Uncertainty-Aware Compositional System-Level Reliability Analysis

Hananeh Aliee, Michael Glaß, Faramarz Khosravi, and Jürgen Teich

Acronyms

BDD	binary decision diagram
CRA	compositional reliability analysis
CRN	compositional reliability node
CDF	cumulative distribution function
DSE	design space exploration
EA	evolutionary algorithm
ESL	electronic system level
MOEA	multi-objective evolutionary algorithm
MPSoC	multiprocessor system-on-chip
MTTF	mean-time-to-failure
RAL	reliability abstraction level
RTC	Real-Time Calculus
SER	soft error rate

H. Aliee
Helmholtz Zentrum München, Munich, Germany
e-mail: hananeh.aliee@helmholtz-muenchen.de

M. Glaß (✉)
Ulm University, Ulm, Germany
e-mail: michael.glass@uni-ulm.de

F. Khosravi · J. Teich
Friedrich-Alexander-Universität Erlangen-Nürnberg, Erlangen, Germany
e-mail: faramarz.khosravi@fau.de; juergen.teich@fau.de

J. Henkel, N. Dutt (eds.), *Dependable Embedded Systems*, Embedded Systems,
https://doi.org/10.1007/978-3-030-52017-5_19

1 Introduction

Continuous technology scaling necessitates to design today's embedded systems from electronic components with growing inherent unreliability. This unreliability arises from susceptibility to neutron-induced soft errors, negative-bias temperature instability, short-channel effect, gate leakage, etc. Therefore, it is vital to analyze system reliability at design time and employ appropriate reliability-improving techniques if necessary. A variety of reliability analysis techniques have been proposed for both the relatively low levels of abstraction that focus on technology as well as the system level that considers the interplay of hardware and software. But, there exists a gap between the levels where the faults originate, e. g., transistor level, and the system level for which the analysis is required. To close this gap and tame the ever increasing system complexity, *cross-level* analysis methodologies are required. These collect knowledge at lower levels by combining different analysis techniques and provide proper data for the analysis at higher levels of abstraction [24].

Evaluating the reliability of a system at design time, proper reliability-improving techniques can be explored and integrated into the system. However, these techniques typically come with higher monetary costs, latency, energy consumption, etc. This necessitates a multi-objective **DSE!** (**DSE!**) which maximizes reliability without deteriorating other design objectives. Usually, **DSE!** explores and evaluates millions of possible design alternatives (also called implementations) to find the Pareto-optimal ones. Herein, the efficiency of the reliability evaluation and exploration algorithm are the main challenging issues [2]. In [3], we propose an efficient and scalable reliability analysis technique based on Success Trees (STs) which is integrated into a **DSE!** framework to automatically evaluate an implementation's reliability. Most existing analysis techniques quantify a system's reliability without giving any hint on what to change to improve it, such that exploration algorithms basically perform random changes, e. g., through genetic operators in case of **EA!**s (**EA!**s). In [4, 6, 7, 11], we propose to employ the notion of *component importance* to rank components based on their contribution to the system reliability. Later, to improve the reliability of a system with limited budgets, we only need to improve the reliability of highly important components. In [5, 28], we show this guides the **DSE!** towards highly reliable, yet affordable implementations. So far, most existing analysis approaches assume that the reliabilities of components—or their lower bound—are more or less known precisely. Due to shrinking cell geometries, semiconductor devices encounter higher susceptibility to environmental changes and manufacturing tolerances such that a component's reliability has to be considered *uncertain*. An overview of the most important types of uncertainties for system design is given in Fig. 1.

Effects of unreliability and the associated uncertainty of components can propagate to the system level and become a challenge for system-level design methodologies. Even worse, destructive effects such as extreme temperature can affect several components simultaneously, resulting in correlated uncertainties. Neglecting such correlations can impose an intolerable inaccuracy to reliability analysis.

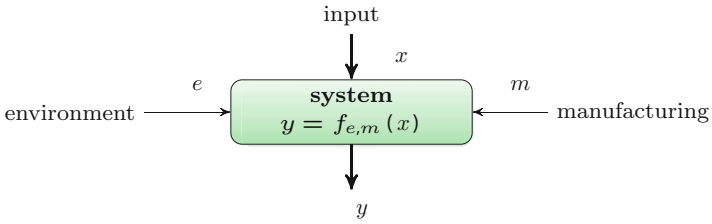

Fig. 1 Uncertainties that influence a system's response and its characteristics: Uncertain environmental influences $\triangle e$ like cosmic rays may cause soft errors. Manufacturing tolerances $\triangle m$ may lead to changing system behavior and permanent defects. Finally, uncertainty may also be present in case the inputs to a system may vary ($\triangle x$) or might not be known at design time

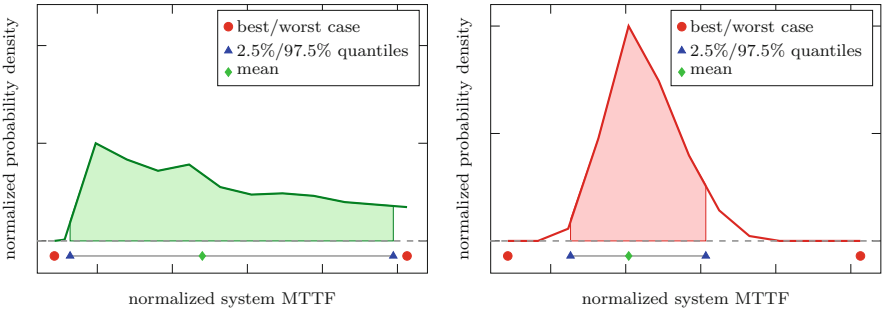

Fig. 2 Impact of uncertainty correlation among the reliability functions of different components on the uncertainty of the system **MTTF!** for an implementation candidate of an H.264 encoder/decoder. (**a**) Correlated component uncertainty. (**b**) Non-correlated component uncertainty

As an example, Fig. 2 depicts the distribution of system **MTTF!** (**MTTF!**) for an H.264 encoder/decoder implementation with and without the consideration of uncertainty correlations. While considering these correlations shows a good match between the simulated cases and the bounds, neglecting them may result in huge deviations from those bounds. This motivates the consideration of uncertainties and especially their correlations in cross-level reliability analysis. In this realm, this chapter introduces a methodology for **CRA!** (**CRA!**) that combines various reliability analysis techniques across different levels of abstraction while being aware of existing uncertainties and their correlations.

