Generating Fault Trees from Mixed Quantitative and Qualitative Electrical Device Models

Heiko Milde[1] and **Lothar Hotz**[1]

Abstract. Computer diagnosis systems grounded on hand-crafted fault trees are wide-spread in industrial practice. Since the complexity of technical systems increases and innovation cycles get shorter, the need for systematic fault tree generation and maintenance arises. In this paper, the MAD system is introduced which generates fault trees based on models of technical devices. In addition to qualitative device modeling, MAD allows context-dependent quantitative measurement modeling such that tests in fault trees refer to quantitative parameter threshold values which corresponds to usual industrial practice. It is demonstrated that quantitative measurement modeling is essential for accurate fault tree generation. MAD has been successfully evaluated in cooperation with the German forklift manufacturer STILL GmbH Hamburg.

1 INTRODUCTION

More than 100.000 forklifts made by the German company STILL GmbH Hamburg are in daily use all over Europe. In order to reduce forklift downtimes, approximately 1100 STILL service workshop trucks utilize fault tree-based computer diagnosis systems for workshop diagnosis. In case of a malfunction, service technicians attach a computer diagnosis system to a forklift. Then the diagnosis system performs automated testing and it also instructs service technicians to carry out manual tests. Test sequences are specified by fault trees which are diagnostic decision trees allowing fault identification. In STILL fault trees, nodes represent fault sets. Edges are labeled by the tests (involving measurements, observations, display values and error codes) which must be carried out to verify the corresponding child node.

Due to the complexity of the electrical circuits employed in forklifts, fault trees may consist of more than 5000 nodes. When forklift model ranges are modified or new model ranges are released, the central service division manually generates or adapts fault trees. Dealing with this task, service engineers apply detailed expert knowledge concerning faults and their effects. Adapting fault trees to new model ranges can take a service engineer several months. This practice is costly and quality management is difficult. Furthermore, fault trees are not optimized and fault identification cost is unnecessarily high. Hence, there is a need for computer methods to systematically support the design, modification, and optimization of fault trees. The introduction of new diagnosis techniques, however, raises challenges.

First, cost of new diagnosis system integration into current diagnosis and service processes has to be low. Long terms of training for service technicians and service engineers are not acceptable. Second, if a new diagnosis system is established, cost of diagnosis equipment generation, modification, and maintenance has to be low. Third, the current diagnostic performance should be exceeded. Fourth, average fault identification cost should be reduced.

Facing these requirements, innovative model-based techniques seem to be advantageous because they provide a systematic way for design, modification, and optimization of diagnosis equipment. But for STILL, completely replacing fault trees is not an immediate option for economical reasons. Hence, in principle, automatically generating fault trees from device models is a promising strategy. Although the basic concepts of model-based fault tree generation are already described in [1] and [2], for the reader's convenience, we briefly outline the main ideas of the approach in the following.

The first step to model-based fault tree generation is to model a device. This step is supported by component libraries and a device model archive (see Figure 1). Design data and knowledge from the design process (knowledge concerning intended device behavior, expected faults, available measurements) are integrated into the device modeling process. In a second step, behavior predictions are automatically computed from the device model and stored in the so-called fault relation. The third step is to build fault trees from the fault relation. This step is supported by a fault tree archive and a cost model for the tests which can be performed. Fault tree generation can be performed automatically or guided by service know-how, i.e. knowledge concerning preferable fault tree topologies.

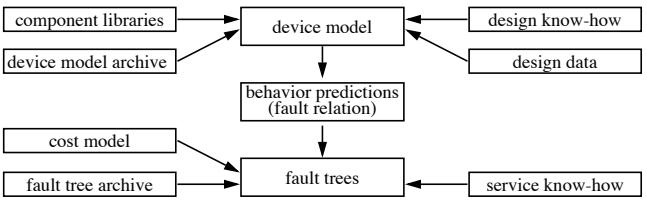

Figure 1. Basic concepts of model-based fault tree generation

In order to realize these concepts, the MAD system (Modeling, Analyzing and Diagnosing) was implemented which is specified in this paper. A more detailed description of MAD can be found in [4]. In industrial practice, in fault trees, component faults are usually described qualitatively. Thus, in principle, for model-based fault tree generation, qualitative system modeling is adequate. To overcome deficiencies of pure qualitative approaches, MAD integrates both qualitative and quantitative system modeling. The latter is realized by the latest extension of MAD, i.e. the so-called context-dependent quantitative measurement modeling which is described in Section 3.3. It is demonstrated that, quantitative measurement mod-

