Model-Based Diagnosis of Hybrid Systems Using Satisfiability Modulo Theory

Alexander Diedrich, Alexander Maier
Fraunhofer IOSB-INA
Fraunhofer Center for Machine Learning
Lemgo, Germany
alexander.diedrich@iosb-ina.fraunhofer.de
alexander.maier@iosb-ina.fraunhofer.de

Oliver Niggemann
Institute Industrial IT
Lemgo, Germany
oliver.niggemann@hs-owl.de

Abstract

Currently, detecting and isolating faults in hybrid systems is often done manually with the help of human operators. In this paper we present a novel model-based diagnosis approach for automatically diagnosing hybrid systems. The approach has two parts: First, modelling dynamic system behaviour is done through well-known state space models using differential equations. Second, from the state space models we calculate Boolean residuals through an observer-pattern. The novelty lies in implementing the observer pattern through the use of a symbolic system description specified in satisfiability theory modulo linear arithmetic. With this, we create a static situation for the diagnosis algorithm and decouple modelling and diagnosis. Evaluating the system description generates one Boolean residual for each component. These residuals constitute the fault symptoms. To find the minimum cardinality diagnosis from these symptoms we employ Reiter's diagnosis lattice.

For the experimental evaluation we use a simulation of the Tennessee Eastman process and a simulation of a four-tank model. We show that the presented approach is able to identify all injected faults.

Introduction

Diagnosing modern production systems is a key element of research agendas such as Cyber-Physical Production Systems (CPPSs) (Lee 2008; Rajkumar et al. 2010) or its German pendant Industry 4.0. In these agendas, a major focus is on the self-diagnosis capabilities for complex and distributed CPPSs. Typical goals of such self-diagnosis approaches are the detection of anomalies, suboptimal energy consumptions, error causes in large plants, or wear (Isermann 2004; Niggemann and Lohweg 2015).

Very roughly, we can differentiate between two types of diagnosis approaches—which come in several flavors:

Heuristic or Phenomenological Approach: Here, the system observations are directly classified as correct or anomalous (Ferracuti et al. 2011). I.e. the diagnosis software uses models which deduce from anomalies, here called symptoms, to faults, here called root causes. Traditionally, the classification know-how is often modeled manually, e.g. in form of rules (expert systems). For fast-changing CPPSs, the classifier is trained using supervised machine learning algorithms (Matias et al. 2013).

Model-based Approach: Model-based diagnosis (MBD) approaches (Struss and Ertl 2009; Niggemann et al. 2012) are better suited to identify root causes in large distributed systems. With MBD, a model is used to simulate the normal behavior of a plant or normal product features, i.e. unlike heuristic diagnosis models, the model infers from causes to symptoms. Such models come in different flavors: Statistical (Ferracuti et al. 2011) and state-based models (Windmann et al. 2013) or physical first principle models (de Kleer et al. 2013). So model-based approaches capture the normal situation while phenomenological approaches capture the differences between normal and anomalous situations. The main challenge for MBD is high engineering efforts for the creation of such models.

While heuristic approaches are often more straightforward and do not require a system model, they have one major inherent drawback: They must deduce against the direction of causality since they deduce from observations to anomalies. For complex distributed systems with their high number of interdependencies between components and their complex causalities, this is a hard task because a high number of classification rules is needed to discriminate between all possible combinations of symptoms. Model-based approaches do not have this problem since system models take all inputs and compute the outputs, i.e. they work in the direction of the physical causality. So in general, phenomenological approaches are chosen for local compact devices while model-based approaches are chosen for complex, distributed plants.

While MBD has proven to be well suited for complex systems, it has for several reasons proven difficult to apply it to CPPS:

Timed and State-Based Behavior: CPPSs are distributed physical systems with complex timing behavior. So any diagnosis must take dynamic behavior characteristics into consideration. Furthermore, often the behavior depends on state variables, i.e. the system has memories influencing the behavior. But MBD so-far mainly deals with static systems.

Hybrid Behavior: CPPSs are hybrid systems which comprise discrete signals, time- and value-continuous signals, and structured data. Often, discrete signals such as opening a valve or turning off a robot trigger mode changes (Buede

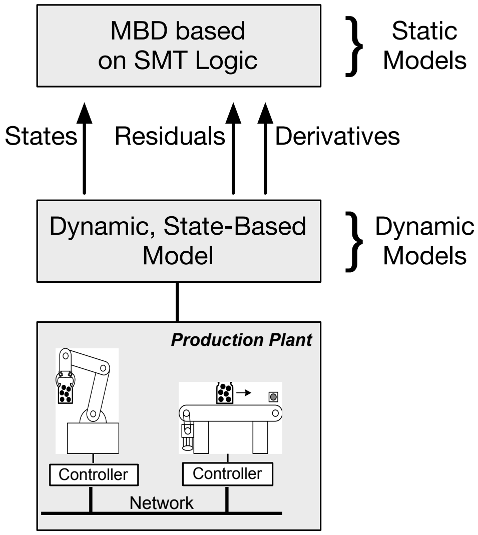

Figure 1: The general solution idea. Process data from the production plant is abstracted through a dynamic, state-based model. Observers attached to the model calculate numerical abstractions such as residuals. SMT translates these abstractions into symbols to perform MBD.

