AN APPROACH FOR FAULT DETECTION IN DEVS MODELS

Diego M. Llarrull[(a)], Norbert Giambiasi [(b)]

[(a)] CIFASIS - CCT-CONICET. 27 de Febrero 210 bis. S2000EZP - Rosario
[(b)] LSIS - UMR CNRS 6168. University Paul Cézanne

[(a)] diego.llarrull@lsis.org, [(b)] norbert.giambiasi@lsis.org

ABSTRACT

The aim of this paper is to present a first approach for building distinguishing, homing, and synchronizing sequences for a subset of DEVS models, in order to apply to them the fault detection techniques developed on Mealy machines. After the definition of the considered subset of DEVS (called MealyDEVS), we present the extension of fault detection techniques on this DEVS subset.

Keywords: DEVS, fault detection, Mealy machines, black-box testing.

1. INTRODUCTION

The design of real-time discrete event control systems is a process that requires dedicated formalisms and adapted tools. In particular, the DEVS formalism is convenient for the low-level phase of the design process since it provides a suitable simulation framework, thus enabling the possibility of validating models by simulation.

In (Dacharry and Giambiasi 2005), a formal methodology for the design and verification of control systems is presented. With this methodology, a high-level specification of a system to be designed is constructed using a network of timed automata, and the corresponding implementation is expressed as a coupled DEVS model. This makes it possible to formally verify the conformance of critical components (atomic DEVS models against timed automata) and the conformance of the whole model. Nevertheless, due to the state explosion problem that frequently appears in the verification of models that deal with a dense time base, the automatic verification of the conformance between the high-level and the low-level models is unfeasible in the general case.

Despite this discouraging result, a partial automatic validation of the conformance relation between an implementation and its specification is possible, for example, by generating test cases on a high-level specification and applying these tests to the low-level model description. Several formalisms have a developed theory of fault detection techniques. Most of them are related to Mealy Machines, which have been widely used for testing purposes in various domains. Mealy Machines are based on the hypothesis of simultaneous input/output events and are untimed models.

Our proposal is then to allow the use of fault detection techniques on timed models of higher complexity. In this paper we will extend the theory presented by (Kohavi 1978) to a subset of the models that can be represented using the DEVS formalism.

The paper is organized as follows: in Section 1 we recall the existing theory, together with the tools and concepts that will be necessary to extend it. In Section 2, we introduce a subset of the DEVS formalism that we take under consideration, and we adapt and extend the existing methods, concepts and definitions to this subset. In Section 3, we propose an extension of the first subset of models in order to enlarge the spectrum of models to which the theory of fault detection can be applied, and we briefly show some considerations about the implementation of these testing methods. Finally, we conclude the paper.

2. PRELIMINARIES

2.1. Mealy Machines

A Mealy Machine (Kohavi 1978, Lee et al 1996) is formally stated as a quintuple $M = (I, O, S, \delta, \lambda)$ where *I*, *O* and *S* are finite and nonempty sets of input symbols, output symbols, and states respectively, $\delta : S \times I \rightarrow S$ is the state transition function and $\lambda : S \times I \rightarrow O$ is the output function.

When the machine is in a current state $s_i \in S$ and receives an input $a \in I$ it moves to the next state s_j specified by $\delta(s_i, a) = s_j$ and produces immediately the output y specified by $\lambda(s_i, a) = y$.

2.1.1. Execution fragments and Traces

In the following, some concepts that will be useful in the subsequent sections are recalled (Kohavi 1978, Lynch and Vaandrager 1993a, Lynch and Vaandrager 1993b) in order to adjust them to the syntax used by the DEVS formalism and the concepts of executions and traces as defined in (Dacharry and Giambiasi 2005).

Definition 2.1 (Execution fragment) *Let $\mathcal{M} = (X_{\mathcal{M}}, Y_{\mathcal{M}}, S_{\mathcal{M}}, \delta_{\mathcal{M}}, \lambda_{\mathcal{M}})$ be a Mealy machine. Then an execution fragment for M is a finite or infinite alternating sequence of the form $s_0, x_0, y_0, s_1, x_1, y_1, \ldots$ beginning with a state (and if it is finite also ending with a state), such that* $\forall i \in \{0..n\} \bullet x_i \in X_{\mathcal{M}}, y_i \in Y_{\mathcal{M}}, s_i \in S_{\mathcal{M}}, \ \delta_{\mathcal{M}}(s_i, x_i) = s_{i+1} \ \wedge$ $\lambda_{\mathcal{M}}(s_i, x_i) = y_i$

Definition 2.2 (Execution of a Mealy machine) *Let* $\mathcal{M} = (X_{\mathcal{M}}, Y_{\mathcal{M}}, S_{\mathcal{M}}, \delta_{\mathcal{M}}, \lambda_{\mathcal{M}})$ *be a Mealy machine. Then an execution for* $\mathcal{M}$ *is a finite execution fragment of* $\mathcal{M}$ *that begins with a starting state. We denote with* $execs^{*}(\mathcal{M})$, $execs^{\omega}(\mathcal{M})$, *and execs(M) the sets of finite, infinite, and all executions of* $\mathcal{M}$, *respectively*

Definition 2.3 (Reachability of a state) *Let* $\mathcal{M} = (X_{\mathcal{M}}, Y_{\mathcal{M}}, S_{\mathcal{M}}, \delta_{\mathcal{M}}, \lambda_{\mathcal{M}})$ *be a Mealy machine, and* ***last*** *a function such that* ***last(*** p ***)*** *is the last element of the finite sequence* π. *Then a state* $s_i \in S_{\mathcal{M}}$ *is reachable if* $s = \textbf{last}(\alpha)$ *for some finite execution* α *of* $\mathcal{M}$

Definition 2.4 (Trace) *Let* $\mathcal{M} = (X_{\mathcal{M}}, Y_{\mathcal{M}}, S_{\mathcal{M}}, \delta_{\mathcal{M}}, \lambda_{\mathcal{M}})$ *be a Mealy machine,* α *an execution fragment of* $\mathcal{M}$ *and* ***states(*** α ***)*** *the set of states that appear in* α. *Let* γ *be the sequence consisting of the events in* α. *Then* ***trace(*** α ***)*** *is defined to be the tuple* (θ_I, θ_O) *consisting of the members of* γ, *where* $\forall a_{O_i} \in \theta_O, a_{I_i} \in \theta_I, s_j \in \textit{states}(\alpha) \bullet \lambda_{\mathcal{M}}(s_j, a_{I_j}) = a_{O_j}$
Moreover, a finite or infinite tuple β *is a trace of* $\mathcal{M}$ *if* $\mathcal{M}$ *has an execution* α *with* $\beta =$ $traces(\alpha)$. *We denote with* $\textbf{traces}^{*}(\mathcal{M})$, $\textbf{traces}^{\omega}(\mathcal{M})$ *and* $\textbf{traces}(\mathcal{M})$ *the sets of finite, infinite, and all traces of* $\mathcal{M}$, *respectively.*

2.1.2. **Minimality - State Equivalence** (Kohavi 1978)

