

Inferring Sensor Placement Using Critical Pairs and Satisfiability Modulo Theory

Alexander Diedrich ✉ 


Helmut-Schmidt-University, Hamburg, Germany

René Heesch ✉ 

Helmut-Schmidt-University, Hamburg, Germany

Marco Bozzano ✉ 

Fundazione Bruno Kessler, Trento, Italy

Björn Ludwig ✉ 

Helmut-Schmidt-University, Hamburg, Germany

Alessandro Cimatti ✉ 

Fondazione Bruno Kessler, Trento, Italy

Oliver Niggemann ✉ 

Helmut-Schmidt-University, Hamburg, Germany

Abstract

Industrial fault diagnosis exhibits the perennial problem of reasoning with partial and real-valued information. This is mainly due to the fact that in real-world applications, industrial systems are only instrumented insofar, as sensor information is required for their functioning. However, such instrumentation leaves out much information that would be useful for fault diagnosis. This is problematic since consistency-based fault diagnosis uses available information and computes intermediate values within a system description. These values are then used to compare expected normal behaviour to actual observed values. In the past, this was done only for Boolean circuits. Recently, satisfiability modulo non-linear arithmetic (SMT) formulations have been developed that allow the calculation of real values, instead of only Boolean ones. Leveraging those formulations, we in this article present a novel method to infer missing sensor values using an SMT system description and the notion of critical pairs. We show on a running example and also empirically that we can infer novel measurements for five process industrial systems. We conclude that, although SMT calculations accumulate some error, we can infer novel optimal measurements for all systems.

2012 ACM Subject Classification Computing methodologies → Causal reasoning and diagnostics; Computing methodologies → Artificial intelligence

Keywords and phrases Sensor Placement, Satisfiability Modulo Theory, Critical Pairs, Diagnosability

Digital Object Identifier 10.4230/OASICS.DX.2024.9

Funding This research as part of the project EKI is funded by dtec.bw – Digitalization and Technology Research Center of the Bundeswehr which we gratefully acknowledge. dtec.bw is funded by the European Union – NextGenerationEU.

Acknowledgements This work has also benefitted from Dagstuhl Seminar 24031 “Fusing Causality, Reasoning, and Learning for Fault Management and Diagnosis.”

1 Introduction

Consistency-based fault diagnosis identifies faults by comparing predictions from a logical system description to actual observations from a system. In the past, the system description was specified by experts, usually in propositional or predicate logic and often solely for Boolean circuits [39]. It was also assumed that the system description is a complete model



© Alexander Diedrich, René Heesch, Marco Bozzano, Björn Ludwig, Alessandro Cimatti, and Oliver Niggemann;

licensed under Creative Commons License CC-BY 4.0

35th International Conference on Principles of Diagnosis and Resilient Systems (DX 2024).

Editors: Ingo Pill, Avraham Natan, and Franz Wotawa; Article No. 9; pp. 9:1–9:19

OpenAccess Series in Informatics



Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

of the underlying system [9] and thus models all the system’s normal behaviour (weak-fault model). However, recently several publications have shown the possibility to learn at least parts of the system description from data for hybrid, i.e. discrete and continuous industrial systems [16, 36, 28, 17]. The challenge with those approaches is that they are neither sound nor complete, i.e., they do not guarantee that the complete behaviour of the system is captured. This is due to two reasons: (i) Many of the methods still have severe limitations, (ii) Often observations about the system are missing. Usually, these approaches use a special formulation for the system description of the form

$$\bigwedge_{c_i \in COMPS_k} c_i \rightarrow obs_i \quad (1)$$

for some component $c_i \in COMPS_k$, with $COMPS_k \subset COMPS$ being a subset of the components whose behaviour has an influence on the observables $obs_i \in OBS$. In other words, those approaches reason against the flow of causality. They conclude from anomalous observations back towards possible components causing the anomaly. To obtain Eq. (1) it is necessary to first identify which observables exist within a system (i.e., the system’s sensors). Then, system identification and causal reasoning techniques are used to identify those components that have an influence on a single observable. And finally, they use this knowledge to formulate propositional logic system descriptions. The problem with this approach is: since those methods only rely on the observable data, the resulting system descriptions may not be able to distinguish each fault. This means faults in some components may be indistinguishable from each other, thus increasing the diagnostic load of service technicians.

A system description that contains components that are not fully diagnosable suffers from partial observability. Observability can only be increased through the addition of sensors. However, possible locations to add sensors increase exponentially with system size and heavily depend on system complexity. Further, some locations may garner more information than other locations, while some locations may be redundant to already existing sensors.

In this article, we present a solution to infer the placement of new sensors based on three main ideas: (i) For modelling the system, we use the more expressive satisfiability modulo theory with non-linear arithmetic (SMT- \mathcal{NRA}) instead of propositional logic. (ii) We compute critical pairs to infer which components are not diagnosable, given a system description, and (iii) we compute the exact locations where to add new sensors as well as the operating range of the sensors’ values for non-faulty behaviour. We require the use of SMT- \mathcal{NRA} to be able to compute the non-linear behaviour of most components that are used in industry. Previous work has already shown how SMT- \mathcal{LRA} , i.e., SMT using linear arithmetic can be used for fault diagnosis [15]. Critical pairs are a common technique used to identify non-diagnosable components within logical knowledge bases [2]. They rely on system traces, meaning temporally ordered sequences of observations that establish, whether some trace exists that can identify a component unambiguously. By computing critical pairs, we can therefore identify a set of components that is non-diagnosable. Following some of the author’s previous work [3] we then show how, given an SMT- \mathcal{NRA} system description, we can compute the possible normal operating ranges of the sensors that would be added for each of the non-diagnosable components. In the end, we can therefore determine which sensors may be added to the system and what the minimum normal operating ranges are. The greatest weakness of our approach is that through the computation of intermediate values through SMT- \mathcal{NRA} significant errors may be introduced depending on the lengths of the paths through the system [18].

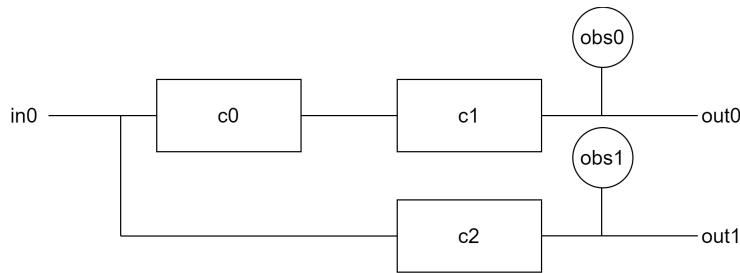
Overall, this article makes the following contributions:

- We show how to compute critical pairs based on a system description for consistency-based diagnosis formulated in SMT- \mathcal{NRA} .
- We present a novel method to compute normal operating ranges for each possible sensor location as well as minimality criteria for optimal placement.
- We present the novel algorithm SYNTHESIZEOBS that first computes critical pairs, then suggests novel observables, and finally selects optimal observables and their normal operating ranges that should be added to an existing system.

With our solution, we thus help practitioners to (i) determine whether their system description is fully diagnosable, and (ii) how to select the optimal sensor placement to increase diagnosability.

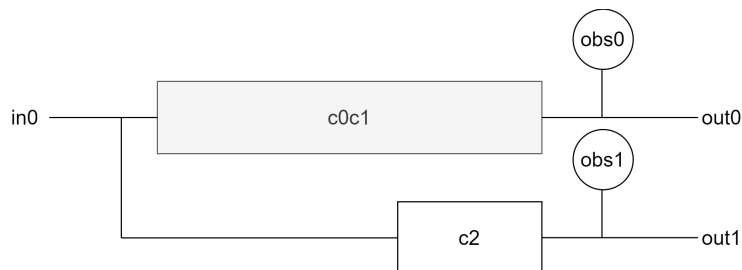
2 Running Example

Figures 1 to 3 present the running example that we will use throughout the rest of the article. It is a simple system with three components and two observables.



