# Remote Diagnosis of Timed I/O-Automata

**Thorsten Schlage** [1], **Jan Lunze** [1]

[1] *Institute of Automation and Computer Control,*
*Ruhr-Universität Bochum, D-44780 Bochum, Germany*
*Schlage@atp.rub.de*
*Lunze@atp.rub.de*

## ABSTRACT

This paper considers the consistency-based remote diagnosis of discrete-event systems, where the diagnostic task is decomposed into an on-board and an off-board task. The system to be diagnosed is modelled by a new type of timed automata possessing discrete inputs and outputs. Faults are modelled as parameters of the edges and invariants of the automata. A consistency-based diagnostic method is derived for this model class by extending a method developed for timed automata without inputs and outputs. This method can be used for fault detection on the on-board component and for fault identification on the off-board component of the remote diagnostic system. The results are illustrated by an example.

## 1 INTRODUCTION

Modern technological systems are subject to faults, which may lead to down time and damage to men and environment. The aim of fault diagnosis is to decide whether faults have occurred in the system and to identify them as early as possible.

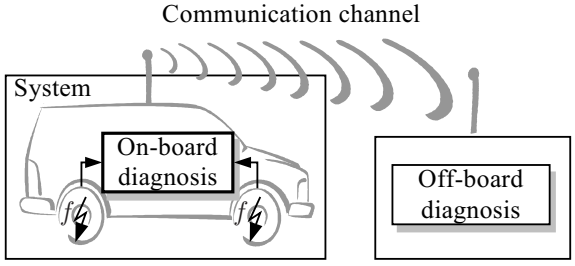


Figure 1: Structure of remote diagnosis

Diagnostic tasks require considerable memory capacity and computing effort, which often exceed the capacity of the system's equipment. To take these limitations into account, different strategies have been developed in (Schumann and Pencolé, 2006), (Zhang and Ding, 2006), (Al-Salami *et al.*, 2008) and (Zhang *et al.*, 2004). In this paper the concept of remote diagnosis shown in Fig. 1 is adopted from (Fritsch *et al.*, 2006), which decomposes the diagnostic problem into an on-board and an off-board diagnostic subproblem as follows:

- **On-board fault detection.** The on-board diagnostic system solves the fault detection problem and controls the data communication between the on-board component and the off-board component.
- **Off-board fault identification.** The fault identification task is solved by the off-board diagnostic system.

The decomposition of the diagnostic problem reduces the computational requirements on the on-board component with respect to memory and computing effort.

This paper investigates remote diagnosis of discrete-event systems (Fig. 2) which are represented by timed automata. The main motivation for dealing with timed models is that the temporal distance between events includes important information for diagnosis.

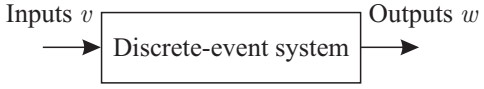


Figure 2: Discrete-event system with inputs and outputs

The contribution of this paper is threefold: First, a new type of timed input/output-automata (I/O-automata) is proposed extending the timed automata introduced by (Alur and Dill, 1994) with inputs and outputs. In (Kaynar *et al.*, 2003) timed I/O-automata are presented, where the set of external actions is partitioned into input and output actions. In contrast, in this contribution

the inputs and the outputs of the automaton are combined as input/output-pairs (I/O-pairs), where the automaton's input causes a state transition, while the output occurs as a result of the state transition. Compared with (Supavatanakul, 2004) and (Schlage and Lunze, 2009), the new model can have an arbitrary number of clocks and allows, among others, to model the communication network.

The second contribution is a new diagnostic method. In the literature, there are only few model-based diagnostic methods using timed automata, for instance, in (Supavatanakul, 2004), (Bouyer *et al.*, 2005) and (Tripakis, 2002). Extending the idea of (Tripakis, 2002), this paper presents a consistency-based diagnostic method for the class of timed I/O-automata proposed before.

The third contribution is a new scheme for remote diagnosis (Fig. 3). The on-board component applies the diagnostic method to the model of the nominal system to detect faults, whereas the off-board component uses this method together with the models of the faulty system for fault identification.

Instead of modelling faults as unobservable events, faults are represented as changes of the behaviour of the automaton. This approach is motivated by our field of application. In technological systems faults do not appear as events after which the system behaves like the nominal one, but faults change the properties of the system.

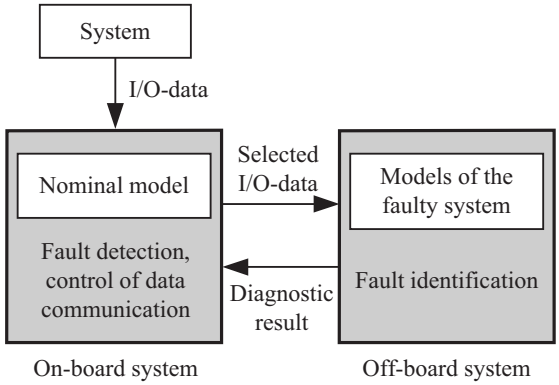


Figure 3: Decomposition of the diagnostic tasks

The paper proceeds as follows. In Section 2 the new type of timed I/O-automata is presented. The consistency-based diagnosis of timed I/O-automata is explained in Section 3. In Section 4 the remote diagnosis using the consistency-based diagnostic approach is described and illustrated by an example in Section 5. Section 6 concludes the paper.