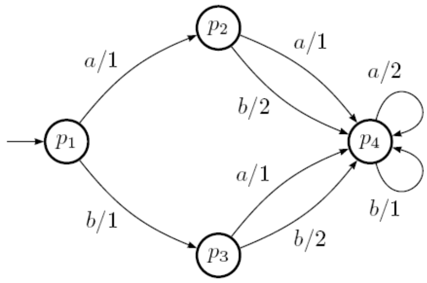
Definition 2.5 (Distinguishing sequence) *Two states* s_i *and* s_j *of a Mealy machine* $\mathcal{M}$ *are distinguishable if and only if there exist at least two execution fragments* α *and* β *of* $\mathcal{M}$ *with* $\textbf{traces}(\alpha) = (\theta_{I_\alpha}, \theta_{O_\alpha})$, $\textbf{traces}(\beta) = (\theta_{I_\beta}, \theta_{O_\beta})$ *where* $first(\alpha) = s_i$, $first(\beta) = s_j$, $\theta_{I_\alpha} = \theta_{I_\beta}$ *and* $\theta_{O_\alpha} \neq \theta_{O_\beta}$. θ_{I_α} $(= \theta_{I_\beta})$ *is called a distinguishing sequence of the pair* (s_i, s_j). *If there exists for pair* (s_i, s_j) *a distinguishing sequence of length k, then the states in* (s_i, s_j) *are said to be k-distinguishable.*

Definition 2.6 (k-equivalence of states) *Two states* s_i *and* s_j *of a Mealy machine M are k-equivalent if and only if they are not k-distinguishable.*

Definition 2.7 (Equivalence of states) *Two states* s_i *and* s_j *of a Mealy machine* $\mathcal{M}$ *are equivalent if and only if they are k-equivalent* $\forall k \in \Re_0^{+}$. *In other words,* s_i *and* s_j *are equivalent if and only if, for every possible input sequence, the same output sequence will be produced regardless of whether* s_i *or* s_j *is the initial state.*

Definition 2.8 (Equivalence of Mealy machines) *Two Mealy machines* $\mathcal{M}_1$ *and* $\mathcal{M}_2$ *are equivalent if and only if, for every state in* $\mathcal{M}_1$, *there is a corresponding equivalent state in* $\mathcal{M}_2$, *and vice-versa.*

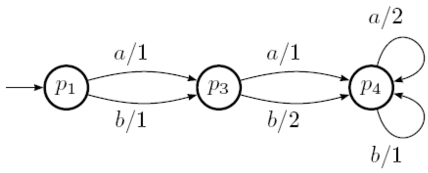
Definition 2.9 (Minimal Mealy machine) *A Mealy Machine is minimal (reduced) if and only if no two states in it are equivalent. Additionally, the Mealy machine* $\mathcal{M}_1$ *which contains no equivalent states and is equivalent to the Mealy machine* $\mathcal{M}_2$ *is said to be the minimal, or reduced form of* $\mathcal{M}_2$.



(a)

State	$x = a$	$x = b$
p_1	$p_2, 1$	$p_3, 1$
p_2	$p_4, 1$	$p_4, 2$
p_3	$p_4, 1$	$p_4, 2$
p_4	$p_4, 2$	$p_4, 1$

(b)



(c)

Phase	$x = a$	$x = b$
p_1	$p_3, 1$	$p_3, 1$
p_3	$p_4, 1$	$p_4, 2$
p_4	$p_4, 2$	$p_4, 1$

(d)

Figure 1: Mealy machine (a) with its associated transition table (b). Resulting minimal Mealy machine (c) after the deletion of p_2 which is equivalent to p_3 and the resulting transition table (d).

2.2. Labelled Timed Transition Systems

Definition 2.10 (Labelled Timed Transition System) A labelled timed transition system $\mathcal{T}_t$ *is an automaton whose alphabet includes* $\Re^{+}$. *The transitions* *corresponding to symbols from $\Re^+$ are referred to as time-passage transitions, while non-time-passage transitions are referred to as discrete transitions. So, a labelled timed transition system consists of:*

S a possibly infinite set of states,

INIT an initial state,

Σ a set of discrete actions,

D a set of discrete transitions, noted $s \xrightarrow{x} s'$, where $x \in \Sigma_{T_t}$ and $s, s' \in S_{T_t}$, asserting that

"from state s the system can instantaneously move to state s' via the occurrence of the event x", and

T a set of time-passage transitions, noted $s \xrightarrow{t} s'$, where $t \in \Re^+$ and $s, s' \in S_{T_t}$, asserting that

"from state s the system can move to state s' during a positive amount of time t in which no discrete events occur".

A labelled timed transition system is assumed to satisfy two axioms.

***S1** If $s \xrightarrow{t} s'$ and $s' \xrightarrow{t'} s''$, with $t, t' \in \Re^+$, then $s \xrightarrow{t+t'} s''$.*

***S2** Each time-passage step, $s \xrightarrow{t} s'$, with $t \in \Re^+$, has a trajectory, where a trajectory describes the state changes than can occur during time-passage transitions. If I is any closed interval of $\Re_0^+$ beginning with 0, an I–trajectory is defined as a function, $v: I \rightarrow S$ such that:*

$$v(t) \xrightarrow{t-t'} v(t') \quad \forall t, t' \in I \,|\, t < t'$$

It will be useful for our purposes to abstract away the quantitative aspect of time in LTSs. The relationship concerned is called *Delay Time-Abstracting Bisimulation* (Tripakis 2001).

Definition 2.11 (Delay Time-Abstracting Bisimulation) *Consider a labelled timed transition system A with sets of discrete transitions D and time-passage transitions E. A binary relation ≈ on the states of A is a delay time-abstracting bisimulation (DTaB) if, for all states $s_1 \approx s_2$, the following conditions hold:*

1. if $s_1 \xrightarrow{d_1} s_3$, for some $d_1 \in D$ then there exists $\delta_1 \in E$ and $d_2 \in D$ such that $s_2 \xrightarrow{\delta_1} \xrightarrow{d_2} s_4$ and $s_3 \approx s_4$;
2. if $s_1 \xrightarrow{\delta_1} s_3$, for some $\delta_1 \in E$ then there exists $\delta_2 \in E$ such that $s_2 \xrightarrow{\delta_2} s_4$ and $s_3 \approx s_4$;
3. The above conditions also hold if the roles of s_1 and s_2 are reversed.

Then, the states s_1 and s_2 are said to be DTa-bisimilar. In general, two TTSs $\mathcal{A}_1$ and $\mathcal{A}_2$ are said to be DTa-bisimilar if there exists a DTaB ≈ on the states of $\mathcal{A}_1$ and $\mathcal{A}_2$ such that $s_0^1 \approx s_0^2$, where s_0^i is the initial state of $\mathcal{A}_i$.

2.3. DEVS formalism

A DEVS model (Zeigler 2000) is a structure $M = \langle X, S, Y, \delta_{int}, \delta_{ext}, \lambda, ta \rangle$ where

- *X* is the set of input values
- *S* is a set of states,
- *Y* is the set of output values
- $\delta_{int}: S \rightarrow S$ is the internal transition function
- $\delta_{ext}: Q \times X \rightarrow S$ is the external transition function, where
- $Q = \{(s,e) \,|\, s \in S, 0 \le e \le ta(s)\}$ is the total state set
- *e* is the time elapsed since the last transition
- $\lambda : S \rightarrow Y$ is the output function
- $ta : S \rightarrow \Re_0^+ \cup \infty$ is the time advance function.

