Discovering of the unobservable behaviour of an Interpreted Petri Net model

Francesco Basile, Gregory Faraut, Luigi Ferrara, Jean-Jacques Lesage

Discovering of the unobservable behaviour of an Interpreted Petri Net model

Francesco Basile[1], Gregory Faraut[2], Luigi Ferrara[1] and Jean-Jaques Lesage[2]

***Abstract*— This paper focuses on the problem of discovering a Petri Net model from long event sequences generated by a discrete event system. Precisely, it is assumed that the behavior of the relations between input and output events (i.e. the observable behaviour of the system) is already modeled by a set of Interpreted Petri Net fragments while the behavior of the internal state evolutions (i.e. the unobservable behaviour) must be discovered. An approach inspired to net synthesis is proposed. It relies on an optimization-based procedure for the identification of the unobservable structure, consisting of a set of places only without adding new transitions, and the initial marking of the added places.**

I. Introduction

The interest for the identification of Discrete Event Systems (DESs) usually comes from reverse engineering for (partially) unknown systems, fault diagnosis, or system verification. Inputs and/or outputs sequences are observed during the operation of the system within its environment. System identification consists in building a model that can reproduce the observed sequences, simulating the observed behavior.

The methods presented in the literature for the identification of DESs produce a mathematical model expressed as a Petri Net (PN) or a finite state automaton model of the system behavior from sequences observed during the system operation [10], [2]. When the resulting model is a PN, like in this paper, the net structure (places, transitions and arcs) and its initial marking must be computed.

The language of the identified model, that is the set of sequences it can generate, in general contains a subset of sequences that do not belong to the observed language. Such a subset represents the exceeding language of the identified system. The size of the exceeding language is a measure of the fitness of the obtained model. Indeed, a large exceeding language is certainly undesired when the identified model is used for diagnostic or verification purposes.

In this work, the identification of the model of closed-loop controlled automation systems is considered. The behaviour of these systems can be split into an *Observable behaviour*, related to direct output changes depending on input changes, and an *Unobservable behaviour*, related to evolutions of the internal state (and variables) of the system without changes of observable data (inputs and outputs). An identification algorithm should provide a model expressing both input/output causal relationships and internal state evolutions due to input changes [15].

[1] F. Basile and L. Ferrara are with DIEM, Università di Salerno, Italy.
[2] G. Faraut and J.J. Lesage are with LURPA, ENS Cachan, Univ. Paris-Sud, Univ. Paris-Saclay, France.

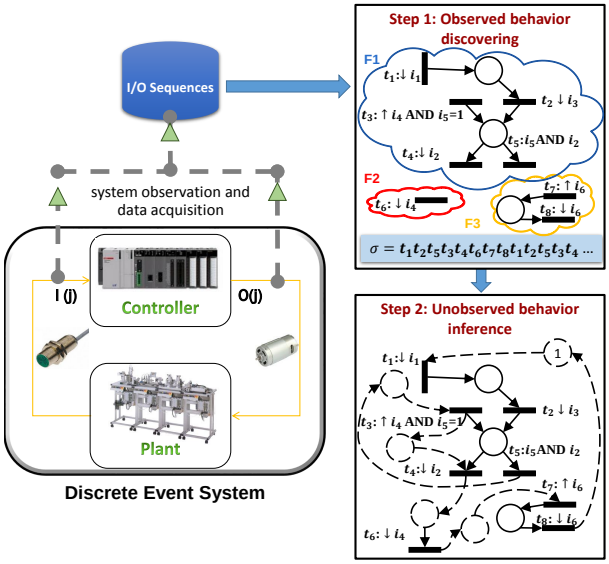

Fig. 1. Identification procedure of closed-loop systems in two steps.

This paper can be positioned in the continuation of [7] as the papers [8], [9], [15]. The authors of [7] provide an approach to discover the observable behaviour as Interpreted PN (IPN) fragments from an input/output observed sequence. This sequence is also converted into a firing sequence on the alphabet of observable transition (see first step in Fig. 1). Therefore, the unobservable behaviour is discovered from such a firing sequence, and the IPN fragments are completed by adding connecting (unobservable) places (see second step in Fig. 1); these places implement a proper ordering of the transition firings and, thus, are essential in reducing the size of the exceeding behavior.

Solutions to implement the second step have been presented in [8], [9], [15]. In [8], [9] it is implemented discovering the causal and concurrent relationships between transition firings in the firing sequence obtained in the first step. In [15] the projection of the firing sequence obtained in the first step on subalphabets is used to discover specific patterns that are characteristic of dependency relationships between the transition firings. Both the approaches return as identified model a 1-bounded net.

In this paper a different procedure is used to discover the unobservable behavior.

First, a synthesis approach is used to solve the problem of reducing the exceeding language. There are approaches to DES identification where it is assumed that either the whole state space of the system, or the whole language generated

by it, is known [3], [12], [4], [5]. If this is the case, the tackled problem is more a *net synthesis* problem, rather than a *net identification* one. In this paper a net synthesis approach based on a graph is used. Precisely, the approach presented in this paper, inspired to [11], forces the reachability graph of the identified model to be isomorphic to a graph generating a behavior having empty exceeding language with respect to words of lenght r, where r is a design parameter. Moreover, the observable net, as well as the identified unobservable one, are not required to be 1-bounded.

Second, the sequences of transition firings of the observable net obtained as result of the first stage of the procedure depicted in Fig. 1, are enriched by the observable markings reached during the firing of the sequence of transitions by the observable net model. The goal is the definition of an exceeding language with respect to sequences of transitions (associated to system inputs) and markings (associated to system outputs) and not transition only to improve the quality of the identified model in terms of accuracy. Indeed, assume that a sequence t_1t_2 is generated by an identified net model; if we consider also system outputs, it may happen that $\boldsymbol{m}_1t_1\boldsymbol{m}_2t_2\boldsymbol{m}_3$ and $\boldsymbol{m}_4t_1\boldsymbol{m}_5t_2\boldsymbol{m}_6$ are generated by the identified model, while only $\boldsymbol{m}_1t_1\boldsymbol{m}_2t_2\boldsymbol{m}_3$ has been observed.

