Limited-Time Lookahead Diagnosability of Rectangular Hybrid Automata

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

Limited-Time Lookahead Diagnosability of Rectangular Hybrid Automata

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

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

Abstract: This paper investigates the diagnosability of Rectangular Hybrid Automata used for modeling a class of hybrid systems. First, a generalized definition of Limited-Time Lookahead diagnosability of timed languages, for multiple failure modes, is proposed. Then, we provide a systematic approach, for checking the LTLa diagnosability of systems modeled with Rectangular Hybrid Automata, and verifying some realistic assumptions. A practical example is considered throughout the paper for illustration purposes.

Keywords: Diagnosability , Limited-Time Lookahead, Rectangular Hybrid Automata, Reachability 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 reliability and robustness 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 fault diagnosis-related problems have been extensively studied within the context of continuous (), discrete (Lin and Wonham, 1994; Sampath et al., 1996), and hybrid systems (Bhowal et al., 2007; Zhao et al., 2005; McIlraith et al., 2000), during the last two decades. Among the important problems, often addressed in the model-based diagnosis literature is the verification of the system diagnosability. A system is said to be *diagnosable* if it is possible to detect every unobservable failure event, within a finite delay from its occurrence, given a record of observable events it generates.

In the model-based diagnosis context, verifying the diagnosability of a system provides modeling its correct and faulty behaviors, using good/faulty partition of the system states (Zad et al., 1998), or observable/unobservable partition of the system events (Lin and Wonham, 1988; Sampath et al., 1996). The verification consists in identifying whether the occurrence of a failure can be inferred within a finite delay of its occurrence, given a trajectory of observable events following it. In (Sampath et al., 1995), a diagnosability verification approach for untimed Discrete Event Systems (DESs), modeled with Finite State Machines (FSMs), has been developed. In fact, it has been shown that a system is diagnosable, if the corresponding FSM diagnoser does not contain F_i-indeterminate cycles. Extensions of this approach to timed and hybrid models have been proposed. In (Tripakis, 2002), a dense-time extension of the untimed diagnosability definition has been proposed, called Δ-diagnosability. A system, modeled with a timed automaton(TA) (Alur, 1994), is Δ-diagnosable if it is possible to detect a failure after a time delay bounded by Δ since the fault has occurred. The diagnosability verification for TA models was reduced to a zeno cycles detection problem. In (Derbel et al., 2006), we proposed an approach to check the diagnosability of a class of TAs, based on the detection of F_i-indeterminate cycles, which provides less computation efforts.

Diagnosability of hybrid systems was considered in (Fourlas et al., 2002), where a notion of diagnosability is proposed for I/O hybrid automata. Case studies of hybrid systems diagnosis and diagnosability have been developed for discrete time hybrid system (DTHS) modeling framework in (Bhowal et al., 2007). To our knowledge, only few results studying the diagnosability of dense-time hybrid models, have been developed. In fact, undecidability problems related to hybrid system models, prevent the development of such works. Indeed, considering simple classes of hybrid system models, can contribute to overcome such difficulties.

In this context, we proposed, in (Derbel et al., 2009), an online diagnosis approach for simple class of hybrid systems, modeled with Rectangular Hybrid Automata ()(RHAs) (Henzinger et al., 1998a). Despite the simplicity of this formalism, the class of RHAs represents an interesting class of hybrid systems, that can very closely capture the dynamics of real systems. They allow for the description of arbitrary closed approximation of continuous behaviors, using lower and upper bounds on derivatives(e.g. $\dot{x} \in [2, 5]$). Unfortunately, RHA model

presents many decidability difficulties, as the undecidability of the reachability problem (Henzinger et al., 1998a). Indeed, studying the diagnosability of RHAs represents an interesting and challenging task, and will correspond to the main topic of this paper.

In this paper, following our previous contribution (Derbel et al., 2009), we focus our attention to the problem of diagnosability for hybrid systems modeled with RHAs. We discuss the notion of diagnosability for timed languages. In fact, we generalize the definition of Δ-diagnosability, defined in (Tripakis, 2002) for timed models, to deal with multiple failure modes. In fact, we will adopt the term *Limited-Time Lookahead*(LTLa) (Ben.Hadj-Alouane et al., 1994) diagnosability rather than Δ-diagnosability, throughout this paper. We consider, from our point of view, that this term is more appropriate to designate this notion. The LTLa diagnosability is defined w.r.t. a 'failure/delay' mapping function θ, specifying that each failure in F_i, must be detected at most in $\theta(i)$ t.u., after its occurrence.

Our main contribution consists of studying the LTLa diagnosability of systems modeled with RHAs. In fact, we propose a systematic approach for checking the LTLa diagnosability of timed languages, accepted by RHAs verifying some realistic assumptions. This approach is inspired from the diagnosability checking approach for timed automata, proposed in (Tripakis, 2002). Indeed, our approach is based on the construction of an RHAs product, for each failure mode, and the application of a reachability analysis on it.

We note that the considered assumptions help us to surmount many decidability problems, related to the the verification of hybrid system models. From our point of view, studying the LTLa diagnosability, rather than the time-unbounded diagnosability, does not decrease the application abilities of our contribution. In fact, failure detection in real processes must be, generally, performed within a bounded time delay, after the failure occurrence. Considering randomly unbounded delays is not suitable for performing the diagnosis of critical systems, and is avoided in real practices.

This paper is organized as follows. The next section provides the necessary background on rectangular hybrid automata. Section 3, defines formally the notion of LTLa diagnosabilty in timed languages; we introduce a simple example to illustrate this notion. Section 4, first describes the needed assumptions for checking the LTLa diagnosability of systems modeled with RHA. Then, a detailed systematic approach for checking LTLa diagnosability, in RHA is given. We conclude in section 5.

2. BACKGROUND ON RECTANGULAR HYBRID AUTOMATA

Let Σ denotes a set of labels (also called, events). We assume that Σ is partitioned into two subsets of observable and unobservable events (Lin and Wonham, 1988) $\Sigma = \Sigma_o \cup \Sigma_{uo}$. Let $X = \{x_1, \ldots, x_n\}$, be a finite set of real-valued variables. We denote by $\dot{X} = \{\dot{x} \mid x \in X\}$ the set of first derivatives of the variables of X. A variable x is called *stop-watch*, if $\dot{x} \in \{0, 1\}$. We use, $\sim$, to denote an element of the set of operators, $\{<, \leq, =, \geq, >\}$. A *rectangular inequality* over X, is an inequality of the form, $x \sim c$, where $x \in X$, and $c \in \mathbb{Z}$ [1]. A *rectangular predicate* over X is a conjunction of rectangular inequalities over X. We denote by $Rect(X)$ the set of rectangular predicates over X. A *polyhedral inequality* over X is an inequality of the form $c_1x_1 + \cdots + c_kx_k \sim c$, where $x_1, \ldots, x_k \in X$, and $c, c_1, \ldots, c_k \in \mathbb{Z}$. A *polyhedral predicate* over X is boolean combination of polyhedral inequalities over X. We denote by $\Psi(X)$ the set of polyhedral predicates over X.

