Optimization of the Design of Diagnostic Systems through Requirement-Based Feature Selection

Melissa Gresser[1], Dominik Brehl[1], Marc David Rabe[2], and Bernard Bäker[2]
[1] BMW Group, Munich, Germany
[2] Institute of Automobile Engineering, Dresden, Germany
Email: {Melissa.Gresser, Dominik.Brehl}@bmw.de, marc_david.rabe@mailbox.tu-dresden.de, bernard.baeker@tu-dresden.de

***Abstract*—The classification of faults based on available diagnostic information in large complex mechatronic systems is a well-studied research subject. One challenge is the classification of faults with missing or ambiguous diagnostic information present. In real-world applications, this has the effect of misclassified faults, resulting in costly replacements of functional components. As even the acquisition of diagnostic information is afflicted with costs and part of the diagnostic functions are mandatory, the design of diagnostic content at an early system development stage is essential. The paper presents a procedure to automatically allocate a set of diagnostic functions for a complex mechatronic system that is informative and low in costs. It proposes a granular graph structure called the Diagnostic Cover Graph (DCG) to represent the system functionalities and the respective diagnostic functions. The DCG can represent restrictions on the availability of functionalities, restrictions on the executability of diagnostic functions and potential interdependence between them. On its basis, a decision table is generated, and a minimal test cost feature selection method such as the Test-Cost-Sensitive Quick Reduct (TCSQR) is conducted. The approach guarantees the proposal of a close-to-optimal subset of diagnostic functions. The capability of the presented procedure is demonstrated for the vehicle brake system for several diagnostic scenarios.**

***Index Terms*—diagnostic system, vehicle diagnostics, cost-sensitive feature selection, granularity graph, rough set theory**

I. Introduction

With the ongoing growth in complexity in large mechatronic systems, the demands for complete and reliable diagnostic coverage to detect and identify present faults is reinforced. Experience shows that missing or ambiguous diagnostic information can lead to components falsely being classified as faulty and as a result, be unnecessarily replaced. The careful planning and securing of diagnostic content in the early development stage of the system thus saves money in the long-term perspective of its usage.

One widely used approach for system diagnosis is the application of expert systems, such as rule-based and model-based diagnostic expert systems. Imitating human decision-making skills, expert systems use a knowledge base containing diagnostic information and determine the cause of a fault through an inference engine. As the process of knowledge acquisition is an essential factor for its fault diagnosis capabilities, the approach reaches its limitations for increasingly complex systems. [1]

This limitation is not the case for machine learning methods, which is why they gain popularity in the field of diagnostics [2]. Machine learning methods use historical system data to identify fault patterns and thereby create artificial knowledge to diagnose faults. However, machine learning methods are prone to suffer from the so-called "curse of dimensionality", which refers to the problems caused by dealing with a large number of features and impairs the significance of the model [3], [4]. For this reason, feature extraction or feature selection methods are applied as a preceding step to reduce the data set to its most informative dimensions.

The diagnostic design for a system includes the selection of diagnostic functions that observe specific system functions and create diagnostic information in case of a malfunction or a fault. In this context, a *malfunction* stands for the abnormal behavior of a component whereas a *fault* denotes the actual defect. The diagnostic design contains pre-planning the available diagnostic information in case of malfunctions which later can be used as features to identify an existing fault cause. Initially, there is access to limited system information as the diagnostic design is carried out in the system's developing time. The functional and structural knowledge of the system is progressively refined without the system's presence. Traditionally, experts of the respective component plan the diagnostic functions using their empirical knowledge about normal and faulty component behavior from experience and historical data [1]. What is mostly unknown in the developing phase and thus mainly disregarded in the design of diagnostics is the cross-system propagation of faults involving overlapping and interacting system functions and resulting in multiple malfunctions.

Manuscript received March 18, 2021; revised May 28, 2021.

doi: 10.18178/ijeee.9.2.26-35

With the trend towards automation of processes and increasing complexity of systems, the design of diagnostic content and hence the allocation of diagnostic functions should be automated as well. Several factors need to be considered for the automation of the diagnostic design. Firstly, restrictions on the availability of a component's functionality can be present that restrict the ability to observe a potential malfunction. Another term for this is *path sensitization*, which means that the right conditions need to prevail in order to be able to observe the respective fault. In particular, the created diagnostic content should achieve both unconditional coverage and resolution. In this context, *coverage* means that the set of diagnostic functions can detect the occurrence of every possible fault. *Resolution* means that the created diagnostic information allows the fault's explicit classification by creating a unique fault pattern. [5]

While the diagnostic content of a system can be mandatory to some extent (e.g. on-board diagnostics for operational security or emission-related diagnostics in vehicles), the remaining system's diagnostic approach must be carefully planned. The challenge is to capture diagnostic functions of different granular level, meaning different levels of detail to the implied diagnostic information, and the various conditions concerning their executability. For instance, diagnostic functions can require a specific operation context to produce reliable results, or there are dependencies between diagnostic functions. At the same time, the implementation of diagnostic functions induces costs, which must be considered in their choice. Other factors include the *robustness* of the fault detection method, which is defined as the "maximization of the detectability and isolability of faults together with the minimization of the effect of uncertainty and disturbance" [6].

This paper proposes a procedure to automate the task of choosing diagnostic functions during the design phase of diagnostic content in a domain-independent, efficient and cost-sensitive way. The implementation of the diagnostic functions itself is not the focus and thus not included.

Its main contribution is the Diagnostic Cover Graph to represent functional correlations and the associated diagnostic functions of a system in a granular graph structure. It can deal with the different granularity of diagnostic functions, the dependencies among diagnostic functions and restrictions on both the availability of the component functionality and the executability of the diagnostic functions. A successive cost-sensitive feature selection method is executed to eliminate diagnostic functions with non-essential information and prioritize low cost and essential diagnostic functions. The procedure's goal is to design a diagnostic knowledgebase that achieves complete coverage and high resolution to classify faults, beginning in a system's development stage and throughout its entire usage.

The paper is organized as follows: In Section II, a short review of related work and a description of the cost-sensitive quick reduct algorithm is presented. Section III describes the proposed procedure to allocate diagnostic functions in detail. In Section IV, the procedure is applied to the brake system of a vehicle. The main conclusions are drawn in Section V, and future research potential is pointed out.

II. Background

System knowledge in its comprehensive nature and varying depth is a cornerstone for system diagnostics. A graph is a perfect structure to store this knowledge as it can represent different types of knowledge bases in an intuitive and versatile manner. For high-dimensional system data, e.g. in the form of measurements, it is advisable to use dimensionality reduction techniques to create a dense and more significant database first.