Third, an optimization approach based on an ILP formulation is used for the synthesis of unobervable places. This is in line with a recent trend in PN-related research which indicates that many analysis, control, fault identification and diagnosis problems can be more conveniently formulated and solved as optimization problems, usually in the form of ILP problems (see, *e.g.*, [17], [6]). Indeed, it turns out that, despite their computational complexity, *optimization-based* approaches can be practically more convenient when compared to alternative solutions, since they rely on *off-the-shelf* optimized solvers, as opposed to *ad hoc* algorithms. In particular, various efficient software suites can be employed to tackle ILP problems, such as CPLEX® or FICO™ Xpress [1].

II. PN BACKGROUND AND LANGUAGE MEASURES

A brief recall on Petri Nets is presented in this section. For a complete review on PNs, the reader can refer to [13].

A *Place/Transition net* (P/T net) is a 4-tuple $N = (P, T, \mathbf{Pre}, \mathbf{Post})$ where: $P = \{p_1, ..., p_{|P|}\}$ is a set of places, $T = \{t_1, ..., t_{|T|}\}$ is a set of transitions and $\mathbf{Pre}$ and $\mathbf{Post}$ are the $|P| \times |T|$ sized, natural valued, incidence matrices; $\mathbf{Post}(p, t) = w$ means that there is an arc from t to p with weight w and $\mathbf{Pre}(p, t) = 0$ indicates no arc from p to t. A *marking* (the net state) is a vector $\boldsymbol{m} : P \to \mathbb{N}$ that assigns to each place a nonnegative integer number of tokens. The token flow matrix $\boldsymbol{C}$ of the net is $\boldsymbol{C} = \mathbf{Post} - \mathbf{Pre}$. A P/T *system* or *net system* $\langle N, \boldsymbol{m}_0\rangle$ is a net N with an initial marking $\boldsymbol{m}_0$. A transition t is *state-enabled* at $\boldsymbol{m}$ *iff* $\boldsymbol{m} \geq \mathbf{Pre}(\cdot, t)$; its firing yields a new marking $\boldsymbol{m}' = \boldsymbol{m} + \boldsymbol{C}(\cdot, t)$.

To make explicit the interaction of a PN model with the external environment, we make use of Interpreted Petri Nets (IPN), whose structure is defined as $N = (\overline{P}, T, \overline{\mathbf{Pre}}, \overline{\mathbf{Post}}, \gamma, \beta)$. The first advantage of IPNs is the possibility to characterize each transition with a logic condition depending on the system's inputs; in particular, the function $\beta : T \to \{0, 1\}$ associates a logical condition to transitions $\forall t_i \in T, \beta(t_i) = F_i(I, E_I)$, where I is the set of input signals and $E_I = \{\uparrow i_i (\downarrow i_i) \mid i_i \in I\}$ is the set their rising (falling) edges. Secondly, the system's outputs are explicitly represented in the net; in particular, the set of places is partitioned into observable and unobservable ($\overline{P} = P \cup P^u$, $P \cap P^u = \varnothing$) and the function $\gamma : \overline{P} \to O \cup \{\epsilon\}$ associates to each observable place an output signal and to each unobservable place the symbol ϵ denoting a null output. Thus, observable places are used to model the output configurations of the DES while the unobservable ones implement a proper ordering of the transition firings.

An *Interpreted PN system* $\langle N, \overline{\boldsymbol{m}}_0\rangle$ is an IPN N with an initial marking $\overline{\boldsymbol{m}}_0$. In an IPN system, a transition t fires at $\overline{\boldsymbol{m}}$ if and only if it is state enabled and logical condition enabled ($\beta(t) = 1$).

In the following, given a set A, the cardinality of A is denoted by $|A|$, and given a sequence σ, $|\sigma|$ denotes its length.

We assume that an IPN system $\langle N, \overline{\boldsymbol{m}}_0\rangle$ modelling the system of interest is available. The set $I\Sigma_{Obs} = \{i\sigma_1, i\sigma_2, ...i\sigma_{|I\Sigma_{Obs}|}\}$ is also available, which represents in form of *interpreted sequences* $i\sigma_j = \boldsymbol{m}_{0,j}t_{1,j}\boldsymbol{m}_{1,j}...t_{|i\sigma_j|-1,j}\boldsymbol{m}_{|i\sigma_j|-1,j}$, a sequence consisting of transition firings and observable markings, the traces of inputs and outputs acquired during the observations of the system. Each sequence is supposed to represent a different trajectory of the system starting from the same initial state.

The minimal requirement for $\langle N, \overline{\boldsymbol{m}}_0\rangle$ is to *simulate* the observations: called $I\Sigma(N, \overline{\boldsymbol{m}}_0)$ the set of interpreted sequences firable by N from $\overline{\boldsymbol{m}}_0$, it must hold $I\Sigma_{Obs} \subseteq I\Sigma(N, \overline{\boldsymbol{m}}_0)$.

Given an interpreted sequence $i\sigma_j \in I\Sigma(N, \overline{\boldsymbol{m}}_0)$, the subsequence of length n starting from the k-th marking is a *word* $w^n_k(i\sigma_j)$, where $w^n_{k,j} \equiv w^n_k(i\sigma_j) = \boldsymbol{m}_{k,j}\, t_{k+1,j}\, \boldsymbol{m}_{k+1,j}...\boldsymbol{m}_{k+n-2,j}\, t_{k+n-1,j}\, \boldsymbol{m}_{k+n-1,j}$, $0 \leq k \leq |i\sigma_j| - n$. Words $w^n_{k,j}$ associated to $i\sigma_j \in I\Sigma_{Obs}$, are called *known* since they directly originate from the observations. Given a length n, they can be collected in the set of known words of length n, $K^n = \{w^n_{k,j},\ 0 \leq k \leq |i\sigma_j| - n,\ i\sigma_j \in I\Sigma_{Obs}\}$, while $L^n_{Obs} = \{w^l_{0,j},\ l \leq n,\ i\sigma_j \in I\Sigma_{Obs}\}$ is called *observed language* of length n and collects known words of maximum length n starting from the initial marking. Similarly, the set of *unknown words* of length n is defined $U^n(N, \overline{\boldsymbol{m}}_0) = \{w^n_{k,j}, \forall\, 0 \leq k \leq |i\sigma_j| - n, i\sigma_j \in (I\Sigma(N, \overline{\boldsymbol{m}}_0) \setminus I\Sigma_{Obs})\}$, and is due to unexpected transition firings.