The interpretation of these elements is the following: at any time the model is in some state, *s*. If no external event occurs the model stay in state *s* for time *ta*(*s*). Notice that *ta*(*s*) could be a real number. But it can also take on the values 0 and ∞. Depending on the value of *ta*(*s*), an atomic DEVS model has two kinds of states:

- *Passive States:* A state $s \in S$ is called *passive* iff *ta*(*s*) = ∞ and no internal transition is defined in it. The set of all passive states of a DEVS model is referred to as S_p.
- *Active States:* A state $s \in S$ is called *active* iff *ta*(*s*) ≠ ∞. The set of all active states of a DEVS model is referred to as S_a.

When the elapsed time in the current state, *e,* equals *ta*(*s*), the system outputs the value, λ(*s*), and changes to state $\delta_{int}(s)$. If an external event *x* occurs before *e = ta(s)*, the model transits into the state $\delta_{ext}(s,e,x)$.

The dense-time characteristics of DEVS models impose a restriction in the tractability of the problem of fault detection techniques in the most general case. It is then necessary to reduce the subset of DEVS models to those models where these techniques can be applied in an efficient way.

2.3.1. Execution fragments and Traces for DEVS model (Giambiasi and Dacharry 2007):

Definition 2.12 (Execution fragment of a DEVS model) *Let* $\mathcal{D} = (X_\mathcal{D}, Y_\mathcal{D}, S_\mathcal{D}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_\mathcal{D}, ta_\mathcal{D})$ *be a DEVS model. Then an execution fragment for D is a finite alternating sequence* $\Upsilon = v_0\, x_1\, v_1\, x_2\, v_2 \ldots x_n\, v_n$ *where:*

- **time-passage transitions:** Each v_i is a function from a real interval $I_i = [0, t_i]$ to the set of total phases of *D*, such that $\forall j, j' \in I_i \mid j < j'$, if $v_i(j) = (s,e)$ then $v_i(j') = (s, e + j' - j)$
- **event transitions:** Each x_i is an input or output event, and if $(s,e) = v_{i-1}(sup(I_{i-1}))$, $(s',0) = v_i(inf(I_i))$, one of the following conditions hold:

1. $x_i \in Y_\mathcal{D}, \delta_{int\mathcal{D}}(s) = s'$, ta(s)=e, and $\lambda(s) = x_i$.
2. $x_i \in X_\mathcal{D}, \delta_{ext\mathcal{D}}(s, e, x_i) = s'$, and e ≤ ta(s).

The definitions of execution and state reachability of DEVS models are analogous to those of Mealy machines. However, due to the timed nature of DEVS models, the definitions of traces and distinguishability of states are more complex (Giambiasi and Dacharry 2007):

Definition 2.13 (Trace of a DEVS) *Let* $\mathcal{D} = (X_\mathcal{D}, Y_\mathcal{D}, S_\mathcal{D}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_\mathcal{D}, ta_\mathcal{D})$ *be a DEVS model,* $\Upsilon = v_0\, x_1\, v_1\, x_2\, v_2 \ldots x_n\, v_n$ *an execution fragment of D. Then trace*(Υ) *is defined to be a tuple* (θ_I, θ_O, t) *such that* θ_I *and* θ_O *are sequences consisting of all pairs of events of* Υ *and their time of occurrence, sorted in chronological order of occurrence, and t is the total time of execution, defined as* $\sum_{0 \le j \le n} (sup(I_j))$*. Formally, the time of occurrence of an event* x_i *of* Υ *is equal to* $\sum_{0 \le j < i} (sup(I_j))$*, with* I_j *the domain of* v_j*.*

The set of all traces of a DEVS model is defined as traces$(\mathcal{D})$ = { *trace*$(\Upsilon) \mid \Upsilon \in$ *execs*$(\mathcal{D})$ }.

2.3.2. Associated Transition System for a DEVS model

The semantics of a DEVS model can be clearly stated by means of its *associated Timed-Transition System* (Giambiasi and Dacharry 2007).

Definition 2.14 (Associated Transition System) *Given a DEVS model D its associated transition system is defined over the alphabet* $\Sigma = X_\mathcal{D} \cup Y_\mathcal{D}$ *, Taut(D) as a labeled timed transition system* $\mathcal{T}_t$*, where:*

1. *the set of states,* $S_{\mathcal{T}_t}$*, consists of the set of total phases of D ,* $Q_\mathcal{D}$*,*
2. *the initial phase, init(*$\mathcal{T}_t$*) is (s,0), where* $s \in S_\mathcal{D}$*, and s is the discrete phase defined as the initial phase of the DEVS model,*
3. *the set of discrete transitions,* $D_{\mathcal{T}_t}$

   $D_{\mathcal{T}_t} = \{(s,e) \xrightarrow{x} (s',0) \mid (\delta_{int}(s) = s'$
   $\wedge \lambda(s) = x \wedge e = ta(s))$ *or*
   $(\delta_{ext}((s,e),x) = s' \wedge e \le ta(s))\}$
4. *the set of time-passage transitions,* $T_{\mathcal{T}_t}$ *,*
5. $T_{\mathcal{T}_t} = \{(s,e) \xrightarrow{t} (s,e') \mid (s,e) \in Q_\mathcal{D}, e' = e + t,$
   $0 \le e + t \le ta(s)\}$

The labelled timed transition system associated with a DEVS model, as defined above, specifies the same set of traces as its corresponding DEVS model (Giambiasi and Dacharry 2007).

3. MEALYDEVS

The basic idea for constructing a DEVS model which behaves exactly like a Mealy machine is that of forcing an immediate input/output response. As a consequence, the subset of DEVS models that we consider first consists only of such models where $\forall s \in S_a \Rightarrow ta(s)$=0. For the models of this subset, all active states are *transitory*. Transitory states are said to be *input-blocking* (Giambiasi and Dacharry 2005). That is, the MealyDEVS model does not stay in an active state; it appears (externally) to be always ready for input. Then, every state transition of a Mealy machine $\mathcal{M} = (I_\mathcal{M}, O_\mathcal{M}, S_\mathcal{M}, \delta_\mathcal{M}, \lambda_\mathcal{M})$ that has the form

$s_i \xrightarrow{x/y} s_j$, where $x \in I_\mathcal{M}$, $y \in O_\mathcal{M}$, $s_i, s_j \in S_\mathcal{M}$

and *x/y* means that the input event *x* is received and the output event *y* is sent in this transition, is translated into two transitions in the corresponding DEVS model $\mathcal{D} = < X_\mathcal{D}, Y_\mathcal{D}, S_\mathcal{D}, \delta_{int_\mathcal{D}}, \delta_{ext_\mathcal{D}}, \lambda_\mathcal{D}, ta_\mathcal{D} >$:

- An external transition of the form $s_i \xrightarrow{x/-} s_{i,x}$ where $\delta_{ext_\mathcal{D}}(s_i, e, x) = s_{i,x}$, $s_i \in S_{s_\mathcal{D}}$, $e \in \Re_0^+ \cup \{\infty\}$, $x \in X_\mathcal{D}$ and $s_{i,x} \in S_{a_\mathcal{D}}$.
- An internal transition of the form $s_{i,x} \xrightarrow{-/y} s_j$ where $\delta_{int_\mathcal{D}}(s_{i,k}) = s_j$, $\lambda(s_{i,k}) = y$, $s_{i,k} \in S_{a_\mathcal{D}}$, $y \in Y_\mathcal{D}$ and $s_j \in S_{s_\mathcal{D}}$.