In the following, an outline of related work is given, focusing on their diagnostic design approach. After that, the used algorithm for test-cost sensitive dimensionality reduction based on rough set theory is explained. This part is included for completeness and can be left out as the algorithm is exchangeable with other feature extraction methods.

A. Related Work

In the literature, there are two main approaches to the theory and design of diagnostic reasoning systems. The first approach uses first-order principles of the system, including structural and functional information. The second approach is also called the experiential approach, as it uses heuristic information such as measurement data and expert knowledge.

In the field of failure and risk analysis, diagnostic reasoning can be done through several methods. Fault Tree Analysis (FTA) is a well-known technique, which uses a graph tree structure with logical gates to represent the dependability of a high-level failure event to contributing failure events on lower levels [7]. Another approach is the failure mode and risk analysis (FMEA), which decomposes the system into finer granular levels. It reasons through a backward logic how a low-level failure event would affect immediate higher-level events [8]. Both approaches depend on already existing fault knowledge and thus can not be used in the initial stage of development.

To overcome this shortcoming, [9] proposes a hierarchical system model of functionality and configuration in the form of a graph, using only information present in the conceptual system design. It serves as a basis to understand functional failures and their propagation paths in complex systems. Another work for the early development stage is [10], which quantifies the failure propagation potential of a failure by using a graph that models the function block diagram. These are just a few examples of the applicability of a graph structure in the design of diagnostic content.

To ensure diagnostic coverage and resolution of the diagnostic content, [5] proposed a procedure for digital electronics using only structural and functional information to automatically design diagnostics for a device to achieve both diagnostic coverage and resolution. An information path model and a simple fault model are

established to minimize inquiry conditions and derive diagnostic functions.

Using minimal hitting set algorithms to compute the minimal diagnostic content was first explored in [11] and was further developed in various other research [12].

The minimal hitting set problem includes the computation of a minimal hitting set of S, which is a minimal set of S that has a non-empty intersection with all subsets of S. The approach is based on the notation that constructing minimal diagnostic content is identical to the computation of a minimal hitting set of a set of conflicts sets, which is the set of system elements potentially responsible for a fault.

Another approach to reducing diagnostic content is to use dimensionality reduction methods. They are mostly known as an optional preprocessing step to remove irrelevant or redundant features from data sets to diminish the curse of dimensionality in machine learning algorithms [4]. While *feature extraction methods* create new features from the initial data set, *feature selection methods* maintain the data's semantic by selecting a feature subset. Only the latter is thus suitable to achieve minimal diagnostic content. In many real-world scenarios, the acquisition of diagnostic information also involves costs, which cannot be neglected while selecting features [13].

The presented technique uses the first principles of the system to model a graph containing system functionalities in a granular view. It can thus be applied in the early stage of the system development. Interactions between functionalities are omitted in the presented scope. Depending on the development progress and the comprehensive knowledge about the diagnostic functions, the technique can be first used to create an initial minimal set of diagnostic functions and later be used to obtain a minimal set that considers the updated diagnostic conditions. For a given set of diagnostic functions, the proposed graph structure is versatile enough to imply their level of detail, potential dependencies between them and possible conditions to their executability. The mentioned methods mostly neglected this circumstance.

To the authors' knowledge, using this kind of multi-granular graph structure to represent diagnostic coverage in combination with the cost-sensitive selection of diagnostic functions is a new approach to the design of diagnostic content.

B. Test-Cost-Sensitive Feature Selection via Rough Sets

One approach is to perform feature selection on data sets of discrete values is rough set attribute reduction. It relates to the formal mathematical tool of the rough set theory first described by Z. Pawlak [14] and aims to reduce redundant attributes of data sets while maintaining its informative value. The approach stands out, as it is fast and efficient, does not change the attribute semantics and uses conventional set theory operations [15].

Formally, an information system is represented as a table where the objects are the rows, and the attributes are the columns. It is denoted as $I = (U, C)$, where U is a non-empty finite set of objects, called the universe and C is the non-empty, finite set of conditional attributes. It is $a: U \rightarrow V_a$ where V_a is the set of values of the attribute $a \in C$ and $a(x)$ denotes the value of $x \in U$ over a. For a decision system, it is $I = (U, C \cup D)$ with $C \cap D = \emptyset$ where D is the set of decision attributes of $x \in U$.

Rough set theory can be used for attribute subset selection (or feature reduction) for discrete-valued attributes as well as for classification [16]. It is based on the establishment of *equivalence relations* or *P indiscernibility relation* $IND(P)$ for $P \subset C$:

$$\mathrm{IND(P)} = \{ (x, y) \in U^2 \mid a(x) = a(y)\ \forall a \in P \}$$

meaning that objects in $IND(P)$ are equivalent or *indiscernible* with respect to the attributes in P. Given the equivalence relation $IND(P)$, a partition of the universe in *equivalence classes* $[x]_P$ is induced. The *quotient set* X/P is then defined as the set of all equivalence classes $[x]_P$ for $x \in X$. The rough set of the set $X \subseteq U$ is approximated using only information in $P \subseteq C$ by the *lower* and *upper approximations* of X as defined below:

$$\underline{\mathrm{P}}\mathrm{X} = \{ x \in U \mid [x]_P \subseteq X \}$$

$$\overline{\mathrm{P}}\mathrm{X} = \{ x \in U \mid [x]_P \cap X \neq \emptyset \}$$

The so-called *positive region* then contains all objects in U that are certain to belong to a target set X considering the attributes in P. For $IND(P), IND(Q)$ with $P, Q \subset C$ as equivalence relations over U it is defined as:

$$\mathrm{POS_P(Q)} = \cup_{X \in U/Q}\ \underline{\mathrm{P}}\mathrm{X}$$

From this, the rough set *degree of dependency* is obtained by [14]:

$$\gamma_P(Q) = \frac{|POS_P(Q)|}{|U|}$$

It is $0 < \gamma_{P(Q)} < 1$, which signifies no dependency in case of 0 to complete dependency in case of 1 between the attribute sets Q and P. Subsequently, the significance of an attribute $f \in P$ is calculated by:

$$\sigma_P(Q, x) = \gamma_P(Q) - \gamma_{P-\{x\}}(Q)$$

Having removed all non-significant attributes from the data set, one receives the *attribute reduct* or just *reduct*, which is defined as:

$$R = \{X: X \subseteq C, \gamma_C(D) = \gamma_R(D)\}$$

for D as the set of decision attributes. There are multiple attribute reducts for one dataset. Finding reducts with minimal cardinality $R_{min} \subseteq R$ is a NP-hard problem [17].