■ **Figure 1** The running example with three components and two observables.

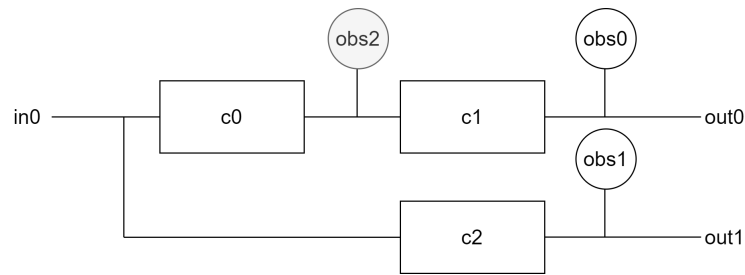
Looking at Figure 1 we assume one input connected to components $c0$ and $c2$, with component $c0$ connected with $c1$. The outputs $out0$ and $out1$ of components $c1$ and $c2$ are monitored by the observables $obs0$ and $obs1$. Obviously, since the input, as well as the connection between components $c0$ and $c1$ is unmonitored, if some fault is recognised in $obs0$ both, $c0$ or $c1$ may be responsible.



■ **Figure 2** The running example with the virtual component $c0c1$.

So for practical purposes, $c0$ and $c1$ may be interpreted as being some virtual component $c0c1$ as depicted in Figure 2.

Only through the introduction of some additional observation (Figure 3) can the observability be increased and thus faults in $c0$ and $c1$ be unambiguously distinguished.



■ **Figure 3** The running example with an additional, synthetically generated, observation to increase the system’s observability.

3 Related Work

Within the fields of diagnosability research and fault diagnosis four different categories can be identified: diagnosability of discrete-event systems, analysing hybrid systems (i.e., systems with discrete and continuous behaviour), fault diagnosis itself, and integration with cyber-physical systems. In all of those fields the goal is to investigate whether a model is suitable for diagnosis, while the type of model varies from automata-like approaches, physical equations, logical approaches, and formal specifications.

Diagnosability of discrete-event systems is an important area of research for many academic and industrial use-cases. Usually, those systems are modelled using Petri-Nets [7], automata [33], or similar approaches that divide a systems’ operation into states and events. With those modelling formalisms it is necessary that practitioners know which places and states can be visited, whether the model correctly models the real systems under investigation, and, in the case of fault diagnosis, if all faults can be distinguished. One of the first authors who analysed diagnosability of discrete event systems systematically was the approach of Sampath et al. [41] in 1995 that they soon extended with active approaches [40]. A brief history of diagnosability research and its relation to diagnosis is given by Lafortune et al. [30]. More recently, Grastien and others [26, 51] have investigated the diagnosability of discrete-event systems with uncertain observations. Boussif et al. [5] have extended diagnosability analysis to the recognition of intermittent faults with the help of a digital twin model. Bittner et al. [2] have developed a comprehensive work on the formal analysis of discrete-event system diagnosability. From the same research group, Bozzano et al. [6] have investigated the testing for diagnosability. Another recent work has been published by Lamperti et al. [31] on sequence oriented discrete event systems.

Hybrid systems are more complex than discrete event systems. And while in several real world use-cases the continuous behaviour can often be abstracted away or can be roughly approximated, in many other use-cases this is impossible. Therefore, also the diagnosability of models used for hybrid systems needs to be investigated. Benedetto et al. [13] investigated the diagnosability of hybrid automata. Trave-Massuyes et al. [47] has analysed the diagnosability of analytical redundancy relations and sensor placement. Those relations are common in control theoretic approaches to fault diagnosis (for details, refer [46]). They have also investigated the sensor placement problem [45], which answers the question of where to place sensors within a system to improve diagnosability. Cordier et al. [11] has compared approaches between discrete system diagnosability and approaches to hybrid system diagnosability. The work of Trave-Massuyes was extended first through Nejjari et al. [37] and later through Verdieri et al. [49]. Fourlas et al. [24] were one of the first to specify hybrid system diagnosability in the terminology of discrete-event system diagnosability. Although

this was later taken up by Bayouhd et al. [1] as well. Diene et al. [19] have developed an approach to analyse the diagnosability of linear time invariant systems. Leal et al. [32] have developed an algorithm to check the diagnosability of hybrid systems specified through analytical redundancy relations. Vignolles et al. [50] have analysed diagnosability of hybrid systems with a special emphasis on fault diagnosis and remaining useful life prognosis. Also from a control theoretic standpoint, a recent approach has been introduced by Su et al. [44] where they investigated the sensitivity of residuals with regard to fault isolability. In line with residuals, fault diagnosability was done with structural analysis as described by Krysander et al. [29] and within the seminal work of Escobet et al. [22].

The notion of fault diagnosis itself in this article is in line with the definitions by De Kleer [12] and Reiter [39]. The theory behind fault diagnosis was recently well explained by Feldman et al. [23], Metodi et al. [35], and Stern et al. [43]. Our notion of system descriptions was introduced in earlier work by Diedrich et al. [18] and follows a line of abductive reasoning by Pill et al. [38] and Bochman et al. [4].

In this article, the goal is to apply the diagnosability analysis to system descriptions in order to infer novel observables in cyber-physical systems. In the area of cyber-physical systems research, aspects of these are touched in the verification of cyber-physical system research. Mehrabian et al. [34] have presented an approach for the run-time verification for uncertainties in cyber-physical systems. Caimilli et al. [8] has investigated the verification of cyber-physical systems when they are reconfigured in the presence of faults. Varela-Vaca et al. [48] have proposed a framework for diagnosing and verifying security requirements. While Dowdeswell [20] has shown how to create function blocks for programmable logic controllers' programming that are diagnosable by design.

Summary. Several research fields are concerned with analysing the diagnosability of their respective models. Most important for this article are those using the Dulmage-Mendelsohn decomposition, i.e., approaches that analyse the diagnosability and sensor placement of analytical redundancy relations. This was primarily done by Krysander et al. [29] and Trave-Massuyes [47]. Concretely, we use the formulation of critical pairs from Bittner et al. [3].

4 Background

We work within the setting of standard first-order logic and employ the standard semantic notions of interpretation, model, satisfiability, and validity. A term is defined as either a constant, an individual variable, or an n -ary function symbol applied to n terms. An atom is defined as either a propositional variable or an n -ary predicate applied to n terms. Subsequently, formulae are defined as either atoms, or as the application of the standard Boolean connectives $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ to formulae, or the application of a quantifier \exists, \forall to one or more variables and a formula.

4.1 Logical Modelling

Satisfiability Modulo Theories (SMT) [10] addresses the problem of determining whether a satisfying assignment to the free variables exists in a first-order formula, where non-logical symbols are interpreted within a decidable background theory \mathcal{T} .

In first-order logic, an atomic formula is the simplest type of formula that cannot be broken down into smaller logical components and is used to construct more complex formulas.

► **Definition 1** (Atomic logical formula). *An atomic logical formula is an expression consisting of one predicate applied to terms and has a Boolean truth value in $\{\top, \perp\}$.*

Within this work, we focus on the theory of *Nonlinear Arithmetic over the reals* (\mathcal{NRA}). The theory of \mathcal{NRA} is the first-order theory with equality whose atoms are nonlinear polynomial constraints interpreted over \mathbb{R} . We have numerical constants and real-valued variables. Using the standard function symbols $+$, $-$, \cdot , \div , these numerical constants and real-valued variables can be combined to form arithmetic terms.

