A relative identification method for reactive systems

Clément Galetta, Jean-Marc Roussel, Jean-Marc Faure

A relative identification method for reactive systems

Clement Galetta * **Jean-Marc Roussel** * **Jean-Marc Faure** **

* *Automated Production Research Laboratory (LURPA), ENS Cachan, Univ. Paris-Sud, Université Paris-Saclay, 94235 Cachan, France (e-mail: firstname.lastname@ens-cachan.fr).*
** *Automated Production Research Laboratory (LURPA), ENS Cachan, Univ. Paris-Sud, Supmeca, Université Paris-Saclay, 94235 Cachan, France (e-mail: firstname.lastname@ens-cachan.fr).*

Abstract: This paper proposes a method to build an identified model from an observed sequence of input/output vectors and a model of knowledge on the system to identify. This method comprises two steps. The first one starts by checking whether the knowledge model is compatible with the observed sequence; when this is the case, the identified model is built automatically. From this result, the observed and unobserved behaviors of the knowledge model are then explicitly determined, in a second step.

Keywords: System identification, Signal Interpreted Petri Nets, Boolean functions, Reachability graph, Programmable Logic Controller

1. INTRODUCTION

The overall objective of identification methods is to build a behavioral model of a system from results of experiments on the real system. This work considers reactive systems where a Programmable Logic Controller (PLC) controls a plant; this class of system can be modeled as a closed-loop Discrete Event System (DES) where the inputs and outputs of the controller are logic variables.

The first results in the domain of identification of DES (Gold (1967), Angluin (1987)) were mainly related to language identication. More recently, construction by identification of models in the form of automata or Petri nets (PN) has been addressed by several authors. Roth et al. (2010) and Schneider et al. (2012), for instance, propose methods to build automata that are used later on for diagnosis purposes. Meda-Campaña and López-Mellado (2005), Giua and Seatzu (2005), Dotoli et al. (2011), Estrada-Vargas et al. (2015), Saives et al. (2015), to name a few, focus on identification based on Petri nets. The latter two references place their works in the context of reverse engineering, i.e. construction of a model that is understandable by experts to revamp an existing machine; this is also the applicative context of this work. A good survey on DES identication can be found in Estrada-Vargas et al. (2010). This survey underlines in particular that an identification method may be absolute (or black-box), when no a priori knowledge about the system behavior is available before identification, or relative otherwise.

Absolute identification is surely more attractive because it is only based on observations of the evolutions of the real system. Nevertheless, it must be noted that the quality of the identified model depends on the observation duration. A good identified model may require a long observation time; this may hinder the application of the method and explains why some researchers have considered a relative identification approach. When the identified model is represented by a PN, Giua and Seatzu (2005), for instance, proposes an algorithm for identication of free-labeled PN that assumes that the number of places is known and the observed language is complete. This latter assumption is not made in Dotoli et al. (2011), where unobservable transitions are added to a labeled PN whose observable behavior is known. Nevertheless, the knowledge models that are proposed in these valuable works are not completely suitable for the class of reactive systems that we focus on because the inputs and outputs of the controller are not explicitly considered.

Hence, this paper proposes a relative identification method (Fig. 1) which starts from both a knowledge model generated by an expert and observed input/output sequences. The knowledge model is represented in the form of a Signal Interpreted Petri Net (SIPN), an appropriate formalism for the class of reactive systems that is considered in this work, and the I/O sequences have been obtained experimentally previously. From these data, the method determines first whether the knowledge model is compatible with the observations, i.e., for every observed sequence, there exists in the model a firing sequence which can generate the same output vectors from the input vectors. When this is the case, the identified model is provided in the form of an augmented reachability graph, a model where input and output combinations are associated to the transitions and markings firings. Moreover, it is possible to derive from this graph the unobserved part of the knowledge model; these information can be the starting point for further active identification, an identification approach where input values are to be forced during the experiments.

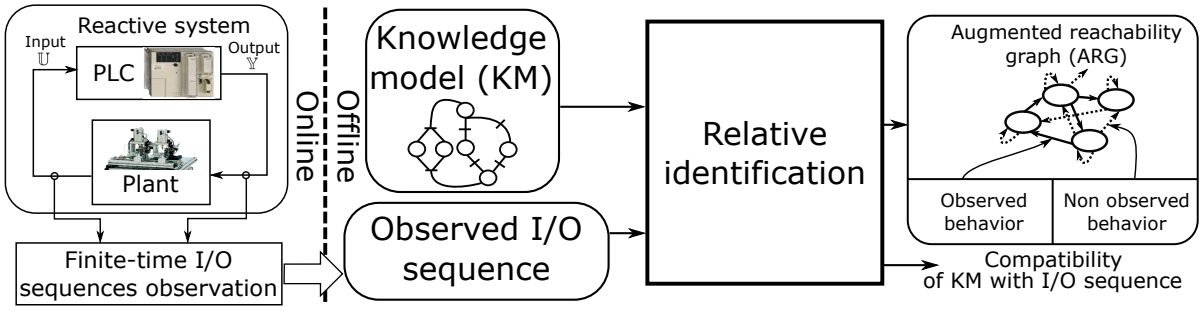

Fig. 1. Objective of the work

The formalism that was selected in this work to describe the knowledge model is presented in the next section. The proposed relative identification method is detailed and illustrated on the basis of small examples at Section 3. Section 4 is dedicated to an example and concluding remarks as well as perspectives for further work are given in section 5.

