Online Diagnosis of Systems with Rectangular Hybrid Automata Models

Haithem Derbel, Hassane Alla, Nejib Ben Hadj-Alouane, Moez Yeddes

To cite this version:


HAL Id: hal-00377226
https://hal.archives-ouvertes.fr/hal-00377226
Submitted on 21 Apr 2011

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Online Diagnosis of Systems with Rectangular Hybrid Automata Models

Haithem Derbel *, Hassane Alla *, Nejib Ben Hadj-Alouane **, Moez Yeddes **

* Department of Automatic Control, Gipsa-lab, France.
** The OASIS Laboratory and, National School of Information Sciences (ENSI), Tunisia.

Abstract: We propose an online diagnosis approach for a class of hybrid systems. The normal and the faulty behaviors of the system are modeled with rectangular hybrid automata. Our approach is based on the use of a diagnosis procedure which performs, online, an estimation of the system states, within a given time window, and based on the current record of observable timed events. Each new estimation can be triggered either, by a new event observation, or simply by the elapse of time. We give examples to illustrate the use of our hybrid systems diagnosis approach.

Keywords: Diagnosis, Online, Rectangular Hybrid Automata, Symbolic Analysis.

1. INTRODUCTION

Fault diagnosis is an important task for the design and development of man-made systems such as embedded systems, industrial process control systems, ... etc. This importance is due to the crucial role diagnosis plays in protecting human life, and increasing the efficiency and productivity of these systems. The diagnosis task consists of the detection of anomalous system behaviors, followed by the isolation and the identification of the causes behind these faults.

The diagnosis problem has been extensively studied in the automatic control and discrete event systems (DES) literatures, during the last two decades. Several model-based diagnosis techniques has been proposed within the context of continuous, discrete, and hybrid systems (Lin and Wonham, 1994; Sampath et al., 1996; McIlraith et al., 2000; Derbel et al., 2006; Tripakis, 2002; Bhowal et al., 2007). In the most widely used DES diagnosis approach (Sampath et al., 1995, 1996), a finite state machine called diagnoser is compiled offline, based on another finite state model, capturing the normal and the faulty behaviors of the system. In (Zad et al., 1999), a timed model of the system is used, capturing time with a tick event. In (Derbel et al., 2006; Tripakis, 2002) dense time extensions have been proposed, using timed automata (Alur, 1994) as a system model. In (Tripakis, 2002), the author proposes an online state estimator, and a good/faulty partition of the state space, to perform the system diagnosis.

Case studies of hybrid systems diagnosis have been developed, using system models capturing both the discrete events and the continuous variables. Most recently, in (Bhowal et al., 2007), a discrete time hybrid system model is used for constructing, offline, a diagnoser.

This paper is inspired from our contribution to the diagnosis of dense time systems (Derbel et al., 2006). It generalizes the online timed systems diagnosis approach in (Tripakis, 2002), to provide an effective online diagnosis for hybrid systems. The fact that our diagnosis approach works online helps us overcome many of the issues related to the undecidability and intractability of hybrid systems (Henzinger et al., 1998).

Our approach is based on Rectangular Hybrid Automata (RHA) (Henzinger et al., 1998) models, under the nonzeno condition. RHA represent an interesting class of hybrid systems, that can very closely capture the dynamics of real systems. They allow the description of arbitrary closed approximation of continuous behaviors, using lower and upper bounds on derivatives. Furthermore, the symbolic analysis of these automata can be easily performed using polyhedron structures (Alur et al., 1995).

The underlying idea of our approach consists of using a diagnosis procedure, which performs an online state estimation. The used system model captures both normal and faulty system behaviors. Failures are modeled as discrete transitions on unobservable events. The diagnosis procedure uses a symbolic analysis approach to perform online estimation of the system states, based on the current record of observable timed events, and within a lookahead time window (Ben.Hadj-Alouane et al., 1994). Our online diagnosis procedure is time/event driven. In fact, performing a new state estimation is triggered either by the arrival of an observable event, or the expiration of a timer. The expiration deadline of this timer, is dynamically computed, after performing each estimation.