When the cost to gather the features is also considered, the problem changes into finding a reduct with minimal test cost. One possible approach is the *test-cost-sensitive quick reduct* (short TSCQR) algorithm introduced in [13], which is an extension of the *QuickReduct* (short QR) algorithm developed in [15]. The TSCQR algorithm shown in Algorithm 1 starts from an empty set and then

adds attributes depending on the locally optimal choice through the optimality criterion

$$\alpha\gamma_{\mathrm{R}}(\mathrm{D}) + (1-\alpha)\left(1-\mathrm{t}_{\mathrm{c_R}}\right)$$

where $\gamma_R(D)$ is the rough set dependency degree and t_{C_R} denotes the sum of feature costs (normalized to the range of 0-1) in the reduct. The parameter $\alpha \in [0,1]$ can be varied to either give more significance to the increase of the dependency degree or to the decrease of total costs in the choice of features to be added. Experiments have shown that choosing $\alpha \in (0.3,0.4)$ improves the optimization's performance in most cases [13].

Algorithm 1 TCSQuickReduct

procedure TCSQUICKREDUCT $(C, D, q(), \alpha\}$
  $C \leftarrow$ the set of all conditional features
  $D \leftarrow$ the set of all decision features
  $q() \leftarrow$ cost function of decision features
  $R \leftarrow \emptyset$
  repeat
    $T \leftarrow R$
    max$T \leftarrow 0$
    for $f \in (C - R)$
      val $= \alpha \cdot \gamma_{R\cup\{f\}}(D) + (1-\alpha) \cdot \left(1 - \mathrm{sum}(q(R \cup \{f\}))\right)$
      if val $>$ max$T$ **then**
        $T \leftarrow R \cup \{f\}$
        max$T \leftarrow$ val
    $R \leftarrow T$
  until $\gamma_R(D) == \gamma_C(D)$
  $MR \leftarrow$ sort attributes in R according to test costs in descending order
  repeat
    $MR \leftarrow MR - \{f\}$
    If $\gamma_{R-\{f\}} == \gamma_C(D)$ **then**
      $R \leftarrow R - \{f\}$
  until $MR == \emptyset$
  return R
end procedure

As stated in [13], the TCSQR uses a greedy algorithm and thus does not necessarily return a global minimum as not all possible combinations of attributes are explored. However, it does derive a close-to-optimal reduct in a decent time.

III. Diagnostic Cover Graph

The Diagnostic Cover Graph is defined for a mechatronic system or an application, containing a wide range of functionalities described on multiple granular levels. Without loss of generality, only one malfunction is assigned to each function, and the malfunction can be observed from a respective diagnostic function.

A. General Structure

Formally, diagnostic coverage of a mechatronic system can be represented as a multi-granular heterogenous attributed network. The *Diagnostic Cover Graph* (DCG) $G = (V, E, A)$ is a graphical representation of the system's functionalities, the malfunctions and the diagnostic functions. The functionalities are arranged according to their granular level, and they have a diagnostic function attached that can observe the potential malfunction. It aims to portray the overall connections between functionalities and their diagnostic functions and illustrate the coverage of finer granular level functionalities by diagnostic functions of a coarser granular level.

A representation of the Diagnostic Cover Graph is given in Fig. 1. The nodes $V = \mathcal{N} \cup \mathcal{D}$ are either of type functionality, or *functionality node* for short, which corresponds to $n_i^{(\mathrm{j})} \in \mathcal{N}$, for $i = 1, \ldots, n$ or of type diagnostic function, which is *diagnostic node* for short, which corresponds to $d_i^{(\mathrm{j})} \in \mathcal{D}$ for $i = 1, \ldots, m$. The granular level is set from coarse to refined for $j = 1, \ldots, p$.

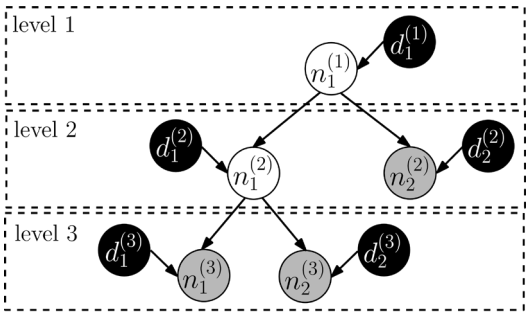

Figure 1. Diagnostic cover graph.

As for the individual attributes per type, the functionality node n_i has the set of attributes $\mathcal{A}_i = \{c(n_i), l(n_i), r(n_i)\}$ where $c: \mathcal{N} \rightarrow \mathcal{C}_i$ and $\mathcal{C}_i \subseteq \mathcal{C}$ represents the availability condition of the functionality and $l: \mathcal{N} \rightarrow \{\text{normal, faulty}\}$ is the set of labels, which are synonymous for the desired and the undesired behavior of a functionality. The condition attribute $\mathcal{C}_i$ of a node is a subset of the finite, discrete condition set $\mathcal{C} = \{\mathrm{cond}_1, \ldots, \mathrm{cond}_z\}$. Additionally, the functionality node has a relevance $r: \mathcal{N} \rightarrow \{0, 1\}$ assigned to it, which specifies the level of detail requested in the diagnostic information. In this case $r(n_i) = 1$ means that a malfunction in n_i needs to be uniquely identifiable, otherwise $r(n_i)$ is set to 0.

For diagnostic nodes d_i, it is $\mathcal{A}_i = \{c(d_i), m(d_i), r(d_i)\}$ where $c: \mathcal{D} \rightarrow \mathcal{C}_i$ and $\mathcal{C}_i \subseteq \mathcal{C}$ represents the executability condition, $m: \mathcal{D} \rightarrow R_+$ its cost and $r: \mathcal{D} \rightarrow \{0,1\}$ its relevance. In this context, the relevance is to differentiate between mandatory diagnostic functions with $r(d_i) = 1$ and non-mandatory diagnostic functions with $r(d_i) = 0$. The attributes of the specific nodes and their granular level can be adapted to match the present system.

