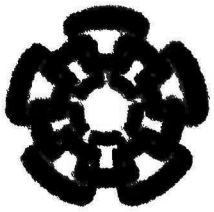
xx (108187.1)



Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional

Unidad Guadalajara

Identificación en Línea de Sistemas de Eventos Discretos: Fundamentos y Algoritmos para la Síntesis de Modelos en Redes de Petri

Tesis que presenta

María Elena Meda Campaña

Para obtener el grado de

Doctor en Ciencias

En la especialidad de

Ingeniería Eléctrica

Guadalajara, Jal., Noviembre de 2002

Identificación en Línea de Sistemas de Eventos Discretos: Fundamentos y Algoritmos para la Síntesis de Modelos en Redes de Petri

Tesis de Doctorado en Ciencias
Ingeniería Eléctrica

Por:

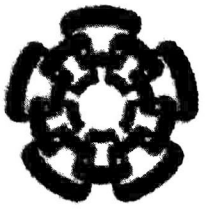
María Elena Meda Campaña

Ingeniera Industrial
Instituto Tecnológico de Culiacán
1991-1996
Maestra en Ciencias en Ingeniería Eléctrica
CINVESTAV-Unidad Guadalajara
1996-1998

Becaria del CONACyT, expediente no. 112941

Director de Tesis
Dr. Luis Ernesto López Mellado

CINVESTAV del IPN Unidad Guadalajara, Noviembre de 2002



Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional

Unidad Guadalajara

On-line Identification of Discrete Event Systems: Fundamentals and Algorithms for the Synthesis of Petri Net Models

Thesis submitted by

María Elena Meda Campaña

For the degree of

Doctor of Sciences

In the speciality of

Electrical Engineering

Guadalajara, Jal., November 2002

On-line Identification of Discrete Event Systems: Fundamentals and Algorithms for the Synthesis of Petri Net Models

Thesis of Doctor of Sciences
Electrical Engineering

By

María Elena Meda Campaña

Industrial Engineer
Instituto Tecnológico de Culiacán
1991-1996
Master of Science in Electrical Engineering
CINVESTAV-Unidad Guadalajara
1996-1998

Scholarship CONACyT no. 112941

Thesis Advisor
Dr. Luis Ernesto López Mellado

CINVESTAV del IPN Unidad Guadalajara, November 2002

Resumen

En esta tesis se presenta el problema de identificación de Sistemas de Eventos Discretos (SED), este problema consiste en determinar un modelo matemático a partir de la observación del comportamiento del sistema, el cual se describe por la evolución de sus señales de entrada y de salida. Los sistemas considerados en este trabajo son aquellos en los cuales su estado no puede ser determinado completamente a partir de la medición de su salida pero que exhiben la propiedad de evento detectabilidad por la salida. Las Redes de Petri (RP) serán el formalismo utilizado para describir un SED.

El esquema de identificación adoptado en este trabajo es un esquema pasivo en el que las señales de entrada no pueden ser manipuladas; las estrategias de identificación propuestas son adaptadas para la operación en línea calculando en forma progresiva un nuevo modelo cada vez que un nuevo comportamiento del sistema es detectado. Cada secuencia de eventos calculada a partir del comportamiento observado es analizada y si ésta aporta nueva información del sistema entonces el modelo calculado se actualiza; el comportamiento que genera el modelo actualizado es el mismo que el que ha sido observado del sistema.

El procedimiento de síntesis del modelado realiza principalmente dos tareas: el cálculo de la parte medible del sistema y la inferencia de la parte no medible, la cual está relacionada con las dependencias formadas por lugares no medibles con respecto a las transiciones calculadas. La primera tarea se hace directamente a partir de la observación de las señales de salida del sistema, mientras que la segunda es una tarea más complicada debido a que se tienen que hacer conjeturas acerca de cómo los lugares no medibles del sistema están relacionados con las transiciones, esto se determina con diferentes evoluciones del sistema. Los algoritmos propuestos para actualizar los lugares no medibles son de complejidad lineal en el número de las secuencias de transiciones calculadas. El algoritmo general, el cual incorpora todos los algoritmos de actualización, también es ejecutado en tiempo polinomial.

Este trabajo es una primera aproximación del problema de identificación en SED y éste puede ser considerado como la base de futuros trabajos en el área, posiblemente orientados a la verificación de sistemas, hardware o software; también este trabajo puede extenderse a problemas de ingeniería de reversa.

Abstract

This tesis addresses the identification problem of Discrete Event Systems (DES); it is devoted to obtain a mathematical model from the observation of the system behavior, which is described by the evolution of its input and output signals. The class of systems dealt in this work are those in which the entire state of the system cannot be determined from the measurement of its outputs, however these systems exhibiting the event-detectability property. The formalism used to describe DES are the Petri nets (PN).

The adopted identification approach is a passive one in which the input signals of the system are not manipulated; the strategies proposed are translated into procedures executed on-line, building the models progressively as new information of system behavior is detected. These models represent the observed behavior of the system and they approach asymptotically to the actual model of the system. Every sequence of events computed from the observed system behavior is analyzed and if it provides new information on the system then the computed model is updated.