Considering uncertainty, system reliability is no longer a single value, but instead represented by a set of samples, upper and lower bound curves, or distribution functions which requires that a **DSE!** can consider implementations with uncertain objectives. Therefore, this chapter focuses on (a) the explicit modeling of uncertainties and their correlations in reliability analysis and (b) the integration of such an analysis into a framework for system-level **DSE!**. The techniques proposed are not tailored to a specific abstraction layer, but can be best classified as combining

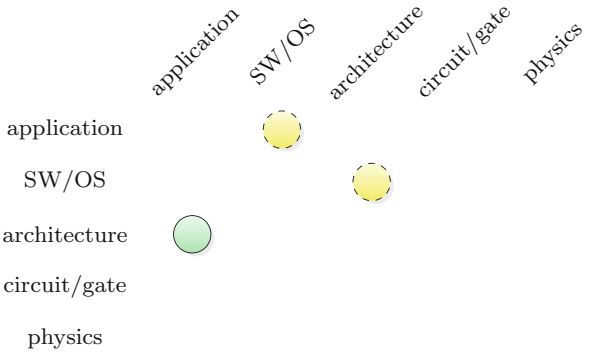

Fig. 3 Main abstraction layers of embedded systems and this chapter's major (green, solid) and minor (yellow, dashed) cross-layer contributions

architecture and application layers according to the embedded system abstraction layers as depicted in Fig. 3.

The rest of this chapter is organized as follows: Sect. 2 reviews related work and introduces required fundamentals. Section 3 introduces a formal **CRA!** framework and its application using a case study. An explicit modeling of uncertainty in reliability analysis and optimization is given in Sect. 4. Finally, Sect. 5 concludes the chapter.

2 Related Work and Fundamentals

2.1 Reliability Analysis and Optimization

Reliability analysis and optimization are thoroughly studied research topics that are of great importance fornearly every safety-critical system [12], especially embedded systems [36]. However, one can observe that the different areas raise significantly diverse needs for the applied analysis techniques. An overview of well-known reliability analysis techniques can be found in [35].

Up to now, several approaches have been presented for analyzing the reliability of embedded systems at system level which are typically integrated into system-level **DSE!**. In [16], fault-tolerant schedules are synthesized using task re-execution, rollback recovery, and active replication. The authors of [47] try to maximize reliability by selectively introducing redundancy while treating area consumption and latency as constraints. Reliability is introduced as an objective into system-level design in [13]. However, the employed reliability analysis techniques are restricted to series-parallel system structures which render them infeasible for typical embedded systems where processing and communication resources have to be shared. On the other hand, reliability analysis at low levels of abstraction has been studied thoroughly, e. g., transistor level [42] or for prospective switching devices like carbon nanotubes [32].

So far, few systematic approaches have been proposed to upscale the knowledge gathered at low abstraction levels to the system level. The work in [1] proposes a system-level dynamic temperature management scheme to prevent thermal hot spots through a run-time application re-mapping, and to efficiently mitigate aging effects in a many-core architecture. In [23], thermal effects in a Multiprocessor System-on-Chip (MPSoC) on its reliability are propagated into a scheduling and binding optimization at system level. Their analysis is based on a simulation of the MPSoC and given relations between the temperature profile and the resulting reliability. Similar reliability analysis techniques are used in [34] in order to optimize the lifetime of MPSoCs using the so-called *slack allocation*. However, these techniques are able to capture thermal effects only, without investigating the possibility to include and propagate these effects into a more holistic analysis that also takes into account, e. g., soft errors or a complex system structure like a networked embedded system consisting of several interconnected processors or MPSoCs.

2.2 Compositional Approaches to Reliability Analysis

A first attempt to close the gap on accurate power models for reliability analysis between the **ESL!** (**ESL!**) and the gate level is presented in [40]. While the approach sounds promising in modeling thermal effects on the component reliability, it fails to offer a formal framework that allows to integrate different analysis techniques cross level. Herkersdorf et al. [24] [RAP-Chap.] propose a framework for probabilistic fault abstraction and error propagation and show that all physically induced faults manifest in higher abstraction levels as a single or multiple bit flip(s). Similarly, the proposed **CRA!** model aims to propagate the effects of uncertainty and the resulting faults originating from lower levels of abstraction into the system-level analysis by incorporating appropriate reliability analysis techniques for each relevant *error model* at a specific level of abstraction. As a result, the developed concepts become independent of an actual error model since it abstracts from the actual source of unreliability during *upscaling*, i. e., the propagation of data from lower levels to higher levels by means of abstraction and data conversion. **CRA!** approaches that consider component-based software are presented in [37]. Although these approaches try to develop a more general compositional analysis scheme, they miss a well-defined mathematical underpinning and do not focus on automatic analysis as needed during **DSE!**.

The use of composition and decomposition in well-defined formal models that allow abstraction to avoid state space explosion has been addressed in, e.g., [25]. An especially interesting and formally sound approach can be found in [9]. In this chapter, we develop a formal approach, inspired by techniques from the verification area, for **CRA!**. A particular challenge will be the consideration and explicit modeling of uncertainties in the formal model where there is no similar technique or need in the area of verification given.

2.3 Uncertainty Considerations

There exist intense studies on the effect of uncertainties on the system reliability for general engineering problems, cf. [38], as well as circuits and microarchitectures, cf. [27]. However, few studies focus on the cross-level reliability analysis of embedded systems in the presence of uncertainty arising from manufacturing tolerances, etc.

Uncertainty-Aware Analysis One example is a cross-level adaptive reliability prediction technique proposed in [17] that derives information from different levels of abstraction and allows to consider the simultaneous effects of process, voltage, temperature and aging variations, and soft errors on a processor. The authors of [18] propose a cross-level framework to analyze the combined impact of aging and process variation on the **SER!** (**SER!**) and static noise margin of memory arrays in near threshold voltage regimes. This framework enables to explore workload, as instruction per second, and cache configuration, as cache size and associativity, in order to minimize **SER!** and its variations for 6T and 8T SRAM cells. Contrary to all mentioned approaches, this chapter explicitly treats each effect of uncertainty during reliability analysis of a system. Proposed is an analysis technique that obtains the range of reliability that is achievable for a system given its configuration and the uncertainties of its components.

Uncertainty-Aware Optimization Optimization problems may be affected by various sources of uncertainty including perturbation of decision variables as well as effects of noise and approximation on objective functions [26]. In this work, uncertainty is explicitly modeled as variations in component failure rates and costs. The uncertainty propagates through reliability analysis and cost evaluation at system level and renders design objectives to be uncertain as well. To make correct decisions when comparing and discriminating implementations during **DSE!**, the employed optimization algorithm needs to take the uncertainty of the design objectives into account as well. The work in [44] proposes a mathematical approach to calculate the probability of an implementation dominating another, given all uncertain objectives follow either uniform or any discrete distributions. However, extending this approach to consider diversely distributed uncertain objectives requires solving difficult integrals demanding a huge computational effort. To this end, approximate simulation-based approaches, e. g., in [30], provide trade-offs between execution time and accuracy of calculating this probability. In [33], it is proposed to compare uncertain objectives with respect to their lower and upper bounds. However, this approach fails to distinguish largely overlapping intervals with even significantly different distributions. A lot of work has been proposed for problems with continuous search spaces and linear objective functions, see e. g.,[15]. However, typical embedded system design problems have discrete search spaces, non-linear and often not differentiable objective functions, and have to cope with stringent constraints. Thus, these optimization techniques cannot be applied without further investigation and modification. In [39], an approach based on an

uncertainty-aware ***MOEA! (MOEA!)*** that targets reliability as one design objective is presented. The approach takes into account the uncertainty of the reliability of each system component and tries to maximize the robustness of the system. This chapter presents a novel uncertainty-aware multi-objective optimization approach applicable for **DSE!** of reliable systems at system level, see Sect. 4.