This translation is possible provided the following constraints are satisfied: firstly, it is needed that both models have the same input and output event sets (that is, $I_\mathcal{M} = X_\mathcal{D} \wedge O_\mathcal{M} = Y_\mathcal{D}$), and the set of passive states of *D* has to be equal to the set of all states of *M* ($S_\mathcal{M} = S_{s_\mathcal{D}}$). It is also required that for each possible transition $\delta_{\mathcal{M}}(s_i, x) = s_j$ of M there exist two transitions in D, $\delta_{ext_{\mathcal{D}}}(s_i, e, x)$ and $\delta_{int_{\mathcal{D}}}(s_{i,x})$ such that $\delta_{\mathcal{M}}(s_i, x) = s_j \Leftrightarrow \forall e \in \Re_0^+ \cup \{\infty\} \bullet \delta_{ext_{\mathcal{D}}}(s_i, e, x) = s_{i,x}$ and $\delta_{int_{\mathcal{D}}}(s_{i,x}) = s_j$. Finally, for each possible output event $y \in I_{\mathcal{M}}$ such that $\lambda_{\mathcal{M}}(s_i, x) = y$ it is required an analogous output event in D (in symbols, $\lambda_{\mathcal{M}}(s_i, x) = y \Leftrightarrow \lambda_{\mathcal{D}}(s_{i,x}) = y$).

Note that in this subset of atomic DEVS models the external transition function *does not depend on the elapsed time*. That is, $\forall s_i \in S_{p\mathcal{D}}; e, e' \in \Re_0^+ \cup \{\infty\} \bullet \delta_{ext\mathcal{D}}(s_i, e, x) = \delta_{ext\mathcal{D}}(s_i, e', x)$.

The subset of DEVS models which capture this behaviour is called MealyDEVS and is formally defined as:

Definition 3.1 (MealyDEVS) *A DEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *is a MealyDEVS model if and only if*

- $S_{\mathcal{D}} = S_{a\mathcal{D}}$ *(active states)* $\cup\, S_{p\mathcal{D}}$ *(passive states).*
- $\delta_{ext\mathcal{D}} : S_{p\mathcal{D}} \times \Re_0^+ \cup \{\infty\} \times X \to S_{a\mathcal{D}}$ *where*
- $\forall s_i \in S_{p\mathcal{D}}; e, e' \in \Re_0^+ \cup \{\infty\} \bullet$ $\delta_{ext\mathcal{D}}(s_i, e, x) = \delta_{ext\mathcal{D}}(s_i, e', x)$
- $\delta_{int\mathcal{D}} : S_{a\mathcal{D}} \to S_{p\mathcal{D}}$
- $\lambda_{\mathcal{D}} : S_{a\mathcal{D}} \to Y_{\mathcal{D}}$ *where* $\forall s_j \in S_{a\mathcal{D}} \bullet ta_{\mathcal{D}}(s_j) = 0$ *(All active states are transitory)*

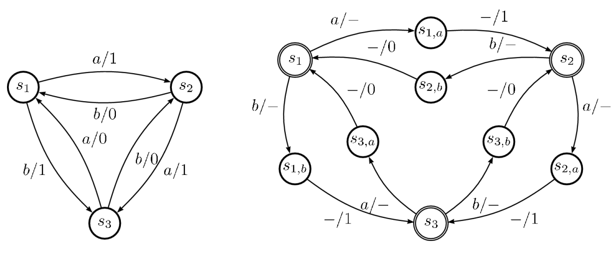


Figure 2: Transition diagram for a Mealy machine and its corresponding MealyDEVS model.

We consider only *completely specified* models.

It is straightforward that every Mealy machine has the same input/output behaviour as its corresponding MealyDEVS model.

Remark: The previous definitions and procedures can be applied to untimed DEVS (Giambiasi and Dacharry 2007).

4. EXTENDED MEALYDEVS

It should be clear that the MealyDEVS subset represents a tiny subset of the systems than can be modelled using the DEVS formalism. It is then of major interest to expand this subset in order to apply fault detection techniques for a wider range of DEVS models.

We consider now a subset called Extended MealyDEVS. In this subset, the considered DEVS models have *a time advance function which can take arbitrary finite values on active states* (in symbols: $\forall s_i \in S_{a\mathcal{D}} \bullet\ ta(s_i) \in \Re_0^+$).

Definition 4.1 (Extended MealyDEVS) *A DEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *is an Extended MealyDEVS if and only if*

- $S_{\mathcal{D}} = S_{a\mathcal{D}} \cup S_{p\mathcal{D}}$ where $S_{a\mathcal{D}} \cap S_{p\mathcal{D}} = \varnothing$
- $\delta_{ext\mathcal{D}} : S_{p\mathcal{D}} \times \Re_0^+ \cup \{\infty\} \times X \to S_{a\mathcal{D}}$ where $\forall s_i \in S_{p\mathcal{D}}; e, e' \in \Re_0^+ \cup \{\infty\} \bullet$ $\delta_{ext\mathcal{D}}(s_i, e, x) = \delta_{ext\mathcal{D}}(s_i, e', x)$
- $\delta_{int\mathcal{D}} : S_{a\mathcal{D}} \to S_{p\mathcal{D}}$
- $\lambda_{\mathcal{D}} : S_{a\mathcal{D}} \to Y_{\mathcal{D}}$
- $ta_{\mathcal{D}} : S_{\mathcal{D}} \to \Re_0^+ \cup \{\infty\}$ where $\forall s_i \in S_{p\mathcal{D}} \bullet ta_{\mathcal{D}}(s_i) = \infty$ and $\exists k \in \Re_0^+ \mid \forall s_j \in S_{a\mathcal{D}} \bullet ta_{\mathcal{D}}(s_j) < k$ (ta takes finite real values on active states).

It is necessary to define a special kind of timed input sequence so that it ensures that the input events always occur when the model is in a passive state. In order to ensure that a model $\mathcal{D}$ is in a passive state, the input has to be *delayed* for at least t_k units of time, where t_k = min $\{x \in \Re^+ \mid > t\}$ and $t = max\{ta(s_i) \mid s_i \in S_{a\mathcal{D}}\}$, which is the maximum time that can elapse before the model reaches another passive state. Such a sequence is called a *slow timed input sequence*:

Definition 4.2 (Slow timed sequence) *Let* $\mathcal{D} =< X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{int_{\mathcal{D}}}, \delta_{ext_{\mathcal{D}}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}} >$ *be a DEVS model, where* $S_{a\mathcal{D}} \subset S_{\mathcal{D}}$. *A timed sequence for* $\mathcal{D}$ *is a finite series of the form* $< (x_0, t_0), (x_1, t_1), .., (x_n, t_n) >$ *such that*

$\forall i \in \{0..n\} \bullet x_i \in X_{\mathcal{D}} \wedge \forall i \in \{0..n\} \bullet t_i \in \Re_0^+ \wedge t_{i+1} - t_i > k$

where $k = max(ta_{\mathcal{D}}(s_{a_i}) \mid s_{a_i} \in S_{a\mathcal{D}})$

In order to formally explicit the relation between a MealyDEVS model and an Extended MealyDEVS model, we show subsequently the existence of a *Delay Time-Abstracting Bisimulation* (Tripakis and Yovine 2001) between them.