The graph is semi-structured as follows: Nodes of type functionality are partitioned into their specific granular level. Depending on the simultaneous occurrence of the 'faulty' label in between granular levels, a directed edge can connect the nodes from coarser granular to more fine granular levels (compare to Fig. 1). The nodes of type diagnostic function $d_i^{(j)} \in \mathcal{D}$ are then connected through a directed edge to the functionality node $n_i^{(j)} \in \mathcal{N}$, given that they are able to detect a malfunction in the functionality. This structure is interpreted as follows: the

diagnostic function $d_i^{(j)}$ is able to detect the change of the label $l\left(n_i^{(j)}\right)$ of all nodes of type functionality that are connected through a directed path in the form of $w = \left(d_i^{(j)}, n_i^{(j)}, n_i^{(j-1)}, n_i^{(j-2)}, \dots, n_i^{(p)}\right)$ with $d_i^{(j)}$. One diagnostic function can also observe several functionalities in the same granular level by adding multiple edges to the respective nodes.

Without loss of generality, all functionality nodes only have two labels, and the attached diagnostic functions can unambiguously observe a label change. If this is not the case, either an additional functionality node needs to be added, or the functionality node needs to be refined to meet the criterion.

A more variable setting is needed to fully represent a potential discrepancy between the availability condition of a functionality node and the executability condition of an attached diagnostic node. Moreover, it is later described how the DCG can be adapted to illustrate dependencies between diagnostic nodes, a lower specified required level of detail for a label change and diagnostic functions that imply a higher-level of detail.

The coverage of label changes of functionalities through diagnostic functions across granular levels is illustrated by transforming it into the *Bipartite Cover Graph*, which is a bipartite directed graph B. The nodes are grouped into the disjoint and independent sets U and V, where $U = \{n_i | \, n_i \in \mathcal{N} \; s.t. \; r(n_i) = 1\}$ is the set of functionality nodes with an appropriate level of detail and $C = \{d_i | \, d_i \in \mathcal{D}\}$ is the set of all diagnostic functions regardless of their granular level. Depending on whether the functionality node $n_k \in U$ is reachable in the DCG, the diagnostic nodes $d_i \in C$ are connected through a directed edge to n_k as is shown in Fig. 2.

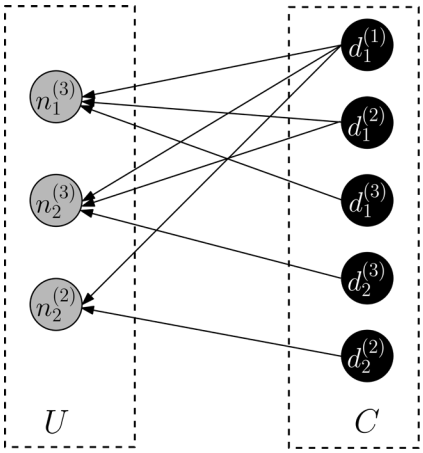

Figure 2. Bipartite cover graph.

Selecting a minimal subset of diagnostic functions in P such that label changes of all nodes in U are covered is equivalent to finding the solution of a *hitting set problem*: For a bipartite graph B with two independent and disjoint sets U and C, find a subset $P \subseteq C$ such that $U = \cup_{\{v \in P\}} N_{B[v]}$ where $N_B[v]$ denotes the neighborhood of node v in B. In this context, however, the solution would favor nodes of diagnostic functions on a coarser granular level as they cover more label changes and thus impede the resolution of diagnostic information on a more refined granular level. To achieve a higher resolution, the subset of diagnostic functions needs to be selected so that no informative diagnostic content is lost, as is the case for feature selection methods. Feature selection methods can be applied to the problem by converting the entailed information of the DCG into a decision table.

The DCG representation as a bipartite directed graph simplifies the transformation into a decision table by merely using the graph's adjacency matrix (see Table I). In this case, the set of relevant functionalities U denotes the universe set, and the set of diagnostic information from diagnostic functions C denotes the set of conditional attributes. The set D of decision attributes is chosen to be distinctive integers in order to distinguish between malfunctions. Hence, the row of the decision table represents a list of diagnostic functions that can observe the label change in the individual functionality.

TABLE I. DECISION TABLE

	$\boldsymbol{d_1^{(1)}}$	$\boldsymbol{d_1^{(2)}}$	$\boldsymbol{d_2^{(2)}}$	$\boldsymbol{d_1^{(3)}}$	$\boldsymbol{d_2^{(3)}}$	**D**
$\boldsymbol{n_1^{(3)}}$	1	1	0	1	0	1
$\boldsymbol{n_2^{(3)}}$	1	1	0	0	1	2
$\boldsymbol{n_2^{(2)}}$	1	0	1	0	0	3
no fault	0	0	0	0	0	0

B. Extended Structure

The above-described approach relates to the very trivial case to design diagnostic content, where there are no conditions to the availability of functionalities or to the executability of diagnostic functions. Moreover, potential interdependence among diagnostic functions is not yet considered. Through the automated adaptation of the DCG according to preset rules, the approach is easily extended to cover these situations. The following paragraphs outline these set of rules.

In case the executability conditions $c(d_i)$ of the diagnostic node $d_i \in \mathcal{D}$ do not match all of the availability conditions $c(n_i)$ of the individual functionality, meaning $c(d_i) \neq c(n_i)$ for $n_i \in \mathcal{N}$, the DCG has to be extended in order to represent this disparity. The conditions of both the diagnostic node and the function node are checked for conformity starting from the coarsest level of granularity. Suppose there is one condition $\text{cond}_k \in c(n_t)$ for $n_t \in \mathcal{N}$ that is not part of the set $c(d_t)$ for $d_t \in \mathcal{D}$, meaning that the diagnostic function can not observe the functionality during this condition. In this case, a copy of the subgraph $S \subset G$ is created starting from the node n_t, including all subsequent functionality nodes. Denoting the copy of the subgraph S', the condition of each $n_i' \in S'$ is set to only include cond_k, meaning $c(n_i') = \{\text{cond}_k\}$. For each $n_i \in S$ of the original subgraph, the condition attribute is set to exclude cond_k, meaning $c_{\text{new}}(n_i) = c_{\text{old}}(n_i) - \{\text{cond}_k\}$. The subgraph S' is then attached to the original graph through an edge of the predecessor diagnostic node

of n_t. Last, the diagnostic nodes of S are connected to nodes of S' according to their coverage of the availability conditions of diagnostic nodes in S'.