[1] Laboratory for Artificial Intelligence, University of Hamburg, Vogt-Koelln-Str. 30, 22527 Hamburg, Germany, email: milde@informatik.uni-hamburg.de, hotz@informatik.uni-hamburg.de

els are essential for accurate fault tree generation. Due to quantitative measurement models, tests in MAD's fault trees refer to quantitative parameter threshold values which corresponds to usual industrial practice. In Section 2, we introduce the accelerator pedal circuit showing typical characteristics of diagnosis in the forklift application scenario. Section 3 presents device modeling including context-dependent quantitative measurement modeling. Fault tree generation is described in Section 4. Finally, Section 5 summarizes our findings including evaluation and conclusions.

2 THE ACCELERATOR PEDAL CIRCUIT

Figure 2 shows the wiring diagram of the accelerator pedal circuit which enables the electronic control unit (ECU) to measure the accelerator pedal position. The ECU provides a supply voltage VCC whose value is in the interval [9.8V, 10.1V]. The accelerator pedal determines a potentiometer position which controls the value of voltage UFG. In addition, the pedal is connected to a switch controlling voltage UFGS. When the pedal is kicked down the switch 1S16 connects wire 2 and wire 5. In this case, UFG is in the interval [8.3V, 9.4V]. If the pedal is not operated, wire 1 and wire 2 are connected and 4.6V < UFG < 5.3V holds. The ECU measures both UFG and UFGS. Thus, the pedal position is supplied redundantly to secure reliability of the measurements. Note that, in order to cope with limited measurement accuracy and production tolerances as well as ageing and temperature effects, in the STILL application, correct circuit behavior is described by intervals rather than by sharp quantitative parameter values.

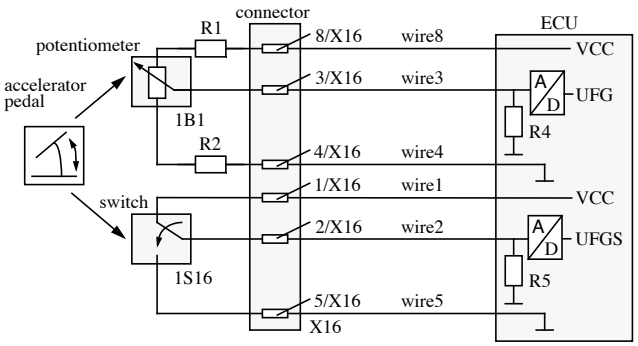

Figure 2. Wiring diagram of a forklift accelerator pedal circuit.

Beside the quantitative value of UFG, the ECU provides two error flags UFG_HIGH and UFG_LOW with values "OK" and "NOT_OK" indicating that UFG exceeds or falls below certain threshold values. The ECU automatically maps the quantitative value of UFGS to two values "OPEN" and "CLOSED". In addition to these automated measurements, for fault identification, service technicians can manually measure the voltage drop from wire 8 at connector X16 to ground.

Fault trees of the current diagnosis system allow to identify the following faults: Wires located between the ECU and connector X16 may break due to mechanical stress. The mechanical connection between the accelerator pedal and the switch 1S16 may also break. The supply voltage VCC may be supplied incorrectly. Additionally, the mechanical connection between accelerator pedal and potentiometer 1B1, may not be adjusted correctly or may even be broken. If a fault occurs, symptoms ranging from slight parameter deviations to total loss of functionality may appear.

3. MODELING AND BEHAVIOR PREDICTION

In our application, qualitative electrical device models are adequate because, in current STILL fault trees, component faults and symptoms are described qualitatively. Additionally, qualitative models are advantageous because, in principle, dealing with product variants is possible. Anyhow, in Section 3.3, it is demonstrated that fault trees based on pure qualitative device models are insufficient for accurate fault identification. To overcome this deficiency, the so-called context-dependent quantitative measurement modeling is introduced. In Section 3.1, MAD's qualitative network analysis is briefly presented. Details of MAD's internal models of electrical circuits and the computation of qualitative parameter values can be found in [3].

