Computational complexity analysis of a Petri net identification procedure

Maria Paola Cabasino, Alessandro Giua, Carla Seatzu

Department of Electrical and Electronic Engineering, University of Cagliari
Piazza d'Armi, Cagliari 09123, Italy
Email: (cabasino,giua,seatzu)@diee.unica.it

Abstract—In previous papers we presented an approach to identify a Petri net system, given its language, based on the solution of an integer programming problem.

In this paper we analyze the complexity of such an approach in terms of computational time required to get an admissible solution, that may also be optimal according to a given performance criterion.

1. Introduction

This paper is based on our previous results in [1, 2]. In particular, in [1] we proposed a linear algebraic characterization of the Petri net systems that are able to generate a given finite set of strings, assuming that the number of places is known a priori. We also shown how this characterization can be used to synthesize a *bounded* net system whose language is given in terms of its *reachability graph*, namely imposing an appropriate set of finite length strings plus the set of minimal T-invariants.

In [2] we shown how the assumption that the number of places is given can be relaxed, and the approach can be generalized to the case in which the number of places is only known to be less or equal to a given value. In both cases, if a performance index is considered, the above linear algebraic characterization can be used to formulate a linear integer programming problem (IPP), that can be solved using ILOG CPLEX.

The goal of this paper is that of analyzing the computational complexity of the above approach. In particular, we want to investigate how the computational time depends on the cardinality of the set of finite length strings that describe the language, and on the chosen performance index.

To this aim we consider the language generated by a particular Petri net system that models a sender-receiver process. Different cases are examined with different number of places and transitions, and thus different languages generated.

The numerical simulations we carried out enabled us to conclude that the computational time become prohibitive for languages that are described by a large number of finite length strings, if we want to determine a solution that is *optimal* with respect to a given performance index. On the contrary, computational times are negligible if we limit to consider any *admissible* net system, e.g., the first admissible solution computed by CPLEX when solving the optimization problem. We believe that this is not a drawback of our procedure because in effect, when solving identification problems like this, the main requirement is that of determining an *admissible* solution, not necessarily an optimal one.

2. Background of Petri nets

In this section we recall the formalism used in the paper. For more details on Petri nets see [3].

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.

Paper presented at *NOLTA'06: 2006 Int. Symp. on Nonlinear Theory and its Applications* (Bologna, Italy), Sep 2006.

A *marking* is a vector $M : P \rightarrow \mathbb{N}$ 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 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 language 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: $L_k(N, M_0) = \{\sigma \in L(N, M_0) \mid |\sigma| \leq k\}$.

3. The identification procedure

In [1] we considered the following problem.

Problem 3.1 *Assume we are given a set of places $P = \{p_1, \ldots, p_m\}$ and a set of transitions $T = \{t_1, \ldots, t_n\}$. Let $\mathcal{L} \subset T^*$ be a finite prefix-closed language[1] over T, and $k = \max_{\sigma\in\mathcal{L}} |\sigma|$ be the length of the longest string in $\mathcal{L}$.*

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 $Pre = \{e_{i,j}\} \in \mathbb{N}^{m\times n}$ and $Post = \{o_{i,j}\} \in \mathbb{N}^{m\times n}$ and the elements of the vector $M_0 = \begin{bmatrix} m_{0,1} & m_{0,2} & \cdots & m_{0,m}\end{bmatrix}^T \in \mathbb{N}^m$. ■

In [1] we proved that a solution to the above identification problem can be computed thanks to the following theorem.