2. FORMALISM

SIPN have been introduced in Minas and Frey (2002). This formalism is particularly appropriate to describe the evolutions of an interpreted PN according to those of input and output signals. The notations and assumptions that are used in this work are presented below.

2.1 Basics of Petri Nets

An ordinary Petri net structure G is a bipartite digraph represented by the 4-tuple $G = (P, T, Pre, Post)$ where: $P = \{p_1, p_2, ..., p_{|P|}\}$ [1] and $T = \{t_1, t_2, ..., t_{|T|}\}$ are finite sets of vertices named places and transitions respectively; $Pre(Post) : P \times T(T \times P) \rightarrow \{0, 1\}$ is a function representing the edges going from places to transitions (from transitions to places).

A marking function $M_g : P \rightarrow \mathbb{N}$ represents the number of tokens residing inside each place where $\mathbb{N}$ is the set of positive integers; it is usually expressed as a $|P|$-entry vector. The net is said *1-bounded* (or *safe*) when $\mathbb{N}$ is replaced by $\{0, 1\}$, i.e there is at most one token residing in every place.

The incidence matrix of G is $\mathbf{W} = \mathbf{Post} - \mathbf{Pre}$, where $\mathbf{Pre} = [pre_{ij}]$; $pre_{ij} = Pre(p_i, t_j)$; and $\mathbf{Post} = [post_{ij}]$; $post_{ij} = Post(p_i, t_j)$ are the pre-incidence and post-incidence matrices respectively. A Petri net (PN) with the given initial marking is denoted by is the pair $N = (G, M_0)$, where G is a PN structure and M_0 an initial marking. More details on the basics of Petri nets can be found in (Murata, 1989). In the case of this study where a control model is considered, every PN will be safe.

2.2 SIPN

A Boolean algebra is defined on $\mathbb{B} = \{0, 1\}$ with $(\wedge, \vee, \neg)$ the operators of conjunction, disjunction and negation respectively.

[1] The cardinality of a set A is noted $|A|$

A Boolean function F is defined as:

$$F : \mathbb{B}^n \longrightarrow \mathbb{B}$$
$$X \longmapsto F(X)$$

$\mathbb{B}^n$ is the set of the n Boolean variables; if $X \in \mathbb{B}^n$, $X = (x_1, ..., x_k, ..., x_n)$. $\mathcal{F}_n$ is the set of Boolean functions that can be defined on $\mathbb{B}^n$. The cardinality of this set is equal to 2^{2^n}.

Definition 1. A Signal Interpreted Petri Net system (SIPN) is a 6-uplet KM= $(G, M_0, \mathbb{U}, \mathbb{Y}, \lambda, \mu)$ with (G, M_0) a Petri net to which are added:

- $\mathbb{U}$ the set of Boolean input variables.
- $\mathbb{Y}$ the set of Boolean output variables.
- $\lambda : T \longrightarrow \mathcal{F}_{|\mathbb{U}|}$ the function of firing conditions $t_j \longmapsto F_{t_j}$ where F_{t_j} is a Boolean function depicting sufficient conditions on the values of the inputs to fire t_j.
- $\mu : P \rightarrow \mathcal{P}(\mathbb{Y})$ the function of outputs activation of places where $\mathcal{P}(\mathbb{Y})$ is the power set of $\mathbb{Y}$. A subset of outputs to activate (set to 1) when the place is marked is associated to every place.

Example 1. An example of SIPN with 3 inputs and 4 outputs such as $\mathbb{U} = \{a, b, c\}$ and $\mathbb{Y} = \{A, B, C, D\}$ is shown in Fig. 2.

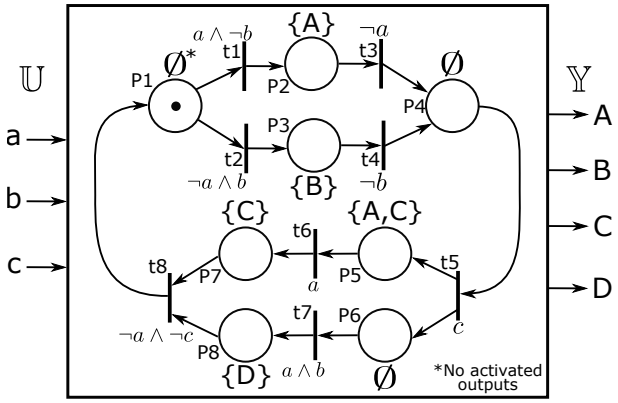

Fig. 2. Example of SIPN

Two transitions of the net are in *structural conflict* when they share the same upstream place. As we are considering a control model, effective conflicts must not be possible to avoid non-determinism, however. This implies that the functions associated to transitions in structural conflict must be exclusive. This can be formalized as follows.

