J Fail. Anal. and Preven. (2018) 18:707–720
https://doi.org/10.1007/s11668-018-0466-9

TECHNICAL ARTICLE—PEER-REVIEWED

An Overview of the Approaches for Automotive Safety Integrity Levels Allocation

Youcef Gheraibia · Sohag Kabir · Khaoula Djafri · Habiba Krimou

Submitted: 3 February 2018 / in revised form: 4 April 2018 / Published online: 11 April 2018
© The Author(s) 2018

Abstract ISO 26262, titled Road Vehicles–Functional Safety, is the new automotive functional safety standard for passenger vehicle industry. In order to accomplish the goal of designing and developing dependable automotive systems, ISO 26262 uses the concept of Automotive Safety Integrity Levels (ASILs), the adaptation of Safety Integrity Levels. ASILs are allocated to the components and subsystems that can cause system failure and malfunctions that lead to hazards. ASILs allocation is a hard problem consists of finding the optimal allocation of safety levels to the system architecture which must guarantee that the highest safety requirements are met while development cost of the automotive system is kept minimum. There were many successful attempts to solve this problem using different techniques. However, it is worth pointing out that there is an absence of a review that provides an in-depth study of all the existing methods and highlights their merits and demerits. This paper presents an overview of different approaches that were used to solve ASILs allocation problem. The review provides an overview of safety requirements including the related standards followed by a study of the resolution methods of the existing approaches. The study of each approach provides a detailed explanation of the used methodology and a discussion of its strength and weaknesses including the main open challenges.

Keywords ISO 26262 · Optimization · Exact solver · ASIL allocation · Automotive system · Safety requirement · Functional safety

Introduction

Due to the rapid advancement of technologies, an increasing number of electric and electronic devices are used every day in people's lives, in home, office and even in public spaces. Some of these devices can represent a danger or a threat to human health and environment by causing harm. The latter is a physical injury or damage to the health of persons [1]. Here is where the focus on safety becomes a primary necessity. IEC 61508 [2] is an international standard for the functional safety of electrical, electronic, and programmable safety-related systems (E/E/PE). Safety-related systems are systems that perform a function or a set of functions that ensure that risks are kept at an acceptable level [3]. IEC 61508's role is to minimize the likelihood of failure of these systems by ensuring that they provide the required safety integrity levels (SILs). SILs are presented as five safety levels, which are used to make sure that E/E/PE systems do meet the intended safety requirements against the highest risks.

It is noticed in the last two decades that the number of road accidents has increased significantly and as an outcome, a massive number of casualties have occurred. According to the global status report on road safety released by the world health organization, over 1.2 million people die each year on the world's roads and between 20 and 50 million suffer non-fatal injuries [4]. Reliability of automotive devices and systems is one of the major factors that have a direct effect on the safety of road users;

Y. Gheraibia · S. Kabir (✉)
School of Engineering and Computer Science, University of Hull, Hull HU6 7RX, UK
e-mail: s.kabir@hull.ac.uk

K. Djafri · H. Krimou
Department of Mathematics and Computer Science, University of Souk Ahras, 41000 Souk Ahras, Algeria

therefore, standards like ISO 26262 have come to light. ISO 26262 is an adaptation of the IEC 61508 standard for the automotive industry. This new automotive functional safety standard defines a safety life cycle applied to automotive electronics, where designs must comply with the standard by going through an overall safety process [5]. ISO 26262 uses the concept of Automotive Safety Integrity Levels (ASILs) which are an adaptation of SILs in the automotive industry.

ASILs are the key component of ISO 26262, used to represent the severity of safety requirements. They are five levels (QM, A, B, C, D) from the least strict ASIL (A) to the strictest ASIL (D), where QM means no safety requirements. Each level has a cost associated with it, which refers to the used cost function. According to ISO 26262's algebra, ASILs are assigned integer values as: ASIL(QM) = 0, ASIL(A) = 1, ASIL(B) = 2, ASIL(C) = 3, and ASIL(D) = 4. ASILs are allocated to hazardous components based on the severity of the hazard caused by the failure of that component. ASILs decomposition concept allows the ASIL to be decomposed over components that together provide the same hazard. ASILs allocation is a hard, complex problem of finding the most appropriate allocation of safety requirements to the components of the automotive system. An appropriate ASILs allocation to components and subsystems must guarantee the fulfillment of least-risk safety requirement with the least development cost.

Due to the criticality of the ASILs allocation problem and its crucial importance in any automotive system's safety, an appropriate allocation must be found. To attain this objective, exact solution techniques and optimization methods are used. Although these two methods aim to solve ASILs allocation problem, generally optimization solvers converge faster by finding at least a near-optimal solution; however, no guarantee is given that this solution will be found. On the other hand, exact solvers may take extra time, but they can find all exact optimal solutions.

In this review, after presenting an overview of the background of safety requirements including safety standards, an in-depth study of the existing approaches for ASILs allocation problem is presented. Here, different approaches are categorized into exact and optimization methods and described while identifying their strength and weaknesses. Finally, future outlook for ASILs allocation approaches is provided.

The rest of the paper is organized as follows: In the next section, the background study of safety requirements is introduced. In the third section, the ASILs allocation problem is described. "ASILs Allocation Algorithms" section reviews both exact and optimization approaches used for solving ASILs allocation problem. "Discussion and Future Outlook" section presents a discussion and summarizes the open challenges for the ASILs allocation approaches. Finally, in "Conclusion" section, concluding remarks are presented.

Safety Requirements: A Background Study

IEC 61508

IEC stands for the International Electrotechnical Commission, which is an international organization for standardization in the electrical and electronic fields and publishing international standards in addition to other activities. IEC 61508 is an international standard for the functional safety of electrical, electronic, and programmable electronic (E/E/PE) equipment [2]. It sets out the requirements to make sure that these systems provide a functional safety, which is defined as the absence of unreasonable risk due to hazards caused by the malfunctioning behavior of E/E/EP systems.

ISO 26262

ISO 26262 is the new automotive functional safety standard for passenger vehicle industry [6], titled Road Vehicles–Functional Safety applied to safety-related systems. It is considered as an adaptation of the IEC 61508 for the automotive industry. It applies to the vehicles which carry passengers or goods, with a number of wheels that equal or greater than 4.

ISO 26262 standard is divided into ten parts [1]:

- *Part 1—Vocabulary* the terms, definitions, and abbreviations.
- *Part 2—Management of Functional Safety* required management of functional safety in automotive industry.
- *Part 3—Concept Phase* the requirements for risk analysis and risk assessment.
- *Part 4—Product Development at the System Level* the requirements for product development at the system level.
- *Part 5—Product Development at the Hardware Level* the requirements for product development at the hardware level.
- *Part 6—Product Development at the Software Level* the requirements for product development at the software level.
- *Part 7—Production and Operation* the requirements for production, operation, and decommissioning services.
- *Part 8—Support Processes* the requirements for support processes.

- *Part 9—Automotive Safety Integrity Level (ASIL)-Oriented and Safety-Oriented Analysis* the requirements for ASIL-oriented analysis and defines the ASIL decomposition concept.
- *Part 10—Guideline on ISO 26262* overview of ISO 26262 and explanations of the other parts of the standard.

ISO 26262 represents the safety-critical systems-specific unifying standardization in the automotive industry to provide systems that must comply with the standard by going through an overall safety process. In order to develop safety-compliant systems, ISO 26262 imposes a functional safety cycle to verify the compliance of a product from its conception to decommission. Decommissioning is the phase to mark the end of the life cycle of a product. The end of life cycle can be determined from the life expectancy of the products that is built into their respective test durability cycles.