Simulation admits that the model can enable unexpected transition firings at each marking in addition to expected ones. On the contrary, an accurate model only enables expected transitions; formally, denoted by $L^n(N, \overline{\boldsymbol{m}}_0) = \{w^l_{0,j}, l \leq n, i\sigma_j \in I\Sigma(N, \overline{\boldsymbol{m}}_0)\}$ the language of length n generated by the identified net system from $\overline{\boldsymbol{m}}_0$, the

exceeding language $L^n_{Exc}(N, \overline{\boldsymbol{m}}_0) = L^n(N, \overline{\boldsymbol{m}}_0) \setminus L^n_{Obs}$ is empty in an accurate model for each length n. Maximum accuracy, however, is quite never the target for a good model. In fact, since real systems usually exhibit a rich behavior, long sequences are needed to capture them; however, observations are generally not complete in practice, *i.e.* not all the possible trajectories are observed. Thus, building a model with maximum accuracy implies that new (future) observations cannot be generated by the identified model.

The empirical experience usually suggests that it makes sense to devise a tolerance length $\tilde{n}$ able to preserve a rich set of observations. In this case a good model is such that every produced word of length $n \leq \tilde{n}$ is a known word; more formally, called $distance$ of length n the quantity $d^n(N, \overline{\boldsymbol{m}}_0) = |U^n(N, \overline{\boldsymbol{m}}_0) \backslash K^n|$, it must hold $d^{\tilde{n}}(N, \overline{\boldsymbol{m}}_0) = 0$. Obviously $\tilde{n}$ cannot exceed the length of the longest observation.

The goal of this paper is to identify the unobservable places to be added to an existing IPN model to make it more accurate. At this aim it is assumed that:

1) the automation system is already modeled by an IPN system $\langle N^{Obs}, \boldsymbol{m}_0 \rangle$ and the set $I\Sigma_{Obs}$ is available; it is also called *observable net*, since it represents the input/output observable behaviour of the automation system and only contains observable places and transitions [16],[14]; obviously, $\langle N^{Obs}, \boldsymbol{m}_0 \rangle$ must simulate the observations;
2) the identification process consists in discovering the unobservable subnet system of the given IPN system; this subnet contains the same transitions of the observable net and is made of unobservable places only;
3) $\tilde{n}$ is given.

III. A SOLUTION BASED ON ILP PROBLEMS SOLVING

It is useful to characterize the accuracy through the design parameter $r \geq \tilde{n}$. The objective is thus to construct a new IPN system $\langle N', \overline{\boldsymbol{m}}_0 \rangle$ such that:

1) $d^r(N', \overline{\boldsymbol{m}}_0) = 0$;
2) $L^r_{Exc}(N', \overline{\boldsymbol{m}}_0) = \varnothing$;

If 1) and 2) hold, the net N' is said to be r-complete. The two constraints are both necessary: if only the first one is imposed, the model just produces known words of length $n \leq r$, but some words could be produced from the initial marking even if they were observed from a state of the system different from the initial one; if the second one only is imposed, unknown words of any length n could be produced after the first $r-1$ firings from the initial marking. Note that r-completeness entails $\tilde{n}$-completeness ($r \geq \tilde{n}$).

This task is accomplished by firstly constructing a Moore machine E_r representing the dynamics that N' should exhibit. For the sake of simplicity, in this paper we suppose that if it holds $w^{r-1}_{0,i} = w^{r-1}_{k,j}, k > 0, i\sigma_i, i\sigma_j \in I\Sigma_{Obs}$ then it also holds $w^r_{0,i} = w^r_{k,j}$; therefore, if a word of length $r-1$ is produced both at the beginning of any observation and also later, it must be followed by the same transition firing and observable marking. This assumption can be easily removed by slightly modifying the algorithm illustrated in Figure 2, which shows how E_r is created for $r = 4$. Starting from the initial marking, for each interpreted sequence $i\sigma_j \in I\Sigma_{Obs}$, a state is created (if does not exist yet) for each prefix of $i\sigma_j$ of length smaller than $r-1$; moreover, a state is created for each sub-sequence of $i\sigma_j$ belonging to K^{r-1}. At the end of the procedure, the words associated to the states of E_r are the ones contained in the following set: $\Phi(r) = \{L^{r-2}_{Obs} \cup K^{r-1}\}$. Each state has as output λ the last observable marking of the associated word and is linked to the next through an arc associated to the transition firing that causes the marking change. In the example, the last word of length 3 generated by $i\sigma_j$ is supposed to be equal to the one associated to q_3; as a consequence, q_3 is re-used and $t_{|i\sigma|-1,j}$ is added as input arc coming from the node associated to $\boldsymbol{m}_{|i\sigma_j|-4,j} t_{|i\sigma_j|-4,j} \boldsymbol{m}_{|i\sigma_j|-3,j} t_{|i\sigma_j|-3,j} t_{|i\sigma_j|-2,j} \boldsymbol{m}_{|i\sigma_j|-2,j}$. By applying this procedure for each $i\sigma_j$, the automaton is completely created.

A word w can be associated to each direct path belonging to such an automaton. Precisely, given a direct path from a state q_i to a state q_j, the word $w = \lambda(q_i) t_i \lambda(q_{i+1}) t_{i+1} ... t_{j-1} \lambda(q_j)$ can be devised, where t_k is one of the transitions associated to the arcs exiting from q_k. Once these words have been defined, as done for IPNs, language and distance can be introduced in a similar way also for E_r. The automaton E_r is r-complete by construction.