Definition 2. A structural conflict for a given place p_i exists if $|D(p_i)| > 1$

where $D(p_i)$ is the function which returns the set of downstream transitions of a given place p_i :

$$D(p_i) = \{t_j | Pre(p_i, t_j) = 1\} \quad (1)$$

Definition 3. A SIPN is deterministic iff:

$\forall p_i \in P$ such that $|D(p_i)| > 1, \forall X \in \mathbb{B}^{|\mathbb{U}|}, \forall (t_j, t_{j'}) \in D^2(p_i)$

$$F_{t_j}(X) \wedge F_{t_{j'}}(X) = 0 \quad (2)$$

The SIPN of the example of Fig. 2 is deterministic: t_1 and t_2 are in structural conflict but their associated functions are exclusive. Only deterministic SIPN will be considered in what follows.

The evolution rules of a SIPN are the following ones:

- A transition t_j is enabled at marking M_g if and only if all its upstream places are marked, i.e $\forall p_i \in P, M_g(p_i) \geq Pre(p_i, t_j)$. The set of enabled transitions for a given marking M_g is noted $T_e(M_g)$.
- A transition t_j is fired immediately when its associated function is True: $F_{t_j}(X) = 1$. The firing of t_j leads to a new marking M_{g+1}, written $M_g \xrightarrow{t_j} M_{g+1}$. The new marking is computed as $M_{g+1} = M_g + \mathbf{W}.s_g$ where $s_g(j) = 1; s_g(i) = 0$ with $i \neq j$.
- All transitions that are simultaneously firable are fired simultaneously: $M_g \xrightarrow{T_f} M_{g+1}$, where T_f is the set of the simultaneously firable transitions.

To illustrate the last point, in Fig. 2, if P_5 and P_6 are both marked with $b = 1$, then $a = 1$ leads to the simultaneous firing of the transitions t_7 and t_6.

Hence, a firing sequence $\sigma = t_1(t_2, t_3)t_4...t_j$ that leads from M_0 to M_g, i.e $M_0 \xrightarrow{\sigma} M_g$, may include firings of only one transition (t_1, t_4 in σ) as well as simultaneous firings of several transitions (t_2 and t_3 in σ).

Last, the value of an output y_l for a marking M_g is :

$$y_l[M_g] = \begin{cases} 1 & \text{if } \exists p_i \in P | [M_g(p_i) = 1] \wedge [y_l \in \mu(p_i)] \\ 0 & \text{else} \end{cases}$$

For a given marking M_g, the set of the output values is given by the function gamma:

$$\gamma : \mathbb{B}^{|P|} \longrightarrow \mathbb{B}^{|\mathbb{Y}|}$$
$$M_g \longmapsto \gamma(M_g)$$

where:

$$\gamma(M_g) = (y_1[M_g], ..., y_l[M_g], ..., y_{|\mathbb{Y}|}[M_g]) \quad (3)$$

The following two properties can then be stated.

Property 1. There exists a unique set of output values $\gamma(M_g)$ associated to a given marking M_g.

All the output values of the SIPN of Fig. 2 are equal to 0 for the initial marking, for instance.

Property 2. Several markings can be associated to a given set of output values.

Either P_1 or P_4 is the only marked place when all the output values are equal to 0, for instance. The current marking cannot be identified unambiguously from the observation of the outputs.

3. PROPOSED METHOD

3.1 Principle

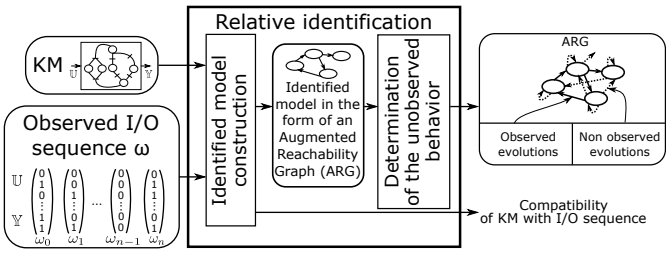

Fig. 3. Overview of the proposed method

The relative identification method that is proposed (Fig.3) starts from a knowledge model designed by an expert and modelled by a SIPN, noted KM, and a sequence ω of observed I/O vectors. When several sequences have been observed on-line, the method must be applied to every sequence. Every vector ω_k of ω includes an input vector $\hat{u_k}$ and an output vector $\hat{y_k}$ such as: $\omega_k = [\hat{u_k}, \hat{y_k}]^T$. The length of ω is noted $|\omega|$.

The aim is to determine:

(1) whether KM is compatible with ω;
(2) the identified model, in the form of an augmented reachability graph;
(3) the part of the knowledge model which has not been observed, from the previous result.

The first two results are obtained at the end of the first phase; the third one at the end of the second phase of the method.

3.2 Construction of the identified model

The model proposed by the expert must be obviously compatible with the observed I/O sequence, i.e. there must exist in this model a firing sequence which can generate the same output vectors sequence from the observed input vectors sequence. Moreover, when this firing sequence exists, the evolutions of the system can be represented by a formal model which is derived from both KM and ω. The final aim of the algorithm which is detailed at Appendix A and sketched at Fig. 4 is to build this model. When this aim cannot be reached, KM is declared inconsistent with the observed I/O sequence. This algorithm comprises three steps:

(1) Initial State Detection (ISD): the first output vector $\hat{y_0}$ of ω does not correspond necessarily to the output values for the initial marking [2] . The aim of this step is therefore to find an observed vector $\hat{y}_{IS}$ that matches with these values; KM and ω are not compatible when $\hat{y}_{IS}$ does not exist. However, according to property 2, a given set of output values does not define necessarily a unique marking. This explains why the following two analyses are to be performed.