## 2 TIMED I/O-AUTOMATA

This section defines a new type of timed automata, which may incorporate an arbitrary number of clocks, whose transitions from one discrete state to another are associated with I/O-pairs and whose behaviour changes in the presence of faults. First, the basic modelling idea, which is similar to (Alur and Dill, 1994), is explained.

### 2.1 Preliminaries

Throughout this paper, $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ denotes the set of $m$ *clock* variables. A *clock valuation* $r$ on $\mathcal{C}$ is a map $r : \mathcal{C} \mapsto \mathbb{R}_{\geq 0}$ which assigns a non-negative value to a clock $c \in \mathcal{C}$, e.g. $r(c_1) = 1$. In the following, $r_\mathcal{C} = (r(c_1), r(c_2), \ldots, r(c_m))$ denotes the $m$-tuple of clock valuations. Given a clock valuation $r_\mathcal{C}$ and a time $\tau \in \mathbb{R}_{\geq 0}$, $r_\mathcal{C} + \tau$ denotes the valuation defined by $(r(c_1) + \tau, r(c_2) + \tau, \ldots, r(c_m) + \tau)$. A valuation $r[C := 0]$ for a set of clocks $C \subseteq \mathcal{C}$ is defined by

$$r[C := 0](c) = \begin{cases} 0 & \text{if } c \in C \\ r(c) & \text{otherwise,} \end{cases}$$

where the values of the clocks $c \in C$ are reset to zero and $r_\mathcal{C}[C := 0]$ denotes the corresponding $m$-tuple of clock valuations. Furthermore, $0_\mathcal{C} = (r(c_1) = 0, r(c_2) = 0, \ldots, r(c_m) = 0)$ is a valuation assigning zero to all clocks $c \in \mathcal{C}$.

A *clock constraint* $\varphi$ is any finite conjunction of the expressions of the form $c_i \bowtie k$, where $c_i$ is a clock, $\bowtie \in \{<, >, \geq, \leq, =\}$ and $k \in \mathbb{N}$, where $\mathbb{N}$ denotes the set of integers. The set of clock constraints on $\mathcal{C}$ is denoted by $\Phi(\mathcal{C})$.

### 2.2 Timed I/O-automata

A timed I/O-automaton is a tuple

$$\mathcal{A} = (\mathcal{Z}, \mathcal{V}, \mathcal{W}, \mathcal{E}, \mathcal{C}, I, \mathcal{Z}_0), \tag{1}$$

where:

- $\mathcal{Z} = \{z_1, z_2, \ldots, z_n\}$ is a finite set of *discrete states*,
- $\mathcal{V} = \{v_1, v_2, \ldots, v_l\}$ is a finite set of *inputs*,
- $\mathcal{W} = \{w_1, w_2, \ldots, w_k\}$ is a finite set of *outputs*,
- $\mathcal{E} = \{e_1, e_2, \ldots, e_p\}$ is a finite set of *edges*,
- $\mathcal{C} = \{c_1, c_2, \ldots, c_m\}$ is a finite set of $m$ *clock* variables,
- $\mathcal{Z}_0 \subseteq \mathcal{Z}$ is the set of potential *initial* discrete states,
- $I(z)$ is the *invariant function* which associates with each discrete state $z \in \mathcal{Z}$ a clock constraint $\varphi \in \Phi(\mathcal{C})$. A discrete state $z$ must be left before its invariant becomes unsatisfied.

The set of inputs $\mathcal{V}$ is partitioned into a set $\mathcal{V}_\mathrm{O}$ of *observable* inputs and a set $\mathcal{V}_\mathrm{U}$ of *unobservable* inputs. Moreover, the set of outputs $\mathcal{W}$ is partitioned into a set $\mathcal{W}_\mathrm{O}$ of *observable* outputs and a set $\mathcal{W}_\mathrm{U}$ of *unobservable* outputs. In the following, it is assumed that

$\mathcal{V}_{\mathrm{U}} = \{\varepsilon\}$ and $\mathcal{W}_{\mathrm{U}} = \{\varepsilon\}$ hold, where $\varepsilon$ is the empty symbol.

An edge $e = (z', w, z, v, \varphi, C)$ represents a transition from the discrete state $z \in \mathcal{Z}$ towards the discrete successor state $z' \in \mathcal{Z}$, where $v \in \mathcal{V}$ is the input, $w \in \mathcal{W}$ is the output, $\varphi \in \Phi(\mathcal{C})$ is a clock constraint, and $C \subseteq \mathcal{C}$ gives the set of clocks to be reset to zero during the transition. An edge $e$ is enabled if the clock constraint $\varphi$ is satisfied and the system has the input $v$, otherwise, $e$ is disabled. An automaton can only change its discrete state from $z$ towards $z'$ using an edge $e$ if $e$ is enabled. During this transition the output $w$ is generated and all clocks $c \in C$ are reset to zero. Note that the automaton may be nondeterministic.

The representation of a timed I/O-automaton by means of an automaton graph is depicted in Fig. 4, where the only initial discrete state $z_0 = 1$ is denoted by an additional arrow.