3.1 MAD's qualitative network analysis

As known from electrical engineering, MAD represents electrical circuits by equivalent networks. These networks consist of standard component models which show no internal structure but they show well-defined and idealized behavior. Controlled versions of standard component models exist. MAD only provides two different types of standard component models, i.e. passive and active models showing passive and active behavior modes, respectively. Passive behavior modes are "consumer", "insulator", and "conductor". Active models qualitatively represent idealized voltage sources providing different voltage levels. Standard component models can be connected in combinations of series, parallel, star and delta (triangular) groupings. This simple internal representation of electrical circuits is sufficient for the following reasons.

- In STILL service workshops, only steady-state diagnosis of electrical circuits is performed. Therefore, only steady-state behavior of physical components has to be represented in component models. In particular, an explicit representation of temporal dependencies is not necessary.
- A small number of qualitative standard component models suffices, because, often, different physical components show similar electrical behavior, i.e. their current/voltage characteristics differ only slightly. Qualitative versions of these current/voltage characteristics are frequently identical.
- MAD's standard component models are deliberately selected so that important behavior classes of the application domain can be adequately represented.

Due to analogies between electrics, mechanics and hydraulics, MAD's internal circuit models are, in principle, also adequate for other technical domains.

Physical parameters are described by qualitative values standing for intervals or landmarks. To facilitate the analysis of faulty device behavior, MAD's parameter representation is three-valued. That is, actual parameter values, reference values, and parameter deviations are explicitly represented. Only signs of parameter **deviations** are represented, i.e. "-", "0", and "+" are MAD's qualitative deviation values. As an example, Figure 3 shows MAD's qualitative **absolute** voltage values and their semantics. These values are utilized to characterize actual and reference values of voltage. In MAD's internal circuit models, there are no quantitative parameter values represented but the value *0*. Note that, *negative_landmark = - positive_landmark* holds. The meaning of these landmarks is further specified by their utilization in the modeling process. For instance, the value of the supply voltage VCC of the accelerator pedal circuit is modeled by "positive_landmark". Thus, for all voltage in the de-

vice model, “positive-landmark” represents the value of VCC.

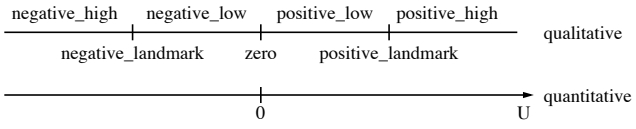

Figure 3. Qualitative and quantitative absolute voltage values

In order to secure high quality of generated fault trees MAD’s qualitative calculus shows certain features to improve the accuracy of circuit behavior prediction. In the following, two of these features are briefly summarized.

- First, rather than relying on qualitative versions of basic arithmetics, MAD computes qualitative values for current and voltage by a set of qualitative operators which are qualitative versions of complex quantitative equations. In principle, for network analysis, a limited number of operators suffices because MAD's internal representation of electrical circuits offers a limited number of standard component models and elementary network structures. Operators are represented by a set of tables comprising more than 30.000 entries which had to be generated by computer in order to secure reliability.
- Second, for computation of qualitative values, MAD utilizes a set of equations which would be redundant if quantitative values were used. It can be shown that if qualitative values are computed, MAD's set of equations is not redundant but sharpens qualitative behavior predictions.

3.2 COMEDI (COmponent Modeling EDItor)

To improve acceptance among engineers, MAD provides a user interface called COMEDI which is similar to a CAD tool (see Figure 6). Figure 4 describes the modeling process which is based on predefined models from three different libraries. Providing library models is fundamental because utilization of libraries massively reduces the complexity of the modeling process which is essential for the acceptance of MAD in the application.

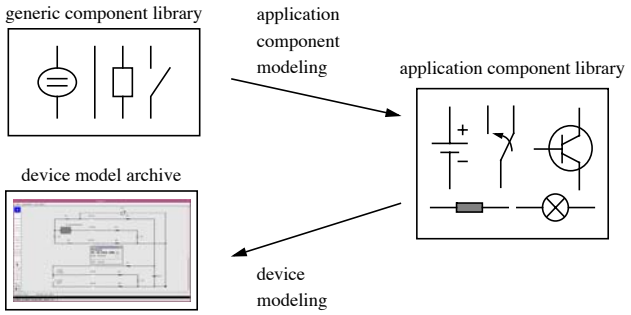

Figure 4. COMEDI libraries and the device modeling process

Using MAD, device modeling is a two step process. In the first step, application component class models are build from generic component class models which are MAD’s modeling primitives. In the second step, device models are assembled from application component class models. By this means, complexity of the modeling process is reduced and modeling is facilitated because if an application component class model (e.g. logical AND gate model) is once defined it can be massively reused for device modeling. In the following, the libraries and the modeling process are briefly described.