This approach is indicated in the *Augmented Diagnostic Cover Graph* (ADCG) in Figure 3. The diagnostic nodes

$$\mathcal{D} = \left\{\mathrm{d}_1^{(1)}, \mathrm{d}_{1\mathrm{a}}^{(2)}, \mathrm{d}_{1\mathrm{bc}}^{(2)}, \mathrm{d}_2^{(2)}, \mathrm{d}_{1\mathrm{a}}^{(3)}, \mathrm{d}_2^{(3)}, \mathrm{d}_{1\mathrm{b}}^{(3)}, \mathrm{d}_{1\mathrm{c}}^{(3)}\right\}$$

have diverse executability conditions, which are depicted as the added spots on the nodes. Assuming that the functionality nodes $\mathcal{N} = \left\{n_1^{(1)}, n_1^{(2)}, n_2^{(2)}, n_1^{(3)}, n_2^{(3)}\right\}$ are available in all conditions, the nodes in $\mathcal{N}$ are copied to match the executability conditions of $\mathcal{D}$, resulting in the set of functionality nodes $\mathcal{N}' = \mathcal{N} \cup \left\{n_1'^{(2)}, n_2'^{(3)}, n_1'^{(3)}, n_1''^{(3)}\right\}$.

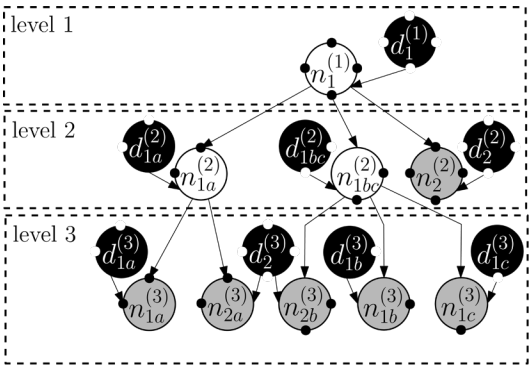

Figure 3. Augmented diagnostic cover graph.

Like the DCG, the ADCG is transformed into its bipartite counterpart with additional elements in the set U and the conditional attribute set C and then represented as the Table II. It should be noted that even though the number of nodes in U increases, the number of decision attributes D stay the same (compare to Table I). This puts the focus on detecting the malfunctions of functions during all their available conditions, without the need to distinguish between them.

TABLE II. DECISION TABLE FOR ADCG

	$d_1^{(1)}$	$d_{1a}^{(2)}$	$d_{1bc}^{(2)}$	$d_2^{(2)}$	$d_{1a}^{(3)}$	$d_{1b}^{(3)}$	$d_{1c}^{(3)}$	$d_2^{(3)}$	D
$n_1^{(3)}$	1	1	0	0	1	0	0	0	1
$n_1'^{(3)}$	1	0	1	0	0	1	0	0	1
$n_1''^{(3)}$	1	0	1	0	0	0	1	0	1
$n_2^{(3)}$	1	1	0	0	0	0	0	1	2
$n_2'^{(3)}$	1	0	1	0	0	0	0	1	2
$n_2^{(2)}$	1	0	0	1	0	0	0	0	3
no fault	0	0	0	0	0	0	0	0	0

As for interdependent diagnostic functions, such as the simultaneous implementation of several diagnostic functions or the diagnostic information generated by one diagnostic function serving as an input for other diagnostic functions, their combined choice must be considered in the design of diagnostic content as well. To represent interdependent diagnostic functions, the DCG is adjusted through additional edges between the diagnostic nodes in question. For $d_1^{(2)}$ as an input and $d_1^{(3)}$and $d_2^{(3)}$ as a simultaneous output, the interdependence is represented as depicted in Fig. 4.

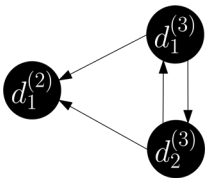

Figure 4. Realization of interdependence of diagnostic functions.

The interdependence between diagnostic functions is implicated in the decision table by grouping the diagnostic information attributes accordingly and assigning them to one column. For the diagnostic functions in Fig. 4. the attributes are grouped as $\{d_1^{(2)}, d_1^{(3)}, d_2^{(3)}\}$ and $\{d_1^{(2)}\}$.

A similar approach is performed if the resolution of diagnostic information for a label change of a function is not necessary to be high. Depending on the prescribed granular level of resolution, a diagnostic collection is added that combines the diagnostic nodes of lower successive functionality nodes. As depicted in Fig. 5. the diagnostic collection is defined as $d^{(3)} = \{d_1^{(3)}, d_2^{(3)}\}$. The diagnostic collection has no attributes except for the executability condition derived from the intersection of the executability conditions of its combined diagnostic functions, meaning $c\left(d^{(3)}\right) = c\left(d_1^{(3)}\right) \cap c\left(d_2^{(3)}\right)$. Thereby it functions as a group error, observing a label change in node $n_1^{(2)}$ with $r\left(n_1^{(2)}\right) = 1$ in level 2 if either of the diagnostic functions d_1^3 or d_2^3 observe a malfunction in the functionality nodes $n_1^{(3)}$ or $n_2^{(3)}$ of level 3.

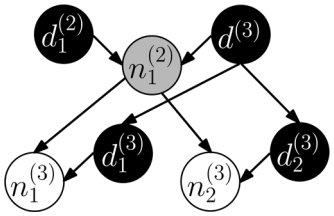

Figure 5. Realization of low diagnostic resolution.

Accordingly, the number of rows in the corresponding Table III are reduced, as the set of relevant functionalities U decreased.

TABLE III. DECISION TABLE FOR LOW DIAGNOSTIC RESOLUTION

	$d_1^{(2)}$	$d^{(3)}$	$d_1^{(3)}$	$d_2^{(3)}$	D
$n_1^{(2)}$	1	1	0	0	1
no fault	0	0	0	0	0

In case the created diagnostic information of one diagnostic function achieves high resolution to the extent such that a label change can be differentiated between multiple functionalities, the decision table as in Table I is not enough to incorporate its information value. One way to incorporate high resolution is to distinguish how the diagnostic node covers the functionality nodes. By adding

attributes to the edges in the form of different numbers, the unique node coverage is represented in the DCG. These attributes are then transferred to the Bipartite Cover Graph and the decision table shown in Table IV accordingly. For a high-resolution diagnostic node $d_1^{(2)}$ from Fig. 1. this means adding edges from $d_1^{(2)}$ to the functionality nodes $n_1^{(3)}$ and $n_2^{(3)}$ with attributes a_1, a_2 as integers and $a_1 \neq a_2$. The edge between $d_1^{(2)}$ and $n_1^{(2)}$ is deleted in the end.

