illustrates the discrepancy of the robustness in the solution space and objective space. The function c is defined by the designer. However, in many cases a weighted euclidean distance in the solution space might be sufficient and the quality deviation can be neglected in high-level design if X_A is a set of acceptable solutions. By definition it holds $c(x, x) = 0$.

A. Neighborhood

A robustness value can be calculated solely based on the neighborhood compared to the set X_A . An intuitive approach determines the costs to the nearest neighbor:

$$r(x) = \min_{\tilde{x} \in X_A \setminus \{x\}} \{c(x, \tilde{x})\} \quad (2)$$

In order to take all other solutions in X_A into account, i.e., the density of the neighborhood, an average of the costs to these solutions is an alternative approach. The generalized mean function is defined as follows:

$$r(x) = \left(\frac{1}{|X_A \setminus \{x\}|} \sum_{\tilde{x} \in X_A \setminus \{x\}} c(x, \tilde{x})^q \right)^{1/q} \quad (3)$$

It is suggested to use the *harmonic mean* with $q = -1$ in order to weight near solutions higher. The complexity of the robustness determination based on the neighborhood is $O(|X_A|)$.

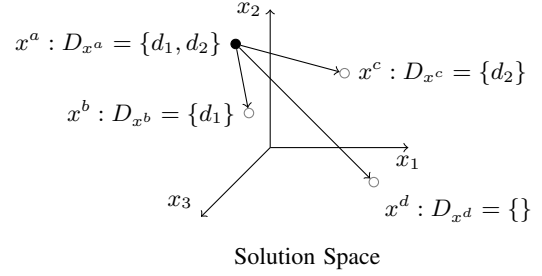


Fig. 2. Possible alternatives for the solution x^a based on the revision of d_1 and d_2 . ($D_x = \{d | t(d, x) = 1\}$)

B. Single Revision

Though the neighborhood might be a good indicator for the robustness, it ignores environmental influences, i.e., the knowledge about critical design decisions. In embedded system design, a critical design decision might be the usage of a component without guaranteed availability, an uncertain implementation of a specific function on a computational unit due to licensing issues, etc. These design decisions are given by a set D . The probability that a design decision has to be revised for a given solution is determined by the function $p : D \times X_f \rightarrow \mathbb{R}_{[0,1]}$. The function $t : D \times X_f \rightarrow \{0, 1\}$ determines if a solution contains a design decision. The set of critical design decisions D and both functions p and t are problem-dependent and defined by the designer.

The robustness value based on the revision of a single design decision is determined as follows:

$$r(x) = \sum_{d \in D} (t(d, x) \cdot p(d, x) \cdot r(d, x)) \quad (4)$$

with

$$r(d, x) = \begin{cases} \min_{\substack{\tilde{x} \in X_A \\ t(d, \tilde{x})=0}} \{c(x, \tilde{x})\} & \text{if } \prod_{\tilde{x} \in X_A} t(d, \tilde{x}) = 0 \\ c_m & \text{else} \end{cases} \quad (5)$$

Equation (4) aggregates the costs for the revision of each design decision in D . Equation (5) determines the minimal revision cost from x to a solution without the design decision d . Here, c_m is a scaling factor used for the residual risk if no solution without the specific design decision d is existent in X_A . In general, if all evaluated solutions x are in X_A , the residual risk is constant and c_m can be set to 0. The complexity of the robustness determination based on single revisions is $O(|D| \cdot |X_A|)$.

C. Multiple Revisions

The evaluation of the single revision robustness is only correct if each solution x contains only at most a single critical design decision. Consider solution x^a in Figure 2 and the case that both design decisions d_1 and d_2 have to be revised. If the evaluation based on single revisions is used, the effective cost $c(x^a, x^d)$ is approximated by $c(x^a, x^b) + c(x^a, x^c)$, leading to a deviation.

