Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Identification of free-labeled Petri nets via integer programming

Alessandro Giua, Carla Seatzu

Abstract— **In this paper we deal with the problem of identifying a Petri net system, given a finite language that it generates. In particular, we consider the problem of identifying a free-labeled Petri net system, i.e., a net system where each transition is assigned a unique label. We show that the identification problem can be solved via an integer programming problem. We also discuss how additional structural constraints can be easily imposed to the net.**

I. Introduction

In this paper we present a linear algebraic approach for the identification of a free-labeled Petri net system from the knowledge of a finite set of strings that it generates. Free-labeled Petri nets are nets where each transition is assigned a unique label: in this case the set of transitions T coincides with the sets of observed events.

A. The proposed approach

Identification is a classical problem in system theory: given a pair of observed input-output signals it consists in determining a system such that the input-output signals approximate the observed ones [11]. We consider this problem in the context of Petri nets. The observed behavior in this case is the language of the net, i.e., the set of transition sequences that can be fired starting from the initial marking.

Assume that a language $\mathcal{L} \subset T^*$ is given, where T is a given set of n transitions. Let this language be finite, prefix-closed and let k be the length of the longest string it contains. Given a fixed number of places m, the identification problem we consider consists in determining the structure of a net N, i.e., the matrices $Pre, Post \in \mathbb{N}^{m \times n}$, and its initial marking $M_0 \in \mathbb{N}^m$ such that the set of all firable transition sequences of length less or equal to k is $L_k(N, M_0) = \mathcal{L}$.

Note that the set $\mathcal{L}$ explicitly lists *positive examples*, i.e., strings that are known to belong to the language, but also, implicitly, defines several *counterexamples*, namely all those strings of length less or equal to k that do not belong to the language.

The solution we propose in Section IV is based on the definition of a linear integer constraint set whose admissible solutions define all nets solving the identification problem. Note that it is also possible to define several suitable objective functions so that, if more that one solution exists, the optimal one (e.g., the one with the minimal structure in terms of number of arcs or tokens) can be found. The approach is extremely general and can be applied to any class of place/transition nets.

There are also many other possibile requirements that can be imposed on the net to be synthesized: as an example, the existence of P or T-invariants, the decomposition of the net in subsystems, or the fact that the net should be a marked graph or a state machine. These additional requirements can be easily characterized by suitable linear constraints as we discuss in Section V.

Finally, in Section VI we show how this approach can be used to solve a different problem, namely that of identifying a bounded net system whose language is given in terms of its reachability tree represented as a finite state automaton. This problem has also been addressed within the framework of the *theory of regions* [2].

An original feature of the proposed approach is the fact that, by choosing a suitable objective function, it can also be used to determine a minimal net according to a given measure. The main drawback is its computational complexity, in the sense that the number of unknowns grows with the number of counterexamples.

The complexity of the constraint sets we use to characterize the set of admissible solutions is analyzed in the paper.

B. Related literature

The idea of learning the structure of an automaton from positive examples and from counterexamples has been explored since the early 80's in the formal language domain. As an example, we recall the early work of Gold [5] and Angluin [1].

One of the first original approaches to the identification of safe Petri nets was discussed by Hiraishi [6], who presented an algorithm for the construction of a free-labeled Petri net model from the knowledge of a finite set of its firing sequences. In a first phase, a language is identified in the form of a finite state automaton from given firing sequences. In a second phase, the dependency relation is extracted from the language, and the structure of a Petri net is guessed. Provided that the language is generated by a special class of net, the algorithm uniquely identifies the original net if a sufficiently large set of firing sequences is given.

A different approach is based on the *theory of regions* whose objective is that of deciding whether a given graph is isomorphic to the reachability graph of some free-labeled net and then constructing it. This problem has been solved in the literature for various types of nets ranging from elementary nets to Petri nets. The general principle for the synthesis is to inspect regions of the graph representing extensions of places of the candidate nets. An excellent

A. Giua and C. Seatzu are with the Department of Electrical and Electronic Engineering, University of Cagliari, Piazza D'Armi, 09123 Cagliari, Italy (giua,seatzu)@diee.unica.it.

0-7803-9568-9/05/$20.00 ©2005 IEEE

survey of this approach, that also presents some efficient algorithms for net synthesis based on linear algebra, can be found in the paper by Badouel and Darondeau [2].