This paper is organized as follows. The next section provides the necessary background on rectangular hybrid
automata. Section 3 formally defines our diagnosis model for hybrid systems. Section 4 describes, in detail, our online diagnosis procedure. We conclude in section 5. Throughout this paper, we use a simple example to illustrate the various features of our online diagnosis approach for hybrid systems.

2. AN OVERVIEW OF RECTANGULAR HYBRID AUTOMATA

Let \( X = \{x_1, \ldots, x_n\} \) be a finite set of real-valued variables. We denote by \( \tilde{X} = \{\tilde{x} \mid x \in X\} \) the set of first derivatives of the variables of \( X \). A variable \( x \) is called clock. We denote by \( \sim \) an element of the operators \( \left\{\prec, \leq, =, \geq, \succ\right\} \). A rectangular inequality over \( X \), is an inequality of the form \( x \sim c \), where \( x \in X \), and \( c \in \mathbb{R}^n \). A rectangular predicate over \( X \) is a conjunction of rectangular inequalities over \( X \). We denote by \( \operatorname{Rect}(X) \) the set of rectangular predicates over \( X \).

A subset of \( \mathbb{R}^n \) is called a region. For a region \( Z \) and a variable \( x_i, x \in X \), \( z(x_i) = \{v \mid v \in Z\} \). We denote by \( \llbracket \psi \rrbracket \), the region composed of the set of vectors \( v \in \mathbb{R}^n \), for which the predicate \( \psi \) is true, when each variable \( x_i \) is replaced by \( v_i \), for each \( i \in \{1, \ldots, n\} \). For a rectangular predicate \( \psi \), and a variable \( x_i \), we write \( \llbracket \psi \rrbracket(z_i) \) to denote the interval of values described by \( v_i \), for all \( v \in \llbracket \psi \rrbracket \).

A Rectangular Hybrid Automaton (RHA) (Henzinger et al., 1998), \( H \), is a tuple \((L, X, \Sigma, E, \text{init}, \text{flow}, \text{init})\), where:

- \( L \) is a finite set of locations.
- \( X \) is a finite set of real valued variables.
- \( \Sigma \) is a set of events.
- \( E \subseteq L \times \Sigma \times \operatorname{Rect}(X) \times \operatorname{Rect}(X) \times \mathbb{R}^n \times L \), is a finite set of edges. An edge \((l, \sigma, g, r, l', \delta)\) corresponds to a switch from location \( l \) to location \( l' \), on the event \( \sigma \), under the condition that \( v \in \llbracket g \rrbracket \), where the vector \( v \) corresponds to current values of the variables.
- The state space of \( H \) is uncountably infinite, we use the symbolic representation to perform the analysis over this space. The symbolic state of the automaton \( H \) is a pair \((l, z)\), where \( l \) corresponds to a location of \( L \), and \( z \) is a region over \( X \). The pair \((l, z)\) represents the set of states \( \{(l, v) \mid v \in z\} \). The continuous successor operator of a symbolic state \((l, z)\) is defined as:

\[
\text{Post}_c((l, z)) = \{((l, v'), (l, v)) \mid v \in z, \delta \in \mathbb{R}^n\}.
\]

Similarly, we define the discrete successor operator of a symbolic state \((l, z)\), over a transition \( e \in E \), defined as:

\[
\text{Post}_d((l, z), e) = \{((l', v'), (l, v)) \mid (l', v') \in E, (l, v) \in (l, z), (l, v) \in (l', v')\}.
\]

Computing discrete and continuous successors is equivalent to performing some geometrical operations on \( n \)-dimensional regions (Alur et al., 1995). We note that some tools like HyTech (Henzinger et al., 1997), and PHAVer (Frehse, 2005), implement such operations on regions, using polyhedral libraries, to perform the symbolic
3. A MODEL FOR THE DIAGNOSIS OF HYBRID SYSTEMS