The vector $\mathbf{v} = (\ v_1, \ldots, v_n\ )$, is an element of $\mathbb{R}^n$, that captures the value, $v_i \in \mathbb{R}$, of every variable $x_i \in X$. [2] A subset of $\mathbb{R}^n$ is called a *region*. For a region z and $x_i \in X$, $z(x_i) = \{v_i | \mathbf{v} \in z\}$. We denote by $[\![\psi]\!]$, the region composed of the set of vectors $\mathbf{v} \in \mathbb{R}^n$, for which the predicate ψ is true when each x_i is replaced by its corresponding v_i, for each $i \in \{1, \ldots, n\}$. For a rectangular predicate ψ, and a variable x_i, we write $[\![\psi]\!](x_i)$ to denote the interval of values captured by v_i, for all $\mathbf{v} \in [\![\psi]\!]$.

A *Rectangular Hybrid Automaton*(RHA) (Henzinger et al., 1998b), H, is a tuple $(Q, X, \Sigma, E, inv, flow, init, M)$, where:

- Q is a finite set of locations,
- X is a finite set of real valued variables,
- Σ is a set of events,
- $E \subseteq Q \times \Sigma \times Rect(X) \times Rect(X) \times 2^X \times Q$, is a finite set of transition edges. A transition, (q, σ, g, r, R, q'), corresponds to a switch from location q to location q', on the event σ, under the condition that $\mathbf{v} \in [\![g]\!]$, where the vector $\mathbf{v}$ corresponds to current values of the variables. Upon the location switch, each variable $x_i \in R$ is reset, nondeterministically, to a value in the interval $[\![r]\!](x)$, the other variables, not in R, remain unchanged. We use the operator $Source(e), e \in E$, to denote the source location of the transition,
- $inv : Q \rightarrow Rect(X)$ captures the invariant conditions for the locations, The automaton H can remain in the same location as long as the value of each variable $x_i \in X$, belongs to the interval $[\![inv(q)]\!](x_i)$,
- The function $flow : Q \rightarrow Rect(\dot{X})$, assigns a flow condition to each location $q \in Q$. While the system H evolves in the location q, the first time derivative of each variable $x_i \in X$ must remain within the interval $[\![flow(q)]\!](x_i)$,
- $init \subseteq Q \times Rect(X)$, specifies the initial condition of the automaton,
- $M \subseteq Q$ corresponds to the subset of marked locations.

An *initialized Rectangular Hybrid Automata* is an RHA where: for each edge $(q, \sigma, g, r, R, q') \in E$, and for each variable $x_i \in X$: $x_i \in R$, if $[\![flow(q)]\!](x_i) \neq [\![flow(q')]\!](x_i)$.

Definition 1. (Synchronized Product). Let $H_1 = (Q_1, X_1, \Sigma_1, E_1, inv_1, flow_1, init_1, M_1)$ and $H_2 = (Q_2, X_2, \Sigma_2, E_2, inv_2, flow_2, init_2, M_2)$ denote two RHA such that $Q_1 \cap Q_2 = \emptyset$. The synchronized product, denoted by, $H_1 \otimes H_2$, is

[1] $\mathbb{Z}$ denotes the set of integers.

[2] $\mathbb{R}$ denotes the set of real, and $\mathbb{R}^n$ denotes the n-dimensional euclidean space over $\mathbb{R}$.

the RHA $H = (Q, X, \Sigma_1 \cup \Sigma_2, E, inv, flow, init, M)$, where $Q = Q_1 \times Q_2$, $X = X_1 \cup X_2$, $init = init_1 \wedge init_2$, and for each pair of locations $q_1 \in Q_1, q_2 \in Q_2$, $flow((q_1, q_2)) = flow(q_1) \wedge flow(q_2)$, and $inv((q_1, q_2)) = inv_1(q_1) \wedge inv_2(q_2)$.

A transition $e = ((q_1, q_2), \sigma, g, r, R, (q_1', q_2')) \in E$ iff either:

- $\sigma \in \Sigma_1 - \Sigma_2$, $(q_1, \sigma, g, r, R, q_1') \in E_1$, and $q_2 = q_2'$,
- $\sigma \in \Sigma_2 - \Sigma_1$, $(q_2, \sigma, g, r, R, q_2') \in E_2$, and $q_1 = q_1'$,
- $\sigma \in \Sigma_1 \cap \Sigma_2$, $(q_1, \sigma, g_1, r_1, R_1, q_1') \in E_1$, $(q_2, \sigma, g_2, r_2, R_2, q_2') \in E_2$, $g = g_1 \wedge g_2$, $r = r_1 \cup r_2$, $R = R_1 \cup R_2$.

A location (q_1, q_2) is marked; i.e., $(q_1, q_2) \in M$, if either, $q_1 \in M_1$ or $q_2 \in M_2$.

Lemma 1. The synchronized product of two initialized RHAs is also an initialized RHA.

A *state* $(q, \mathbf{v})$ of the RHA H, consists of a discrete part $q \in Q$ together with a continuous part $\mathbf{v} \in \mathbb{R}^n$ such that $\mathbf{v} \in [\![inv(q)]\!]$. The trajectories of the states of H progress in the system state space by performing one of the following transitions:

- Flow transitions: The discrete part of the state remains in the same location q, while the continuous part evolves from valuation $\mathbf{v}$ to valuation $\mathbf{v}'$, via any smooth trajectory satisfying the constraints imposed by $inv(q)$, and with derivatives of the variables remaining within the flow intervals specified by $flow(q)$. We denote this transition by $(q, \mathbf{v}) \xrightarrow{\delta} (q, \mathbf{v}')$, where $\delta \in \mathbb{R}_+$ is the time elapsed during the transition. [3]
- Discrete transitions: corresponds to a discrete and instantaneous location switch, as specified by the tuple $e = (q, \sigma, g, r, R, q') \in E$. We denote this transition by $(q, \mathbf{v}) \xrightarrow{e} (q', \mathbf{v}')$. The discrete transition is enabled only if the guard predicate is satisfied by the continuous part of the state; i.e., $\mathbf{v} \in [\![g]\!]$. The discrete part of the state changes from q to q', and the continuous part is updated according to the reset assignments.

A *run* is a finite or infinite sequence of transitions $(q_0, \mathbf{v_0}) \rightarrow (q_1, \mathbf{v_1}) \rightarrow (q_2, \mathbf{v_2}) \rightarrow \ldots$, where $(q_0, \mathbf{v_0}) \in init$. An *edge cyclic* run is a run that crosses one transition edge, at least, more than once. A state $(q_k, \mathbf{v_k})$ is said to be *reachable*, if there exists a finite run leading to it; i.e., $(q_0, \mathbf{v_0}) \rightarrow (q_1, \mathbf{v_1}) \ldots \rightarrow (q_k, \mathbf{v_k})$.