Theorem 3.2 *[1] A solution to the identification problem (3.1) satisfies the following set of linear algebraic constraints*

$$\mathcal{G}(m, T, \mathcal{L}) \triangleq$$

$$\left\{\begin{array}{lll} M_0 + Post\cdot\vec{\sigma} - Pre\cdot(\vec{\sigma}+\vec{\varepsilon}_j) \geq \vec{0} & & \\ \qquad\qquad\qquad\qquad \forall(\sigma, t_j) \in \mathcal{E} & & (a) \\ -KS(\sigma, t_j) + M_0 + Post\cdot\vec{\sigma} & & \\ \quad -Pre\cdot(\vec{\sigma}+\vec{\varepsilon}_j) \leq -\vec{1}_m & \forall(\sigma, t_j) \in \mathcal{D} & (b) \\ \vec{1}^{\,T} S(\sigma, t_j) \leq m-1 & \forall(\sigma, t_j) \in \mathcal{D} & (c) \\ M_0 \in \mathbb{N}^m & & (d) \\ Pre, Post \in \mathbb{N}^{m\times n} & & (e) \\ S(\sigma, t_j) \in \{0,1\}^m & & (f) \end{array}\right. \qquad (1)$$

where

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

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

and K is a very large constant.

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

Constraints (a) are the *enabling constraints*, i.e., a transition t_j is enabled at $M_0 + (Post - Pre) \cdot \vec{\sigma}$ if and only if $M_0 + (Post - Pre) \cdot \vec{\sigma} \geq Pre \cdot \vec{\varepsilon}_j$.

Constraints (b) and (c) are the *disabling constraints*: if a transition t_j is disabled at $M_0 + (Post - Pre) \cdot \vec{\sigma}$ then there exists at least one place $p \in P$ such that

$$M_0(p) + (Post(p, \cdot) - Pre(p, \cdot)) \cdot \vec{\sigma} \leq Pre(p, \cdot) \cdot \vec{\varepsilon}_j - 1. \tag{2}$$

Indeed, by constraint (c) at least one entry of $S(\sigma, t_j)$ is null, thus eq. (2) holds for at least one $p \in P$. On the contrary, no constraint is given for the other places to which it correspond a non null entry of $S(\sigma, t_j)$ because in this case constraint (b) is redundant.

In general the set (1) is not a singleton, thus there exists 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[2]. In particular, if $f(M_0, Pre, Post)$ is the considered performance index, an identification problem can be formally stated as follows.

Problem 3.3 *Let us consider the identification problem (3.1) and let $f(M_0, Pre, Post)$ be a given performance index. The solution to the identification problem (3.1) that minimizes $f(M_0, Pre, Post)$ can be computed by solving the following IPP*

$$\left\{ \begin{array}{ll} \min & f(M_0, Pre, Post) \\ s.t. & \mathcal{G}(m, T, \mathcal{L}). \end{array} \right. \tag{3}$$

■

3.1. Complexity of IPP (3)

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 (1) 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.

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

3.2. Optimizing the number of places

If we assume the number m of places is not given, but it is only known to be less or equal to a given value, the identification problem 3.1 can be reformulated as follows.

Problem 3.4 *Let us consider an identification problem in the form 3.1 where m is only known to be less or equal to a given value $\bar{m}$, and let $f(m, M_0, Pre, Post)$ be a given performance index. The solution to the identification problem that minimizes $f(m, M_0, Pre, Post)$ with the smallest number of places can be computed solving the following nonlinear IPP*

$$\begin{cases} \min\limits_{m \le \bar{m}} & \min \quad f(m, M_0, Pre, Post) \\ s.t. & \qquad \mathcal{G}(m, T, \mathcal{L}). \end{cases} \tag{4}$$

A trivial solution to the above identification problem 3.4 consists in solving IPP of the form (3) for increasing values of m, until a feasible solution is obtained. Alternatively, the following result can be used.

Theorem 3.5 *[2] Solving the identification problem 3.4 is equivalent to solving the following IPP:*

$$\begin{cases} \min & J = \bar{K} \cdot \vec{1}_{\bar{m}}^T \vec{z} + f(\bar{m}, M_0, Pre, Post) \\ s.t. & \mathcal{G}(\bar{m}, T, \mathcal{L}) \\ & K \cdot \vec{z} - Pre \cdot \vec{1}_n - Post \cdot \vec{1}_n \ge \vec{0}_{\bar{m}} \\ & z_{i+1} \le z_i, \quad i = 1, \dots, \bar{m} - 1 \\ & \vec{z} \in \{0, 1\}^{\bar{m}} \end{cases} \tag{5}$$

for a sufficiently large constant $\bar{K}$.