► **Definition 2** (Arithmetic term). *An arithmetic term in SMT is an expression consisting of a variable, a numerical constant, or combinations thereof connected through $\{\cdot, +, -, \div\}$.*

With the predicate symbols $<, \leq, >, \geq, =$, arithmetic terms can be combined with numerical constants or other arithmetic terms to form arithmetic formulae. Depending on the number of combinations, the resulting formula can be an atomic arithmetic formula.

► **Definition 3** (Atomic arithmetic formula). *An atomic arithmetic formula is an expression consisting of two arithmetic terms connected through $\{<, \leq, >, \geq, =\}$ and has a Boolean truth value in $\{\top, \perp\}$.*

SMT allows the combination of first-order logical and theory specific formulas with the logical connectives $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$ in order to express complex contexts.

► **Definition 4** (SMT- \mathcal{NRA} formula). *An SMT- \mathcal{NRA} formula denotes the combination of (multiple) atomic first-order logical formulas and (multiple) atomic arithmetic formulas, connected via the logical connectives $\neg, \wedge, \vee, \rightarrow, \leftrightarrow$.*

If φ_0, φ_1 are atomic formulae or SMT formulae, then $\varphi = \varphi_0 \oplus \varphi_1$ with $\oplus \in \{\wedge, \vee, \rightarrow, \leftrightarrow, =, \leq, \geq\}$ is also an SMT formula. This SMT formula φ is considered satisfiable if and only if it has a model M .

► **Definition 5** (SMT Model). *A model M of a formula φ is defined as a pair consisting of an assignment, which maps each variable to an element of its domain, and an interpretation of the non-logical symbols that satisfies the formula.*

► **Definition 6** (Satisfiability). *An formula φ is satisfiable, when a model M exists, where $[[\bigwedge_{\varphi_i \in \varphi} \varphi_i]]_M = \top$.*

The satisfiability of an SMT formula φ can be checked by an SMT solver. An SMT solver is a software tool designed to determine whether a given formula is satisfiable; it may either provide a satisfying assignment or a proof of unsatisfiability. Most modern SMT solvers employ a lazy approach: they first use a SAT solver to enumerate potential satisfying assignments of the formula's Boolean abstraction. Then, theory solvers verify the satisfiability of the corresponding set of constraints for each assignment. If the theory solver finds the set of constraints consistent, a model is established. Otherwise, new clauses are generated to prevent the recurrence of assignments that lead to inconsistencies.

Especially in the context of fault diagnosis, often it is not only relevant to determine the satisfiability of an SMT formulae or the proof of unsatisfiability, but also to find the maximum satisfiability set of a formula φ that is unsatisfiable.

► **Definition 7** (Maximum Satisfiability). *A maximum satisfiable set (mss) of a set of SMT expressions φ are those expressions that can be satisfied even though φ itself is unsatisfiable, with $mss = \operatorname{argmax}_{\alpha, M} \varphi$, where α is a complete assignment to all variables within φ , using a single model M .*

4.2 Fault Diagnosis

This section presents the theoretical background on i) how to obtain observations $OBS : \mathbb{R}^{n \times m}$, $n, m \in \mathbb{N}$ from some system, and ii) how a given system description can be used to perform diagnosis.

We begin by defining a single measured value from a cyber-physical system as a sensor reading $m_i \in \mathbb{R}$. The explicit mention of time is omitted, as we assume that all sensor values are read concurrently at a single point in time, given a suitable sampling rate. Therefore, we write m_i instead of $m_i(t)$. Observations are discretised sensor readings that we define by some suitable function $d(\cdot)$.

► **Definition 8** (Observation). *A single observation $obs \in \{\top, \perp\}$ is a discretised assignment $obs_i = d(m_i, \lambda)$ to some or all inputs and outputs of a cyber-physical system, where $d(\cdot)$, with $d : \mathbb{R} \rightarrow \{\top, \perp\}$, is a suitable discretisation function, $\lambda \in \mathbb{R}^{|\lambda|}$ are parameters, and $m_i \in \mathbb{R}$.*

We will write $|\lambda|$ to denote the cardinality of λ . The set of all observations is denoted as OBS . In other words, we require the existence of a function $d(\cdot)$ that discretises a real-valued input with the parameters λ into a Boolean value. Usually, function $d(\cdot)$ is realised through residual values [27]. A residual takes some of the observable parameters of a cyber-physical system in the form of input, output, and state variables and outputs 0 in the fault-free case and $\neq 0$, otherwise. Formally, a residual is a function

$$r(m_i, \lambda) = \begin{cases} 0; & \text{no fault} \\ \neq 0; & \text{else} \end{cases} \quad (2)$$

with sensor value $m_i \in \mathbb{R}$ and $\lambda \in \mathbb{R}^{|\lambda|}$ is a set of parameters. Usually, the set of parameters λ contains guard conditions around the input, output, and state variables. The set of all residuals is denoted as \mathcal{R} . Several existing approaches use observer or structural equations to calculate residual values, but rely on expert-defined first-principles models. [25] provide a good overview over existing methods to generate residuals.

Example. An implementation of the discretization function in the form of a residual can be done through, for example, threshold values τ , such that $d(m_i, \tau) : \tau_l \leq m_i \leq \tau_h \rightarrow obs_i$, where $\tau_l, \tau_h \in \mathbb{R}$ are some threshold values, obs_i is a symbol denoting the observation, and $m_i \in \mathbb{R}$ is the denoised sensor value. In previous work, we showed that such threshold equations can be translated into propositional logic through satisfiability modulo linear arithmetic theory [15].

With residuals, real-valued signals from a cyber-physical system are discretised into Boolean observations. But how is it possible to use these for fault diagnosis? Fault diagnosis algorithms are employed to identify inconsistencies between the observations from a cyber-physical system and its system description. Formally, a diagnostic system is defined as follows.

► **Definition 9** (Diagnostic System). *A diagnostic system is a three-tuple $(SD, COMPS, OBS)$, where SD is the system description formulated in propositional logic, $COMPS$ is the set of components, and OBS are the Boolean observations grounded as propositional symbols, such that $SD \cup COMPS \cup OBS$ is satisfiable.*

In order to perform fault diagnosis, non-monotonic and abductive reasoning are employed. This allows the system description to reason with an inconsistent knowledge base and conclude from observations to causes. Reiter [39] called non-monotonic reasoning default

reasoning whose main benefit is that it contains logical statements that are assumed to be true unless proof to the contrary exists. Diagnosis uses non-monotonic reasoning to retract those assumptions within SD (i.e. those components that might have caused some observation to deviate from its normal behaviour) until a consistent diagnostic system is achieved. From those retracted assumptions the diagnosis is then computed.

► **Definition 10** (Diagnosis). *A diagnosis ω is a set derived from the set of retracted assumptions, which satisfies $SD \wedge OBS \wedge COMPS_\omega \wedge COMPS_{-\omega}$*

where $COMPS_\omega$ are the faulty components and the set $COMPS_{-\omega}$ are the remaining healthy components. Within this article we use a compilation-based approach and compute the diagnosis by calculating the maximum satisfiable set within an SMT- \mathcal{NRA} system description. In many cases, the size of individual diagnoses can be quite large and sometimes contain hundreds of components. Therefore, the size of a diagnosis needs to be limited. A minimum cardinality diagnosis is a diagnosis that contains the smallest possible number of components.

► **Definition 11** (Minimum Cardinality Diagnosis). *A Minimum Cardinality Diagnosis is a diagnosis ω' , so that $|\omega'| \leq |\omega|$ for each diagnosis.*

Whenever we employ a diagnosis algorithm, the goal is always to find the minimum cardinality diagnosis.