2.4 System-Level Design Fundamentals

This chapter targets the system-level design of embedded MPSoCs, typically specified by an application graph, a resource graph, and a set of possible task-to-resource mappings. The application graph includes a set of tasks to be executed and specifies the data and control flow among them. The resource graph consists of hardware resources, namely, processors and accelerators connected by communication infrastructures such as buses or networks-on-a-chip. The mappings specify which tasks can be executed on which resources. Figure 4 shows an example specification with three tasks t_i, $i \in [0 \ldots 2]$, five resources r_j, $j \in [0 \ldots 4]$, and eight mappings $m_{i,j}$ from t_i to r_j.

Implementation candidates are derived via system-level synthesis [10] performing the steps: (a) *Resource allocation* selects a subset of resources that are part of the implementation. (b) *Task binding* associates at least one instance of each task to an allocated resource by activating the respective task-to-resource mapping. (c) *Scheduling* determines a feasible start time for each task instance. An implementation is *feasible* if and only if all constraints regarding, e. g., communication, timing, or utilization are fulfilled. Figure 4 highlights a possible feasible implementation

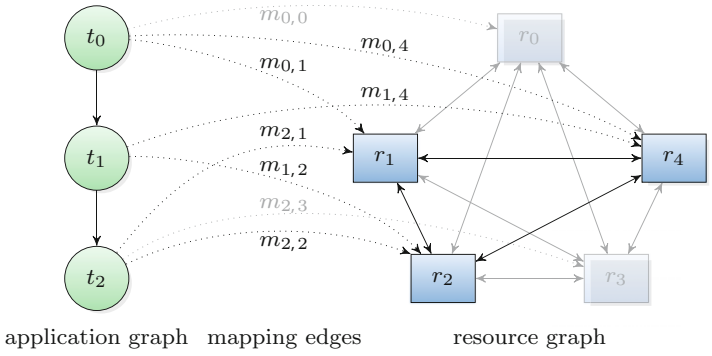

Fig. 4 A specification comprising (**a**) an application graph where edges indicate data dependencies of tasks, (**b**) a resource graph with edges representing dedicated communication between resources, and (**c**) a set of task-to-resource mappings which model possible execution of tasks on resources. A possible implementation candidate obtained by system-level synthesis is depicted with non-allocated resources and non-active bindings being grayed out

with non-allocated resources and non-activated mappings being grayed out. More details of the underlying system synthesis and **DSE!** in the context of reliability analysis and optimization can be found in [22, 43].

3 Compositional Reliability Analysis (CRA)

This section introduces models and methods for **CRA!** as proposed in [21]. Figure 5 shows a schematic view of **CRA!** and its required mechanisms. To realize a cross-level analysis, it encapsulates existing reliability analysis techniques in **CRN!**s (**CRN!**s) at multiple **RAL!**s (**RAL!**s). It tames analysis complexity within a certain **RAL!** using composition and decomposition and connects different **RAL!**s through adapters. Each **CRN!** applies an analysis step $\mathcal{Y}(t) = X(S)$ at a specific **RAL!** where X is a concrete analysis technique and S is a (sub)system. A **CRN!** derives a specific measure $\mathcal{Y}$ over time t. A **RAL!** in **CRA!** may combine several (design) abstraction levels where the same errors and, especially, their causes are significant.

Adjacent **RAL!**s are connected by the concept of *adapters* that have to perform three tasks: (a) *refinement* provides the data required for the analysis in the lower **RAL!**, (b) *data conversion* transforms the output measures from the lower **RAL!** to the input required at the higher **RAL!**, and (c) *abstraction* during both refinement and data conversion tames analysis complexity. A concrete example of **CRA!** describing a temperature-reliability adapter for MPSoCs is presented in Sect. 3.1.

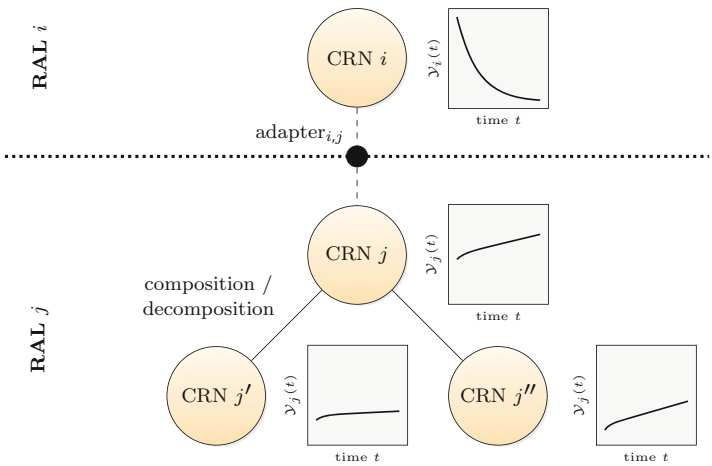

Fig. 5 A schematic view of **CRA!**

Another important aspect of **CRA!** concerns the feasibility of composition and decomposition with respect to reliability analysis. While, of course, composition and decomposition should reduce the complexity of the analysis, errors caused by approximation or abstraction should be bounded. For example, a rule to bound the approximation error of a decomposition D of a given system S into n subsystems $S_1, \ldots, S_n$ is as follows:

$$D(S) = \{S_1, \ldots, S_n\} \text{ is feasible, if } \exists \epsilon : |X(S) - \big(X(S_1) \circ \ldots \circ X(S_n)\big)| \leq \epsilon$$

with $\circ$ being an analysis-dependent operator, e. g., multiplication, and ϵ being the maximum approximation error. A special focus of these investigations is the proper handling of decomposed nodes that influence each other. Nowadays, hardly any subsystem of an embedded system is truly independent of all other subsystems. Thus, this rule should be extended as follows to consider both the truly independent individual properties of the decomposed nodes and their dependencies during composition C:

$$D(S) = \{S_1, \ldots, S_n\} \text{ is feasible, if } \exists \epsilon :$$
$$|X(S) - C\big(X(S_1) \circ \ldots \circ X(S_n),\, P(\{S_1, \ldots, S_n\})\big)| \leq \epsilon. \tag{1}$$

In this case, the composition C not only takes into account the parts of the subsystem that can be analyzed independently, but also performs a corrective postprocessing P to take into account their interactions.

Similarly, we have developed rules for the connection of different **RAL!**s. The task of an adapter is to convert the measure $\mathcal{Y}$ used at the lower **RAL!** into the measure $\mathcal{Y}'$ used at the higher **RAL!**s, for example, $\mathcal{Y}' = A(\mathcal{Y})$. Especially because of the models and methods needed for converting from one **RAL!** to another, a thorough analysis of the function A needs to be carried out. In most cases, this function will not provide an exact result, but will require an abstraction such as by the determination of tight upper and lower bounds. Thus, the developed rules will define requirements for the functions in the adapter used for abstraction and data conversion.