Theorem 4.1: *The MealyDEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *is DTa-bisimilar to the Extended MealyDEVS model* $\mathcal{E} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{E}})$ *which only differs from D in the values of ta(s).*

Proof: Since $\mathcal{D}$ and $\mathcal{E}$ have identical $\delta_{ext\mathcal{D}}$ and $\delta_{int\mathcal{D}}$ functions, it is straightforward that:

- For each time-passage transition of the form $s^{\mathcal{D}} \xrightarrow{\delta_1} s^{\mathcal{D}}$ in $\mathcal{D}$, where $\delta_1 \in \Re_0^+$ there is a corresponding transition $s^{\mathcal{E}} \xrightarrow{\delta_1} s^{\mathcal{E}}$ in $\mathcal{E}$, where s is a passive state and since there are no internal transitions in this kind of states.
- For each discrete transition of the form $s_0^{\mathcal{D}} \xrightarrow{x_i} s_{0,x_i}^{\mathcal{D}}$ where $x_i \in X_{\mathcal{D}}\ \forall i = 1..n_x$ there is a corresponding discrete transition of the form $s_0^{\mathcal{E}} \xrightarrow{x_i} s_{0,x_i}^{\mathcal{E}}$ in $\mathcal{E}$.
- For each discrete transition of the form $s_{0,x_i}^{\mathcal{D}} \xrightarrow{y_i} s_{x_{i+1}}^{\mathcal{D}}$ where $y_i \in Y_{\mathcal{D}}\ \forall i = 1..n_y$ there is a pair composed of a time-passage transition of the form $s_{0,x_i}^{\mathcal{E}} \xrightarrow{\delta_2} s_{0,x_i}^{\mathcal{E}}$, where $\delta_2 = ta_{\mathcal{E}}(s_{0,x_i}^{\mathcal{E}})$ and the corresponding discrete transition of the form $s_{0,x_i}^{\mathcal{E}} \xrightarrow{y_i} s_{x_{i+1}}^{\mathcal{E}}$.

Then, the relation $\approx: X_{\mathcal{D}}^{\mathcal{D}} \to X_{\mathcal{D}}^{\mathcal{E}}$ such that $s_i^{\mathcal{E}} \in \approx [s_i^{\mathcal{D}}]$ provides a DTaB between $\mathcal{D}$ and $\mathcal{E}$.

4.1. Minimality on Extended MealyDEVS models

For an Extended MealyDEVS model *D*, if *two passive states of D are distinguishable, all the active states to where they can transition will be distinguishable among each other*. Then, for our distinguishing purposes, we consider only passive states

Definition 4.3 (Distinguishing sequence) *Two passive states s_i and s_j of an extended MealyDEVS model D are distinguishable if and only if there exists at least two execution fragments* $\alpha = v_{\alpha_0}\ x_{\alpha_1}\ v_{\alpha_1}\ x_{\alpha_2}\ v_{\alpha_2} \ldots x_{\alpha_n}\ v_{\alpha_n}$ *and* $\beta = v_{\beta_0}\ x_{\beta_1}\ v_{\beta_1}\ x_{\beta_2}\ v_{\beta_2} \ldots x_{\beta_n}\ v_{\beta_n}$ *of* $\mathcal{M}$ *with trace(α)* $= (\theta_{I_\alpha}, \theta_{O_\alpha}, t_\alpha)$, *trace(β)* $= (\theta_{I_\beta}, \theta_{O_\beta}, t_\beta)$ *where* $v_{\alpha_0}(e) = (s_i, ta(s_i))\ \ \forall e \in I_{\alpha_0}$, $v_{\beta_0}(e) = (s_j, ta(s_j))\ \ \forall e \in I_{\beta_0}$, $\theta_{I_\alpha} = \theta_{I_\beta}$ *and* $(\theta_{O_\alpha} \neq \theta_{O_\beta})$. *The timed sequence* θ_{I_α} *(and* θ_{I_β} *) is called a distinguishing sequence of the pair* (s_i, s_j) . *If there exists for pair* (s_i, s_j) *a distinguishing sequence of length k, then the states in* (s_i, s_j) *are said to be k-distinguishable.*

Lemma 4.1 *In the previous definition,* θ_{I_α} *(and also* θ_{I_β} *) are slow timed input sequences.*

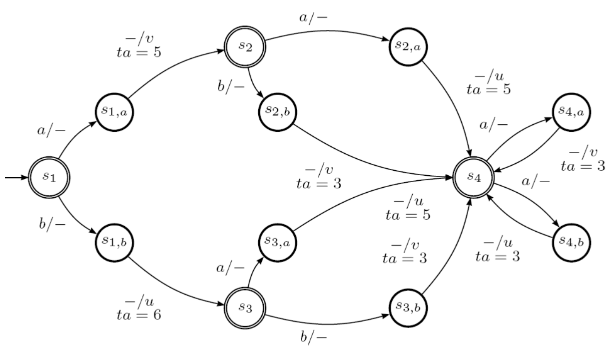
It should be straightforward to verify that this definition takes into account the discrepancies in the values of *ta* for active states: if $ta(s_{i,x}) \neq ta(s_{j,x})$ for some $s_{i,x}, s_{j,x} \in S_{a\mathcal{D}}$ then they will unavoidably force $s_i, s_j \in S_{p\mathcal{D}}$ to be distinguishable, since the output trace of s_i, $\theta_{O_{si}}$, will have the form $(\theta_{O_{si}} = < \ldots, (x, t_i), \ldots >) \neq (\theta_{O_{sj}} = < \ldots, (x, t_i + \Delta), \ldots >)$, where $\theta_{O_{sj}}$ is the output trace of s_j and $|\Delta| > 0 \wedge (t_i + \Delta) > 0$.

This lemma allows us to define a procedure to be used for *distinguishing* states in Extended MealyDEVS models.

4.1.1. Minimization procedure for Extended MealyDEVS models

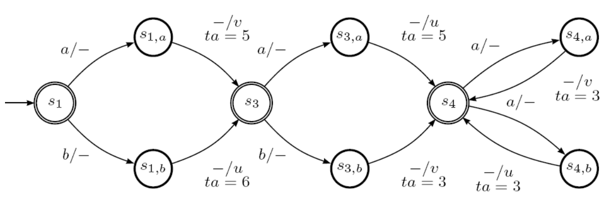
A state transition table can be used to represent the functions of an Extended MealyDEVS model, but in this case, we add, in the table, the value of the lifetime of the next active state (

Figure **3**). The minimization procedure defined in (Kohavi 1978) is extended in order to take into account the value of $ta(s_i)$ (lifetime of the state s_i).