The model synthesis procedure performs mainly two tasks: the computation of the measurable part of the system and the inference of the non measurable part of the system, which is related with the dependencies formed by the non measurable places with respect to the computed transitions. The first task is made directly from the observation of the output system signals, while the second task, rather difficult, derived a more detailed study about the dependencies formed by a non measurable places into a model. The proposed algorithms to updated the non measurable places are of lineal complexity in the number of the transitions computed and the transition sequences detected. The general algorithm to update a model that incorporates all the updating procedures of non measurable places is also executed in polynomial time.

This work is a first approximation of the identification problem in DES and it can be considered as a basis for future works on the matter, possibly oriented towards the verification of systems, hardware or software, or it can be extended to address problems of reverse engineering.

Agradecimientos

Al Dr. Luis Ernesto López Mellado por la dirección y apoyo recibidos, por la confianza que tuvo en mí para continuar con este trabajo, por sus consejos y por su tiempo.

A los miembros del jurado:

Dra. Ofelia Begovich,
Dra. Xiaoou Li,
Dr. Antonio Ramírez,
Dr. Alejandro Malo y
Dr. Victor Larios

por su tiempo y por sus valiosas críticas que contribuyeron a que esta tesis resultara lo mejor posible.

Al Dr. Antonio Ramírez por el tiempo extra dedicado a la revisión de mi tesis y por sus acertados comentarios.

Al Dr. Jean-Luc Koning por su amabilidad al aceptar ser revisor externo de la tesis y por sus comentarios incluidos en su reporte.

Al Dr. Mengchu Zhou por su valiosa crítica sobre mi trabajo, su amabilidad y su gran disposición para gestionar mi estancia en el Instituto de Tecnología de Nueva Jersey.

A mis profesores por los conocimientos transmitidos.

A los doctores Ofelia Begovich, Félix Ramos, Bernardino Castillo, Edgar Sánchez, Federico Sandoval, Manuel Guzmán, Deni Torres, Javier Ruiz y José Luis Leyva por su amistad y sus valiosos consejos.

A Lupita, Aracely, Aurora, Tere, Jara, Toño, Don Martín y Patricio por su amistad, amabilidad y asistencia.

A mis amigos a quienes les debo muchos momentos de alegría, de quienes aprendí muchas cosas y a los que siempre recordaré.

A mi familia y a Santiago.

Contents

Chapter 1

Introduction

Inferring models from observations and studying their properties is really what science is about. The models ("hypothesis", "laws of nature", "paradigms", etc.) may be of more or less formal character, but they have the basic feature that they attempt to link observations into some pattern. System identification deals with the problem of building mathematical models of dynamical systems based on the observed behavior of the system. This thesis is devoted to the study of the identification problem of discrete event systems, a class of dynamics systems which evolve with respect to the occurrence of events instead of time. The identification strategy here introduced is named *asymptotic identification* and will consist in build a model for a system as it evolves.

1.1 Systems models

A system is an agent in which variables of different kinds interact and produce observable signals. The observable signals that are of interest are usually called outputs. The system is also affected by external stimuli; external signals that can be manipulated by an external agent are called inputs. Other signals are called disturbances and can be divided into those that are directly measured and those that are only observed through their influence on the output. The distinction between inputs and measured disturbances is often less important for the modelling process.

When it is dealt with a system, it is very useful to know how its variables are related to each other. Within a broad definition such an assumed relationship among observed signals it is called a model of the system. Clearly, models may come in various shapes and be phrased with varying degrees of mathematical formalism. The intended use will determine the degree of sophistication that is required to make the model purposeful.

In daily life many systems are dealt using mental models, which do not involve any mathematical formalization at all. An example of this kind of models is to drive a car.

For certain systems it is appropriated to describe their properties using numerical tables and/or plots. Such descriptions are called graphical models. Linear systems for example, can be uniquely described by their impulse or step responses or by their frequency functions. Graphical representation of these systems are widely used for various design purposes.

For more advanced applications, it may be necessary to use models that describe the relationships among the system variables in terms of mathematical expressions like difference or differential equations. This kind of models are called mathematical or analytical models. Mathematical models may be characterized by a number of adjectives (time continuous or time discrete, deterministic or stochastic, linear or non linear, etc.) meaning the type of difference or differential equation used. The use of mathematical models is inherent in all fields of engineering and physics. In fact, a major part of the engineering field deals with how to make good design based on mathematical models.

Basically, a model has to be constructed from the observed behavior of the system. The mental model of car steering dynamics, for example, is developed through driving experience. Graphical models are built from certain measurements. Mathematical models may be developed along two routes (or a combination of them). One route is to split up the system, into subsystems, whose properties are well understood from previous experience. This basically means that it is relied on earlier empirical work. These subsystems are then joined mathematically and a model of the whole system is obtained. This route is known as *modelling* and does not necessarily involve experimental of the actual system. The procedure of modelling is quite application dependent and often has its roots in traditional and specific techniques in a given application area. Basic techniques typically involve structuring of the process into block diagrams with blocks consisting of simple elements. The reconstruction of the system from these simple blocks is now increasingly being done by computer, resulting a software model rather than a mathematical model.