Meda and Mellado [7], [8] have also presented an approach for the identification of free-labeled Interpreted Petri nets. Their approach consists in observing the marking of subset of places and, given some additional information on the dependency between transitions, allows one to reconstruct the part of the net structure related to unobservable places.

Bourdeaud'huy and Yim [4] have presented an approach to reconstruct the incidence matrix and the initial marking of a free-labeled net given some structural information on this net, such as the existence of P-invariants or T-invariants. This approach can also deal with positive examples of firing sequences but not with counterexamples. Unlike the approach we present in this paper, that is based on linear algebraic formalism, the approach of the authors is based on logic constraints.

Finally, in a recent paper Sreenivas [10] dealt with a related topic: the minimization of Petri net models. Given a λ-free labeled Petri net generator and a measure function — that associates to it, say, a non negative integer — the objective is that of finding a Petri net that generates the same language of the original net while minimizing the given measure. In our approach we are able to use as a performance index of the identification procedure some of the measures considered by Sreenivas, thus we can identify a minimal solution among all the possible ones. Note that the undecidability results proved by Sreenivas do not apply to our approach because we only ensure the identity between a given finite language and the set of finite prefixes of the synthesized net language.

II. Background on Petri nets

In this section we recall the formalism used in the paper. For more details on Petri nets we address to [9].

A *Place/Transition net* (P/T net) is a structure $N = (P, T, Pre, Post)$, where P is a set of m places; T is a set of n transitions; $Pre: P \times T \to \mathbb{N}$ and $Post: P \times T \to \mathbb{N}$ are the pre– and $post$– incidence functions that specify the arcs; $C = Post - Pre$ is the incidence matrix.

A *marking* is a vector $M: P \to \mathbb{N}$ that assigns to each place of a P/T net a non–negative integer number of tokens, represented by black dots. We denote $M(p)$ the marking of place p. A P/T *system* or *net system* $\langle N, M_0 \rangle$ is a net N with an initial marking M_0.

A transition t is enabled at M iff $M \geq Pre(\cdot, t)$ and may fire yielding the marking $M' = M + C(\cdot, t)$. We write $M\ [\sigma\rangle$ to denote that the sequence of transitions σ is enabled at M, and we write $M\ [\sigma\rangle\ M'$ to denote that the firing of σ yields M'. Note that in this paper we always assume that two or more transitions cannot simultaneously fire (non-concurrency hypothesis).

A marking M is *reachable* in $\langle N, M_0 \rangle$ iff there exists a firing sequence σ such that $M_0\ [\sigma\rangle\ M$. The *firing vector* of σ is denoted $\vec{\sigma}$. The set of all markings reachable from M_0 defines the *reachability set* of $\langle N, M_0 \rangle$ and is denoted $R(N, M_0)$.

Given a Petri net system $\langle N, M_0 \rangle$ we define its *free-language*[1] as the set of its firing sequences

$$L(N, M_0) = \{\sigma \in T^* \ \mid \ M_0[\sigma\rangle\}.$$

We also define the set of firing sequences of length less than or equal to $k \in \mathbb{N}$ as:

$$L_k(N, M_0) = \{\sigma \in L(N, M_0) \ \mid \ |\sigma| \leq k\}.$$

III. Logical constraints transformation

In this section we provide an efficient technique to convert logical *or* constraints into linear algebraic constraints, that is inspired by the work of Bemporad and Morari [3]. In particular, we consider two different cases: *inequality* constraints and *equality* constraints.

A. Inequality constraints

Let us consider the following constraint:

$$\bigvee_{i=1}^{r} \vec{a}_i \leq \vec{0}_n \tag{1}$$

where $\vec{a}_i \in \mathbb{R}^n$, $i = 1, \ldots, r$, and $\bigvee$ denotes the logical *or* operator.

Equation (1) can be rewritten in terms of linear algebraic constraints as:

$$\begin{cases} \vec{a}_1 \leq z_1 \cdot \vec{K} \\ \vdots \\ \vec{a}_r \leq z_r \cdot \vec{K} \\ z_1 + \ldots + z_r = r - 1 \\ z_1, \ldots, z_r \in \{0, 1\} \end{cases} \tag{2}$$

where $\vec{K}$ is any constant vector in $\mathbb{R}^n$ that satisfies the following relation

$$K_j > \max_{i \in \{1, \ldots, r\}} a_i(j), \quad j = 1, \ldots, n.$$

In fact, if $z_i = 0$ then the i-th constraint is active, while if $z_i = 1$ it is trivially verified, thus resulting in a redundant constraint. Moreover, the condition $z_1 + \ldots + z_r = r - 1$ implies that one and only one z_i is equal to zero, i.e., only one constraint is active. This means that $\vec{a}_i \leq \vec{0}_n$ for one i, while no condition is imposed for the other i's (in such cases the corresponding constraints may either be violated or satisfied). Obviously, analogous considerations can be repeated if the $\leq$ constraints in (1) are replaced by $\geq$ constraints.