2009), i.e. they abruptly change the system behavior. MBD so-far mainly deals with binary values.

Of course, several authors have developed solutions to these two challenges (Struss 1997; Grastien 2013; Daigle et al. 2010; Fränzle, Hermanns, and Teige 2008; Khorasgani and Biswas 2017), but they in general replace the dynamic and hybrid model by a static and discrete model—often by means of value- and time discretization. This leads to problems concerning the synchronization between system and model, often require a full observability of the system and lead to rather complex model formalisms.

Here, a different approach is presented: As can be seen in Figure 1, we first use arbitrary dynamic behavior models to predict all significant system behaviors, e.g. using differential equations or state machines.

The behavior models especially compute, via standardized interfaces, variables such as residuals, system states, and derivatives. These variables form the basis for a generic extension of MBD by means of Satisfiability Modulo Theories (SMT). With SMT, the semantics of functions and predicates are determined by an underlying theory—here this theory is fed by the interfaces of the behavior model.

So our contribution in this paper is threefold:
(i) A generic extension of MBD is developed which allows for the integration of arbitrary dynamic behavior models. For each diagnosis run, all relevant information is communicated via pre-defined variables such as states or derivatives, i.e. from the point-of-view of the MBD algorithm, a static situation exists and no discretization of the timing is necessary. I.e. we leverage on significant experience concerning of MBD for static system but integrate dynamic aspects via the underlying theory.
(ii) By integrating only residuals into the underlying theory, no discretization of signal values is necessary. Real valued residuals are easily discretized.
(iii) We show that the necessary system knowledge for the MBD model can be easily derived from engineering artifacts, e.g. modeled in AutomationML.

State of the Art

Struss (1997) published a paper on the fundamentals of MBD of dynamic systems. In this he described how hybrid systems can be modelled without resorting to a complete simulation of the system under investigation. He proposed to capture the temporal and dynamic behaviour of a hybrid system in a set of modes which model the system. Each mode has distinct state and temporal constraints in addition to so called Continuity, Integration, and Derivatives (CID) constraints that affect all modes.

Daigle et al. (2010) have adapted a discrete event approach to diagnose continuous systems. They state that each fault that occurs in a continuous system has a unique fault signature. A fault signature denotes a qualitative effect that a fault occurs in an observation. Under the assumption that all fault signatures and measurement orderings are known, they employ a diagnoser that traces the states through a temporal causal graph based on measurements.

Roychoudhury et al. (2011) have shown how to use hybrid bond graphs (HBG) to diagnose hybrid systems. HBGs abstractly model the system by describing causal, continuous relationships between components. In Daigle et al. (2010) have employed the developed HBGs to diagnose a space-craft power distribution system. Prakash et al. (2017) have used an extended framework with HBGs to make improvements in diagnosing two-tank systems.

Grastien (2013) used SMT for the diagnosis of hybrid systems. He discretizes values in a hybrid system into a set of distinct states. Each observation $< \tau, A >$ is understood as a behaviour A at time τ, where A is a partial assignment of the variables in a state. Each variable is augmented with an indicator stating at which time-step the variable expression is valid.

Fränzle et al. (2008) have augmented SMT with probabilistic approaches in order to analyse stochastic hybrid systems. By using bounded-model checking together with probabilistic hybrid automata, piecewise deterministic Markov processes, and stochastic differential equations they are able to create a fault analysis system without the need to formulate intermediate finite-state abstractions as the methods mentioned above do.

In another work, Khorasgani (2017) describe a hybrid system model through hybrid minimal structurally overdetermined sets (HMSOs). These are sets of differential equations and (in-) equations which model the behaviour of a hybrid system.

In contrast to Struss (1997) and Provan (2009) we do not use automatons and mode estimation to partition the system into different states. Instead, we only sample the system at some suitable interval and use the obtained information directly to model the states in the state-space representation. Unlike in space-craft, which where analyzed by

Daigle (2007), fault signatures and measurement orderings are unknown in industrial systems. This requires us to pursue a more uninformed approach. Our approach is an alternative to hybrid bond graphs used by Roychoudhury (2011), while they are at the same time an extension to the work of Grastien (2013) and Khorasgani (2017). In comparison to Grastien we do not singly use satisfiability modulo theory, but instead capture system behaviour in a state-space representation. We expect this to reduce the required computational effort. We also make use of (in-) equations and differential equations as were used by Khorasgani and Biswas, but augment these with the diagnostic reasoning of traditional model-based diagnosis. Compared to Fränzle, we do not make use of stochastic SMT at this point to keep the system more explainable for users.