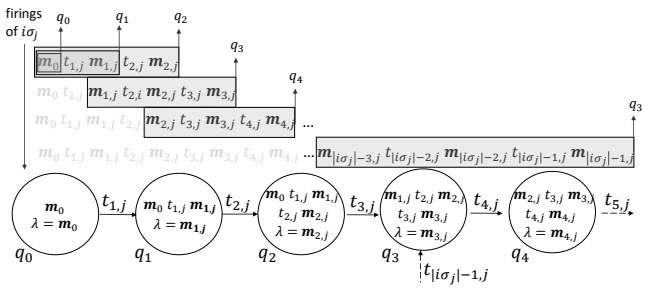

Fig. 2. Creation of the Moore Machine E_r for $r = 4$.

Being the net system $\langle N^{Obs}, \boldsymbol{m}_0 \rangle$ not r-complete, some sequences exist that can be produced in N^{Obs} from $\boldsymbol{m}_0$ but not in E_r from q_0. In the general case, each one of these sequences can be splitted in two parts: the first subsequence (that can be empty) is still feasible in E_r; the second one begins with the first infeasible transition firing, *i.e.* the first transition that is not expected in E_r. The general idea is to add in the unobservable subnet system a new unobservable place that blocks the first unexpected transition firing of the infeasible (sub)path. These places are synthetized by solving ILP problems based on proper algebraic constraints, which are splitted into *common* and *specific* constraints.

A. Common algebraic constraints

In this section, we devise the set of constraints that each unobservable place must ensure; more specifically:

1) each known interpreted sequence in $I\Sigma_{Obs}$ must fire from the initial marking;
2) each time a word $w \in \Phi(r)$ is produced, the marking reached at the end of w must be always the same.

In order to formalize these constraints, the following definitions come in help:

1) The sequence of transitions associated to $i\sigma_j$ is $\sigma_j = t_{1,j}t_{2,j}...t_{|i\sigma_j|-1,j}$; it holds $|i\sigma_j| = |\sigma_j| + 1$.
2) The function $\Theta : w \to \{\sigma\}$, defined for each $w \in K^n$, yields the (sub)sequences of transitions starting from the initial marking that produce w in the last $|w|-1$ firings, *i.e.* $\Theta(w) = \{\sigma_j(1, k+|w|-1) \,|\, \exists\, w_{k,j}^{|w|} = w, i\sigma_j \in I\Sigma_{Obs}\}$, where $\sigma(m,n)$ is the subsequence of σ having all its elements from the m-th to the n-th; if $n < m$ the empty sequence is returned.
3) An unobservable place p^U of the unobservable subnet system, is said to be $r-$*consistent* if and only if $\forall w \in \Phi(r)$ it holds $\boldsymbol{c}^U \boldsymbol{\sigma}_i = \boldsymbol{c}^U \boldsymbol{\sigma}_j$ where $\boldsymbol{c}^U$ is the associated token flow row, $\sigma_i, \sigma_j \in \Theta(w), i \neq j$ and $\boldsymbol{\sigma} \in \mathbb{N}^{|T|}$ is the firing count vector associated to σ whose k-th component is the number of occurrences of t_k in σ. If $\sigma = t_i$, then the associated firing count vector is denoted by $\mathbf{t}_i$.

The previously enumerated constraints can now be formally expressed. Chosen r, any unobservable place p^U of the unobservable subnet system, identified by the incidence rows $\mathbf{pre}^U, \mathbf{post}^U$ and the initial marking m_0^U, must ensure the following algebraic system:

$$\begin{cases} m_0^U + \mathbf{c}^U \tilde{\boldsymbol{\sigma}}_1 \geq \mathbf{pre}^U \tilde{\boldsymbol{\sigma}}_2, & (1) \\ \quad \tilde{\sigma}_1 = \sigma_k(1, l-1), \tilde{\sigma}_2 = \sigma_k(l), \\ \quad \forall\, i\sigma_k \in I\Sigma_{Obs}, \forall\, 1 \leq l \leq |\sigma_k|; \\ \mathbf{c}^U \boldsymbol{\sigma}_i = \mathbf{c}^U \boldsymbol{\sigma}_j, & (2) \\ \quad \forall \sigma_i, \sigma_j \in \Theta(w), \forall w \in \Phi(r), i \neq j; \end{cases}$$

where $\mathbf{c}^U = \mathbf{post}^U - \mathbf{pre}^U$, since by inequalities (1) legal firings are admitted, and by equations (2) consistency is ensured.

B. Specific algebraic constraints

In this section, the *sets* of *specific* constraints are presented; we use the term *specific* to remark that each *set* will be implemented by a single unobservable place, in addition to common constraints.

Specific constraints are responsible for enforcing the disabling of undesired transition firings; as a consequence, they depend on the accuracy of the given net. For sake of generality, it is convenient to suppose that the given net system $\langle N, \overline{\boldsymbol{m}}_0 \rangle$ satisfies the hypothesis and already contains some r-consistent unobservable places.

Let us consider the word $w \in \Phi(r)$ associated to a state of E_r. Constraint (2) guarantees that the IPN system always reaches the same marking of the unobservable subnet from the initial marking by firing a sequence σ that produces w in the last $|w|-1$ firings, *i.e.* $\sigma \in \Theta(w)$. The same occurs also for the observable subnet by definition of interpreted sequence that always terminates with an observable marking. Consequently, the set of state-enabled transitions by the IPN system after w occurred is uniquely determined by w and we denoted it by $\mathcal{E}(N, w)$.

In addition we denote by $\mathcal{A}_r(w)$ the set of transitions associated to the output arcs of the state of E_r associated to w.