[1] As it will appear in the next subsection, *free* specifies that no labeling function is assigned to the considered Petri net system.

B. Equality constraints

Let us now consider the constraint

$$\bigvee_{i=1}^{r} \vec{a}_i = \vec{b}_i \tag{3}$$

where $\vec{a}_i, \vec{b}_i \in \mathbb{R}^n$, $i = 1, \ldots, r$.

Equation (3) can be rewritten in terms of linear algebraic constraints as:

$$\begin{cases} \vec{a}_1 - \vec{b}_1 \leq z_1 \cdot \vec{K} \\ \vec{a}_1 - \vec{b}_1 \geq -z_1 \cdot \vec{K} \\ \vdots \\ \vec{a}_r - \vec{b}_r \leq z_r \cdot \vec{K} \\ \vec{a}_r - \vec{b}_r \geq -z_r \cdot \vec{K} \\ z_1 + \ldots + z_r = r - 1 \\ z_1, \ldots, z_r \in \{0, 1\} \end{cases} \tag{4}$$

where $\vec{K}$ is any constant vector in $\mathbb{R}^n$ such that

$$K_j > \max_{i \in \{1,\ldots,r\}} |a_i(j) - b_i(j)|, \quad j = 1, \ldots, n.$$

Repeating a similar reasoning as in the previous case, we can immediately observe that, if $z_i = 0$ then

$$\begin{cases} \vec{a}_i - \vec{b}_i \leq \vec{0}_n \\ \vec{a}_i - \vec{b}_i \geq \vec{0}_n \end{cases} \quad \Rightarrow \quad \vec{a}_i = \vec{b}_i.$$

On the contrary, if $z_i = 1$ then

$$\begin{cases} \vec{a}_i - \vec{b}_i \geq \vec{K} \\ \vec{a}_i - \vec{b}_i \leq -\vec{K} \end{cases}$$

that are trivially verified, i.e., they are redundant constraints. Finally, the condition on the sum of z_i's imposes that one constraint is active, i.e., $\vec{a}_i = \vec{b}_i$ for at least one $i \in \{1, \ldots, r\}$.

IV. Free-labeled Petri nets

The problem we consider in this paper can be formally stated as follows.

Problem 4.1: Let $\mathcal{L} \subset T^*$ be a finite prefix-closed language[2], and

$$k = \max_{\sigma \in \mathcal{L}} |\sigma|$$

be the length of the longest string in $\mathcal{L}$. Chosen a set of places P of cardinality m, we want to identify the structure of a net $N = (P, T, Pre, Post)$ and an initial marking M_0 such that

$$L_k(N, M_0) = \mathcal{L}.$$

The unknowns we want to determine are the elements of the two matrices $Pre,\ Post \in \mathbb{N}^{m \times n}$ and the elements of the vector $M_0 \in \mathbb{N}^m$. ■

A solution to the above identification problem can be computed thanks to the following theorem, that provides a linear algebraic characterization of the place/transition nets with m places and n transitions such that $L_k(N, M_0) = \mathcal{L}$.

Theorem 4.2: A net system $\langle N, M_0 \rangle$ is a solution of the identification problem (4.1) if and only if it satisfies the following set of linear algebraic constraints