As a remedy, the presented approach is extended to determine the robustness based on the revision of an arbitrary number of design decisions, including multiple revisions, as follows:

$$r(x) = \sum_{\tilde{D} \subseteq D} (t(\tilde{D}, x) \cdot p(\tilde{D}, x) \cdot r(\tilde{D}, x)) \quad (6)$$

with

$$t(\tilde{D}, x) = 1 - \left(\prod_{\tilde{d} \in \tilde{D}} (1 - t(\tilde{d}, x)) \right) \quad (7)$$

$$p(\tilde{D}, x) = \prod_{d \in \tilde{D}} p(d, x) \cdot \prod_{d \in D \setminus \tilde{D}} (1 - p(d, x)) \quad (8)$$

$$r(\tilde{D}, x) = \begin{cases} \min_{\substack{\tilde{x} \in X_A \\ t(\tilde{D}, \tilde{x})=0}} \{c(x, \tilde{x})\} & \text{if } \prod_{\tilde{x} \in X_A} t(\tilde{D}, \tilde{x}) = 0 \\ c_m & \text{else} \end{cases} \quad (9)$$

Equation (6) is an extension of Equation (4) by considering all subsets \tilde{D} of design decisions in D instead of only the single design decisions. The function t is extended in Equation (7) to determine if the solution x contains at least one design decision from \tilde{D} . The function p is extended in Equation (8) such that it determines the probability that exactly the design decisions \tilde{D} are revised for a given solution x . Equation (9) corresponds to Equation (5) such that the minimal revision cost to a solution \tilde{x} that contains no design decision in \tilde{D} is determined. The complexity of the robustness determination grows exponentially in $|D|$ with $O(2^{|D|} \cdot |X_A|)$.

An efficient way to implement the robustness determination for multiple design revisions is the following set-based approach: Instead of iterating over all $\tilde{D} \subseteq D$ as in Equation (6), the approach iterates over all solutions X_A sorted in ascending order by the costs $c(x, x^i)$. It holds that x^i is the i -th solution in $X_A \cup \{x\}$ with $x^0 = x$. Here, all $2^{|D|}$ subsets of D are subsequently removed depending on which design decisions the current solution x^i might avoid. This set of subsets of D is symbolically encoded as a Boolean function $b: 2^D \rightarrow \{0, 1\}$ using *Binary Decision Diagrams* (BDDs) [14]. The function b evaluates to 1 if the binary encoded input \tilde{D} is part of b and else to 0. The aggregated probability

$$p(b, x) = \sum_{\tilde{D} \subseteq D \wedge b(\tilde{D})=1} p(\tilde{D}, x) \quad (10)$$

is efficiently determined by the SHANNON decomposition scheme as proposed in [15]:

$$p(b, x) = p(d, x) \cdot p(b_{|d=1}, x) + (1 - p(d, x)) \cdot p(b_{|d=0}, x) \quad (11)$$

The formal approach is given in the following equations:

$$r(b, x, i) = \begin{cases} c(x, x^i) \cdot p(b \wedge \neg b_{x^i}, x) \\ \quad + r(b \wedge b_{x^i}, x, i + 1) & \text{if } i < n \\ 0 & \text{else} \end{cases} \quad (12)$$

with the Boolean functions

$$b_x(\mathbf{D}) = \bigvee_{d \in D} (d \cdot t(d, x)) \quad (13)$$

with $\mathbf{d} \in \mathbf{D}$ being binary variables. The robustness value is determined as follows:

$$r(x) = r(b_x, x, 1) + c_m \cdot p\left(\bigwedge_{\tilde{x} \in X_A \cup \{x\}} b_{\tilde{x}}\right) \quad (14)$$

The complexity of the decomposition in Equation (11) equals the number of nodes in the BDD that scales exponentially in worst-case. Thus, the worst-case complexity remains identical with $O(2^{|D|} \cdot |X_A|)$. However, in many cases the size of the BDDs is not growing exponentially and BDDs can improve the runtime of complex problems by many orders of magnitude, cf. [16].

