From Function to Failure

Hamid Jahanian
Macquarie University, Sydney, Australia
hamid.jahanian@hdr.mq.edu.au

Abstract. Failure Mode Reasoning (FMR) is a method for formal analysis of system-related faults. The method was originally developed for identifying failure modes of safety-critical systems based on an analysis of their programs. In this paper, we generalize the method and present a mathematical framework for its use in model-based system and safety analyses. We explain the concepts, formalize the method, formulate models for example systems, and discuss the practical application of the method.

Keywords: Failure Mode Reasoning · FMR.

1 Introduction

The term ‘failure’ is defined as the state of not functioning\(^1\). This, however, does not mean that we can somehow negate the definition of a system function to derive its failure model. Functional and failure models are of two different natures and dealt with in two different spaces.

Failure Mode Reasoning (FMR) was an attempt to bridge the two spaces. By implementing an automated reasoning process, FMR derives the failure model of a system from the definition of its functional behavior. The method was originally designed for identifying failure modes of Safety Instrumented Systems (SIS\(^2\)) based on an analysis of their software programs \[16,20,18,17\]. However, the application of FMR is not limited to SIS programs; any system comprising interrelated components, where the component functions and the interactions between them are defined, can be modeled and analyzed in FMR.

Various other methods exist for system failure analysis, including the conventional methods, such as Fault Tree Analysis (FTA) \[32,31\] and Failure Mode and Effect Analysis (FMEA) \[11\], and the more advanced methods, such as HiP-HOPS \[28,27,26\] and Model-Based Safety Analysis (MBSA) \[22,21\]. FMR differs from the majority of these methods in two main aspects: a) FMR does not require a separate failure model and it extracts it directly from the function model; and b) FMR does not rely on analyst’s perception of failure behavior.

\(^1\) Failure is the termination of the ability of a functional unit to provide a required function or operation of a functional unit in any way other than as required \[12\].

\(^2\) In the process industry, SIS is a system that performs safety functions to protect the plant from major hazards \[13\].
because it uses an automated process to create the model. These two qualities are particularly important in complex systems, for which by-hand analysis of interrelations between numerous components is not feasible.

In this paper, we introduce FMR in its general form and elaborate on its mathematical principles. Our goal is to provide a framework for applying FMR in generic model-based system and safety analysis. The rest of this paper is organized as follows: we begin with the basic definitions and assumptions used in FMR in Section 2. We then explain the FMR analysis process, including abstraction, causation, and composition, in Section 3. We use a small worked example for demonstration in Sections 2 and 3. In Section 4, we expand the discussion by providing some general modeling rules and deriving FMR models for common system structures DNF, CNF, and KooN. In Section 5, we discuss three special topics related to FMR: modeling function loops, measuring the impact of parametric changes, and the application of FMR. In Section 6, we cover a comparison between FMR and other relevant methods.

2 The context

A system is a combination of interacting components organized to achieve a purpose [13]. Thus, a system can be modeled by a set of interconnected functions, where each individual function models the functional behavior of a corresponding component, and the interconnections between the functions represent the interactions between the components.

For a small example, consider a system that is meant to read a real-valued input and decide whether the input value is outside a permissible range. Fig. 1 shows one possible representation for this system. The model comprises three component functions: \( L_{\text{com}} \) for less comparison; \( G_{\text{com}} \) for greater comparison, and \( \text{Or} \) for Boolean disjunction. The system output will be \( y = t \) if \( x < p_1 \) or \( x > p_2 \), and will be \( y = f \) if \( p_1 \leq x \leq p_2 \).

![Fig. 1: A small system comprising three components](image)

In FMR, component functions are defined in mathematical terms. So the resultant system function will be a mathematical function too, except that a
system function is (typically) much more complex than its constituting component functions.

**Definition 1.** Let \( V \) be a state space. A system (or component) function is a function of type \( f: V \rightarrow V \) that defines the functional behavior of a system (or component).

The result (output) of function \( f \) depends on its arguments (inputs), the architecture of \( f \) itself, and, possibly, a number of ‘parameters.’ A parameter can be a control setting that is adjusted by system designers to influence the behavior of the component (e.g., \( p_1 \) and \( p_2 \) in Fig. [1]) or a modeling parameter that represents an aspect of a component (e.g., the travel time of a valve). We express function \( f \) as follows:

\[
y = f(x, p)
\]

where \( y, x, p \in V \) are the outputs, inputs, and parameters, respectively.\(^5\)

We said earlier that failure is the state of not functioning. An advantage of defining system functions in mathematical terms is that we can define their failure in numerical form too.

**Definition 2.** Given the system function \( y = f(x, p) \), a failure at output \( y \) occurs when the ‘reported’ value at \( y \) differs from the ‘intended’ value at \( y \).

Failure is the characteristic of a system output. For a more generic expression of deviation between the reported and intended values at an arbitrary system variable, we use the term ‘fault.’

**Definition 3.** Let \( x \) be a system variable, \( x \) the reported value at \( x \) and \( x' \) the intended value at \( x \). We say \( x \) is faulty if \( x \neq x' \), and we say \( x \) is fault-free if \( x = x' \).

In simple terms, given a system function \( y = f(x, p) \) and a failure at output \( y \), FMR identifies the triggering faults at inputs \( x \) and/or parameters \( p \). A key assumption here is that the function model \( f \) is correct. This is particularly important because the reasoning process of FMR is entirely based on the structure of this function. Thus, in FMR:

**Assumption 1** The only possible causes for a system output failure are faulty inputs and parameters.

An output failure can present itself in the form of an ‘omission’ or a ‘commission.’ Omissions occur when the output of a system does not change state when it is expected to, and commissions occur when the system output changes state when it is not expected to. This obviously implies the existence of an intended

---

\(^5\) We use bold font for vectors and matrices. Also, note that we only separate between \( x \) and \( p \) for better clarity in our failure discussions. Mathematically speaking, \( x \) and \( p \) together represent one vector of arguments for \( f \).
state prior to the failure. We may as well say that omissions occur when the system output does not change to an intended state and commissions occur when the system output changes state to an unintended state. An obvious assumption in both cases is that the system is in an intended state prior to the occurrence of a failure.

**Assumption 2** The inputs and outputs of a system are in an intended state prior to a failure being observed at its output.

With Assumptions 1 and 2 together, we can be more specific about causes of an output failure:

**Assumption 3** An output failure can only occur following one of the following events:

(i) an external fault occurs at the input and propagates through the system, or
(ii) some system parameters are faulty when an intentional change occurs at the input.

Scenario (i) represents a situation in which the system is in normal operation and an external fault occurs at its input. Scenario (ii) shows a test setup, where intentional inputs are injected at the system input for validation testing, for instance.

3 The analysis process

FMR is a hierarchical analysis at three levels: variables, component functions, and system function. The method uses three techniques that correspond these hierarchical levels: abstraction for variables, causation reasoning for component functions, and logical composition for system functions. In this section, we elaborate on these principles. For demonstration, we use the small system depicted in Fig. 1.

3.1 Abstraction and variables

In a general sense, abstraction is the process of mapping a representation of a problem onto a new representation, which helps solve the original problem by only preserving certain desirable properties [10]. In FMR, variable faults are abstracted into ‘failure modes.’

**Definition 4.** A failure mode is a member of $\mathcal{P} = \{h, t, f, m\}$ and it depicts the type of fault at a system variable. Let $x$ be a variable and $x$ and $x'$ its reported

---

6 Readers can find more elaborate examples in [16, 20, 17].
and intended values, respectively. We denote the failure mode at \( x \) by \( \hat{x} \in \mathcal{P} \), where \( \hat{x} \) is generated by \( \text{md}_P(x, x') \), as follows:

\[
\text{md}_P(x, x') \equiv \hat{x} :=
\begin{cases}
    h, & \text{if } x, x' \in \mathbb{R} \text{ and } x > x' \\
    l, & \text{if } x, x' \in \mathbb{R} \text{ and } x < x' \\
    t, & \text{if } x, x' \in \mathbb{B} \text{ and } x = t, x' = t \\
    f, & \text{if } x, x' \in \mathbb{B} \text{ and } x = f, x' = t \\
    m, & \text{if } x = x'
\end{cases}
\] (2)

As an example, \( \hat{x} = h \) means that \( x \) is reporting a higher value than it should (i.e., the reported value at \( x \) is higher than the intended value).

Note that \( m \) is not a fault, and we only define it for the completeness of \( \text{md}_P \). We may also use \( a \) to indicate any failure mode:

\[
(\hat{x} = a) \equiv (\hat{x} = h \lor \hat{x} = m \lor \hat{x} = l), \quad \text{for } x \in \mathbb{R}.
\] (3)

\[
(\hat{x} = a) \equiv (\hat{x} = t \lor \hat{x} = m \lor \hat{x} = f), \quad \text{for } x \in \mathbb{B}.
\] (4)

By using failure mode abstraction, unnecessary numerical calculations are eliminated from the analysis and the problem is reduced to failure mode calculation. As an example, all the combinations of \( x, x' \in \mathbb{R} \) where \( x > x' \) are summarized into one failure mode \( \hat{x} = h \). What we need to do then is to compute with few simple failure modes instead of a wide range of numerical values.

Note that the abstraction used in FMR is not on concrete values or concrete deviations but rather on the deviation between a known, certain value and an unknown, probable value — both of them related to the same system variable. Hence, FMR is a non-deterministic analysis, and failure modes represent uncertain events.

**Types of variables:** From the FMR perspective, a variable can be either ‘certain’ or ‘suspicious.’ Suspicious variables are those in which we search for faults (i.e., their current values may differ from their intended values). Certain variables, on the other hand, are those that are given (i.e., their current values are as intended and, thus, they cannot be faulty).

Recall the \( G_{\text{com}} \) function from our example in Fig. 1: \( z_2 = G_{\text{com}}(x, p_2) := (x > p_2) \). This function compares its input \( x \) to a threshold parameter \( p_2 \) and produces \( z_2 = t \) if \( x > p_2 \). In general, \( \hat{z}_2 = t \) (i.e., when the expected value at \( z_2 \) is \( f \) but \( z_2 \) is reporting \( t \)) implies that either \( \hat{x} = h \) or \( \hat{p}_2 = l \). Now, if we know that the input has changed by intention (i.e., if \( x \) is a ‘certain’ variable) we can rule out \( \hat{x} = h \) and consider \( \hat{p}_2 = l \) as the only possible cause for \( \hat{z}_2 = t \).

### 3.2 Causation and reasoning

Where a functional relationship between two variables \( x \) and \( y \) is known, we can also determine the causal relationship between their corresponding failure mode.
variables $\hat{x}$ and $\hat{y}$. Consider the Boolean negation function $y = \text{Not}(x) = \neg x$. From this functional definition, we can follow that a fault $t$ (or $f$) at input $x$ will cause an opposite fault $\bar{f}$ (or $\bar{t}$) at output $y$. If a failure at output $y$ is given, we can also reason in backward direction: $\hat{y} = t \Rightarrow \hat{x} = f$ and $\hat{y} = f \Rightarrow \hat{x} = t$.

Notice that these implication statements are only valid in the context of the original function $\text{Not}$. To include the function names in our failure models, we use the following form of triples, originally introduced in [18]:

\[
\begin{align*}
\{ \hat{x} = f \} & \quad \text{Not} \quad \{ \hat{y} = t \} \\
\{ \hat{x} = t \} & \quad \text{Not} \quad \{ \hat{y} = f \}
\end{align*}
\]

In these triples, the left-hand side sentences indicate the cause of failure at the input of the function, and the sentences on the right show the effect of the failure at the output.