State	x = a	x = b
s1	s2, 5, v	s3, 6, u
s2	s4, 5, u	s4, 3, v
s3	s4, 5, u	s4, 3, v
s4	s4, 3, v	s4, 3, u

Figure 3: An Extended MealyDEVS model with its associated transition table.



State	x = a	x = b
s1	s3, 5, v	s3, 6, u
s3	s4, 5, u	s4, 3, v
s4	s4, 3, v	s4, 3, u

Figure 4: Resulting minimal Extended MealyDEVS model (c) after the deletion of s_2 (which is equivalent to s_3), $s_{2,a}$ and s_{2_b}, and the resulting transition table (d).

4.2. Fault detection techniques on Extended MealyDEVS models

4.2.1. Experiments on Extended MealyDEVS models

We introduce the concept of a timed *relative* input sequence in order to refer not to the absolute time of each event, but to its relative time with respect to the previous input event. Given a timed input sequence $\pi = \langle (x_0,t_0),(x_1,t_1),\ldots,(x_n,t_n)\rangle$, we refer not to π, but to *rel*(π), where the function *rel* is defined as follows:

$$rel(\langle (x_0,t_0),(x_1,t_1),\ldots,(x_n,t_n)\rangle) = \langle (x_0,t_0),(x_1,t_1-t_0),\ldots,(x_n,t_n-t_{n-1})\rangle$$

It is easy to show that *rel* is bijective, so it is equivalent to talk either about π or about *rel*(π) as we can easily define $rel^{-1}(\pi)$. Then, we say that a timed relative input sequence π is *slow* on an Extended MealyDEVS model *D* iff $rel^{-1}(\pi)$ is slow on *D*.

With the previous remark, we will define the experiments for Extended MealyDEVS models:

Definition 4.4 (Preset Experiment) *A timed relative input sequence* $\pi \in (X_{\mathcal{D}} \times \Re_0^+)^*$ *of the form* $\langle (x_0,t_0),..,(x_n,t_n)\rangle$ *defines a preset experiment when an Extended MealyDEVS model D receives it as input. The sequence of output events (the output trace) that D generates in response to* π *is the result of the experiment.*

For example, the following sequence constitutes an experiment when inputted:

$$\pi = \langle (a,0)(b,2)(c,1)\rangle.$$

This experiment is to be interpreted as follows:

> Issue input a; wait 0 units of time; issue input b; wait 2 units of time; issue c; wait 1 unit of time; collect the observed output.

It is straightforward that not all sequences that belong to $(X_{\mathcal{D}} \times \Re_0^+)^*$ can be applied to a given Extended MealyDEVS model. The subset of sequences that can be accepted constitutes the set of valid experiments:

Definition 4.5 (Valid Preset Experiment) *A timed relative input sequence* $\pi \in (X_{\mathcal{D}} \times \Re_0^+)^*$ *of the form* $\langle (x_0,t_0),..,(x_n,t_n)\rangle$ *defines a valid preset experiment when an Extended MealyDEVS model D receives it as input iff* $\forall i = 0..n \bullet t_i > k = \max \{ta(s) \mid s \in S_{a\mathcal{D}}\}$.

That is to say, a timed input sequence will define a valid experiment if and only if it is slow on *D*.

An *adaptive experiment* is defined as follow:

Definition 4.6 (Adaptive Experiment) *A timed relative input sequence* $\pi \in (X_{\mathcal{D}} \times \Re_0^+)^*$ *of the form* $\langle (x_0,t_0),..,(x_n,t_n)\rangle$ *defines an adaptive experiment when an Extended MealyDEVS model D receives it as input, and provided that there exists a function*

$$f: \ Seq(X_{\mathcal{D}}) \times Y_{\mathcal{D}} \to X_{\mathcal{D}}$$

such that $\forall i = 2..n \bullet \exists y_i \in Y_{\mathcal{D}} \mid x_i = f(\langle x_0,..,x_{i-1}\rangle, y_i)$

That is to say that the value of the i^{th} *input event in* π *depends on all the previous input events and on an (unspecified) output event. The sequence of output events (the output trace) that D generates in response to* π *is the result of the experiment.*

Definition 4.7 (Valid Adaptive Experiment) *An adaptive experiment which consists of inputting the sequence* $\pi = \langle (x_0,t_0),..,(x_n,t_n)\rangle$ *into an Extended MealyDEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *is considered to be valid if it is slow on D and its function f satisfies the following property:*

- $x_1 = f(\langle x_0\rangle, y_0) \Leftrightarrow y_0 = \lambda_{\mathcal{D}}(\delta_{ext\mathcal{D}}(s_0, e, x_0))$
- $x_i = f(\langle x_0,..,x_{i-1}\rangle, y_{i-1}) \Leftrightarrow$ $y_{i-1} = \lambda_{\mathcal{D}}(\delta_{ext\mathcal{D}}(f(\langle x_0,..,x_{i-2}\rangle, y_{i-2}), e, x_{i-1}))$

That is to say that the value of the i^{th} input event in π depends on all the previous input events and the last output event that the model generated. This is equal to saying that the i^{th} input event depends on all the output events that the Extended MealyDEVS model has generated so far.

Depending on the results obtained at the end of an experiment, it can be classified into different categories: let *state* be the function defined as:

$$state: \mathbb{N}_0 \to S_{p\mathcal{D}}$$
$$state(0) = s_0$$
$$state(n) = \delta_{int\mathcal{D}}(delta_{ext\mathcal{D}}(state(n-1), e, x_{n-1})) \quad \forall n > 0$$

then the following categories of experiments are defined:

Definition 4.8 (Distinguishing Experiment) *A valid experiment which consists of inputting the timed relative sequence* $\pi = \langle (x_0,t_0),..,(x_n,t_n)\rangle$ *into an*

Extended MealyDEVS model $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *, and whose result is the output sequence* υ*, is distinguishing if there exists an injective function* $first : Seq(X_{\mathcal{D}}) \times Seq(Y_{\mathcal{D}}) \rightarrow S_{p\mathcal{D}}$ *such that first(*π,υ*) = state(0)* $\forall \upsilon$.

Definition 4.9 (Homing Experiment) *A valid experiment which consists of inputting the timed relative sequence* $\pi = \langle (x_0, t_0), .., (x_n, t_n) \rangle$ *into an Extended MealyDEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$*, and whose result is the output sequence* υ*, is homing if there exists an injective function* $last : Seq(X_{\mathcal{D}}) \times Seq(Y_{\mathcal{D}}) \rightarrow S_{p\mathcal{D}}$ *such that last(*π,υ*) = state(n+1)* $\forall \upsilon$.

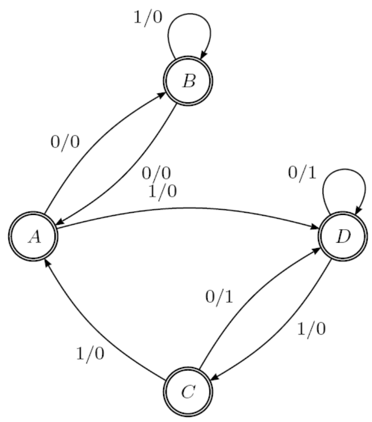
Definition 4.10 (Synchronizing Experiment) *A valid experiment which consists of inputting the timed relative sequence* $\pi = \langle (x_0, t_0), .., (x_n, t_n) \rangle$ *into an Extended MealyDEVS model* $\mathcal{D} = (X_{\mathcal{D}}, Y_{\mathcal{D}}, S_{\mathcal{D}}, \delta_{ext\mathcal{D}}, \delta_{int\mathcal{D}}, \lambda_{\mathcal{D}}, ta_{\mathcal{D}})$ *, and whose result is the output sequence* υ*, is synchronizing for state* $s_k \in S_{p\mathcal{D}}$ *if there exists a function* $last : Seq(X_{\mathcal{D}}) \times Seq(Y_{\mathcal{D}}) \rightarrow S_{p\mathcal{D}}$ *such that last(*π,υ*)* $= s_k \;\; \forall \upsilon$.