1.2 System identification

The other route to compute mathematical as well as graphical models is directly based on experimentation. Input an output signals from the system, are recorded and analyzed in order to infer a model. This route is

called *system identification.*
Nowadays there exist two kind of system identification methods each method is related with the available information of the system:

a) Parametric system identification. To use this method it is necessary to know the structure of the system, then the identification method consists in the estimation of its parameters. The estimators are considered from a statistic point of view. There exists a straightforward relation between the a priori information of the system and the method used to estimate the parameters. For example, to use the least square estimator it is assumed that the behavior (dynamics) of the system can be approached just with the knowledge of the system inputs and outputs, to use the Markov estimator it is required, besides the system input/output information, the covariance matrix of the noise, while to use the maximum likelihood estimator it is needed the density function of noise besides the system input/output information, etc.

b) Structural system identification. To use this method it is necessary to know the representation that it is desired to identify; it could be a transfer function or a state variable representation besides others. Also to use this approach it is required the input and output information of the system.

The system identification approach introduced in this thesis is in first instance a structural approximation in the sense that the representation of the system that we want to obtain is an interpreted Petri net, however the proposed method to compute the model is based on the behavioral information of the system, thus the model will be approaching to the actual model of the system (structurally) as new input/output information of the system is detected.

1.3 Discrete event systems identification

In essence this thesis is devoted to the study of system identification of a class of systems known as Discrete Event Systems (DES).
A DES is a system which its state space is numerable, possibly infinite. The paradigm of the difference or differential equations are not suitable to model this class of systems since their evolution is related with the occurrence of events instead of the time. The formal languages, temporal logic, minimax algebra and Petri nets are some of the formalisms used to describe this class of systems.
The problem addressed in this work is to build a model for a DES as it evolves from the observation of its input and output signals. A sequence of models is built in such way that the current model represents or describes the entire observed behavior of the system; so, every new computed model acquires more details than the previous one approaching to the actual model of the system; this strategy is called *asymptotic identification* [32][33][34][35]. The formalism adopted in this thesis to describe a DES is the Petri net (PN) formalisms, since PN can capture the main characteristics of this class of systems like concurrence, synchronization, causal relationships between events, mutual exclusions and decisions. Specially will be used the interpreted Petri nets (IPN) an extension to PN that relate the input and output system signals to the structure of a PN, adding to them a physical meaning.
The identification approach considered in this thesis is a *passive approach*, in which the model is built just from the observation of input and output system signals, the updating of a computed model (building a new one) is made when new information of the system is observed.

1.4 Thesis organization

The chapter 2 introduces the definition of the Interpreted Petri nets, the formalism used to describe DES and how they can be used in the identification problem.

Chapter 3 overviews the reported works on identification of DES including those that use PN as the model computed.

The chapter 4 introduces the problem to identify a DES just with the information of output system signals. This approach is named passive asymptotic identification. In this chapter is presented how to compute each new IPN model fulfilling that it will be more similar to the system than the previous model, this new model is built with new information of the system.

In chapter 5 it is presented the identification procedure resulting from the analysis presented in previous chapter. Also in this chapter is presented an illustrative example of the identification process.

The chapter 6 presents the conditions under which a DES can be identified. In this chapter it is introduced how need to be the input signal given to the system to allow its identification, also is presented how need to be the signals assigned to the transitions and places of the IPN representation of a DES to can be detected from the observed information of the system.

Finally the conclusions of this work are presented in chapter 7.

Chapter 2

Interpreted Petri nets

Petri nets are a graphical and mathematical modeling tool applicable to many systems, this formalism is used in this thesis to describe DES, since they are suitable to capture most of the own characteristics of the DES like concurrence, decisions, mutual exclusions, causal relationships besides others. Petri nets are also useful to analyze qualitative and quantitative properties of DES.

To describe a DES in this thesis will be used a class of Petri nets called Interpreted Petri nets, this class of nets are defined to relate the input and output system signals to a Petri net structure giving to it a physical meaning. This chapter summarizes concepts and notation of PN and IPN used in this work.

2.1 Introduction

There exists several formalisms for modelling and analyzing DES. One of these formalisms are the Petri nets (PN). PN can describe most of the own characteristics of DES such as decisions, synchronizations, mutual exclusions, concurrence, causal relationships besides others DES characteristics.
One of the advantages of the PN is that they have a mathematical and a graphical representation which made their use more friendlier. As a graphical tool PN can be used as flow charts or block diagrams. As a mathematical tool it is possible to establish a state equation governing the behavior of systems.
In this work it is used a class of PN named interpreted Petri nets (IPN). The IPN have the property to relate the input and output signals of the system to the structure of a PN giving to it a physical meaning.
In this chapter are introduced the basic concepts and properties of PN also it is introduced the definition of IPN and related concepts on DES modeling used in this work. Fine PN surveys can be found in [53][36][17].

2.2 Petri nets

A Petri net is a directed, weighted bipartite graph consisting of two kinds of nodes; the places and the transitions these items are graphically depicted as circles and bars respectively, the arcs are either from a place to a transition or from a transition to a place. Arcs are labeled with their weights (positive integers), where a k-weighted arc can be interpreted as the set of k parallel arcs. Label for unity weight are usually omitted. The formal definition of a PN is following presented.