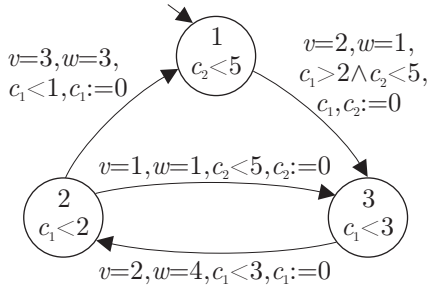


Figure 4: Graph of a timed I/O-automaton

The state of the timed automaton $\mathcal{A}$ is a pair $s = (z, r_{\mathcal{C}})$, where $z \in \mathcal{Z}$ is a discrete state and $r_{\mathcal{C}}$ is a clock valuation on $\mathcal{C}$ such that $r_{\mathcal{C}}$ satisfies the invariant $I(z)$. The set of states is denoted by $\mathcal{S} \subseteq \mathcal{Z} \times \mathbb{R}^m_{\geq 0}$. The set of potential *initial states* of $\mathcal{A}$ is $\mathcal{S}_0 = \mathcal{Z}_0 \times \overline{0}_{\mathcal{C}}$. It is required that $0_{\mathcal{C}}$ satisfies the invariant $I(z_0)$ for all $z_0 \in \mathcal{Z}_0$.

There are two different types of state transitions in $\mathcal{A}$. Firstly, the state $s$ can change due to the elapse of time

$$\mathrm{succ}_\tau(s) = \begin{cases} (z, r_{\mathcal{C}} + \tau) & \text{if } s = (z, r_{\mathcal{C}}) \text{ is a state of } \mathcal{A} \text{ and } \forall \tau' \leq \tau,\ r_{\mathcal{C}} + \tau' \text{ satisfies } I(z) \\ \text{not defined} & \text{otherwise,} \end{cases}$$

where $\mathrm{succ}_\tau(s)$ is the successor state of the automaton which is reached from $s$ after the time $\tau$. Secondly, the state of the automaton can change due to an edge $e$

$$\mathrm{succ}_e(s) = \begin{cases} (z', r_{\mathcal{C}}[C := 0]) & \text{if } s = (z, r_{\mathcal{C}}) \text{ is a state of } \mathcal{A},\ r_{\mathcal{C}} \text{ satisfies } \varphi,\ \mathcal{A} \text{ has the input } v \text{ and } r_{\mathcal{C}}[C := 0] \text{ satisfies } I(z') \\ \text{not defined} & \text{otherwise,} \end{cases}$$

where $\mathrm{succ}_e(s)$ is the successor state of the automaton which is reached from $s$ using the edge $e$.

The inputs $v \in \mathcal{V}$ and outputs $w \in \mathcal{W}$ of the automata are combined as I/O-pairs $(v, w) \in \mathcal{IO}$, where the input causes a state transition if the corresponding clock constraints are satisfied, while the output occurs as a result of the state transition. The set of I/O-pairs $\mathcal{IO} = \mathcal{V} \times \mathcal{W}$ is partitioned into a set $\mathcal{IO}_{\mathrm{O}}$ of observable I/O-pairs and a set $\mathcal{IO}_{\mathrm{U}}$ of unobservable I/O-pairs. An I/O-pair $(v, w) \in \mathcal{IO}$ is unobservable if $v \in \mathcal{V}_{\mathrm{U}}$ and $w \in \mathcal{W}_{\mathrm{U}}$ holds, otherwise, it is observable.

A timed *input/output-sequence* (I/O-sequence) $B(t_h)$ is described by

$$B(t_h) = (\tau_0, (v_0, w_0); \tau_1, (v_1, w_1); \ldots; \tau_h, (v_h, w_h)),$$

where $v_i$ and $w_i$ are the $i$-th values of the I/O-sequence, and $\tau_i \geq 0$ is the time between the $(i-1)$-st and the $i$-th I/O-pair. The set of all I/O-sequences is denoted by $\mathcal{B}$. Moreover,

$$time(B(t_h)) := \sum_{i=0}^{h} \tau_i$$

denotes the sum of all $\tau_i \in B(t_h)$. Consequently, for the I/O-sequence $B(t_h)$ the relation $time(B(t_h)) = t_h$ holds.

For the set of timed I/O-sequences the equations $\mathcal{B}_{\mathrm{O}} \cup \mathcal{B}_{\mathrm{U}} = \mathcal{B}$ and $\mathcal{B}_{\mathrm{O}} \cap \mathcal{B}_{\mathrm{U}} = \emptyset$ hold, where $\mathcal{B}_{\mathrm{O}}$ is the set of observable I/O-sequences and $\mathcal{B}_{\mathrm{U}}$ is the set of unobservable I/O-sequences. An I/O-sequence $B(t_h)$ is called observable if all I/O-pairs $(v, w) \in B(t_h)$ are observable, otherwise, the I/O-sequence is called unobservable. In the following an observable I/O-sequence is denoted by $B_{\mathrm{O}} \in \mathcal{B}_{\mathrm{O}}$, whereas $B_{\mathrm{U}} \in \mathcal{B}_{\mathrm{U}}$ denotes an unobservable I/O-sequence. For a given unobservable I/O-sequence $B_{\mathrm{U}}(t_h)$, the corresponding observable I/O-sequence $B_{\mathrm{O}}(t_h) = obsv(B_{\mathrm{U}}(t_h), \mathcal{IO}_{\mathrm{U}})$ is obtained by erasing all unobservable I/O-pairs $(v, w) \in \mathcal{IO}_{\mathrm{U}}$ from $B_{\mathrm{U}}(t_h)$ and summing all times $\tau$ between two successive observable I/O-pairs in the resulting I/O-sequence.