Demonstration Use Case

For this work we will use the four tank system depicted in Figure 2 as a running example. The system consists of four water tanks t, seven electric valves p with integrated flow sensors, an unlimited water source and an unlimited water sink (not shown). Valve p_0 controls water from the unlimited water source, for example the public water mains, into tank t_0. From there, three pipes with an equal diameter divide the water flow. Finally, valve p_6 drains tank t_3 into the unlimited water sink, for example a river or a processing facility.

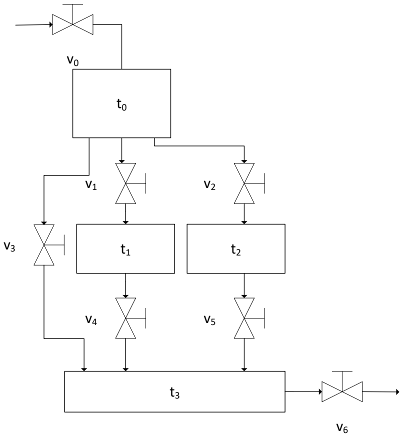

Figure 2: The demonstration use case showing a four-tanks model

Each tank has two binary sensors which indicate overflow and underflow, respectively. There are no provisions to directly measure the water level. Each valve has a switch which indicates whether or not the valve is open. In addition, each valve has an associated flow sensor.

Solution Approach

Integration of Residuals

For this work we use a state-space representation to model the dynamic behaviour of the hybrid system over time. This section shows how the state-space representation is realised and how we can calculate Boolean residuals. We assume that the data fed into the state-space equations is sampled in a suitable interval.

The state is propagated through the explicit Euler method

$$\begin{aligned}\mathbf{x}(t+1) &= f(\mathbf{x}(t), \mathbf{u}(t)) \\ \mathbf{y}(t) &= g(\mathbf{x}(t), \mathbf{u}(t), \tau)\end{aligned} \tag{1}$$

where $\mathbf{x}(t+1)$ is a vector of the state in the next time step, $\mathbf{x}(t)$ is the current state vector, $\mathbf{u}(t)$ is the observable input vector, $\mathbf{y}(t)$ is the observable output vector, and τ is a vector of expert-defined threshold values.

Usually, the state of a system cannot be measured directly as in the case of biological reactors, for example. In these reactors only a subset of all possible inputs and states can be measured. Industrial processes in general are instrumented only to the extend that is necessary to safely control them (Lee and Weekman Jr 1976). This often makes diagnosis information unavailable. The unavailable information must be calculated through the state-space model.

Therefore, we assume in the four-tank model that each tank's water level needs to be calculated through its inflow and outflow and cannot be observed directly. The inflow and outflow can be measured at the associated valves in each inflow and outflow pipe. Assuming that each tank has some sensors to indicate under- and overflow, these are used for the target (output). As an example, for the state, input, and output vectors we thus have

$$\mathbf{x} = \begin{bmatrix} h_0 \\ h_1 \\ h_2 \\ h_3 \end{bmatrix} \mathbf{u} = \begin{bmatrix} \texttt{flow}_0 \\ \texttt{flow}_1 \\ \vdots \\ \texttt{flow}_6 \end{bmatrix} \mathbf{y} = \begin{bmatrix} \texttt{overflow}_0 \\ \vdots \\ \texttt{overflow}_3 \\ \texttt{underflow}_0 \\ \vdots \\ \texttt{underflow}_3 \end{bmatrix}.$$

The function $f(\mathbf{x}, \mathbf{u})$ models the current state and its current input and from this computes the next state. Therefore, we can write

$$f(\mathbf{x}(t), \mathbf{u}(t)) = A\boldsymbol{\Delta}(\mathbf{x}, \mathbf{u}, t) + B\mathbf{u}(t) \tag{2}$$

with A being a matrix and

$$B = \begin{bmatrix} 1 & -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & -1 \end{bmatrix}$$

being the incidence matrix of connected components, and $\boldsymbol{\Delta}$ being a vector of functions. $B_{ij} = 1$ denotes an input, $B_{ij} = -1$ an output, and $B_{ij} = 0$ indicates no connection.

In the real world it is infeasible to observe all values for each component. Instead, unobserved values need to be inferred from the rest of the system. For example, the behaviour of a passive component such as a filter, which does not have any associated sensors, needs to be inferred or estimated.

In this approach we create for each time-step t a procedure which can infer values for all components that are not observable. For the four-tank model we inferred the values by calculating them through difference equations in SMT logic. Difference equation are a common tool to propagate values in the process industry and are supported in SMT. The unobservable values are denoted as $\tilde{\mathbf{u}}(t)$. Thus, the input vector $\hat{\mathbf{u}}(t)$ of observable and non-observable values is constructed $\hat{\mathbf{u}}(t) = [\tilde{\mathbf{u}}(t) \quad \mathbf{u}(t)]^T$. In the following we will use $\hat{\mathbf{u}}(t)$ for all occurences of $\mathbf{u}(t)$.

Matrix A states which current state influences the next state of the same component. Matrix B shows the connections between the system's components, which in this case are the pipes between the tanks.