5 Problem Description

To diagnose faults in technical systems, a system description is required that describes the system's normal behaviour. For this work, we assume the existence of an SMT- \mathcal{NRA} system description. The system description is divided into two parts: The first part describes each component's behaviour and the second part describes the observables, modelling the evaluation of each observable value. More formally, we divide SD into $SD = \mathcal{B} \cup \mathcal{O}$, for the SMT- \mathcal{NRA} behavior description \mathcal{B} and the observable description \mathcal{O}

For modelling \mathcal{B} we extend the formalism that was originally suggested by Feldman et al. [23] for propositional logic

$$h_i \rightarrow out_i = g_i(\mathbf{in}_i) \tag{3}$$

where $h_i \in \{\top, \perp\}$ indicates the health of the respective component with index $i \in \mathbb{N}$. The function $g_i(\cdot) : \mathbb{R}^n \rightarrow \mathbb{R}$ computes the output $out_i \in \mathbb{R}$ from inputs $\mathbf{in}_i \in \mathbb{R}^n$, with index $i \in \mathbb{N}$ and number of input signals $n \in \mathbb{N}$. Overall, Equation (3) is said to be a weak-fault model of the system, i.e., it describes only the system's normal behaviour.

To observe whether the system behaves normally, we need to add the second part of the system description \mathcal{O} , using the observables $obs \in OBS$. We therefore use expressions of the form

$$h_i \rightarrow obs_i = d(out_i, \theta_i) \tag{4}$$

with observable $obs_i \in \{\top, \perp\}$, input $out_i \in \mathbb{R}$, a set of parameters $\theta \in \mathbb{R}^n$, and an observation evaluation function $d(\cdot) : \mathbb{R} \rightarrow \{\top, \perp\}$, where $h_i \in \{\top, \perp\}$ indicates the health of the sensor.

The problem we tackle in this article is two-fold: (i) We investigate which components within a given system description SD can be diagnosed given the available observations. In other words, which observations will allow us to diagnose a fault in a component unambiguously, such that only the real faulty component is identified? For this we use a single-fault

assumption such that we assume only one component is faulty at a given time. (ii) Given a set of non-diagnosable components, which observables do we need to increase the system's observability and decrease the set of non-diagnosable components?

To illustrate the problems we use the running example with an exemplary and simplified system descriptions such as

$$h_0 \rightarrow int_0 = g_0(in_0) \quad (5)$$

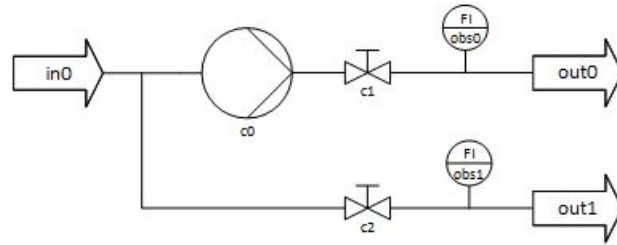
$$h_1 \rightarrow out_0 = g_1(int_0) \quad (6)$$

$$h_2 \rightarrow out_1 = g_2(int_0) \quad (7)$$

$$h_a \rightarrow obs_0 = d(out_0, \theta_0) \quad (8)$$

$$h_b \rightarrow obs_1 = d(out_1, \theta_0) \quad (9)$$

We can now assume concrete realisations of functions g, d . For this, let's assume in the running example we have a process industrial system with a pump c_0 , and valves c_1, c_2 .



■ **Figure 4** The running example using real physical components.

Using this, we can specify the physical modelling functions $g(\cdot)$ as part of the behaviour system description \mathcal{B} through the equations for tanks and valves. For pumps, we use the model

$$\Delta P = P_{max} - R_p Q \quad (10)$$

where $Q \in \mathbb{R}$ is the flow rate, $\Delta P \in \mathbb{R}$ is the pressure difference, $P_{max} \in \mathbb{R}$ is the maximum pressure the pump can generate, and $R_p \in \mathbb{R}$ is the pump resistance. For a valve, we can for simplicity assume the following model

$$Q = Cx\Delta P \quad (11)$$

where $Q \in \mathbb{R}$ is the flow rate, $\Delta P \in \mathbb{R}$ is the pressure difference, $C \in \mathbb{R}$ is the valve coefficient, and $x \in [0 \dots 1]$ is the valve opening.

We then model the observables within part \mathcal{O} of the system description through

$$obs_i = \tau_l \leq out_i \leq \tau_h \quad (12)$$

with the thresholds $\tau_l, \tau_h \in \theta_i$.

Overall, we can now specify the complete system description as SMT- \mathcal{NRA} expressions

$$\text{Pump } c_0: h_0 \rightarrow int_0 = P_{max} - R_p in_0 \quad (13)$$

$$\text{Valve } c_1: h_1 \rightarrow out_0 = Cx\Delta int_0 \quad (14)$$

$$\text{Valve } c_2: h_2 \rightarrow out_1 = Cx\Delta in_0 \quad (15)$$

$$\text{Observable } obs_0: h_a \rightarrow obs_0 = \tau_{l_0} \leq out_0 \leq \tau_{h_0} \quad (16)$$

$$\text{Observable } obs_1: h_b \rightarrow obs_1 = \tau_{l_1} \leq out_1 \leq \tau_{h_1} \quad (17)$$

It is evident that by placement of the observables component c_0 (the pump) cannot be diagnosed. In what follows we present first a method that establishes the diagnosability of systems using these types of system descriptions and then, in section 7, we will describe how to infer additional sensor values to increase a system's diagnosability.

6 Finding Critical Pairs for Diagnosability in SMT- \mathcal{LRA} System Descriptions

Historically, diagnosability can be reduced to the search for critical pairs [3, 2]. A critical pair is defined based on the definition of traces (formalizing an execution of the plant).

► **Definition 12 (Plant).** *A plant P is a symbolic transition system $\langle V, V_o, I, T \rangle$, where V is a finite set of state variables; $V_o \subseteq V$ is the set of observable state variables; I is a formula over V defining the initial states, and T is a formula over V, V' (with V' being the next version of the state variables) defining the transition relation.*

A state s is an assignment to the state variables V . The *observable part* $obs(s)$ of a state s is the projection of s on the subset V_o of observable state variables.

► **Definition 13 (Trace).** *A system trace is an infinite sequence of states s_0, s_1, s_2, \dots starting from s_0 such that, for each $k \geq 0$, $\langle s_k, s'_{k+1} \rangle$ satisfies T .*

The observable part of a trace $\pi = s_0, s_1, s_2, \dots$, denoted $obs(\pi)$, is the sequence $obs(s_0), obs(s_1), obs(s_2), \dots$

In this paper, we are interested in diagnosing the faulty status of components, i.e., a diagnosis for component c can be defined as the healthy status h_i of the i -th component. We give the following definition for critical pairs.

► **Definition 14 (Critical Pair).** *Given a plant P , a diagnosis condition c and a delay d , a critical pair for P and c at i is a pair of traces π_1 and π_2 of length $i + d$ such that $obs(\pi_1) = obs(\pi_2)$ (traces are observationally equivalent), $\pi_1, i \models c$ (trace π_1 satisfies c at i), and $\pi_2, j \not\models c$ for all j such that $j \in [i - d, i + d]$ (trace π_2 does not satisfy c in the interval $[i - d, i + d]$).*

Our definition of diagnosability corresponds to the notion of *bounded delay diagnosability*, i.e., diagnosability within a delay d , following [2].

Searching for critical pairs can be done using the twin-plant construction [2], and can be implemented using an SMT solver, by unrolling the transition relation and looking for a pair of traces satisfying the conditions of Def. 14.

7 A Novel Method to Synthesize new Observables

We have seen in the last section how critical pairs identify those components that, given two observable traces, cannot be distinguished in terms of their faulty behaviour. In this section, we describe a novel method to synthesize new minimal observables, given the SMT- \mathcal{NRA} system description SD and a set of critical pairs.