Unlike IEC 61508 that measures the reliability of safety functions and uses the maximum target probability, ISO 26262 is based on the violation of safety goals and provides requirements to achieve an acceptable level of risk. It uses the concept of ASILs, which are the key component for ISO 26262 to classify the strictness of safety requirement with respect to the software and systematic failure. ASILs are fully dedicated to the safety of the driver and road users and take no consideration of the system technologies.

In order to determine which ASIL to assign to a safety goal, the product goes through a safety process according to ISO 26262 to reduce the risk that can cause any harm or threat to the driver or the road users. The process starts by identifying hazards that could possibly be produced by the malfunctions of the system. After identifying hazards, the associated hazardous events are then identified. After selecting the failure modes of the system, an ASIL will be associated with the related safety requirement based on the risk posed by that failure mode. The safety goal is determined by a combination that gathers the probability of exposure, the controllability by the driver, and the severity of the failure. Once the risk is estimated, according to the safety goal, an ASIL to be assigned is determined.

Safety Integrity Levels

Safety Integrity Levels (SILs) are used to classify the strictness of safety requirement in safety-critical systems. SILs were originally described in UK Health Safety Executive guidelines [7]. They represent the levels of safety assigned to a system or a subsystem in order to avoid a systematic failure. SILs were adopted by many safety standards, including IEC 61508 which defines five levels of SILs from the least strict level SIL1 to the most strict SIL4 while SIL0 level means no special safety requirements.

SILs are assigned to system functions, subsequently to their components in order to avoid the systematic failure and the unacceptable risk according to a process defined in several standards. This process performs the hazard estimation and risk analysis, to eventually associate the appropriate SIL that reduces the hazard with the desired level. SILs play a dual role in the development and verification of systems [7] by allowing the top-down allocation of safety requirements to subsystems and components according to their criticality. It also allows the bottom-up verification in order to ensure that the safety requirements have been met. SILs are used to allocate safety requirements to hazardous components of critical systems to make sure that their design is composed of safety-critical functions that were assigned with an effective SIL without necessarily forcing all components of those functions to be associated with the maximum SIL.

SILs are defined as a manual and unaided process and can be determined based on qualitative and quantitative analysis. For example, Sallak et al. [8], Beugin et al. [9], Zhang et al. [10], and Lee et al. [11] considered determining SILs either qualitatively or quantitatively. SILs allocation process was first systematized and automated in [7]. This allows finding the feasible optimal SILs allocations where the systems architectures are complex because it delivers several safety-critical functions. This was achieved using the fault tree analysis [12, 13] through HiP-HOPS (Hierarchically Performed Hazard Origin and Propagation Studies) [14–16]. HiP-HOPS is a state-of-the-art model-based safety analysis (MBSA) [17–19] technique compliant with the algebra defined by the functional safety standard ISO 26262. The technique is supported by an automated safety and reliability analysis tool that provides an algorithm to automate the whole process in such complex architectures, by doing so, it may be possible to find the ASILs allocation in such systems.

ASILs Allocation Problem

Description

As previously mentioned, ASILs are classification levels used to describe the stringency of safety requirements to be met in a systems' development life cycle, while taking in consideration the systemic failure. ASILs are five levels: QM, A, B, C, and D. Each ASIL is assigned an integer value: ASIL(QM) = 0, ASIL(A) = 1, ASIL(B) = 2, ASIL(C) = 3, and ASIL(D) = 4. ASIL (D) imposes the highest safety requirement, ASIL (A) imposes the least strict safely requirements, and ASIL (QM) implies no special safety requirement. Each ASIL has a heuristic cost value defined within a scale which could be linear, logarithmic, or exponential.

The failure of one or more components in the system may cause a hazard. Severity, controllability, and exposure of each hazard are used as a base, on which the hazard is grouped into a given category. Each category is assigned an ASIL. For each assigned ASIL, a safety goal is imposed. The concept of cut sets is used when the combination of failure modes of different components can lead to a hazard. A cut set is a combination of events that can cause a system failure. In such a case, the ASIL allocated to each of the elements of a cut set should satisfy the ASIL allocated to the cut set itself. Assigning the same ASIL as that of the cut set to each element of a cut set would be prohibitively expensive, which further lead to increase in the development cost of the system considerably. In order to overcome this obstacle and make sure that safety requirements are met, the ASILs decomposition algebra proposed in ISO 26262 is taken into account. The algebra specifies that if the failure of a set of components caused a given hazard, then the ASIL assigned to the hazard is decomposed over the set of components. As an example, consider a system with C$_n$ components. The failure of two components C_1 and C_2 caused a hazard of ASIL(D); then, this ASIL can be decomposed over C_1 and C_2 as follows: ASIL(D) = ASIL(B) + ASIL(B), or ASIL(D) = ASIL(A) + ASIL(C).

As formerly stated, ASILs decomposition and allocation aim to ensure safety requirements fulfillment during the development life cycle of the system on the one hand, and reduce the development cost on the other hand. Thus, finding an appropriate ASILs allocation and decomposition is both critical and crucial tasks due to the huge size of solutions search space. For example, NYTT60 is a steer-by-wire system [20] having 8218 cut sets and 185 failure modes, which results in a search space size of 2.04*10e219. Finding an optimal solution to the ASILs allocation problem within such a huge search space is not possible in a reasonable time. For this reason, ASILs allocation gained its combinatorial nature and was classified as a hard problem.

ASILs Allocation Quality Measurement

ISO 26262 aims to produce and develop the most reliant systems that provide the highest level of safety. Requirements are allocated iteratively to elements, components, and subsystems. ASILs allocation falls under the influence of the different requirements severities and cost development. Therefore, for these reasons, ASIL allocation is not just an optimization problem of simply assigning levels to components. It is in fact a complex, critical, optimization problem of finding the most appropriate ASILs allocation that minimizes the cost while meeting the given safety requirements. This allocation represents the optimal solution in the search space.

A feasible solution for ASILs allocation must fulfill the safety requirements, meaning that it does not violate any ASIL allocated to the safety goal of the corresponding hazard. On the other hand, unfeasible solutions have at least one failure mode that violates the given requirements. The feasibility of each candidate solution is tested by using the fault propagation information which is described as minimal cut set (MCS) of the fault tree analysis (FTA). MCS is extracted from fault trees and is defined as a combination of system components' failures that may lead jointly to a hazard [21, 22]. The FTA identifies the functions (components) over which an ASIL is decomposed and allocated. These fault trees describe how the failure of a system component can cause a hazard, which is assigned an ASIL as well as a safety goal that is associated with it. Allocated ASILs are implemented by independent components where the failure of these components altogether causes a safety goal violation. The top events in these fault trees are the safety goals violations identified by the preliminary hazard analysis.

A MCS may contain a single failure mode that can cause the failure of the system directly or a combination of different failure modes that must fail together to cause the failure of the functionality of the system. As it is defined in the ISO 26262 algebra, the sum of ASILs allocated to each event of a cut set is either equal or greater than the ASIL allocated to the cut set. In order to determine the fitness of each candidate solution, a cost function must be used, and there is a group of cost heuristics which are presented in Table 1.

The fitness of each candidate solution is simply calculated by summing up the cost of all ASILs assigned to the failure modes. For example, if we have an automotive system S containing five components {C1, C2, C3, C4, C5} that might cause a hazard, a feasible candidate solution can possibly be: {1, 3, 4, 2, 3} relating directly to {ASIL (A), ASIL(C), ASIL(D), ASIL(B), ASIL (C)}. If we use a linear cost function, the fitness of this particular solution would be 130 because 10 + 30 + 40 + 20 + 30 = 130.