TABLE IV. DECISION TABLE FOR HIGH DIAGNOSTIC RESOLUTION

	$d_1^{(1)}$	$d_1^{(2)}$	$d_2^{(2)}$	$d_1^{(3)}$	$d_2^{(3)}$	D
$n_1^{(3)}$	1	a_1	0	1	0	1
$n_2^{(3)}$	1	a_2	0	0	1	2
$n_2^{(2)}$	1	0	1	0	0	3
no fault	0	0	0	0	0	0

C. Diagnostic Function Selection

In all mentioned cases, a cost-sensitive feature selection algorithm is conducted based on the decision table to select the features in the form of diagnostic functions. At the same time, the grouping of attributes is enforced such that groups of diagnostic functions can only be added collectively to the feature set or not at all. In this manner, the set of possible diagnostic functions is reduced to an extent such that the total costs for diagnostics are minimized but not as far as any informative content, especially in the sense of diagnostic coverage or resolution, is lost.

It should be added that the costs for the optimization are not only restricted to monetary costs but can be a combination of factors (for instance diagnostic robustness or execution time). One way to do cost-sensitive feature selection is through the TCSQuickReduct Algorithm addressed in section II.A., which is adapted to handle the grouping of features. Whether a global cost-minimal set of diagnostic functions for the system is achieved is strongly dependent on the choice of the feature selection method.

IV. EXPERIMENTAL ANALYSIS

An example system is constructed to demonstrate the above-described procedure through the Diagnostic Cover Graph. The experiments' objective is to present the general approach of the procedure and point out its capabilities using various scenarios. The procedure can be used in a universal context for both technical and non-technical systems. Hereafter, the application is presented for enabling diagnostic information of the brake system of a vehicle with varying conditions to the selection of diagnostic functions.

A vehicle brake system's function is to decelerate the vehicle even as far as to a standstill, to prevent an unintentional acceleration or to hold a stationary vehicle in position. It can be divided into the operational brake, the emergency brake and the holding brake. Simply put, the operational brake functionality is as follows (see Fig. 6. for reference): As soon as the driver puts force on the brake pedal, the force is boosted by the brake booster to be then transformed by the main cylinder into hydraulic pressure. The brake fluid is thus pressured through the brake pipes to the disk brake in the front and the drum brake in the back, creating the tension needed to increase the wheels' friction force.

For this example, the operating brake system is extended with the holding brake. When the parking brake is tightened, it puts tension on the brake cable, which in turn creates tension in the brakes. The different granular levels for the vehicle brake system consist of - from coarse to refined -: system, sub-system, component, and state. Henceforth the vehicle brake system is limited to the elements outlined in Table V. Moreover, every element is comprised of only one functionality with one associated malfunction.

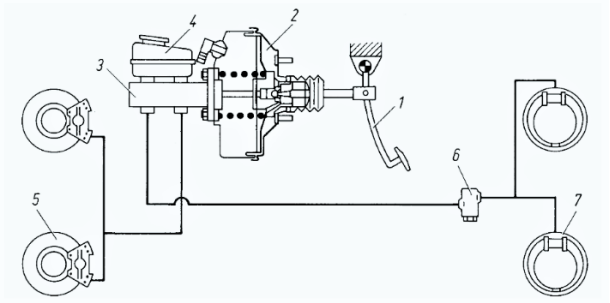

Figure 6. Brake system for the vehicle. 1. Brake pedal, 2. Brake booster, 3. Main cylinder, 4. Brake fluid reservoir, 5. Disk brake (in front), 6. Brake power distributor, 7. Drum brake (in back) (compare to [18]).

The column 'availability condition' states the modes, in which the element's functionality is available. Here it only differentiates between the modes $\mathcal{C} = \{'\text{park}', '\text{drive}'\}$. Accordingly, the operational brake functionality is only active in the 'drive' mode, whereas the holding brake can only be activated in the 'park' mode. The 'rel' column indicates the predefined level of detail for the diagnostic content. Here, the holding brake functionality only needs to be monitored on the granular level of the sub-system. The table also describes the relation of the elements across different granular levels in the column 'belongs to'.

TABLE V. ELEMENTS OF GRANULAR GRAPH FOR THE BRAKE SYSTEM

Granular Level	ID	Name	Availability Condition	Rel	Belongs To
System	h1	brake system	park, drive	0	
Sub-system	n1	operational brake	drive	0	h1
	n2	holding brake	park	1	h1
Component	c01	brake pedal	drive	1	n1
	c02	brake booster	drive	1	n1
	c03	main cylinder	drive	1	n1
	c04	brake fluid reservoir	drive	0	n1
	c05	disk brake	park, drive	1	n1, n2
	c06	brake power distributor	drive	1	n1
	c07	drum brake	park, drive	1	n1, n2
	c08	brake pipe	drive	0	n1
	c09	parking brake	park	0	n2
	c10	brake cable	park	0	n2
State	s	brake fluid fill level	drive	1	c04
	s	brake fluid pressure	drive	1	c08
	s	brake cable tension	park	0	c10

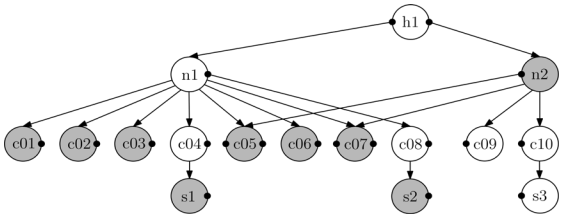

Figure 7. Granularity graph of the vehicle brake system.

TABLE VI. DIAGNOSTIC ELEMENTS OF THE BRAKE SYSTEM

ID	Observes	Cost	Executability Condition	Rel
d_h1	$h1$	4	park, drive	0
d_n1	n1	3	drive	0
d_n2	n2	3	park	0
d_c01	c01	2	drive	0
d_c02	c02	2	drive	0
d_c03	c03	2	drive	0
d_c04	c04	2	drive	0
d_c05	c05	2	park, drive	0
d_c06	c06	2	drive	0
d_c07	c07	2	park, drive	0
d_c08	c08	2	drive	0
d_c09	c09	2	park	0
d_c10	c10	2	park	0
d_s1	s1	1	drive	0
d_s2	s2	1	drive	0
d_s3	s3	1	park	0

The DCG is constructed by adding nodes according to the granular levels and adding edges according to the 'belongs to' column, as shown in Fig. 7. It is assumed that each malfunction can be observed by a diagnostic function, as implied in Table VI in the first stage. These represent the set of functionality nodes $\mathcal{N}$. The nodes $\mathcal{D}$ of the diagnostic functions are attached to the granularity graph as described in Section III. Table VII is derived by following the described approach. Since the resolution is only necessary as fine as the sub-system level of the holding brake, a diagnostic collection for its diagnostics is established. The diagnostic collection d_{coll}_n2 consists of the diagnostic functions of associated lower-level functionalities that is:

$$\mathrm{d_{coll1}\_n2} = \{\mathrm{d\_c05}, \mathrm{d\_c07}, \mathrm{d\_c09}, \mathrm{d\_c10}\}$$

$$\mathrm{d_{coll2}\_n2} = \{\mathrm{d\_c05}, \mathrm{d\_c07}, \mathrm{d\_c09}, \mathrm{d\_s3}\}$$

Moreover, the functionality nodes for the disk brake c05 and the drum brake $c07$ are split into their 'drive' and 'park' availability as they belong to both the operational brake and holding brake sub-system and have different availability conditions. A range of different scenarios is imaginable for possible restrictions on the design of diagnostic content in a vehicle system.