$\mathcal{G}(\mathcal{E}, \mathcal{D}) \triangleq$

$$\begin{cases} M_0 + Post \cdot \vec{\sigma} - Pre \cdot (\vec{\sigma} + \vec{\varepsilon}_j) \geq \vec{0} \\ \qquad\qquad \forall (\sigma, t_j) \in \mathcal{E} & (a) \\ -KS_{\sigma,j} + M_0 + Post \cdot \vec{\sigma} \\ \quad -Pre \cdot (\vec{\sigma} + \vec{\varepsilon}_j) \leq -\vec{1}_m \qquad \forall (\sigma, t_j) \in \mathcal{D} & (b) \\ \vec{1}^{\,T} S_{\sigma,j} \leq m - 1 \qquad \forall (\sigma, t_j) \in \mathcal{D} & (c) \\ M_0 \in \mathbb{N}^m & (d) \\ Pre, Post \in \mathbb{N}^{m \times n} & (e) \\ S_{\sigma,j} \in \{0, 1\}^m & (f) \end{cases} \tag{5}$$

where

$$\mathcal{E} = \{(\sigma, t_j) \mid \sigma \in \mathcal{L}, |\sigma| < k, \sigma t_j \in \mathcal{L}\} \tag{6}$$

and

$$\mathcal{D} = \{(\sigma, t_j) \mid \sigma \in \mathcal{L}, |\sigma| < k, \sigma t_j \notin \mathcal{L}\}. \tag{7}$$

Proof:

- Assume that $t_j \in \mathcal{L}$. Then transition t_j is enabled at M_0 and the following relation must hold: $M_0 \geq Pre(\cdot, t_j)$. This relation can be rewritten as

  $$M_0 - Pre \cdot \vec{\varepsilon}_j \geq \vec{0}_n$$

  where $\vec{\varepsilon}_j$ is the j-th canonical basis vector.
  Generalizing, assume that $\sigma t_j \in \mathcal{L}$. Then transition t_j is enabled from the marking $M_\sigma = M_0 + (Post - Pre) \cdot \vec{\sigma}$ and the following relation must hold

  $$M_\sigma \geq Pre(\cdot, t_j).$$

  This relation can be rewritten as

  $$M_0 + Post \cdot \vec{\sigma} - Pre \cdot (\vec{\sigma} + \vec{\varepsilon}_j) \geq \vec{0}_n. \tag{8}$$

- Assume that $t_j \notin \mathcal{L}$. Then transition t_j is not enabled from M_0, and for at least one place p_i it must hold:

  $$M_0(p_i) < Pre(p_i, t_j).$$

  We want now to define a vector

  $$S_{\varepsilon,j} = \begin{bmatrix} s_1 \\ \vdots \\ s_m \end{bmatrix} \in \{0, 1\}^m,$$

  such that for all $i = 1, \ldots, m$ it holds

  $$s_i = 0 \quad \Longrightarrow \quad M_0(p_i) < Pre(p_i, t_j).$$

  Assume that each component of M_0 is less or equal to K. Then the i-th component of $S_{\varepsilon,j}$ (for $i = 1, \ldots, m$) must satisfy the equation

  $$-Ks_i + M_0(p_i) - Pre(p_i, t_j) < 0, \tag{9}$$

[2] A language $\mathcal{L}$ is said to be *prefix-closed* if for any string $\sigma \in \mathcal{L}$, all prefixes of σ are in $\mathcal{L}$.

so that if $s_i = 0$ it must hold $M_0(p_i) - Pre(p_i, t_j) < 0$, while if $s_i = 1$ equation (9) is trivially verified. In vector form (and taking into account that all variables are integers) this equation rewrites

$$-KS_{\varepsilon,j} + M_0 - Pre \cdot \vec{\varepsilon}_j \leq -\vec{1}_m. \quad (10)$$

Finally, there exists *at least* a place that disables t_j if

$$\sum_{i=1}^{m} s_i \leq m - 1, \quad (11)$$

so that at least one s_i is null. In vector form this equation rewrites

$$\vec{1}^{\,T} S_{\varepsilon,j} \leq m - 1. \quad (12)$$

Generalizing, assume that $\sigma \in \mathcal{L}$ and $\sigma t_j \notin \mathcal{L}$. Then transition t_j is not enabled from the marking

$$M_\sigma = M_0 + (Post - Pre) \cdot \vec{\sigma}.$$

We now define a vector

$$S_{\sigma,j} = \begin{bmatrix} s_1 \\ \vdots \\ s_m \end{bmatrix} \in \{0,1\}^m,$$

such that for all $i = 1, \ldots, m$ it holds

$$s_i = 0 \quad \Longrightarrow \quad M_\sigma(p_i) < Pre(p_i, t_j).$$

If we assume that each component of M_σ is less than or equal to K, following the previous reasoning we can immediately see that there exists *at least* a place that disables t_j if the following equations are verified

$$-KS_{\sigma,j} + M_0 + Post \cdot \vec{\sigma} - Pre \cdot (\vec{\sigma} + \vec{\varepsilon}_j) \leq -\vec{1}_m \quad (13)$$