A *timed trace*, is a finite or infinite, sequence of events alternated with positive reals: $\delta_0, \sigma_1, \delta_1, \sigma_2, \delta_2, \ldots, \sigma_k, \delta_k \ldots$, where $\sigma_{k \geq 1}$ are elements of Σ, and $\delta_{k \geq 1} \in \mathbb{R}_+$, denotes the elapsed time between the occurrences of the events, σ_i, and σ_{i+1}. We define the operator $time(\omega)$, which gives the (limit of the) sum of all the delays in the timed trace ω; i.e., $time(\omega) = \sum_{i \geq 0} \delta_i$. We define the observable projection operator P, which erases all unobservable events from a given timed sequence ω, and updates the delays between the remaining events. As an example, given the timed trace $\omega = 5, \sigma_1, 23, \sigma_u, 5, \sigma_2, 12$, its corresponding observable projection is $P(\omega) = 5, \sigma_1, 28, \sigma_2, 12$. A timed trace ω is accepted by an RHA H, if there exists a run of H over the elements of ω. A *timed language* is a (finite or infinite) set of timed traces. A timed language L is said to be *accepted* by an RHA H, if every trace in L is accepted by H.

A run over an infinite timed trace ω is said to be *time-divergent*, if $time(\omega) = \infty$. An RHA is said to be *Strongly Non-Zeno*(SNZ), (**?**) if there exists a integer $d > 0$, such that, all edge-cyclic runs have durations greater than d. We give in the following, a sufficient structural condition, to guarantee the strongly non-zenoness of an RHA.

An RHA H is SNZ, if every cycle of transitions, $q_1 \xrightarrow{e_1} q_2 \xrightarrow{e_2} \ldots \xrightarrow{e_{k-1}} q_k \xrightarrow{e_k} q_1$, in H, verifies the following conditions:

- There exists a variable $x \in X$, such that $\dot{x}$ is strictly positive (or, respectively, strictly negative) in all locations of the cycle.
- There exist a pair transitions $e, e' \in \{e_1, \ldots, e_k\}$, and two integers $c, c' \in \mathbb{Z}$, with $c < c'$, such that: (1) If $\dot{x} > 0$, x is reset to c in e, and upper-bounded by c'; i.e., $x \leq c'$, in e'. (2) If $\dot{x} < 0$, x is reset to c' in e, and lower-bounded by c; i.e., $x \leq c$, in the guard condition of e'.

Lemma 2. The synchronized product of two SNZ automata is a SNZ automata.

Proof: Each cycle in the product $H_1 \otimes H_2$ corresponds to a cycle in H_1 or H_2, or to the synchronization of a pair of cycles, the first in H_1, and and the second in H_2. Hence, an edge-cyclic run in the product automaton have a duration greater or equal to the minimal duration of edge-cyclic runs in H_1 or/and H_2. Since, every edge-cyclic run in H_1 (respect. in H_2) is greater than $d_1 > 0$ (respect. $d_2 > 0$). We conclude that every edge-cyclic run in $H_1 \otimes H_2$ is greater than the minimum between d_1 and d_2.

Since the state space of an RHA is uncountably infinite, we use the symbolic representation to perform analysis over its space. The *symbolic state* of the automaton H is a pair (q, z), where q corresponds to a location of Q and z is a region over X. The pair (q, z) represents the set of states $\{(q, \mathbf{v}) | \mathbf{v}) \in z\}$. The continuous successor operator of a symbolic state (q, z) is defined as: $Post_c((q, z)) = \{(q, \mathbf{v}') | (q, \mathbf{v}) \xrightarrow{\delta} (q, \mathbf{v}'), \mathbf{v} \in z, \delta \in \mathbb{R}_+\}$. Similarly, we define the discrete successor operator of a symbolic state (q, z), over a transition $e \in E$, denoted by $Post_d((q, z), e)$ as: $Post_d((q, z), e) = \{(q', \mathbf{v}') | (q, \mathbf{v}) \xrightarrow{e} (q', \mathbf{v}'), \mathbf{v} \in z\}$. We define the successor operator $Post$, over an edge $e \in E$, as the composition of the discrete and continuous successor operators; i.e., $Post((q, z), e) = Post_c \circ Post_d((q, z), e)$.

Computing discrete and continuous successors is equivalent to performing some geometrical operations on n-dimensional polyhedra (Alur et al., 1995). The polyhedral part, z', of the continuous successor $(q, z') = Post_c((q, z))$ is equal to the intersection of the corresponding invariant, $inv(q)$, with the continuous extension [4] of z. The discrete successor $(q', z') = Post_d((q, z), e)$, where q' is the desti-

[3] $\mathbb{R}_+$ denotes the set of positive real numbers

[4] defined as the set of values reached from each elements of z, by applying the flow condition.

nation location of the transition e, and z' is computed as follows: (1) compute z_1 by intersecting z with the guard g of the considered transition; (2) compute z_2 by applying on z_1 the update function of the transition e; (3) finally get z' by intersecting z_2 with $inv(q')$, the invariant of the location q'. We note that some tools like HyTech (Henzinger et al., 1997), and PHAVer (Frehse, 2005), implement such operations using polyhedral libraries, to analyze a more general class of automata called *linear hybrid automata*.

Given an RHA H, and two states $(q, \mathbf{v})$, and $(q', \mathbf{v}')$, the *reachability problem* consists of verifying whether there exists a run of H which starts from $(q, \mathbf{v})$ and ends at $(q', \mathbf{v}')$. The *forward reachability analysis* uses an iterative symbolic computation, to check whether a state $(q', \mathbf{v}')$ is reachable from an initial state. The following function performs forward reachability analysis. It returns 'YES', when there exists a reachable state, having as discrete part a marked location, starting from an initial state in (q_0, z_0), and returns 'NO', otherwise. The pair (q_0, z_0) corresponds to the initial symbolic state, which captures all initial states of H.

Algorithm 1 Symbolic Reachability Analysis Function

```
1: function Reachable(H):{'YES','NO'}
2: Wait := {(q0, z0)}; Visited := ∅
3: while Wait ≠ ∅ do
4:    Get and remove (q, z) from Wait
5:    if q ∈ M then
6:       return 'YES'.
7:    else if z ⊈ z_k, ∀(q, z_k) ∈ Visited then
8:       Add (q, z) to Visited
9:       for all e ∈ E such that q = Source(e) do
10:         (q', z') = Post((q, z), e)
11:         Add (q', z') to Wait, if z' ≠ ∅.
12:      end for
13:   end if
14: end while
15: return 'NO'.
16: end function
```

It has been shown that this algorithm may not terminate, since the symbolic forward reachability analysis of RHA is only semi-decidable (Henzinger et al., 1998a). However, some decidability results of forward reachability analysis, have been established for subclasses of RHA, like the initialized RHA (Henzinger et al., 1998a). Indeed, the function described in algorithm 1 will alawys halt after a bounded number of iterations.

3. LIMITED-TIME LOOKAHEAD DIAGNOSABILITY OF TIMED LANGUAGES

We give in this section a definition of the Limited-Time Lookahead (LTLa) diagnosability for general timed languages. We suppose that the considered languages are time-divergent. We illustrate the intuition behind our definition using an example of timed language accepted by an RHA.

Our definition of diagnosability deals with multiple failure sets. In fact, we assume that all failure events are unobservable, and the failures set $\Sigma_f \subseteq \Sigma_{uo}$ is partitioned into $m > 0$ disjoint failure subsets (or modes) $\Sigma_f = \{F_1, \ldots, F_m\}$. Moreover, all failures are assumed to be permanent; i.e., no return to the normal behavior is possible. In our work, multiple failure scenarios are not considered; i.e., only one failure mode can affect the system behavior at once. However, extensions dealing with multiple failure can be developed in future works. We suppose that the initial states are failure safe (also called, *normal*).