Since ASILs allocation is a hard and complex problem with an enormous number of feasible solutions, it has been

Table 1 ASIL cost heuristics

Cost heuristics	QM	A	B	C	D
Linear	0	10	20	30	40
Logarithmic	0	10	100	1000	10000
Experiential-I	0	10	20	40	50
Experiential-II	0	5	30	35	50

at the core of interest of researchers. Over the years, several techniques considering diverse methods have been developed to find an economic ASILs allocation. These techniques are divided into two main categories: exact techniques and optimization techniques.

ASILs Allocation Algorithms

Exact Approaches

Exact solvers aim to find the exact solution to a given problem. There were some attempts to solve the ASILs allocation problem using these solvers to find all exact solutions. This section introduced the exact solvers that were proposed and developed to solve ASILs allocation.

Exact Approach Using Three State-of-the-Art Solvers

Murashkin et el. [23] used three off-the-shelve solvers such as CSP (Constraint Satisfaction Problem) [24], SMT (Satisfiability Modulo Theories) [25], and ILP (Integer Linear Programming) [26] to find the exact solution to the ASILs allocation problem. More specifically, they used Choco CSP [27], Z3 SMT [28], and CPLEX ILP [29] solvers.

Choco is a problem modeler for the constraint satisfaction problem (CSP) and constraint programming (CP) [30]. It is built and distributed under the BSD license (Berkeley Software Distribution License). It is an efficient and a readable constraint system for research, development, and describing hard combinatorial problems in the form of CSP and solving them. After modeling the problem by defining the set of constraints that need to be met, Choco solves it by using programming techniques. Z3 2.0. SMT is an SAT (SATisfiability) standard of first-order formulas with respect to other background theories such as lists, arrays, bit vectors, and equality reasoning [25]. This solver is a theorem prover developed by Microsoft Research implemented in C++. Z3 integrates a modern DPLL-based SAT solver, a core theory solver that handles equalities and uninterpreted functions, satellite solvers, and an E-matching abstract machine [31]. In the past, SMT solvers have been successfully applied to the multi-objective optimization of software product lines [32, 33]. The third exact solver is CPLEX ILP solver from IBM ILOG CPLEX Optimization Studio 12.3, 64-bit edition [29].

Figure 1 shows the general workflow of finding exact solutions to the ASILs allocation problem using each of these solvers. As seen in the figure, the first step is to encode the ASILs allocation problem for each of the solvers so that the specification is compatible with the reasoning mechanism of respective solver. This encoding includes the representation of the key entities in the ASILs allocation problem such as *hazard*, *component failure mode, minimal cut set,* and readjustment of the cost functions shown in Table 1 for the simplification of the solution generation process. The detailed descriptions of how the encoding was done for all the solvers are out of scope of this paper. Interested readers are referred to [23].

Clafer 0.3.6 [34] was used to specify the ASILs allocation problem for the Choco solver. Clafer is a lightweight modeling language that aims at improving the understanding of the problem domain in the early stages of software development determining the requirements with fewer defects. Clafer's optimization backend, based on Choco CSP Solver 3.2 [27], was used to generate all the optimal solutions. For Z3 SMT solver, the ASILs allocation problem was directly encoded in a Python file that references Python Z3 library. For CPLEX ILP solver, the problem was encoded using a natural, algebraic linear programming formulation, CPLEX LP file format (.lp files).

The effectiveness and performance of this exact approach using three different solvers were evaluated by applying them on three variants of real-world Hybrid Braking System Models (HBSMs) [35, 36]: HBSM 1, HBSM 2, and HBSM 3. HBSM 1 has 31 MCSs and 24 failure modes; HBSM 2 has 94 MCSs and 60 failure modes; and HBSM 3 has 11573 MCSs and 60 failure modes. According to the workflow of Fig. 1, the encoding of the ASILs allocation problem for each of the case study and for each of the four cost functions was automated, meaning that a total of 12 experimentation settings have

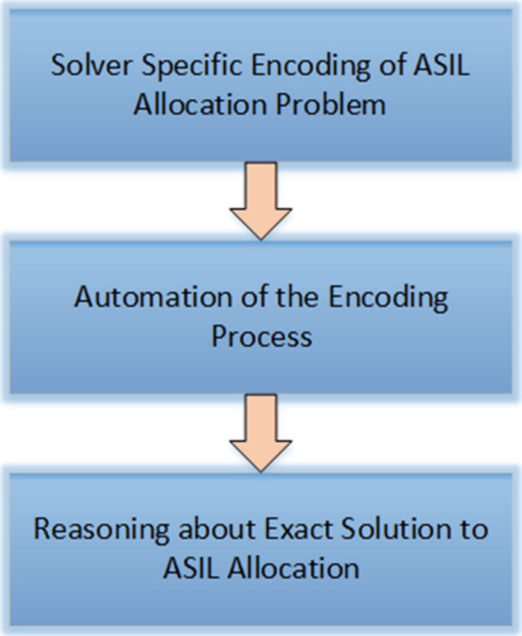
Solver Specific Encoding of ASIL Allocation Problem
Automation of the Encoding Process
Reasoning about Exact Solution to ASIL Allocation

Fig. 1 Workflow for ASILs allocation using exact solvers

been produced for each case study. Murashkin *et el.* [23] performed all the experiments on an Ubuntu 12.04LTS 64-bit machine with AMD Lisbon CPU 2.8 GHz processor and 32 GB RAM. Based on their experiment, for HBSM 1 (smallest case study), all three solvers produced solutions reasonably quickly. On the other hand, for the HBSM 2, the Choco CSP did not perform well. More specifically, it was not able to produce solutions in reasonable time for any cost function except for the linear cost function. In the contrary, Z3 SMT and CPLEX ILP solvers did actually provide all optimal solutions for most instances in a quite efficient time, but it was not clear which one outperformed the other. However, for the HBSM 3, with the linear cost function, Z3 SMT solver took 18.9 h while CPLEX ILP took 13.3 days to complete.

System of Linear Equations

Dhouibi et al. [37] proposed a new approach to decompose and allocate ASILs using a system of linear equations. This latter is an ensemble or a collection of two or more linear equations that has the same variables and has to be solved all at once. A linear equation is defined as any equation that has the standard form $ax = b$, where a and b are real numbers while x is the variable. It may have one or more variables and is exponent free [38]. There are several methods to solve a system of linear equations including graphing method, substitution method, elimination method, and Gaussian method [39]. The solution to a system of linear equations implies assigning values to the variables in a way that ensures that all equations are satisfied. In other words, solving a system of linear equations means finding the intersection points of the equations. It may admit one solution, an infinite set of solutions, or zero solution depending on the equations.

The proposed approach intends to interpret ASILs decomposition and allocation problem as a system of linear equations, where each equation is formed in the following form.

$$\sum a_i \times \mathrm{ASIL}_{Fi} = \mathrm{ASIL}_{\mathrm{SR}} \qquad \text{(Eq 1)}$$

In the above equation, it is considered that ASIL allocated to a function Fi (i.e., ASIL_{Fi}) in the architecture refers to the ASIL allocated to the safety requirement SR (i.e., $\mathrm{ASIL}_{\mathrm{SR}}$) provided by this function. The value of the coefficient a_i can be either 0 or 1, defining whether an MCS contributes to the function loss. In other words, a_i is 1 if the reason of function loss is in the MCS and a_i is 0 otherwise.