In the first scenario the brake system has no predefined diagnostic scope and no access to past knowledge about restrictive conditions of potential suitable diagnostic functions. Thus, the executability condition for the diagnostic function is initially set to match the availability condition of the individual functionality. The cost of all diagnostic functions is assumed to be one without knowing the dependency among them.

This baseline is extended in the second scenario by adding costs to diagnostic functions according to the granular level of the associated functionality. The 'cost' column in Table VI has adapted accordingly.

In a third scenario, a dependency among diagnostic functions is added. It is assumed that the diagnostic information of the diagnostic function of the brake pipe d_c08 is used as an input for the diagnostic functions d_c05, d_c06, and d_c07. Meaning if either d_c05, d_c06, or d_c07 are chosen for the diagnostic content, d_c08 necessarily also needs to be added. The added dependency results in the grouping $\{d\_c05, d\_c08\}$, $\{d\_c06, d\_c08\}$, $\{d\_c07, d\_c08\}$ and $\{d\_c08\}$.

In the fourth scenario, it is assumed that in addition to the other conditions, both the diagnostic function of the disk brake d_c05 and the drum brake d_c07 can only be executed in the 'drive' mode by setting $c(d\_c05) = \{\text{drive}\}$ and $c(d\_c07) = \{\text{drive}\}$. The 'executability condition' in Table VI has adapted accordingly.

Table VIII shows the selection of diagnostic functions for each scenario using the TSCQReduct Algorithm with $\alpha = 0.35$ and the sum of the associated costs. For comparison, the selection of all diagnostic functions is given as scenario 0. Consequently, the procedure was able to achieve a reduction of cost for all scenarios.

TABLE VII. DECISION TABLE OF THE VEHICLE BRAKE SYSTEM

	d_h1	d_n1	d_n2	d_{coll}_n2	d_c01	d_c02	d_c03	d_c04	d_c05	d_c06	d_c07	d_c08	d_s1	d_s2	D
n02	1	0	1	1	0	0	0	0	0	0	0	0	0	0	1
c01	1	1	0	0	1	0	0	0	0	0	0	0	0	0	2
c02	1	1	0	0	0	1	0	0	0	0	0	0	0	0	3
c03	1	1	0	0	0	0	1	0	0	0	0	0	0	0	4
c05$_{\text{drive}}$	1	1	0	1	0	0	0	0	1	0	0	0	0	0	5
c05$_{\text{park}}$	1	0	1	1	0	0	0	0	1	0	0	0	0	0	5
c06	1	1	0	0	0	0	0	0	0	1	0	0	0	0	6
c07$_{\text{drive}}$	1	1	0	1	0	0	0	0	0	0	1	0	0	0	7
c07$_{\text{park}}$	1	0	1	1	0	0	0	0	0	0	1	0	0	0	7
s01	1	1	0	0	0	0	0	1	0	0	0	0	1	0	8
s02	1	1	0	0	0	0	0	0	0	0	0	1	0	1	9
no fault	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

TABLE VIII. RESULTS FOR DIFFERENT SCENARIOS

	Test Reduct	Total Costs
0	$\{d\_h1, d\_n1, d\_n2, d\_c01, d\_c03, d\_c04, d\_c05, d\_c06, d\_c07, d\_c08, d\_c09, d\_c10, d\_s1, d\_s2\}$	33
1	$\{d\_c01, d\_c02, d\_c03, d\_c04, d\_c05, d\_c06, d\_c07, d\_c08, d\_h1\}$	9
2	$\{d\_c01, d\_c02, d\_c03, d\_c05, d\_c06, d\_c07, d\_s1, d\_s2\} \cup \{d_{coll2\_}n2\}$	17
3	$\{d\_c01, d\_c02, d\_c03, d\_c05, d\_c06, d\_c07, d\_c08, d\_s1\} \cup \{d_{coll2\_}n2\}$	18
4	$\{d\_n2, d\_c01, d\_c02, d\_c03, d\_c05, d\_c06, d\_c07, d\_c08, d\_s1\}$	18

The evaluation shows that the sequence in which the diagnostic functions appear in the decision table influences the selection of diagnostic content through the TSCQR algorithm significantly. With alphabetically sorted diagnostic functions, the algorithm prefers the diagnostic functions on the component level over diagnostic functions on the state level, although they are equivalent in this scenario. The preference becomes apparent in the second scenario, where the diagnostic functions on the state level are to be favored as they have the same information value with fewer costs. In the third scenario, the diagnostic function d_c08 is added over the diagnostic function d_s2 despite its cost. As it supplies diagnostic input for other diagnostic functions in the same granular level, the Diagnostic Cover Graph interlinks their respective selection. Although adding d_c08 increases the total costs, it is needed to achieve the demanded resolution, which has higher priority. The last scenario's restrictions lead to the selection of the diagnostic function d_n2 of the sub-system level over the diagnostic collection. This selection results from the augmentation of the Diagnostic Cover Graph to represent the impaired coverage of the diagnostic functions d_c05 and d_c07. Note that the added restriction to the executability condition reduces the resolution of the diagnostic content since a malfunction in the disk brake and drum brake during 'park' mode cannot be distinguished with the provided diagnostic information anymore.

V. CONCLUSION AND FUTURE ANALYSIS

A new procedure to automate the selection of diagnostic functions for the design of diagnostic content in a mechatronic system was presented. It uses a granular graph structure to represent diagnostic coverage of a system, considering restrictions on the availability of a function and the executability of a diagnostic function. On its basis, a test-cost sensitive feature selection (e.g. TSCQR algorithm) is implemented to select an optimal set of diagnostic features. Experiments with different example scenarios have shown that the approach can cope with various restrictions and interdependence of diagnostic functions and that the selected diagnostic functions are plausible. Although the example scenarios are relatively trivial, they foreshadow the procedure's capabilities for large and complex systems.