Generic component class models show both correct and faulty behavior modes. To COMEDI users, these behavior modes are described in colloquial language similar to engineers thinking of how components work. For example, a certain generic component class model shows two behavior modes, i.e. “ok_consumer” and “fault_insulator”. Internally, behavior modes are represented by MAD’s three-valued qualitative parameter description. 728 different generic component class models exist which are generated by the combinatorics of correct and faulty behavior modes. All these models are provided by the generic component library which cannot be extended. COMEDI completely hides MAD’s internal parameter representation from users to facilitate the modeling process.

The application component library contains models of component and subcircuit classes occurring in a certain application. To build a certain application component model, structure and behavior of the model have to be determined. The model structure is generated by assembling generic component class models on the screen. Model behavior is implicitly given by the model structure and the behavior of generic component class models. Additionally, causal parameter dependencies can be represented in behavior tables based on user-defined qualitative parameter values. By this means, generation of abstract qualitative models for complex components or subsystems such as logical circuits and software controlled components is facilitated. For accelerator pedal circuit modeling, only 11 different application component class models are required. Since other circuits were also modeled, up to now, the application component library consists of about 50 models.

The device model archive allows systematic reuse and modification of device models that were created during former modeling sessions. These models are assemblies of application component models. That is, for device modeling a structure description is created on the screen. Device behavior modes are interactively defined by combinations of application component behavior modes. Measurements and their costs can be determined. Multiple faults to appear in fault trees can also be defined. Additionally, context-dependent quantitative measurement models can be individually defined which is described in the following.

3.3 Context-dependent quantitative measurement modeling

Usually, if physical parameters are measured, sharp quantitative values are obtained. In order to utilize results of measurements for fault identification together with pure qualitative circuit models, quantitative parameter values have to be mapped to qualitative values. This mapping seems to be simple if qualitative values show well-defined quantitative semantics, e.g. semantics of sign-based qualitative parameter values “-”, “0”, and “+” is obvious. Surprisingly, strictly mapping quantitative measurement values to qualitative values according to their semantics can lead to spurious fault identifications. This can be demonstrated considering voltage UFG of the accelerator pedal circuit. It is assumed that the pedal is not kicked down.

According to MAD’s qualitative accelerator pedal circuit model, in the faultless state, the actual value of UFG is “positive_low”. If wire 8 is broken, the actual value of UFG is “zero”. In practice, if wire 8 is broken, the circuit part downstream from wire 8 is not connected with any source and it is connected with ground only once. In such circuit topologies, slight physical phenomena occur, such that voltage drops may be around 0V. Thus, if wire 8 is broken, 0.05V

may be measured for UFG. Strictly following semantics of qualitative values, if 0.05V is measured, “positive_low” holds for UFG and “zero” is not valid. Thus, according to the measurement and the qualitative circuit model, correct circuit behavior is confirmed and faulty behavior is ruled out although, in practice, wire 8 is broken. For correct fault identification, it must be taken into account that all UFG values between -0.05V and +0.1V may be instances of the same fault symptom and, thus, these values have to be treated identical. Hence, for adequate qualitative modeling of UFG, the interval [-0.05V, 0.1V] should be explicitly represented. Obviously, MAD’s internal qualitative parameter values cannot deal with this task because semantics of landmarks is predefined.

In general, as described in [6], significant parameter value distinctions are context-dependent and, thus, for adequate qualitative parameter modeling, predefined landmarks are inadequate. That is, landmarks have to be individually defined. In order to deal with this challenge, MAD allows context-dependent quantitative measurement modeling. For each measurement of the device model, COMEDI users can individually specify significant landmarks bringing in expert knowledge concerning nominal circuit behavior and domain specific typical fault symptoms. Note that, typical fault symptoms are not known from simulation but from service and diagnosis practice. Nominal circuit behavior is usually known from the design phase.

User-defined landmarks divide the quantitative parameter value space into intervals. For fault identification, all quantitative parameter values located in the same interval are treated identical. One interval can be marked as reference interval, i.e. in the faultless state, the parameter value is expected to be in this interval. Landmarks must be given by sharp quantitative parameter values. They are defined with respect to MAD’s internal qualitative absolute parameter values only. For quantitative measurement modeling, MAD’s internal qualitative deviation values do not have to be considered. The quantitative counterparts of MAD’s three-valued qualitative parameter descriptions can be computed from definitions of quantitative landmarks and the reference interval. Note that, quantitative measurement models hold with respect to a certain device operating mode. That is, landmarks represent a certain context. In the following, considering voltage UFG of the accelerator pedal circuit, quantitative measurement modeling is demonstrated assuming that the accelerator pedal is not kicked down.