To model the water level in each tank it is possible to use difference equations

$$x(t+1) = \frac{1}{D}(Q_i(t) - C_d a\sqrt{2gx(t)}) \tag{3}$$

with the tank diameter D, the discharge coefficient C_d, the size of the orifice a, and the gravitational constant g. $Q_i(t)$ is the sum of all inputs to the current tank $Q_i = \sum_{q \subset \hat{u}} q$. A vector describing the current output of a state-based component (such as a tank) $\boldsymbol{\Delta}(\mathbf{x}, \mathbf{u}, t)$ can be created, given its new state.

$$\boldsymbol{\Delta}(\mathbf{x}, \mathbf{u}, t) = \begin{bmatrix} \hat{u_i}(t+1) = \frac{1}{A_i}(C_{d,i}a_i\sqrt{2gx_i(t+1)}) \\ \hat{u_j}(t+1) = \frac{1}{A_j}(C_{d,j}a_j\sqrt{2gx_j(t+1)}) \\ \vdots \end{bmatrix}.$$

With this model it is possible to propagate the state of the system as it evolves through time by calculating each component's output value. However, given this information, a control system cannot yet determine the full behaviour of the system. For this, the output vector $\mathbf{y}(t) = g(\mathbf{x}(t), \mathbf{u}(t), \tau)$ needs to be calculated by

$$g(\mathbf{x}(t), \hat{\mathbf{u}}(t), \tau) = \mathcal{C}\begin{bmatrix} r(x_0, \tau_0) \\ \vdots \\ r(x_n, \tau_n) \end{bmatrix}. \tag{4}$$

The vector τ are threshold parameters based on meta-information or expert knowledge. The thresholds determine when a given parameter exceeds the range of normal behaviour. $r(\cdot)$ are partial functions

$$r(x_i, \tau_i) = \begin{cases} 0 & \text{if } x_i \leq \tau_i, \\ 1 & \text{else} \end{cases} \tag{5}$$

that relate the thresholds to the current state x. The matrix $\mathcal{C}$ maps the results of the functions $o(h, \tau)$ and $l(h, \tau)$ into the output vector y. Applying this to the four-tank model results in

$$g(\mathbf{x}(t), \hat{\mathbf{u}}(t), \tau) = \mathcal{C}\begin{bmatrix} o(h_0, \tau_0^o) \\ \vdots \\ o(h_3, \tau_3^o) \\ l(h_0, \tau_0^l) \\ \vdots \\ l(h_3, \tau_3^l) \end{bmatrix} \tag{6}$$

For ease of notation we use τ_i^o to denote the threshold for the upper limit of tank i and τ_i^l to denote the lower limit of tank i. The function $o(h, \tau)$ indicates when the water level within the tank has approached the upper limit. This is calculated by

$$o(h_i, \tau_i^o) = \begin{cases} 0 & \text{if } h_i \leq \tau_i^o, \\ 1 & \text{else} \end{cases}. \tag{7}$$

Likewise, the lower limit of the water level can be calculated with function $l(h_i, \tau_i^l)$.

In the following, we will create an observer pattern which uses SMT expressions of these residuals to construct hypotheses for fault diagnosis.

Extension of MBD using SMT Logic

To diagnose faults within the described state-space system it is necessary to obtain health information about single components. For this we can use the output of functions $r(\cdot)$. We will first introduce classical MBD and then show how we extend the theory with expressions from SMT to diagnose faults.

Classical consistency-based MBD uses observations (OBS), a system description (SD), and a component description (COMPS) for describing a system, so that

$$\mathrm{SD} \cup \mathrm{COMPS} \cup \mathrm{OBS} \vdash \top \tag{8}$$

is satisfiable.

Traditional MBD was executed on Boolean expressions, for example, in case of the electronic repairman (Brown 1974). Here we need a more general approach that covers real valued functions. We treat Boolean functions only as a special case. The theoretical foundations are mostly based on work by Reiter (Reiter 1987) and de Kleer (De Kleer and Williams 1987), or more recently, Feldman (Feldman, Provan, and Van Gemund 2010).

Definition 1 (Basis) *Basis $\mathcal{B}$ is a set of single-output functions $\{B_1, B_2, \ldots, B_n\}$.*

Usually, a basis in MBD is the set of Boolean circuits within the system. For a standard full-adder, for example, $\mathcal{B}$ contains two AND-gates, two XOR-gates, and one OR-gate. For hybrid systems we permit each function to have real valued input and output parameters $B_i : \mathcal{R} \rightarrow \mathcal{R}$.

Definition 2 (System) *Given a basis $\mathcal{B}$, a hybrid system $\mathrm{M}(\mathcal{B}) = \langle V \cup \{I^\star, O^\star\}, E\rangle$ is a directed graph in which each edge $e \in E$ is a variable, each node $v \in V$ is a function drawn from $\mathcal{B}$, $I^\star$ is a primary input source, and $O^\star$ is a primary output sink.*

$\mathrm{M}(\mathcal{B})$ represents a connectivity graph between all components. For each component an approximation function is defined to approximate its behaviour.