In a rather general term, we refer to $\{ \phi_c \} f \{ \phi_e \land \phi_p \}$ as a ‘failure scenario,’ in which $\phi_c$ is the conclusion we can make about input failure modes based on $\phi_e \land \phi_p$ as a premise. The latter part itself comprises $\phi_e$ as the output failure mode and $\phi_p$ as any additional context information given. Hence, this expression can be read as follows: given the premise $\phi_p$, if $\phi_e$ occurs at the output of $f$, the cause must be the occurrence of $\phi_c$ at the input of $f$.

Regardless of depictions and symbols, FMR is a logical reasoning process. Reasoning is the process of developing certainty on what we know less about, based on the certainty of what we know more about. Reasoning, in general, may be divided into deductive, inductive, and abductive forms [29]. FMR employs deductive reasoning. The main question we try to answer in FMR is: Given what we know about a failure at the output of a known function, what can we say about faults at its input? The minimum information required to begin an FMR analysis is a given failure mode at the output. In some scenarios, additional information about the current values of variables may be given, which can help narrow down the scope of the search for faulty variables.

From this perspective, we classify the types of reasoning into two main categories:

(i) value-independent, where we only use the failure modes defined in [2].
(ii) value-dependent, where we use failure modes as well as some additional information related to the current values of some of the variables.

Selecting the right type of reasoning obviously depends on the information available prior to the analysis. Nonetheless, the stronger the prior knowledge, the more precise the final results. Take a multiplication function as an example: $y = Mul(x, p) := x \times p$ where $x$ is a suspicious input and $p$ a certain parameter, and $x, p, y \in \mathbb{R}$. Suppose the given output failure mode is $\hat{y} = h$ and we are only studying faulty inputs. Although $p$ is certain and cannot be faulty, its sign still affects the transition of fault from $x$ to $y$:

\[
\begin{align*}
(\hat{y} = h \land p > 0) & \Rightarrow (\hat{x} = h) \\
(\hat{y} = h \land p < 0) & \Rightarrow (\hat{x} = \ell) \\
(p = 0) & \Rightarrow (\hat{y} \neq h \land \hat{y} \neq \ell)
\end{align*}
\]
As these statements indicate, what we can conclude about \( \hat{x} \) based on \( \hat{y} \) varies depending on the value of \( p \). Now, consider a scenario in which our prior knowledge about \( p \) is limited to the fact that \( p \) is certain, but we do not know the actual value or sign of this parameter. The only conclusion that we can make will be the following: \( \hat{y} = \bar{h} \Rightarrow \hat{x} = \bar{a} \), which obviously provides a different level of information about \( \hat{x} \) compared to what we had in (7)–(9).

FMR uses the premises given about output failure to draw conclusion about input faults. One of the following outcomes may be achieved as a result:

(i) minimum conditions, expressing the least we can state about the fault at an input, considering what we know about the output fault and the context.

(ii) certain causes, expressing the conjunctions of input failure modes that, if they occur, will certainly lead to the given output failure mode, regardless of the conditions of the other inputs.

**Definition 5.** Let \( y = f(x, z) \) be a system (or component) function and \( u, v \in \{\bar{h}, \bar{t}, \bar{l}, \bar{f}\} \) two failure modes. We say \( \hat{x} = v \) is a certain cause for \( \hat{y} = u \) in \( f \) if the occurrence of \( \hat{x} = v \) while \( z \) remains unchanged leads to the occurrence of \( \hat{y} = u \) regardless of the value of \( z \). That is: \( (\hat{x} = v \land \hat{z} = m) \Rightarrow \hat{y} = u \).

Def. 5 requires that the inputs and output of the function are all in an intended state first. This is based on Assumption 2. It also considers that part of the inputs remain unchanged during the transition to a faulty state. Thus, the observed change at the output can only be due to the input fault.

In a general form, let \( y = f(x_1, \ldots, x_n) \) be the function under study and \( \phi(\hat{x}_1, \ldots, \hat{x}_m) \) an expression on \( \hat{x}_1, \ldots, \hat{x}_m \), with \( m \leq n \). If \( \phi \) is a certain cause for \( \hat{y} = u \), then:

\[
(\phi(\hat{x}_1, \ldots, \hat{x}_m) \land \bigwedge_{i=m+1}^{n} (\hat{x}_i = m)) \Rightarrow \hat{y} = u
\]  

(10)

Where \( \phi \) is the disjunction of all the certain causes, we can also say:

\[
((\hat{y} = u) \land \bigwedge_{i=m+1}^{n} (\hat{x}_i = m)) \Rightarrow \phi(\hat{x}_1, \ldots, \hat{x}_m)
\]  

(11)

Certain causes provide a strong basis for reasoning; however, depending on the function, they may not be easy to calculate. In such cases, we may need to use minimum conditions to reason about failures, even though the outcome of such reasoning may not be as strong.

**Definition 6.** Let \( y = f(x, z) \) be a system function and \( u, v \in \{\bar{h}, \bar{t}, \bar{l}, \bar{f}\} \) two failure modes. We say \( \hat{x} = v \) is a conditional cause for \( \hat{y} = u \) in \( f \) if the occurrence of \( \hat{x} = v \) while \( z \) remains unchanged can lead to \( \hat{y} = u \) for some values of \( z \).

Compared to Def. 5 of certain causes, Def. 6 leads to the minimal conditions that can result in the nominated output failure.
We now demonstrate how failure models are derived from function models. The \textit{Or} function in Fig. 1 is defined as: \( y := \text{Or}(z_1, z_2) := z_1 \lor z_2 \), with \( z_1, z_2, y \in \mathbb{B} \).

It follows from this definition that:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (y = f \land y' = t) \\
& \Rightarrow ((z_1 = f \land z_2 = f) \land (z_1' = t \lor z_2' = t)) \\
& \Rightarrow ((\hat{z}_1 = f \land z_2 = f) \lor (\hat{z}_2 = f \land z_1 = f)) \\
\hat{y} = t & \Rightarrow (y = t \land y' = f) \\
& \Rightarrow ((z_1 = t \lor z_2 = t) \land (z_1' = f \land z_2' = f)) \\
& \Rightarrow ((\hat{z}_1 = t \land z_2' = f) \lor (\hat{z}_2 = t \land z_1 = f))
\end{align*}
\] (12)

The minimum conditions (see Def. 6) that we can draw from (12) and (13) are both in disjunctive terms:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (\hat{z}_1 = f \lor \hat{z}_2 = f) \\
\hat{y} = t & \Rightarrow (\hat{z}_1 = t \lor \hat{z}_2 = t)
\end{align*}
\] (14)

This is obviously not a surprise; if the output of \textit{Or} is faulty, it can only be due to a fault at either input. The certain causes (see Def. 5), also drawn from (12) and (13), provide more information:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (\hat{z}_1 = f \land \hat{z}_2 = f) \\
\hat{y} = t & \Rightarrow (\hat{z}_1 = t \land \hat{z}_2 = t)
\end{align*}
\] (15)

For value-dependent scenarios, (12) and (13) should be directly used.

\textit{Gcom} is another function used in our worked example in Fig. 1. This function is defined as follows: \( z_2 = Gcom(x, p_2) := (x > p_2) \). From this definition, it easily follows that:

\[
\begin{align*}
\hat{z}_2 = f & \Rightarrow (\hat{x} = \ell \lor \hat{p}_2 = \bar{h}) \\
\hat{z}_2 = t & \Rightarrow (\hat{x} = h \lor \hat{p}_2 = \ell)
\end{align*}
\] (18)

Similarly, from the definition of function \textit{Lcom}: \( z_1 = Lcom(x, p_1) := (x < p_1) \), it follows that:

\[
\begin{align*}
\hat{z}_1 = f & \Rightarrow (\hat{x} = \bar{h} \lor \hat{p}_1 = \ell) \\
\hat{z}_1 = t & \Rightarrow (\hat{x} = \ell \lor \hat{p}_1 = h)
\end{align*}
\] (20)

Note that unlike (12) and (13) for the \textit{Or} function, the derived failure models of some functions, such as (18) - (21) for \textit{Gcom} and \textit{Lcom}, may only contain value-independent terms.

\section{Composition}

Where multiple individual functions are combined to compose a complex system architecture, the global failure behavior of the system can be determined based
on the local models that represent the failure behavior of individual functions. What logically relates the local failure models to each other is the interconnections between the component functions. That is, we do not need to define the functional behavior of the whole system to determine its failure behavior. Instead, we can combine the local failure models of individual components, and draw the overall failure model by using rules of logic.

Logically, each component model will be a set of implication statements. The logical statement related to interconnected components can be merged by applying logic rules. In addition, we use the following FMR-specific rule to trim out the always-True and always-False sentences.

**Axiom 1** A system variable can only be in one state of failure at once. For $x \in \mathbb{R}$ and $y \in \mathbb{B}$:

\[
\begin{align*}
(\hat{x} = \ell \land \hat{x} = \bar{h}) &\equiv (\hat{x} = \ell \land \hat{x} = m) \equiv \hat{x} = \ell \land \hat{x} = m \equiv f \quad (22) \\
(\hat{y} = t \land \hat{y} = f) &\equiv (\hat{y} = t \land \hat{y} = m) \equiv \hat{y} = t \land \hat{y} = m \equiv f \quad (23) \\
(\hat{x} = a) &\equiv (\hat{y} = a) \equiv t \quad (24)
\end{align*}
\]

We now finalize the failure modeling for our worked example in Fig. 1. Suppose we are interested in finding the causes of a $\hat{y} = t$ failure and we know that input $x$ is a ‘certain’ variable. We know from (17), (19) and (21) that:

\[
\begin{align*}
\hat{y} = t &\Rightarrow (\hat{z}_1 = t \lor \hat{z}_2 = t) \quad (25) \\
\hat{z}_2 = t &\Rightarrow \hat{p}_2 = \ell \quad (26) \\
\hat{z}_1 = t &\Rightarrow \hat{p}_1 = \bar{h} \quad (27)
\end{align*}
\]

By substituting (26) and (27) in (25), we compose the overall model of the system failure:

\[
\hat{y} = t \Rightarrow (\hat{p}_1 = \bar{h} \lor \hat{p}_2 = \ell) \quad (28)
\]

4 Modeling rules

A component is the smallest part for which we can define a function. This clearly depends on the context of analysis and the suitability of the level of details. For an ordinary car owner, the whole engine will be one component, but a mechanic breaks that component into smaller parts, of which the mechanic can (and needs to) define the functions. Component functions can be distinguished with their following two main features:

(i) A component function is unbreakable to smaller parts, and is always used as a whole, even if some parts of it are not utilized.

(ii) The functional behavior of a component is fixed and well defined.

The failure behavior of a component is linked to its functional behavior. We should remember, however, that functional and failure behaviors are expressed in two different spaces.
We earlier showed FMR failure models for Not, Or, Gcom, and Lcom. The detail derivation of these and some more functions is given in the Appendix. In what follows, we set out some common rules that can be used for modeling individual components and we formulate FMR models for common system structures of DNF, CNF, and KooN.