In particular, let us denote as $\vec{z}^*$, $\bar{M}_0^*$, $\overline{Pre}^*$ and $\overline{Post}^*$ the solution of (5), and let m^* be the number of nonzero components of $\vec{z}^*$.

Let M_0^* be the vector obtained from $\bar{M}_0^*$ by only keeping its first m^* components. Analogously, let Pre^* and $Post^*$ be the matrices obtained from $\overline{Pre}^*$ and $\overline{Post}^*$, respectively, by only keeping their first m^* rows.

Then, m^*, M_0^*, Pre^*, $Post^*$ is a solution of the identification problem 3.4.

3.3. Identification from the reachability graph

As discussed in detail in [1] the above procedure can also be used to solve identification problems starting from the reachability graph of the net system, that is represented as a finite state automaton G, provided that the net system is bounded.

To this aim, it is sufficient to define $\mathcal{L}$ as the set of sequences that are generated by the automaton without passing through a cycle. Then, we need to impose as additional constraints the set of minimal T-invariants $\Gamma_{\min}(G)$ [1]. More precisely, for any minimal T-invariant $\vec{y} \in \Gamma_{\min}(G)$ we have to impose an additional constraint of the form

$$(Post - Pre)\vec{y}^T = \vec{0}_m^T.$$

4. Numerical simulations

Let us consider the Petri net system in Figure 1 consisting of $2(q+1)$ places and $2q$ transitions. It models a sender-receiver process. In particular, places p_i, with $i = 1, \dots q$, model the sender process; places p_i, with $i = q+1, \dots, 2q$, model the receiver process, and places p_i, with $i = 2q+1, 2q+2$, correspond to the communication channels between the two processes. When place p_1 is marked (as in Figure 1) the sender is ready to transmit a message. After the firing of t_1 the message is in the

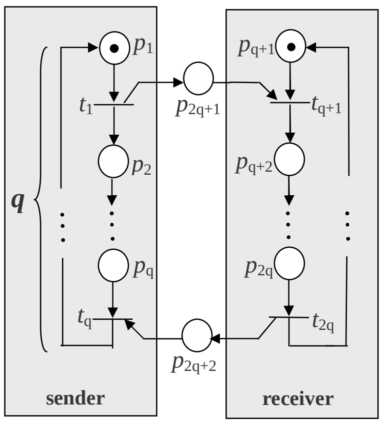

Figure 1: The sender-receiver process.

channel, ready to be received, provided that the receiver is also ready, namely that place p_{q+1} is marked. Now, transitions $t_2, \ldots, t_{q-1}, t_{q+1}, \ldots, t_{2q}$ can fire. In particular the firing of t_{2q} corresponds to the acknowledge from the receiver. Finally, the receipt from the sender is modeled by transition t_q.

In this section we present the results of various identification problems carried out considering different values of q, namely $q = 2, \ldots, 6$. In particular, our goal here is that of synthesizing a net system that generates the same language of the net system in Figure 1.

Note that for sake of brevity we do not report here the language $\mathcal{L} = L_k(N, M_0)$, but we limit to observe that $k = 2q$. Moreover, in order to obtain a net system that generates exactly the same language we also need to impose the minimal T-invariant $\vec{y} = \vec{1}_n$.

Note that we do not need to impose the P-invariants corresponding to the sender and the receiver. In fact, our requirement here is not that of obtaining exactly the net system in Figure 1, but a net system that generates the same language.

For any q we considered two different cases. In particular, using the notation of IPP (5), we assume:

(C1) $J = \bar{K} \cdot \vec{1}^T_{\bar{m}} \vec{z} + \vec{1}^T_{\bar{m}} \cdot M_0 + \vec{1}^T_{\bar{m}} \cdot (Pre + Post) \cdot \vec{1}_n$,

(C2) $J = \vec{1}^T_{\bar{m}} \cdot M_0$.

Therefore, in case (C1) we assume that the number of places is not known a priori, and our goal is that of determining among all the net systems that satisfy the given language specifications, that one who minimizes the number of places, the initial number of tokens and the arc weights.