By applying this to all MCSs, a system of linear equations is created in the form of a matrix as seen below:

$$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ 1_{m1} & \cdots & a_{mn} \end{bmatrix} \times \begin{bmatrix} \mathrm{ASIL}_{F1} \\ \vdots \\ \mathrm{ASIL}_{Fn} \end{bmatrix} = \begin{bmatrix} \mathrm{ASIL}_{\mathrm{SR}1} \\ \vdots \\ \mathrm{ASIL}_{\mathrm{SR}n} \end{bmatrix} \qquad \text{(Eq 2)}$$

Different possible ASIL allocations can be obtained by evolving this system of equations for ASIL_{F1} to ASIL_{Fn}. In some cases, the safety engineer may have some safety preferences to apply, such as assigning a specific ASIL to a certain component or manipulating the dependency of system components. To do so, new constraints can be added to the system to incorporate these preferences by modifying the corresponding equations. In order to assign a certain component a specific ASIL, its corresponding column is removed from the system of equations. The non-dependency of two variables is handled by using the logic operator "OR" to blend their corresponding operator in the system. As previously stated, there are numerous ways to solve a system of linear equations. To do so, the augmented matrix of the system is taken into consideration. An augmented matrix of a system of linear equations is a matrix where its elements represent the coefficients of the variables in the system [40]. Since the augmented matrix resulting from the system is not always square, many of the solution methods become inapplicable. In [37], Dhouibi et al. used the Row Reduced Echelon Form (RREF) to find solutions to the system of equations. RREF of a matrix is generally obtained by Gauss-Jordan elimination where the rows are the target of the elimination. After performing a sequence of operations, the result will be a system in the row echelon form, which is a matrix with all zero rows being at the bottom of the matrix [41]. The first number of the nonzero row is in a column to the right of the first number of the row above it, and all leading entries in the row below are zeros. This helps to identify the pivot and free variables that correspond to the columns with no leading entry. After this, to find solution(s) to the ASILs allocation problem, these variables are assigned with values within ASILs numerical values range {0, 1, 2, 3, 4}. The values of the rest of the variables are then obtained using back-substitution. The echelon form of the system of equations reveals whether the system may admit a possible ASILs allocation or not. Solvability is determined from the system of equations; if it contains an equation in the form of 0 = Cst [37], then the system is not solvable. In such cases, the safety engineer can either review the system or just ignore the preferences he made earlier. The approach proposed by Dhouibi et al. [37] was used in [42] to assist with the system design and architecture optimization by considering safety and cost as constraints.

Optimization Approaches

Exact solvers can produce an exact solution to a given problem, but they may take too long to find a solution. In order to improve the performance of the exact solvers in terms of execution time, optimization approaches are proposed. Optimization solvers are algorithms which produce a near-optimal solution, it may not be the best, but it represents a feasible solution generated within a small amount of time. The following sections present a set of optimization algorithms used to solve ASILs allocation problem.

Integer Linear Programming Approach

The approach presented in [43] proposes a method that allows an automatic allocation of ASILs to the system components by considering the ASILs allocation problem as an integer linear programming problem. First of all, an integer linear programming optimization problem is the type of problems that aims to either maximize or minimize an objective function that is subject to given constraints. All (or some) variables are tied up to integer values, and both objective function and constraints are within a linear scale [44]. The solution to this problem grants an optimal ASILs allocation by optimizing an objective function that takes into consideration a set of constraints. Constraints used in the proposed approach are identified in the preliminary hazard analysis, extracted from fault tree analysis results, and defined by the safety engineer as safety preferences.

This approach uses a modeling framework to create an EAST-ADL (Electronic Architecture and Software Technology-Architecture Description language) [45] model, which is a UML-based domain-specific language. EAST-ADL is applied in the automotive software-based systems domain. It aims to build a model to illustrate the automotive electronic system using meta-models and natural language [46]. This model is used to describe the system components, the relations connecting them as well as their reliance, which subsequently leads to model their failures. Moreover, it allows modeling safety goals and assigning ASILs to system components. The automatic generation of fault trees from the EAST-ADL model is supported by the previously mentioned framework. MCSs of the fault trees are then used later as inputs by the framework's solver dedicated to solve the ILP problems.

A constraints solver designed to solve ILP problems is provided by the framework to find an optimal ASILs allocation. The idea is to extract MCSs from fault tree analysis results, preliminary hazard analysis results and take into consideration the engineer's preferences. This combination was used to build constraints that represent an input to the solver. The solver considers two main types of constraints on each system component's ASIL. The first constraint is related to the minimal cut sets. For each minimal cut set $j(j = 1\ldots n)$ where j comprises of $f_i(i = 1\ldots m)$, a failure of a system component C_k that may cause a safety goal violation, a constraint is defined. Each constraint has a coefficient C_{xy} for each component, which is set to 1 if f_i belongs to the MCS$_j$, otherwise its value is set to 0. In case where the conjunction of components failure may lead to a safety goal violation, their coefficients are set to 1 and the rest of the coefficients are set to 0 in the constraint. Finally, if a single component causes a safety goal violation, its coefficient is set to 1, while the rest of the coefficients in the corresponding constraint are set to 0. The second type of constraint is related to the safety engineer's preferences. These preferences aim to reduce the cost of the allocated ASILs. From this perspective, a preferred ASIL is set to selected system components. For each component C_k, if a preferred ASIL is defined to C_k, a coefficient is defined and set to 1.

The constraint solver is then used to solve the ILP problem to find the optimal allocation of ASILs to system components. In this approach, the authors created a plug-in for an open-source tool called Papyrus [47]. This plug-in embeds a constraint solver with the Papyrus tool to support the fault tree generation and automatic ASILs allocation. Once the solver is executed, if a solution is found, it is reported to the user. The solution is then examined by the engineer who will choose the action to be taken according to the quality of the solution. If it is adequate, the ASILs allocation to the system components is initiated and the EAST-ADL model is automatically changed in accordance with the new solution, else, the safety engineer will have to make some changes to his/her preferences and rerun the solver.

Penalty-Based Genetic Algorithm

Originally developed by John Holland in 1975, genetic algorithm (GA) is a search metaheuristic that mimics the process of natural evolution. It uses the concepts of natural selection and genetic inheritance [48]. The genetic algorithm is applied to unconstrained combinatorial optimization problems; therefore, the need for a method to get a handle of the constraints has appeared. There are several methods that can be used with GA to ensure the non-violation of the constraints.

In [5], Parker et al. used a penalty-based genetic algorithm [49] to solve ASILs allocation problem. The flowchart of the approach is shown in Fig. 2. This approach uses a fixed-length encoding where each "case" can hold a value from 0 to 4 (QM to D, respectively). The fitness function defined in this approach is equal to the sum of the

ASILs costs of each candidate solution. Different scaling is used to define the cost of each ASIL, as seen in Table 1.

In the first step, initial solutions for ASILs allocation problem are generated randomly. In the second step, genetic operators are applied to create new candidate solutions by manipulating the existing ones which were selected in the previous phase. Genetic operators are: recombination and mutation. In the selection process, the algorithm selects solutions from the population to perform genetic operations. The idea is to apply "the survival of the fittest" principle. First of all, candidate solutions are classified according to their penalized fitness. Selection is done based on how close the candidate is to a randomly chosen number between 1 and the square root of population size.

Mutation stochastically disorders the population by changing the value of each "case" of the selected candidate solution to a random number between 0 and 4 (ASIL levels) according to an antecedently specified mutation probability. Recombination creates new candidate solutions by imitating biological reproduction. The recombination operator used in [5] is the uniform crossover, which consists of creating offspring by randomly selecting an encoding "case" from one of the parents encoding.