4.1 Sign, and direction of fault

Let us begin with the relation between sign of numerical variables and change of direction in failure mode variables. The sign function is defined as follows:

\[
\text{sgn}(x) = \begin{cases} 
\frac{x}{|x|} & \text{if } x \neq 0 \\
0 & \text{if } x = 0
\end{cases} \quad \text{for } x \in \mathbb{R} \quad (29)
\]

With relation to failure mode variables, we use \{\sim, \text{null}\} to express possible changes of direction for faults, with null indicating the same direction and \sim indicating the opposite direction. Thus, if \hat{x} = t, then \sim \hat{x} = f; if \hat{x} = f, \sim \hat{x} = t; if \hat{x} = h, \sim \hat{x} = l; and if \hat{x} = l, \sim \hat{x} = h.

Now, we define the failure mode operator \tilde{s}(x) to relate \sim to \text{sgn}(x):

\[
\tilde{s}(x) \equiv \begin{cases} 
\sim & \text{if } \text{sgn}(x) = -1 \\
\text{null} & \text{otherwise}
\end{cases} \quad \text{for } x \in \mathbb{R} \quad (30)
\]

Axiom 2 Two consecutive changes of direction means no change: \sim\sim \equiv \text{null}.

4.2 Monotonic functions

The reasoning process in FMR is based on abstracting the values into directions of change. One particularly interesting group of functions that can be well suited for such abstractions is the category of monotonic functions.

Axiom 3 Let \( y := f(x) \) represent a monotonic function. If \( f \) is increasing (i.e., \( x' > x'' \Rightarrow f(x') \geq f(x'') \)), then \( (\hat{y} \equiv h \lor \hat{y} = l) \Rightarrow \hat{x} = \hat{y} \). If \( f \) is decreasing (i.e., \( x' > x'' \Rightarrow f(x') \leq f(x'') \)), then \( (\hat{y} = h \lor \hat{y} = l) \Rightarrow \hat{x} = \sim \hat{y} \).

When the inequality \( f(x') \geq f(x'') \) (or \( f(x') \leq f(x'') \)) is replaced with \( f(x') > f(x'') \) (or \( f(x') < f(x'') \)), the function is 'strictly' monotonic.

Corollary 1. Let \( f : \mathbb{R} \to \mathbb{R} \) be a continuous strictly monotonic function. Let \( x, y \in \mathbb{R} \) and \( y := f(x) \). Then: \( (\hat{y} = h \lor \hat{y} = l) \Rightarrow \hat{x} = \tilde{s}(\frac{\partial f}{\partial x})\hat{y} \).

Here, \( \frac{\partial f}{\partial x} \) denotes the gradient of \( f \). Depending on the sign of this gradient, \( \tilde{s}(\frac{\partial f}{\partial x}) \) determines if \( \hat{x} \) and \( \hat{y} \) are in the same or opposite direction.

One particular property of strictly monotonic functions is that the composition of two strictly monotonic functions is itself a strictly monotonic function.

---

8 Arbitrary values \( x' \) and \( x'' \) are not to be confused with the ‘intended value’ symbol \( x' \).
Corollary 2. Let $f, g : \mathbb{R} \to \mathbb{R}$ be two continuous strictly monotonic functions. Let $x, y \in \mathbb{R}$ and $y := f(g(x))$. Then: $(\hat{y} = \bar{h} \lor \hat{y} = \ell) \Rightarrow \hat{x} = \hat{\tilde{s}}(\frac{\partial f}{\partial g})\hat{\tilde{s}}(\frac{\partial g}{\partial x})\hat{y}$.

We use these properties of monotonic functions in identifying certain causes.

Lemma 1. Let $y = f(x_1, \ldots, x_n)$ be a function that is strictly monotonic with respect to $x_i$ (i.e., $x_i' > x_i'' \Rightarrow f(x_1, \ldots, x_i', \ldots, x_n) > f(x_1, \ldots, x_i'', \ldots, x_n)$, or $x_i' > x_i'' \Rightarrow f(x_1, \ldots, x_i', \ldots, x_n) < f(x_1, \ldots, x_i'', \ldots, x_n)$). Then $\hat{x}_i$ is a certain cause for $\hat{y}$.

Proof. Follows from the definition of certain causes (see Def. 5).

Corollary 3. Let $y = f(x_1, \ldots, x_n)$ be a continuous strictly monotonic function with respect to each of its individual arguments. Then: $(\hat{y} = \bar{h} \lor \hat{y} = \ell) \Rightarrow \bigvee_{i=1}^n (\hat{x}_i = \hat{\tilde{s}}(\frac{\partial f}{\partial x_i})\hat{y})$.

4.3 Common system structures

DNF and CNF are two standard forms in which composition rules can be generalized. These normal forms also play an important role in Boolean functions; every Boolean function can be transformed into a DNF or CNF [3]. Therefore, formulating FMR models for these two normal forms can be helpful in many other scenarios in which Boolean functions are the subject of the study.

**DNF:** Let $x = [x_{lk}]_{L \times K}$ be a matrix of Boolean variables. We define the DNF function $y := DNF(x)$ as follows:

$$y := DNF(x) := \bigvee_{l=1}^L y_l := \bigvee_{l=1}^L \bigwedge_{k=1}^K x_{lk}, \text{ with } y_l := \bigwedge_{k=1}^K x_{lk}$$

(31)

Based on Def. 5, the certain causes of fault at $y$ will be as follows:

$$\{ \bigvee_{l} \bigwedge_{k} \hat{x}_{lk} = \hat{y} \} \quad \text{DNF} \quad \{ \hat{y} = t \}$$

(32)

$$\{ \bigwedge_{l} \bigvee_{k} \hat{x}_{lk} = \hat{y} \} \quad \text{DNF} \quad \{ \hat{y} = f \}$$

(33)

And the minimum conditions will be as follows:

$$\{ \bigvee_{l,k} \hat{x}_{lk} = \hat{y} \} \quad \text{DNF} \quad \{ \hat{y} = t \lor \hat{y} = f \}$$

(34)

Proof: To prove the minimum conditions, let us begin with the disjunction $y = \bigvee_{l} y_l$. Recall that the reported and intended values at $y$ are denoted by $y$ and $y'$, and the reported and intended values at $y_l$ by $y_l$ and $y'_l$. From $\hat{y} = f$, it
follows that:

\[
\hat{y} = f \Rightarrow (y = f \land y' = t) \Rightarrow \bigwedge_{j=1}^{j=L} (y_j = f) \land \bigvee_{l=1}^{l=L} (y'_l = t)
\]

\[
\Rightarrow \bigvee_{l=1}^{l=L} (\hat{y}_l = f) \bigwedge_{j=1}^{j=L} (y_j = f) \Rightarrow \bigvee_{l=1}^{l=L} (\hat{y}_l = f)
\]

\[
\hat{y} = t \Rightarrow (y = t \land y' = f) \Rightarrow \bigvee_{l=1}^{l=L} (y_l = t) \land \bigwedge_{j=1}^{j=L} (y'_j = f)
\]

\[
\Rightarrow \bigvee_{l=1}^{l=L} (\hat{y}_l = t) \bigwedge_{j=1}^{j=L} (y'_j = f) \Rightarrow \bigvee_{l=1}^{l=L} (\hat{y}_l = t)
\]

Further, each conjunctive clause in DNF is defined by \(y_l = \land_{k} y_{lk}\). Similar to the disjunctive part, we can say the following:

\[
\hat{y}_l = f \Rightarrow (y_l = f \land y'_l = t) \Rightarrow \bigwedge_{k=1}^{k=K} (x_{lk} = f) \land \bigwedge_{j=1}^{j=K} (x'_j = t)
\]

\[
\Rightarrow \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = f) \bigwedge_{j=1}^{j=K} (x'_j = t) \Rightarrow \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = f)
\]

\[
\hat{y}_l = t \Rightarrow (y_l = t \land y'_l = f) \Rightarrow \bigwedge_{j=1}^{j=K} (x_{lj} = t) \land \bigvee_{k=1}^{k=K} (x'_k = f)
\]

\[
\Rightarrow \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = t) \bigwedge_{j=1}^{j=K} (x_{lj} = t) \Rightarrow \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = t)
\]

By substituting (37) in (35), and (38) in (36), we will have the following:

\[
\hat{y} = f \Rightarrow \bigvee_{l=1}^{l=L} \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = f)
\]

\[
\hat{y} = t \Rightarrow \bigvee_{l=1}^{l=L} \bigvee_{k=1}^{k=K} (\hat{x}_{lk} = t)
\]

which together prove (34).
We now prove the certain causes by using composition rules. Let us begin with the disjunction \( y = \vee y_l \):

\[
\hat{y} = f \Rightarrow (y = f \land y_l = t)
\]

\[
\Rightarrow \bigwedge_{l=1}^{L} (y_l = f) \land \bigvee_{j=1}^{L} (y'_j = t)
\]

\[
\Rightarrow \bigvee_{j=1}^{L} (y'_j = t) \bigwedge_{l=1}^{L} (y_l = f \land (y'_l = f \lor y'_l = t))
\]

\[
\Rightarrow \bigvee_{j=1}^{L} (y'_j = t) \bigwedge_{l=1}^{L} ((y_l = f \land y'_l = f) \lor (y_l = f \land y'_l = t)) \quad (41)
\]

To simplify the formulation, let \( A_l := (y_l = f \land y'_l = f) \), \( B_l := (y_l = f \land y'_l = t) \) and \( C := \bigvee_{j=1}^{L} (y'_j = t) \). Thus, the conclusion part of (41) will be the following:

\[
C \bigwedge_{l=1}^{L} (A_l \lor B_l) \equiv (C \bigwedge_{l=1}^{L} A_l) \lor \ldots \lor (C \bigwedge_{m \neq n}^{L} A_m B_n) \lor \ldots \lor (C \bigwedge_{l=1}^{L} B_l) \quad (42)
\]

The certain causes of a DNF, if any, should come from the conjunction of failure modes included in (42). So, let us examine each constituting term separately. First, the term \( C \bigwedge_{l=1}^{L} A_l \) does not hold; because it requires \( \bigwedge_{l=1}^{L} (y'_l = f) \), which contradicts the premise \( \bigvee_{l=1}^{L} (y'_l = t) \).

The middle terms in (42) are combinations of faulty and non-faulty variables. The non-faulty variables (in \( A_m \)'s) remain unchanged during the time when the failure occurs. Thus, we want to see if the conjunction of faulty variables (in \( \bigwedge_{n} B_n \)) is consistent with Def. 5 because it is only in that case that \( \bigwedge_{n} B_n \) can be a certain cause. We know that for \( \bigwedge_{n} B_n \) to be a certain cause, \( \bigwedge_{n} B_n \) needs to cause a change of output state independently of the status of other \( y_l \) variables. That is, regardless of the reported values at \( y_l \), the occurrence of \( \bigwedge_{n} B_n \) should always lead to the same change of state at the DNF output. This includes the case where there is a clause with output state \( y'_l = t \land y_l = t \), which is, obviously, not possible because if \( y_l = t \) before, during, and after the occurrence of failure, then the output \( y \) will remain \( t \) throughout, and no change will be observed at the output.