Definition 3 (Fault-Augmented Model) *Given $\mathcal{B}$, a system $\mathrm{M}(\mathcal{B})$ and a second basis $\mathcal{B}^\star$, a fault-augmented model $\mathrm{SD}(\mathcal{B}, \mathcal{B}^\star)$ is defined as the ordered triple $\langle \mathrm{COMPS}, V, E, F\rangle$ where $\mathrm{COMPS} = \{f_1, f_2, \ldots, f_n\}$ with $f_i \in \{\bot, \top\}$, $n = |V|$, and F is a mapping $F : \mathcal{B} \rightarrow \mathcal{B}^\star$.*

When a system is modeled through $\mathrm{M}(\mathcal{B})$ only the correct behaviour of the model is specified. In this case, there are no provisions to inject a fault into the system. It is necessary to create a model which supports the injection of faults into the system. For Definition 3 the model $\mathrm{M}(\mathcal{B})$ is the correctly functioning system, and $\mathcal{B}^\star$ is a basis with variables that can induce components to fail.

Based on these traditional MBD definitions, we augment these with satisfiability theory modulo linear arithmetic (SMT-$\mathcal{LRA}$). SMT-$\mathcal{LRA}$ generalises to the satisfiability (SAT) problem. Given a complex term in predicate logic, satisfiability theory (SAT) seeks an assignment of all free variables of this term to achieve satisfiability. For example, provided a formula $\phi(x_0, \ldots, x_n)$, an interpretation I, a variable assignment α is computed so that $[[\phi(x_0, \ldots, x_n)]]_{I,\alpha} = \top$.

Definition 4 (Satisfiability Modulo Linear Arithmetic) *The SMT syntax consists of the symbols* COMPS*, a relational signature Σ^R, and a functional signature Σ^F with $\sigma : \mathcal{R} \rightarrow \{\bot, \top\}$ and $\sigma \in \Sigma^F$.*

Contrary to predicate logic in SMT-$\mathcal{LRA}$ each term ϕ can be an (in-)equation or some other algebraic formula.

We use weak fault models (WFMs) to model the normal behaviour of the system. The health of each component $H_{c,i}$ is implied by the conjunction

$$\sigma_i : \bigwedge_{|\hat{\mathbf{u}}(t)|} (\hat{u}_i \leq \tau_i) \rightarrow H_{c,i} \tag{9}$$

over all (in-)equations which are relevant for component $c_i \in$ COMPS. For unobservable components missing values $\tilde{\mathbf{u}}(t)$ are calculated through the SMT expression

$$\delta_i \colon d(u_i) = \tilde{u_i} \tag{10}$$

where $d(\cdot)$ is some suitable approximation function. For the linear case $d(\cdot)$ can be computed within the SMT framework. For non-linearities $d(\cdot)$ is calculated using an external toolbox and only the result is inserted as a predicate into an SMT expression.

Formulating WFMs with SMT-$\mathcal{LRA}$ for the valves in the four tank model results in

$$\sigma_{V,i} : (flow_i^l \leq flow_i) \wedge (flow_i^u \geq flow_i) \rightarrow H_{P,i}. \tag{11}$$

When the valve is healthy its actual flow will be between two thresholds ($flow_i^l$ and $flow_i^u$).

Definition 5 (Observation) *An observation α is an assignment to some or all inputs and outputs of a circuit* SD*.*

With an observation the state of the system at one specific point in time (a variable assignment α) is measured. The set of observations α is called OBS.

Definition 6 (Fault-Injection) *Given* SD *with fault variables* COMPS*, a fault-injection ϕ is an assignment to all fault variables in* COMPS*.*

With a fault injection it becomes possible to model faulty behaviour. Injecting a fault is done by forcing one term to be $\bot$.

Definition 7 *(Diagnostic System) A diagnostic system is defined as the triple* (SD, COMPS, OBS)*, with* SD *being the system description,* COMPS *being the set of components, and* OBS *being the set of observations.*

A diagnostic system contains all the information that a diagnosis algorithm needs to identify and locate faults within a system. With SD the causal relationships are known. COMPS shows which components are contained within the system and whether or not those components are healthy. OBS are the observations at one specific point in time.

Definition 8 (Diagnosis) *Given a fault-augmented model* SD *with fault variables* COMPS *and an observation α, a diagnosis ω is defined as an assignment to all fault variables in* COMPS *such that $\omega \models \mathrm{SD} \wedge \alpha$.*

Each diagnosis specifies that, given a fault-augmented model and some observation, one can obtain an assignment ω which states, which components exhibit faulty behaviour. Usually diagnosed systems contain hundreds or thousands of components (such as the larger systems in the ISCAS-85 benchmark). Usually, no intermediate values can be measured in these systems and the faults of some components can be masked by other components. Those components c_i that are not observable due to some dominating component c_j are said to be in the cone of c_j.