We define the failure/time mapping function $\theta : \{1, \ldots, m\} \rightarrow \mathbb{N}$ associating with each failure set $F_i, i \in \{1, \ldots, m\}$, a positive number, $\theta(i)$, denoted also by θ_i.

Let L denotes a timed language. We associate with each failure mode $F_i, i \in \{1, \ldots, m\}$, a sublanguage L_i of L. Each trace of L_i contains at least one failure from the set F_i. Let $L_i^{\theta_i}$, $i \in \{1, \ldots, m\}$, denotes a sublanguage of L_i, where in each trace of $L_i^{\theta_i}$, at least θ_i t.u. have been elapsed, since the first occurrence of a failure from F_i. In fact, given a timed trace ω in $L_i^{\theta_i}$, $\omega = \delta_0, \sigma_1, \delta_1, \sigma_2, \ldots, \sigma_n, \delta_n \ldots$, we have $\sum_{k \geq j} \delta_k \geq \theta_i$, where σ_j, $j > 0$, denotes the first occurrence of F_i in ω. We denote by $L_0 \subseteq L$, the subset of traces of L, containing no failure events.

The following is a formal definition of the LTLa diagnosability, for a partition of m failure modes:

Definition 2. (LTLa diagnosability). Given a failure/time mapping function θ, a timed language L is said to be *LTLa diagnosable* w.r.t. θ iff:

$\forall i \in \{1, \ldots, m\}$, $\forall \omega \in L_i^{\theta_i}$, $\forall \omega' \in L$:

$$P(\omega) = P(\omega') \Longrightarrow \omega' \in L_i$$

According to definition 2, a timed language is LTLa diagnosable if, each pair of timed traces, the first containing a failure from F_i, $i \in \{1, \ldots, m\}$, and the second not, must have different observable projections, within θ_i t.u. of the failure occurrence.

In fact, the LTLa diagnosability of a timed language guarantees the detection of all failures $F_i, i \in \{1, \ldots, m\}$, at latest, after θ_i t.u. from its occurrence, given the observable behavior of the system.

The interval $[0, \theta_i]$ constitutes a limited-time lookahead window, in which we compare the observable behaviors of the system, after the occurrence of a failure F_i. Each system behavior affected by a failure from F_i, must be distinguished from other behaviors within this time window.

The non LTLa diagnosability of a timed language does not imply its absolute non diagnosability. In fact, it can be LTLa diagnosable w.r.t. larger values of θ. However, we cannot decide the existence or not of such values.

To illustrate the LTLa diagnosability, let us consider the following example, consisting of a simple fluid heating system. We will use this example for illustration purposes throughout the paper.

Example 1. Consider the simple fluid heating system, shown in Figure 1.

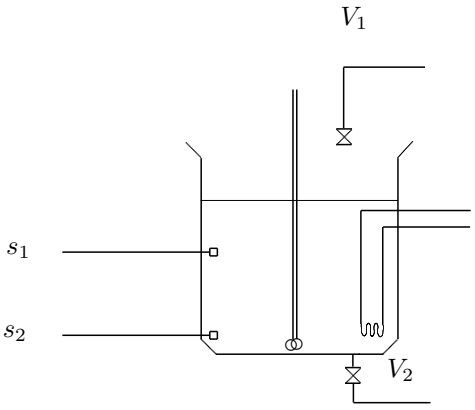

Fig. 1. *A Simple Fluid Heating System*

The fluid to be heated, is introduced through the valve V_1. As soon as the fluid level, measured by the variable x, reaches the maximal level, $x = 500$, the sensor S_1 sends a notification to the controller, thereby, closing the valve V_1. Then, the fluid is heated for 40 t.u.; i.e., until the timer t reaches the value 40. In the next step, the fluid is evacuated through the valve V_2; i.e., until the level variable, x, reaches the value 0. Following this, the sensor S_2 sends a notification to the controller to close the valve V_2; and, the heating system proceeds to the next cycle.

The RHA H, illustrated in Fig. 2, captures the normal and the faulty behaviors of the system. For simplicity purposes, only important events are shown in the model. Moreover, we suppose that only one failure can affect the system, represented by the event *leak*, which corresponds to a leakage in the fluid tank. Due to the viscosity of the fluid, the occurrence of a leakage affects its input and output rating, during the phases of filling, heating, and evacuation. We note that the transitions on unobservable events (*leak*, and u) are represented using dashed arrows. The events s_1, s_2 are observable and correspond, respectively, to sensor notifications S_1, and S_2.

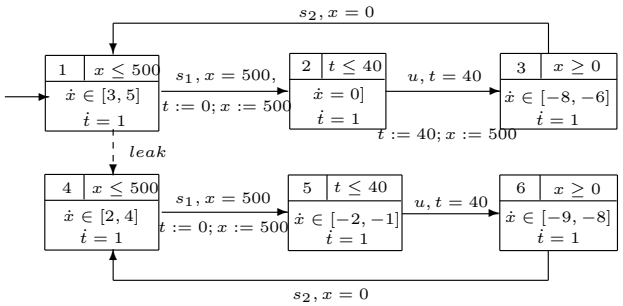

Fig. 2. *An RHA Model of the Simple Fluid Heating System*

In this example, we have one failure set $\Sigma_f = \{F_1\}, F_1 = \{leak\}$. In what follows, we shall consider two values for θ_1, to illustrate the LTLa diagnosability of the timed language, accepted by H.

In the first case, we take θ_1 equal to 150. We consider a pair of timed traces, from timed language accepted by H, denoted $\omega_1 = 125, s_1, 40, u, 70, s_2, 143, s_1, 40, u, 10$ and $\omega_2 = 125, s_1, 40, u, 70, s_2, 10, leak, 133, s_1, 40, u, 10$. It is obvious that $P(\omega_1) = P(\omega_2) = 125, s_1, 110, s_2, 143, s_1, 50$ and $\omega_2 \in L_1^{\theta_1}$. Since the timed trace ω_1 does not contain any failures from F_1, we conclude that the language L is not LTLa diagnosable w.r.t. the given θ.

Let us now take $\theta_1 = 300$. Intuitively, we can detect the occurrence of the leakage within 300 t.u.. In fact, the presence of the leakage affects the quantity of fluid contained in the tank. This quantity will be less than the expected quantity in the normal operating mode, and thus, the event s_2 will occur later. Indeed, by analyzing the occurrence time of the event s_2, we are able to distinguish between the normal and the faulty behaviors. However, the event s_2 may not be observed within 150 t.u. of the failure occurrence, keeping the observable projections, of the normal and faulty behaviors, non distinguishable.

Intuitively, we can conclude that timed language accepted by H is LTLa diagnosable for $\theta_1 = 300$, and is not, for $\theta_1 = 150$. In the next section, we shall develop the systematic approach that will enable us to prove the LTLa diagnosability of this system for $\theta = 300$.

4. CHECKING THE LTLA DIAGNOSABILITY OF RHA

We propose in the following a systematic approach for checking the LTLa diagnosability of the timed languages accepted by RHA, under some assumptions.