Definition 2.1 *A Petri Net structure G is a directed, weighted digraph defined by the 4-tuple $G = (P, T, F, W)$ where*

- $P = \{p_1, p_2, ..., p_n\}$ *is a finite set of elements called places,*
- $T = \{t_1, t_2, ..., t_m\}$ *is a finite set of elements called transitions,*
- $F \subseteq \{(P \times T) \cup (T \times P)\}$ *is a set of arcs (flow relation) and*
- $W : F \longrightarrow \{\mathbb{Z}^+\}^n$ *is a weight function.*

Example. Consider the PN depicted on figure 2.1. The sets P, T and F and the function W are the following:

$P = \{p_1, p_2, p_3, p_4, p_5, p_6\}$

$T = \{t_1, t_2, t_3, t_4, t_5\}$

$F = (p_1, t_1), (p_2, t_2), (p_3, t_5), (p_4, t_3), (p_5, t_4), (p_6, t_5), (t_1, p_2), (t_2, p_3), (t_5, p_1), (t_3, p_5), (t_4, p_6), (t_5, p_4)\}$

$w(t_5, p_1) = 2$, $w(p_1, t_1) = 2$, $w(t_4, p_6) = 3$ and $w(p_6, t_5) = 3$, the weight of the remaining arcs is 1.

Definition 2.2 *Let G be a PN, the sets of input and output places of each transition in G and the sets of input and output transitions of each place in G are the following:*

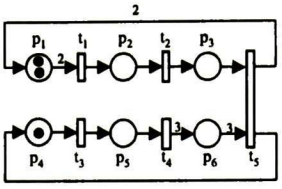


Figure 2.1: Graphical representation of a PN.

${}^\bullet t = \{p|(p,t) \in F\}$ *denotes the set of all input places of t.*

$t^\bullet = \{p|(t,p) \in F\}$ *denotes the set of all output places of t.*

${}^\bullet p = \{t|(t,p) \in F\}$ *denotes the set of all input transitions of p.*

$p^\bullet = \{t|(p,t) \in F\}$ *denotes the set of all output transitions of p.*

In the PN depicted on figure 2.1 ${}^\bullet t_1 = p_1$, $t_4^\bullet = p_6$, ${}^\bullet p_1 = t_5$ and $p_2^\bullet = t_2$.

Definition 2.3 *A transition t is called source transition if it does not have any input place, i.e.* ${}^\bullet t = \emptyset$.

A source transition is unconditionally enabled. The transition t_1 of the PN depicted on figure 2.2 is a source transition.

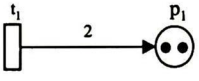


Figure 2.2: t_1 is a source transition.

Definition 2.4 *A transition t is called sink transition if it does not have any output place, i.e.* $t^\bullet = \emptyset$.

The firing of a sink transition remove tokens but does not produce any. The transition t_1 of the PN depicted on figure 2.3 is a sink transition.

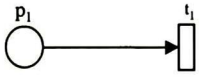


Figure 2.3: t_1 is a sink transition.

Definition 2.5 *A pair of a place p and a transition t is called self-loop if p is both an input and output place of t.*

The place p_1 and the transition t_1 of the PN depicted on figure 2.4 describe a self-loop.

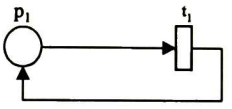


Figure 2.4: p_1 and t_1 form a self_loop.

Definition 2.6 *A PN is said to be pure if it has no self loops.*

The PN depicted on figure 2.5 is a non pure PN.

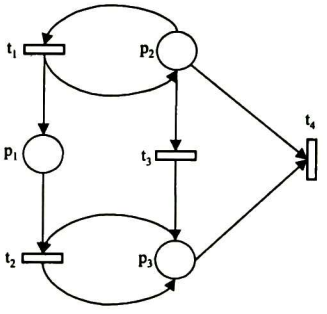


Figure 2.5: Non pure Petri net.

2.2.1 State equation of a PN

The behavior of a DES can be described in terms of its states and their changes, in a PN the states of a system can be represented by *markings*. Following are presented the basic notions and the needed concepts to describe the behavior of a DES using PN.

Incidence matrix of a Petri net

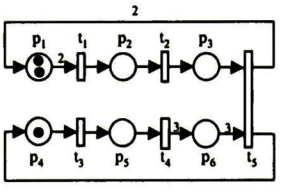
For a PN with n places and m transitions, the incidence matrix $C = [c_{ij}]$ is a $n \times m$ matrix. The entry c_{ij} is defined by:

$$c_{ij} = c_{ij}^{+} - c_{ij}^{-}$$

where $c_{ij}^{+} = w(t_j, p_i)$ is the weight of the arc from transition t_j to its output place p_i and $c_{ij}^{-} = w(p_i, t_j)$ is the weight of the arc to the transition t_j from its input place p_i.
Notice that c_{ij}^{-}, c_{ij}^{+} and c_{ij} represents the number of tokens removed, added and changed respectively of the place p_j when the transition t_j is fired.

Example. Consider the incidence matrix of the PN depicted on figure 2.6. The entry c_{11} is computed as $c_{11} = c_{11}^{+} - c_{11}^{-} = w(t_1, p_1) - w(p_1, t_1) = 0 - 2 = -2$, the entry c_{64} is computed as $c_{64} = c_{64}^{+} - c_{64}^{-} = w(t_4, p_6) - (p_6, t_4) = 3 - 0 = 3$ while the entry c_{32} is computed as $c_{32} = c_{32}^{+} - c_{32}^{-} = w(t_2, p_3) - w(p_3, t_2) = 1 - 0 = 1$.