The last term \( C \bigwedge_{l=1}^{L} B_l \), however, depicts a certain cause: \( \bigwedge_{l=1}^{L} (\hat{y}_l = f) \). If all the \( y_l \) variables fail to \( \hat{y}_l = f \), the output of the disjunction will certainly fail to \( \hat{y} = f \). Thus, the (41) can be simplified as follows:

\[
C \bigwedge_{l=1}^{L} B_l \equiv \bigvee_{j=1}^{L} (y'_j = t) \bigwedge_{l=1}^{L} (y_l = f \land y'_l = t) \equiv \bigwedge_{l=1}^{L} (\hat{y}_l = f) \quad (43)
\]

\[9\] This is similar to our discussion in Section 3.2 where we used truth table to prove a small AND gate with two inputs.
We now use the conjunctive part of DNF: \( y_l = \bigwedge_{k=1}^K x_{lk} \), to identify the certain causes of \( \hat{y}_l = f \). The result can then be substituted in (43). From \( \hat{y}_l = f \), we have:

\[
\hat{y}_l = f \Rightarrow (y_l = f \wedge y'_l = t)
\]

\[
\Rightarrow \bigvee_{k=1}^K (x_{lk} = f) \wedge \bigwedge_{j=1}^K (x'_{lj} = t)
\]

\[
\Rightarrow \bigvee_{k=1}^K (\hat{x}_{lk} = f) \wedge \bigwedge_{j=1}^K (x'_{lj} = t)
\]

\[
\Rightarrow \bigvee_{k=1}^K (\hat{x}_{lk} = f)
\]  

(44)

Substituting (44) in (43) proves one part of the proposed model (i.e., failure scenario (33)). The other part of the proposed model (i.e., failure scenario (32)) can be proved in a similar manner. We begin with the disjunctive part of the DNF:

\[
\hat{y} = t \Rightarrow (y = t \wedge y' = f)
\]

\[
\Rightarrow \bigvee_{l=1}^L (y_l = t) \wedge \bigwedge_{j=1}^K (y'_j = f)
\]

\[
\Rightarrow \bigvee_{l=1}^L (\hat{y}_l = t) \wedge \bigwedge_{j=1}^K (y'_j = f)
\]

\[
\Rightarrow \bigvee_{l=1}^L (\hat{y}_l = t)
\]  

(45)

We now use the conjunctive part of DNF: \( y_l = \bigwedge_{k=1}^K x_{lk} \), to identify the certain causes of \( \hat{y}_l = t \), which we will later substitute in (45):

\[
\hat{y}_l = t \Rightarrow (y_l = t \wedge y'_l = f)
\]

\[
\Rightarrow \bigwedge_{k=1}^K (x_{lk} = t) \wedge \bigvee_{j=1}^K (x'_{lj} = f)
\]

\[
\Rightarrow \bigvee_{j=1}^K (x'_{lj} = f) \bigwedge_{k=1}^K (x_{lk} = t \wedge (x'_{lk} = t \vee x'_{lk} = f))
\]

\[
\Rightarrow \bigvee_{j=1}^K (x'_{lj} = f) \bigwedge_{k=1}^K ((x_{lk} = t \wedge x'_{lk} = t) \vee (x_{lk} = t \wedge x'_{lk} = f))
\]  

(46)
Let $A_{lk} := (x_{lk} = t \land x'_{lk} = t)$, $B_{lk} := (x_{lk} = t \land x'_{lk} = f)$ and $C := \bigvee_{j=1}^{K} (x'_{lj} = f)$. Thus, the conclusion part of (46) will be the following:

$$C_k = K \land \sum_{j=1}^{K} (x'_{lj} = f) \equiv (C_k = K \land \sum_{j=1}^{K} (x_{lk} = t \land x'_{lk} = f)) \equiv \bigwedge_{k=1}^{K} (\hat{x}_{lk} = t)$$

(47)

Similar to (42), the only possible certain causes from (47) will be the following:

$$\hat{y}_l = t \Rightarrow \bigwedge_{k=1}^{K} (\hat{x}_{lk} = t)$$

(49)

Substituting (49) in (48) proves the failure scenario (32).

**CNF:** Similar to a DNF, we define the CNF function $y := CNF(x)$ as follows:

$$y := CNF(x) := \bigwedge_{l=1}^{L} y_l := \bigwedge_{l=1}^{L} \bigvee_{k=1}^{K} x_{lk}, \text{ with } y_l := \bigvee_{k=1}^{K} x_{lk}$$

(50)

Based on Def. [5] the certain causes of fault at $y$ will be as follows:

$$\{ \bigwedge_{l=1}^{L} \bigvee_{k=1}^{K} \hat{x}_{lk} = \hat{y} \} \quad \text{CNF} \quad \{ \hat{y} = t \}$$

(51)

$$\{ \bigvee_{l=1}^{L} \bigwedge_{k=1}^{K} \hat{x}_{lk} = \hat{y} \} \quad \text{CNF} \quad \{ \hat{y} = f \}$$

(52)

And the minimum conditions will be as follows:

$$\{ \bigvee_{l=1}^{L} \bigwedge_{k=1}^{K} \hat{x}_{lk} = \hat{y} \} \quad \text{CNF} \quad \{ \hat{y} = t \lor \hat{y} = f \}$$

(53)

**Proof:** Similar to the DNF function.

**KooN:** A KooN system is a composition of $n$ independent and identical components, where $k$ out of $n$ components need to function for the overall system to function. This system can be modeled as follows. For $y, x_1, ..., x_n \in \mathbb{B}$, where $x_1, ..., x_n$ are suspicious variables:

$$y := KooN(x_1, ..., x_n, k) = (x_1 \land .. \land x_{k-1} \land x_k) \lor (x_1 \land .. \land x_{k-1} \land x_{k+1}) \lor .. \lor (x_{n-k-1} \land .. \land x_{n-1} \land x_n)$$

(54)
In a KooN function, the output will be \( y = t \) if \( k \) inputs are \( t \), and the output will be \( y = f \) otherwise. The certain causes of this function are as follows:

\[
\{ (\hat{x}_1 = t \land .. \land \hat{x}_{k-1} = t \land \hat{x}_k = t) \lor (\hat{x}_1 = t \land .. \land \hat{x}_{k-1} = t \land \hat{x}_{k+1} = t) \\
\lor .. \lor (\hat{x}_{n-k+1} = t \land .. \land \hat{x}_{n-1} = t \land \hat{x}_n = t) \} \quad \text{KooN} \quad \{ \hat{y} = t \} \quad (55)
\]

\[
\{ (\hat{x}_1 = f \land .. \land \hat{x}_{n-k} = f \land \hat{x}_{n-k+1} = f) \lor (\hat{x}_1 = t \land .. \land \hat{x}_{n-k} = f \land \hat{x}_{n-k+2} = f) \\
\lor .. \lor (\hat{x}_k = f \land .. \land \hat{x}_{n-1} = f \land \hat{x}_n = f) \} \quad \text{KooN} \quad \{ \hat{y} = f \} \quad (56)
\]

**Proof:** Follows from the proof of DNF. Alternatively, truth tables can be used for specific values of \( k \) and \( n \) (see Appendix for truth-table-based proof for the And function).

Our discussion around the DNF, CNF, and KooN functions highlights a few interesting facts:

(i) First, if the output of a DNF or CNF is deviated in \( f \) or \( t \) direction, at least one of its literals is deviated in the same direction (see (34) and (53)).

(ii) More generally, if the output of a Boolean function is deviated in \( f \) or \( t \) direction, at least one of its inputs is deviated in the same direction; because every Boolean function can be converted to a DNF or CNF.

(iii) We also know that the certain causes in DNF and CNF systems follow the architecture of the system. That is, the certain causes for an output failure \( \hat{y} = t \) in a DNF (or CNF) function are expressed by the same DNF (or CNF) used for its success \( y = t \), if we just replace the literals \( x_{lk} \) with failure modes \( \hat{x}_{lk} = t \). This is based on (32), (33), (51), and (52). Conversely, for the output failure \( \hat{y} = f \), the dual architecture CNF (or DNF) should be used for failure modeling, while the literals \( x_{lk} \) are replaced with failure modes \( \hat{x}_{lk} = f \).

(iv) In a system that needs at least \( k \) out of \( n \) of its components for functioning, the system can fail in two ways: commission of \( k \) components resulting in a commission at the system output, and omission of \( n - k + 1 \) components resulting in an omission at the system output (see (55) and (56)).

### 4.4 Duality and equality

Where the functional behaviors of two systems are the same, one may also expect to observe the same failure behaviors from the two systems. Conversely, where two system functions ‘negate’ each other, one may expect to see opposite failure behaviors from the two systems. In this section, we examine these questions. First, we address the question of equality.

**Theorem 1.** Given two functions \( y = f(x_1, .., x_n) \) and \( y = g(x_1, .., x_n) \) and failure mode \( u \in \{ f, t, f, \bar{f} \} \), if \( \hat{y} = u \) is reachable in \( f \) and \( g \), and if \( f \) and \( g \) are equal (i.e., if for any given input \( x_1, .., x_n \) we have \( f(x_1, .., x_n) = g(x_1, .., x_n) \)), then the certain causes of \( f \) and \( g \) for \( \hat{y} = u \) are identical.
Definition 7. Given is similar notation for failure models. Let \( \phi(x_1, \ldots, x_m) \) represent a certain cause for \( \hat{y} = u \) in \( f \). Thus:

\[
\phi \wedge_{i>m} (\hat{x}_i = m) \Rightarrow \hat{y} = u, \text{ due to } f
\]

\[
\Rightarrow m_d(\phi(x_1, \ldots, x_n), f(x'_1, \ldots, x'_n)) = u
\]

\[
\Rightarrow m_d(\phi(x_1, \ldots, x_n), g(x'_1, \ldots, x'_n)) = u
\]

\[
\Rightarrow \hat{y} = u, \text{ due to } g
\]

which means \( \phi(x_1, \ldots, x_m) \) is also a certain cause for \( \hat{y} = u \) in \( g \).

We now discuss the ‘duality’ in failure models. In standard logic, the dual expression will be derived from the original expression by exchanging all the \( \wedge \) and \( \vee \) operators and negating all the literals in \( f \). We first introduce a similar notation for failure models.

**Definition 7.** Given is \( \phi(\hat{x}_1, \ldots, \hat{x}_n) \), a Boolean expression on failure mode variables \( \hat{x}_i \)'s. The failure mode dual of \( \phi \) is depicted by \( \phi^D(\hat{x}_1, \ldots, \hat{x}_n) \), in which all the \( \wedge \) and \( \vee \) operators are exchanged and all the failure mode values inverted by applying the \( \sim \) operator introduced in Section 4.1.

As an example, for \( \phi \equiv (\hat{x}_1 = \bar{h} \wedge \hat{x}_2 = \bar{f} \wedge \hat{x}_1 = m \vee \hat{x}_3 = \bar{f}) \), its failure mode dual expression will be \( \phi^D \equiv (\hat{x}_1 = \bar{f} \vee \hat{x}_2 = \bar{h} \wedge (\hat{x}_1 = m \wedge \hat{x}_3 = \bar{f})) \).

**Theorem 2.** Given are failure mode \( u \in \{f, t\} \) and Boolean function \( y = f(x_1, \ldots, x_n) \). If \( \phi(\hat{x}_1, \ldots, \hat{x}_m) \) is a certain cause for \( \hat{y} = u \), then \( \phi^D(\hat{x}_1, \ldots, \hat{x}_m) \) is a certain cause for \( \hat{y} = \sim u \).

**Proof.** We proved the failure models for DNF and CNF, in which this duality rule is apparent (see (32), (33), (51), and (52)). Every Boolean function can be converted to a DNF (or CNF). Let \( g \) be the DNF equivalent of \( f \). According to Theorem 1, the certain causes of \( f \) and \( g \) are identical. Therefore, we can conclude that the duality rule applies to all Boolean functions.

**Corollary 4.** Given Boolean function \( f \) and its logical dual \( f^d \), the certain causes in \( f^d \) are the failure mode duals of the certain causes in \( f \).