Now, specific constraints can be formally expressed as follows. Choosen a word $w \in \Phi(r)$ and selected a $\sigma_i \in \Theta(w)$, the unobservable place p^U identified by the incidence rows $\mathbf{pre}^U, \mathbf{post}^U$ and the initial marking m_0^U, disables the undesired firing of a transition $t_u \in \{\mathcal{E}(N, w) \setminus \mathcal{A}_r(w)\}$ after the firing of σ_i from m_0^U *iff*:

$$\begin{cases} m_0^U + \mathbf{c}^U \boldsymbol{\sigma}_i < \mathbf{pre}^U \mathbf{t}_u; & (3) \\ \mathbf{pre}^U\, \mathbf{t}_u > 0; & (4) \end{cases}$$

where $\mathbf{c}^U = \mathbf{post}^U - \mathbf{pre}^U$, since by inequality (3) the disabling of the undesired firing is accomplished while by (4) the blocking arc is imposed to the transition t_u.

C. The core algorithm

In this section, the core algorithm that produces the desired net system $\langle N', \overline{\boldsymbol{m}}_0' \rangle$ is presented.

Algorithm 1: r-completeness enforcement

input : $E_r, I\Sigma_{Obs}, r, \langle N^{Obs}, \boldsymbol{m}_0 \rangle$.
output: A new IPN system $\langle N', \overline{\boldsymbol{m}}_0' \rangle \equiv \langle (\overline{P}', T, \overline{\mathbf{Pre}}', \overline{\mathbf{Post}}', \gamma', \beta'), \overline{\boldsymbol{m}}_0' \rangle$ that tries to ensure r-completeness.

Initialize: $\langle N', \overline{\boldsymbol{m}}_0' \rangle = \langle N^{Obs}, \boldsymbol{m}_0 \rangle$
$f_{min} = (\mathbf{pre}^U + \mathbf{post}^U) \cdot \mathbf{1} + m_0^U$
for *each* $w \in \Phi(r)$ **do**
  Select a random $\sigma_i \in \Theta(w)$
  $T_u^{treated} = \varnothing$
  while $\{\{\mathcal{E}(N', w) \setminus \mathcal{A}_r(w)\} \setminus T_u^{treated}\} \supset \varnothing$ **do**
    Select a random $t_u \in \{\mathcal{E}(N', w) \setminus \mathcal{A}_r(w)\} \setminus T_u^{treated}$
    Add in the set of constraints, named S, (1), (2), (3), (4) devised from w, t_u and σ_i
    if $\nexists\, p^U \in \overline{P}'$ *s.t. S is satisfied* **then**
      a new unobservable place p^U is constructed: $[\mathbf{pre}^U, \mathbf{post}^U, m_0^U, solved] = solveILP(S, f_{min})$
      if *solved* **then**
        $\overline{P}' = \overline{P}' \cup p^U$, $\gamma'(p^U) = \epsilon$,
        $\overline{\mathbf{Pre}}' = \begin{bmatrix} \overline{\mathbf{Pre}}' \\ \mathbf{pre}^U \end{bmatrix}$,
        $\overline{\mathbf{Post}}' = \begin{bmatrix} \overline{\mathbf{Post}}' \\ \mathbf{post}^U \end{bmatrix}$, $\overline{\boldsymbol{m}}_0' = \begin{bmatrix} \overline{\boldsymbol{m}}_0' \\ m_0^U \end{bmatrix}$
    $T_u^{treated} = T_u^{treated} \cup t_u$

Proposition 1: The net system $\langle N', \overline{\boldsymbol{m}}_0' \rangle$ produced by Algorithm 1 is r-complete if and only if all the ILP problems admit a solution, otherwise it holds $d^r(N', \overline{\boldsymbol{m}}_0) \leq d^r(N^{Obs}, \boldsymbol{m}_0)$ and $L^r_{Exc}(N', \overline{\boldsymbol{m}}_0) \subseteq L^r_{Exc}(N^{Obs}, \boldsymbol{m}_0)$.

Proof: The algorithm cycles over each word $w \in \Phi(r)$ and constructs a set of algebraic constraints for each transition t_u whose firing is not expected after w; if not already implemented by any previously added place, each

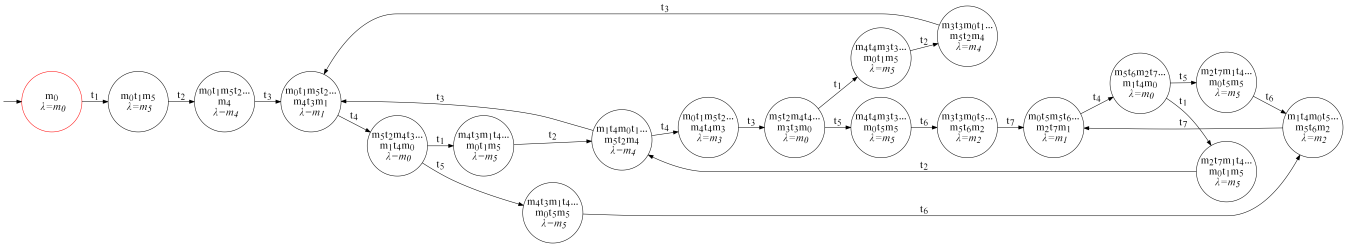

Fig. 3. E_r for the sorting system and $r = 5$.

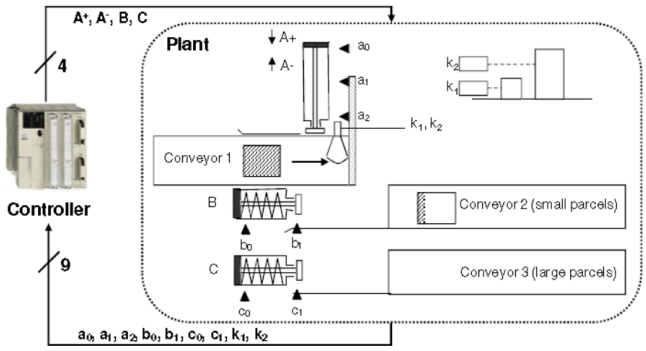

Fig. 4. A simple system to be modeled.

set is given to an ILP solver which, if it exists, returns as a solution the new place of the unobservable subnet system. Such a place disables the firing of t_u (due to (3) and (4)) and satifies the common constraints; thanks to (2) the disabling is performed indipendently on how w is generated by the net system, even if enforced on a single $\sigma_i \in \Theta(w)$.