and

$$\vec{1}^{\,T} S_{\sigma,j} \leq m - 1. \quad (14)$$

Note that for determining the value of K it is not necessary that the net be K-bounded. In fact, since we are given a set of finite sequences $\mathcal{L}$ of length less than or equal to k, regardless of the value of σ it is sufficient to take a value

$$\begin{aligned} K &\geq \max_i M_0(p_i) + k \cdot \max_{i,j} Post(i,j) \\ &\geq \max_i M(p_i) + |\sigma| \cdot \max_{i,j} Post(i,j) \\ &\geq \max_i M_\sigma(p_i). \end{aligned}$$

□

In general the set (5) is not a singleton, thus there exist more than one Petri net system $\langle N, M_0 \rangle$ such that $L_k(N, M_0) = \mathcal{L}$. To select one among these Petri net systems we choose a given performance index and solving an appropriate IPP we determine a Petri net system that minimizes the considered performance index[3]. In particular, if $f(M_0, Pre, Post)$ is the considered performance index, an identification problem can be formally stated as follows.

[3]Clearly, also in this case the solution may be not unique.

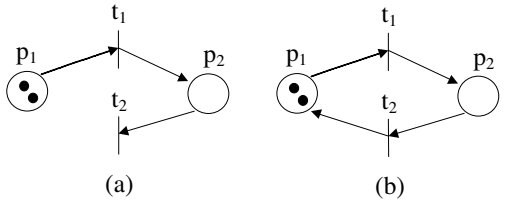

Fig. 1. The Petri net system of Example 4.4 (a); the Petri net of the same example when the additional constraint $m_1 + m_2 = \text{const}$ is added.

Problem 4.3: Let us consider the identification problem (4.1) and let $f(M_0, Pre, Post)$ be a given performance index. The solution to the identification problem (4.1) that minimizes $f(M_0, Pre, Post)$ can be computed by solving the following IPP

$$\begin{cases} \min & f(M_0, Pre, Post) \\ \text{s.t.} & \mathcal{G}(\mathcal{E}, \mathcal{D}). \end{cases} \quad (15)$$

■

As an example, assume we want to determine a Petri net system that minimizes the sum of the tokens in the initial marking and of the arc weights. In such a case we choose

$$f(M_0, Pre, Post) = \vec{1}_m^T \cdot M_0 + \vec{1}_m^T \cdot (Pre + Post) \cdot \vec{1}_n.$$

Example 4.4: Let $\mathcal{L} = \{\varepsilon, t_1, t_1t_1, t_1t_2, t_1t_1t_2, t_1t_2t_1\}$ and $m = 2$, thus $k = 3$. Assume that we want to determine the Petri net system that minimizes the sum of initial tokens and all arcs such that $L_3(N, M_0) = \mathcal{L}$. This requires the solution of an IPP of the form (15) where

$$\mathcal{E} = \{(\varepsilon, t_1), (t_1, t_1), (t_1, t_2), (t_1t_2, t_1), (t_1t_1, t_2)\}$$

and

$$\mathcal{D} = \{(\varepsilon, t_2), (t_1t_2, t_2), (t_1t_1, t_1)\}.$$

The procedure identifies a net system with

$$Pre = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \quad Post = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}, \quad M_0 = \begin{bmatrix} 2 \\ 0 \end{bmatrix}$$

namely the net system in Fig. 1.a. ■

A. Complexity of IPP (15)

Let n be the cardinality of T, k the length of the longest string in $\mathcal{L}$, and ν_r (for $r = 0, \ldots, k$) the number of strings in $\mathcal{L}$ of length r.

Then the constraint set (5) contains $\sum_{r=1}^{k} \nu_r$ constraints of type (a) and $\sum_{r=0}^{k-1}(n\nu_r - \nu_{r+1})$ constraints of type (b) and of type (c). The total number of scalar constraints is thus:

$$m\left(\sum_{r=1}^{k} \nu_r\right) + (m+1)\left(\sum_{r=0}^{k-1}(n\nu_r - \nu_{r+1})\right).$$