An entry c_{ij} of an incidence matrix C also is denoted as $C[i,j]$.



$$C = \begin{bmatrix} -2 & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 3 & -3 \end{bmatrix}$$

Figure 2.6: Petri net with its incidence matrix

Marking of a PN

Definition 2.7 *The marking (state) $M : P \longrightarrow \{\mathbb{Z}^+\}^n$ of a PN is a mapping of each place to the nonnegative integers.*

By definition a marking (state) assigns to each place p a nonnegative integer k, then the place p is said to be marked with k tokens. Tokens are depicted as black dots inside a place p.
A marking M_k is represented by a $n \times 1$ column vector, where $n = |P|$. The $i-$th entry of M_k denoted as $M_k(p_i)$ is the number of tokens in the place p_i immediately after the $k-$th firing in some firing transition sequence.

> The marking of the PN depicted on figure 2.6 is $[\ 2\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$ since the place p_1 and p_3 have 2 and 1 tokens inside respectively.

Definition 2.8 *A Petri Net is the pair $N = (G, M_0)$, where G is a PN structure and M_0 is an initial token distribution (initial marking).*

> The marking $[\ 2\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$ is the initial marking of the PN depicted on figure 2.6.

A marking can be represented as a i-tuple such that i is the number of marked places and the entries are the number of tokens inside the marked places. For example the initial marking of the PN depicted on figure 2.6 can be described as $M_0 = [2p_1, p_4]$.

Definition 2.9 *The kth firing or control vector v_k is a $m \times 1$ column vector, where $m = |T|$. The jth entry of v_k is defined as* $v_k(t_j) = \begin{cases} 1 & \text{if } t_j \text{ fires at the kth firing} \\ 0 & \text{otherwise} \end{cases}$

If the transition t_3 of the PN depicted on figure 2.6 is the transition fired at the k-th firing of the PN, then $v_k = [\ 0\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$

Transition enabling and firing transition rules

A state or marking in a PN can change according to the following transition (firing) rule:

1. A transition t is said to be enabled if each input place p of t is marked with at least $w(p,t)$ tokens, where $w(p,t)$ is the weight of the arc from p to t.

2. An enabled transition may or may not fire (depending on whether or not the event actually take place)

3. A firing of an enabled transition t removes $w(p,t)$ tokens from each input place p of t and adds $w(t,p)$ tokens to each output place p of t, where $w(t,p)$ is the weight of the arc from t to p.

   Example. Consider the IPN G depicted on figure 2.6. The enabled transitions are the transitions t_1 and t_3 since all their input places are marked with the required number of marks, hence these two transitions can be fired. If the transition t_1 is fired, will be removed 2 marks from the place p_1 and will be added one mark to the place p_2. The new marking reached after the firing of t_1 will be $\begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 \end{bmatrix}^T$

State equation of a PN

Since the ith column of the incidence matrix C represents the change of marking as the result of firing the transition t_i then the state equation of a Petri net could be defined as:

$$M_{k+1} = M_k + Cv_k \tag{2.1}$$

The above equation states that if an enabled transition t_j is fired in a marking M_k, then a new marking M_{k+1} is reached. This fact can be represented as: $M_k \xrightarrow{t_j} M_{k+1}$.

Example. Consider N the PN depicted on figure 2.6, and $M_1 = \begin{bmatrix} 2 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}^T$ its marking. The marking M_2 is reached from the marking M_1 when the transition t_1 is fired, the computation of M_2 using the state equation 2.1 is as follows: $\overset{M_1}{\begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} + \overset{C}{\begin{bmatrix} -2 & 0 & 0 & 0 & 2 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 3 & -3 \end{bmatrix}} \overset{v_k}{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}} = \overset{M_1}{\begin{bmatrix} 2 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} + \overset{Cv_k}{\begin{bmatrix} -2 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}} = \overset{M_2}{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}}$.

Definition 2.10 *The reachability set* $\mathbf{R}(N) = \{M_k | M_0 \xrightarrow{\sigma} M_k\}$ *of a PN* N *is the set of all possible reachable markings from* M_0*, firing only enabled transitions.*

Example. Consider N the PN depicted on figure 2.6, the reachability set of N is:

$R(N) = \{[2p_1, p_4], [p_2, p_4], [2p_1, p_5], [p_3, p_4], [p_2, p_5], [2p_1, 3p_6], [p_3, p_5], [p_2, 3p_6], [p_3, 3p_6]\}$.

Definition 2.11 *The characteristic vector or Parikh vector of a transition sequence* σ *is a vector* $\overrightarrow{\sigma} \in \{\mathcal{N}^+\}^m$*, where* $m = |T|$*.* $\overrightarrow{\sigma}(i)$ *represents the number of firings of* t_i *in* σ*.*

Example. Consider the transition sequence $\sigma_1 = t_1t_2t_3t_4t_5t_3t_1t_2$ fired on the PN N depicted on figure 2.6, the Parikh vector of σ_1 is $\overrightarrow{\sigma_1} = \begin{bmatrix} 2 & 2 & 2 & 1 & 1 \end{bmatrix}$.