We start by characterizing the different assumptions we make on the system model, on which we apply our LTLa diagnosability checking approach. Then, we illustrate the different steps to check the LTLa diagnosability of the timed language accepted by an RHA. Finally, we present the key theorem linking the LTLa diagnosability of RHA and our proposed approach for checking it.

We assume that both normal and faulty behaviors of our system are captured by a given RHA H. Let L denotes the timed language accepted by H. We denote by H_0, the restricted version of the system model, capturing only system's normal behavior. It can be obtained from H, by simply removing all the transitions labeled with events in Σ_f (failure events).

We define the location/failure mapping function $\varphi : Q \mapsto \{0, \ldots, m\}$ as following: to each location q of H, φ associates to it, either $i \in \{1, \ldots, m\}$, if there exists a run of H reaching q, and containing a failure from F_i, otherwise, $\varphi(q) = 0$. However, some locations may be reached by runs containing different mode failures. Indeed, some structural transformations on the system model are needed, to guarantee the existence of exactly one value, for each location of H. This transformation consists in duplicating some transition edges and locations, to separate transition sequences, containing failures of different modes.

We propose to check the LTLa diagnosability of the timed language accepted by an RHA H, and verifying the following assumptions:

A_1: H is SNZ.
A_2: The normal behavior automaton, resulting from he transformation of H, described above, is an initialized RHA.

Our LTLa diagnosability checking approach works by exploring all the pairs of timed traces ω and ω', where $\omega \in L_i^{\theta_i}$, $\omega' \in L - L_i$, and $P(\omega) = P(\omega')$. The existence of such pairs of traces implies the non LTLa diagnosability of L, w.r.t. to the failure/time map θ.

Our approach consists of four steps. In the first step, we construct, for each failure set $F_i, i \in \{1, \ldots, m\}$, a pair of RHAs, denoted H_i and $H_{\bar{i}}$, corresponding respectively, to the system behaviors containing, respectively, not containing, failures from F_i. In the second step, we build, for each $i \in \{1, \ldots, m\}$, the product of H_i and $H_{\bar{i}}$, synchronized on observable events. Then, each obtained RHA is composed with a simple RHA, denoted TB. Finally, a reachability analysis of the resulting RHAs, is performed.

Our approach consists of three steps. In the first step, we build, for each failure set $F_i, i \in \{1, \ldots, m\}$, a pair of RHAs, denoted H_i and $H_{\bar{i}}$, corresponding respectively, to system behavior containing, respectively, not containing, failures from F_i.

The RHA H_i and $H_{\bar{i}}$ are obtained by performing some structural transformations on copies of the system model H, as follows. The RHA $H_i = (Q_i, X_i, \Sigma_i, E_i, inv_i, flow_i, init_i, M_i)$ $i \in \{1, \ldots, m\}$ is a copy of H, on which the following modifications are applied:

(1) Rename each variable x_k to x_k^i.
(2) Rename each location q_k to q_k^i.
(3) Remove all events of $\Sigma_f - F_i$ from Σ_i, and all the corresponding transitions.
(4) Rename each event σ_k in $\Sigma_{uo} - \Sigma_f$ to σ_k^i.
(5) Update, correspondingly to the above modifications, all the names of all the variables, events, and locations used in all the edges, invariance, and flow conditions.
(6) Add a stop-watch y^i as follows: (1) y^i is initialized to 0 in the initial condition; (2) For each location $q \in Q_i$, we add the condition $\dot{y}^i = 1$ to $flow_i(q)$, if $\varphi(q) = i$, otherwise, we add $\dot{y}^i = 0$.

The RHA $H_{\bar{i}} = (Q_{\bar{i}}, X_{\bar{i}}, \Sigma_{\bar{i}}, E_{\bar{i}}, inv_{\bar{i}}, flow_{\bar{i}}, init_{\bar{i}}, M_{\bar{i}})$, $i \in \{1, \ldots, m\}$ is a copy of H, on which the following transformations are applied:

(1) Remove all events of F_i from $\Sigma_{\bar{i}}$, and all the the corresponding transitions.
(2) For all $j \in \{1, \ldots, m\}$, $j \neq i$, we add a stop-watch y^j as follows: (1) y^j is initialized to 0 in the initial condition; (2) For each location $q \in Q_{\bar{i}}$, we add the condition $\dot{y}^j = 1$ to $flow_{\bar{i}}(q)$, if $\varphi(q) = j$, otherwise, we add $\dot{y}^j = 0$.

We note that each stop-watch y^k, $k \in \{1, \ldots, m\}$, is activated when a location mapped to a failure mode F_k ($\varphi(q) = k$) is reached, and will not be stopped. Thus, each stop-watch y^k measures the elapsed time, since the first occurrence of a failure from F_k

Lemma 3. We denote by $L(H_i)$ and $L(H_{\bar{i}})$, the timed languages accepted by the RHAs H_i and $H_{\bar{i}}$, respectively. For $i \in \{1, \ldots, m\}$, we consider the following equivalences:

- $L(H_{\bar{i}}) = L - L_i$
- $L(H_i) = L - \bigcup_{k=1, k\neq i}^{m} L_k = L_0 \cup L_i$

We remark that each stop-watch $y^k, k \in \{1, \ldots, m\}$, is activated when a location, mapped to a failure mode F_k ($\varphi(q) = k$) is reached, and will not stop (permanent failures assumption). Thus, the stop-watch $y^k, k \in \{1, \ldots, m\}$ measures the elapsed time, since the first occurrence of a failure from F_k.

The second step of the LTLa diagnosability verification approach consists of building, for each $i \in \{1, \ldots, m\}$, the synchronous product, $H_{i,\bar{i}} = H_i \otimes H_{\bar{i}}$. Note that the RHA $H_{i,\bar{i}}$ synchronizes H_i and $H_{\bar{i}}$, on observable events, since $\Sigma_i \cap \Sigma_{\bar{i}} = \Sigma_o$. Hence, each timed trace accepted by $H_{i,\bar{i}}$, and which contains a failure from F_k, proof the existence of a pair of traces in L_k and $L_{\overline{k}}$, with identical observable projections.

Lemma 4. For a given $i \in \{1, \ldots, m\}$, the following statements are equivalent:

(1) There exists a pair of runs in H_i and $H_{\bar{i}}$, over timed traces ω_1 and ω_2, respectively, and reaching states (q_1, v_1) and (q_2, v_2), such that $P(\omega_1) = P(\omega_2)$.
(2) There exists a run of $H_{i,\bar{i}}$, over a timed trace ω, and reaching the state $((q_1, q_2), [v_1 \ v_2]^T)$.

Lemma 5. For a given $i \in \{1, \ldots, m\}$, the following statements are equivalent:

(1) There exists a run of $H_{i,\bar{i}}$, over a timed trace ω, and reaching the state $((q_1, q_2), [v_1 \ v_2]^T)$, such that ω contains a failure from F_k.
(2) There exists a pair of runs of L_k and $L - L_k$, $k \in \{1, \ldots, m\}$, respectively, over timed traces ω_1 and ω_2, and reaching states (q_1, v_1) and (q_2, v_2), such that $P(\omega_1) = P(\omega_2)$.