The behaviour $\mathcal{B}_{\mathcal{A}}$ of the timed automaton $\mathcal{A}$ is the set of observable I/O-sequences $B_{\mathrm{O}}(t_h) \in \mathcal{B}_{\mathrm{O}}$ which are consistent with the automaton.

### 2.3 Timed I/O-automata subject to faults

The application of timed I/O-automata for fault diagnosis requires the extension of the timed I/O-automaton (1) to capture the faulty behaviour of the system. Therefore, the timed automaton has to include the dependence upon the fault $f$. The timed I/O-automaton extended for fault diagnosis is a tuple

$$\mathcal{A}_{\mathcal{F}} = (\mathcal{Z}, \mathcal{V}, \mathcal{W}, \mathcal{E}, \mathcal{C}, \mathcal{F}, I, \mathcal{Z}_0), \tag{2}$$

where $\mathcal{F} = \{f_0, f_1, \ldots, f_s\}$ is the set of considered faults with $f_0$ denoting the faultless case.

In technological systems, faults change the system properties as a leakage in a batch process disables

a reactor to reach a high liquid level. In other words, faults change the edge set $\mathcal{E}$. An edge $e = (z', w, z, v, \varphi, C, f)$ is only enabled if the clock constraint $\varphi$ is satisfied, the system has the input $v$ and the fault $f$ has occurred, otherwise, $e$ is disabled.

Furthermore, the invariant function $I(z, f)$ depends upon the faults $f \in \mathcal{F}$. If the fault $f$ has occurred, the discrete state $z \in \mathcal{Z}$ must be left before the corresponding invariant $I(z, f)$ is violated.

Figure 5 illustrates the modelling idea by means of a timed I/O-automaton of a system, in which the faults $f_0$, $f_1$ and $f_2$ can occur. Evidently, the edges and the invariants differ for the considered faults.

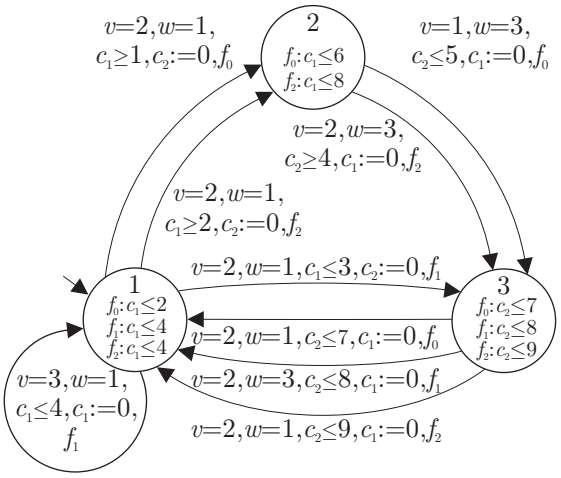


Figure 5: Graph of a timed automaton for $\mathcal{F} = \{f_0, f_1, f_2\}$

For a set of faults $\mathcal{F} = \{f_0, f_1, \ldots, f_s\}$ the automaton $\mathcal{A}_\mathcal{F}$ can be interpreted as a set $\mathcal{N}_\mathcal{A} = \{\mathcal{A}_{f_0}, \mathcal{A}_{f_1}, \ldots, \mathcal{A}_{f_s}\}$ of timed automata

$$\mathcal{A}_{f_i} = (\mathcal{Z}, \mathcal{V}, \mathcal{W}, \mathcal{E}(f_i), \mathcal{C}, I(z, f_i), \mathcal{Z}_0),$$

where $\mathcal{E}(f_i) \subset \mathcal{E}$ denotes the set of edges of the automaton $\mathcal{A}_\mathcal{F}$ labeled with $f_i$ and $I(z, f_i)$ denotes the invariant function concerning the fault $f_i$. For the set of edges the relation

$$\bigcup_{f_i \in \mathcal{F}} \mathcal{E}(f_i) = \mathcal{E}$$

holds. Thus, for a set of faults $\mathcal{F}$ there are $s + 1$ timed automata to be used for fault diagnosis, which generally differ in their behaviour so that a consistency-based diagnostic method can be applied to detect and identify a fault. The behaviour of the automaton $\mathcal{A}_{f_i}$ is denoted by $\mathcal{B}_\mathcal{A}(f_i)$. In the following only single, permanent faults are considered.

## 3 CONSISTENCY-BASED DIAGNOSIS OF TIMED I/O-AUTOMATA

This section presents a consistency-based diagnostic method for the proposed timed I/O-automata which is based on the diagnostic method proposed in (Tripakis, 2002).

The principle of consistency-based diagnosis is to check whether or not the measured observable I/O-sequence $B_\mathrm{O}(t_h)$ is consistent with the timed I/O-automaton $\mathcal{A}$.

**Definition 1 (Consistency (Blanke *et al.*, 2006))** *An automaton $\mathcal{A}$ is called consistent with a measured observable I/O-sequence $B_\mathrm{O}(t_h)$ if $B_\mathrm{O}(t_h) \in \mathcal{B}_\mathcal{A}$ holds.*

The consistency-based diagnostic method requires to find for the given input sequence a state sequence that satisfies the clock constraints of the corresponding edges and the invariants $I$ and on which the automaton generates the output sequence measured. Hence, diagnosis is based on state observation, which concerns the problem of determining the current state $s$ of the discrete-event system for the measured I/O-sequence $B_\mathrm{O}(t_h)$. Since nondeterministic timed I/O-automata are considered, the solution to the observation problem is, in general, not unique.