The intuition behind our approach is the following: We assume that there exists an SMT- \mathcal{NRA} system description. Then we first compute the intermediate values (assumables) that are the output of the non-observable component through logical inference. For each of those outputs we then attempt to compute a new observable that is described through a logical expression template and its historical values.

We therefore use the following definition.

► **Definition 15** (Novel observable). *A novel observable is an additional measurement $obs_i \in \mathbb{R}$ with an associated range $r_l, r_h \in \mathcal{R}$ with $r_l \leq obs_i \leq r_h$ during normal operating conditions.*

we say that the range of a variable is defined through the two functions

$$obs_{i_l} = \underset{in_i}{\operatorname{argmin}} g(in_i, \theta_i) \quad (18)$$

and

$$obs_{i_h} = \underset{in_i}{\operatorname{argmax}} g(in_i, \theta_i) \quad (19)$$

where argmin and argmax take the ranges of the normal operating conditions, given by the underlying system. In other words, the values of in_i are constrained by the values seen in historical data under normal working conditions. The calculation of these intermediate values is easy, since we use SMT- \mathcal{NRA} for the behaviour models of all components. Thus, we can calculate the outputs of each component up to an accuracy limited by the SMT solver.

To combat the accumulation of errors in the sequential computation of ranges [18] we introduce several heuristics. The first is the range-minimal novel observable.

► **Definition 16** (Range-minimal Novel observable). *A range-minimal novel observable is an additional measurement $obs_i \in \mathbb{R}$ with additional constraints limiting the range $r_l, r_h \in \mathcal{R}$ with $r_l \leq obs_i \leq r_h$.*

We use this observable to obtain those novel observables that have the smallest normal operating range and thus are assumed to be the most sensitive to faults. In other words, we want those observables whose range does not include too much operational variability.

Another helpful definition is the count-minimal novel observable. Using these observables the highest number of critical pairs can be removed and the maximum number of components become diagnosable.

► **Definition 17** (Count-minimal Novel observable). *A Count-minimal novel observable is an additional measurement $obs_i \in \mathbb{R}$ with range $r_l, r_h \in \mathcal{R}$ with $r_l \leq obs_i \leq r_h$ that removes the maximum number of critical pairs.*

The range- and count-minimal novel observable can be determined using weak-fault models, i.e., models only describing the normal working behaviour of the system [23]. These are always calculated first. It is then possible to apply heuristics to these novel observables in order to improve the diagnostic performance.

To obtain heuristics, we require more parameters than a common weak-fault model provides. After having calculated the range- or count-minimal observables we can thus calculate those observables that provide the maximum information gain. The intuition is as follows: Assuming we want to add those observables that provide the maximum information gain, we first need to determine how much more we can diagnose, if specific sensors are added. In earlier work, we demonstrated that such information can be obtained using entropy and information-gain calculations [14]. To calculate the entropy we require information about the reliability of single components which usually can be obtained either from experience or through design documentation such as mean-time-between-failure analysis.

► **Definition 18** (Entropy-maximal Novel observable). *An entropy-maximal novel observable is an additional measurement $obs_i \in \mathbb{R}$ with range $r_l, r_h \in \mathcal{R}$ with $r_l \leq obs_i \leq r_h$ whose value adds the maximal amount of knowledge about the system.*

We suggest an entropy-based approach [42] to reduce critical pairs by introducing an entropy-maximal novel observable. An entropy-based approach facilitates that an observable leads to a maximal information gain. We assume that the set of observables OBS can be divided into two subsets OBS_u, OBS_c , with $OBS_u \cap OBS_c = \emptyset$, and $OBS_u \cup OBS_c = OBS$, where $OBS_u = \{obs_0, obs_1, \dots, obs_{n-1}\}$, $n \in \mathbb{N}$ is the set of sensors currently used to diagnose the system and $OBS_c = \{obs_{n+1}, obs_{n+2}, \dots, obs_m\}$, $n, m \in \mathbb{N}$ is the set of candidate sensors that could be added to the system (i.e. the determined range-minimal, or count-minimal observables). The intuition now is to calculate the entropy based on the current sensor setup OBS_u , reflecting the uncertainty in distinguishing between different states given the existing sensors. For each sensor of the set OBS_c calculate the conditional entropy to see how much uncertainty remains if that sensor is added.

The entropy is calculated through

$$H(X) = - \sum_{x \in X} p_{OBS_c}(y) \log(p_{OBS_c}(y)) \quad (20)$$

with $x \in X$ describing the different possible states of the system's components and $p_{OBS_c}(y)$ describing the component's probability of being in that faulty state, under the condition that the component is not observable.

We can then calculate the information gain for each possible observable placement through the individual information gain

► **Definition 19 (Information Gain).** *The information gain by adding a sensor $obs_i \in OBS_c$ to the set OBS_u is defined over the given entropy H as follows:*

$$IG_H(obs_i) = H(X) - H(X|obs_i) \quad (21)$$

Note that $H(X|obs_i) \leq H(X)$ resulting in $IG_H(obs_i) \geq 0$ and that maximizing the information gain $\max_{s_i \in S_c} IG_H(s_i)$ can be directly reformulated as a minimization problem of the form $\min_{s_i \in S_c} H(X|s_i)$.

To illustrate, consider equipping the running example shown in Figure 3 with novel observables: First we have to determine all ways in which the system may fail. There are multiple cases that could lead to faulty outputs. Overall, faults within c_0 , c_1 , and c_2 might cause one or both outputs to be incorrect. These possible states are detailed in Table 1. The last column of this table lists the minimal hitting sets, indicating which components, if faulty, would result in the observed output.

Assuming a fictitious fault probability of five percent for all components, the entropy values and information gains are presented in Table 2. The first column provides the index, while the second column lists the possible diagnosis in terms of single faults. Subsequent columns show the entropy values for the current set of sensors and for scenarios with one additional observable. Table 3 presents the entropy value $H()$ for each index and the corresponding information gain $IG_H()$. The optimal observable to place is the one that minimizes entropy or maximizes information gain. In this case, the decision is not straightforward, as obs_0 and obs_1 both yield the same information gain.

7.1 Algorithm

Algorithm SYNTHESIZEOBS summarises the above definitions into pseudocode. The intuition behind the algorithm is to detect non-observable components using the calculation of critical pairs. Once the critical pairs within a system are known, we calculate optimal observables.

■ **Table 1** Different states of the running example and possible diagnosis.

Index	out_0	out_1	ω_{min}
0	0	0	-
1	1	0	$c_0 \vee c_1$
2	1	1	$(c_0 \vee c_1 \vee) \wedge c_2$
3	0	1	c_2

■ **Table 2** Entropy for all possible observations.

Index	$\omega_{min} \cap \mathcal{S}_c$	$H(x)$	$H(X c_0)$	$H(X c_1)$	$H(X c_2)$
1	$c_0 \vee c_1$	0.09860	0.0629	0.0629	0.09860
2	$(c_0 \vee c_1) \wedge c_2$	0.0108	0.006	0.006	0.09860
3	c_2	0.0629	0.0629	0.0629	0

■ **Table 3** Entropy and Information Gain for possible new observables.

	X	$X obs_0$	$X obs_1$	$X obs_2$
$H()$	0.1723	0.1318	0.1318	0.1972
$IG_H()$	0	0.0405	0.0405	-0.0249

If information about failure rates is available, we use it to calculate the entropy-maximal observables. If no further information is available, we return either the count-minimal, or the range-minimal novel observable, depending on the concrete implementation.