Notice that a Parikh vector does not provide information about the occurrence order of the transitions in a transition sequence.

2.2.2 Properties of PN

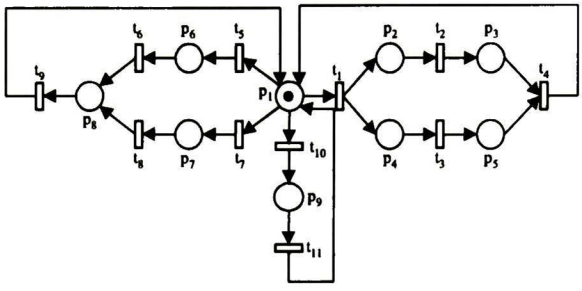


Figure 2.7: Petri net.

The incidence matrix C of the PN N depicted on figure 2.7 is:

$$C = \begin{bmatrix} -1 & 0 & 0 & 1 & -1 & 0 & -1 & 0 & 1 & -1 & 1 \\ 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

Definition 2.12 *A p-semiflow of a PN N is a nonnegative vector Y fulfilling $Y^T C = 0$. $\langle Y \rangle = \{p_i | Y(i) > 0\}$ is called the support of Y. A p-component is a subnet generated by the support of a p-semiflow Y considering also the input and output transitions of each place in $\langle Y \rangle$.*

Example. The vector $Y_1 = [\ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 1\ ]$ is a p-semiflow of the PN N depicted on figure 2.7, since $Y_1 C = 0$. The support of Y_1 is $\langle Y_1 \rangle = \{p_1, p_2, p_3, p_6, p_7, p_8, p_9\}$. The subnet N_1 generated by the p-semiflow Y_1 is depicted on figure 2.8.

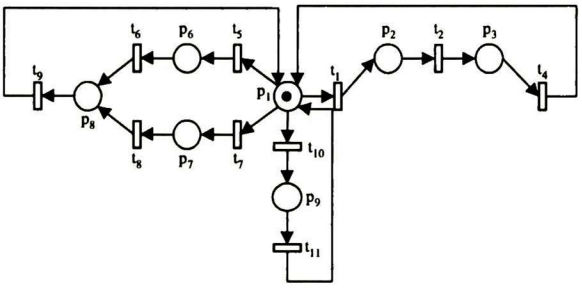


Figure 2.8: Subnet generated by the p-semiflow Y_1 of Q.

The other p-semiflow of N is $Y_2 = [\ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ ]$

Proposition 2.1 *Fundamental property of p-semiflows [17]. Let N be a Petri net and let Y be a p-semiflow of N. If $M_0 \xrightarrow{\sigma} M_k$, then $YM_k = YM_0$.*

Proof. Since M_k is reachable from M_0 with some firing sequence σ. By the state equation 2.1, $M_k = M_0 + Cv_k$. Therefore,

$$YM_k = YM_0 + YC\vec{\sigma} = YM_0$$

because $YC = 0$. ■

A p-semiflow induces a conservative component of a PN N, implying that the number of tokens in a p-component is preserved in each marking of a PN N.

Definition 2.13 *A t-semiflow of a PN N is a nonnegative vector X fulfilling $CX = 0$. $\langle X \rangle = \{t_i | X(i) > 0\}$ is called the support of X. A t-component is a subnet generated by the support of a t-semiflow X considering also the input and output places of each transition in $\langle X \rangle$.*

Example. The vector $X_1 = [\ 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ ]^T$ is a t-semiflow of the PN N depicted on figure 2.7. The support of X_1 is $\langle X_1 \rangle = \{t_1, t_2, t_3, t_4\}$. The subnet generated by X_1 is depicted on figure 2.9.

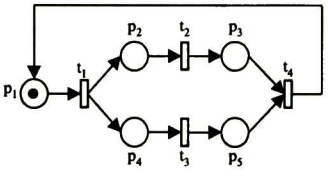


Figure 2.9: Subnet generated by the t-semiflow X_1 of Q.

The remaining t-semiflows of N are: $X_2 = [\ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$

$X_3 = [\ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$, $X_4 = [\ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ ]^T$ and $X_5 = [\ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 1\ \ 1\ ]^T$

Proposition 2.2 *Fundamental property of T-semiflows [17]. Let σ be a finite transition sequence of a Petri net N which is enabled at a marking M. Then the Parikh vector $\vec{\sigma}$ is a t-semiflow of N iff $M \xrightarrow{\sigma} M$ (i.e. the occurrence of σ reproduces the marking M).*

Proof. ($\Rightarrow$): Since σ is enabled at marking M, it is obtained that $M \xrightarrow{\sigma} M'$ for some marking M' By the state equation 2.1 $M' = M + C\vec{\sigma}$. As $\vec{\sigma}$ is a t-semiflow then $C\vec{\sigma} = 0$ and hence $M' = M$.
($\Leftarrow$): If $M \xrightarrow{\sigma} M$ then by the marking equation $C\vec{\sigma} = 0$. So $\vec{\sigma}$ is a t-semiflow of N. ■

A t-semiflow represents the repetitive components of a Petri net N describing its cyclic behavior.