[2] The reasoning assumes that the initial marking has been observed. If this is not the case, the algorithm can be applied by starting from a different marking.

(2) Forward Analysis (FA): its objective is to check that every I/O change, from ω_{IS} to ω_n, can be explained from the SIPN:
   - When only the input vector changes, either no transition which is enabled for the current marking must be firable for the observed values of the inputs, or a transition (or a set of simultaneously firable transitions) which is firable for these values is fired and the output values associated to the source and destination markings are the same.
   - When both the input and output vectors change, at least one transition must be fired for the observed values of the inputs and the observed values of the outputs must correspond to those of the marking which is reached by firing this (these) transition(s).

   When the result of this analysis is negative a new $\hat{y}_{IS}$ is searched in ω.

(3) Backward Analysis (BA): this analysis is similar to the FA but the exploration of ω is made from ω_{IS} to ω_0.

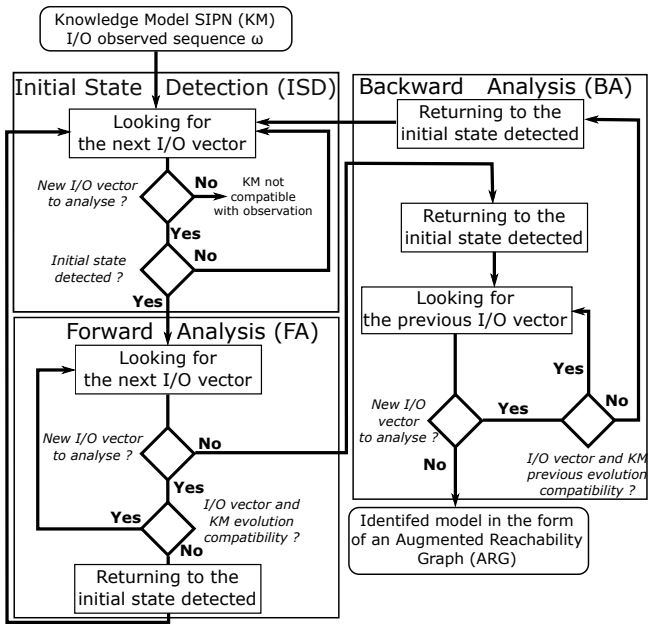

Fig. 4. Identified model construction

An Augmented Reachability Graph (ARG) is constructed during the exploration of the I/O sequence. This kind of graph is similar to the Stable Location Automaton introduced in Provost et al. (2011) and used to represent formally the behavior of a Grafcet model. An ARG is a pair $ARG = (RM, E)$ with RM the set of vertices and E the set of edges. A vertex v is a 4-tuple $v = (M_g, \gamma(M_g), T_e(M_g), I_s)$ with M_g a marking of the SIPN, $\gamma(M_g)$ the output values for this marking, $T_e(M_g)$ the set of transitions which are enabled for this marking and I_s the set of input vectors which have been observed and did *not provoke* a marking change. An edge $e \in E$ with $e = (M_g, M_{g'}, T_f, I_f)$ represents $M_g \xrightarrow{T_f} M_{g'}$ where T_f is the set of simultaneously fired transitions and I_f the set of input vectors which *provoke* these firings. It must be underlined that, even if some elements of an ARG $(M_g, \gamma(M_g), T_e(M_g))$ are obtained only from the SIPN, T_f, I_s and I_f are yielded by the exploration of ω.

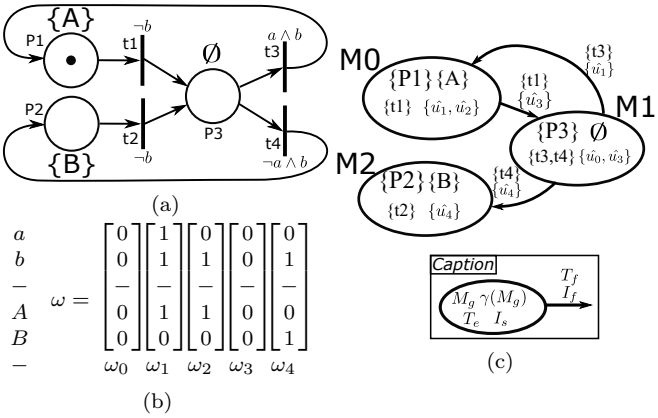

Fig. 5. Illustration of the example 2: SIPN (a), observed I/O sequence (b), ARG built (c)

Example 2. This algorithm is exemplified on the model of Fig. 5a where the SIPN owns 2 inputs $\mathbb{U} = \{a, b\}$ and 2 outputs $\mathbb{Y} = \{A, B\}$. The obtained ARG is depicted at Fig. 5c. As the PN is safe and the outputs are Boolean variables, M_g and $\gamma(M_g)$ are represented in this figure as sets of marked places and True variables, instead of vectors, in the sake of brevity.

ISD starts from $\omega_0 = [\hat{u_0}, \hat{y_0}]^T$. The output vector $\hat{y_0} = [0, 0]^T$ does not match with the output vector associated to the initial marking $\gamma(M_0) = [1, 0]^T$. Hence, ω_1 is then analyzed. The output vector corresponds in this case to that of the initial marking; the input vector $\hat{u_1} = [1, 1]^T$ is stored in the set I_s for the initial marking and ISD stops.