We will now explain the algorithm in detail. In Line 3 we use the SMT system description and the current observations to infer all intermediate values within the system using an appropriate solver for SMT- \mathcal{NRA} . This is basically how GDE [12] works on propositional logic. Line 4 calculates the critical pairs from the system description by generating system traces using the available observations. If the set of critical pairs is non-empty, Line 6 generates either a count-minimal, or range-minimal novel observable through properly designing function GENERATEOBSERVABLE. The function determines the ranges from the inferred values \mathcal{I} of the system description, and through the historical observations $HIST$. Line 7 determines whether failure probabilities are known for each component. If this is not the case the algorithm only returns the range-minimal or count-minimal novel observable. Contrarily, if the failure information is available, the algorithm computes the entropy of possible novel observables and their information gain (Line 9). All possible novel observables are added to set \mathcal{C} . Line 11 finally only returns the entropy-maximal novel observable.

8 Evaluation

To evaluate our approach, we have used the Berfipl benchmark with four tank systems developed by Ehrhardt et al. [21]. The benchmark describes an industrial bottling plant and was implemented as a simulation using OpenModelica 1.13. The four modules are a mixer, a distill, a filter, and a bottling module. The modules may be used stand-alone or may be connected. For the evaluation in this article, it was assumed that the modules are stand-alone and effects from one module do not propagate into another module. The mixer perturbs three substances and stores the mixed products in a tank. The mixer is connected to a distill, which splits substances with different boiling temperatures. The filter subsystem

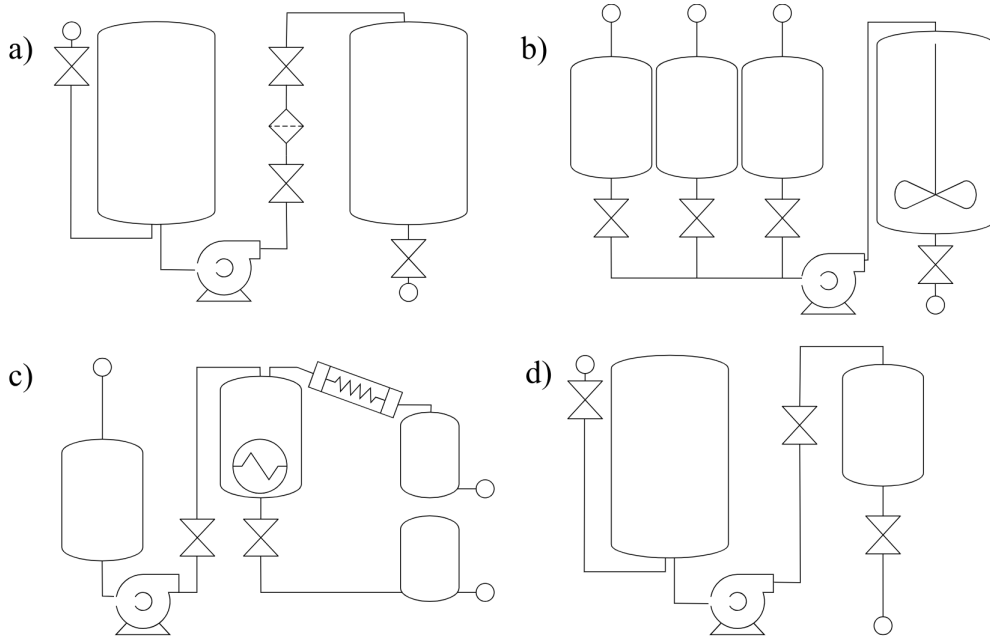
■ **Algorithm 1** *SynthesizeObs*: Calculate synthetic novel observables.

```

Data:  $SD, OBS, HIST$ 
Result: Entropy-maximal observable
1  $\mathcal{C} \leftarrow \emptyset$ ;
2  $\mathcal{I} \leftarrow inferValues(SD, OBS)$ ;
3 for  $c_i \in criticalPairs(SD, OBS)$  do
4    $obs_{new} \leftarrow generateObservable(c_i, \mathcal{I}, HIST)$ ;
5   if  $\forall c_i, p(c_i)$  is known then
6      $H(X) = -\sum_{x \in X} p(obs_i) \log(p(obs_i))$ ;
7      $IG_H(obs_i) \leftarrow H(X) - H(X|obs_i)$ ;
8      $\mathcal{C} \leftarrow \mathcal{C} \cup (obs_{new}, IG_H(obs_i))$ ;
9   else
10     $\mathcal{C} \leftarrow \mathcal{C} \cup obs_{new}$ ;
11  return( $max(\mathcal{C})$ );

```

removes residue from the fluid stream. As the last module, the bottling subsystem fills the produced substance into tiny glass bottles. Figure 5 shows the piping and instrumentation diagram of the four modules: a) filter, b) mixer, c) distill, and d) bottling.



■ **Figure 5** Four modules of the Berfpl benchmark presented by Ehrhardt et. al [21].

For the evaluation we have used an abstract model of each subsystem implemented in Python 3.11 and run on a 64-bit Windows 11 operating system with 32GB RAM, and Intel Core i7-12800H processor. For simplicity, and to keep the number of SMT expressions small we only calculate the flow through all components, while we did not model the intricacies of the special components filter and mixer. We modelled valves as a constant parameter that adjust the flow rate by multiplication

$$outflow = C \times a \times inflow \tag{22}$$

■ **Table 4** Experimental results for the Berfipl benchmark [21].

System	Critical Pair	Range	Correct
DS1	pump0, valve1	[0..9]	✓
	valve1, filter0	[0..9]	✓
	filter0, valve2	[0..9]	✓
DS2	valve0, pump0	[0..6.72]	✓
	valve1, pump0	[0..6.72]	✓
	valve3, pump0	[0..6.72]	✓
DS3	pump0, valve1	[0..9]	✓
DS4	pump0, valve1	[0..9]	✓

with parameter $C \in \mathbb{R}$, the size of the opening $a \in [0..1]$, and the inflow $inflow \in \mathcal{R}$. In the same way we have modelled pumps such that they increase the flow, using

$$outflow = P_{max} - Rp \times inflow \quad (23)$$

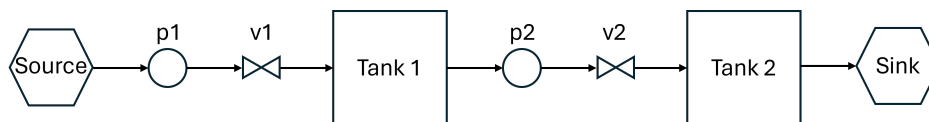
with $P_{max}, Rp \in \mathbb{R}$ modelling the pressure drop within the pump. Finally we have modelled tanks through

$$outflow = C \times a \times \sqrt{2g(level + inflow)} \quad (24)$$

with parameter $C \in \mathbb{R}$, size of the output orifice $a \in \mathbb{R}$, the gravitational constant g , and sum of the tank level and inflow. We then ran algorithm SYNTHESIZEOBS on each of the model, identified the critical pairs and calculated the range-maximal novel observations for each possible sensor location.

Table 4 presents the empirical results of algorithm SYNTHESIZEOBS. Since we have used the same parameters for the tanks, pipes, pumps etc. the different ranges are comparable. For each experiment, the algorithm was able to calculate the correct ranges for potential synthetic observations. Limitations are evident that by using SMT- \mathcal{NRA} we calculate the square root within the tank equations. Here we observed errors up to five percent.

We have also evaluated the generation of entropy-maximal novel observables. For this evaluation, we have used a slightly larger system than the running example. Consider the system in figure 6 that contains a source, two tanks and a sink. The pipes between source and tank1 as well as between the two tank are equipped with a pump and a valve each. Consider the system to be equipped with one observable per tank that measures the fill level and provides two observations: high-level and low-level. The possible states of the observables can be seen in Table 1. The pumps can have two fault modes: higher p_xh or lower p_xl performance than expected. The valve may fail through clogging v_x . The possible faults are summarised in the last column as single faults. Possible combinations are not shown.



■ **Figure 6** Example tank system.