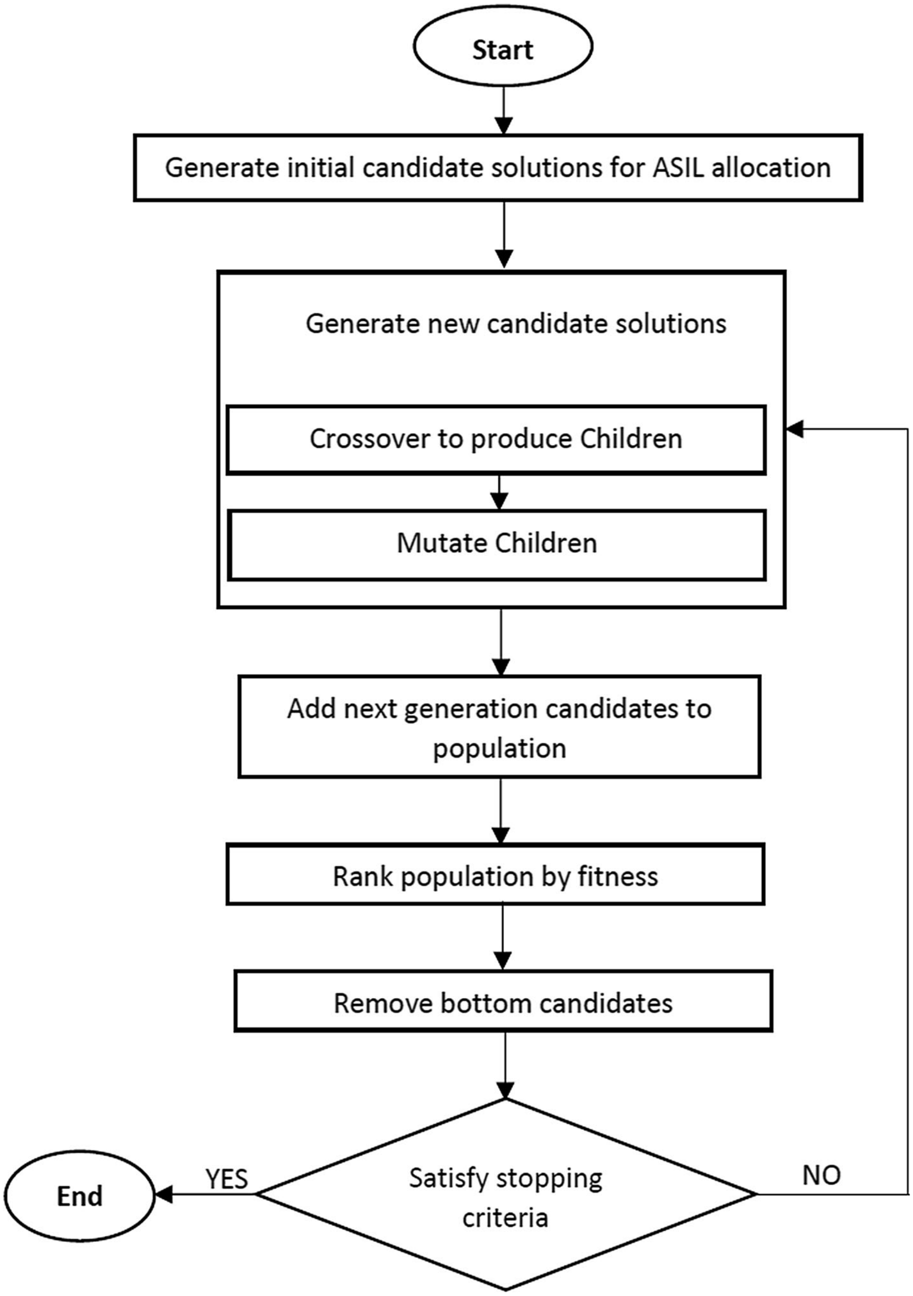
Start
Generate initial candidate solutions for ASIL allocation
Generate new candidate solutions
Crossover to produce Children
Mutate Children
Add next generation candidates to population
Rank population by fitness
Remove bottom candidates
Satisfy stopping criteria
YES
NO
End

Fig. 2 Flowchart of penalty-based GA for ASILs allocation problem

In the next step, the newly generated solutions are added to the existing population. Each of the solutions within this population is then ranked by their fitness. During the ranking process, candidate solutions with lower fitness value are favored, unless they are considered as infeasible solutions. The penalty method used in [5] is applied to increase the penalty on infeasible candidate solutions, i.e., reduces the fitness of candidate solutions that violate the constraints as the search progresses. A set of minimal cut sets (MCSs) of the fault tree analysis using HIP-HOPS is used to check for candidate solutions feasibility. The feasibility of a solution is checked by comparing the total value of the ASILs allocated to different MCSs with the ASIL required for the hazard caused by these MCSs. The violation of a requirement is represented by an "invalid_ASILs" counter that increases when sum of ASILs assigned to different failure modes differs from the one required to be met by the MCS containing these failure modes. The "invalid_ASILs" counter values are measured and added as a penalty to the ASIL cost of the candidate solutions. A feasible solution will have an "invalid_ASIL" equal to zero and thus will receive no penalty. On the other hand, an infeasible solution will have a nonzero count for "invalid_ASILs," and the magnitude of the value will depend on the number of safety requirements violated by that particular solution. The higher the value of "invalid_ASILs," the greater the penalty received by the infeasible solution. In this way, if two solutions have equal cost, the solution with higher penalty will deem as less fit solution. However, a dynamic threshold is defined to allow near-feasible solution a certain degree of constraint violations in the early search generations. As the search progresses, the generated solutions are designed to face harsher penalties by varying the threshold value. When all the solutions are ranked by their fitness, the less fit solutions are removed from the bottom, unless the population size reaches the initial size. The whole process continues until the stopping criteria are reached.

The effectiveness of this approach has been evaluated by applying it to three different versions of the hybrid braking systems mentioned in "Exact Approach Using Three State-of-the-Art Solvers" section. The computational performance was evaluated on a machine with Intel core i5 3.40 GHz processor with 8 GB RAM. For different cost functions, the average run time to find solutions for each of the systems is shown in Table 2.

Table 2 Average run time in seconds to find optimal solution using penalty-based GA approach

Cost heuristics	HBSM 1	HBSM 2	HBSM 3
Linear	1.04	4.67	15.51
Experiential-I	1.21	7.32	14.63
Experiential-II	1.07	2.47	3.31
Logarithmic	1.98	2.35	3.04

Tabu Search Algorithm

Developed by Fred Glover in 1986, Tabu search (TS) is an optimization metaheuristic that uses local search methods to solve combinatorial problems [50]. TS starts from a feasible solution and creates a neighborhood of the current solution. Neighborhood solutions are evaluated, and only one candidate solution is chosen on the basis of its fitness to replace the current solution until a local optimum is reached. In this case, a worse neighbor can be selected to replace the current solution even if it does not improve the current one. Tabu search has a so-called Tabu list, which represents a short-term memory to store previously visited solutions; the list is updated whenever a new solution is found. This way search would not return to recent moves. The search stops when a termination criterion is met.

Tabu search was used in [6] for automatic decomposition of safety integrity levels based on the work presented in [51] by employing the Steepest Descent Mildest Ascent (SDMA) method. The original version of the method follows Steepest Ascent Mildest Decent (SAMD) [52], i.e., the steepest ascent until a local maximum is reached and then it uses the mildest descent to escape from it. Since the original method is meant for maximization problems, SDMA is developed as an adaptation to the SAMD to minimize ASILs cost.