In the forklift application, service and diagnosis practice has shown that a voltage measurement in the interval [-0.05V, 0.1V] may indicate that the connection to source is broken or the source does not provide any voltage drop at all. A voltage measurement in the interval [9.8V, 10.1V] indicates that, possibly, there is a short circuit to the source. From the design phase, it is known that 4.6V < UFG < 5.3V holds in the faultless state. Thus, reasonable landmarks for the UFG measurements are -0.05V, 0.1V, 4.6V, 5.3V, 9.8V, and 10.1V. Note that, these landmarks are not derived from complex simulation but they are grounded on service, diagnosis, and design expertise.

In order to model significant distinctions concerning UFG, COMEDI users can partition the quantitative value space of voltage as presented in Figure 5. The interval [4.6V, 5.3V] can be explicitly marked as reference behavior. Since the supply voltage VCC is between 9.8V and 10.1V and, in the supply voltage model, MAD’s internal qualitative value “positive_landmark” is utilized to represent the value of VCC, in Figure 5, “positive_landmark” also has to be located between 9.8V and 10.1V.

Table 1 shows some examples of MAD’s three-valued qualitative voltage representation and their quantitative counterparts which are computed from the measurement model shown in Figure 5. For example, MAD’s qualitative voltage value triple (positive_low, positive_low, -) (see shaded row in Table 1) corresponds to all quantitative values which are larger than the smallest quantitative value corresponding to qualitative “zero” and smaller than the largest quantitative confirmation of the reference behavior. That is, (positive_low, positive_low, -) is mapped to (-0.05, 5.3).

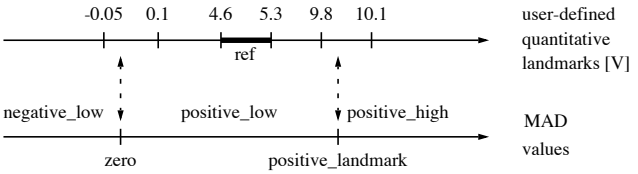

Figure 5. User-defined quantitative measurement model

Table 1. MAD’s internal voltage values and corresponding intervals

internal actual value	internal reference value	internal deviation	quantitative UFG model [V]
negative-low	positive_low	-	[-10.1, 0.1]
zero	positive_low	-	[-0.05, 0.1]
positive_low	positive_low	-	(-0.05, 5.3)
positive_low	positive_low	0	[4.6, 5.3]
positive_low	positive_low	+	(4.6, 10.1)
positive_landmark	positive_low	+	[9.8, 10.1]

Considering the UFG measurement of 0.05V again, according to Table 1, among other qualitative actual values, “zero” is confirmed and, thus, according to MAD’s qualitative device model, wire 8 is possibly broken which is a correct fault identification. This example demonstrates that, due to context-dependent quantitative measurement modeling, accurate fault identification is possible.

3.4 Modeling the accelerator pedal circuit

In Figure 6, the COMEDI device model of the accelerator pedal circuit is presented. The device operating mode “STANDARD” is modeled which means that the switch 1S16 is in the position shown in Figure 2, i.e. the pedal is not kicked down. Correct and faulty behavior of wire 4 is presented in the small window in the center of Figure 6.

To model the error flags UFG_LOW, UFG_HIGH, and UFGS, functional labeling [5] is utilized, i.e. strings such as “OPEN”, “CLOSED”, “OK”, and “NOT_OK” can be attached to MAD’s qualitative parameter values. Note that, these strings occur in the fault relation shown in Figure 7. The manual voltage measurement UG described in Section 2 is modeled by a special multimeter component model. There are context-dependent quantitative measurement models for UFG and UG. Thus, for these parameters, quantitative intervals occur in the fault relation.

Automated behavior predictions are performed for all possible component behavior (correct and faulty) and all device operating modes considered in the accelerator pedal circuit device model. Figure 7 shows fault symptom associations which hold in the device operating mode “STANDARD”.