Proof:

(1) $\Rightarrow$ (2) : We suppose that there exists a run of $H_{i,\bar{i}}$, over a timed trace ω, and reaching the state $((q_1, q_2), [v_1 \ v_2]^T)$, such that ω contains a failure from F_k. By Lemma 4, there exists a pair of runs of H_i and $H_{\bar{i}}$, respectively, over timed traces ω_1 and ω_2, and reaching states (q_1, v_1) and (q_2, v_2), such that $P(\omega_1) = P(\omega_2)$. By Lemma 3, $\omega_1 \in L_0 \cup L_i$, and $\omega_2 \in L - L_i$.

Since, ω contains a failure from F_k, $k \in \{1, \ldots, m\}$, then, $v(y^k) > 0$. According to the value of k, we have either, $v_1(y^k) > 0$, or $v_2(y^k) > 0$. Indeed, one of the traces, ω_1 or ω_2, contains a failure from F_k. Two cases can be distinguished:

- $i = k : \omega_1 \in (L_0 \cup L_i) \cap L_i$ and $\omega_2 \in L - L_i$. Then, $\omega_1 \in L_i$, and $\omega_2 \in L - L_i$.

- $i \neq k : \omega_1 \in L_0 \cup L_i$, and $\omega_2 \in L - L_i$. Since $L_0 \subset L - L_k$ and $L_i \subset L - L_k$, then, $L_0 \cup L_i \subset L - L_k$, and thus, $\omega_1 \in L - L_k$.

  ω_2 contains a failure from F_k, then, $\omega_2 \in (L - L_i) \cap L_k$. However, $(L - L_i) \cap L_k = (L_k - L_i) \cap L = L_k$, since $L_i \cap L_k = \emptyset$, and $L_k \subset L$.

  In conclusion, $\omega_1 \in L - L_k$, and $\omega_2 \in L_k$.

(2) $\Rightarrow$ (1) : Immediate.

Lemma 6. Let H_0^i (respectively $H_0^{\bar{i}}$) denotes a restriction of the RHA H_i (respectively $H_{\bar{i}}$), obtained by removing all transitions on failure events, then:

- $H_i^0 \otimes H_{\bar{i}}^0$ is identical to the normal behavior version of $H_{i,\bar{i}}$, obtained by removing from all transitions on failure events.
- $H_i^0 \otimes H_{\bar{i}}^0$ is an initialized RHA (by assumption A_2 and Lemma 1).

The third step of our approach consists of constructing the product of each RHA $H_{i,\bar{i}}$, with the following two location automaton, $TB = (\{q_{\overline{M}}, q_M\}, \{y^1, \ldots, y^m\}, \{\mu\}, E, inv, flow, init, \{q_M\})$ shown in Fig. 3. Let $\widehat{H}_{i,\bar{i}} = (Q_{i,\bar{i}}, X_{i,\bar{i}}, \Sigma_{i,\bar{i}}, E_{i,\bar{i}}, inv_{i,\bar{i}}, flow_{i,\bar{i}}, init_{i,\bar{i}}, M_{i,\bar{i}})$, denotes the RHA resulting from this product. In fact, in $\widehat{H}_{i,\bar{i}}$, the time progress is blocked on the runs containing failures from $F_k, k \in \{1, \ldots, m\}$, after the expiration of the lookahead window; i.e., θ_k t.u.. Indeed, a run, in $\widehat{H}_{i,\bar{i}}$, containing a failure $F_k, k \in \{1, \ldots, m\}$, will perform a discrete jump to a marked location, when y^k reaches θ_k, from where, the invariant is automatically violated, leading to the blocking of time.

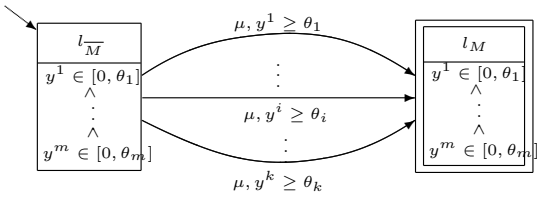

Fig. 3. *Time Bounder Automaton TB*

The final step of our approach for checking LTLa diagnosability, consists of determining whether there exists a run, reaching a marked location, in the product automaton $\widehat{H}_{i,\bar{i}}$, for each $i \in \{1, \ldots, m\}$. For this, we make use of our forwards reachability function, illustrated in Section 2. The halting of this function in this case, is guaranteed by the following theorem.

Theorem 1. For all $i \in \{1, \ldots, m\}$, the forward reachability analysis of the RHA $\widehat{H}_{i,\bar{i}}$, using Algorithm 1, is decidable.

Proof: To proof the decidability of the algorithm, we split the execution of the algorithm into two steps: (1) In the first step, the algorithm will not explore the successors of symbolic states, reached on edges labeled with failure events(from Σ_f). Hence, each symbolic state, reached over a transition on a failure event, will be put into an intermediate set, called *Pending*, and will not be added to the *Wait* set. (2) In the second step, we initialize the set *Wait* by *Pending*, and we execute the algorithm to explore the successors of symbolic states in *Pending*, not explored during the first step. It is easy to establish that performing these two steps is equivalent to executing the original version of the algorithm.

The termination of the first step is guaranteed. In fact, successors of symbolic states, reached over transitions on failure events, will not be explored. Thus, the algorithm will only perform the analysis of the restriction $H_i^0 \otimes H_{\bar{i}}^0 \otimes TB$ of $\widehat{H}_{i,\bar{i}}$. Since $H_i^0 \otimes H_{\bar{i}}^0$ is an initialized RHA (Lemma 6), the first step of the algorithm will halt (Henzinger et al., 1998a).

In the second step, we execute the algorithm, starting from symbolic states in *Pending*. Each symbolic state in *Pending* is reached through a a transition on a failure event. By Lemma 2, $\widehat{H}_{i,\bar{i}}$ is SNZ. Furthermore, for each symbolic state in *Pending*, there exists a stop-watch $y^k, k \in \{1, \ldots, m\}$, such that, y^k will be active ($\dot{y}^k = 1$), in all the locations of its successors. Moreover, when performing successors computation, the SNZ property make each stop-watch y^k to increase from 0 to θ_k, by at least d t.u. for each edges cyclic run. Consequently, the the time upper bound θ_k will be reached within a finite number of steps, and the time will be blocked at $y^k = \theta_k$, after reaching a marked location. In fact, the upper bounded invariance condition associated with SNZ condition, guarantees the termination of the symbolic successors computation starting from the set *Pending*, and thus, the halting of the algorithm.

The product RHAs, $H_{i,\bar{i}} \otimes TB$, $i \in \{1, \ldots, m\}$, allow us to isolate all pairs of traces in L_k and $L_{\overline{k}}$, $k \in \{1, \ldots, m\}$, with θ_k t.u. having passed after the first occurrence of a failure from F_k. In fact, the existence of a timed trace, reaching a marked location in one of these RHAs, is equivalent to the existence of a pair of traces in $L_k^{\theta_k}$ and $L_{\overline{k}}$, with identical observable projections.

Lemma 7. The following statements are equivalent:

(1) There exists a reachable marked location in $\widehat{H}_{i,\bar{i}}$, for $i \in \{1, \ldots, m\}$.