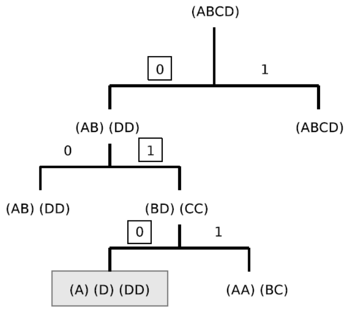
4.2.2. Sequence Finding

In order to perform a *homing experiment* for an Extended MealyDEVS model, the procedure described in (Kohavi 1978) can be utilized in a straightforward way, by considering the uncertainty of the model to be composed of all passive states in it.

By applying the algorithm given in (Kohavi 1978), a preset homing sequence can be designed for minimal Extended MealyDEVS models by adjoining, to each input event, a value $t_k > t = max \;\; \{ta(s) \;\; | \, s \in S_{a\mathcal{D}}\}$. Thus, we extend the requirements on the models to be tested (Kohavi 1978, Lee and Yannakakis 1996) so as to have *t* as a given value.

As an example, we give a Mealy machine *M*, its associated homing tree, and the resulting homing sequence for this machine (Figure 5). Secondly, we give (Figure 6) an Extended MealyDEVS model (which has a delay time-abstracting bisimulation relation with *M)*, and the resulting homing sequence (note that $k = max\{ta_{\mathcal{D}}(s_{a_i}) \,|\, s_{a_i} \in S_{a\mathcal{D}}\} = 9$. Then, every waiting value after an input event should be > 9).

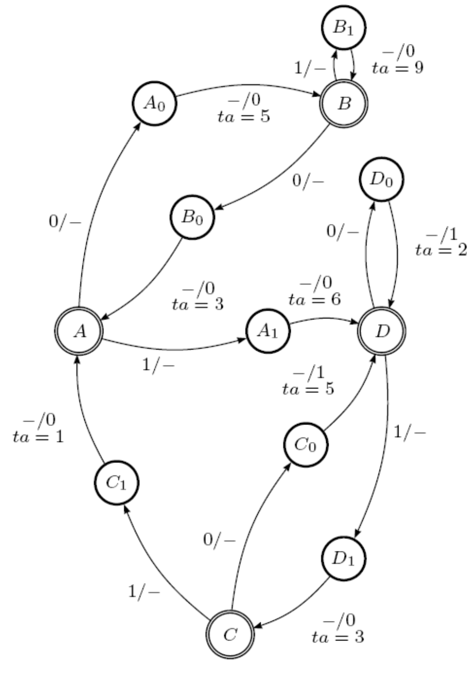


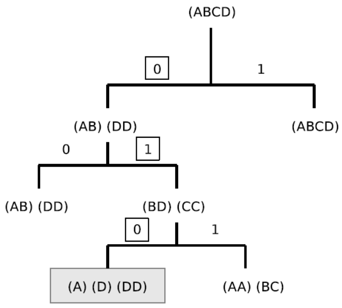


State	x = 0	x = 1
A	B, 0	D, 0
B	A, 0	B.0
C	D, 1	A, 0
D	D, 1	C, 0

$\pi = \langle 0\,1\,0 \rangle$

Figure 5: A Mealy machine with its homing tree, associated transition table, and a homing sequence π for it.





State	$x = 0$	$x = 1$
A	$B, 0.5$	$D, 0.6$
B	$A, 0.3$	$B, 0, 9$
C	$D, 1.5$	$A, 0.1$
D	$D, 1.2$	$C, 0.3$

$\pi = \langle(0, 9.2)\ (1, 10)\ (0, 12)\rangle$

Figure 6: Extended MealyDEVS model derived from the one in Figure 5 with its homing tree, associated transition table, and one possible homing sequence (π) for it.

Given the previous considerations on the timed nature of sequences, both *distinguishing* and *synchronizing* sequences can be obtained by using the methods and algorithms described in (Kohavi 1978, Lee and Yannakakis 1996) and afterwards adjoining the time components as was previously described in this section. It is straightforward to prove that all the properties and theorems are valid for Extended MealyDEVS models. In particular, the following results are valid also for Extended MealyDEVS models:

Theorem 4.2: *A preset homing sequence, whose length is at most* $(n-1)^2$, *exists for every minimal Extended MealyDEVS model* $\mathcal{D}$ *, where* n *is the number of passive states in* $\mathcal{D}$.
Theorem 4.3: *If there exists a synchronizing sequence for an Extended MealyDEVS model D that has n passive states, then its length is at most* $(n-1)^2 n/2$.

Proof: *See (Kohavi 1978).*

The following result sums up the preceding discussion:

Theorem 4.4: *Let* $\pi = \langle x_1 x_2 .. x_n \rangle$ *be either a synchronizing, homing, preset distinguishing or adaptive distinguishing sequence for a Mealy Machine M. Then the sequence* $\upsilon = \langle (x_1, t_1)(x_2, t_2)..(x_n, t_n) \rangle$ *obtained from* π *is, respectively, either a synchronizing, homing, preset distinguishing or adaptive distinguishing sequence for the Extended MealyDEVS model D obtained from M if* $t = max \{ta_{\mathcal{D}}(p) | p \in S_{a\mathcal{D}}\} < t_i, \forall i = 1..n$.

4.2.3. Testing procedure

In order to be able to represent the behaviour defined by the semantics of a timed sequence, it is necessary to model a *tester* (Krichen and Tripakis 2005), that is, a DEVS model that emits the events (at a specified time) needed to test a given Extended MealyDEVS model.

The general scheme of the coupling between a tester and the model to be tested is:

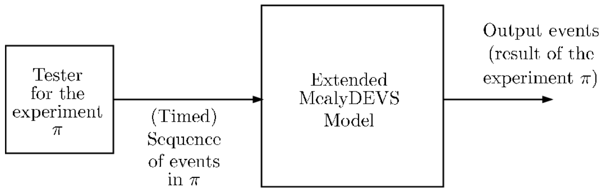


Figure 7: Coupling scheme of an Extended MealyDEVS model and a valid tester for it.