The total number of unknown is

$$u = m + 2(m \times n) + m\left(\sum_{r=0}^{k-1}(n\nu_r - \nu_{r+1})\right).$$

Note that given a value of k and of n, it is possible to find a worst case bound for $\rho = \sum_{r=0}^{k-1}(n\nu_r - \nu_{r+1})$. In fact, it holds:

$$\begin{aligned}\rho &= \textstyle\sum_{r=0}^{k-1}(n\nu_r - \nu_{r+1}) \\ &= \nu_0 + (n-1)\left(\textstyle\sum_{r=1}^{k-1}\nu_r\right) - \nu_k \\ &= n + (n-1)\left(\textstyle\sum_{r=1}^{k-1}\nu_r\right) - \nu_k.\end{aligned}$$

This expression is maximized if we assume $\nu_k = 0$ while all other ν_r must take the largest value, i.e., $\nu_r = n^r$. Hence we have

$$\rho = n + (n-1)(n + \cdots + n^{k-1}) = n^k,$$

and the total number of unknowns in the worst case is

$$\begin{aligned}u &= m + 2(m \times n) + m\ n^k \\ &= m(1 + 2n + n^k) \\ &= \mathcal{O}(m\ n^k),\end{aligned}$$

i.e., it has exponential complexity with respect to k.

B. Linear relaxation of integer programming

The main drawback of the proposed procedure is that it requires solving an integer programming problem to identify the net system. For large values of m, n and in particular of k, the solution of this problem may be computational demanding, thus reducing the usefulness of the proposed approach. The most natural way to overcome this difficulty consists in the linear relaxation of some constraints in the integer programming problem (15). In particular, we may relax the constraints $M_0 \in \mathbb{N}^m$ and $Pre,\ Post \in \mathbb{N}^{m\times n}$ into $M_0 \in (\mathbb{R}_0^+)^m$ and $Pre,\ Post \in (\mathbb{R}_o^+)^{m\times n}$.

In most of the cases we have considered the relaxation of the constraints on M_0, Pre and $Post$ does not affect the admissibility of the solution.

On the contrary, if the integer constraints on the binary variables $S_{\sigma,j}$ are relaxed, in all the cases we considered we found out that only unfeasible solutions are obtained.

V. Structural constraints

In our approach it is also possible to force the net to satisfy some structural constraints. In particular, the presence of *P-invariant* (*P-decreasing*, *P-increasing*) and *T-invariant* (*T-decreasing*, *T-increasing*) can be imposed by simply adding appropriate linear constraints to the set $\mathcal{G}(\mathcal{E},\mathcal{D})$. As an example, assume that we want to determine a net system that satisfies the language specifications on $L_k(N, M_0)$ and such that $\vec{x} \in \mathbb{R}^m$ is a given P-invariant. To this aim, given a performance index $f(M_0, Pre, Post)$, we need to solve an IPP of the form

$$\left\{\begin{array}{lll}\min & f(M_0, Pre, Post) & \\ \text{s.t.} & \mathcal{G}(\mathcal{E},\mathcal{D}) & (a) \\ & \vec{x}^T(Post - Pre) = \vec{0}_n^T & (b)\end{array}\right. \tag{16}$$

Analogously, if our goal is that $\vec{x} \in \mathbb{R}^m$ be a P-increasing (P-decreasing) for the net, we simply need to replace the above constraint (b) with $\vec{x}^T(Post - Pre) > \vec{0}_n^T$ ($\vec{x}^T(Post - Pre) < \vec{0}_n^T$).

Finally, if we want $\vec{x} \in \mathbb{R}^n$ be a T-invariant (T-increasing, T-decreasing), constraint (b) of equation (16) becomes $(Post - Pre)\cdot\vec{x} = \vec{0}_m$ ($> \vec{0}_m$, $< \vec{0}_m$, respectively).

***Example** 5.1:* Let us consider again the case of Example 4.4 but assume we want the net to be conservative. In particular, we want $m_1 + m_2$ always keeps constant. To this aim we solve an IPP of the form (16) where $\vec{x} = [1\ \ 1]^T$. We identify a net system with

$$Pre = \left[\begin{array}{cc}1 & 0\\ 0 & 1\end{array}\right],\quad Post = \left[\begin{array}{cc}0 & 1\\ 1 & 0\end{array}\right],\quad M_0 = \left[\begin{array}{c}2\\ 0\end{array}\right]$$

namely the net in Fig. 1.b. ■

Adding appropriate constraints to problem (15) we can also impose that the net should have a particular structure, e.g., it should be *ordinary*, a *marked graph* or a *state machine*.