If at least one ILP is admits a solved, the exceeding language and the distance are reduced since an undesired firing is disabled; if all the sets are implemented, the new system is r-complete. ■

Note that the unsolvability of some ILPs is due to the nature of a Petri net place: it is mostly like a counter since it only maintains memory of its current status. Being it insensible to the order in which increments and decrements occur, it is not able to implement any constraint that is order-sensible. In addition, note that the algorithm produces unobservable places with minimum arcs weight and minimal initial tokens, as the function $f_{min} = (\mathbf{pre}^U + \mathbf{post}^U) \cdot \mathbf{1} + m_0^U$ imposes, where $\mathbf{1}$ is a column vector having all elements equal to 1. Notice that Proposition 1 still holds if a trade-off between arc weights and the number of initial tokens is imposed, i.e. if the three terms $(\mathbf{pre}^U, \mathbf{post}^U, m_0^U)$ are weighted. The produced net system is in general not 1-bounded.

IV. The Example

The purpose of the system in figure 4 is to sort parcels according to their size. It has nine sensors: $a_0, a_1, a_2, b_0, b_1, c_0, c_1, k_1, k_2$ and four actuators: $A+, A-, B, C$. The size of a parcel arrived on conveyor 1 is detected as either small (k_1) or big (k_2). The small (resp. big) parcel is then pushed from the cylinder A to the cylinder B (resp. C) that pushes it on conveyor 2 (resp. 3). The pushing of Cylinder B (C) is detected by means of b_0 (c_0) and b_1 (c_1). Cylinder A moves the parcel as soon as $A+$ is asserted; if a_1 (resp. a_2) rises, it means that the cylinder B (resp. C) has been reached and, thus, the cylinder A is retracted by asserting $A-$, until a_0 rises. The system is sequential: it works only one parcel at a time.

During the observation step of figure 1, a single observation V of 222 I/O vectors is carried out, which captures the treatment of 20 parcels and completely acquires the system dynamics. Figure 5 shows the first eight I/O vectors in V; they correspond to the arrival and sorting of a small parcel.

	$V(1)$	$V(2)$	$V(3)$	$V(4)$	$V(5)$	$V(6)$	$V(7)$	$V(8)$
k_1	0	1	1	0	0	0	0	0
k_2	0	0	0	0	0	0	0	0
a_0	1	1	0	0	0	0	0	0
a_1	0	0	0	0	1	1	0	0
a_2	0	0	0	0	0	0	0	0
b_0	1	1	1	1	1	0	0	0
b_1	0	0	0	0	0	0	0	1
c_0	1	1	1	1	1	1	1	1
c_1	0	0	0	0	0	0	0	0
$A+$	0	1	1	1	0	0	0	0
$A-$	0	0	0	0	1	1	1	1
B	0	0	0	0	1	1	1	0
C	0	0	0	0	0	0	0	0

Fig. 5. First eight acquired I/O vectors for the sorting system.

The observable IPN system $\langle N^{Obs}, \boldsymbol{m}_0 \rangle$ is then computed in [16]; it is depicted by solid lines in figure 6. Then V is translated into an interpreted sequence $i\sigma = i\sigma_1 i\sigma_2 i\sigma_3 i\sigma_1 i\sigma_3 i\sigma_1 i\sigma_1 i\sigma_3 i\sigma_3 i\sigma_1 i\sigma_3 i\sigma_3 i\sigma_3 i\sigma_3 i\sigma_3 i\sigma_2 i\sigma_1 i\sigma_2 i\sigma_2 i\sigma_1$, where $i\sigma_1 = \boldsymbol{m}_0 t_1 \boldsymbol{m}_5 t_2 \boldsymbol{m}_4 t_3 \boldsymbol{m}_1 t_4 \boldsymbol{m}_0$, $i\sigma_2 = \boldsymbol{m}_0 t_1 \boldsymbol{m}_5 t_2 \boldsymbol{m}_4 t_4 \boldsymbol{m}_3 t_3 \boldsymbol{m}_0$ and $i\sigma_3 = \boldsymbol{m}_0 t_5 \boldsymbol{m}_5 t_6 \boldsymbol{m}_2 t_7 \boldsymbol{m}_1 t_4 \boldsymbol{m}_0$, where $\boldsymbol{m}_0 = [0000]^T, \boldsymbol{m}_1 = [0010]^T, \boldsymbol{m}_2 = [0011]^T, \boldsymbol{m}_3 = [0100]^T, \boldsymbol{m}_4 = [0110]^T, \boldsymbol{m}_5 = [1000]^T$. The set of known interpreted sequences is simply $I\Sigma_{Obs} = \{i\sigma\}$ and the initial marking is $\boldsymbol{m}_0$. In this example, $\tilde{n}$ can be quite easily fixed by analyzing the system and looking at the net. Noted that each parcel treatment consists of four transition firings, then $\tilde{n} = 5$; thus, the method can be instanciated for each $r \geq 5$.

By choosing $r = 5$, the automaton E_r in figure 3 is obtained. The new Petri net system $\langle N', \overline{\boldsymbol{m}}_0' \rangle$ is then computed; it is represented in figure 6 (added unobservable places and added arcs in dot lines).