**Proof.** Let \( y = f(x_1, \ldots, x_n) \) and \( y^d = f^d(x_1, \ldots, x_n) \). Let \( \phi(\hat{x}_1, \ldots, \hat{x}_m) \) depict a certain cause for \( \hat{y} = t \) in \( f \), with \( m \leq n \). Thus:

\[
\phi(\hat{x}_1, \ldots, \hat{x}_m) \wedge_{i=m+1} (\hat{x}_i = m) \Rightarrow \hat{y} = t
\]

\[
\Rightarrow (y = t \wedge y^d = f)
\]

\[
\Rightarrow (y^d = f \wedge y'^d = t)
\]

\[
\Rightarrow y'^d = f
\]

which means \( \phi \) is a certain cause for \( y'^d = f \). From Theorem 2, it follows that \( \phi^D \) is a certain cause for \( y^d = f \), and that proves this corollary for output failure mode \( t \). Similarly, we can show that if \( \phi \) is a certain cause for \( \hat{y} = f \), then \( \phi^D \) is a certain cause for \( y^d = t \). Thus, the corollary is proven for both \( f \) and \( t \).
5 Discussion

Loops: Some practical systems use loops in their architectures. A loop connects the output of a function back to its own input through direct or indirect connections. We discuss the effect of loops on the transition of faults.

Consider the looped system defined as follows:

$$y = f(x, y)$$

where $x = [x_1...x_n]$ is the system input vector, and the argument $y$ depicts a loop to the same output $y$. Considering the two moments in time $t_1$ and $t_2$ such that $t_2 = t_1 - dt$, the system equation (59) can be written as follows:

$$y(t_2) = f(x(t_1), y(t_1))$$

Suppose a failure is observed at $t_2$ (i.e., $\hat{y}(t_2) \neq m$ and $y(t_2) \neq y'(t_2)$). Therefore:

$$f(x(t_1), y(t_1)) \neq f(x'(t_1), y'(t_1))$$

From Assumption 2, we know that the system is in an intended state at $t_1$, which means $y(t_1) = y'(t_1)$ and thus $\hat{y}(t_1) = m$. This simply means that the only possible cause for $\hat{y}(t_2) \neq m$ could be $\hat{x}(t_1)$ and not $\hat{y}(t_1)$. That is, the loop does not affect the output failure.

In practice, the relationship between $\hat{y}(t_2)$ and $\hat{x}(t_1)$ and $\hat{y}(t_1)$ is always expressed in the form of a Boolean expression, comprising literals such as $\hat{x}(t_1) = v$ and $\hat{y}(t_1) = u$. Given that we know for certain that $\hat{y}(t_1) = m$, all sentences like $\hat{y}(t_1) = u$ in the final expression, where $u \neq m$, can be replaced with certain $False$, and all $\hat{y}(t_1) = m$ sentences can be replaced with certain $True$. These replacements will help further simplify the Boolean expression and reduce it to an expression based on $\hat{x}(t_1)$ only.

The value of a loop connection, however, can affect the transition of faults where we choose to take value-dependent reasoning. In (7)-(9), we showed how the sign of a certain variable could affect the reasoning about the failure mode of another (suspicious) variable. As another example, a correct $f$ at one input of an And function will block all the other faulty inputs from affecting the output.

Change and impact: In some analyses, we may be interested in the impact of a particular variable on the overall system output failure. Consider a system in which the settings of parameters can be adjusted. When an output failure is observed, one may want to see if altering a parameter can potentially fix the output failure. In this section, we introduce an ‘impact index’ that can be useful in such scenarios. First, we introduce a comparison function that can help measure the difference between two given failure modes:
Definition 8. Let $\hat{x}$ and $\hat{z}$ be two failure mode variables such that $\hat{x}, \hat{z} \in \{h, m, f\}$ (or $\hat{x}, \hat{z} \in \{t, m, f\}$). The failure mode comparison function defined below measures the difference between $\hat{x}$ and $\hat{z}$:

$$\text{cmp}(\hat{x}, \hat{z}) := \begin{cases} 
1 & \text{if } \hat{x} = h \text{ and } \hat{z} = f, \text{ or } \hat{x} = t \text{ and } \hat{z} = t \\
0.5 & \text{if } \hat{z} = m, \text{ and } \hat{x} = h \text{ or } \hat{x} = t \\
0 & \text{if } \hat{x} = \hat{z} \\
-0.5 & \text{if } \hat{z} = m, \text{ and } \hat{x} = \ell \text{ or } \hat{x} = f \\
-1 & \text{if } \hat{x} = \ell \text{ and } \hat{z} = h, \text{ or } \hat{x} = f \text{ and } \hat{z} = t
\end{cases}$$

The impact of a variable on the output failure can then be defined as follows.

Definition 9. Let $y = f(x_1, \ldots, x_n)$ be a system function, $\hat{y}$ the failure mode at output $y$, and $v$ and $w$ two possible assignments to $x_j$. The impact of a change from $x_j = v$ to $x_j = w$ on the output failure mode $\hat{y}$ is measured by the following:

$$I_{x_j} := |\text{cmp}(\hat{y}_v, \hat{y}_w)|,$$

where $\hat{y}_v = \hat{y}|_{x_j=v}$ and $\hat{y}_w = \hat{y}|_{x_j=w}$

The impact index indicates how effectively the change of the current value at $x_j$ can impact the occurrence of a failure mode at $y$.

The measure we introduced in (63) is useful in systems with single outputs. Where the system produces multiple outputs (e.g., where the system output is $y = [y_1 \ldots y_m]$), a collective measure can be used to show the impact as follows:

$$I_{x_j} := \sum_{i=1}^{m} |\text{cmp}(\hat{y}_{iv}, \hat{y}_{iw})|,$$

where $\hat{y}_{iv} = \hat{y}_i|_{x_j=v}$ and $\hat{y}_{iw} = \hat{y}_i|_{x_j=w}$

The impact measure can be particularly useful when the impact of a parameter change is to be evaluated. Where an output failure is an indication of an incorrect parameter and the correct value of the parameter is unknown, one possible way to fix the problem is to propose a new setting for the parameter. In such scenarios, the impact index can help to evaluate the effectiveness of the new setting on output failure.

Application: Our focus in this paper is on the theoretical aspects of FMR. However, we have also realized these concepts in the form of an analysis tool. With respect to SIS programs, the tool has already been used in multiple projects in the power industry. The FMR tool imports the SIS program, analyzes it like a function model of a system, and identifies the possible input and/or parameter faults. We have demonstrated this in detail in the previous works (e.g., in [2017]). At its current capacity, the tool is capable of analyzing models of cyber physical systems too. We are currently working on a modeling project for a road tunnel safety scenario where the tool is to analyze failures in a complex tunnel ventilation system.
6 Related works

FMR was not created to expand or enhance an existing method. It is therefore hard to discuss its relative advantages or limitations in direct comparison with another existing method. However, we can still compare some of the concepts that are commonly utilized by both FMR and the other methods.

In the previous works on FMR [16,20,18,17], we compared various aspects of FMR with FTA [32,31], FMEA [11], inference-based diagnostics [40,30], Abstract Interpretation [2,8], Qualitative Reasoning [5], component-based methods [6,34,7,28,23], diagnosis methods [30,4] and program debugging [15,34].

One particular area with which FMR does not have much in common is formal verification. Model Checking [1], for instance, is a method for checking whether a model of a system meets a given specification. FMR does not verify a model against a specification, nor does it model a system function. It reads the functional behavior and extracts the failure behavior as it actually is, without making any judgment. It will be then up to the engineers to decide if that identified behavior is acceptable. Similarly, FMR is not to be compared with Model-Based Safety Analysis (MBSA) [22] because as MBSA does not derive the failure mode from function model.

The closest work to FMR’s application in SIS programs is Software FTA [25]. In simple terms, a software fault tree is a composition of a set of smaller fault trees, each of which represents the failure behavior of a small piece of code or an instruction. The overall fault tree represents the failure behavior of the whole program. The main difference between FMR and SFTA is that SFTA is based on the standard FTA whereas FMR is a method based on abstract deviations at variables (i.e., failure modes). Also, FMR is a method for generic composition of functions whereas SFTA is used solely for failures related to the execution of software programs.

Despite the differences, both methods address fault propagation, which is also shared with other methods, such as FFIP [24]. FFIP models the hardware and software components based on their functional behavior. An FFIP model is used to analyze the propagation of failure through the (function) model, determining the final failure at the system level. Failure scenarios are then simulated to assess the effect of failure in the design, which is different to what FMR does.

7 Conclusion

We generalized and extended the FMR formalism for its use in generic systems. Our main objective was to establish the mathematical foundation that is required for failure reasoning, independent from specific application contexts. The framework provided here can be used by safety analysts for all failure mode reasoning scenarios, including SIS programs. Potential future work include implementation of FMR in more industrial cases, inclusion of time delays in FMR models as parameters, and extension of FMR for system architectural errors.
References