An iterative algorithm that implements this approach is given in Algorithm 1. The algorithm is initialized with a Boolean function b that encodes the design decisions in x and

Algorithm 1 Iterative algorithm using BDDs for the determination of the robustness $r(x) = value$ based on multiple revisions.

```

1:  $b = b_x$ 
2:  $value = 0$ 
3: for  $i = 1, \dots, |X_A \cup \{x\}|$  do
4:    $value = value + c(x, x^i) \cdot p(b \wedge \neg b_{x^i}, x)$ 
5:    $b = b \wedge b_{x^i}$ 
6:   if  $b = 0$  then
7:     break
8:   end if
9: end for
10:  $value = c_m \cdot p(b, x)$ 

```

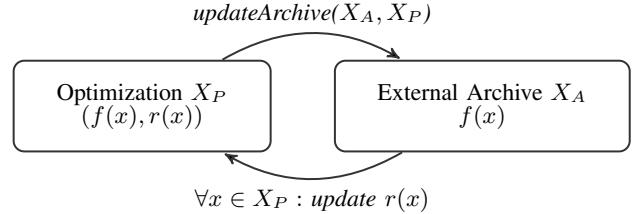


Fig. 3. Illustration an optimization algorithm that also optimizes the robustness by using an external archive.

the robustness $value = 0$ (line 1,2). The solutions in X_A are sorted in ascending order by the revision costs from x . The algorithm iterates the alternative solutions in this order (line 3) and adds the revision costs times the probability of this revision to the $value$ (line 4). For each iteration, the algorithm removes the design revisions that are already avoided by the current solution x^i from the Boolean function b (line 5). If the Boolean function b becomes 0, the algorithm aborts the iteration since all design revisions can be avoided (line 6-8). Finally, the costs for the residual risk times the corresponding probability are added (line 10). The robustness $r(x)$ equals the final $value$.

IV. ROBUSTNESS OPTIMIZATION

This section presents a methodology that allows to incorporate the robustness evaluation into an iterative optimization algorithm. To overcome the drawbacks of known methods, an approach based on an external archive is presented. This approach, as illustrated in Figure 3, assumes an iterative (multi-objective) optimization algorithm, e.g., an *Evolutionary Algorithm*. Additionally, to the original m optimization objectives¹ $f(x)$, the algorithm is extended by an auxiliary robustness objective $r(x)$. The robustness $r(x)$ is calculated based on the set of solutions X_A in the external archive. This external archive X_A is updated each iteration with the new solutions from the optimization algorithm X_P . Since the set X_A is updated iteratively, the robustness of each solution in X_P has to be updated for each iteration. Note that the external archive X_A only considers the original objectives $f(x)$, ensuring that the reference set for the determination of the robustness value $r(x)$ only considers solutions that are acceptable in terms of the original objectives.

In order to tolerate also near-optimal solutions in the external archive X_A , an archiving strategy based on the ϵ -dominance [17] is suggested. Compare Figure 4 to Figure 1 where the robustness of x^a in the solution space and x^b in the

¹Without loss of generality, a minimization of all objectives is assumed.

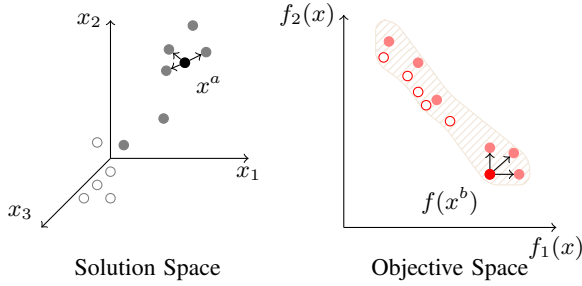


Fig. 4. Increase of the reference set X_A in the external archive in the solution and objective space by tolerating near-optimal solutions.

objective space is significantly increased by tolerating near-optimal solutions. This approach is reasonable since a slight deterioration of the objective functions might be acceptable in case of design revisions.