(2) There exist a pair of timed traces $\omega \in L_k^{\theta_k}$, and $\overline{\omega} \in L - L_k$, for $k \in \{1, \ldots, m\}$, such that $P(\omega) = P(\overline{\omega})$.

Proof:

(1) $\Rightarrow$ (2) : We suppose that there exits $i \in \{1, \ldots, m\}$ and a run over a trace $\widehat{\omega}$ in $\widehat{H}_{i,\bar{i}}$, reaching a state $(\widehat{q}, v)$, such that, $\widehat{q}$ is a marked location ($\widehat{q} \in M_{i,\bar{i}}$), and v is a valuation of $X_{i,\bar{i}}$. The timed trace $\widehat{\omega}$, has as suffix the event μ, which labels the last discrete transition of the run. Since this transition has a guard on the form $y^k \geq \theta_k$, $k \in \{1, \ldots, m\}$, we conclude that there exists a stop-watch $y^k, k \in \{1, \ldots, m\}$, such that $v(y^k) \geq \theta_k$. Therefore, the trace $\widehat{\omega}$ contains necessarily a failure from F_k.

Since $\widehat{H}_{i,\bar{i}} = H_{i,\bar{i}} \otimes TB$, we suppose that the location $\widehat{q}$ corresponds to the pair $(q_{i,\bar{i}}, q_M)$, where $q_{i,\bar{i}}$ is a location of the RHA $H_{i,\bar{i}}$, and q_M is a location TB. Let ω denotes the timed trace reaching the state $(q_{i,\bar{i}}, v)$ (ω is obtained by trimming the event μ from $\widehat{\omega}$). Since ω contains a failure from F_k, and according to Lemma 5, there exists

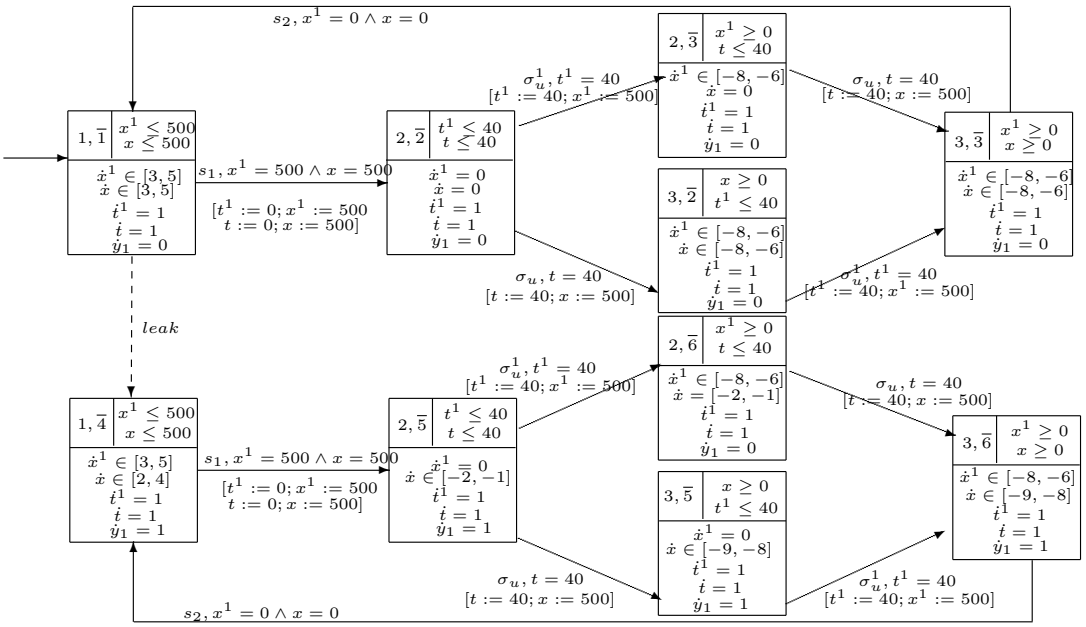

Fig. 4. *Product RHA* $H_{1,\overline{1}}$

a pair of traces, $\omega_1 \in L_k$ and $\omega_2 \in L - L_k$, such that $P(\omega_1) = P(\omega_2)$, reaching, respectively, the states (q_1, v_1) in H_i, and (q_2, v_2) in $H_{\overline{i}}$, where $q_{i,\overline{i}} = (q_1, q_2)$ and $v = [v_1 \ v_2]^T$. Two cases can be distinguished:

- First case: $k = i$, ω_1 contains a failure from F_i, and $v_1(y^i) > \theta_i$. Therefore, $\omega_1 \in L_i^{\theta_i}$ and $\omega_2 \in L - L_i$.
- Second case: $k \neq i$, ω_2 contains a failure from $F_k, k \neq i$, and $v_2(y^k) > \theta_k$. Therefore, $\omega_1 \in L - L_k$ and $\omega_2 \in L_k^{\theta_k}$.

We conclude that, in both cases, there exist two traces in, respectively, $L_k^{\theta_k}$ L_k, $k \in \{1, \ldots, m\}$, having identical observable projections.

$(2) \Rightarrow (1)$:

We suppose that there exist $k \in \{1, \ldots, m\}$, and two traces, $\omega_1 \in L_k^{\theta_k}$, and $\omega_2 \in L - L_k$, such that $P(\omega_1) = P(\omega_2)$. Since $L_k^{\theta_k} \subseteq L_k$, by Lemma 5, there exists $i \in \{1, \ldots, m\}$, and a timed trace ω, accepted by the RHA $\widehat{H}_{i,\overline{i}}$, such that, ω contains at least a failure from F_k. Two cases are possibles:

- $i \neq k$, and $\omega_2 \in L_i^{\theta_i}$. Then, ω contains two failures from F_i and F_k. Let d denotes the minimum between $(t_i + \theta_i)$ and $(t_k + \theta_k)$, where t_i and t_k denote, respectively, the dates of the first occurrences of failures from F_i and F_k in ω.

  Let us consider the prefix $\widehat{\omega}$ of ω, such that $time(\widehat{\omega} = d$. We merge the trace $\widehat{\omega}$ with the event μ, in the suffix position.
- $i = k$, or $\omega_2 \notin L_i^{\theta_i}$. We consider the prefix $\widehat{\omega}$ of ω, such that $time(\widehat{\omega} = t_k + \theta_k$, where t_k denotes the date of the first occurrence of a failures from F_k in ω. We merge the trace $\widehat{\omega}$ with the event μ, in the suffix position.

In both cases, we conclude that there exists a timed trace $\widehat{\omega}$, reaching a marked location in $\widehat{H}_{i,\overline{i}}$.

Let H denotes an RHA verifying assumptions A_1 and A_2; let L denotes the corresponding accepted language, and θ a failure/delay mapping function; and, let $\widehat{H}_{i,\overline{i}}$, for $i \in \{1, \ldots, m\}$, denotes the corresponding product RHAs, as given above. The following theorem establish the relation between the reachability of marked locations in $\widehat{H}_{i,\overline{i}}$, and the LTLa diagnosability of L.