The current scope of the procedure is to represent relations between different granular levels. However, especially when dealing with complex and large systems, one fault can propagate through the system and creates a vast amount of diagnostic information in the process. One potential extension of the procedure is to include relations between functionalities within a granular level into the Diagnostic Cover Graph to obtain a more realistic idea of the actual diagnostic information set being produced in case of a fault.

Another possible extension is to capture not only discrete conditions on the availability of functionalities and executability of diagnostic functions but also continuous restrictions such as a range in temperature or voltage. In case the selection of minimal cost set of diagnostic functions is crucial, the granular graph structure can be used as a basis for other cost-sensitive feature selection methods which might produce better results. These topics will be part of future research.

CONFLICT OF INTEREST

The authors declare no conflict of interest.

AUTHOR CONTRIBUTIONS

Melissa Gresser conducted the research and wrote the paper, Marc David Rabe, Dominik Brehl and Bernhard Bäker supported the work with their experience and technical knowledge and devoted many hours to discussing the research topic. All authors approved the final version.

REFERENCES

[1] C. Angeli, "Diagnostic expert systems: From expert's knowledge to real-time systems," *Advanced Knowledge Based Systems: Model, Applications & Research*, vol. 1, pp. 50-73, Jan. 2010.

[2] N. Lo, J. M. Flaus, and O. Adrot, "Review of machine learning approaches in fault diagnosis applied to IoT systems," in *Proc. International Conference on Control, Automation and Diagnosis*, Grenoble, France, July 2019, pp. 1-6.

[3] R. Bellman, *Adaptive Control Processes*, Princeton Legacy Library, 1961.

[4] C. Aldrich and L. Auret, *Unsupervised Process Monitoring and Fault Diagnosis with Machine Learning Methods*, London: Springer, 2013.

[5] W. Hamscher, "Using structural and functional Information in diagnostic design," in *Proc. the National Conference on Artificial Intelligence*, Washington, D.C., USA, 1983.

[6] R. J. Patton, "Robustness in model-based fault diagnosis: The 1995 situation," *IFAC Proceedings Volumes*, vol. 28, no. 12, pp. 45-67, June 1995.

[7] D. F. Haasl, N. H. Roberts, W. E. Vesely, and F. F. Goldberg, *The Fault Tree Handbook*, Washington, D.C., USA: U.S. Nuclear Regulatory Commission, 1981.

[8] U.S. Department of Defense, *Procedures for Performing Failure Mode, Effects, and Criticality Analysis*, Washington, DC, USA, 1980.

[9] T. Kurtoglu and I. Tumer, "A graph-based fault identification and propagation framework for functional design of complex systems," *Journal of Mechanical Design*, vol. 130, May 2008.

[10] B. O'Halloran, N. Papakonstantinou, K. Giammarco, and D. V. Bossuyt, "A graph theory approach to functional failure propagation in early Complex Cyber-Physical Systems (CCPSs)," in *Proc. INCOSE International Symposium*, July 2017, vol. 27, pp. 1734-1748.

[11] R. Reiter, "A theory of diagnosis from first principles," *Artificial Intelligence,* vol. 32, no. 1, pp. 57-95, 1987.
[12] I. Pill, T. Quaritsch, and F. Wotawa, "From conflicts to diagnoses: An empirical evaluation of minimal hitting set algorithms," in *Proc. 22nd International Workshop on Principles of Diagnosis*, Murnau, Germany, Oct. 2011.
[13] A. Ferone, T. Georgiev, and A. Maratea, "Test-Cost-Sensitive quick reduct," in *Proc. 12th International Workshop on Fuzzy Logic and Applications*, R. Fullér, S. Giove and F. Masulli, Eds., Genoa, Italy, Springer, 2018, vol. 11291, pp. 29-42.
[14] Z. Pawlak, "Rough sets," *International Journal of Computer & Information Sciences*, vol. 11, pp. 341-356, 1982.
[15] Q. Shen and A. Chouchoulas, "Modular approach to generating fuzzy rules with reduced attributes for the monitoring of complex systems," *Engineering Applications of Artificial Intelligence*, vol. 13, pp. 263-278, June 2000.
[16] J. Han, M. Kamber, and J. Pei, "Classification: Advanced methods," in *Data Mining: Concepts and Techniques*, 3rd ed., J. Han, M. Kamber, and J. Pei, Eds., Boston, USA: Morgan Kaufmann, 2012, ch. 9, pp. 393-442.
[17] A. Skowron and C. Rauszer, "The discernibility matrices and functions in information systems," in *Intelligent Decision Support: Handbook of Applications and Advances of the Rough Sets Theory*, R. Słowiński, Ed., Dordrecht: Springer, 1992, vol. 11, pp. 331-362.
[18] V. Schindler and S. Müller, "Kraftfahrzeugtechnik," in *Dubbel - Taschenbuch für den Maschinenbau*, Berlin, Heidelberg: Springer, 2018, ch. Q1.

Melissa Gresser received her B. Sc. in mathematics and business in 2014 and her M. Sc. in mathematics in 2018 from the University of Ulm, Germany. She took part in a double master program in applied mathematics in 2016 at the Florida Institute of Technology, USA. She is currently pursuing her Ph.D. at BMW Group in Munich in collaboration with the Institute of Automotive Technologies at Technical University of Dresden, Germany. Her research interests include graph theory, optimization algorithms and machine learning algorithms.

Dominik Brehl studied electrical engineering and information technology and received his M.Sc. degree (with distinction) focusing on automation, control and systems engineering from the Technical University Ilmenau, Germany in 2011. Subsequently, he joined BMW Group in Munich where he was responsible, among others, for diagnostic and availability concepts of power trains. Since 07/2020 he works on functional safety concepts for new power train strategies.

Marc David Rabe studied mechanical engineering at Technical University Dresden (2012-2018) majoring in automotive engineering. He is a PhD student at the chair of vehicle mechatronics at Technical University Dresden since 10/2018. Focusing on automotive diagnostics, he utilizes machine-learning concepts for data-driven fault detection and diagnosis.

Bernard Bäker has his background in the field of Electrical Engineering with a focus on mobile and immobile energy and information management. He studied at the Technical University Braunschweig (1989-1993) and has written his doctorate dealing with energy efficient vehicle systems in cooperation with the research and development of Volkswagen AG in Wolfsburg (1994-1998). After that he was seven years with Daimler AG in Stuttgart, responsible for advanced vehicle test and diagnosis systems and corresponding production approaches. Since 2005 he is the chair of the vehicle mechatronics department at the Institute of Automotive Technologies Dresden - IAD at the Technical University of Dresden.
From 2015 until 2019 Prof. Bäker was predean and dean of the faculty for traffic and transportation sciences at the TU Dresden.