The archiving strategy based on the ϵ -dominance as studied in [18] is adapted and given in Algorithm 2. The algorithm

Algorithm 2 *updateArchive*(X_A, X_P)

```

1: for all  $x \in X_P \cap X_f$  do
2:   if  $\forall x' \in X_A : \neg(\text{dom}_\epsilon(x', x)) \wedge c(x, x') > 0$  then
3:      $X_A = X_A \cup \{x\}$ 
4:     for all  $\tilde{x}'' \in X_A$  do
5:       if  $\text{dom}_\epsilon(x, \tilde{x}'')$  then
6:          $X_A = X_A \setminus \{\tilde{x}''\}$ 
7:       end if
8:     end for
9:   end if
10: end for

```

iterates over all feasible solutions in X_P (line 1). A solution is added to the archive if it is not ϵ -dominated (dom_ϵ) by any solution in the archive and there is not already an equal solution, i.e., a solution with the revision cost of 0, in the archive (line 2-3). The ϵ -dominance is based on the pareto-dominance:

Definition 1: A solution x weakly dominates \tilde{x} ($x \preceq \tilde{x}$) if $f_i(x) \leq f_i(\tilde{x}) \forall i = \{1, \dots, m\}$.

The common ϵ -dominance is defined as $\text{dom}_\epsilon(x, \tilde{x}) = (x + \epsilon \preceq \tilde{x})$ [18]. In contrast, we suggest the more restrictive dominance

$$\text{dom}_\epsilon(x, \tilde{x}) = (x \preceq \tilde{x}) \wedge \neg(\tilde{x} \preceq x + \epsilon). \quad (15)$$

Figure 5 illustrates the difference: While the former approach tolerates solutions in Y_A and Y_B , the latter approach only tolerates solutions Y_A . Thus, the ϵ_i states the maximal deterioration for each objective f_i compared to the best solutions. If a new solution is added to the archive, all ϵ -dominated solutions in the archive are removed (line 4-8).

V. EXPERIMENTAL RESULTS

The experimental results show the applicability of the presented methodology on two case studies and a synthetic example to emphasize the scalability. The first case study shows how the robustness evaluation supports the designer in the decision making for a *Motion-JPEG decoder system*. The second case study demonstrates the presented robustness-aware optimization approach for the design space exploration of an *automotive subnetwork*. In both case studies, the robustness is determined by the presented evaluation based on multiple design revisions. All following experiments were

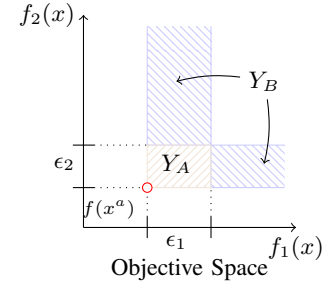


Fig. 5. The common ϵ -dominance tolerates solutions Y_A and Y_B .

carried out on an Intel Core 2 Quad 2.66 GHz with 3GB RAM.

A. Decision Making

The first case study shows the benefits of the presented robustness evaluation for the decision making. In total, 25 solutions for the mapping of an Motion-JPEG decoder application onto a multimedia architecture template are obtained from an optimization algorithm [19]. The objectives of the optimization are the *throughput* in terms of frames per second (fps) and the *chip area* in terms of the number of gates. The application model of the decoder is specified as a *Transaction-Level Model* (TLM) using 21 SystemC single-threaded modules and 56 SystemC channels for data storage communication between modules. The architecture template consists of 12 multi-purpose processing elements and communication resources, as well as 10 dedicated hardware accelerators connected via 36 point to point communication resources including storage memory. Here, the set of multi-purpose processing elements consists of *ARM7*, *ARM9*, and *ARM11* processor-cores, one *DSP*, three memories, two buses, and one gateway connecting these buses.

To allow an appropriate robustness evaluation, several assumptions on the practicability of diverse design decision are made as outlined in Table I: The availability of the *ARM9* processor is guaranteed solely with 80%. The implementation of the *Shuffle1* function on the *ARM7* as well as the *Huffman-Decoder* data mapping on the *Local Memory* are considered as uncertain design decisions due to licensing issues. Moreover, the availability of the *HuffmanDecoder* and *InverseZigZag* IP-cores is uncertain. Here, the cost function c is a weighted euclidean distance in the solution space.