The algorithm starts with a number of iterations p from a feasible solution. Using the steepest descent decrement, the ASIL of the failure mode that results in the highest decrease of the cost of the system decremented iteratively. This process continues until it leads to the ASIL decomposition rules violation. This situation implies that a minimum has been reached and the algorithm uses the mildest ascent to avoid being trapped in the local minima. A variable f_i is set to be equal to p and decrements when every iteration is complete. f_i prohibits the search from deceasing the ASIL of a failure mode i after an ascent move is performed. Another variable $f_i^{'}$ is set to prohibit the increase of ASIL of failure mode i after a descent move is performed. The prohibition is limited within a number of iterations $p^{'}$, but unlike p, $p^{'}$ does not decrement after every iteration, and it only decrements in case where an additional ASIL decrease has been done in a specific iteration. It is worth pointing out that the value of p differs from $p^{'}$ and both of their values dynamically change as the search evolves. The value of p is altered with a single increment after three iterations are done while the value of $p^{'}$ is altered with a single increment after four iterations are done. Both of their counts are reset when each reaches its

Table 3 Average run time in seconds to find optimal solution using Tabu search-based approach

Cost heuristics	HBSM 1	HBSM 2	HBSM 3
Linear	0.05	3.65	4.44
Experiential-I	0.74	4.20	14.70
Experiential-II	0.84	0.79	5.72
Logarithmic	0.31	1.82	1.91

limits. In order to find a better solution, the approach presented in [6] declines move restriction to ensure better diversification of the search. The search ends if a number of repetitions are achieved without improving the solution.

In terms of time required to find an optimal ASILs allocation, Tabu search-based approach made improvements over the penalty-based GA approach. Table 3 shows the average run time of the Tabu search-based approach to finding optimal ASILs allocation for the three variants of the hybrid braking system.

Penguin Search Optimization Algorithm

Proposed by Gheraibia and Moussaoui [53], Penguin Search Optimization Algorithm (PeSOA) is a new algorithm in the nature-inspired meta-heuristics. PeSOA imitates flightless birds hunting mechanisms by guiding the search strategies using several rules. Those rules resume the foraging behavior of penguins. A summary of the rules is presented as follows.

The optimization process starts by generating a population P of penguins. The population is then divided into several groups, where each group contains n number of penguins. The value of n varies depending on the food availability in a specific area, more food implies more penguins. Regarding their oxygen reserve, penguins stochastically explore the sea searching for food. They dive to different levels and each group hunts in a particular depth depending on the energy gain. Communication among sea birds is presented in two forms: intra-group and inter-group. As the oxygen reserve depletes penguins surface up and share information about food location and quantity with the rest of the group. This is the intra-group communication. The food available in certain areas may not be sufficient to feed the whole group; in such case, some members of the group or the whole group moves to another group through inter-group communication.

The PeSOA was used by Gheraibia et al. [54] to find an optimal ASILs allocation. The algorithm uses the concept of equivalent classes to prioritize a set of failure modes. Equivalent classes are the class of failure modes that produce the same integrity, but may have different costs. Elements of lower cost are assigned a higher priority while elements of higher costs are assigned a lower priority. The classes are then constructed by building all possible combinations of each ASIL (from ASIL 1 to ASIL 4). Next, the cost heuristic function is used to calculate the cost of each combination, followed by priority generation which consists of sorting the cost of elements of the classes ascendingly. After that, as previously mentioned, grant low-cost elements a high priority and high-cost elements low priority. Priorities guide the penguins search while improving the current solution by replacing the low-priority ASILs with others with higher priority. This strategy may violate the system safety requirements and lead to turn the current solution into infeasible one, but the algorithm got a handle of the situation by incrementing the ASIL 0 components to ensure solution feasibility.

Since priority classes help to reduce the solution space by considering equivalent classes, it leads to a better estimation of the population size. A population of P penguins that comprehends a set of feasible solutions is generated and divided into several groups; this technique guarantees that the search would not be trapped in the local optima. Population partitioning is done on the basis of how frequently a failure mode appears in the minimal cut sets. To do so, minimal cut sets of the system are parsed. After that, failure mode frequencies are calculated and ordered in a way that ensures a good distribution of the population by assigning each group of penguins a set of failure mode ASILs to handle during the search progress. This will further lead to an optimal ASILs allocation. Each penguin generates a neighborhood of solutions and chooses the best one among them subject to the fitness function, i.e., if the fitness of the new solution is higher than the current one. The penguin is allowed to move to another position in the upcoming iteration, and the oxygen reserve is updated which grants the penguin further extendibility of the search. The size of the neighborhood is related to the oxygen reserve, which differs from a penguin to another. Higher oxygen reserve implies larger neighborhood of solutions. Contrariwise, if the solution is not improved, then the oxygen reserve decrements; consequently, the solutions neighborhood size is reset to 1. Penguin's population is redistributed according to the improvement each penguin has achieved. The improvement is represented by the difference between the cost of the current solution and the new one. The termination criterion is reached when the total cost stops decreasing.

By using the concept of equivalent class and multi-group communication, the PeSOA-based approach finds solutions quickly compared to the GA and the Tabu search-based approaches. PeSOA-based approach was applied to the three variants of hybrid braking systems, and the results presented in Table 4 show the improvements over other approaches.

Table 4 Average run time in seconds to find optimal solution using PeSOA-based approach

Cost heuristics	HBSM 1	HBSM 2	HBSM 3
Linear	0.08	0.11	0.27
Experiential-I	0.09	0.38	0.59
Experiential-II	0.10	0.41	0.89
Logarithmic	0.12	0.21	0.32

Table 5 Average run time in seconds to find solution using ACO-based approach

Cost heuristics	HBSM 1	HBSM 2	HBSM 3
Linear	0.003	0.01	0.16
Experiential-I	0.007	0.02	0.35
Experiential-II	0.009	0.03	0.62
Logarithmic	0.008	0.02	0.20

Ant Colony Algorithm

Recently, a methodology has been proposed by Gheraibia et al. [55] for ASILs allocation problem. This method extends the work presented in [56] and uses the Ant Colony Optimization (ACO) [57, 58] to allocate safety requirements. The ant colony algorithm is a well-known swarm intelligence algorithm. It imitates the collaborative behavior of an ant colony. In ACO, each ant acts separately by investigating the area and after each tour, each ant updates pheromone to inform its affiliates for the quality of its previous tour.

ACO has been used to solve the ASILs allocation problem by incorporating a couple of strategies to reduce the research space before starting the optimization process. These strategies are based on both internal and external information on the cut sets and the cost functions. The first strategy is based on the cost function values. After analyzing the different combination of failure modes ASILs, it was found that according to the cost function, it is possible to discard some combinations, and by doing so, the search space is reduced considerably. The second search space reduction strategy is based on the cut sets. This approach aimed to reduce the search space by reducing the number of possible ASILs for each failure mode (i.e., reduce the range of each failure mode ASIL from five possibilities to lower value).

The ant colony algorithm uses these strategies to achieve the optimal ASILs allocation at a reasonable time. The ASILs allocation problem has been formulated as a graph where the nodes represent the failure modes and the edges represent the ASIL allocated to the destination node of these edges. This new representation of the ASILs allocation problem helps the ants to explore the search space efficiently. Each ant explores the graph, and at the end, it updates the pheromone value to communicate to other ants the quality of the explored path. In this way, ants can make informed decisions while exploring the search space, which in turn results into improved solution. Ant colony algorithm for ASILs allocation has been evaluated using two different case studies: the hybrid braking system and the steer-by-wire system. Table 5 shows the results for the three variants of the HBSM. As seen from this table, in terms of execution time, the ACO-based approach outperformed all the existing approaches.

Discussion and Future Outlook

ASILs allocation is one of the important tasks that is needed to be performed in the automotive industry to comply with ISO26262 standard in order to satisfy the safety requirements of the system being developed while reducing the development cost. Due to the nature of the problem and the ASILs decomposition rules defined in the ISO26262 standard, the ASILs allocation problem is categorized to have a combinatorial nature. Over the years, researchers have devoted their efforts to find an efficient solution to the ASILs allocation problem. As an outcome, various techniques have been developed to solve this problem and these techniques tend to gravitate toward two different paradigms—exact and optimization approaches.