Given a set of states $S \in \mathcal{S}$ and an I/O-pair $(v, w) \in \mathcal{IO}$, the set of states that can be reached from a state $s \in S$ using an edge $e \in \mathcal{E}(v, w)$ can be determined by means of the transition function

$$G_\mathrm{O}(S, (v, w)) = \{\mathrm{succ}_e(s) | s \in S, e \in \mathcal{E}(v, w)\}, \quad (3)$$

where $\mathcal{E}(v, w)$ is the set of all edges $e \in \mathcal{E}$ labeled with the input $v$ and the output $w$. Moreover, given a set of states $S \in \mathcal{S}$ and a time $\tau$, the set of states that can be reached from a state $s \in S$ in exactly the time $\tau$ using at most the edges labeled with unobservable I/O-pairs can be determined by means of the transition function

$$G_\mathrm{U}(S, \tau) = \{S(s, B(\tau)) | s \in S, obsv(B(\tau), \mathcal{IO}_\mathrm{U}) = \tau\}, \quad (4)$$

where $S(s, B(\tau))$ is the set of all states reachable from $s$ via $B(\tau)$. Note that Eqns. (3) and (4) could result in an empty set of states if no corresponding transitions exist.

The set of states which can be reached from the set of initial states $S_0$ measuring the I/O-sequence $B_\mathrm{O}(t_h)$ can be determined using the function $E_\mathcal{A} : 2^\mathcal{S} \times \mathcal{B}_\mathrm{O} \to 2^\mathcal{S}$, which is defined recursively as follows:

$$\begin{aligned} E_\mathcal{A}(S, \epsilon) &= S, \\ E_\mathcal{A}(S, (\tau_i, (v_i, w_i); \ldots; \tau_h, (v_h, w_h))) &= \\ E_\mathcal{A}(G_\mathrm{U}(S, \tau_i), ((v_i, w_i); \ldots; \tau_h, (v_h, w_h))), & \\ E_\mathcal{A}(S, ((v_i, w_i); \tau_{i+1}, \ldots; \tau_h, (v_h, w_h))) &= \\ E_\mathcal{A}(G_\mathrm{O}(S, (v_i, w_i)), (\tau_{i+1}, \ldots; \tau_h, (v_h, w_h))). & \end{aligned}$$

Furthermore, for checking the emptiness of a state set $S$, the function $D : 2^\mathcal{S} \to \{0, 1\}$ is defined as follows:

$$D(S) = \begin{cases} 1 & \text{if } S \neq \emptyset \\ 0 & \text{otherwise.} \end{cases} \quad (5)$$

A nonempty set of states, which can be reached from $S_0$ measuring $B_{\mathrm{O}}(t_h)$ and determined by $E_{\mathcal{A}}(S_0, B_{\mathrm{O}}(t_h))$, indicates that the timed automaton $\mathcal{A}$ is consistent with the I/O-sequence $B_{\mathrm{O}}(t_h)$. Thus, the timed automaton $\mathcal{A}$ is consistent with $B_{\mathrm{O}}(t_h)$ if

$$D(E_{\mathcal{A}}(S_0, B_{\mathrm{O}}(t_h))) = 1 \tag{6}$$

holds, otherwise, it is inconsistent with $B_{\mathrm{O}}(t_h)$.

The on-line procedure of checking the consistency of an I/O-sequence $B_{\mathrm{O}}(t_h)$ with a timed automaton $\mathcal{A}$ is summarised in Algorithm 1. The observed state set or the result of the consistency test, respectively, are updated whenever an observable I/O-pair is measured or the timer $T$ is equal to the step size $\Delta\tau$. This approach ensures that an inconsistency can even be detected if no observable I/O-pair is measured. Moreover, the step size $\Delta\tau$ can be chosen depending on the computing power available for diagnosis. A smaller step size guarantees a more frequent update of the consistency result, which generally improves the overall diagnosis.

**Algorithm 1** On-line consistency test

**Require:**
- Model $\mathcal{A}$
- Step size $\Delta\tau$
- Measured I/O-sequence $B_{\mathrm{O}}(t_h)$ with increasing $t_h$

**Find:**
- Is $\mathcal{A}$ consistent with $B_{\mathrm{O}}(t_h)$?

**Initialisation:**
Set timer $T = 0$
Set $S = S_0$
**loop**
**WAIT UNTIL** $\big($(I/O-pair $(v_i, w_i)$ is measured at time $T$) **or** $(T = \Delta\tau)\big)$
**IF** (I/O-pair $(v_i, w_i)$ is measured) **THEN**
$S = G_{\mathrm{O}}(G_{\mathrm{U}}(S, T), (v_i, w_i))$
**ELSE IF** $T = \Delta\tau$ **THEN**
$S = G_{\mathrm{U}}(S, T)$
**END IF**
Set $D(S)$ according to Eqn. (5)
**IF** $(D(S) = 1)$ **THEN**
$\mathcal{A}$ is consistent with $B_{\mathrm{O}}(t_h)$
**ELSE**
$\mathcal{A}$ is inconsistent with $B_{\mathrm{O}}(t_h)$
**STOP ALGORITHM**
**END IF**
Set timer $T = 0$
**end loop**