1. Clarke Jr, E.M., Grumberg, O., Kroening, D., Peled, D., Veith, H.: Model checking. MIT press (2018)
2. Cousot, P., Cousot, R.: Abstract interpretation frameworks. Journal of logic and computation 2(4), 511–547 (1992)
3. Crama, Y., Hammer, P.L.: Boolean functions: Theory, algorithms, and applications. Cambridge University Press (2011)
4. Davis, R.: Diagnostic reasoning based on structure and behavior. Artificial intelligence 24(1-3), 347–410 (1984)
5. De Kleer, J., Brown, J.S.: A qualitative physics based on confluences. Artificial intelligence 24(1-3), 7–83 (1984)
6. Fenelon, P., McDermid, J.A.: An integrated tool set for software safety analysis. Journal of Systems and Software 21(3), 279–290 (1993)
7. Ge, X., Paige, R.F., McDermid, J.A.: Probabilistic failure propagation and transformation analysis. In: International Conference on Computer Safety, Reliability, and Security. pp. 215–228. Springer (2009)
8. Gehr, T., Mirman, M., Drachsler-Cohen, D., Tsankov, P., Chaudhuri, S., Vechev, M.; AI2: Safety and robustness certification of neural networks with abstract interpretation. In: 2018 IEEE Symposium on Security and Privacy (SP). pp. 3–18. IEEE (2018)
9. Genesereth, M.R.: The use of design descriptions in automated diagnosis. Artificial Intelligence 24(1-3), 411–436 (1984)
10. Giunchiglia, F., Walsh, T.: A theory of abstraction. Artificial intelligence 57(2-3), 323–389 (1992)
11. IEC: IEC 60812: Analysis Techniques for System Reliability: Procedure for Failure Mode and Effects Analysis (FMEA). IEC (2006)
12. IEC: IEC 61508-4: Functional safety of electrical/electronic/programmable electronic safety related systems - Part 4: Definitions and abbreviations (2010)
13. IEC: IEC 61511-1: Functional safety-Safety instrumented systems for the process industry sector - Part 1: Framework, definitions, system, hardware and application programming requirements (2016)
14. ISO: ISO/IEC/IEEE 15288: Systems and software engineering – System life cycle processes (2015)
15. J, M.G., Tom, B., Corey, S.: Debugging. In: Art of Software Testing, pp. 157–174. John Wiley & Sons, 3rd edn. (2012)
16. Jahanian, H.: Failure Mode Reasoning. In: 2019 4th International Conference on System Reliability and Safety (ICSRS). pp. 295–303. IEEE (2019)
17. Jahanian, H.: Parametric faults in safety critical programs. In: International Conference on Formal Methods for Industrial Critical Systems. pp. 51–66. Springer (2021)
18. Jahanian, H., McIver, A.: Reasoning with failures. In: International Conference on Formal Engineering Methods. pp. 36–52. Springer (2020)
19. Jahanian, H., McIver, A.: Reasoning with failures. arXiv preprint arXiv:2007.10841 (2020)
20. Jahanian, H., Parker, D., Zeller, M., McIver, A., Papadopoulos, Y.: Failure Mode Reasoning in Model Based Safety Analysis. In: 7th International Symposium on Model-Based Safety and Assessment (2020)
21. Joshi, A.: Behavioral fault modeling and model composition for model-based safety analysis. Ph.D. thesis, University of Minnesota (2008)
22. Joshi, A., Miller, S.P., Whalen, M., Heimdahl, M.P.: A proposal for model-based safety analysis. In: 24th Digital Avionics Systems Conference. vol. 2, pp. 13–pp. IEEE (2005)
23. Kaiser, B., Liggesmeyer, P., Mäckel, O.: A new component concept for fault trees. In: Proceedings of the 8th Australian workshop on Safety critical systems and software-Volume 33. pp. 37–46. ACS, Inc. (2003)
24. Kurtoglu, T., Tumer, I.Y.: A graph-based fault identification and propagation framework for functional design of complex systems (2008)
25. Leveson, N.G., Harvey, P.R.: Analyzing software safety. IEEE Transactions on Software Engineering (5), 569–579 (1983)
26. Papadopoulos, Y.: Safety-directed system monitoring using safety cases. Ph.D. thesis, University of York (2000)
27. Papadopoulos, Y., McDermid, J., Sasse, R., Heiner, G.: Analysis and synthesis of the behaviour of complex programmable electronic systems in conditions of failure. Reliability Engineering & System Safety 71(3), 229–247 (2001)
28. Papadopoulos, Y., McDermid, J.A.: Hierarchically performed hazard origin and propagation studies. In: International Conference on Computer Safety, Reliability, and Security. pp. 139–152. Springer (1999)
29. Peirce, C.S.C.S.: Collected papers of Charles Sanders Peirce / edited by Charles Hartshorne and Paul Weiss. [V. 5]. Belknap Press of Harvard University Press, Cambridge, Mass (1974)
30. Reiter, R.: A theory of diagnosis from first principles. Artificial intelligence 32(1), 57–95 (1987)
31. Stamatelatos, M., Vesely, W., Dugan, J., Fragola, J., Minarick, J., Railsback, J.: Fault tree handbook with aerospace applications (2002)
32. Vesely, W.E., Goldberg, F.F., Roberts, N.H., Haasl, D.F.: Fault Tree Handbook (NUREG-0492). US Nuclear Regulatory Commission (1981)
33. Wallace, M.: Modular architectural representation and analysis of fault propagation and transformation. Electronic Notes in Theoretical Computer Science 141(3), 53–71 (2005)
34. Whittington, J.: Debugging Functional Programs by Interpretation. Ph.D. thesis, University of Leicester (2020)
A Failure models for selected functions

In this appendix, we propose FMR models for a selected group of basic functions, and we demonstrate their correctness. We will cover a set of numerical functions, including addition ($Add$), subtraction ($Sub$), average ($Avg$), limiter ($Lim$), inverse ($Inv$), absolute ($Abs$), multiplication ($Mul$); a set of Boolean functions, including AND gate ($And$), OR gate ($Or$) and negation ($Not$); and also a set of “combined” functions, including greater comparison ($Gcom$) and less comparison ($Lcom$).

Consistent with our notation in the rest of this paper, in this appendix, $x$, $x'$ and $\hat{x}$ depict the reported value, intended value, and the failure mode at variable $x$, respectively. The same applies to any other variable named here.

A.1 Numerical functions

Addition: For $y, x_1, x_2 \in \mathbb{R}$ where $x_1, x_2$ are two suspicious variables and $y := Add(x_1, x_2) := x_1 + x_2$, the certain causes and minimum conditions for an output failure are as follows:

$$\{\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y}\} \quad Add \quad \{\hat{y} = h \lor \hat{y} = l\} \quad (65)$$

Proof: Let $\delta \in \mathbb{R}^+$ be a positive real-valued numbers. We first show that $\hat{x}_1 = l$ is a certain cause for $\hat{y} = l$, as follows:

$$(\hat{x}_1 = l \land \hat{x}_2 = m) \Rightarrow (x_1 = x'_1 - \delta \land x_2 = x'_2) \Rightarrow y = y' - \delta \Rightarrow \hat{y} = l \quad (66)$$

Similarly, we can show that $\hat{x}_2 = l$ and $(\hat{x}_1 = l \land \hat{x}_2 = l)$ are two certain causes. Thus, we will have the following:

$$\hat{y} = l \Rightarrow (\hat{x}_1 = l \lor \hat{x}_2 = l) \quad (67)$$

In the same way, it can be shown that $\hat{x}_1 = h, \hat{x}_2 = h$, and $(\hat{x}_1 = h \land \hat{x}_2 = h)$ are the certain causes for $\hat{y} = h$. Therefore:

$$\hat{y} = h \Rightarrow (\hat{x}_1 = h \lor \hat{x}_2 = h) \quad (68)$$

which together with (67) means:

$$(\hat{y} = l \lor \hat{y} = h) \Rightarrow (\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y}) \quad (69)$$

which is the logical representation of the proposed model in (65).

Where $x_2$ is a certain variable, $\hat{x}_2 = l$ and $\hat{x}_2 = h$ are ruled out, and the model in (65) will be reduced to:

$$\{\hat{x}_1 = \hat{y}\} \quad Add \quad \{\hat{y} = h \lor \hat{y} = l\} \quad (70)$$

Note that $Add$ is an increasing function with respect to both $x_1$ and $x_2$. Therefore, we could as well use Corollary 3 to prove (65).
**Subtraction:** For $y, x_1, x_2 \in \mathbb{R}$ where $x_1, x_2$ are two suspicious variables and $y := Sub(x_1, x_2) := x_1 - x_2$, the certain conditions and minimum conditions for an output failure are as follows:

$$\{ \hat{x}_1 = \hat{y} \lor \hat{x}_2 = \sim \hat{y} \} \quad Sub \quad \{ \hat{y} = h \lor \hat{y} = l \} \quad (71)$$

**Proof:** $Sub$ is an increasing function with respect to $x_1$ and a decreasing function with respect to $x_2$. Therefore, as per Corollary 3, we can say the following:

$$(\hat{y} = l \lor \hat{y} = h) \Rightarrow (\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \sim \hat{y}) \quad (72)$$

which is the logical representation of the proposed model in (71). Alternatively, we could prove the failure model (71) by using $\delta \in \mathbb{R}^+$, as we did earlier for the $And$ function.

Obviously, if $x_1$ is certain and $x_2$ suspicious, the model (71) will be reduced to the following:

$$\{ \hat{x}_2 = \sim \hat{y} \} \quad Sub \quad \{ \hat{y} = h \lor \hat{y} = l \} \quad (73)$$

and if $x_1$ is the suspicious variable and $x_2$ the certain one, the model (71) will be reduced to:

$$\{ \hat{x}_1 = \hat{y} \} \quad Sub \quad \{ \hat{y} = h \lor \hat{y} = l \} \quad (74)$$

**Average:** For $y, x_1, x_2 \in \mathbb{R}$ where $x_1, x_2$ are two suspicious variables and $y := Avg(x_1, x_2) := (x_1 + x_2)/2$, the certain causes and minimum conditions for an output failure are as follows:

$$\{ \hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y} \} \quad Avg \quad \{ \hat{y} = h \lor \hat{y} = l \} \quad (75)$$

**Proof:** $Avg$ is an increasing function with respect to both $x_1$ and $x_2$. Therefore, as per Corollary 3, we have the following:

$$(\hat{y} = l \lor \hat{y} = h) \Rightarrow (\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y}) \quad (76)$$

which is the logical representation of the proposed model in (75). Readers can also refer to [19] for an alternative proof, using failure truth tables.

**Limiter:** A limiter function is one that limits its real-valued argument between a lower and an upper boundary. Let $x, p_l, p_u \in \mathbb{R}$ be three variables. We define $y$ as a limiter function, as follows:

$$y := Lim(x, p_l, p_u) := \begin{cases} p_u & \text{if } x \geq p_u \\ x & \text{if } p_l \leq x \leq p_u \\ p_l & \text{if } x \leq p_l \end{cases} \quad (77)$$

Our proposed model for this function and for a suspicious $x$ and certain $p_l$ and $p_u$ is as follows:

$$\{ \hat{x} = \hat{y} \} \quad Lim \quad \{ \hat{y} = h \lor \hat{y} = l \} \quad (78)$$

**Proof:** The $Lim$ function defined above is an increasing function with respect to $x$. Therefore, as per Corollary 3, the proposed model is correct.
Inverse: Let \( x \in \mathbb{R} \) be a suspicious variable and \( y := Inv(x) := 1/x \). Our proposed model is as follows:

\[
\{ \hat{x} = \sim \hat{y} \} \quad Inv \quad \{ \hat{y} = \bar{h} \lor \hat{y} = \ell \}
\]  

(79)

Proof: The \( Inv \) function defined above is a decreasing function with respect to \( x \). Therefore, as per Corollary 3, the proposed model is correct.

Absolute value: Let \( x \in \mathbb{R} \) be a suspicious variable and \( y := Abs(x) := |x| = sgn(x)x \). It is not possible to determine a certain cause in \( Abs \) function. Deviation in either direction \( \ell \) or \( \bar{h} \) at the input may result in either direction at the output, depending on the scale of the deviation. Suppose the correct value is \( x' = 5 \). A deviation to \( x = 1 \), which is a \( \hat{x} = \ell \), causes a \( \hat{y} = \ell \) at the output, but a deviation to \( x = -6 \), although it is a \( \hat{x} = \ell \) too, leads to \( \hat{y} = \bar{h} \). Therefore, we can only define the minimum conditions, as follows:

\[
\{ \hat{x} = \bar{h} \lor \hat{x} = \ell \} \quad Abs \quad \{ \hat{y} = \bar{h} \lor \hat{y} = \ell \}
\]  

(80)

This model can only be narrowed down if we have additional information about the sign of \( x \). Let \( s = sgn(x) \) and \( a = |x| ; \) thus, \( x = sa \). If we know that \( \hat{s} = m \) during the occurrence of the failure, then we can say \( \hat{x} = \hat{s}(s)a = \hat{s}(x)a \). On the other hand, \( y \) is an increasing function with respect to \( a \). Therefore:

\[
((\hat{y} = \ell \lor \hat{y} = \bar{h}) \land \hat{s} = m) \Rightarrow \hat{a} = \hat{y} \\
\Rightarrow \hat{x} = \hat{s}(x)\hat{y}
\]  

which can be summarized in:

\[
\{ \hat{x} = \hat{s}(x)\hat{y} \} \quad Abs \quad \{ ((\hat{y} = \ell \lor \hat{y} = \bar{h}) \land \hat{s} = m) \}
\]  

(82)

Note that the right-hand side of (82) expresses the premises of the reasoning, part of which is the effect \((\hat{y} = \ell \lor \hat{y} = \bar{h})\).

Multiplication: For \( y, x, p \in \mathbb{R} \) where \( x \) is suspicious and \( p \) certain and \( y := Mul(x,p) := x \times p \):

\[
\{ \hat{x} = \hat{s}(p)\hat{y} \} \quad Mul \quad \{ \hat{y} = \bar{h} \lor \hat{y} = \ell \}
\]  

(83)

Proof for (83): Follows from the definition of the sign function \( \hat{s}(x) \).