3.1 CRA Case Study and Uncertainty Investigations

In [21], a concrete application of **CRA!** to realize a temperature-aware redundant task mapping approach is presented. In the following, a brief summary of the case study is given with focus being put on the aspect of uncertainty introduced due to the application of composition and decomposition. This further motivates the need to develop techniques to explicitly model and consider uncertainty during analysis and optimization as is presented in Sect. 4.

3.1.1 CRA Case Study

In the context of system-level design and especially the design of reliable embedded systems as introduced in Sect. 2, deploying redundant (software) tasks can be considered a rather cost-efficient technique to enhance system reliability. However, the resulting additional workload may lead to increased temperature and, thus, a reliability degradation of the (hardware) components executing the tasks. The case study in [21] combines three different techniques on three **RAL!**s: At the highest **RAL!**, a reliability analysis based on **BDD!** (**BDD!**), see, e. g.,[20], computes the system reliability of a complete 8-core MPSoC and requires the reliability function of each component (core) in the system. To determine the latter, an intermediate **RAL!** uses the behavioral analysis approach **RTC!** (**RTC!**) [45] to derive the upper bound for the workload of each core over time. This workload is passed to the lowest **RAL!** where this information is used to carry out a temperature simulation based on HotSpot [41] to deliver a temperature profile of each core. Using these temperature profiles and assuming electromigration as a fault model, [21] proposes an adapter that—based on the works in [14, 46]—delivers a temperature-aware reliability function for each core back to the highest **RAL!** in order to complete the system analysis.

3.1.2 Uncertainty Investigations

As given in Eq. 1, composition/decomposition may result in an imprecision ϵ_o of an output measure $o \in O$. In [19], we present techniques for formal decomposition and composition for **CRN!**s that describe the system via Boolean formulas, typically used by **BDD!**s, Fault Trees, etc. Here, functional correlations between components are fully captured in the Boolean formulas, and we propose an exact composition/decomposition scheme on the basis of early quantification. However, correlations are typically non-functional, with heat dissipation between adjacent cores being a prominent one. Consider again the case study described before and Fig. 6: Not decomposing the system into individual cores results in a temperature simulation of all cores at the lowest level, implicitly including the effect of heat dissipation in-between cores, see Fig. 6 (top-left). A naive decomposition could decompose the system into independent cores such that the workload of each core is determined and a reliability function would be gathered by per-core temperature simulations on the lowest level, see Fig. 6 (middle-left). This, however, would completely neglect the effect of heat dissipation between cores. As a third option, [21] investigates a *corrective postprocessing* within the adapter between the lower levels where the workload of cores and the temperature simulation are analyzed independently, while a simple model that considers the distance and steady-state temperature of each core is used to approximate the respective heat flow, see Fig. 6 (bottom-left).

The imprecision resulting from the three discussed decomposition variants is given in Fig. 6 (right), derived from ≈8000 different system implementations

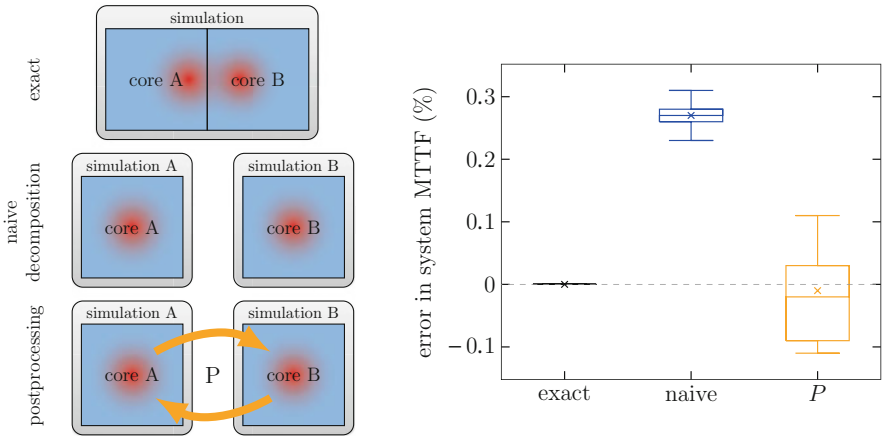

Fig. 6 A simulation of two cores captures heat dissipation (exact, top) while a decomposition (naive, middle) is unable to capture heat dissipation between cores. A corrective postprocessing (P, bottom) enables to reduce analysis complexity while providing a basic notion of heat dissipation. The resulting imprecisions in percentage on system-wide **MTTF!** are depicted on the right

analyzed as part of a **DSE!**: While no decomposition is treated as an exact base value—with respect to heat dissipation being considered and not the overall exactness of the simulation—the naive decomposition constantly overestimates the system-wide **MTTF!** by ≈26%. On the other hand, the corrective postprocessing delivers results with a rather good match in terms of the median and average error, but also shows that the correction may come at errors of up to ≈10%. At the same time, compared to the complete simulation, the decomposition including corrective postprocessing achieves a ≈2 × average speed-up. These results further motivate the need for analysis and optimization techniques—as presented in the next section—that can explicitly model uncertainty such as the shown imprecision.

4 Uncertainty in Reliability Analysis and Optimization

To design and optimize systems for reliability, existing uncertainties in their environment and internal states, see Fig. 1, must be explicitly integrated into reliability analysis techniques. Implicit uncertainty modeling hides the effects of controllable and non-controllable uncertainties, e. g., into a single reliability function, and fails to distinguish between them. On the other hand, explicit modeling determines the range of achievable reliability of a component or subsystem, e. g., using upper and lower bound functions.

We introduce two solutions for uncertainty modeling: (a) using upper and lower bounding curves for the achievable reliability and (b) abstracting various uncertainties into a finite set of typical use cases and providing a system reliability

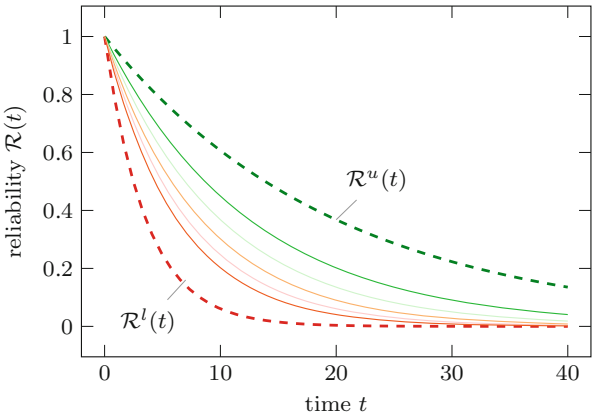

Fig. 7 Reliability functions $\mathcal{R}(t)$ that result from uncertainties in the internal heat dissipation of a silicon system that arise from, e. g., changing task binding on a processing unit. Shown is a range that is determined by an upper $\mathcal{R}^u(t)$ and a lower bound $\mathcal{R}^l(t)$ reliability function and the reliability functions for 5 use cases, e. g., 5 favored task schedules that show a distribution within the range

function for each case, see Fig. 7. While the former offers a range and abstracts from the distributions in between bounds, the latter variant explicitly determines important cases in that range, but of course, comes with an increased complexity. This section covers both approaches and assumes that uncertainty obtained from lower abstraction levels is available at higher levels as known distributions or sampled data.