Some faults have just one single element as a possible diagnosis. These failure states can be directly diagnosed. Others show multiple components that could cause this effect. However, given the observables, in many cases the diagnosis can contain multiple components.

The problem of critical pairs arise if two components cannot be distinguished. This is the case for the combination of pump and valve in each pipe.

If we let the probability of a fault in the pumps and valves be five percent this leads to the entropy values and information gain stated in Table 5. The first column denotes the index. The second column shows the possible diagnosis. The following columns show the entropy values for the current sensors and one additional sensor. The lower table shows the entropy value $H()$ for each index and the information gain $IG_H()$. The best sensor to place is the one that results in the minimum entropy or gives the maximum information gain. In this case this is sensor p_2 .

■ **Table 5** Possible states of the system.

State ID x	$\omega_{min} \cap \mathcal{S}_c$	$H(x)$	$H(x p_1)$	$H(X v_1)$	$H(X p_2)$	$H(X v_2)$
1	$p_1h \vee p_2l \vee v_2$	0.195	0.13	0.195	0.13	0.13
2	p_1h	0	0	0	0	0
3	$p_2l \vee v_2$	0.13	0.13	0.13	0	0
4	$p_1l \vee v_1 \vee p_2h$	0.195	0.13	0.13	0.13	0.195
5	p_2h	0	0	0	0	0
6	$p_1l \vee v_1$	0.13	0	0	0.13	0.13
7	p_2h	0	0	0	0	0
8	$p_2l \vee v_2$	0.13	0.13	0.13	0	0
		X	$X p_1$	$X v_1$	$X p_2$	$X v_2$
	$H()$	0.78	0.52	0.585	0.39	0.455
	$IG_H()$	0	0.26	0.195	0.39	0.325

9 Discussion and Conclusion

While the evaluation has shown that we can successfully infer range- and count-minimal novel observations using SMT- \mathcal{NRA} system descriptions and the computation of critical pairs, our method has some practical drawbacks:

1. Using SMT- \mathcal{NRA} for process industrial systems requires the computation of square roots. But computing the square roots introduces significant errors through chaining effects.
2. The SMT- \mathcal{NRA} system descriptions we use are quite simple. Computing critical pairs on those system descriptions does not distinguish between the type of component. In practice, some components, such as a filter will lead to different downstream effects than, for example, a tank. Therefore, some components and their effects may be diagnosable, even though there is no sensor directly attached to them.
3. We performed the entropy calculations outside of the SMT- \mathcal{NRA} system description. In future work this should be integrated to obtain a unified formulation for calculating novel observables.
4. Our approach requires significantly more expert knowledge than is available for many industrial processes. While the additional historical data that we require can often be obtained, the fault probabilities for individual components are often not widely available.
5. We do not take different operating modes into account and thus need to assume there is only a normal working mode.

Overall we have presented a new method to compute novel observables using algorithm SYNTHESIZEOBS. For this we use the well-known tuple (SD, COMPS, OBS) along with historical data in the form of traces for the computation of critical pairs. The critical pairs

provide us with the information which components within a system are non-diagnosable. We then use algorithm SYNTHESIZEOBS to compute optimal novel observables that include synthesized normal operating ranges which can thus be used for anomaly detection and fault diagnosis. If information about individual fault probability is given, then we also compute the entropy-maximal novel observable, i.e. that observable whose addition in the system would gather the most information.

We hope that our method aids practitioners in improving the observability particularly of learned system descriptions in industrial use-cases.

References

- 1 Mehdi Bayouhd and Louise Travé-Massuyès. Diagnosability analysis of hybrid systems cast in a discrete-event framework. *Discrete Event Dynamic Systems*, 24(3):309–338, 2014. doi:10.1007/S10626-012-0153-Z.
- 2 Benjamin Bittner, Marco Bozzano, Alessandro Cimatti, Marco Gario, Stefano Tonetta, and Viktoria Vozarova. Diagnosability of fair transition systems. *Artificial Intelligence*, 309:103725, 2022. doi:10.1016/J.ARTINT.2022.103725.
- 3 Benjamin Bittner, Marco Bozzano, Alessandro Cimatti, and Xavier Olive. Symbolic synthesis of observability requirements for diagnosability. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 26, pages 712–718, 2012. doi:10.1609/AAAI.V26I1.8225.
- 4 Alexander Bochman. *A logical theory of causality*. MIT Press, 2021.
- 5 Abderraouf Bousif and Mohamed Ghazel. Diagnosability analysis of intermittent faults in discrete event systems using a twin-plant structure. *International Journal of Control, Automation and Systems*, 18(3):682–695, 2020.
- 6 Marco Bozzano, Alessandro Cimatti, and Stefano Tonetta. Testing diagnosability of fair discrete-event systems. In *Proc. International Workshop on Principles of Diagnosis (DX-19)*, 2019.
- 7 Maria Paola Cabasino, Alessandro Giua, Stéphane Lafortune, and Carla Seatzu. A new approach for diagnosability analysis of petri nets using verifier nets. *IEEE Transactions on Automatic Control*, 57(12):3104–3117, 2012. doi:10.1109/TAC.2012.2200372.
- 8 Matteo Camilli, Raffaella Mirandola, and Patrizia Scandurra. Enforcing resilience in cyber-physical systems via equilibrium verification at runtime. *ACM Transactions on Autonomous and Adaptive Systems*, 2023.
- 9 Cody James Christopher and Alban Grastien. Critical observations in model-based diagnosis. *Artificial Intelligence*, page 104116, 2024. doi:10.1016/J.ARTINT.2024.104116.
- 10 Edmund M Clarke, Thomas A Henzinger, Helmut Veith, Roderick Bloem, et al. *Handbook of model checking*, volume 10. Springer, 2018.
- 11 Marie-Odile Cordier, Louise Travé-Massuyès, Xavier Pucel, et al. Comparing diagnosability in continuous and discrete-event systems. In *Proceedings of the 17th international workshop on principles of diagnosis (DX-06)*, pages 55–60, 2006.
- 12 Johan De Kleer and Brian C Williams. Diagnosing multiple faults. *Artificial intelligence*, 32(1):97–130, 1987. doi:10.1016/0004-3702(87)90063-4.
- 13 Maria D Di Benedetto, Stefano Di Gennaro, and Alessandro D’Innocenzo. Verification of hybrid automata diagnosability by abstraction. *IEEE transactions on automatic control*, 56(9):2050–2061, 2011. doi:10.1109/TAC.2011.2105738.
- 14 Alexander Diedrich, Patrick Deutschmann, and Caroline Junker. Servicenavigator - a bayesian assistance system for diagnosing industrial production systems. In *2022 5th IEEE International Conference on Industrial Cyber-Physical Systems (ICPS)*. IEEE, 2022.
- 15 Alexander Diedrich, Alexander Maier, and Oliver Niggemann. Model-based diagnosis of hybrid systems using satisfiability modulo theory. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19)*. Thirty-Third AAAI Conference on Artificial Intelligence (AAAI-19), 2019.