To perform a reliable model-based diagnosis, we must have a complete system model, describing both normal and faulty behaviors of the system. The normal behavior of the system corresponds to the expected operating modes and trajectories, whereas the faulty behaviors correspond to deviations from the normal behaviors, due to malfunctions called failures. These malfunctions typically affect the discrete and continuous dynamics of the system. Generally, due to limited instrumentations, these malfunctions are not directly measurable, and so, their detection cannot be straightforwardly performed. In a hybrid systems framework, a failure can affect the continuous dynamics; e.g., a stuck of a valve modifying the corresponding feeding rate, or the discrete behavior; e.g., a fluid level sensor generating an alarm when it detects an overflow.

In our work, we use the RHA model as our basis for the diagnosis of hybrid systems. We assume that all continuous variables are not measurable. Furthermore, the event set $\Sigma$ is partitioned into observable and unobservable sets of events $\Sigma = \Sigma_o \cup \Sigma_uo$ (Lin and Wonham, 1988). Hence, the diagnosis system can only observes the events in $\Sigma_o$, which correspond, typically, to discrete control commands and sensor feedbacks.

The diagnosis model, we use, must fulfill the following requirements:

- The system is modeled by an RHA, $H = (L, X, \Sigma, E, \text{inv}, \text{flow}, \text{init})$.
- Failures are modeled using discrete transitions on unobservable events. We denote by $\Sigma_f \subseteq \Sigma_uo$, the set of failure events.
- The failure set $\Sigma_f$ is partitioned into $m$ disjoint failure subsets (or modes) $\Sigma_f = \{F_1, \ldots, F_m\}$.
- Failures are permanent; i.e., return to the normal behavior is not possible.
- Only single failure scenarios are considered. However, extensions of our work to deal with multiple failures scenarios can be easily developed.
- The system starts in normal mode; i.e., no failure has already happened.

To illustrate our approach, we consider the following manufacturing system example. This example is used for illustration purposes throughout the paper.

Consider the simple fluid mixing system, shown in Figure 1. Two fluids of different types are mixed into an empty container. The first fluid is introduced by opening the valve $V_1$. The second fluid, is introduced through the valve $V_2$ (the first fluid serves to dilute the second). As soon as the level of the first fluid reaches the maximal level, detected by the sensor $LS_2$, the valve $V_1$ is closed, and the second fluid is introduced until reaching the desired concentration of the mix, notified by the concentration sensor $CS$. In the next step, the mix is evacuated through the valve $V_3$.

The RHA in Fig. 2, captures the normal and the faulty behaviors of the system. The normal behavior is not possible. The failure set $\Sigma_f$ includes the possible failures, modeled by the event set $\Sigma_f = \{sc_1, sc_2, sv_1, sv_2, sv_3\}$.

For simplicity purposes, we only capture the case of two failures, affecting the dynamics of the variables $x_1$ and $x_2$, of the mixing system. The first failure, modeled by the event $sc_1$, corresponds to a stuck of the valve $V_1$ in the close position. The second failure, modeled by the event $sv_3$, corresponds to a stuck of the valve $V_3$ in the open position. We note that the transition edges on unobservable events (sc2, so3, and $\sigma_u$) are represented using dashed arrows. The observable events $ls_1$, $ls_2$, and $cs$, correspond to notifications of the sensors $LS_1$, $LS_2$, and $CS$, respectively.

We describe in this section our failure diagnosis procedure which constitutes the heart of our diagnosis approach. We assume that the system is modeled as an RHA, $H =$
(L, X, Σ, E, inv, flow, init), designed under the conditions discussed in Section 3. We assume that the system model H is nonzeno. This requirement assures the time progress of the system runs, which constitutes a necessary condition to apply our diagnosis procedure.