However, defining the failure modes of \( Mul \) for two suspicious variables is rather complicated. Let \( x_1, x_2 \in \mathbb{R} \) be two suspicious variables and \( y := Mul(x_1, x_2) := x_1 \times x_2 \). In a value-independent context, the effect of the signs of \( x_1 \) and \( x_2 \) prevents us from nominating specific certain causes. An increase (or decrease) in either input may result in either an increase or a decrease at the output. Thus, the minimum conditions for \( Mul \) can be defined as follows:

\[
\{ \hat{x}_1 = \bar{h} \lor \hat{x}_1 = \ell \lor \hat{x}_2 = \bar{h} \lor \hat{x}_2 = \ell \} \quad Mul \quad \{ \hat{y} = \bar{h} \lor \hat{y} = \ell \}
\]  

(84)
The above statement is very broad. It simply suggests that if output \( y \) is faulty, then at least one of the inputs \( x_1 \) and \( x_2 \) must be faulty. This outcome is still helpful in some scenarios in which we analyze the input faults and would like to examine all possibilities of failure. However, we can narrow this model further down if we know more about the input variables. One interesting case is when we know that only one input is faulty. In this scenario, the following model can be used:

\[
\{ \hat{x}_1 = \bar{s}(x_2)\hat{y} \lor \hat{x}_2 = \bar{s}(x_1)\hat{y} \} \quad \text{Mul} \quad \{ (\hat{y} = \bar{h} \lor \hat{y} = \ell) \land (\hat{x}_1 = m \lor \hat{x}_2 = m) \} \tag{85}
\]

which is basically an extension of (83).

In another scenario, suppose we know the signs of \( x_1 \) and \( x_2 \), and we know that the signs will not change as part of the fault. This is obviously a context-based assumption, but not an unrealistic one; in real systems, we often do know the upper and lower bounds for the variables. Let \( s_1 = sgn(x_1) \) and \( s_2 = sgn(x_2) \) represent the signs of \( x_1 \) and \( x_2 \). The following model can be used for \( \text{Mul} \):

\[
\{ \hat{x}_1 = \bar{s}(x_2)\hat{y} \lor \hat{x}_2 = \bar{s}(x_1)\hat{y} \} \quad \text{Mul} \quad \{ (\hat{y} = \bar{h} \lor \hat{y} = \ell) \land (\hat{s}_1 = m \land \hat{s}_2 = m) \} \tag{86}
\]

The term \( (\hat{s}_1 = m \land \hat{s}_2 = m) \) indicates that we know that the signs of the two inputs do not change during the occurrence of failure. Note that the conclusion parts in both (85) and (86) are the same, but the premises are different.

**Proof for (86):** Let \( s_1 = sgn(x_1), s_2 = sgn(x_2), s = sgn(y) \), and \( a_1 = |x_1|, a_2 = |x_2|, a = |y| \); thus, \( x_1 = s_1a_1, x_2 = s_2a_2 \), and \( y = sa \). Given that \( a_1, a_2 \in \mathbb{R}^+ \), \( a \) is an increasing function with respect to both \( a_1 \) and \( a_2 \). Thus:

\[
\begin{align*}
(\hat{a} = \bar{h} \lor \hat{a} = \ell) & \Rightarrow (\hat{a}_1 = \bar{a} \lor \hat{a}_2 = \hat{a}) \\
(\bar{s}(s)\hat{y} = \bar{h} \lor \bar{s}(s)\hat{y} = \ell) & \Rightarrow (\bar{s}(s_1)\hat{x}_1 = \bar{s}(s)\hat{y} \lor \bar{s}(s_2)\hat{x}_2 = \bar{s}(s)\hat{y}) \\
(\hat{y} = \bar{h} \lor \hat{y} = \ell) & \Rightarrow (\bar{s}(s_1)\hat{x}_1 = \bar{s}(s_1)\hat{s}(s_2)\hat{y} \lor \bar{s}(s_2)\hat{x}_2 = \bar{s}(s_1)\hat{s}(s_2)\hat{y}) \\
(\hat{y} = \bar{h} \lor \hat{y} = \ell) & \Rightarrow (\hat{x}_1 = \hat{s}(s_2)\hat{y} \lor \hat{x}_2 = \hat{s}(s_1)\hat{y}) \\
(\hat{y} = \bar{h} \lor \hat{y} = \ell) & \Rightarrow (\hat{x}_1 = \bar{s}(x_2)\hat{y} \lor \hat{x}_2 = \bar{s}(x_1)\hat{y}) \\
(\hat{y} = \bar{h} \lor \hat{y} = \ell) & \Rightarrow (\hat{x}_1 = \bar{s}(x_2)\hat{y} \lor \hat{x}_2 = \bar{s}(x_1)\hat{y}) \
\end{align*}
\]

which is the logical expression of the model in (86).

At a different level, where possible (e.g., in system testing scenarios), we can examine the reported values of the variables to draw conclusions. Using truth
tables to group failure scenarios, an alternative model for $Mul$ can be as follows:

\[
\begin{align*}
\{\hat{x}_1 & = h \lor \hat{x}_2 = h \lor (\hat{x}_1 = \ell \land \hat{x}_2 = \ell) \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 < 0 \land x_2 < 0\} & \quad (88) \\
\{\hat{x}_2 & = h \lor (\hat{x}_1 = \ell \land \hat{x}_2 = \ell) \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 < 0 \land x_2 = 0\} & \quad (89) \\
\{\hat{x}_1 & = \ell \lor \hat{x}_2 = h \lor x'_1 = 0 \lor x_2 = 0 \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 < 0 \land x_2 > 0\} & \quad (90) \\
\{\hat{x}_1 & = h \lor (\hat{x}_1 = \ell \land \hat{x}_2 = \ell) \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 = 0 \land x_2 < 0\} & \quad (91) \\
\{\hat{x}_1 = h \land \hat{x}_2 = h \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 = 0 \land x_2 = 0\} & \quad (92) \\
\{\hat{x}_1 = \ell \land (\hat{x}_1 = \ell \land \hat{x}_2 = \ell) \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 = 0 \land x_2 > 0\} & \quad (93) \\
\{\hat{x}_1 = h \lor \hat{x}_2 = \ell \lor x'_1 = 0 \lor x_2 = 0 \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 > 0 \land x_2 < 0\} & \quad (94) \\
\{\hat{x}_2 = \ell \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = \ell \land x_1 > 0 \land x_2 = 0\} & \quad (95) \\
\{\hat{x}_1 = \ell \lor \hat{x}_2 = \ell \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 < 0 \land x_2 < 0\} & \quad (96) \\
\{\hat{x}_1 = \ell \lor \hat{x}_2 = \ell \lor x'_1 = 0 \lor x_2 = 0 \} & \quad Mul \quad \{\hat{y} = h \land x_1 < 0 \land x_2 < 0\} & \quad (97) \\
\{\hat{x}_2 = \ell \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 < 0 \land x_2 > 0\} & \quad (98) \\
\{\hat{x}_1 = \ell \lor \hat{x}_2 = \ell \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 < 0 \land x_2 > 0\} & \quad (99) \\
\{\hat{x}_1 = \ell \lor \hat{x}_2 = \ell \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 = 0 \land x_2 < 0\} & \quad (100) \\
\{\hat{x}_1 = h \land \hat{x}_2 = \ell \} & \quad Mul \quad \{\hat{y} = h \land x_1 = 0 \land x_2 = 0\} & \quad (101) \\
\{\hat{x}_1 = h \lor \hat{x}_2 = h \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 > 0 \land x_2 > 0\} & \quad (102) \\
\{\hat{x}_1 = h \lor \hat{x}_2 = h \lor x'_1 = 0 \lor x_2 = 0 \} & \quad Mul \quad \{\hat{y} = h \land x_1 > 0 \land x_2 < 0\} & \quad (103) \\
\{\hat{x}_2 = h \lor (\hat{x}_1 = h \land \hat{x}_2 = h) \} & \quad Mul \quad \{\hat{y} = h \land x_1 > 0 \land x_2 = 0\} & \quad (104) \\
\{\hat{x}_1 = h \lor \hat{x}_2 = h \lor x'_1 = 0 \lor x_2 = 0 \} & \quad Mul \quad \{\hat{y} = h \land x_1 > 0 \land x_2 > 0\} & \quad (105)
\end{align*}
\]

A.2 Boolean functions

**AND gate:** For $y, x_1, x_2 \in B$, where $x_1, x_2$ are two suspicious variables and $y:= And(x_1, x_2):= x_1 \land x_2$, we propose failure models at three levels. As the minimum conditions, we use:

\[
\{\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y}\} \quad And \quad \{\hat{y} = t \lor \hat{y} = f\} \quad (106)
\]

For certain causes, based on Def. [5] we use the following:

\[
\begin{align*}
\{\hat{x}_1 = \hat{y} \land \hat{x}_2 = \hat{y}\} & \quad And \quad \{\hat{y} = t\} & \quad (107) \\
\{\hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y}\} & \quad And \quad \{\hat{y} = f\} & \quad (108)
\end{align*}
\]

And for value-dependent modeling, we use the following:

\[
\begin{align*}
\{\{\hat{x}_1 = \hat{y} \land x_2 = t\} \lor (\hat{x}_2 = \hat{y} \land x_1 = t)\} & \quad And \quad \{\hat{y} = t\} & \quad (109) \\
\{\{\hat{x}_1 = \hat{y} \land x'_2 = t\} \lor (\hat{x}_2 = \hat{y} \land x'_1 = t)\} & \quad And \quad \{\hat{y} = f\} & \quad (110)
\end{align*}
\]
Proof: Let us begin with value-dependent scenarios (109) and (110). From the definition of \textit{And}, it follows that:

\[
\hat{y} = f \Rightarrow (y = f \land y' = t)
\]

\[
\Rightarrow ((x_1 = f \lor x_2 = f) \land (x'_1 = t \land x'_2 = t))
\]

\[
\Rightarrow ((\hat{x}_1 = f \land \hat{x}_2 = t) \lor (\hat{x}_2 = f \land x'_1 = t)) \quad (111)
\]

\[
\hat{y} = t \Rightarrow (y = t \land y' = f)
\]

\[
\Rightarrow ((x_1 = t \land x_2 = t) \land (x'_1 = f \lor x'_2 = f))
\]

\[
\Rightarrow ((\hat{x}_1 = t \land \hat{x}_2 = t) \lor (\hat{x}_2 = t \land x'_1 = t)) \quad (112)
\]

Statements (111) and (112) logically represent the failure scenarios given in (109) and (110).

The minimum conditions for \(\hat{y} = f\) and \(\hat{y} = t\) simply follow from (111) and (112), as follows:

\[
\hat{y} = f \Rightarrow (\hat{x}_1 = f \lor \hat{x}_2 = f) \quad (113)
\]

\[
\hat{y} = t \Rightarrow (\hat{x}_1 = t \land \hat{x}_2 = t) \quad (114)
\]

which, together, represent the failure scenarios given in (106).

To draw the certain conditions, we use truth tables. Table 1 shows the failure states related to all the possible combinations of inputs, and their corresponding output failure modes.