- 16 Alexander Diedrich, Lukas Moddemann, and Oliver Niggemann. Learning system descriptions for cyber-physical systems. In *Proceedings of 12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, 2024.
- 17 Alexander Diedrich and Oliver Niggemann. Diagnosing systems through approximated information. *13th Annual Conference of the Prognostics and Health Management Society*, 2021.
- 18 Alexander Diedrich and Oliver Niggemann. On residual-based diagnosis of physical systems. *Engineering Applications of Artificial Intelligence*, 109:104636, 2022. doi:10.1016/j.engappai.2021.104636.
- 19 Oumar Diene, Marcos V Moreira, Eduardo A Silva, Victor R Alvarez, and Claudionor F Nascimento. Diagnosability of hybrid systems. *IEEE Transactions on Control Systems Technology*, 27(1):386–393, 2017.
- 20 Barry Dowdeswell, Roopak Sinha, and Stephen G MacDonell. Diagnosable-by-design model-driven development for iec 61499 industrial cyber-physical systems. In *IECON 2020 the 46th annual conference of the IEEE industrial electronics society*, pages 2183–2188. IEEE, 2020. doi:10.1109/IECON43393.2020.9254620.
- 21 Jonas Ehrhardt, Malte Ramonat, Rene Heesch, Kaja Balzereit, Alexander Diedrich, and Oliver Niggemann. An ai benchmark for diagnosis, reconfiguration & planning. In *2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA)*. IEEE, 2012.
- 22 Teresa Escobet, Anibal Bregon, Belarmino Pulido, and Vicenç Puig. *Fault Diagnosis of Dynamic Systems*. Springer, 2019.
- 23 Alexander Feldman, Gregory Provan, and Arjan Van Gemund. Approximate model-based diagnosis using greedy stochastic search. *Journal of Artificial Intelligence Research*, 38:371–413, 2010. doi:10.1613/JAIR.3025.
- 24 GK Fourlas, Kostas J Kyriakopoulos, and NJ Krikelis. Diagnosability of hybrid systems. In *Proceedings of the 10th IEEE mediterranean conference on control and automation*, 2002.
- 25 Zhiwei Gao, Carlo Cecati, and Steven X Ding. A survey of fault diagnosis and fault-tolerant techniques—part i: Fault diagnosis with model-based and signal-based approaches. *IEEE Transactions on Industrial Electronics*, 62(6):3757–3767, 2015. doi:10.1109/TIE.2015.2417501.
- 26 Alban Grastien and Marina Zanella. Diagnosability of discrete faults with uncertain observations. *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*, pages 253–278, 2018.
- 27 Daniel Jung. Structural methods for distributed fault diagnosis of large-scale systems. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 2690–2695. IEEE, 2020. doi:10.1109/CDC42340.2020.9303744.
- 28 Samuel Kolb, Stefano Teso, Andrea Passerini, and Luc De Raedt. Learning smt (lra) constraints using smt solvers. In *IJCAI International Joint Conference on Artificial Intelligence*, volume 2018, pages 2333–2340. ijcai.org, 2018. doi:10.24963/IJCAI.2018/323.
- 29 Mattias Krysander and Mattias Nyberg. Fault diagnosis utilizing structural analysis. *CCSSE, Norrköping, Sweden*, 2002.
- 30 Stephane Lafortune and Feng Lin. From diagnosability to opacity: A brief history of diagnosability or lack thereof. *IFAC-PapersOnLine*, 50(1):3022–3027, 2017.
- 31 Gianfranco Lamperti, Stefano Trerotola, Marina Zanella, and Xiangfu Zhao. Sequence-oriented diagnosis of discrete-event systems. *Journal of Artificial Intelligence Research*, 78:69–141, 2023. doi:10.1613/JAIR.1.14630.
- 32 Ruben Leal, Jose Aguilar, Louise Travé-Massuyès, Edgar Camargo, and Addison Ríos. A genetic algorithm approach for diagnosability analysis. *International Journal of Engineering Development and Research*, 2(4):3786–3799, 2014.
- 33 Alexander Maier. Online passive learning of timed automata for cyber-physical production systems. In *Industrial Informatics (INDIN), 2014 12th IEEE International Conference on*, pages 60–66. IEEE, 2014. doi:10.1109/INDIN.2014.6945484.

- 34 Mohammadreza Mehrabian, Mohammad Khayatian, Aviral Shrivastava, Patricia Derler, and Hugo Andrade. A run-time verification method with consideration of uncertainties for cyber-physical systems. *Microprocessors and Microsystems*, page 104890, 2023. doi: 10.1016/J.MICPRO.2023.104890.
- 35 Amit Metodi, Roni Stern, Meir Kalech, and Michael Codish. A novel sat-based approach to model based diagnosis. *Journal of Artificial Intelligence Research*, 51:377–411, 2014. doi:10.1613/JAIR.4503.
- 36 Lukas Moddemann, Henrik Sebastian Steude, Alexander Diedrich, and Oliver Niggemann. Discret2di - deep learning based discretization for model-based diagnosis. In *Proceedings of 12th IFAC Symposium on Fault Detection, Supervision and Safety for Technical Processes*, 2024.
- 37 Fatiha Nejari, Ramon Pérez, Teresa Escobet, and Louise Travé-Massuyès. Fault diagnosability utilizing quasi-static and structural modelling. *Mathematical and computer modelling*, 45(5-6):606–616, 2007. doi:10.1016/J.MCM.2006.06.008.
- 38 Ingo Pill and Franz Wotawa. Fault detection and localization using modelica and abductive reasoning. *Diagnosability, Security and Safety of Hybrid Dynamic and Cyber-Physical Systems*, pages 45–72, 2018.
- 39 Raymond Reiter. A theory of diagnosis from first principles. *Artificial intelligence*, 32(1):57–95, 1987. doi:10.1016/0004-3702(87)90062-2.
- 40 Meera Sampath, Stéphane Lafortune, and Demosthenis Teneketzis. Active diagnosis of discrete-event systems. *IEEE transactions on automatic control*, 43(7):908–929, 1998. doi: 10.1109/9.701089.
- 41 Meera Sampath, Raja Sengupta, Stéphane Lafortune, Kasim Sinnamohideen, and Demosthenis Teneketzis. Diagnosability of discrete-event systems. *IEEE Transactions on automatic control*, 40(9):1555–1575, 1995. doi:10.1109/9.412626.
- 42 Claude Elwood Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. doi:10.1002/J.1538-7305.1948.TB01338.X.
- 43 Roni Stern, Meir Kalech, and Orel Elimelech. Hierarchical diagnosis in strong fault models. In *Twenty Fifth International Workshop on Principles of Diagnosis*, 2014.
- 44 Jinya Su and Wen-Hua Chen. Model-based fault diagnosis system verification using reachability analysis. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 49(4):742–751, 2017.
- 45 L Travé-Massuyès, T Escobet, and R Milne. Model-based diagnosability and sensor placement. *12th Intl. Work. on Principles of Diagnosis*, 2001.
- 46 Louise Travé-Massuyès and Teresa Escobet. Bridge: Matching model-based diagnosis from fdi and dx perspectives. In *Fault Diagnosis of Dynamic Systems*, pages 153–175. Springer, 2019.
- 47 Louise Travé-Massuyès, Teresa Escobet, and Xavier Olive. Diagnosability analysis based on component-supported analytical redundancy relations. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 36(6):1146–1160, 2006. doi:10.1109/TSMCA.2006.878984.
- 48 Ángel Jesús Varela-Vaca, David G Rosado, Luis E Sánchez, María Teresa Gómez-López, Rafael M Gasca, and Eduardo Fernández-Medina. Carmen: A framework for the verification and diagnosis of the specification of security requirements in cyber-physical systems. *Computers in Industry*, 132:103524, 2021. doi:10.1016/J.COMPIND.2021.103524.
- 49 Nathalie Verdière, Carine Jaubertie, and Louise Travé-Massuyès. Functional diagnosability and detectability of nonlinear models based on analytical redundancy relations. *Journal of Process Control*, 35:1–10, 2015.
- 50 Amaury Vignolles, Elodie Chanthery, and Pauline Ribot. An overview on diagnosability and prognosability for system monitoring. In *European conference of the Prognostics and Health Management Society (PHM Europe)*, 2020.
- 51 Su Xingyu, Marina Zanella, Alban Grastien, et al. Diagnosability of discrete-event systems with uncertain observations. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016*, pages 1265–1271. AAAI Press/International Joint Conferences on Artificial Intelligence, 2016.