Theorem 2. The timed language L is LTLa diagnosable w.r.t. θ, iff, for all $i \in \{1, \ldots, m\}$, no marked locations are reachable in $\widehat{H}_{i,\overline{i}}$ $(\forall i \in \{1, \ldots, m\}, Reachable(\widehat{H}_{i,\overline{i}}) = `FALSE')$.

Proof:

$\Longrightarrow$) We suppose that a marked location is reachable is in $\widehat{H}_{i,\overline{i}}$. Hence, $\exists q \in M_{i,\overline{i}}, i \in \{1, \ldots, m\}$, such that the marked location q is reachable, over a run of $\widehat{H}_{i,\overline{i}}$, on a timed trace ω. By Lemma 7, there exist two timed traces, $\omega \in L_k^{\theta_k}$, and $\overline{\omega} \in L - L_k$, $k \in \{1, \ldots, m\}$, such that $P(\omega) = P(\overline{\omega})$. We conclude that L is not LTLa diagnosable w.r.t. θ.

$\Longleftarrow$) We suppose that L is not LTL a diagnosable. Then, there exist $k \in \{1, \ldots, m\}$, and a pair of traces $\omega \in L_k^{\theta_k}$ and $\overline{\omega} \notin L_k$, such that $P(\omega) = P(\overline{\omega})$. Since $\overline{\omega} \in L - L_k$, and by Lemma 7, we conclude that there exist $i \in \{1, \ldots, m\}$, and a run in $\widehat{H}_{i,\overline{i}}$, reaching a marked location.

Example 2. As an illustration of our LTLa diagnosability verification approach, we apply it below on the system of Example 1, given above.

It is clear by inspection of the RHA in Fig.**??**, that is the system model in Example 1, satisfies the assumptions A_1 and A_2. In this example we consider two applications of our LTLa diagnosability procedure for two values of θ: 125 t.u., and 300 t.u..

As a first step, we build the RHA $H_{1,\overline{1}} = H_1 \otimes H_{\overline{1}}$, given in Fig4. For space limitations reasons, we do not provide the product $H_{1,\overline{1}} \otimes TB$.

By substituting θ_1 by the value 125 in $H_{1,\overline{1}} \otimes TB$, and applying the reachability function, given in Algorithm 1, we conclude that there exists a reachable marked location in $H_{1,\overline{1}} \otimes TB$. In fact, it can be easily checked that the state, $((2,\overline{5},q_M),(x = x^1 = 500, t = t^1 = 0, y^1 = 125))$, having as discrete part the marked location $(2,\overline{5},q_M)$, is reachable. Therefore, there exists a pair of traces in $L_1^{\theta_1}$ and $L_{\overline{1}}$, having identical observable projections, that implies the non LTLa diagnosability of the system, w.r.t. $\theta_1 = 125$.

In the second case, we substitute θ_1 by the value 300, The reachability function, applied to the RHA $H_{1,\overline{1}} \otimes TB$, returns '$FALSE$', indicating the non existence of runs reaching marked locations. We conclude that the timed language, corresponding our system model, is LTLa diagnosable for $\theta_1 = 300$, confirming our intuitive result, established in Section3.

5. CONCLUSION

In this paper we presented a definition of LTLa diagnosability for timed languages, accepted by RHAs. A systematic approach for checking the LTLa diagnosability, of these systems, under some special assumptions, is given. Our approach consists of constructing a set of RHA products, and exploring them with a reachability analysis function. Finally, we have formally established the relation between our checking approach, and the the LTLa diagnosability.

REFERENCES

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

Alur, R., Courcoubetis, C., Halbwachs, N., Henzinger, T.A., h. Ho, P., Nicollin, X., Olivero, A., Sifakis, J., and Yovine, S. (1995). The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138, 3–34.

Ben.Hadj-Alouane, N., Lafortune, S., and Lin, F. (1994). Variable lookahead supervisory control with state information. *IEEE Transactions on Automatic Control*, 39(12), 2398–2410.

Bhowal, P., Sarkar, D., Mukhopadhyay, S., and Basu, A. (2007). Fault diagnosis in discrete time hybrid systems - a case study. *Inf. Sci.*, 177(5), 1290–1308.

Derbel, H., Alla, H., Ben.Hadj-Alouane, N., and Yeddes, M. (2009). Online diagnosis of systems with rectangular hybrid automata models. In *to appear in INCOM'09, Moscow.*

Derbel, H., Yeddes, M., Ben.Hadj-Alouane, N., and Alla, H. (2006). Diagnosis of a class of timed discrete event systems. In *8th International Workshop on Discrete Event Systems*, 256–261.

Fourlas, K., Kyriakopoulos, K., and Krikelis, N. (2002). 10th mediterranean conference on control and automation-med2002, lisbon, portugal. In *8th International Workshop on Discrete Event Systems*, 3994–3999.

Frehse, G. (2005). Phaver: Algorithmic verification of hybrid systems past hytech. In *Fifth International Workshop on Hybrid Systems: Computation and Control (HSCC)*, 258–273.

Henzinger, T., Kopke, P., Puri, A., and Varaiya, P. (1998a). The what's decidable about hybrid automata? *Journal of Computer and System Sciences*, 57, 94–124.

Henzinger, T.A., Ho, P.H., and Wong-Toi, H. (1997). HYTECH: A model checker for hybrid systems. *International Journal on Software Tools for Technology Transfer*, 1(1–2), 110–122.

Henzinger, T.A., Kopke, P.W., Puri, A., and Varaiya, P. (1998b). What's decidable about hybrid automata. *Journal of Computer and System Sciences*, 57, 94–124.

Lin, F. and Wonham, W.M. (1988). On observability of discrete-event systems. *Information sciences*, 44(3), 173–198.

Lin, F. and Wonham, W.M. (1994). Diagnosability of discrete event systems and its applications. *Discrete Event Dynamic Systems*, 4(2).

McIlraith, S.A., Biswas, G., Clancy, D., and Gupta, V. (2000). Hybrid systems diagnosis. In *HSCC '00: Proceedings of the Third International Workshop on Hybrid Systems: Computation and Control*, 282–295. Springer-Verlag, London, UK.

Sampath, M., Sengupta, R., Lafortune, S., and Sinnamohideen, K. (1996). Failure diagnosis using discrete event models. *IEEE Trans. on Control Systems Technology*, 4(2), 105–124.

Sampath, M., Sengupta, R., Lafortune, S., Sinnamohideen, K., and Teneketzis, D. (1995). Diagnosability of discrete event systems. *IEEE Trans. on Automatic Control*, 40(9), 1555–1575.

Tripakis, S. (2002). Fault diagnosis for timed automata. In *FTRTFT '02: Proceedings of the 7th International Symposium on Formal Techniques in Real-Time and Fault-Tolerant Systems*, 205–224. Springer-Verlag, London, UK.

Zad, S.H., Kwong, R.H., and Wonham, W.M. (1998). Fault diagnosis in discrete-event systems. In *in Proc. 1998 IEEE Conference on Decision and Control (CDC98*, 3769–3774.

Zhao, F., Koutsoukos, X.D., Haussecker, H.W., Reich, J., and Cheung, P. (2005). Monitoring and fault diagnosis of hybrid systems. *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 35(6), 1225–1240.