The computed net system is not 5-complete but, as figure 7

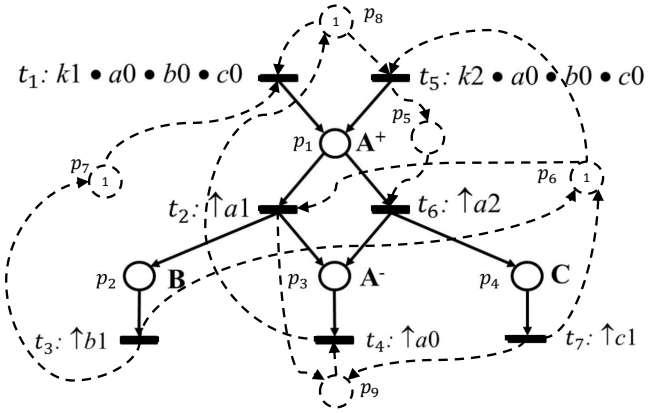

Fig. 6. IPN system $\langle N', \boldsymbol{m}'_0 \rangle$.

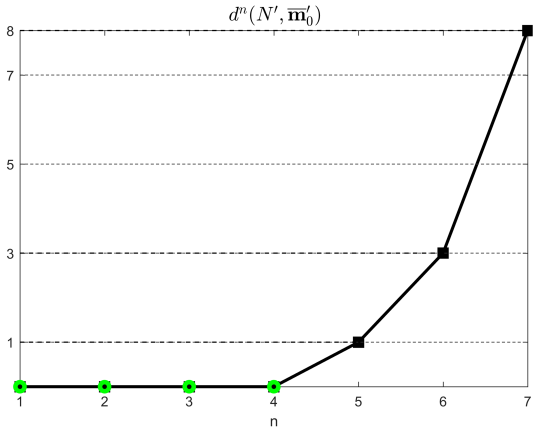

Fig. 7. Distance between the observed language and the produced one. Green points mean 0-distance.

shows, the distance of length 4 is zero. This is a good result, considering that $d^1(N^{Obs}, \boldsymbol{m}_0)) = \infty$, due to the source transitions t_1 and t_5. The exceeding language is reduced also, as figure 8 shows, but *exceeding words* (i.e. words belonging to the exceeding language) of length 2 are still produced; for example, the word $w = \boldsymbol{m}_0 t_5 \boldsymbol{m}_5$ can be produced from the initial marking, even if it is not expected in E_r.

In [16] the unobservable places of N^{Obs} are computed as well, producing the system $\langle N'', \overline{\boldsymbol{m}}''_0 \rangle$. The comparison with $\langle N', \overline{\boldsymbol{m}}'_0 \rangle$, shows that both the exceeding language and the distance are larger, as depicted in figure 9 and 10.

It is interesting to observe that for small values of n the exceeding languages produced by these two nets is quite similar, while the distances are not; the cause is that the distance, differently from the exceeding language, is a global measure which depends from the initial marking and also all the markings reachable from it. As a consequence, deviating behaviours that the exceeding language can show only for great values of the length n are immediately represented for smaller distance lenghts. For example, N'' reaches the illegal observable marking $\boldsymbol{m} = [1100]^T$ by firing the sequence of transitions $\sigma = t_1 t_2 t_4 t_5$; this is represented by $d^1(N'', \overline{\boldsymbol{m}}''_0) > 0$, but requires a length $n \geq 5$ in order to be shown up in $L^n_{Exc}(N'', \overline{\boldsymbol{m}}''_0)$.

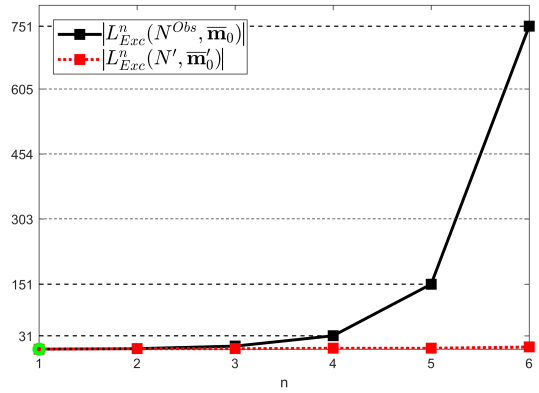

Fig. 8. Comparison on the exceeding language of the observable net *vs* the net produced by the proposed method. Green points mean 0 exceeding words.

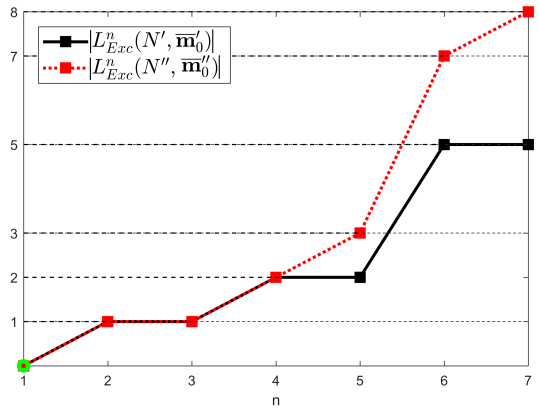

Fig. 9. Comparison on the exceeding language of the observable net *vs* the net produced by the proposed method.

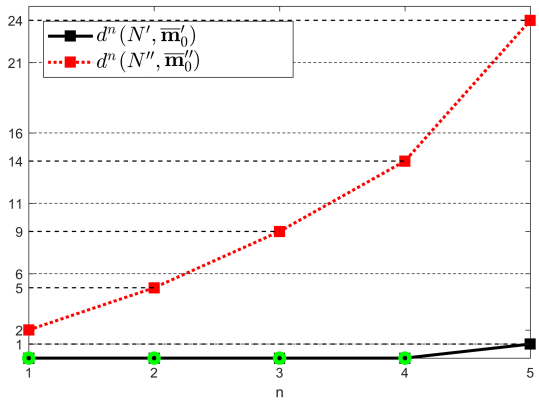

Fig. 10. Comparison on the distances of the net produced by the proposed method *vs* the net produced in [16].

Increasing the r parameter up to $r = 15$ does not produce any modification in the produced net system. For $r = 16$, additional constraints are implemented, resulting in the net system $\langle N''', \overline{\boldsymbol{m}}_0''' \rangle$ that greatly reduces the exceeding language (see figure 11). However, the distance, as shown in figure 12, is quite similar to the case $r = 5$: even if a lot of exceeding words are removed, many unknown words are still generated.