**Result:**
- $\mathcal{A}$ is consistent with $B_{\mathrm{O}}(t_h)$ if Eqn. (6) holds

For diagnosis, the consistency of the models $\mathcal{A}_{f_i}$ of the system subject to the faults $f_i \in \mathcal{F} = \{f_0, f_1, \ldots, f_s\}$ with the measured I/O-sequence $B_{\mathrm{O}}(t_h)$ is checked. The inconsistency of $\mathcal{A}_{f_i}$ with $B_{\mathrm{O}}(t_h)$ means that the fault $f_i \in \mathcal{F}$ cannot have occurred. Thus, the faults for which the behaviour $\mathcal{B}(f_i)$ is consistent with the I/O-sequence $B_{\mathrm{O}}(t_h)$ form the set of fault candidates:

$$\mathcal{F}^*(t_h) = \{f | f \in \mathcal{F}, D(E_{\mathcal{A}_f}(S_0, B_{\mathrm{O}}(t_h))) = 1\}.$$

## 4 REMOTE DIAGNOSIS

### 4.1 On-board diagnosis

The on-board diagnostic system solves the fault detection task by testing the consistency of the measured I/O-sequence $B_{\mathrm{O}}(t_h)$ with the timed automaton $\mathcal{A}_{f_0}$ of the faultless system using Algorithm 1. For on-board diagnosis, the step size $\Delta\tau$ must be adapted to the limited computing power of the on-board diagnostic system.

The system to be diagnosed is known to be faulty if the timed automaton $\mathcal{A}_{f_0}$ is inconsistent with the measured I/O-sequence $B_{\mathrm{O}}(t_h)$. However, in the sense of consistency-based diagnosis, the fault $f_0$ of the faultless case belongs to the set of fault candidates if the timed automaton $\mathcal{A}_{f_0}$ is consistent with the I/O-sequence. The on-board fault detection is summarised in Algorithm 2.

**Algorithm 2** On-board fault detection

**Require:**
- Model $\mathcal{A}_{f_0}$
- Step size $\Delta\tau$
- Measured I/O-sequence $B_{\mathrm{O}}(t_h)$ with increasing $t_h$

**Find:**
- Is a fault $f_i \neq f_0$ detected?

Consistency test: For $\mathcal{A}_{f_0}$ perform Algorithm 1
**IF** $\mathcal{A}_{f_0}$ is consistent with $B_{\mathrm{O}}(t_h)$ **THEN**
A fault $f_i \neq f_0$ is not detected $(f \in \mathcal{F}^*(t_h))$
**ELSE**
A fault $f_i \neq f_0$ is detected $(f \notin \mathcal{F}^*(t_h))$
**END IF**

**Result:**
- Fault $f_i \neq f_0$ detected: yes/no

As a further task the on-board component has to provide the necessary signals for the off-board component. As proposed by (Schlage and Lunze, 2009), the measured I/O-sequence $B_{\mathrm{O}}(t_h)$ is sent from the on-board towards the off-board component as data packets

$$\boldsymbol{p}_k = \begin{pmatrix} v_k \\ w_k \\ \tau_k \\ k \end{pmatrix},$$

where $k$ is the counter of events, which is used to detect packet losses and to ensure the right order of the received packets.

### 4.2 Off-board diagnosis

The off-board diagnostic system solves the fault identification task. In order to determine which faults might have occurred in the system, the off-board diagnostic system checks the consistency of the timed automata of the faulty system $\mathcal{A}_{f_i}$, $i = 1, \ldots, s$, with the received I/O-sequence $B_{\mathrm{O}}(t_h)$ using Algorithm 1. Considering the high computing performance of the off-board diagnostic system, the step size $\Delta\tau$ of Algorithm 1 can be chosen smaller than the step size on the on-board component.

The faults $f_i \in \mathcal{F} \setminus f_0$, for which the I/O-sequence $B_{\mathrm{O}}(t_h)$ is consistent with the corresponding timed automaton $\mathcal{A}_{f_i}$, may have occurred in the system and form the set of potential faults $\mathcal{F}_{\mathrm{I}}(t_h)$, which can be determined by Algorithm 3. Assuming a perfect data communication between the on-board and the off-board component the relation $\mathcal{F}_{\mathrm{I}}(t_h) = \mathcal{F}^*(t_h) \setminus f_0$ holds for all time.

A fault $f_i \in \mathcal{F} \setminus f_0$ is unambiguously identified on the off-board component if $\mathcal{F}_{\mathrm{I}}(t_h)$ is a singleton and the faultless case $f_0$ is excluded from the set of fault candidates on the on-board component.