Before describing the diagnosis procedure, let define the needed formal background to perform that. We define the projection operator P, which erases all unobservable events from a given timed sequence, and summing the delays of the erased events. We associate to each failure mode $F_i$, $i \in \{1, \ldots, m\}$, a failure label $F_i$. We denote by $N$ the normal label, associated to normal states of the system. Let $\Phi$ denotes the set of $m + 1$ diagnosis labels $\{N, F_1, \ldots, F_m\}$. We define the label propagation operator $\otimes : \Phi \times \Sigma \rightarrow \Phi$ as follows:

$$\phi \otimes \sigma = \left\{ \begin{array}{ll} F_i, & \text{if } \phi = F_i, \\ F_i, & \text{if } \phi = N \text{ and } \sigma \in F_i, \\ \bot, & \text{if } \phi = N \text{ and } \sigma \notin X_i. \end{array} \right.$$ 

A diagnosis state $q$ is a set of triplets, $\{(l_i, z_i, \phi_i) \in L \times \Psi(X) \times \Phi, i \in \{1, \ldots, k\}\}$. Let $Q$ denotes the set of all diagnosis states. We define the function $\mathcal{A} : Q \rightarrow \Phi \cup \{\bot\}$, which evaluates a given diagnosis state $q \in Q$. The function $\mathcal{A}$ returns a failure label $F_i$, $i \in \{1, \ldots, m\}$ (respect. the normal label $N$), whether all the elements in $q$ contain the label $F_i$ (respect. $N$), and returns the symbol $\bot$, otherwise. The function $\mathcal{A}$ is formally defined as follows:

For a given state $q = \{(l_1, z_1, \phi_1), \ldots, (l_k, z_k, \phi_k)\}$, 

$$\mathcal{A}(q) = \left\{ \begin{array}{ll} \phi_i, & \text{if } q \neq \emptyset \text{ and } \phi_1 = \cdots = \phi_k = \phi, \\ \bot, & \text{otherwise}. \end{array} \right.$$ 

A diagnosis state $q$ is said to be $F_i$-certain, for $i \in \{1, \ldots, m\}$, if $\mathcal{A}(q) = F_i$.

Before applying our diagnosis procedure, we add to the system model $H$, an extra clock $y$, taking the value $0$, in all the initial states, and resets to 0 after any location switches, on observable events. The clock $y$ allows the measurement of the elapsed time, since the occurrence of the last observable event.

In the following, we give a formal definition of the time bounded unobservable reach function, denoted $UR$.

The function $UR$, illustrated in Algorithm 1, computes the set of reachable states from $q_{in}$, over executing any possible sequences of unobservable events, and the elapse of exactly $y_{max}$ t.u., since the last occurrence of an observable event. In fact, the clock $y$ will be equal to $y_{max}$, for all reached states. The failure labels are propagated during this computation using the operator $\otimes$.

We note that, due to the time divergence property provided by the nonzeno assumption, the halting condition of this algorithm is guaranteed. Hence, the clock $y$ will necessarily exceed the bound $y_{max}$, after a finite number of successor computations (especially, when considering the case of a cycle of transitions on unobservable events).

The diagnosis procedure is a passive, state estimator algorithm, run on-line with the system. The procedure takes as input a sequence of observable events, and estimates the possible states of the system. Each estimated state can be reached over a run, on a timed trace, having an identical observable projection with the timed sequence of events, generated by the system. Using the estimated states, the procedure performs the diagnosis of the system.

Algorithm 2, illustrates our diagnosis procedure.

Our diagnosis procedure operates as a state machine. The current state, denoted $q_{cur} \in Q$, captures an estimation of the current system states, reached after observing a sequence of events generated by the system. The set of estimated states are symbolically captured, using symbolic states $\{(l_i, z_i), i \in \{1, \ldots, k\}\}$. We associate to each symbolic state $(l_i, z_i)$ in $q_{cur}$, a diagnosis label $\phi_i$. The label $\phi_i$ tracks the occurred failures in runs, reaching each state of $(l_i, z_i)$. The label $\phi_i$ is equal to $F_j, j \in \{1, \ldots, m\}$, whether a failure from the set $F_j$ occurred. If the system operates in the normal mode, the label $\phi_i$ is equal to $N$.