The approaches used for ASILs allocation have their own advantages and disadvantages. In general, if there exists a solution to a particular ASILs allocation problem, the exact approaches guarantee to find the solution. Although these approaches guarantee to find a solution, they may take an exceptionally long time to find it. For this reason, these approaches can be used for ASILs allocation of smaller systems, but if the system is complex and large, application of the exact approaches will not be feasible. Introduction of optimization approaches has improved the execution performance of the ASILs allocation process greatly. Unlike the exact approaches, the optimization approaches cannot guarantee an optimal solution, but they can provide the near-optimal solutions within a reasonable period of time. In many cases, these near-optimal solutions are sufficient and can guide to an improved design of critical systems by satisfying critical safety requirements. More importantly, the optimization approaches can be applied to ASILs allocation of large-scale systems, which is not possible using exact approaches.

Although, in terms of execution time, optimization approaches generally perform better than exact approaches, these approaches have their own issues. For example, the integer linear programming approach is flexible and simple, and it can solve complex problems. The issue with this approach is that the problem in hand must be linear in nature, but in the real world, problems rarely have a linear nature. Moreover, the ILP approach concentrates on either maximizing or minimizing an objective function, meaning it is not suitable for multi-objective optimization problems, but most real-world problems in dynamic environments have multiple objectives to optimize.

From the review, it is noticed that improvement (in terms of execution time) has been made within the optimization approaches by following different strategies. For example, the PeSOA approach has made improvement over GA and Tabu search-based approaches by reducing the search space by using the concept of equivalent classes and also by applying the concept of multi-group communication. However, the multi-group communication can lead to longer execution time if there exists only one solution to a given problem. On the other hand, the ant colony-based approach made improvement over all other approaches by cutting the search space by adopting two strategies and by formulating the ASILs allocation problem as a graph-search problem, which makes it the superior approach so far.

Although the approaches found in the literature help to automate the ASILs allocation problem, there exist many challenges in this area. From the review, it is seen that the exact approaches guarantee a solution for smaller systems. However, they do not scale for larger systems. Future research may attempt to improve the performance of the exact solvers by developing a more efficient encoding mechanism or by utilizing the increased computing power available in modern machines. As suggested in [23], one possible option may be the use of parallelization to expedite the search process. This is also applicable to the optimization approaches.

At present, all the approaches find solution to the ASILs allocation problem based on qualitative analysis of fault trees, i.e., based on the minimal cut sets. The failure rates/probabilities of the basic events of the fault trees are not utilized in the ASILs allocation process. In the future, it is worthwhile performing research to develop higher level functionalities by taking extra parameters into account, such as the quantitative data (e.g., failure rates) related to the basic events of the minimal cut sets. At the same time, some of the existing approaches allow a system analyst to incorporate their preferences in the ASILs allocation process. Future trend of ASILs allocation problem may see the inclusion of other stakeholders' preferences in the allocation process. For instance, new techniques may be developed to allow the designers to add their preferences so that the designers' choices would be taken into consideration when identifying good solutions. Moreover, designers may be given more control on the ASILs allocation to components of the system by using a mix of cost heuristics and/or to take into account the cost information related to certain categories of components. In terms of experimental studies to verify the performance of the approaches, currently the approaches are mostly evaluated by applying them to different version of hybrid braking systems. In the future, we may expect to see the use of different case studies and problem formulation methods to test each approach in order to achieve better performance.

Conclusion

In this paper, we presented an overview of various concepts related to the automotive safety integrity levels allocation and decomposition problem. We introduced ASILs allocation as a real-world problem having a combinatorial nature that is directly related to peoples' lives. The lack of such a review of the literature that gives an overview of the problem was the main motivation to review the concepts and the approaches used to allocate ASILs. The review introduced the origins of the problem and covered a background study of the fundamental concepts related to the ASILs allocation problem. We briefly presented the IEC 61508 and ISO26262 standards; then, we provided a detailed explanation of the ASILs allocation problem, including the fitness function and the solution quality measurements. Next, we reviewed different methods dedicated to allocate ASILs by categorizing them as exact and optimization approaches according to their nature. We provided a detailed explanation for the resolution methodology of each approach, followed by a review that contains the strengths and weaknesses of each technique. Finally, we discussed the remaining challenges in the ASILs allocation problem. In the future, we plan to extend the review by including the safety requirement allocation in other areas such as Development Assurance Levels (DALs) in the aerospace industry.

Open Access This article is distributed under the terms of the Creative Commons Attribution 4.0 International License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made.

References

1. *Road Vehicles—Functional Safety, ISO 26262*, International Organization for Standardization (2011)
2. *Functional Safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems, IEC* 61508, International Electrotechnical Commission (1997)