Based on this data, the robustness of the given solutions is evaluated (the runtime of the evaluation is negligible low with less than 1 ms). The results are given in Figure 6 (large circles indicate a high $r(x)$ value, i.e., a low robustness) and some selected solutions are given in detail in Table II. The design decision that has the highest influence on $r(x)$ is d_1 . In many cases, exchanging the *ARM9* processor leads to a string of additional design changes deteriorating the robustness $r(x)$. In this context, the four solutions that fulfill the real-time requirements of at least 24 frames per second and use

	$p(d, x)$	Description
d_1	20%	<i>ARM9</i>
d_2	15%	<i>Shuffle1</i> function on <i>ARM7</i>
d_3	10%	<i>HuffmanDecoder</i> data on <i>Local Memory</i>
d_4	20%	<i>HuffmanDecoder</i> IP-core
d_5	15%	<i>InverseZigZag</i> IP-core

TABLE I
CRITICAL DESIGN DECISIONS OF THE MOTION-JPEG CASE STUDY.

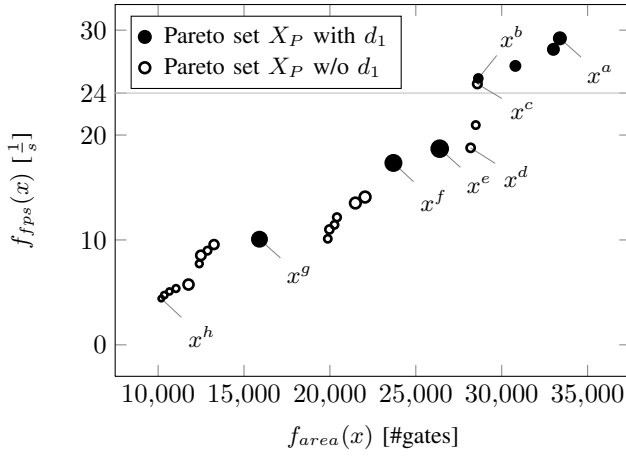


Fig. 6. Results of the evaluation of the robustness for the Motion-JPEG case study. Larger circles indicate a larger $r(x)$ value, i.e., a low robustness.

an *ARM9* processor are more robust: Only some parts of the application are implemented on the *ARM9* and the rest on dedicated hardware as can be seen by the higher number of gates. Table II reveals that some solutions such as x^a, x^b, x^e , and x^f containing the same design decisions have a markedly different $r(x)$ value, ranging from 9.4 to 35.0. This shows that the determination of the probability of design revisions would not be sufficient. This is also emphasized by x^g that just contains a subset of critical design decisions of x^b but has a higher $r(x)$ value and, thus, a lower robustness.

B. Optimization

The second case study gives evidence of the applicability of the proposed robustness-aware optimization approach for the exploration of a typical automotive subnetwork. The network architecture consists of 15 *Electronic Control Units* (ECUs), connected via two *Control Area Network* (CAN) buses, one *FlexRay* bus, and a central gateway. The 9 sensors, and 5 actuators are connected via *Local Interconnect Network* (LIN) buses to the *ECUs*. The application consist of four functions, an *adaptive cruise control*, a *brake-by-wire*, an *air conditioning function*, and a *multimedia control* containing 46 processes and 42 messages in total. The automotive network is optimized in terms of the monetary costs in *Euro*(€) and energy consumption in *Milliamperes*(mA). The monetary costs are approximated by a linear function based on the cost per resource with additional costs, like wiring, being neglected. The energy consumption is approximated by a non-linear energy model based on the average utilization of the *ECUs*. Additionally, each function has a maximal end-to-end delay that has to be satisfied. The optimization is carried out by a

X_P	$f_{fps}(x)$ [$\frac{1}{s}$]	$f_{area}(x)$ [#gates]	$r(x)$	$t(d, x)$				
				d_1	d_2	d_3	d_4	d_5
x^a	29.2	33386	21.2	✓			✓	✓
x^b	25.4	28338	9.4	✓			✓	✓
x^c	24.9	28582	11.3			✓	✓	✓
x^d	18.8	28191	10.2			✓	✓	✓
x^e	18.7	26390	37.2	✓			✓	✓
x^f	17.3	23696	35.0	✓			✓	✓
x^g	10.1	15899	31.1	✓				✓
x^h	4.4	10184	0					

TABLE II
SELECTED SOLUTIONS IN X_P OF THE MOTION-JPEG CASE STUDY.