2.2.3 Live and bounded Petri nets

Two important properties of a DES are boundedness and liveness. Boundedness is often interpreted as stability and it is used to identify the existence of overflows; while, liveness means that, for every event e_k in a system, it is always possible to reach a state, at which e_k can occur. Moreover, if a DES is live then it is deadlock-free. In PN terms, boundedness and liveness are defined as follows:

Definition 2.14 *A place p_i of a PN N is b-bounded, if $\forall M_k \in \mathbf{R}(N)$, $M_k(p_i) \leq b$, where b is an nonnegative integer. A PN is b-bounded, if all its places are b-bounded.*

Definition 2.15 *A transition t_k of a PN N is said to be live if $\forall M_j \in \mathbf{R}(N)$, $\exists \sigma$ such that $M_j \xrightarrow{\sigma} M_r$ and $t_k \in \sigma$. A PN is live if all its transitions are live.*

Another desired property of a DES is that it has a cyclic behavior. A SED has a cyclic behavior if there exists a sequence of events that allows to reach the initial state from any reachable state, it means that a task can be infinitely performed. In PN terms a cyclic behavior is defined as follows.

Definition 2.16 *A PN N is cyclic if $\forall M \in \mathbf{R}(N)$, $\exists \sigma$ such that $M \xrightarrow{\sigma} M_0$.*

2.3 Interpreted Petri nets

The formalism used in this thesis to describe DES are Interpreted Petri Nets (IPN). These nets are a particular case of PN in which the transitions and the places have a physical meaning associated, representing the input and output signals of the system.

Definition 2.17 *An Interpreted Petri Net is the 5-tuple $Q = (N, \Sigma, \Phi, \lambda, \varphi)$ where*

- $N = (G, M_0)$ *is a PN,*
- $\Sigma = \{\sigma_1, \sigma_2, ..., \sigma_r\}$ *is a finite set of elements σ_i called input symbols,*
- $\Phi = \{\phi_1, \phi_2, ..., \phi_p\}$ *is a finite set of elements ϕ_i called output symbols. In this work Φ is considered as the set of the nonnegative integer numbers $\mathbb{Z}$ i.e. $\Phi = \mathbb{Z}^+$*
- $\lambda : T \rightarrow \Sigma \cup \{\varepsilon\}$ *is a function that assigns an input symbol to each transition of the net, where ε represents an internal system event. This function has the following restriction: $\forall t_j, t_k \in T$, $j \neq k$ if $w(p_i, t_j) = w(p_i, t_k) \neq 0$ and both $\lambda(t_j)$, $\lambda(t_k) \neq \varepsilon$, then $\lambda(t_j) \neq \lambda(t_k)$, and*
- $\varphi : \mathbf{R}(N) \rightarrow \{\mathbb{Z}^+\}^q$ *is an output function; where q is the number of sensors associated to places in Q.*

The marking evolution of an IPN Q evolves according to the following rules:

- A transition t_j is enabled at marking M_k iff $\forall p_i \in P$, $M_k(p_i) \geqslant w(p_i, t_j)$.
- If $\lambda(t_j) = a_i \neq \varepsilon$ is present and t_j is enabled, then t_j must fire.
- If $\lambda(t_j) = \varepsilon$ and t_j is enabled then t_j can be fired.

If an enabled transition t_j is fired in a marking M_k, then a new marking M_{k+1} is reached. M_{k+1} can be computed using the PN state equation: $M_{k+1} = M_k + Cv_j$.

Definition 2.18 *A transition $t_i \in T$ is said to be **manipulable** if $\lambda(t_i) \neq \varepsilon$, and **non manipulable** if $\lambda(t_i) = \varepsilon$. Non manipulable transitions represent internal events of the system.*

Definition 2.19 *A place $p_i \in P$ is said to be **measurable** if it has a sensor signal assigned, and **non measurable** otherwise. Non measurable places are depicted as dark circles.*

In this work it is considered that the function $\varphi : \mathbf{R}(N) \rightarrow \{\mathbb{Z}^+\}^q$ is a linear function that can be represented as a $q \times n$ matrix $\varphi = [\varphi_{ij}]$ where n is the number of places, q is the number of measurable places, and the i-th row vector φ_i of φ is the transpose of the elemental vector e_j such that $e_j[i] = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$ if p_j is the i-th measurable place ($\varphi_i = e_j^T$), according to the order given by the place labeling. This implies that p_i is a measurable place if there exists an elemental vector e_j such that $e_j[i] = 1$ in matrix φ. Thus, a place p_i is measurable if the i-th column of φ matrix is a non null column $\varphi(\bullet, i) \neq 0$, and non measurable otherwise. Also, it is considered that a sensor signal assigned to any place is different to the sensor signal assigned to any other place.

The state equation can be completed as:

$$\begin{aligned} M_{k+1} &= M_k + Cv_k \\ y_k &= \varphi \cdot M_k \end{aligned} \tag{2.2}$$

Arranging the rows of the incidence matrix C of an IPN Q in measurable and non measurable places, the C matrix can be decomposed as $C = \begin{bmatrix} \varphi C \\ \gamma C \end{bmatrix}$, where γ is a linear function that can be represented by a $(n-q) \times n$ matrix, the i-th row vector γ_i of γ is the transpose of the elemental vector e_j ($e_j[i \neq j] = 0$, $e_j[j] = 1$), if p_j is the i-th non measurable place ($\gamma_i = e_j^T$), according to the order given by the place labeling.