As introduced earlier, incorporating reliability-increasing techniques into a system at design time may deteriorate other design objectives. Due to the explicit modeling of uncertainties, a multi-objective uncertainty-aware optimization becomes necessary. Given that the system reliability is no more a single value, optimization algorithms must be able to handle uncertain objectives given as probability distributions, a set of samples or upper and lower bound curves, and allow for a quantitative comparison of different designs.

4.1 Uncertainty-Aware Reliability Analysis

The uncertainty-aware reliability analysis technique introduced in the following is originally proposed in [29]. It models the reliability of a component r with uncertain characteristics $\mathcal{U}_r$ using reliability functions $\mathcal{R}_r(t)$ that are distributed within given lower and upper bound reliability functions, i. e., $\mathcal{U}_r = \left[\mathcal{R}_r^l(t), \mathcal{R}_r^u(t)\right]$. A *sampler* is used to take $\mathcal{U}_r$ as input and deliver a sampled reliability function $\mathcal{R}_r^s(t)$ with $\mathcal{R}_r^l(t) \leq \mathcal{R}_r^s(t) \leq \mathcal{R}_r^u(t)$. It ensures that the sampled reliability functions follow the intended distribution within the given bounds, and enables

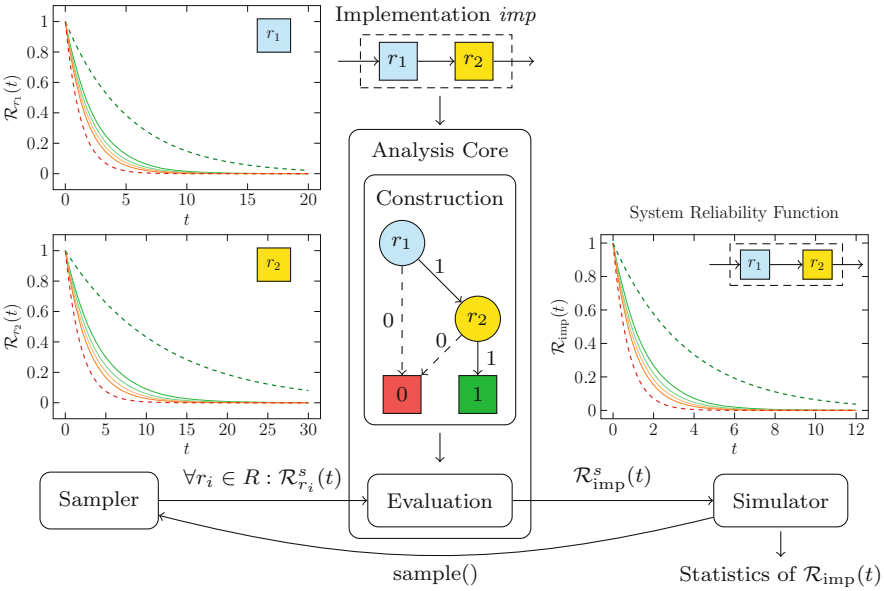

Fig. 8 An overview of the proposed uncertainty-aware reliability analysis

the consideration of arbitrary distributions, in particular well-known discrete and continuous distributions.

In practice, component reliability is typically derived from measurements that are fitted to closed-form exponential ($\mathcal{R}_r(t) = e^{-\lambda_r \cdot t}$) and Weibull ($\mathcal{R}_r(t) = e^{-\lambda_r \cdot t^{\beta_r}}$) reliability functions with λ being the component's failure rate. The uncertainty model $\mathcal{U}_r$ includes a set of uncertain parameters P_r, distributed within the bounds $[P_r^l, P_r^u]$. The sampler takes a sample from each parameter $p_r \in P_r$ and constructs a sample reliability function. For example, for an exponential distribution with bounds $[\lambda_r^l, \lambda_r^u] = [0.0095, 0.0099]$, a sample reliability function $\mathcal{R}_r^s(t) = e^{-0.0098\,t}$ can be generated.

The overall flow of the analysis approach is shown in Fig. 8 and includes the following steps: (a) The sampler samples a reliability function $\mathcal{R}_r^s(t)$ from the uncertainty distribution of each component r, (b) an analysis core uses these samples and calculates a sample reliability function for the given system implementation $\mathcal{R}^s_{\text{imp}}(t)$, and (c) a statistical *simulator* collects a set of sampled reliability functions $\Phi_{\text{imp}} = \bigcup_{s=1}^{n} \{\mathcal{R}^s_{\text{imp}}(t)\}$ and constructs the uncertainty distribution of the system reliability.

The analysis core can be realized by any existing technique that requires a reliability function of each component and calculates the system reliability function. In the concrete case, Fig. 8 shows a formal technique based on **BDD!**s that models the reliability of a system implementation with two components in series. The statistical simulator determines the number of required samples n to later obtain

desired statistics like mean and quantiles from Φ_{imp} with a guaranteed confidence level.[1] As an example, for each sample $\mathcal{R}^s_{\text{imp}}(t)$ in Φ_{imp}, the **MTTF!** can be calculated using the integration below:

$$\textbf{MTTF!}_s = \int_0^\infty t \cdot f^s(t)\,\mathrm{d}t \quad \text{where} \quad f^s(t) = -\frac{\mathrm{d}\mathcal{R}^s_{\text{imp}}(t)}{\mathrm{d}t}. \tag{2}$$

Using sample **MTTF!**s, design objectives such as the best-, worst-, and average-case **MTTF!** can be derived.

4.1.1 Uncertainty Correlation

To model any existing correlation between uncertain parameters of system components, we investigate whether they are exposed to common uncertainty sources, and are, thus, subject to correlative variations. Take temperature as an example: Components that are fabricated in the same package may be exposed to the same temperature, which means their reliability characteristics can be considered in a *correlation group*, whereas components in different packages might be considered independent. Assuming that the uncertainty sources and the correlation groups are given, we introduce models for obtaining correlated samples from the uncertainty distribution of component reliability functions in [29, 31]: To sample from an uncertain parameter p, we check if it is a member of any correlation group or not. If p is a member of G, we first generate a random probability g for the group G at the beginning of each implementation evaluation step and then calculate a sample from p using the inverse **CDF!** (**CDF!**) of the probability distribution of p at point g. Otherwise, a sample is taken independently from the distribution of p. Note that since the uncertain parameters in a correlation group might be differently distributed, returning the same quantile g from their distributions does not necessarily yield the same value, see Fig. 9. Thus, through sampling, the uncertain parameters in G vary together, and their variations are independent of those of the parameters outside G.