Let us examine \(\hat{y} = f\) first, which represents a deviation between \(y' = t\) and \(y = f\). We claim that \(\hat{x}_1 = f\) is a certain cause for this output failure. According to Def. 5, we should look for those rows in Table 1 that indicate \(y' = t, \hat{x}_1 = f,\) and \(\hat{x}_2 = m\), which includes Row 8 only. Since \(x_2\) is unchanged in this scenario, the only cause of \(\hat{y} = f\) can be \(\hat{x}_1 = f\). Similarly, we can show that \(\hat{x}_2 = f\) and the conjunction \((\hat{x}_1 = f \land \hat{x}_2 = f)\) are two other certain causes; see Rows 14 and 6, respectively. Therefore, all the certain causes for \(\hat{y} = f\) can be summarized in: \((\hat{x}_1 = f \lor \hat{x}_2 = f)\).

The situation for \(\hat{y} = t\) is somewhat different. From (114), we know that the certain cause for this failure, if any, would be related to \(\hat{x}_1 = t\) and \(\hat{x}_2 = t\). Let us try to see if we can claim \(\hat{x}_1 = t\) as a certain cause. As we did earlier, we need to examine those rows in Table 1 where \(y' = f, \hat{x}_1 = t,\) and \(\hat{x}_2 = m\), which include Rows 9 and 12. However, the output failures in these two rows are not consistent; therefore, we cannot claim that \(\hat{x}_1 = t\) always causes \(\hat{y} = t\). Similarly, we cannot claim that \(\hat{x}_2 = t\) is a certain cause for \(\hat{y} = t\) (Rows 3 and 15). We can only claim \((\hat{x}_1 = t \land \hat{x}_2 = t)\) as a certain cause for \(\hat{y} = t\) (see Row 11).

In summary, the certain causes of failure in the \textit{And} function can be expressed by the following:

\[
\hat{y} = f \Rightarrow (\hat{x}_1 = f \lor \hat{x}_2 = f) \quad (115)
\]

\[
\hat{y} = t \Rightarrow (\hat{x}_1 = t \land \hat{x}_2 = t) \quad (116)
\]

which represent the failure scenarios given in (107) and (108).
Note: both the minimum conditions (given in (113) and (114)) and the certain causes (given in (115) and (116)) are logically correct; however, these two outcomes present two different approaches to reasoning: the former is obviously easier to obtain but provides more disjunctive outcomes, whereas the latter requires more effort to reason but provides more conjunctive terms. For some functions (e.g., \( \text{Not} \)), the minimum conditions and certain causes are the same.

| No. | \( x_1 \) | \( x'_1 \) | \( \hat{x}_1 \) | \( x_2 \) | \( x'_2 \) | \( \hat{x}_2 \) | \( y \) | \( y' \) | \( \hat{y} \) |
|-----|--------|--------|--------|--------|--------|--------|--------|--------|--------|
| 1   | \( f \) | \( f \) | \( m \) | \( f \) | \( f \) | \( m \) | \( f \) | \( f \) | \( m \) |
| 2   | \( f \) | \( f \) | \( m \) | \( f \) | \( t \) | \( f \) | \( f \) | \( f \) | \( m \) |
| 3   | \( f \) | \( f \) | \( m \) | \( t \) | \( f \) | \( t \) | \( f \) | \( f \) | \( m \) |
| 4   | \( f \) | \( f \) | \( m \) | \( t \) | \( t \) | \( m \) | \( f \) | \( f \) | \( m \) |
| 5   | \( f \) | \( t \) | \( f \) | \( f \) | \( f \) | \( m \) | \( f \) | \( f \) | \( m \) |
| 6   | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( f \) | \( m \) |
| 7   | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( f \) | \( m \) |
| 8   | \( f \) | \( t \) | \( f \) | \( t \) | \( t \) | \( m \) | \( f \) | \( t \) | \( f \) |
| 9   | \( t \) | \( f \) | \( f \) | \( f \) | \( m \) | \( f \) | \( f \) | \( m \) |
| 10  | \( t \) | \( f \) | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( m \) |
| 11  | \( t \) | \( f \) | \( t \) | \( t \) | \( t \) | \( t \) | \( f \) | \( t \) | \( t \) |
| 12  | \( t \) | \( f \) | \( t \) | \( t \) | \( m \) | \( t \) | \( f \) | \( m \) |
| 13  | \( t \) | \( t \) | \( m \) | \( f \) | \( f \) | \( m \) | \( f \) | \( f \) | \( m \) |
| 14  | \( t \) | \( t \) | \( m \) | \( f \) | \( t \) | \( f \) | \( t \) | \( f \) | \( m \) |
| 15  | \( t \) | \( t \) | \( m \) | \( t \) | \( f \) | \( t \) | \( f \) | \( t \) | \( m \) |
| 16  | \( t \) | \( t \) | \( m \) | \( t \) | \( t \) | \( m \) | \( t \) | \( t \) | \( m \) |

\[ y = x_1 \land x_2, \quad y' = x'_1 \land x'_2 \]
\[ \hat{x}_1 = \text{md}_x(x_1, x'_1), \quad \hat{x}_2 = \text{md}_x(x_2, x'_2), \quad \hat{y} = \text{md}_x(y, y') \]

**Table 1: Failure truth table for And**

**OR gate:** For \( y, x_1, x_2 \in \mathbb{B} \), where \( x_1, x_2 \) are two suspicious variables and \( y := \text{Or}(x_1, x_2) := x_1 \lor x_2 \), we propose failure models at three levels. As the minimum conditions, we use the following:

\[ \{ \hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y} \} \quad \text{Or} \quad \{ \hat{y} = t \lor \hat{y} = f \} \quad (117) \]

For certain causes, based on Def. [5] we use the following:

\[ \{ \hat{x}_1 = \hat{y} \lor \hat{x}_2 = \hat{y} \} \quad \text{Or} \quad \{ \hat{y} = f \} \quad (118) \]
\[ \{ \hat{x}_1 = \hat{y} \land \hat{x}_2 = \hat{y} \} \quad \text{Or} \quad \{ \hat{y} = t \} \quad (119) \]
and for value-dependent modeling, we use the following:

\[
\begin{align*}
\{(\hat{x}_1 = \hat{y} \land \hat{x}_2 = f) \lor (\hat{x}_2 = \hat{y} \land \hat{x}_1 = f)\} & \text{ Or } \{\hat{y} = t\} \quad \text{(120)} \\
\{(\hat{x}_1 = \hat{y} \land \hat{x}_2 = f) \lor (\hat{x}_2 = \hat{y} \land \hat{x}_1 = f)\} & \text{ Or } \{\hat{y} = f\} \quad \text{(121)}
\end{align*}
\]

\textbf{Proof:} From \(\hat{y} = f\) and \(\hat{y} = t\) we will have:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (y = f \land y' = t) \\
& \Rightarrow ((x_1 = f \land x_2 = f) \land (x'_1 = t \lor x'_2 = t)) \\
& \Rightarrow ((\hat{x}_1 = f \land x_2 = f) \lor (\hat{x}_2 = f \land x_1 = f)) \quad \text{(122)} \\
\hat{y} = t & \Rightarrow (y = t \land y' = f) \\
& \Rightarrow ((x_1 = t \lor x_2 = t) \land (x'_1 = f \land x'_2 = f)) \\
& \Rightarrow ((\hat{x}_1 = t \land x'_2 = f) \lor (\hat{x}_2 = t \land x'_1 = f)) \quad \text{(123)}
\end{align*}
\]

(122) and (123) prove the value-dependent models in (120) and (121). From (122) and (123), we can also conclude the following:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (\hat{x}_1 = f \lor \hat{x}_2 = f) \quad \text{(124)} \\
\hat{y} = t & \Rightarrow (\hat{x}_1 = t \lor \hat{x}_2 = t) \quad \text{(125)}
\end{align*}
\]

which together prove (117).

To demonstrate the correctness of (118) and (119), we will use truth tables. Table 2 is the failure truth table for an Or function.

From (125), we know that the certain cause for this failure is related to \(\hat{x}_1 = t\) and \(\hat{x}_2 = t\). For \(\hat{x}_1 = t\) as a certain cause, we need to examine those rows in Table 2 where \(y' = f\), \(\hat{x}_1 = t\), and \(\hat{x}_2 = m\), which only includes Row 9. Since \(x_2\) is unchanged in this scenario, the only cause of \(\hat{y} = t\) can be \(\hat{x}_1 = t\). Similarly, we can show that \(\hat{x}_2 = t\) and the conjunction \(\hat{x}_1 = t \land \hat{x}_2 = t\) are two other certain causes (see Rows 3 and 11, respectively). For \(\hat{y} = f\), the only certain cause will be in Row 6. The output failure is not consistent in the other rows with only one of the two input failures \(\hat{x}_1 = f\) and \(\hat{x}_2 = f\). In summary, the certain causes of failure in Or function can be as follows:

\[
\begin{align*}
\hat{y} = f & \Rightarrow (\hat{x}_1 = f \land \hat{x}_2 = f) \quad \text{(126)} \\
\hat{y} = t & \Rightarrow (\hat{x}_1 = t \lor \hat{x}_2 = t) \quad \text{(127)}
\end{align*}
\]

which represent the logical form of the model in (118) and (119).

\textbf{Boolean negation:} For \(y, x \in \mathbb{B}\), where \(x\) is a suspicious variable and \(y := \text{Not}(x)\) := \(\neg x\):

\[
\{\hat{x} = \sim \hat{y}\} \quad \text{Not} \quad \{\hat{y} = a\} \quad \text{(128)}
\]

\textbf{Proof:} Follows from the definition of \(\neg\) and \(\sim\).
Table 2: Failure truth table for Or

A.3 Combined functions

Greater comparison: Let \( x_1, x_2 \in \mathbb{R} \) and \( y \in \mathbb{B} \) and \( y := G_{com}(x_1, x_2) := (x_1 > x_2) \). Where both \( x_1 \) and \( x_2 \) are suspicious, the minimum conditions for a failure at \( y \) are modeled as follows:

\[
\begin{align*}
\hat{x}_1 &= h \lor \hat{x}_2 = l \qquad G_{com} \quad \{\hat{y} = t\} \\
\hat{x}_1 &= l \lor \hat{x}_2 = h \qquad G_{com} \quad \{\hat{y} = f\}
\end{align*}
\]

If either \( x_1 \) or \( x_2 \) is certain, its corresponding sentence can be removed from the above disjunctions, and the other variable becomes the certain cause.

Proof: The correctness of minimum conditions (129) and (130) follows from the definition of \( G_{com} \): if the output \( y \) is faulty, it can only due to a fault at \( x_1 \) or \( x_2 \), and only in the fault directions given here in (129) and (130). It is also obvious that if either input is certain, then the only cause of the fault could be the other input and, again, in the direction of fault indicated in (129) and (130). Readers can also refer to [19] for a proof of single suspicious variable scenarios based on failure truth table.
**Less comparison:** Let \( x_1, x_2 \in \mathbb{R} \) and \( y \in \mathbb{B} \) and \( y := Lcom(x_1, x_2) := (x_1 < x_2) \).

Where both \( x_1 \) and \( x_2 \) are suspicious, the minimum conditions for a failure at \( y \) are modeled as follows:

\[
\begin{align*}
\{ \hat{x}_1 = \ell \lor \hat{x}_2 = h \} & \quad Lcom \quad \{ \hat{y} = t \} \quad (131) \\
\{ \hat{x}_1 = h \lor \hat{x}_2 = \ell \} & \quad Lcom \quad \{ \hat{y} = f \} \quad (132)
\end{align*}
\]

If either \( x_1 \) or \( x_2 \) is certain, its corresponding sentence can be removed from the above disjunctions, and the other variable becomes the certain cause.

*Proof:* Similar to \( Gcom \).