The initial diagnosis state is denoted $(l_0, z_0, N)$. A switch of the procedure, to the next diagnosis state, is triggered by (1) the observation of an event, or (2) the expiration of a timer $t$, without observing any events.

The diagram in Fig. 3, informally describes, the different operations performed while executing the main loop of the diagnosis procedure.

As illustrated in Fig. 3, we start the main loop of the procedure, by computing an integer waiting delay $\tau$. The value $\tau$ corresponds to the smallest integer delay, after which, all the estimated states are $F_i$-faulty; i.e., a failure from the set $F_i$ has affected the system before reaching any estimated states. So, the estimated diagnosis state $q_{cur}$, after this delay, will be $F_i$-certain. A finite value

\begin{algorithm}[h]
\caption{Function $UR(q_{in} \in Q, y_{max} \in N) : Q$}
\begin{algorithmic}[1]
\State Initially, $Waiting := \{(l, Post_e(z)), \phi) \mid (l, z, \phi) \in q_{in}\}$, 
\State $Visited := Waiting$.
\Repeat
\State Get and remove an element $(l, z, \phi)$ from $Waiting$.
\ForAll{transitions $l \xrightarrow{\sigma} l'$ in $E$, such that $\sigma_{uo} \in \Sigma_{uo}$}
\State Compute $(l', z', \phi')$, such that $(l', z') := Post((l, z), l \xrightarrow{\sigma_{uo}} l')$, and $\phi' := \phi \otimes \sigma_{uo}$.
\State Add $(l', z', \phi')$ to $Visited$, if $z' \neq \emptyset$.
\State Add $(l, z', y \wedge (y < y_{max}), \phi)$ to $Waiting$, if $z' \wedge (y < y_{max}) \neq \emptyset$.
\EndFor
\Until{Waiting $= \emptyset$}.
\State $q_{out} := \{(l, z, \phi), \text{such that } (l, z, \phi) \in Visited, z := z \wedge (y = y_{max})\}$.
\State return $q_{out}$.
\end{algorithmic}
\end{algorithm}

\footnote{We add the assignment $y := 0$ to the reset condition of any transition edges, labeled with observable events.}

\footnote{The procedure does not influence the behavior of the system to be diagnosed.}
Algorithm 2 Diagnosis Procedure:

Require: Parameter $T$.
1. $q_{cur} := \{(\ell_0, \emptyset, \mathcal{N})\}$.
2. $y_{ref} := 0$.
3. loop
4. if $A(UR(q_{cur}, y_{ref} + T)) \neq \mathcal{F}_i$, for any $i \in \{1, \ldots, m\}$ then
5. $\tau := T$.
6. else
7. $\tau := \inf\{\theta \in \{1, \ldots, T\}, \text{ such that } A(UR(q, y_{ref} + \theta)) = \mathcal{F}_i, i \in \{1, \ldots, m\}\}$.
8. end if
9. Initialize the timer $t$ to 0.
10. Arm the expiration of the timer $t$ after $\tau$ t.u.
11. Wait for an event $\sigma \in \Sigma_o$, or for the expiration of the timer $t$.
12. Compute $q_{succ} := UR(q_{cur}, y_{ref} + t)$.
13. if an event $\sigma \in \Sigma_o$ is observed then
14. $q_{cur} := \{(l'_i, z'_i, \phi_i)\}$, such that $(l'_i, z'_i) = Post_g((l_i, z_i, \sigma_{t.u})), \text{ for all } (l_i, z_i, \phi_i) \in q_{succ}$.
15. $y_{ref} := 0$.
16. else
17. $q_{cur} := q_{succ}$.
18. $y_{ref} := y_{ref} + t$.
19. end if
20. if $A(q_{cur}) = \mathcal{F}_i, i \in \{1, \ldots, m\}$ then
21. Announce the occurrence of a ‘Failure $F_i$’.
22. Halt the algorithm.
23. else if $A(q_{cur}) = \mathcal{N}$ then
24. Announce that the system has ‘No failures’.
25. else
26. Announce that current state of system is an ‘Uncertain state’.
27. end if
28. end loop