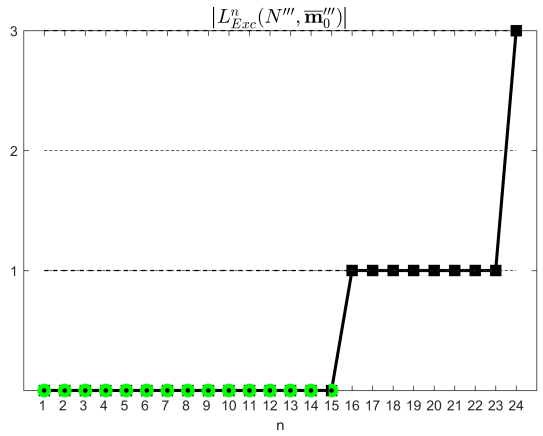

Fig. 11. Exceeding language of the IPN system computed for $r = 16$.

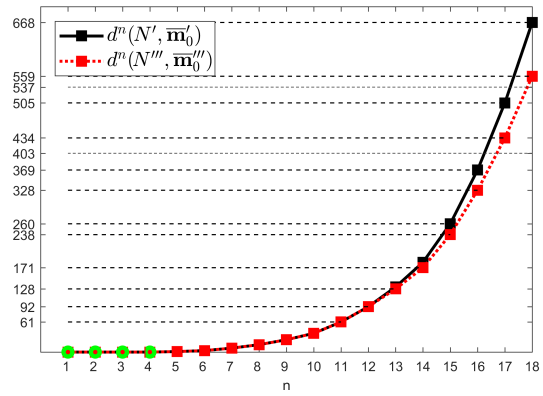

Fig. 12. Comparison on the distance for $r = 5$ and $r = 16$.

V. Conclusions

An approach for the identification of the unobservable behavior of a Petri Net model from long event sequences has been presented. The proposed approach exploits the mathematical representation of PNs and improves previous approaches by evaluating the accuracy of the identified model with respect to sequences of transitions (inputs) and markings (outputs) and not only transitions. At this aim, ILPs are employed as the core element of an optimization-based procedure; this formulation is particularly convenient since commercial optimization tools that are available off-the-shelf can be employed for its solution. The resulting model is a general net, while previous solutions only return 1-bounded nets.

References

[1] Xpress-Optimizer - Reference manual, release 31.01. FICO^{TM} Xpress Optimization Suite, April 2017.

[2] M.P. Cabasino, P. Darondeau, M.P. Fanti, and C. Seatzu. Model identification and synthesis of discrete-event systems. In Mengchu Zhou, Han-Xiong Li, and Margot Weijnen, editors, *Contemporary Issues in Systems Science and Engineering*, IEEE/Wiley Press Book Series, pages 343–366. John Wiley & Sons, Inc., 2015.

[3] M.P. Cabasino, A. Giua, and C. Seatzu. Identification of Petri nets from knowledge of their language. *Discrete Event Dynamic Systems*, 17:447–474, December 2007.

[4] J. Cortadella, M. Kishinevsky, L. Lavagno, and A. Yakovlev. Deriving Petri nets from finite transition systems. *IEEE Trans. on Computers*, 47(8):859–852, August 1998.

[5] P. Darondeau. Region Based Synthesis of P/T-Nets and Its Potential Applications. In *21st International Conference on Application and Theory of Petri Nets 2000 (ICATPN 2000) Aarhus, Denmark*, volume 1825 of *Lecture Notes in Computer Science*, pages 16–23. Springer, June 2000.

[6] M Dotoli, M.P. Fanti, and A.M. Mangini. Real time identification of discrete event systems using Petri nets. *Automatica*, 44(5):1209 – 1219, 2008.

[7] A. P. Estrada-Vargas, J. Lesage, and E. Lopez-Mellado. Identification of industrial automation systems: Building compact and expressive petri net models from observable behavior. In *2012 American Control Conference (ACC)*, pages 6095–6101, June 2012.

[8] A. P. Estrada-Vargas, E. Lopez-Mellado, and J. Lesage. Identification of partially observable discrete event manufacturing systems. In *2013 IEEE 18th Conference on Emerging Technologies Factory Automation (ETFA)*, pages 1–7, Sept 2013.

[9] A. P. Estrada-Vargas, E. Lopez-Mellado, and J. Lesage. A black-box identification method for automated discrete-event systems. *IEEE Transactions on Automation Science and Engineering*, 14(3):1321–1336, July 2017.

[10] A.P. Estrada-Vargas, E. Lopez-Mellado, and J. Lesage. A Comparative Analysis of Recent Identification Approaches for Discrete-Event Systems. *Mathematical Problems in Engineering*, 2010.

[11] A. Ghaffari, N. Rezg, and Xiaolan Xie. Design of a live and maximally permissive petri net controller using the theory of regions. *IEEE Transactions on Robotics and Automation*, 19(1):137–141, Feb 2003.

[12] K. Hiraishi. Construction of a class of safe Petri nets by presenting firing sequences. In Jensen, K., editor, *Lecture Notes in Computer Science; 13th International Conference on Application and Theory of Petri Nets 1992, Sheffield, UK*, volume 616, pages 244–262. Springer-Verlag, June 1992.

[13] T. Murata. Petri nets: Properties, analysis and applications. *Proc. of the IEEE*, 77(4):541–580, April 1989.

[14] J. Saives. *Black-box Behavioural Identification of Discrete Event Systems by Intepreted Petri Nets. PhD.* Univ. Paris-Saclay, 2016.

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

[16] Ana Paula Estrada Vargas. *Black-box Identification of Automated Discrete Event Systems. PhD.* Centro de Investigacin y de Estudios Avanzados del I.P.N., 2013.

[17] G. Zhu, Z. Li, and N. Wu. Model-based fault identification of discrete event systems using partially observed petri nets. *Automatica*, 96:201 – 212, 2018.