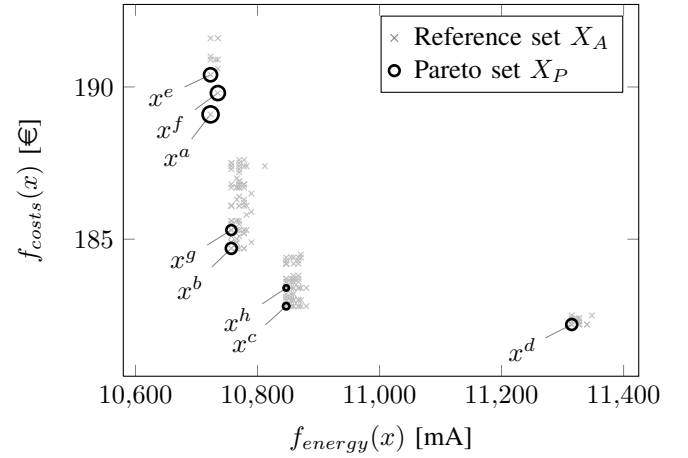


Fig. 7. Optimization results of the automotive case study. Larger circles indicate a larger $r(x)$ value, i.e., a low robustness.

multi-objective Evolutionary Algorithm as proposed in [20].

The critical design decisions for this case study are given in Table III. In particular, diverse design decision, i.e., d_1, d_2, d_3, d_5, d_6 are critical due to the packaging and cable routing in the car. Additionally, d_4 states that it is not verified whether the *Air Conditioning Main function* can be implemented on *ECU8*. The revision costs c are determined by a problem specific non-linear function. Regarding the external ϵ -dominance archive, ϵ is set to $\epsilon_{costs} = 3, \epsilon_{energy} = 60$.

The results of the optimization are given in Figure 7, with $|X_P| = 8$ high quality solutions that are compared to $|X_A| = 173$ solutions in the external archive in the final iteration. The overall runtime was 5 hours and 10 minutes for 5000 function evaluations. Thus, the evaluation of the objective function for each solution took about 3.7 seconds in average. On the other hand, the overhead from the robustness optimization was negligibly small with 17.8 seconds in total for the whole optimization process.

A detailed listing of the solutions in X_P is given in Table IV. An optimization without the robustness evaluation would find only the solutions x^a to x^d and neglect the solutions x^e to x^h as suboptimal, i.e., dominated. However, e.g., x^g and x^h are reasonable alternatives to x^b and x^c , respectively, since they have a higher robustness with a minimal additional cost.

In order to show the applicability of the presented robustness values, a correlation analysis of the robustness evaluation based on multiple design revisions (r_m) is carried out for the 173 solutions in X_A for the evaluation based on the single revisions (r_s), neighborhood with harmonic mean ($r_{q=-1}$), neighborhood with arithmetic mean ($r_{q=1}$), and the nearest neighbor (r_n). The correlations of the robustness value of this case study are given in Figure 8. The correlation between r_m and r_s is very high with 0.99. The high correlation is

	$p(d, x)$	Description
d_1	20%	<i>ECU2</i> on <i>CAN bus 1</i>
d_2	10%	<i>ECU8</i> on <i>FlexRay</i> bus
d_3	15%	<i>DistanceSensor</i> over <i>LIN</i> on <i>ECU9</i>
d_4	25%	<i>Air Conditioning Main function</i> on <i>ECU8</i>
d_5	20%	<i>ECU15</i> on <i>CAN bus 1</i>
d_6	10%	Integration of <i>ECU9</i>

TABLE III
CRITICAL DESIGN DECISIONS OF THE AUTOMOTIVE CASE STUDY.