**Algorithm 3** Off-board fault identification

**Require:**
- Models $\mathcal{A}_{f_1}, \mathcal{A}_{f_2}, \ldots, \mathcal{A}_{f_s}$
- Step size $\Delta\tau$
- Measured I/O-sequence $B_{\mathrm{O}}(t_h)$ with increasing $t_h$

**Find:**
- Set of potential faults $\mathcal{F}_{\mathrm{I}}(t_h) \subset \mathcal{F}$, $f \in \mathcal{F}_{\mathrm{I}}$

**FOR ALL** $\mathcal{A}_{f_i}, i = 1, \ldots, s$ **DO**
  Consistency test: Perform Algorithm 1
  **IF** $\mathcal{A}_{f_i}$ is consistent with $B_{\mathrm{O}}(t_h)$ **THEN**
    $f_i \in \mathcal{F}_{\mathrm{I}}(t_h)$
  **ELSE**
    $f_i \notin \mathcal{F}_{\mathrm{I}}(t_h)$
  **END IF**
**END FOR**

**Result:**
- Set of potential faults $\mathcal{F}_{\mathrm{I}}(t_h)$ for increasing $t_h$, with $\mathcal{F}_{\mathrm{I}}(t_h) = \mathcal{F}^*(t_h) \setminus f_0$

The result of the off-board diagnosis is submitted from the off-board diagnostic system towards the on-board diagnostic system (Fig. 3). Thus, the on-board system is able to take countermeasures if a fault $f_i \in \mathcal{F} \setminus f_0$ is unambiguously identified.

## 5 EXAMPLE

Figures 6 and 7 show the timed automata $\mathcal{A}_{f_0}$, $\mathcal{A}_{f_1}$ and $\mathcal{A}_{f_2}$, which result from the timed automaton depicted in Fig. 5. While the automaton $\mathcal{A}_{f_0}$ is used for on-board fault-detection, the automata $\mathcal{A}_{f_1}$ and $\mathcal{A}_{f_2}$ are used for off-board fault identification. In the following, a perfect data transmission from the on-board towards the off-board component is assumed.

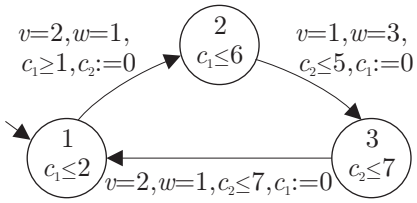


Figure 6: Automaton $\mathcal{A}_{f_0}$ used for fault detection

For the sake of simplicity, all inputs $v \in \mathcal{V}$ and outputs $w \in \mathcal{W}$ of the automata $\mathcal{A}_{f_i}$, $i = 0, 1, 2$, are observable. The initial state is $s_0 = (1, 0\,\mathrm{s}, 0\,\mathrm{s})$. While the step size $\Delta\tau_{\mathrm{on}} = 0.5\,\mathrm{s}$ of the consistency test according to Algorithm 1 is used for on-board diagnosis, the off-board diagnosis uses the step size $\Delta\tau_{\mathrm{off}} = 0.2\,\mathrm{s}$.

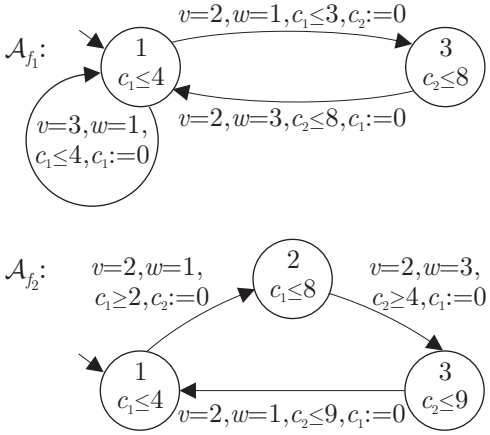


Figure 7: Automata $\mathcal{A}_{f_1}$ and $\mathcal{A}_{f_2}$ used for fault identification

It is assumed that the fault $f_2$ has occurred and the I/O-sequence

$$B_{\mathrm{O}}(11\,\mathrm{s}) = (3\,\mathrm{s}, (2,1); 4\,\mathrm{s}, (2,3); 4\,\mathrm{s}, (2,1))$$

is measured.

In Fig. 8 the results of the on-board and off-board diagnosis are shown. The corresponding data which are transmitted from the on-board towards the off-board component are depicted in Fig. 9.

The consistency check of the measured I/O-sequence with $\mathcal{A}_{f_0}$ results in an inconsistency at time $t = 2.5\,\mathrm{s}$ as the invariant $I(z = 1)$ of the automaton $\mathcal{A}_{f_0}$ is violated. Consequently, a fault $f_i \neq f_0$ is detected on the on-board component. Afterwards, the on-board component only controls the data traffic towards the off-board component while the fault identification task is solved by the off-board component.

The first data packet $\boldsymbol{p}_1 = (2, 1, 3\,\mathrm{s}, 1)^T$ is sent at time $t = 3\,\mathrm{s}$. The consistency check shows that the re-

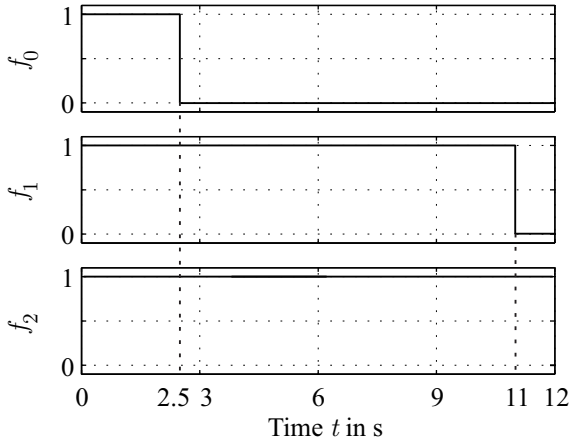


Figure 8: Results of the on-board and off-board diagnosis (1: consistency, 0: inconsistency)