3. W.W. Lowrance, *Of Acceptable Risk: Science and the Determination of Safety* (William Kaufmann Inc, Los Altos, 1976)
4. T. Toroyan, Global status report on road safety. Inj. Prev. **15**, 286 (2009)
5. D. Parker, M. Walker, L.S. Azevedo, Y. Papadopoulos, R.E. Araújo, Automatic decomposition and allocation of safety integrity levels using a penalty-based genetic algorithm, in *International Conference on Industrial Engineering Other Applied Intelligent Systems* (Springer, Berlin, Heidelberg, 2013), pp. 449–459
6. L.P. Azevedo, D. Parker, M. Walker, Y. Papadopoulos, R.E. Araujo, Automatic decomposition of safety integrity levels: Optimization by Tabu search, in *SAFECOMP 2013-Workshop CARS (2nd Workshop on Critical Automotive Applications Robustness Safety) 32nd International Conference on Computer Safety, Reliability, and Security* (2013)
7. Y. Papadopoulos, M. Walker, M.-O. Reiser, M. Weber, D. Chen, S.D. Törngren, A. Abele, F. Stappert, H. Lönn, L. Berntsson, R. Johansson, F. Tagliabo, S. Torchiaro, A. Sandberg, Automatic allocation of safety integrity levels, in *Proceedings of 1st Workshop Critical Automotive Applications: Robustness and Safety* (2010), pp. 7–10
8. M. Sallak, C. Simon, J.F. Aubry, A fuzzy probabilistic approach for determining safety integrity level. IEEE Trans. Fuzzy Syst. **16**, 239–248 (2008)
9. J. Beugin, D. Renaux, L. Cauffriez, A SIL quantification approach based on an operating situation model for safety evaluation in complex guided transportation systems. Reliab. Eng. Syst. Saf. **92**, 1686–1700 (2007)
10. H. Zhang, W. Li, J. Qin, Model-based functional safety analysis method for automotive embedded system application, in *International Conference on Intelligent Control and Information Processing* (2010), pp. 761–765
11. Y. Lee, J. Kim, J. Kim, I. Moon, A verification of fault tree for safety integrity level evaluation, in *ICCAS-SICE* (2009), pp. 5548–5551
12. W.E. Vesely, M. Stamatelatos, J. Dugan, J. Fragola, J. Minarick, J. Railsback, *Fault tree Handbook with Aerospace Applications* (NASA Office of Safety and Mission Assurance, Washington, 2002)
13. S. Kabir, An overview of Fault Tree Analysis and its application in model based dependability analysis. Expert Syst. Appl. **77**, 114–135 (2017)
14. Y. Papadopoulos, J. Mcdermid, R. Sasse, G. Heiner, Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. J. Reliab. Eng. Syst. Saf. **71**, 229–247 (2001)
15. Y. Papadopoulos, M. Walker, D. Parker, S. Sharvia, L. Bottaci, S. Kabir, L. Azevedo, I. Sorokos, A synthesis of logic and bio-inspired techniques in the design of dependable systems. Annu. Rev. Control **41**, 170–182 (2016)
16. S. Kabir, M. Walker, Y. Papadopoulos, Dynamic system safety analysis in HiP-HOPS with Petri Nets and Bayesian networks. Saf. Sci. **105**, 55–70 (2018)
17. A. Joshi, M.P.E. Heimdahl, S.P. Miller, M.W. Whalen, *Model-Based Safety Analysis,* NASA Technical Report, NASA/CR-2006-213953 (NASA Langley Research Center, Hampton, 2006)
18. S. Sharvia, S. Kabir, M. Walker, Y. Papadopoulos, Model-based dependability analysis: state-of-the-art, challenges, and future outlook, in *Software Quality Assurance in Large Scale and Complex Software-Intensive Systems* (2015), pp. 251–278
19. J.I. Aizpurua, E. Muxika, Model-based design of dependable systems: limitations and evolution of analysis and verification approaches. Int. J. Adv. Secur. **6**, 12–31 (2013)
20. M. Bertoluzzo, G. Buja, R. Menis, G. Sulligoi, Approach to steer-by-wire system design, in *International Conference on Industrial Technology* (2005), pp. 443–447
21. Y. Papadopoulos, M. Walker, D. Parker, E. Rüde, R. Hamann, A. Uhlig, U. Grätz, R. Lien, Engineering failure analysis and design optimisation with HiP-HOPS. Eng. Fail. Anal. **18**, 590–608 (2011)
22. S. Kabir, M. Walker, Y. Papadopoulos, E. Rüde, P. Securius, Fuzzy temporal fault tree analysis of dynamic systems. Int. J. Approx. Reason. **77**, 20–37 (2016)
23. A. Murashkin, L.P. Azevedo, J. Guo, E. Zulkoski, J.H. Liang, K. Czarnecki, D. Parker, Automated decomposition and allocation of automotive safety integrity levels using exact solvers. SAE Int. J. Passeng. Cars Electron. Electr. Syst. **8**, 70–78 (2015)
24. E. Tsang, *Foundation of Constraint Satisfaction* (Academic Press, London, 1993)
25. L. De Moura, N. Bjørner, Satisfiability modulo theories: introduction and applications. Commun. ACM **54**, 69–77 (2011)
26. R. Vanderbei, *Linear Programming* (Springer, New York, 2007)
27. C. Prud'homme, J.G. Fages, X. Lorca, *Choco Documentation. TASC, LS2N, CNRS UMR 6241 and COSLING S.A.S.* http://www.choco-solver.org (2017)
28. Z3 Solver, The Z3 theorem prover. https://github.com/Z3Prover/z3 (2017)
29. IBM, CPLEX Optimizer. http://www-01.ibm.com/software/commerce/optimization/cplex-optimizer/ (2017)
30. N. Jussien, G. Rochart, X. Lorca, Choco: an open source java constraint programming library, in *CPAIOR'08 Work. Open-Source Software for Integer Constraint Programming* (2008), pp. 1–10
31. L. De Moura, N. Bjørner, Z3: an efficient SMT solver, in *International Conference on Tools and Algorithms for the Construction and Analysis of Systems* (Springer, Berlin, 2008), pp. 337–340
32. J. Guo, E. Zulkoski, R. Olaechea, D. Rayside, K. Czarnecki, S. Apel, J.M. Atlee, Scaling exact multi-objective combinatorial optimization by parallelization, in *Proceedings of 29th ACM/IEEE International Conference on Automation* (2014), pp. 409–420
33. R. Olaechea, D. Rayside, J. Guo, K. Czarnecki, Comparison of exact and approximate multi-objective optimization for software product lines, in *Proceedings on 18th International Software Product Line Conference* (2014), pp. 92–101
34. Clafer, Clafer lightweight modelling language. http://www.clafer.org/ (2017)
35. R. de Castro, R.E. Araújo, D. Freitas, Hybrid ABS with electric motor and friction Brakes, in *22nd International Symposium on Dynamics of Vehicles on Roads and on Tracks* (2011), pp. 1–7
36. L.P. Azevedo, Hybrid Braking System for Electrical Vehicles: Functional Safety. M.Sc. thesis, Porto University (2012)
37. M.S. Dhouibi, J.M. Perquis, L. Saintis, M. Barreau, Automatic decomposition and allocation of safety integrity level using system of linear equations, in *4th International Conference on Performance, Safety and Robustness in Complex Systems and Applications* (2014), pp. 1–5
38. H. Anton, *Elementary Linear Algebra* (Wiley, New York, 2010)
39. G. Allaire, S.M. Kaber, *Numerical Linear Algebra* (Springer, New York, 2008)
40. C.Y. Young, *College Algebra* (Wiley, New York, 2012)
41. G. Williams, *Linear Algebra with Applications* (Jones & Bartlett Publishers, Burlington, 2012)
42. M.S. Dhouibi, L. Saintis, M. Barreau, J.M. Perquis, Safety driven optimization approach for automotive systems, in *Annual Reliability and Maintainability Symposium* (2015), pp. 1–7
43. R. Mader, E. Armengaud, A. Leitner, C. Steger, Automatic and optimal allocation of safety critical integrity levels, in *Annual Reliability and Maintainability Symposium* (2012)
44. R.S. Garfinkel, G.L. Nemhauser, *Integer Programming* (Wiley, New York, 1972)

45. EAST-ADL Association, EAST-ADL V2.1.12 specification. http://www.east-adl.info/Specification.html (2014)
46. ATESST2 Project Consortium, *EAST-ADL Domain Model Specification. ATESST2, Deliverable D.4* (2010)
47. A. Lanusse, Y. Tanguy, H. Espinoza, C. Mraidha, S. Gerard, P. Tessier, R. Schnekenburger, H. Dubois, F. Terrier, Papyrus UML: an open source toolset for MDA, in *Proceedings of the Fifth European Conference on Model-Driven Architecture Foundations and Applications* (2009), pp. 1–4
48. M. Mitchell, *An Introduction to Genetic Algorithms* (MIT Press, Cambridge, 1998)
49. D.W. Coit, A.E. Smith, Reliability optimization of series-parallel systems using a genetic algorithm. IEEE Trans. Reliab. **45**, 254–260 (1996)
50. F. Glover, M. Laguna, Tabu search, in *Handbook of Combinatorial Optimization*, ed. by P.M. Pardalos, D.-Z. Du, R.L. Graham (Springer, New York, 2013), pp. 3261–3362
51. P. Hansen, K.-W. Lih, Heuristic reliability optimization by tabu search. Ann. Oper. Res. **63**, 321–336 (1996)
52. P. Hansen, B. Jaumard, Algorithms for the maximum satisfiability problem. Computing **44**, 279–303 (1990)
53. Y. Gheraibia, A. Moussaoui, Penguins search optimization algorithm (PeSOA), in *International Conference on Industrial Engineering Other Applied Intelligent Systems* (Springer, Berlin, Heidelberg, 2013), pp. 222–231
54. Y. Gheraibia, A. Moussaoui, L.S. Azevedo, D. Parker, Y. Papadopoulos, M. Walker, Can aquatic flightless birds allocate automotive safety requirements? in *Seventh International Conference on Intelligence Computational Intelligence Systems* (2015), pp. 1–6
55. Y. Gheraibia, K. Djafri, H. Krimou, Ant colony algorithm for automotive safety integrity level allocation. Appl. Intell. **48**, 555–569 (2017)
56. Y. Gheraibia, K. Djafri, H. Krimou, Reduction of solution space in the automotive safety integrity levels allocation problem, in *Modeling Implementation Complex System* (Springer, Cham, 2016), pp. 67–76
57. M. Dorigo and G. Di Caro, Ant colony optimization: a new metaheuristic, in *Proceedings of the 1999 Congress on Evolutionary Computation* (1999), pp. 1470–1477
58. M. Dorigo, M. Birattari, T. Stutzle, Ant colony optimization. IEEE Comput. Intell. Mag. **1**, 28–39 (2006)