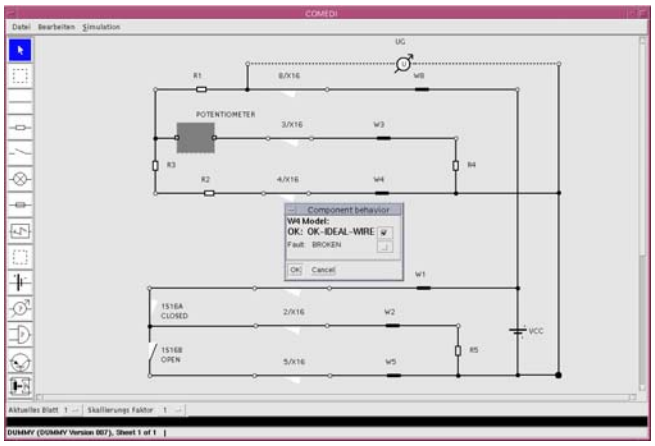

Figure 6. COMEDI model of accelerator pedal circuit

Fault Relation Frame

Behavior modes	Tests: UFG_LOW_STANDARD	UG_STANDARD	UFG_STANDARD	UFG_HIGH_STANDARD	UFGS_STANDARD
1S16A-is_STUCK-AT-OPEN	OK	[9.8 10.1]	[4.6 5.3]	OK	CLOSED
W5-is_BROKEN-WIRE	OK	[9.8 10.1]	[4.6 5.3]	OK	OPEN
W2-is_BROKEN-WIRE	OK	[9.8 10.1]	[4.6 5.3]	OK	CLOSED
W1-is_BROKEN-WIRE	OK	[9.8 10.1]	[4.6 5.3]	OK	CLOSED
W8-is_BROKEN-WIRE	NOT_OK	[-0.05 0.1]	[-0.05 0.1]	OK	OPEN
W4-is_BROKEN-WIRE	OK	[9.8 10.1]	[4.6 10.1]	NOT_OK	OPEN
W3-is_BROKEN-WIRE	NOT_OK	[9.8 10.1]	[-0.05 0.1]	OK	OPEN
PEDAL-is_DEFECT	NOT_OK	[9.8 10.1]	[-0.05 0.1]	OK	OPEN
PEDAL-is_NOT-ADJUSTED	NOT_OK	[9.8 10.1]	[-0.05 5.3]	OK	OPEN
VCC-is_LOW-VOLTAGE	NOT_OK	[-0.05 10.1]	[-0.05 5.3]	OK	OPEN
VCC-is_HIGH-VOLTAGE	OK	[9.8 INF]	[4.6 10.1]	NOT_OK	OPEN
VCC-is_NO-VOLTAGE	NOT_OK	[-0.05 0.1]	[-0.05 0.1]	OK	CLOSED
OK-CASE	OK	[9.8 10.1]	[4.6 5.3]	OK	OPEN

Figure 7. Parts of fault relation of the accelerator pedal circuit

4. FAULT TREE GENERATION

MAD offers three different possibilities to generate fault trees. First, based on fault relations, fault trees can be created automatically. Second, fault trees from archives can be reused. Third, in order to permit manual adoption and modification of fault trees, MAD offers basic editing operations, such as moving a certain fault from one fault set to another and recomputing the corresponding tests. In the following, automated fault tree generation is presented in more detail. Since MAD generates fault trees based on fault relations, alternative device operating modes are considered because fault relations contain fault symptom associations for all operating modes defined in a device model. For automated fault tree generation, one can choose from the following criteria to guide fault tree generation.

- **Grouping by symptoms.** Fault trees are generated such that subsets of faults correspond to a prespecified symptom. For instance, all faults are grouped together which lead to an unexpected value of the accelerator pedal voltage UFG.
- **Grouping by aggregate structure.** If the aggregate structure of the device is known, fault trees can be generated such that subsets of faults correspond to the same physical component. For instance, faults occurring in the ECU may be grouped together.
- **Minimization of average diagnosis cost.** Automated fault tree generation uses the well-known A*-algorithm to select the tests which minimize the average fault identification cost according to a cost model.

Since minimization of average diagnosis cost is one of the basic requirements for the acceptance of MAD in the STILL application, in the following, cost optimized fault tree generation is described. Fault probabilities are not taken into account because, in the STILL application, these probabilities are not available. A promising approach for fault tree generation considering fault probabilities is presented in [2]. Figure 8 shows parts of a cost optimized fault tree of the accelerator pedal circuit.