FA starts with $\omega_2 = [\hat{u_2}, \hat{y_2}]^T$ whose output vector is identical to that of ω_1. This means that the only enabled transition from M_0 (t_1) cannot be fired for the observed input combination $\hat{u_2}$, what is consistent with the knowledge model. The input vector $\hat{u_2} = [0, 1]^T$ is stored in the set I_s for the initial marking then $\omega_3 = [\hat{u_3}, \hat{y_3}]^T$ is analyzed. The observed output vector $\hat{y_3} = [0, 0]^T$ means that a new marking M_1 has been reached; $\hat{u_3} = [0, 0]^T$ is stored in the set I_f for the edge from M_0 to M_1, as well as in the set I_s for M_1. $\omega_4 = [\hat{u_4}, \hat{y_4}]^T$ is at last analyzed. This I/O vector corresponds to a marking change from M_1 to M_2; the appropriate sets I_f and I_s are updated.

As the forward analysis is positive, backward analysis is performed from ω_1. The marking that corresponds to this observation (M_0) can be reached only by firing from M_1 the transition t_3 whose associated function is $a \wedge b$. The observed output vector $\hat{y_0} = [0, 0]^T$ and input vector $\hat{u_1} = [1, 1]^T$ show that KM is compatible with the observations. BA is positive.

3.3 Determination of the unobserved behavior

As it is based on observations, the ARG which has been built by the previous algorithm does not include necessarily every marking and marking change; the change from M_2 to M_1 has not been observed, for instance in the example 2. In a similar way, all the input vectors which provoke/do not provoke a marking change have not been

necessarily observed. The aim of this section is to show how these missing information can be explicitly expressed.

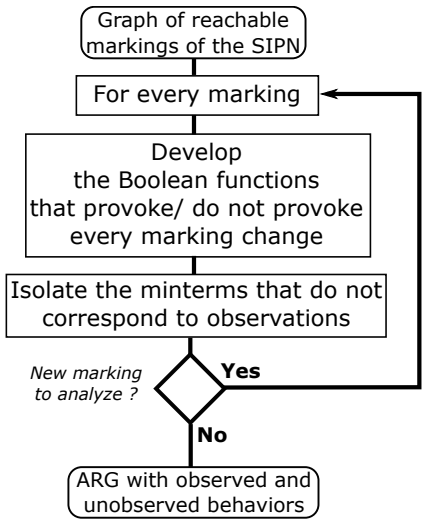

Fig. 6. Finding the unobserved behavior

The behavior of the knowledge model that has not been observed can be found by the algorithm that is detailed at Appendix B and sketched at Fig. 6. This algorithm relies on the development of Boolean functions in the form of disjunction of minterms. It is first reminded that a minterm is a logical expression of n Boolean variables that uses only the negation and conjunction operators. If $b = (b_1, ..., b_n)$ is an element of $\mathbb{B}^n$, a minterm defined from b is $min(b) = u'_1 \wedge u'_2 \wedge ... \wedge u'_n$, where $u'_i = u_i$ if $b_i = 1$, $u'_i = \neg u_i$ if $b_i = 0$.

The graph of reachable markings, starting point of the algorithm, must be constructed by considering that:

- Several enabled transitions of the SIPN may be simultaneously fired
- Functions on the inputs are associated to transitions.

If the set of enabled transitions for a marking M_g is noted $T_e(M_g)$, the set of possible firings from this marking, noted $\mathbb{E}$, is indeed the power set of $T_e(M_g)$: $\mathbb{E} = \mathcal{P}(T_e(M_g))$. The number of firings in this set is $|\mathbb{E}| = 2^{|T_e(M_g)|}$.

Example 3. If $T_e(M_g) = \{t_1, t_2, t_3\}$ then $\mathbb{E} = \{\emptyset, \{t_1\}, \{t_2\}, \{t_3\}, \{t_1, t_2\}, \{t_2, t_3\}, \{t_1, t_3\}, \{t_1, t_2, t_3\}\}$ and $|\mathbb{E}| = 8$ because $n = 3$.

In the graph of reachable markings, the edges that start from M_g correspond to the elements of $\mathbb{E}$. A Boolean function f_e, firing condition of a set of transitions, is associated to each edge e, with:

$\forall T_f \in \mathbb{E}$,

$$f_e = \bigwedge_{t_j \in T_f} F_{t_j} \bigwedge_{t_j \notin T_f} \neg F_{t_j} \quad (4)$$

This Boolean function may be sometimes equal to zero. If $T_e(M_g) = \{t_1, t_2\}$ with $F_{t_1} = a$ and $F_{t_2} = a \wedge b$, the boolean function f_e associated to the firing of only t_2 is : $f_e = F_{t_2} \wedge \neg F_{t_1} = (a \wedge b) \wedge \neg a = 0$. In the reachability graph of the SIPN as well as in the ARG, the corresponding edge is not represented because this element of $\mathbb{E}$ is not feasible.