4.2 Uncertainty-Aware Multi-Objective Optimization

Finally, to enable the optimization of system implementations with multiple uncertain objectives, we propose an uncertainty-aware framework in [29]. It extends a state-of-the-art **DSE!** [43] and employs a **MOEA!** as the optimization core. These techniques introduce *dominance criteria* to compare different implementations and select which one to store in an *archive* and vary for the next iteration.

[1] Efficient sampling techniques [8] can be used to reduce the number of required samples.

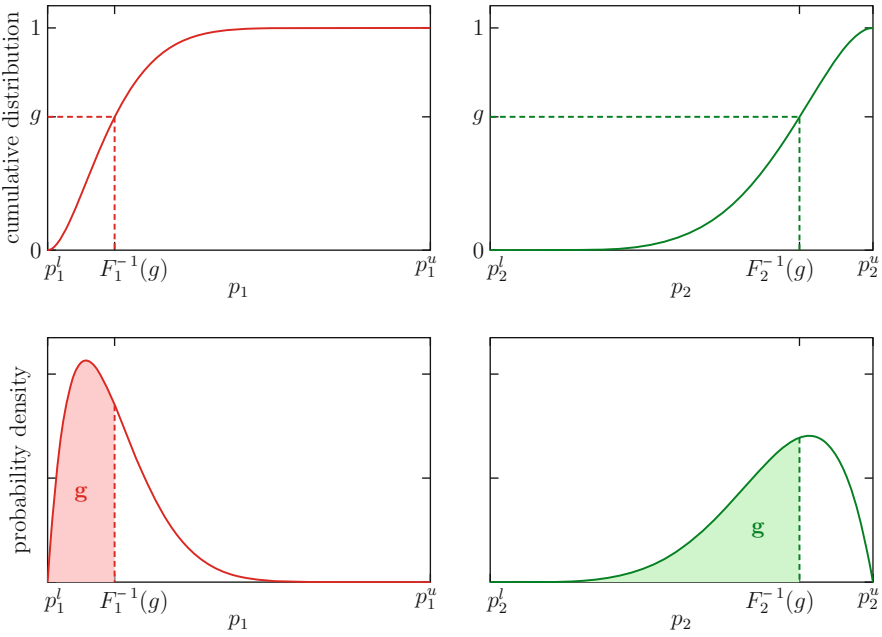

Fig. 9 Generating samples for correlated uncertain parameters p_1 and p_2

To maximize m objectives $O_1, \ldots, O_m$, each being a single value, the dominance of two implementations A and B is defined as follows:

$$A \succ B \iff \forall i \in [1, v] : O_A(i) \geq O_B(i) \ \wedge \ \exists j \in [1, v] : O_A(j) > O_B(j) \tag{3}$$

Here, $A \succ B$ means "A dominates B" and $O_A(j) > O_B(j)$ means "A is better than B in the j-th objective." Since this dominance criterion compares each of the m objectives independently, we refer to $O(i)$ as O for brevity.

The proposed uncertainty-aware optimization compares uncertain objectives using the following three-stage algorithm: (a) If the intervals of O, specified by the lower bound O^l and upper bound O^u, of two implementations A and B do not overlap, one is trivially better ($>$) than the other. (b) If the intervals overlap, we check if one objective is significantly better with respect to an *average criterion*, e. g., mean, mode, or median. (c) If the average criterion does not find a preference, a *spread criterion* compares objectives based on their deviation, e. g., standard deviation, variance, or quantile intervals, and judges whether one is considerably better. In case none of the three stages determines that one objective is better, the objectives are considered equal. The flow of this comparison operator is illustrated in Fig. 10.

To find if one uncertain objective has significantly better average O^{avg} or deviation O^{dev} compared to the other, we use two configurable threshold values

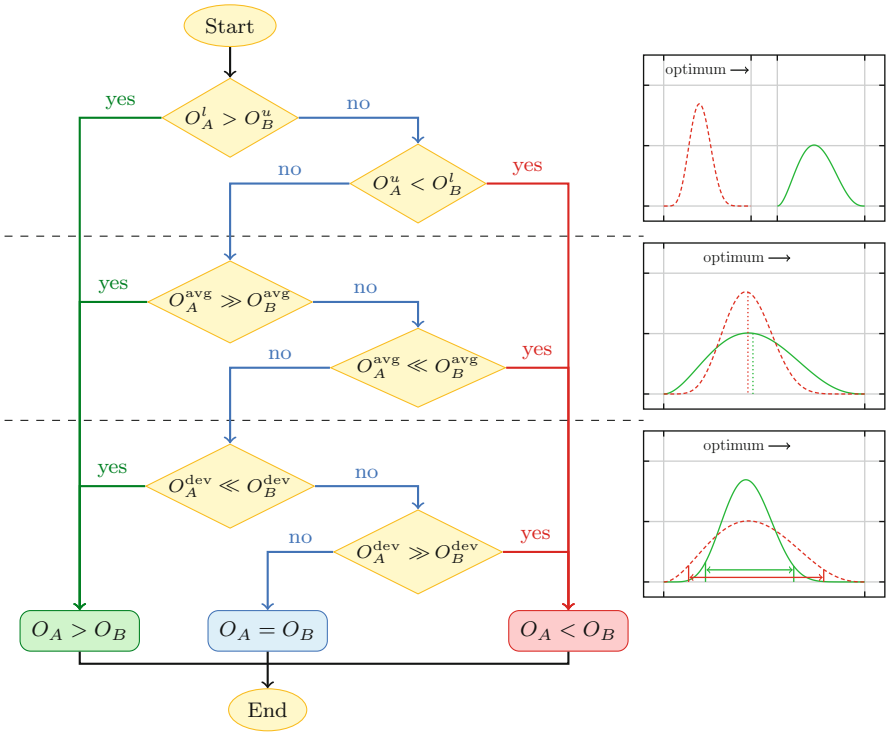

Fig. 10 The flow of the proposed three-stage comparison operator

ε^{avg} and ε^{dev}, respectively. For the average criterion, a configurable threshold value ε^{avg} determines if the difference of the considered average-case objective values is significant with respect to the given objective bounds. This enables to control the sensitivity of the second stage of the comparison:

$$\frac{O_A^{\text{avg}} - O_B^{\text{avg}}}{\max\left(O_A^u, O_B^u\right) - \min\left(O_A^l, O_B^l\right)} \geq \varepsilon^{\text{avg}}. \tag{4}$$

Here, $\varepsilon^{\text{avg}} = 0$ always prefers the objective with better average case, while $\varepsilon^{\text{avg}} = 1$ renders the average criterion ineffective since the left-hand side of Eq. (4) is always less than one. Thus, the scope of ε^{avg} must be carefully selected based on the objective's criticality to guarantee a required precision.

The spread criterion prefers the objective value with smaller deviation and uses a threshold value ε^{dev} to control the sensitivity of the comparison, i. e.,

$$\frac{O_B^{\text{dev}} - O_A^{\text{dev}}}{O_A^{\text{dev}} + O_B^{\text{dev}}} \geq \varepsilon^{\text{dev}} \Rightarrow O_A > O_B. \tag{5}$$

Given $\varepsilon^{\text{dev}} = 0$, any small difference between O_A^{dev} and O_B^{dev} is reckoned, which can lead to crowding in the solution archive. It causes solution A which is indeed weakly dominated by another solution B to be regarded as non-dominated because one of its uncertain objectives has a slightly better deviation than the corresponding objective of B. On the other hand, the spread criterion becomes ineffective if $\varepsilon^{\text{dev}} = 1$ and any significant difference between deviations of two uncertain objectives would be overlooked. Therefore, the value of ε^{dev} must be carefully selected.