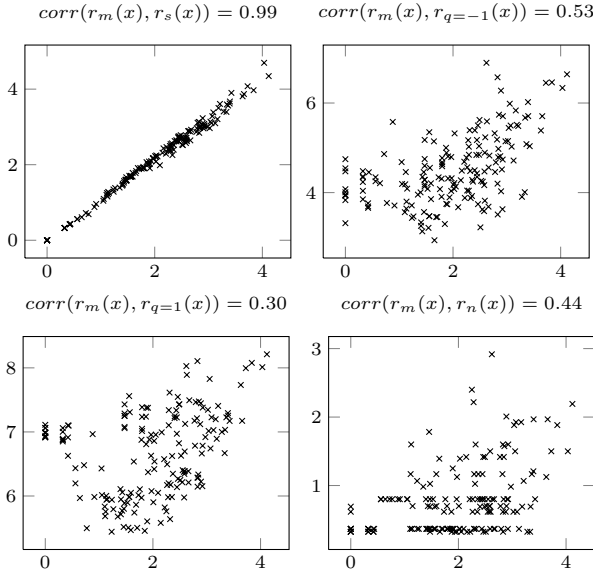


Fig. 8. Correlation results of the automotive case study.

explained by the high number of solutions with only few critical design decision and may decrease as the size of the set D increases. The correlation between r_m and the values based on neighborhood is low with 0.53 for $r_{q=-1}$ and 0.30 for $r_{q=1}$, respectively, as well as 0.44 for r_n . These low values show the restricted significance of the neighborhood based approaches for the determination of the robustness.

C. Scalability

In order to give evidence of the scalability of the presented approach, a large synthetic testcase is used. The testcase is a *system synthesis* problem, consisting of the mapping of 156 tasks to 75 resources. The set of critical design decisions D has the size 30. The robustness evaluation based on multiple revisions is used in a multi-objective optimization that is terminated after 10000 objective function evaluations. Though the worst case complexity for evaluation of the robustness based on multiple revisions is exponential, the overhead in runtime for the evaluation is 443 seconds, i.e., only about 44 milliseconds per objective function evaluation. This runtime is only a fraction of a single evaluation of real-world problems and, thus, to a high degree acceptable.

VI. CONCLUSION

This paper presents a methodology that introduces the robustness in the decision making process as well as in the optimization of embedded systems. For the evaluation of the robustness of a solution, several measures are proposed.

X_p	$f_{costs}(x)$ [€]	$f_{energy}(x)$ [mA]	$r(x)$	d_1	d_2	d_3	d_4	d_5	d_6
x^a	189.1	10723	3.93	✓				✓	✓
x^b	184.7	10757	1.80			✓		✓	✓
x^c	182.8	10847	0.325					✓	
x^d	182.2	11315	1.78	✓				✓	
x^e	190.4	10723	2.49	✓		✓			✓
x^f	189.8	10735	2.77	✓		✓		✓	✓
x^g	185.3	10757	1.47			✓			✓
x^h	183.4	10847	0.0						

TABLE IV
SOLUTIONS IN X_P OF THE AUTOMOTIVE CASE STUDY.

Taking multiple design revisions into account, the complexity becomes exponential. To cope with this complexity, an efficient approach based on Binary Decision Diagrams is presented. Moreover, a methodology that aims to integrate the optimization of the robustness in a design space exploration is proposed. Since this approach requires no additional objective function evaluations, the runtime of the optimization is not increased significantly.

Two realistic case studies give evidence of the benefits of the presented methods. A case study of a Motion-JPEG decoder shows how a set of potential implementation are evaluated to support the decision making. A second case study demonstrates the robustness-aware optimization of an automotive subnetwork and reveals that a common optimization method would ignore reasonable robust solutions. A scalability analysis based on a synthetic testcase proves the applicability in terms of runtime of the methodology also for large-scaled problems.