In the first case we need to impose that $Pre, Post \in \{0,1\}^{m\times n}$.

In the second case, being a marked graph an ordinary Petri net such that each place has exactly one input and one output transition, we need to force the following additional constraints

$$\left\{\begin{array}{l}Pre\cdot\vec{1}_n = 1\\ Post\cdot\vec{1}_n = 1.\end{array}\right.$$

Finally, being a state machine an ordinary Petri net where every transition has exactly one input and one output place, the following additional constraints should be verified

$$\left\{\begin{array}{l}\vec{1}_m^T\cdot Pre = 1\\ \vec{1}_m^T\cdot Post = 1.\end{array}\right.$$

A *structural decomposition* of the net in a given number r of subnets can similarly be imposed. Let

$$P = P_1 \cup P_2 \cup \ldots \cup P_r$$

be a given partition of P. Assume that for all $t \in T$ we are given a set $\Pi(t) \subset \{1,\ldots,r\}$ such that $q \in \Pi(t)$ implies ${}^\bullet t^\bullet \cap P_q = \emptyset$. In plain words, $\Pi(t)$ represents the set of indices of P_q's such that t has no input/output arc from/to a place in P_q. This can be imposed solving an IPP of the form (16) where constraint (b) is replaced by the following set of linear constraints for all $t \in T$:

$$\sum_{q\in\Pi(t)}\sum_{p\in P_q}(Pre(p,t) + Post(p,t)) = 0.$$

VI. Synthesis of a Petri net system from its reachability graph

In this section we assume that the net system we want to synthesize is bounded, and the language it generates is given in terms of its reachability graph that is represented as a finite state automaton $G = (Q, T, \delta, q_0)$ where Q is the set of states, the alphabet T is the set of transitions of the net, $\delta : Q \times T \to Q$ is the transition function, and q_0 is the initial state.

A cycle in the automaton identifies a repetitive sequence and we define the corresponding set of firing vectors, that are the T-invariants of the net, as

$$\Gamma(G) = \{\vec{y} \in \mathbb{N}^n \mid (\exists\, \sigma \in T^+)\ (\exists\, q \in Q)\ : \ \delta(q,\sigma) = q \ \wedge\ \vec{y} = \vec{\sigma}\} \quad (17)$$

while the set of minimal T-invariants is

$$\Gamma_{\min}(G) = \{\vec{y} \in \Gamma(G) \mid (\nexists\ \vec{y}\,' \lneq \vec{y})\ \vec{y}\,' \in \Gamma(G)\}. \quad (18)$$

Finally, we define the set of sequences that are generated by the automaton without passing through a cycle as

$$\mathcal{L}_T(G) = \{\sigma \in T^* \mid \forall\ u, v \preceq \sigma,\ u \neq v \Rightarrow \delta(q_0,u) \neq \delta(q_0,v)\} \subseteq \mathcal{L}(G), \quad (19)$$

where $\mathcal{L}(G)$ denotes the language generated by the automaton, and the subscript T denotes the words generated by its spanning tree with root q_0.

We consider the following problem.

***Problem** 6.1:* Let $G = (Q, T, \delta, q_0)$ be a given finite state automaton. Chosen a set of places P of cardinality m, we want to identify the structure of a net $N = (P, T, Pre, Post)$ and an initial marking M_0 such that $L(N, M_0) = \mathcal{L}(G)$. The unknowns we want to determine are the elements of the two matrices $Pre,\ Post \in \mathbb{N}^{m\times n}$ and the elements of the vector $M_0 \in \mathbb{N}^m$. ■

***Theorem** 6.2:* A net system $\langle N, M_0\rangle$ is a solution of the identification problem (6.1) if and only if it satisfies the following set of linear algebraic constraints

$$\begin{cases} \mathcal{G}(\mathcal{E}, \mathcal{D}) & (a) \\ (Post - Pre)\cdot \vec{y} = \vec{0} \quad \forall\ \vec{y} \in \Gamma_{\min}(G) & (b) \end{cases} \quad (20)$$

where $\mathcal{E} = \{(\sigma, t) \mid \sigma \in \mathcal{L}_T(G), \sigma t \in \mathcal{L}(G)\}$

and $\mathcal{D} = \{(\sigma, t) \mid \sigma \in \mathcal{L}_T(G), \sigma t \notin \mathcal{L}\}$.