Note that the statistics of an uncertain objective O required in the proposed comparison operator are calculated using samples from its distribution. Given a set of n samples for O, its variance can be calculated as follows:

$$O^{\sigma^2} = \frac{1}{n}\sum_{i=j}^{n}\left(O^j - O^{\mu}\right)^2 \quad \text{where} \quad O^{\mu} = \frac{1}{n}\sum_{i=j}^{n} O^j, \tag{6}$$

with O^{μ} denoting the mean of the distribution of O. Moreover, to find the q^{th} quantile of this distribution, we use the *inverse empirical distribution function* which traverses the samples in the ascending order and returns the very first sample after the $q\%$ smallest samples.

Figure 11 shows the resulting Pareto fronts for optimizing **MTTF!** and cost of an H.264 specification using the proposed comparison operator vs. a common uncertainty-oblivious approach that compares instances of uncertain objectives with respect to their mean values. The specification incorporates 15 resources, 66 tasks, and 275 mappings. The proposed operator uses mean and 95% quantile interval as the average and spread criteria, respectively. Depicted are the mean values,

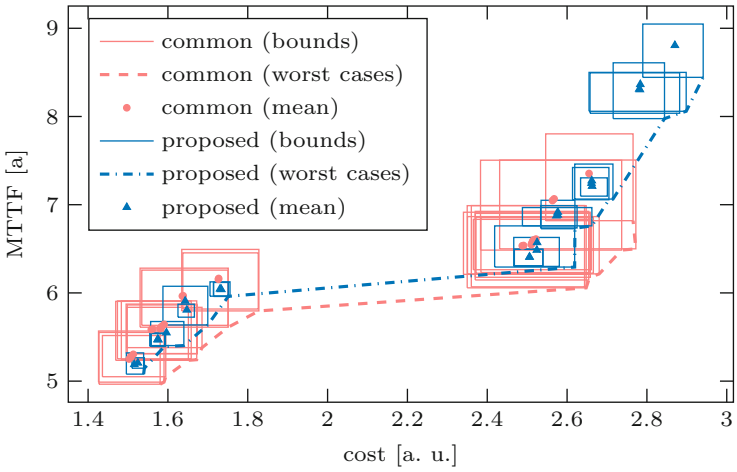

Fig. 11 Pareto fronts when optimizing **MTTF!** and cost of an H.264 encoder/decoder using a common uncertainty-oblivious approach vs. the proposed comparison operator

boxes enclosing the uncertainty distributions, and lines connecting the worst cases. The results show that the proposed comparison operator enables the **DSE!** to find implementation candidates of smaller uncertainty, and yet comparable quality in the average case.

5 Conclusion

Progressive shrinkage in electronic devices has brought them vulnerabilities to manufacturing tolerances as well as environmental and operational changes. The induced uncertainty in component reliability might propagate to system level, which necessitates uncertainty-aware cross-level reliability analysis. This chapter presents a cross-level reliability analysis methodology that enables handling the ever increasing analysis complexity of embedded systems under the impact of different uncertainties. It combines various reliability analysis techniques across different abstraction levels by introducing mechanisms for (a) the composition and decomposition of the system during analysis and (b) converting analysis data over abstraction levels through adapters. It also provides an explicit modeling of uncertainties and their correlations. The proposed methodology is incorporated in an automatic reliability analysis tool that enables the evaluation of reliability-increasing techniques within a **DSE!** framework. The **DSE!** employs meta-heuristic optimization algorithms and is capable of comparing system implementation candidates with objectives regarded as probability distributions.

Acknowledgments This work is supported in part by the German Research Foundation (DFG) as associated project CRAU (GL 819/1-2 & TE 163/16) of the priority program Dependable Embedded Systems (SPP 1500).

References

1. Al Faruque, M., Jahn, J., Ebi, T., Henkel, J.: Runtime thermal management using software agents for multi-and many-core architectures. IEEE Design Test Comput. **27**(6), 58–68 (2010)
2. Aliee, H.: Reliability analysis and optimization of embedded systems using stochastic logic and importance measures. Doctoral Thesis, Friedrich-Alexander-Universität Erlangen-Nürnberg (FAU) (2017)
3. Aliee, H., Glaß, M., Reimann, F., Teich, J.: Automatic success tree-based reliability analysis for the consideration of transient and permanent faults. In: Design, Automation & Test in Europe (DATE), pp. 1621–1626 (2013)
4. Aliee, H., Glaß, M., Khosravi, F., Teich, J.: An efficient technique for computing importance measures in automatic design of dependable embedded systems. In: Hardware/Software Codesign and System Synthesis (CODES+ISSS), pp. 34:1–34:10 (2014)
5. Aliee, H., Borgonovo, E., Glaß, M., Teich, J.: Importance measures in time-dependent reliability analysis and system design. In: European Safety and Reliability Conference (ESREL) (2015)