However, a self-loop on M_g must be introduced in the ARG for the empty set of transitions $\emptyset$, to model that the marking remains the same for this element of $\mathbb{E}$. The following boolean function $f_{e'}$ is associated to this self-loop:

$$f_{e'} = \bigwedge_{t_j \notin T_f} \neg F_{t_j}$$

In the ARG that is provided by the algorithm, the observed behavior corresponds to the minterms such as the corresponding inputs combination has been observed, the unobserved behavior to the other minterms.

An edge e is *completely observed* if all minterms in the expression of f_e have been observed. A marking is completely observed if all edges starting from this marking, including the self-loop, have been completely observed.

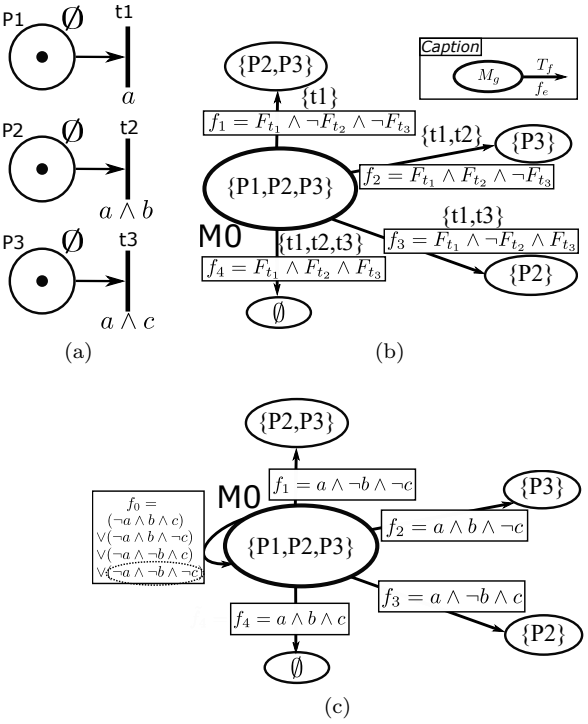

Fig. 7. KM SIPN (a), reachability graph of the SIPN (b), ARG with the observed (minterms circled with dotted line) and unobserved behaviors (minterms not circled) (c)

Example 4. The graph of reachable markings of the SIPN of Fig. 7a is given at Fig. 7b. Three transitions are enabled for M_0 but only four firings are possible because the functions associated to $\{t_2\}$, $\{t_3\}$, $\{t_2, t_3\}$ are equal to zero (when $T_f = \{t_2, t_3\}$, $f_e = (a \wedge b) \wedge (a \wedge c) \wedge \neg a = 0$ for instance). The Boolean functions associated to the four marking changes as well as the absence of marking change (self-loop on M_0) are shown on Fig. 7c; it may be noted that the eight minterms which can be defined on the set of the three inputs are present in the disjunction of the four functions. If, for instance, the only observed vector is $\omega_0 = [0, 0, 0|0, 0, 0]^T$, the observed behavior corresponds to the self-loop on M_0 with the only minterm of f_0 that is circled with dotted line ($\neg a \wedge \neg b \wedge \neg c$); the unobserved behavior to the other minterms.

4. EXAMPLE

This example is based on the knowledge model that is represented at Fig. 2 and the following observed I/O sequence:

$$\omega = \begin{array}{c} a \\ b \\ c \\ A \\ B \\ C \\ D \\ - \end{array} \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \\ \omega_0 & \omega_1 & \omega_2 & \omega_3 & \omega_4 & \omega_5 & \omega_6 & \omega_7 & \omega_8 & \omega_9 & \omega_{10} & \omega_{11} & \omega_{12} & \omega_{13} & \omega_{14} \end{bmatrix}$$

4.1 Construction of the identified model

ISD starts from ω_0. The observed output vector $\hat{y}_0 = [0,0,0,0]^T$ matches with $\gamma(M_0)$, the output set associated to the initial marking of the SIPN. Therefore, ISD stops and FA starts from ω_1. However, the observed output vector $\hat{y}_1 = [1,0,1,0]^T$ means that either both P_2 and P_7 are marked or that P_5 is marked. As it not possible to fire transitions that lead from the initial marking to one of these markings [3], FA is stopped and ISD is relaunched.

ISD starts this time from ω_1 but the observed output vector does not match with $\gamma(M_0)$, from this observation to ω_4. The initial state is only detected for ω_5. Hence, ISD stops and FA starts from ω_6. This analysis shows easily that the observed vectors ω_6 to ω_{14} correspond respectively to an input change without marking change, the firing of t_2, three consecutive input changes without marking change, the firing of t_4, then successively those of t_5, t_6 and t_7. FA concludes positively.

The BA is then performed from ω_5. The observed vectors from ω_5 to ω_0 correspond respectively to the firing of t_8, the simultaneous firings of t_6 and t_7, two successive input changes without marking change and the firing of t_5. BA concludes positively; hence the knowledge model is compatible with the observed sequence.

The identified model that is built during these analyses is given at Fig. 8. It must be noted that the firing of some transitions (t_4, t_5) with the same value of the input vector ($\hat{u}_0 = \hat{u}_{11} = [0,0,0]^T$ for instance for t_4) has been observed two times and that several different values of the input vector that do not change a marking ($\hat{u}_8$, $\hat{u}_9$, $\hat{u}_{10}$ for M_1, for instance) have been observed, too.