In many cases the size of individual diagnoses can be quite large and contain sometimes hundreds of components. Confronting an operator with such a large set of possible components to be checked and repaired is infeasible. Instead, the size of a diagnosis needs to be limited. To this end, minimal-cardinality diagnoses are introduced. A minimum cardinality diagnosis is a diagnosis that contains the smallest possible number of components.

Definition 9 *(Minimal-Cardinality Diagnosis) A Minimal Cardinality Diagnosis is a diagnosis ω', so that $|\omega'| \leq |\omega|$*

For diagnosis we start by describing the triple (SD, COMPS, OBS). SD is given by the set of SMT expression $\mathrm{SD} = \bigwedge_i \sigma_i \bigwedge_i \delta_i$. OBS are given by the input vector $\mathbf{u}(t)$. The component mode COMPS is described by the partial functions $r(x_i, \tau_i)$. The Boolean output of these functions is interpreted within the vector

$$\mathbf{C}' = \begin{bmatrix} \mathcal{I}(|\sigma_i(t) - \tilde{\sigma}_i(t)| \leq e) \\ \vdots \\ \mathcal{I}(|\sigma_j(t) - \tilde{\sigma}_j(t)| \leq e) \end{bmatrix} \tag{12}$$

where vector $\mathbf{C}'$ is the comparison between the observation $\sigma_i(t)$ and the model prediction $\tilde{\sigma}_i(t)$. If this comparison is smaller than some error bound e the corresponding component is healthy. If the elements of $\mathbf{C}'$ are semantically interpreted through an SMT solver, we obtain the diagnosis vector

$$\mathbf{C}' = [c_0 \quad \ldots \quad c_n]^T$$

with $c_i \in \{\top, \bot\}$. This vector shows for each component whether it is faulty or not, given the current observations from the sensors.

The numerical information for the statements σ_i is obtained through the observer-pattern from the state space model. By interpreting the statements it is possible to translate the numerical data within the state-space model into symbolic information used for diagnosis through the vector $\mathbf{C}'$. For each new time step the SMT expressions are reevaluated, thus generating new hypotheses. Reevaluation is necessary since we do not specify a sampling rate for the underlying data. Therefore, we start the diagnosis as soon as sufficient data is available. This increases computation, but will lead to faster diagnosis in the real world.

Given the vector $\mathbf{C}'$ as COMPS and the system description SD it is possible to employ any diagnosis algorithm such as CDA* (Williams and Ragno 2007), GDE (Reiter 1987), or SATbD (Metodi et al. 2014) to find minimum cardinality diagnoses. Here we have used Reiter's diagnosis lattice.

Automatic Derivation of the MBD Knowledge

Creating state space models manually is costly and error-prone. Instead, we take the required knowledge for SD and COMPS from the available engineering tools automatically. The structure of a process plant, the material flow, and the built-in components are available in a piping and instrumentation diagram (P&ID). It also contains the information about causal relationships between components. This information can be represented in a digraph, which makes up $\mathrm{M}(\mathcal{B})$. Additional information can be extracted automatically e.g. by using OCR technology (Arroyo et al. 2016). In (Barth et al. 2009) the authors present an approach to create object-oriented simulation models out of the derived data. In our approach, we automatically construct directed graphs by parsing and interpreting a proprietary format from P&ID diagrams.

Information such as component models is stored, for example, in AutomationML (AutomationML 2009). This includes data about geometry and kinematics, the plant topology (CAEX, (IEC 62424 2008)) and the control logic (PLCopen XML). For components using more complex models such as differential equations for tanks, AutomationML is extended with MathML.

Similar information is available for other types of systems. In electric circuits schematic diagrams are available to specify the connectivity and component models can be extracted from available truth tables.

This shows that, given a self-descriptive industrial process (Bunte, Diedrich, and Niggemann 2016) and the information from state-of-the-art engineering tools it is possible to realise our approach in a self-autonomous manner. Fully automating the generation of numerical and causal models remains a research challenge.

Evaluation

Empirical Evaluation

Table 1 shows the experiments of the simulated four-tank model for constant input stream, the injected faults and whether or not the fault was detected. An x in the column *Detected* denotes that the injected fault was among the result set of the diagnosis algorithm. This means the algorithm is complete. An x* denotes that exclusively the injected fault was detected, which corresponds to soundness of the algorithm. It must be noted here, that finding only the injected faults depends heavily on the granularity of the underlying data source. For example, if valve 5 stops working its flow would immediately go to 0. The sampling frequency is high enough to detect this decrease in the flow rate early enough that the water level in the tanks is not yet significantly affected. However, in large industrial plants sampling rates are often far lower. A faulty component might then only be recognised once its effects have propagated into other observations from other components. Further, in the semi- and non-observable cases not every status of every component is known. In this case, too, the set of possible faulty components will grow in size.

The algorithm was executed for 300 time-steps. As the criterion in Table 1 we evaluated the output of the diagnosis algorithm in the time step 101, which was directly after the fault had been injected. It is evident that due to the SMT logic constraints such as equation (9) every unexpected change in the behaviour of the components would be immediately detected.