6. Aliee, H., Borgonovo, E., Glaß, M., Teich, J.: On the Boolean extension of the Birnbaum importance to non-coherent systems. Reliab. Eng. Syst. Safe. **160**, 191–200 (2016)
7. Aliee, H., Banaiyianmofrad, A., Glaß, M., Teich, J., Dutt, N.: Redundancy-aware design space exploration for memory reliability in many-cores. In: Methoden und Beschreibungssprachen zur Modellierung und Verifikation von Schaltungen und Systemen (MBMW) (2017)
8. Aliee, H., Khosravi, F., Teich, J.: Efficient treatment of uncertainty in system reliability analysis using importance measures. In: 2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN) (2019)
9. Bensalem, S., Bozga, M., Sifakis, J., Nguyen, T.: Compositional verification for component-based systems and application. In: Automated Technology for Verification and Analysis (ATVA), pp. 64–79 (2008)
10. Blickle, T., Teich, J., Thiele, L.: System-level synthesis using evolutionary algorithms. Des. Autom. Embed. Syst. **3**(1), 23–58 (1998)
11. Borgonovo, E., Aliee, H., Glaß, M., Teich, J.: A new time-independent reliability importance measure. Eur. J. Oper. Res. **254**(2), 427–442 (2016)
12. Bowen, J., Stavridou, V.: Safety-critical systems, formal methods and standards. Softw. Eng. J. **8**, 189–189 (1993)
13. Coit, D.W., Smith, A.E.: Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Trans. Reliab. **45**(1), 254–260 (1996)
14. Council, J.E.D.E.: Failure Mechanisms and Models for Semiconductor Devices. JEDEC Publication JEP122-B (2003)
15. Deb, K., Gupta, H.: Searching for robust Pareto-optimal solutions in multi-objective optimization. In: Evolutionary Multi-Criterion Optimization (EMO), pp. 150–164 (2005)
16. Eles, P., Izosimov, V., Pop, P., Peng, Z.: Synthesis of fault-tolerant embedded systems. In: Design, Automation & Test in Europe (DATE), pp. 1117–1122 (2008)
17. Farahani, B., Safari, S.: A cross-layer approach to online adaptive reliability prediction of transient faults. In: Defect and Fault Tolerance in VLSI and Nanotechnology Systems (DFTS), pp. 215–220 (2015)
18. Gebregiorgis, A., Kiamehr, S., Oboril, F., Bishnoi, R., Tahoori, M.B.: A cross-layer analysis of soft error, aging and process variation in near threshold computing. In: Design, Automation & Test in Europe (DATE), pp. 205–210 (2016)
19. Glaß, M., Lukasiewycz, M., Haubelt, C., Teich, J.: Towards scalable system-level reliability analysis. In: Design Automation Conference (DAC), pp. 234–239 (2010)
20. Glaß, M., Lukasiewycz, M., Reimann, F., Haubelt, C., Teich, J.: Symbolic system level reliability analysis. In: International Conference on Computer-Aided Design (ICCAD), pp. 185–189 (2010)
21. Glaß, M., Yu, H., Reimann, F., Teich, J.: Cross-level compositional reliability analysis for embedded systems. In: International Conference on Computer Safety, Reliability and Security (SAFECOMP), pp. 111–124 (2012)
22. Glaß, M., Teich, J., Lukasiewycz, M., Reimann, F.: Hybrid Optimization Techniques for System-Level Design Space Exploration, pp. 1–31. Springer, Dordrecht (2017)
23. Gu, Z., Zhu, C., Shang, L., Dick, R.: Application-specific MPSoC reliability optimization. IEEE Trans. Very Large Scale Integr. Syst. **16**(5), 603 (2008)
24. Herkersdorf, A., Aliee, H., Engel, M., Glaß, M., Gimmler-Dumont, C., Henkel, J., Kleeberger, V., Kochte, M., Kühn, J., Mueller-Gritschneider, D., Nassif, S., Rauchfuss, H., Rosenstiel, W., Schlichtmann, U., Shafique, M., Tahoori, M., Teich, J., Wehn, N., Weis, C., Wunderlich, H.: Resilience articulation point (RAP): cross-layer dependability modeling for nanometer system-on-chip resilience. Microelectr. Reliab. **54**(6–7), 1066–1074 (2014)
25. Hooman, J.: Specification and Compositional Verification of Real-Time Systems. Springer, Berlin (1991)
26. Jin, Y., Branke, J.: Evolutionary optimization in uncertain environments–a survey. IEEE Trans. Evol. Comput. **9**(3), 303–317 (2005)
27. Jung, S., Lee, J., Kim, J.: Variability-aware, discrete optimization for analog circuits. IEEE Trans. Comput. Aided Design Integr. Circuits Syst. **33**(8), 1117–1130 (2014)

28. Khosravi, F., Reimann, F., Glaß, M., Teich, J.: Multi-objective local-search optimization using reliability importance measuring. In: Design Automation Conference (DAC), pp. 1–6 (2014)
29. Khosravi, F., Müller, M., Glaß, M., Teich, J.: Uncertainty-aware reliability analysis and optimization. In: Design, Automation & Test in Europe (DATE), pp. 97–102 (2015)
30. Khosravi, F., Borst, M., Teich, J.: Probabilistic dominance in robust multi-objective optimization. In: IEEE Congress on Evolutionary Computation (CEC), pp. 1597–1604 (2018)
31. Khosravi, F., Müller, M., Glaß, M., Teich, J.: Simulation-based uncertainty correlation modeling in reliability analysis. The Institution of Mechanical Engineers, Part O J. Risk Reliab. **232**, 725–737 (2018)
32. Liang, S., Zhang, Z., Pei, T., Li, R., Li, Y., Peng, L.: Reliability tests and improvements for Sc-contacted n-type carbon nanotube transistors. Nano Res. **6**(7), 535–545 (2013)
33. Limbourg, P.: Multi-objective optimization of problems with epistemic uncertainty. In: Evolutionary Multi-Criterion Optimization (EMO), pp. 413–427 (2005)
34. Meyer, B., Hartman, A., Thomas, D.: Cost-effective slack allocation for lifetime improvement in NoC-based MPSoCs. In: Design, Automation & Test in Europe (DATE), pp. 1596–1601 (2010)
35. Misra, K.B.: Reliability Analysis and Prediction: A Methodology Oriented Treatment, vol. 15. Elsevier, Amsterdam (2012)
36. Narayanan, V., Xie, Y.: Reliability concerns in embedded system designs. Computer **39**, 118–120 (2006)
37. Pham, T.T., Defago, X., Huynh, Q.T.: Reliability prediction for component-based software systems: dealing with concurrent and propagating errors. Sci. Comput. Program. **97**, 426–457 (2015)
38. Rakshit, P., Konar, A., Das, S.: Noisy evolutionary optimization algorithms–a comprehensive survey. Swarm Evol. Comput. **33**, 18–45 (2017)
39. Salazar, A.D., Rocco, S.C.: Solving advanced multi-objective robust designs by means of multiple objective evolutionary algorithms (MOEA): a reliability application. Reliab. Eng. Syst. Safe. **92**(6), 697–706 (2007)
40. Sander, B., Schnerr, J., Bringmann, O.: ESL power analysis of embedded processors for temperature and reliability estimations. In: Hardware/Software Codesign and System Synthesis (CODES+ISSS), pp. 239–248 (2009)
41. Skadron, K., Stan, M.R., Huang, W., Velusamy, S., Sankaranarayanan, K., Tarjan, D.: Temperature-aware microarchitecture. In: 30th Annual International Symposium on Computer Architecture (ISCA), pp. 2–13 (2003)
42. Stathis, J.: Reliability limits for the gate insulator in CMOS technology. IBM J. Res. Dev. **46**(2–3), 265–286 (2002)
43. Streichert, T., Glaß, M., Haubelt, C., Teich, J.: Design space exploration of reliable networked embedded systems. J. Syst. Architect. **53**(10), 751–763 (2007)
44. Teich, J.: Pareto-front exploration with uncertain objectives. In: Evolutionary Multi-Criterion Optimization (EMO), pp. 314–328 (2001)
45. Wandeler, E., Thiele, L.: Real-Time Calculus (RTC) Toolbox (2006). http://www.mpa.ethz.ch/Rtctoolbox
46. Xiang, Y., Chantem, T., Dick, R.P., Hu, X.S., Shang, L.: System-level reliability modeling for MPSoCs. In: Hardware/Software Codesign and System Synthesis (CODES+ISSS), pp. 297–306 (2010)
47. Xie, Y., Li, L., Kandemir, M., Vijaykrishnan, N., Irwin, M.: Reliability-aware co-synthesis for embedded systems. J. VLSI Signal Proce. **49**(1), 87–99 (2007)