4.2 Determination of the unobserved behavior

From the previous result and the knowledge of the Boolean functions associated to the transitions of the SIPN, the ARG of Fig. 9 can be obtained. In this model, the observed behavior corresponds to the minterms which are circled with a dotted line and the unobserved behavior to the minterms which are circled with a solid line. Only two markings will be somewhat detailed.

Two transitions are enabled for M_0: $T_e = \{t1, t2\}$. However, as F_{t_1} and F_{t_2} are exclusive, only one of them is possible for a given combination of the input values. No

[3] It is also possible to remark that none of the transitions that follow P_1 is firable for the current values of the inputs $\hat{u}_1 = [0,0,1]^T$

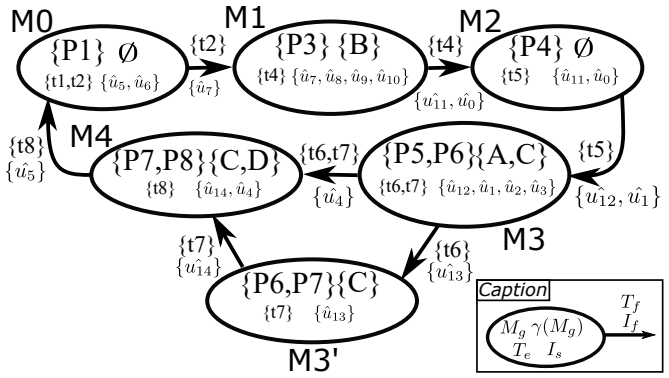

Fig. 8. Identified model

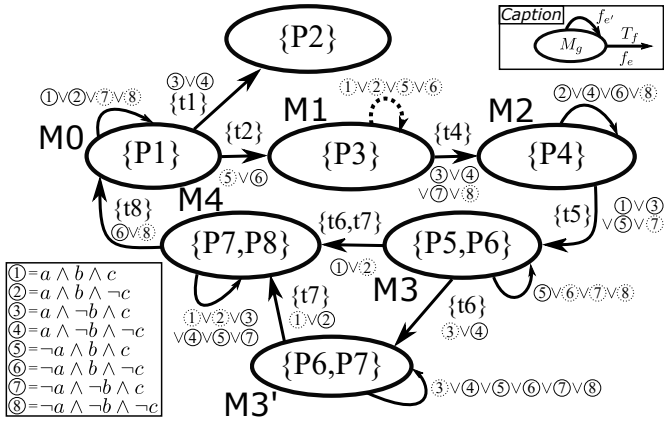

Fig. 9. ARG with the observed (minterms circled with dotted line) and unobserved behaviors (minterms circled with continuous line)

firing of t_1 has been observed; hence the behavior of the system from M_0 and for the input combinations that correspond to the minterms 3 and 4 is unobserved. A firing of t_2 for the minterm 5 has been observed; this is not the case for the minterm 6. The self-loop on M_0 has been partially observed (observed for the minterms 7 and 8, unobserved for the minterms 1 and 2). Two transitions are also enabled for M_3: $T_e = \{t_6, t_7\}$. Given the Boolean functions associated to these transitions, either only t_6 is fired, or both transitions are simultaneously fired. The two firings have been partially observed. This is also the case of the self-loop on M_3.

Globally, the observed behavior is composed of 20 couples (marking, minterm), for instance $(M_3, ②)$ or $(M_3, ⑦)$, and the behavior which has not been observed with 28 such couples, for instance $(M_3, ①)$ or $(M_3, ⑤)$. It may be noted that only the self-loop on M_1 has been fully observed.

5. CONCLUSION

This paper has presented a relative identification method for closed-loop reactive systems. The development of the Boolean functions on inputs which are associated to the transitions of the knowledge model permits to point out the behaviors which have been observed and unobserved. These information can be the starting point of research on active identication. In this approach indeed, the experiments to obtain the I/O sequences from which the identified model is to be built do not come from purely

passive observation of the evolutions of the closed-loop system but require that some inputs of the controller are to be forced. The forcing sequences must be constructed to observe evolutions which have not been obtained by passive observation; therefore, their construction relies on the accurate knowledge of the unobserved behavior. Construction of these sequences is the issue that will be addressed in our future work.

REFERENCES

Angluin, D. (1987). Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2).

Dotoli, M., Fanti, M.P., Mangini, A.M., and Ukovich, W. (2011). Identification of the unobservable behaviour of industrial automation systems by Petri nets. *Control Engineering Practice*, 19(9), 958–966.

Estrada-Vargas, A.P., Lpez-Mellado, E., and Lesage, J.J. (2015). A Black-Box Identification Method for Automated Discrete-Event Systems. *IEEE Transactions on Automation Science and Engineering*, 14(3), 1321–1336.

Estrada-Vargas, A.P., López-Mellado, E., and Lesage, J.J. (2010). A Comparative Analysis of Recent Identification Approaches for Discrete-Event Systems. *Mathematical Problems in Engineering*, 2010, 1–21.

Giua, A. and Seatzu, C. (2005). Identification of free-labeled Petri nets via integer programming. In *Proceedings of the 44th IEEE Conference on Decision and Control*, 7639–7644.

Gold, E.M. (1967). Language identification in the limit. *Information and Control*, 10(5), 447 – 474.