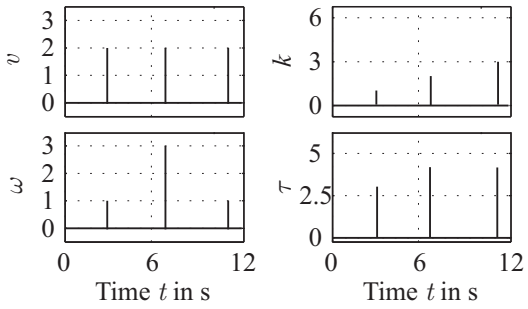


Figure 9: Transmitted data

ceived I/O-pair $(v_1 = 2, w_1 = 1)$ is consistent with the automata $\mathcal{A}_{f_1}$ and $\mathcal{A}_{f_2}$ as Eqn. (6) holds for both faults. While the automaton $\mathcal{A}_{f_1}$ changes its discrete state from $z = 1$ towards $z = 3$, the discrete state of $\mathcal{A}_{f_2}$ changes from $z = 1$ towards $z = 2$. Thereby, the clock $c_2$ of each automaton is reset to zero.

During the subsequent diagnosis, the off-board component receives the second data packet $\boldsymbol{p}_2 = (2, 3, 4\,\mathrm{s}, 2)^T$ at time $t = 7$ s and the third data packet $\boldsymbol{p}_3 = (2, 1, 4\,\mathrm{s}, 3)^T$ at time $t = 11$ s. The third data packet leads to an inconsistency of the automaton $\mathcal{A}_{f_1}$ with the measured I/O-sequence because for the I/O-pair $(v_3 = 2, w_3 = 1)$ a state transition from the state $s = (1, 4\,\mathrm{s}, 8\,\mathrm{s})$ does not exist. Therefore, the fault $f_1$ is excluded from the set of potential faults $\mathcal{F}_\mathrm{I}$. Consequently, the off-board diagnostic system identifies the fault $f_2$ at time $t = 11$ s as the automaton $\mathcal{A}_{f_2}$ is still consistent with the measurement.

## 6 CONCLUSION AND FUTURE WORK

In this paper a novel method for consistency-based remote diagnosis using a new type of timed I/O-automata, whose behaviour changes in the presence of faults, has been proposed. The new diagnostic approach has been illustrated by an example.

Due to the utilisation of remote resources, the limitations of the data communication networks have to be taken into account for remote diagnosis. Hence, our future work will be focussed on additionally modelling the communication network by the new type of timed I/O-automata and the consideration of the network properties at the off-board diagnosis.

## ACKNOWLEDGMENTS

This work was supported by the *Deutsche Forschungsgemeinschaft* under grant LU 462/22-2.

## REFERENCES

(Al-Salami *et al.*, 2008) I. M. Al-Salami, S. X. Ding, and P. Zhang. Fault detection system design for networked control system with stochastically varying transmission delays. In *Proceedings of the 17th World Congress The International Federation of Automatic Control*, Seoul, 2008.

(Alur and Dill, 1994) R. Alur and D. Dill. A theory of timed automata. *Theoretical Computer Science*, volume 126:183–235, 1994.

(Blanke *et al.*, 2006) M. Blanke, M. Kinnaert, J. Lunze, and M. Staroswiecki. *Diagnosis and Fault-Tolerant Control*. Springer Verlag, Heidelberg, 2006.

(Bouyer *et al.*, 2005) P. Bouyer, F. Chevalier, and D. D'Souza. Fault diagnosis using timed automata. In V. Sassone, editor, *Foundations of Software Science and Computational Structures*, pages 219–233. Springer, Berlin, 2005.

(Fritsch *et al.*, 2006) C. Fritsch, J. Lunze, M. Schwaiger, and V. Krebs. Remote diagnosis of discrete-event systems with on-board and off-board components. In *Proceedings of the 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, 2006.

(Kaynar *et al.*, 2003) D. K. Kaynar, N. Lynch, R. Segala, and F. Vaandrager. Timed I/O automata: A mathematical framework for modeling and analyzing real-time systems. In *Proceedings of the 24th IEEE International Real-Time Systems Symposium*, Cancun, 2003.

(Schlage and Lunze, 2009) T. Schlage and J. Lunze. Data communication reduction in remote diagnosis of discrete-event systems. In *Proceedings of 7th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Barcelona, 2009.

(Schumann and Pencolé, 2006) A. Schumann and Y. Pencolé. Efficient on-line failure identification for discrete-event systems. In *Proceedings of the 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, 2006.

(Supavatanakul, 2004) P. Supavatanakul. *Modelling and Diagnosis of Timed Discrete-Event Systems*. Shaker, Aachen, 2004.

(Tripakis, 2002) S. Tripakis. Fault diagnosis for timed automata. In W. Damm and E.-R. Olderog, editors, *Formal Techniques in Real-Time and Fault-Tolerant Systems*, pages 205–221. Springer, Berlin, 2002.

(Zhang and Ding, 2006) P. Zhang and S.X. Ding. Fault detection of networked control systems with limited communication. In *Proceedings of the 6th IFAC Symposium on Fault Detection, Supervision and Safety of Technical Processes*, Beijing, 2006.

(Zhang *et al.*, 2004) P. Zhang, S.X. Ding, P.M. Frank, and M. Sader. Fault detection of networked control systems with missing measurements. In *Proceedings of the 5th Asian Control Conference*, Melbourne, 2004.