In case (C2) we assign no weight on the number of places because we assume it is given. Our goal here is simply that of minimizing the number of tokens in the initial marking.

We carried out all numerical simulations using an appropriate tool we developed in MATLAB: given the language $\mathcal{L} = L_k(N, M_0)$, the structural constraints, and an upper bound on the number of places $\bar{m}$, it generates the set of constraints of IPP (5) in the syntax of CLPEX; then, given the desired performance index, the resulting IPP can be directly solved using ILOG CPLEX.

For both cases C1 and C2, and for any value of q, we limit the computational time to one hour. This constraint did not allow us to obtain the optimal solution in all cases examined.

Simulations have been run on a PC Athlon 64, 4000+ processor. Numerical results are summarized in Table 1.

In the first column we have reported the time (in seconds) and the number of iterations (between parenthesis) to find the first admissible solution.

In the second column we may have either an empty box or two numbers. The box is empty if CPLEX is not able to compute the optimal solution within an hour. If an optimal solution is determined within an hour, in the corresponding box we can read the time in seconds it took to compute it, and the number of iterations between parenthesis.

		First admissible solution	Optimal solution (max 1 hour)	% after 1 hour
q=2	**C1**	0,03 sec (141)	0,03 sec (141)	0%
	C2	<0,01 sec (113)	<0,01 sec (113)	0%
q=3	**C1**	< 4 sec (598)		16,71%
	C2	<0,6 sec (607)	0,78 sec (2744)	0%
q=4	**C1**	< 29 sec (1706)		59,91%
	C2	<8 sec (2479)		50,00%
q=5	**C1**	<200 sec (3855)		75,05%
	C2	<20 sec (3625)		75,00%
q=6	**C1**	< 65 sec (6949)		97,61%
	C2	<20 sec (6067)		50,00%

Table 1: Numerical results

In the third column we reported a measure of the distance (as a percentage) between the optimal solution and the solution computed within an hour. If such a value is equal to 0% it means that the optimal solution has been obtained in the allowed time. As large is the percentage, as far the solution is from the optimum.

From Table 1 we can be easily observe that the optimal solution can be computed within an hour only for $q = 2$ (both in case C1 and C2), and for $q = 3$ (in case C2). In all the other cases the number of constraints was too high, and regardless of the considered performance index, one hour was not enough to determine the optimal solution. In particular, the distance from the optimal solution in quite all cases examined also depend on the considered performance index (case C1 or C2).

Note however that this is not a serious limitation of our procedure because in general, when computing an identification problem, we are mainly interested in determining an admissible solution, rather than an optimal one. As it can be seen by looking at the first column of Table 1, the computational times are very very short also for large values of q if we consider an arbitrary solution, e.g., the first admissible one computed by CPLEX when solving an optimization problem.

Note that in case C1 we assumed that $\bar{m} = 2q + 2$ but we always found out a net with a minor number of places. This is not surprising because we are not imposing the P-invariants relative to the sender and the receiver, thus the structure of the net is different even if the language generated is the same.

5. Conclusions and future work

In this paper we investigated the computational complexity of solving Petri nets identification problems starting from the language generated. Our attention here was limited to bounded Petri net systems. Our future work will be that of extending the proposed procedure to unbounded nets. Moreover, we are working on the design of a MATLAB tool that starting from a given automaton G, identifies a net system whose coverabilty graph is isomorphic to G, by solving an appropriate ILL using CPLX.

6. Acknowledgements

We thank Mauro Franceschelli for his help in the development of he tool MATLAB for the generation of the IPP (5) in the syntax of CPLEX.

References

[1] A. Giua, C. Seatzu . *Identification of free-labeled Petri nets via integer programming.* In *Proc. IEEE 44rd Int. Conf. on Decision and Control (Siviglia), Dec 2005.*

[2] M. P. Cabasino, A. Giua, C. Seatzu . *Identification of deterministic Petri nets* . In *Proc. WODES'06, 8th Work. on Discrete Event Systems (Ann-Arbor,MI,USA), Jul 2006.*

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