Meda-Campaña, M. and López-Mellado, E. (2005). Identification of concurrent discrete event systems using Petri nets. In *Proceedings of the 17th IMACS World Congress on Computational and Applied Mathematics*, 11–15.

Minas, M. and Frey, G. (2002). Visual PLC-programming using signal interpreted Petri nets. In *Proceedings of the 2002 American Control Conference (IEEE Cat. No.CH37301)*, volume 6, 5019–5024 vol.6.

Provost, J., Roussel, J.M., and Faure, J.M. (2011). A formal semantics for Grafcet specifications. In *2011 IEEE International Conference on Automation Science and Engineering*, 488–494.

Roth, M., Lesage, J.J., and Litz, L. (2010). Black-box identification of discrete event systems with optimal partitioning of concurrent subsystems. In *Proceedings of the 2010 American Control Conference*, 2601–2606.

Saives, J., Faraut, G., and Lesage, J.J. (2015). Identification of discrete event systems unobservable behaviour by petri nets using language projections. In *2015 European Control Conference (ECC)*, 464–471.

Schneider, S., Litz, L., and Lesage, J.J. (2012). Determination of timed transitions in identified discrete-event models for fault detection. In *2012 IEEE 51st IEEE Conference on Decision and Control (CDC)*, 5816–5821.

Appendix A

Algorithm 1 Identifed model construction

procedure ModelConstruction(KM, ω)
  $k = IS = 0$
  ▷ *Initial state detection (ISD)*
  $M = M_0$
  InitRG(M_0) ▷ Initializing the identified model
  while $\hat{y}_k \neq \gamma(M_0)$ **do**
    if $k > |\omega|$ **return** False ▷ Not compatible
    $k = k + 1$
  end while
  $IS = k$ ▷ Saving the initial state detected
  $k' = k$
  ▷ *Forward analysis (FA)*
  while $\hat{y}_{k'} \neq \hat{y}_k$ **do** ▷ Looking for an output vector change
    if $k' > |\omega|$ **then**
      $k' = k = IS$ ▷ Returning to the initial state
      $M = M_0$ **go to** BA
    end if
    $k' = k' + 1$
  end while
  $I_e = \{\hat{u}_{k'}\}$
  if $T_f = \emptyset$ **go to** ISD ▷ No transitions can be fired
  $I_s = \emptyset$
  for k'' **From** k **To** k' **do**
    if $\exists t_j \in T_e(M), F_j(\hat{u}_{k''}) = 1$ **then**
      go to ISD ▷ One transition is fired without output change
    end if
    $I_s = I_s \cup \{\hat{u}_{k''}\}$
  end for
  $s(t_j) = 1$ for $t_j \in T_f$
  $M = M + \mathbf{W}.s$ ▷ Computing next marking
  AddingMarkingRG($M, T_e(M), T_f, I_e, I_s$)
  if $\hat{y}_{k'} = \gamma(M)$ **go to** FA
  else go to ISD
  ▷ *Backward analysis (BA)*
  while $\hat{y}_{k'} \neq \hat{y}_k$ **do** ▷ Searching for an output vector change
    if $k' = 0$ **return** RG ▷ Compatible
    $k' = k' - 1$
  end while
  if $\forall t_j \in \tilde{T}_e(M), F_j(\hat{u}_{k'}) = 0$ **then** ▷ $\tilde{T}_e(M)$ Set of previous enabled transitions
    $k = IS$ ▷ Returning to the initial state
    go to ISD
  end if
  for k'' **From** k' **To** k **do**
    if $\exists t_j \in \tilde{T}_e(M), F_j(\hat{u}_{k''}) = 1$ **then**
      $k = IS$ ▷ Returning to the initial state
      go to ISD
    end if
  end for
  $s(t_j) = 1$ for $t_j \in \tilde{T}_f$
  $M = M - \mathbf{W}.s$ ▷ Computing the previous marking
  go to BA
end procedure

Appendix B

Algorithm 2 Unobserved behavior determination

procedure UnObsMinterms(v) ▷ For one vertex v of the RG
  $\mathbb{E} = \mathcal{P}(T_e)$
  $R = \emptyset$
  f_E
  for T_f **in** $\mathbb{E}$ **do** $f_e = 1$
    for t_j **in** T_e **do**
      if t_j **in** T_f : $f_e = f_e \wedge F_{t_j}$
      else $f_e = f_e \wedge \neg F_{t_j}$
    end for
    if $f_e(T_f)! = 0$: $f_E = f_E \cup f_e$
  end for
  AddEdgeARG(RG, f_E) ▷ Add missing evolutions to the ARG
  T_e = InputCollection(v, ARG) ▷ Set of input vectors leading to an evolution
  $E_{min} = \emptyset$ ▷ Set of minterms
  $O_{min} = \emptyset$ ▷ Set of observed minterms
  for f_e **in** f_E **do**
    min = Minterms(f_e) ▷ Set of minterms
    for m **in** min **do**
      $E = E \cup m$
      for $\hat{u}$ **in** $I_s \cup I_e$ **do**
        if $m(\hat{u}) = True$ **then**
          $O_{min} = O_{min} \cup m$ ▷ the minterm m has been observed
        end if
      end for
    end for
  end for
  $U_{min} = E_{min} - O_{min}$ ▷ Unobserved minterms
  return (U_{min}, O_{min})
end procedure