The numbers in brackets in Table 1 show the results when the output of the diagnosis algorithm was evaluated directly before removing a fault at time-step 199. The number denotes the size of the minimum-cardinality set. Since in this time step the abnormal behaviour caused by the fault has reached many other components the result set also contains components apart from the injected faults.

Table 1: Recognized faults for experiments with constant input stream. Results with sinusoidal input stream are similar.

Index	**# Faults**	**Detected**
0	p_0	x* (11)
2	p_3	x* (3)
3	p_5	x* (3)
4	p_6	x* (3)
5	p_1, p_3	x* (5)
6	p_4, p_5	x* (6)

Beside the simulation of the four-tank model we used a quantitative simulation of the Tennessee Eastman process. The implementation of Downs et al. (Downs and Vogel 1993) was used which contains 20 different injected faults (process disturbances). However, the instrumentation of the simulated process is such that not all faults will be identified exactly. Table 2 shows the results for six experiments. The injected faults for IDV 16 through 20 have an insufficient description for validating results. The other faults not described contain fault modes that are not associated with single components and thus cannot be evaluated. This leaves us with the experiments shown in Table 2.

An x indicates that the faulty component was found as part of the minimal cardinality diagnosis and a - shows that the faulty component was not found. We were able to find all faults that had an identifiable component fault as the cause of the process disturbance. The change of input ratios can only be detected indirectly, since no observations are available at the inputs.

Table 2: Experimental results with the Tennessee Eastman process.

IDV	Fault isolated	Injected Fault
1	-	Feed ratio changed
6	x	Pipe A feed loss
8	x	Feed ratio changed
13	-	Reactor kinetics fault
14	x	Reactor cooling fault
15	x	Condenser cooling fault

Theoretical Evaluation

To model industrial processes with state-space models it is necessary to solve ordinary differential equations (ODE) such as equation (3). Finding solutions for ODEs was proven to be NP-hard (Ko 1983).

For diagnosis we make use of the SMT solver *z3* and calculate Reiter's diagnosis lattice. For the purposes presented here the *z3* solver converts the SMT expressions into a MAXSAT problem. The computationally hardest problem is solving the MAXSAT formulation which is known NP-hard (Impagliazzo and Paturi 1999).

Conclusion

In this paper we presented a novel approach to diagnose faults in industrial systems. Through an example four-tank system it was shown how to model a typical industrial use-case with state-space models. The first contribution of this paper is to show how an observer-pattern can be created which translates the numeric state-space model into Boolean residuals. The second contribution is our presented generic extension of MBD using satisfiability theory modulo linear arithmetic building on these residuals. This provides an abstraction from many kinds of numeric models into the symbolic reasoning of MBD. Thus, a standard MBD algorithm can be used to diagnose faults in many kinds of industrial systems. We successfully evaluated the presented approach through a common four-tank model and by using a simulation of the Tennessee Eastman process.

Future work includes making the state-space model more expressive by including a model of set-points. It is also necessary to evaluate the approach on an industrial system from areas such as process industry.

Acknowledgments

This work was developed within the Fraunhofer Cluster of Excellence "Cognitive Internet Technologies".

References

Arroyo, E.; Royston, S.; Fay, A.; Hoernicke, M.; and Rodriguez, P. 2016. From paper to digital. 65–69.

AutomationML. 2009. Spezifikation V1.1. www.automationml.org.

Barth, M.; Strube, M.; Fay, A.; Weber, P.; and Greifeneder, J. 2009. Object-oriented engineering data exchange as a base for automatic generation of simulation models. In *2009 35th Annual Conference of IEEE Industrial Electronics*, 2465–2470.

Brown, A. L. 1974. Qualitative knowledge, causal reasoning, and the localization of failures.

Buede, D. 2009. *The Engineering Design of Systems: Models and Methods*. John Wiley & Sons.

Bunte, A.; Diedrich, A.; and Niggemann, O. 2016. Semantics enable standardized user interfaces for diagnosis in modular production systems. In *International Workshop on the Principles of Diagnosis (DX)*.

Daigle, M. J.; Roychoudhury, I.; Biswas, G.; Koutsoukos, X. D.; Patterson-Hine, A.; and Poll, S. 2010. A comprehensive diagnosis methodology for complex hybrid systems: A case study on spacecraft power distribution systems. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans* 40(5):917–931.

Daigle, M.; Koutsoukos, X.; and Biswas, G. 2007. A discrete event approach to diagnosis of continuous systems. In *Proceedings of the 18th International Workshop on Principles of Diagnosis*, 259–266.

De Kleer, J., and Williams, B. C. 1987. Diagnosing multiple faults. *Artificial intelligence* 32(1):97–130.

de Kleer, J.; Janssen, B.; Bobrow, D. G.; Saha, T. K. B.; Moore, N. R.; and Sutharshana, S. 2013. Fault augmented modelica models. *The 24th International Workshop on Principles of Diagnosis* 71–78.

Downs, J. J., and Vogel, E. F. 1993. A plant-wide industrial process control problem. *Computers & chemical engineering* 17(3):245–255.