Fig. 3. Diagnosis Procedure Flow-Chart.

for $\tau$ may not exist in some cases (e.g., when the system operates permanently in the normal behavior). Therefore, we check the existence of the value $\tau$, into a bounded set of discrete integer values $\{1, \ldots, T\}$, called the lookahead time window (Ben.Hadj-Alouane et al., 1994), given the parameter $T$. If the estimated diagnosis state $q_{cur}$, after the elapse of $T$ t.u., is not $F_i$-certain, $\tau$ will be equal to the value $T$. In this case, the diagnosis procedure will only update the diagnosis state $q_{cur}$, without announcing any failures.

In the next step, we arm the expiration of the timer $t$ after the elapse $\tau$ t.u., and we wait for the occurrence of an observable event, or the expiration of the timer $t$. If one of these conditions occurs, the algorithm updates the diagnosis state $q_{cur}$, w.r.t. the elapsed time $t$, and the eventual observed event $\sigma$, as described in the following:

We compute, in the first step, the time successor of the diagnosis state $q_{cur}$, denoted $q_{succ}$, using the function $UR$.

In the second step, we compute the discrete successors of the elements of $q_{succ}$, if an event $\sigma \in \Sigma_o$ have been observed. The obtained result corresponds to the current diagnosis state $q_{cur}$. If no events have been observed, and the timer $t$ expires, the current diagnosis state $q_{cur}$ corresponds to the time successor $q_{succ}$.

In the last step, we evaluate the diagnosis state $q_{cur}$, using the function $A$. As illustrated in Diagram 3, there are three possible diagnostics, regarding the result of $A(q_{cur})$:

1. $A(q_{cur})$ is $F_i$-certain, for $i \in \{1, \ldots, m\}$: we announce the occurrence of a ‘Failure $F_i$’. In this case, all the estimated system states, elements of $q_{succ}$, are reached over runs, containing a failure from the set $F_i$.
2. $A(q_{cur}) = \mathcal{N}$: we announce that the system operates in the ‘Normal’ mode. So, no failures has been occurred, before reaching any estimated states in $q_{succ}$.
3. $A(q_{cur}) = \bot$: we announce that the system has an ‘uncertain state’. In this case, the estimated states of the system contain at least a pair of states, where the first is $F_i$-faulty, $i \in \{1, \ldots, m\}$, and the second is Normal, or $F_j$-faulty, $j \in \{1, \ldots, m\}$ and $j \neq i$.

We note that our procedure may output, indefinitely, the diagnostic 'uncertain state'. In this case, our procedure cannot distinguish between two runs of the system, having identical observable behavior, the first contains a failure from $F_i$, and the second not, for a given $i \in \{1, \ldots, m\}$. In this case, the system model is said to be not diagnosable (Sampath et al., 1995). Diagnosability issues constitute the interest of our future works.

**Example 2.** To illustrate the execution of our diagnosis algorithm, let consider the Example 1. We suppose that the system generates the following traces: $\omega_1 = (ls_2, 22)(cs, 22)(so_5, 14)(ls_1, 23)(s_5, 5)(ls_2, 50)$ and $\omega_2 = (ls_2, 22)(cs, 22)(ls_1, 37)(s_5, 5)(sc_1, 10)(e, 100)$. Both traces $\omega_1$ and $\omega_2$ correspond to failure behaviors, the former contains a failure $so_3$, and the latter contains a failure $sc_1$. We note the use of the silent action $e$, to represent the time progress of the run over $\omega_2$, without observing any more events. We denote by $P(\omega_1) = (ls_2, 22)(cs, 22)(ls_1, 37)(ls_2, 55)$ and $P(\omega_2) = (ls_2, 22)(cs, 22)(ls_1, 37)(e, 115)$. The observable projections of the traces $\omega_1$ and $\omega_2$, respectively. We assume that the failure set $\Sigma_f = \{so_3, sc_1\}$ is partitioned as: $F_1 = \{so_3\}$, $F_2 = \{sc_1\}$, and $T$ equals to 30. We give in the following, some execution statements, of our diagnosis procedure, given in the Algorithm 2. We
suppose that the system generates the timed trace $P(\omega_1)$ in our first execution example:

1. Initially $q_{\text{cur}} := \{(x_1 = x_2 = x_3 = y = 0, \mathcal{N})\}$.
2. $\tau := 26$. The event $ls_2$ is observed at $t = 22$, $q_{\text{cur}} := \{(x_1 = 200 \land x_2 = x_3 = y = 0, \mathcal{N})\}$, the diagnostic ‘Normal’ is output.
3. $\tau := 30$. The event $cs$ is observed at $t = 22$, $q_{\text{cur}} := \{(x_1 = 222 \land x_2 \in [66, 80] \land x_3 = y = 0, \mathcal{N})\}$, the diagnostic ‘Normal’ is output.
4. $\tau := 30$. The timer expires at $t = 30$, $q_{\text{cur}} := \{(3, x_1 = 42 \land x_2 \in [66, 80] \land y = 30 \land x_3 = 0, \mathcal{N})\}$, $\tau := 22$, $x_3 = 30 \land x_3 = 0, \mathcal{F}_1$), the diagnostic ‘Uncertain state’ is output.
5. $\tau := 30$. The event $ls_1$ is observed at $t = 7$, $q_{\text{cur}} := \{(4, x_1 = x_2 = x_3 = y = 0, \mathcal{N})\}$, $\tau := 30$, $x_1 = 2 = x_3 = y = 0, \mathcal{F}_1$). The procedure announces a ‘Failure $F_1$’, and the algorithm halts.

We remark that, the observation of the event $ls_2$, later than the expected date in the normal behavior, allows the detection of the failure stuck open, affecting the valve $V_1$. In other cases, the system may become silent after the occurrence of a failure; i.e., the system stop generating events, as illustrated in the following example.

Let us consider now the example of the trace $\omega_2$. We suppose that the system generates the timed sequence $P(\omega_2)$. When the timed sequence $(ls_2, 22)(cs, 22)(ls_1, 37)$, is observed, the diagnosis procedure repeats the statements 1 to 6, performed in the previous example. After that, the system will stop generating events, while the time continue to elapse for 115 t.u. We give in the following, the continuation of the diagnosis procedure execution.

- $\tau := 30$. The timer expires at $t = 30$, $q_{\text{cur}} := \{(5, x_1 \in [110, 200] \land y = 60 \land x_2 = x_3 = 0, \mathcal{F}_1)\}$, $\tau := 30$, $x_1 \in [0, 200] \land y = 60 \land x_2 = x_3 = 0, \mathcal{F}_2$), the diagnostic ‘Uncertain state’ is output.
- $\tau := 30$. The timer expires at $t = 30$, $q_{\text{cur}} := \{(5, x_1 \in [170, 200] \land y = 60 \land x_2 = 0 \land x_3 = 0, \mathcal{F}_1)\}$, $\tau := 16$, $x_1 \in [0, 200] \land y = 76 \land x_2 = x_3 = 0, \mathcal{F}_2$). The procedure announces the occurrence of a ‘Failure $F_2$’, and the algorithm halts.

5. CONCLUSION AND FUTURE WORKS

In this paper, we presented an online approach for the diagnosis of systems modeled with Rectangular Hybrid Automata. This approach is based on the use of a diagnosis procedure, performing an online estimation of the system states, within a lookahead time window, given a record of observable timed events, generated by the system. The diagnosis procedure is driven by event observations, or time elapsing. Our approach is applicable under the nonzenoness condition on the used system model.

REFERENCES