REFERENCES

- [1] M. Li, S. Azarm, and V. Aute, "A Multi-objective Genetic Algorithm for Robust Design Optimization," in *Proc. of GECCO '05*, 2005, pp. 771–778.
- [2] Y. Jin and B. Sendhoff, "Trade-off between Performance and Robustness: An Evolutionary Multiobjective Approach," in *Proc. of EMO 2003*, 2003, pp. 237–251.
- [3] J. Branke, "Creating Robust Solutions by Means of Evolutionary Algorithms," in *Proc. of PPSN V*, 1998, pp. 119–128.
- [4] I. Paenke, J. Branke, and Y. Jin, "Efficient Search for Robust Solutions by Means of Evolutionary Algorithms and Fitness Approximation," *Evolutionary Computation*, vol. 10, no. 4, pp. 405–420, 2006.
- [5] X. Du and W. Chen, "Towards a better understanding of modeling feasibility robustness in engineering design," *ASME Journal of Mechanical Design*, vol. 122, no. 4, pp. 385–394, 2000.
- [6] H.-G. Beyer and B. Sendhoff, "Robust Optimization - A Comprehensive Survey," *Computer Methods in Applied Mechanics and Engineering*, vol. 196, no. 33-34, pp. 3190–3218, 2007.
- [7] K. Deb and H. Gupta, "Introducing Robustness in Multi-objective Optimization," *Evolutionary Computation*, vol. 14, no. 4, pp. 463–494, 2006.
- [8] A. Hamann, R. Racu, and R. Ernst, "Multi-dimensional Robustness Optimization in Heterogeneous Distributed Embedded Systems," in *Proc. of RTAS '07*, 2007, pp. 269–280.
- [9] —, "A Formal Approach to Robustness Maximization of Complex Heterogeneous Embedded Systems," in *Proc. of CODES+ISSS '06*, 2006, pp. 40–45.
- [10] Y. Jin and J. Branke, "Evolutionary Optimization in Uncertain Environments - A Survey," *Evolutionary Computation*, vol. 9, no. 3, pp. 303–317, 2005.
- [11] T. Homma and A. Saltelli, "Importance Measures in Global Sensitivity Analysis of Nonlinear Models," *Reliability Engineering and Systems Safety*, vol. 52, no. 1, pp. 1–17, 1996.
- [12] R. Racu, A. Hamann, and R. Ernst, "A Formal Approach to Multi-Dimensional Sensitivity Analysis of Embedded Real-Time Systems," in *Proc. of ECRTS '06*, 2006, pp. 3–12.
- [13] S. Vestal, "Fixed-Priority Sensitivity Analysis for Linear Compute Time Models," *IEEE Transactions on Software Engineering*, vol. 20, no. 4, pp. 308–317, 1994.
- [14] R. E. Bryant, "Symbolic Boolean Manipulation with Ordered Binary Decision Diagrams," *ACM Computing Surveys (CSUR)*, vol. 24, no. 3, pp. 293–318, 1992.
- [15] A. Rauzy, "New Algorithms for Fault Tree Analysis," *Reliability Eng. and System Safety*, vol. 40, pp. 202–211, 1993.
- [16] J. Burch, E. Clarke, K. Mcmillan, D. Dill, and L. Hwang, "Symbolic model checking," in *Proc. of Symposium on Logic in Computer Science*, 1990, pp. 1–33.
- [17] P. Loridan, " ϵ -Solutions in Vector Minimization Problems," *Journal of Optimization, Theory and Applications*, vol. 43, no. 2, pp. 265–276, 1984.
- [18] O. Schuetze, C. A. Coello Coello, E. Tantar, and E.-G. Talbi, "Computing Finite Size Representations of the Set of Approximate Solutions of an MOP with Stochastic Search Algorithms," in *In Proc. of GECCO '08*, 2008, pp. 713–720.
- [19] M. Lukasiewicz, M. Streubühr, M. Glaß, C. Haubelt, and J. Teich, "Combined System Synthesis and Communication Architecture Exploration for MPSoCs," in *Proc. of DATE '09*, 2009, pp. 472–477.
- [20] M. Glaß, M. Lukasiewicz, J. Teich, U. Bordoloi, and S. Chakraborty, "Designing Heterogeneous ECU Networks via Compact Architecture Encoding and Hybrid Timing Analysis," in *Proc. of DAC '09*, 2009, pp. 43–46.