And the tester for a given PX is defined as follows:

Definition 4.11 (DEVS tester model) *Given a PX* $\pi = (x_0, t_0), (x_1, t_1), \ldots, (x_n, t_n)$ *, its associated tester is the DEVS model* $\mathcal{T} =< X_{\mathcal{T}}, Y_{\mathcal{T}}, S_{\mathcal{T}}, \delta_{int_{\mathcal{T}}}, \delta_{ext_{\mathcal{T}}}, \lambda_{\mathcal{T}}, ta_{\mathcal{T}} >$ *, where:*

- $X_{\mathcal{T}} = \{Reset\}$ *(Event that restores the tester to its initial state)*
- $Y_{\mathcal{T}} = x_0, x_1, \ldots, x_n$
- $S_{\mathcal{T}} = s_0, s_1, \ldots, s_n \cup s_{STOP}$
- $\delta_{ext_{\mathcal{T}}}(s_i, e, x) = s_0$ *(Restores the tester to its initial state)*
- $\delta_{int_{\mathcal{T}}}(s_i) = \begin{cases} s_{i+1} & if\ i < n \\ s_{STOP} & if\ i = n \vee s_i = s_{STOP} \end{cases}$
- $ta_{\mathcal{T}}(s_i) = \begin{cases} 0 & if\ i = 0 \\ t_{i-1} & if\ i = 1..n \wedge s_i \neq s_{STOP} \\ \infty & if\ s_i = s_{STOP} \end{cases}$
- $\lambda_{\mathcal{T}}(s_i) = \begin{cases} x_i & if\ s_i \neq s_{STOP} \\ \varnothing & if\ s_i = s_{STOP} \end{cases}$

  (The case $\lambda_{\mathcal{T}}(s_i) = \varnothing$ *never happens, as* $ta_T(s_{STOP}) = \infty$*)*

As an example, the tester that implements the experiment $\pi = (a,0)(b,2)(c,1)$ is given in figure 8.

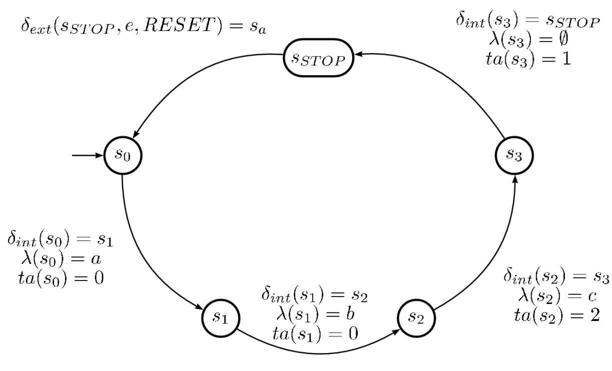


Figure 8: Tester for the preset experiment (PX) $\pi = (a,0)(b,2)(c,1)$.

If we take into account the fact that after the tester sends the output event x_i, the Extended MealyDEVS model under test will transition up to the passive state s_{i+1}, then it is assured that the Extended MealyDEVS model will remain in the same state until the tester issues the output event x_{i+1} (in case it exists).

In order to implement a valid adaptive experiment for an Extended MealyDEVS model, the basic idea is to define a tester which represents the decision tree associated with the adaptive experiment (Kohavi 1978). It is then required to have one active state for each node in the tree (that is, one node for each possible uncertainty in the tree). Each such state will output the event that the Extended MealyDEVS model under test is going to receive in order to solve the uncertainty. Additionally, each of these states will (internally) transition to a passive state that represents the response of the model under test. That is, it will be able to receive any of the possible output events that the Extended MealyDEVS model will generate (in order to do this, the output of the tested model needs to be connected to the tester's input). Depending on the value of the received event, the tester will transition to one of the active states that represent the consequential uncertainties (the nodes one that are one level lower). Finally, all the leaf nodes have to be represented as passive states which only accept the RESET event in order to reinitialize the experiment. The following figure shows a concrete example of an adaptive tester:

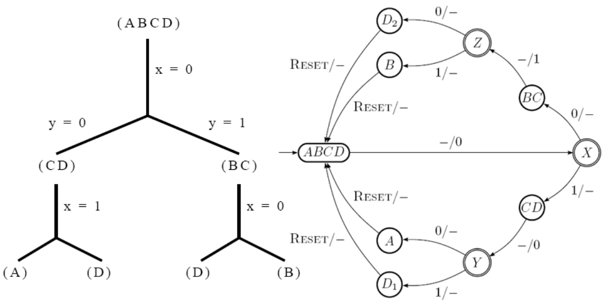


Figure 9: Sample adaptive experiment and corresponding DEVS tester model.

5. CONCLUSION

In this paper, we have proposed a first approach, on a subset of DEVS models, on which we can apply extensions of fault detection techniques from Mealy machines.

For the given subset of DEVS models to which the theory of black-box testing can be applied, we have proposed:

- a proper extension and formalization of the concepts of homing, distinguishing and synchronizing sequences,
- the preset and adaptive experiments that the three previously mentioned types of sequences allow for.

Finally, we have given the procedure and structure of the DEVS models that implement the testing procedure for both preset and adaptive fault detection experiments.

Further lines of work involve expanding the proposed approach to more general DEVS and to define the limitations of these techniques in a timed formalism as DEVS in order to clearly define the broadest possible subset of DEVS models which can be black-box tested.

REFERENCES

Dacharry, H. P. and Giambiasi, N., 2005. Formal verification with timed automata and DEVS models: a case study. *ASSE 2005 Simposio Argentino de Ingeniería de Software - 34 JAAIO Jornadas Argentinas de Informatica e Investigacion Operativa*, pp. 251–265. August 24 – September 2, Rosario (Santa Fe, Argentina).

Dacharry, H.P. and Giambiasi, N., 2007. DEVS based timed hierarchy of formalisms. *Proceedings of the International Modeling and Simulation Multiconference 2007 (IMSM07)*. SCS -- The Society for Modeling and Simulation International. February 8-10. Buenos Aires (Argentina).

Giambiasi, N., Dacharry, H.P., 2007. DEVS and Timed Automata for the Design of Control Systems. *Robotics Research Trends*, Xing P. Guo (Ed.), Nova Science Publishers, ch. 5, pp. 193-222.

Kohavi, Z., 1978. *Switching and Finite Automata Theory: Computer Science Series*. McGraw-Hill Higher Education.

Krichen, M. and Tripakis, S., 2005. State identification problems for timed automata. In *The 17th IFIP Intl. Conf. on Testing of Communicating Systems (TestCom'05)*, volume 3502 of *LNCS*. Springer.

Lee, D. and Yannakakis M., 1996. Principles and methods of testing finite state machines - A survey. *Proceedings of the IEEE*, volume 84, pages 1090–1126.

Lynch, N. A. and Vaandrager, F. W., 1993. Forward and backward simulations – part I: untimed systems. *Technical Report: CS-R9313*, page 35. Centrum voor Wiskunde en Informatica (CWI).

Lynch, N. A. and Vaandrager, F. W., 1993. Forward and backward simulations – part II: timing-based systems. *Technical Report: CS-R9314*, page 36. Centrum voor Wiskunde en Informatica (CWI).

Springintveld, J, Vaandrager, F. and D'Argenio, P. R., 2001. Testing timed automata. *Theoretical Computer Science*, 254(1–2):225–257.

Tripakis, S., and Yovine, S., 2001. Analysis of timed systems using time-abstracting bisimulations. *Formal Methods in System Design*, 18(1):25–68.

Zeigler B. P., Praehofer, H. and Kim, T. G., 2000. *Theory of Modeling and Simulation, Second Edition*. Academic Press.