Feldman, A.; Provan, G.; and Van Gemund, A. 2010. Approximate model-based diagnosis using greedy stochastic search. *Journal of Artificial Intelligence Research* 38:371–413.

Ferracuti, F.; Giantomassi, A.; Longhi, S.; and Bergantino, N. 2011. Multi-scale pca based fault diagnosis on a paper mill plant. In *Emerging Technologies Factory Automation (ETFA), 2011 IEEE 16th Conference on*, 1–8.

Fränzle, M.; Hermanns, H.; and Teige, T. 2008. Stochastic satisfiability modulo theory: A novel technique for the analysis of probabilistic hybrid systems. In *International Workshop on Hybrid Systems: Computation and Control*, 172–186. Springer.

Grastien, A. 2013. Diagnosis of hybrid systems by consistency testing. In *24th International Workshop on Principles of Diagnosis (DX-13)*, 9–14. Citeseer.

IEC 62424. 2008. Festlegung für die Darstellung von Aufgaben der Prozessleittechnik in Fliessbildern und für den Datenaustausch zwischen EDV-Werkzeugen zur Fliessbilderstellung und CAE-Systemen.

Impagliazzo, R., and Paturi, R. 1999. Complexity of k-sat. In *Computational Complexity, 1999. Proceedings. Fourteenth Annual IEEE Conference on*, 237–240. IEEE.

Isermann, R. 2004. Model-based fault detection and diagnosis - status and applications. In *16th IFAC Symposium on Automatic Control in Aerospace*.

Khorasgani, H., and Biswas, G. 2017. Structural fault detection and isolation in hybrid systems. *IEEE Transactions on Automation Science and Engineering*.

Ko, K.-I. 1983. On the computational complexity of ordinary differential equations. *Information and control* 58(1-3):157–194.

Lee, W., and Weekman Jr, V. W. 1976. Advanced control practice in the chemical process industry: A view from industry. *AIChE Journal* 22(1):27–38.

Lee, E. 2008. Cyber physical systems: Design challenges. In *Object Oriented Real-Time Distributed Computing (ISORC), 2008 11th IEEE International Symposium on*, 363–369.

Matias, T.; Gabriel, D.; Souza, F.; Araujo, R.; and Costa Pereira, J. 2013. Fault detection and replacement of a temperature sensor in a cement rotary kiln. In *Emerging Technologies Factory Automation (ETFA), 2013 IEEE 18th Conference on*, 1–8.

Metodi, A.; Stern, R.; Kalech, M.; and Codish, M. 2014. A novel sat-based approach to model based diagnosis. *J. Artif. Int. Res.* 51(1):377–411.

Niggemann, O., and Lohweg, V. 2015. On the Diagnosis of Cyber-Physical Production Systems - State-of-the-Art and Research Agenda. In *Twenty-Ninth Conference on Artificial Intelligence (AAAI-15)*.

Niggemann, O.; Stein, B.; Vodenčarević, A.; Maier, A.; and Kleine Büning, H. 2012. Learning behavior models for hybrid timed systems. In *Twenty-Sixth Conference on Artificial Intelligence (AAAI-12)*, 1083–1090.

Prakash, O., and Samantaray, A. 2017. Model-based diagnosis and prognosis of hybrid dynamical systems with dynamically updated parameters. In *Bond Graphs for Modelling, Control and Fault Diagnosis of Engineering Systems*. Springer. 195–232.

Provan, G. 2009. Model abstractions for diagnosing hybrid systems. In *Proceedings of the 20th International Workshop on Principles of Diagnosis, DX-09, Stockholm, Sweden*, 321–328. Citeseer.

Rajkumar, R. R.; Lee, I.; Sha, L.; and Stankovic, J. 2010. Cyber-physical systems: The next computing revolution. In *Proceedings of the 47th Design Automation Conference*, DAC '10, 731–736. New York, NY, USA: ACM.

Reiter, R. 1987. A theory of diagnosis from first principles. *Artificial intelligence* 32(1):57–95.

Roychoudhury, I.; Daigle, M. J.; Biswas, G.; and Koutsoukos, X. 2011. Efficient simulation of hybrid systems: A hybrid bond graph approach. *Simulation* 87(6):467–498.

Struss, P., and Ertl, B. 2009. Diagnosis of bottling plants - first success and challenges. In *20th International Workshop on Principles of Diagnosis, Stockholm*.

Struss, P. 1997. Fundamentals of model-based diagnosis of dynamic systems. In *IJCAI (1)*, 480–485.

Williams, B. C., and Ragno, R. J. 2007. Conflict-directed a* and its role in model-based embedded systems. *Discrete Applied Mathematics* 155(12):1562 – 1595. SAT 2001, the Fourth International Symposium on the Theory and Applications of Satisfiability Testing.

Windmann, S.; Jiao, S.; Niggemann, O.; and Borcherding, H. 2013. A Stochastic Method for the Detection of Anomalous Energy Consumption in Hybrid Industrial Systems. In *INDIN*.