Cost optimized fault tree generation is based on the A*-algorithm which performs a search in a state space. A state contains a set of fault sets which are the current leaves of a growing fault tree. The start state consists of one fault set containing all n faults of the fault relation. The goal state consists of fault sets containing faults which cannot be discriminated. A successor of a state is generated by partitioning one leaf fault set into at least two subsets by selecting a partitioning test T. All successors of a state are generated by applying each partitioning test to each leaf. Each path in the state space represents a possible fault tree.

A test has corresponding costs and a set of different possible test results, i.e. a test domain. For instance, the seven quantitative intervals presented in Figure 5 are the domain of UFG if the accelerator pedal is not kicked down. In general, test are not exclusive, i.e. if a test partitions a fault set into subsets, certain faults can occur in more than one subset.

Each state is evaluated by the functions g and h. g is defined as the sum of the diagnostic effort for each fault f. The diagnostic effort of a fault f is the sum of all test cost $C(T)$ on the path between the current leaf fault set containing f and the root fault set. To guide the search, the heuristic function h estimates cost of fault identification assuming that, in the fault identification process, a certain state is already reached. In Figure 9, the definitions of $g()$ and $h()$ are given.

To demonstrate that h never overestimates real cost of fault identification in a cost-optimized tree, a certain leaf b is considered. There is a set of available tests T_i not yet used on the path between b and the root. These test allow the generation of a cost-optimized subtree below b. Available tests show costs c_i and domains. $kmax$ is the maximum size of these test domains.

The following two properties guarantee that the heuristic function h underestimates fault identification cost in a cost-optimized subtree. First, h considers an impossible subtree in which fault identification is cheaper than in a cost-optimized subtree. Second, rather than precisely computing fault identification cost in the impossible subtree, h underestimates diagnosis cost.

The impossible subtree and the cost-optimized subtree show the same faults, but, in the impossible subtree, the following characteristics secure that fault identification in the impossible subtree is cheaper than in the cost-optimized subtree.

First, in the impossible subtree, all available test are exclusive. That is, if a test partitions a fault set, each fault occurs in only one subset. Second, all available tests split fault sets into $kmax$ subsets. Due to these two properties, in the impossible subtree, the discriminating power of available tests is higher than in the cost optimized subtree. Third, in the impossible subtree, the test first performed for fault identification, is as expensive as the cheapest available test of the cost-optimized subtree. All tests performed in the second level of the impossible subtree are as expensive as the second cheapest test of the cost-optimized subtree, and so on. Fourth, in the impossible subtree, if a fault set is partitioned into subsets, all of these subsets contain the same number of faults, i.e. the impossible subtree is balanced. Provided the first three properties hold, a balanced tree yields to lowest fault identification cost.

For the estimation of fault identification cost in the impossible subtree, it is assumed that the depth of the subtree is $\lfloor \log_{kmax}(|b|) \rfloor$, ($\lfloor \ldots \rfloor$ is the floor operation) which, in general, is

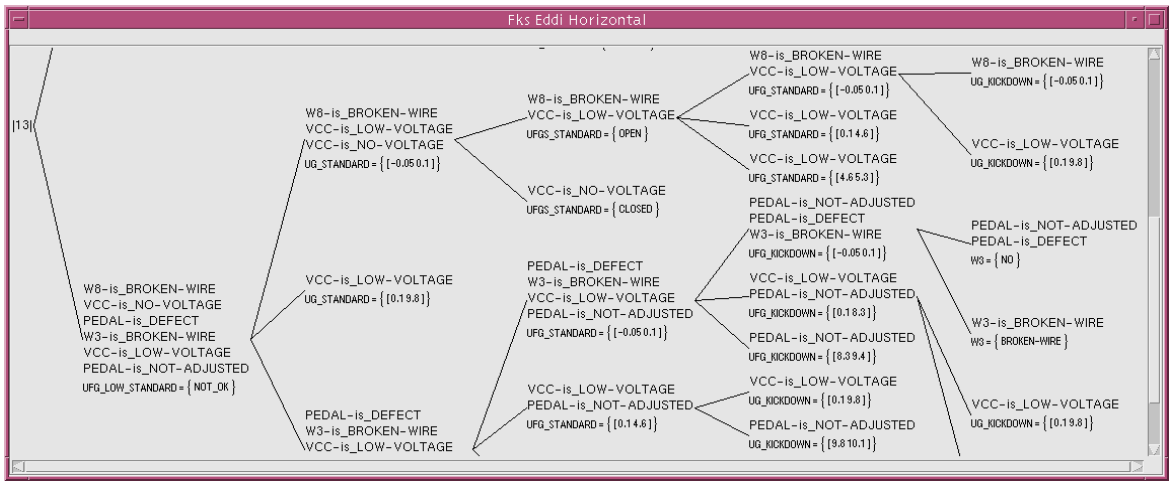

Figure 8. Parts of cost optimized fault tree of accelerator pedal circuit

an underestimation. Based on this assumption, for each fault in *b*, cost of fault identification in the impossible subtree can be underestimated as:

$$\sum_{i=1}^{\lfloor \log_{kmax}(|b|) \rfloor} C(T_i), \qquad \text{with } T_i \text{ is i-th cheapest unused Test T}$$

In the fault tree, in Figure 8, the final row of a fault set is the corresponding test. The root contains 13 faults which are not explicitly shown. Note that, in MAD fault trees, faults are sometimes discriminated by so-called direct fault identifications such as "w3 = {BROKEN-WIRE}" (see Figure 8). Direct fault identifications are described in detail in [4].

$$g(state) = \sum_{i=1}^{n} de(f_i), \quad with\ de(f_i) = \sum_{T \in path(f_i)} C(T)$$

$$h(state) = \sum_{b \in leaves(state)} h(b), \ with$$

$$h(b) = |b| \cdot \sum_{i=1}^{\lfloor \log_{kmax}(|b|) \rfloor} C(T_i), \qquad \text{with } T_i \text{ is i-th cheapest unused Test T}$$

Figure 9. Cost function *g* and heuristic function *h*

5 EVALUATION AND CONCLUSIONS

The prototypical implementation of MAD allows model-based behavior prediction and automatic generation as well as manual modification of fault trees. All faults considered in the device model occur in the generated fault tree, and tests are selected correctly to discriminate fault sets. This holds even when fault trees are modified manually. Furthermore, average diagnosis cost is minimal within the constraints imposed by a prespecified fault tree structure. Fault trees have been successfully integrated into existing STILL diagnosis systems. It has been found, that due to the development of MAD's context-dependent quantitative measurement modeling, accuracy of behavior predictions and fault trees is significantly improved.

MAD has been evaluated in the STILL application scenario and it has been found that using the modeling techniques of MAD with some extensions regarding the implementation of certain network analysis concepts, more than 90% of the faults of the current handcrafted diagnosis system can be handled successfully. In some cases, since component-dependent parameter threshold values are not explicitly represented in MAD models, in fault trees, correct and faulty behavior cannot be distinguished definitely. Using MAD, an accelerator pedal fault tree was automatically generated from the circuit model and imported into the STILL diagnosis system. STILL service experts found that this fault tree can be used for fault identification.

ACKNOWLEDGEMENTS

This research has been supported by the Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie (BMBF) under the grant 01 IN 509 D 0, INDIA - Intelligente Diagnose in der Anwendung.

REFERENCES

[1] Cascio, F., Console, L., Guagliumi, M., Osella, M., Panati, A., Sottano, S., Theseider Dupré, D.: *On-board diagnosis of automotive systems: from dynamic qualitative diagnosis to decision trees*, IJCAI-99, Workshop on Qualitative Reasoning for Complex Systems and their Control,1999.

[2] Faure, P.-P., Trave-Massuyes, L., Poulard, H.: *An Interval Model-Based Approach for Optimal Diagnosis Tree Generation*, in: Proc. DX-99, 10th International Workshop on Principles of Diagnosis, 1999.

[3] Milde, H., Hotz, L., Kahl, J., Wessel, M.: *Qualitative Analysis of Electrical Circuits for Computer-based Diagnostic Decision Tree Generation*, in: Proc. DX-99, 10th Int. Workshop on Principles of Diagnosis, 1999.

[4] Milde, H., Hotz, L.: *Facing Diagnosis Reality - Model-Based Fault Tree Generation in Industrial Application*, in: Proc. DX-00, The Eleventh International Workshop on Principles of Diagnosis, 2000.

[5] Price, C., Pugh, D.: *Interpreting Simulation with Functional Labels*, in: Proc. QR'96, 10th International Workshop on Qualitative Reasoning about Physical Systems, 1996.

[6] Struss, P., Sachenbacher, M.: *Significant Distinctions Only - Context-dependent Automated Qualitative Modeling*, in: Proc. DX-99, 10th International Workshop on Principles of Diagnosis, 1999.