Proof: For sake of brevity, we just give a sketch of the proof. Let us first consider a word $\sigma = \sigma' t$ with $\sigma' \in \mathcal{L}_T(G)$: since all constraints corresponding to this word are explicitly listed in (20), then the net solution of (20) generates σ if and only if $\sigma \in \mathcal{L}(G)$.

Consider now any word $\sigma = \sigma' t \sigma''$ where $\sigma' \in \mathcal{L}_T(G)$ and $\sigma' t \in \mathcal{L}(G) \setminus \mathcal{L}_T(G)$ (i.e., $\sigma' t$ passes though a cycle). The constraints corresponding to $\sigma' t$ are explicitly listed in (20), and if we denote u the sequence obtained by $\sigma' t$ removing the cycle, it is easy to see that $\sigma' t \sigma''$ is generated by the net if and only if $u\sigma''$ is generated. □

***Example** 6.3:* Let us consider the finite state automaton G in Fig. 2.a. It holds $\Gamma_{\min}(G) = \{[1\ 1]^T\}$ and $\mathcal{L}_T(G) = \{\varepsilon, t_1, t_1 t_1\}$ thus $\mathcal{E} = \{(\varepsilon, t_1), (t_1, t_1), (t_1, t_2), (t_1 t_1, t_2)\}$ and $\mathcal{D} = \{(\varepsilon, t_2), (t_1 t_1, t_1)\}$. Now, assume that we want to determine the Petri net system that minimizes the sum of initial tokens and all arcs.

For $m = 1$ we get no feasible solution, while for $m = 2$ we found the net system in Fig. 1.b whose reachability graph is shown in Fig. 2.b. Note that in this particular case the reachability graph of the net is isomorphic to the given automaton G. ■

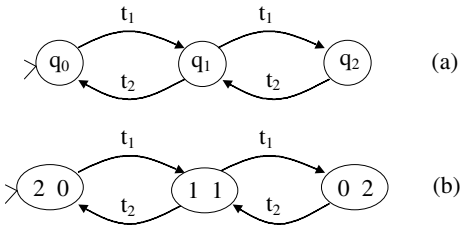

Fig. 2. The finite state automaton G of Example 6.3.

ACKNOWLEDGEMENTS

We thank the anonymous referees for their useful comments and suggestions that we have incorporated in the present version.

VII. CONCLUSIONS AND FUTURE WORK

In this paper we provided a solution to the problem of identifying a Petri net system that generates a given language, that is based on the solution of appropriate IPP. In particular, the case of free-labeled Petri net systems is considered. The problem of imposing structural constraints on the net is also addressed.

Our future work in this topic will be twofold. First, we want to extend the proposed approach to the case of λ-free labeled Petri nets, i.e., nets where two ore more transitions may share the same label. Secondly, we plan to derive appropriate heuristics in order to overcome problems related to the computational complexity.

REFERENCES

[1] D. Angluin. Inference of reversible languages. *Journal of the ACM*, 29(3):741–765, 1982.

[2] E. Badouel and P. Darondeau. Theory of regions. *Lecture Notes in Computer Science: Lectures on Petri Nets I: Basic Models*, 1491:529–586, 1998.

[3] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics and constraints. *Automatica*, 35(3):407–429, 1999.

[4] T. Bourdeaud'huy and P. Yim. Synthèse de réseaux de Petri à partir d'exigences. In *Actes de la 5me conf. francophone de Modélisation et Simulation*, pages 413–420, Nantes, France, September 2004.

[5] E. Mark Gold. Complexity of automaton identification from given data. *Information and Control*, 37(3):302–320, 1978.

[6] 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.

[7] M.E. Meda-Campaña and E. López-Mellado. Incremental synthesis of Petri net models for identification of discrete event systems. In *Proc. 41th IEEE Conf. on Decision and Control*, pages 805–810, Las Vegas, Nevada USA, December 2002.

[8] M.E. Meda-Campaña and E. López-Mellado. Required event sequences for identification of discrete event systems. In *Proc. 42th IEEE Conf. on Decision and Control*, pages 3778–3783, Maui, Hawaii, USA, December 2003.

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

[10] R.S. Sreenivas. On minimal representations of Petri net languages. In *Proc. WODES'02: 6th Work. on Discrete Event Systems*, pages 237–242, Zaragoza, Spain, October 2002.

[11] J.H. van Schuppen. System theory for system identification. *Journal of Econometrics*, 118(1-2):313–339, January-February 2004.