Example. Consider as a physical system the water tank depicted on figure 2.10.*a*, the IPN Q depicted on figure 2.10.*b* is an IPN model describing this system.

The manipulable transitions are the transitions t_1 and t_2 due to the actions O ="open valve" and C ="close valve"are assigned to t_1 and t_2 respectively. The others transitions represent the internal events of the tank and hence they are non manipulable transitions. The λ function takes the following values: $\lambda(t_1) = O \quad \lambda(t_2) = C \quad \lambda(t_3) = \epsilon \quad \lambda(t_4) = \epsilon \quad \lambda(t_5) = \epsilon \quad \lambda(t_6) = \epsilon$

In this example the tank has two sensors measuring the low level (L) or high level (H) of the water in the tank, the model Q has 5 places, hence $q = 2$ and $n = 5$. The matrix representing the φ function is then defined as $\varphi = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$, while the matrix representing the γ function is $\gamma = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}$
Notice that the places having assigned a sensor signal are the places p_4 and p_5, hence the columns 4 and 5 of the φ matrix have an one (1), indicating that p_1 and p_4 are measurable places.

Consider the current marking $M_k = [\ 1\ \ 0\ \ 1\ \ 0\ \ 0\ ]^T$ of Q. The output symbol generating Q is $\varphi M_k = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$, due to $\overset{\varphi}{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \overset{M_k}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} = \overset{\varphi M_k}{\begin{bmatrix} 0 \\ 0 \end{bmatrix}}$, this fact implies that the measurable places are not marked. However when the valve is open; Q could reach the marking $M_i = [\ 0\ \ 1\ \ 0\ \ 1\ \ 0\ ]$, if M_i is reached then the output symbol generating Q will be $\varphi M_i = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$ which is computed as

$$\overset{\varphi}{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \overset{M_i}{\begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}} = \overset{\varphi M_i}{\begin{bmatrix} 1 \\ 0 \end{bmatrix}}.$$

The incidence matrix of Q is $C = \begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}$

The matrix $\varphi C = \overset{\varphi}{\begin{bmatrix} 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \overset{C}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}} = \overset{\varphi C}{\begin{bmatrix} 0 & 0 & -1 & 1 & 1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}}$.

Notice that φC is a submatrix of C, composed by the rows of C representing the measurable places, in this case φC is composed by the rows 4 and 5 of C because p_4 and p_5 are the measurable places of Q.

The matrix γC is a submatrix of C, composed by the rows of C representing the non measurable places, in this case γC is composed by the rows 1, 2 and 3 of C because p_1, p_2 and p_3 are the non measurable places of Q, hence $\gamma C = \overset{\gamma}{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix}} \overset{C}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 1 & 1 & -1 \\ 0 & 0 & 0 & -1 & 0 & 1 \end{bmatrix}} = \overset{\gamma C}{\begin{bmatrix} 1 & -1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & -1 & 0 \end{bmatrix}}$.

Consider a DES S_f in which all its states are completely measurable, by definition all the places of the system model Q describing S_f will be measurable places, hence the number q of sensors in Q is equal to n, i.e. $q = n$ (where n is the number of places of Q). It follows that for this class of systems an output symbol is an entire marking of Q since the output matrix φ is the $n \times n$ identity matrix then $\varphi M_k = M_k$ and $\varphi C = C$.

Definition 2.20 *A firing transition sequence of an IPN Q is a sequence $\sigma = t_i t_j ... t_k$ such that $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \cdots M_w \xrightarrow{t_k} M_x$. The set of all firing sequences is called the language of Q and it is denoted as $\pounds(Q) = \{\sigma \mid \sigma = t_i t_j ... t_k$ and $M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \cdots M_w \xrightarrow{t_k} M_x\}$.*

Definition 2.21 *The input language of an IPN Q is defined as $\pounds_{in}(Q) = \{\lambda(t_i)\lambda(t_j)... \lambda(t_k) \mid t_i t_j ... t_k \in \pounds(Q)\}$, while the output language of Q is defined as $\pounds_{out}(Q) = \{\varphi(M_0)\varphi(M_1)\cdots\varphi(M_w)\varphi(M_x) \mid M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} \cdots M_w \xrightarrow{t_k} M_x$ and $t_i t_j ... t_k \in \pounds(Q)\}$.*

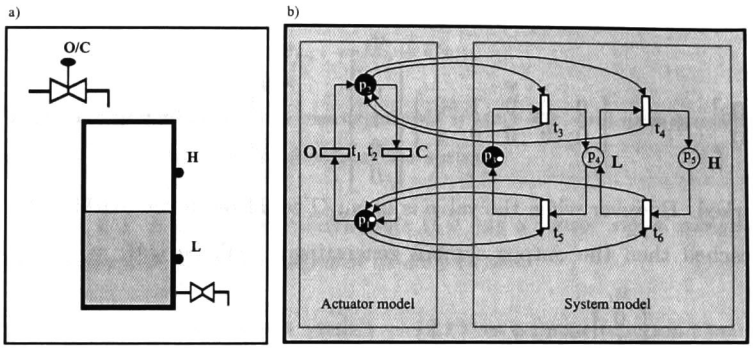


Figure 2.10: a) Physical system, b) IPN system model Q.

2.3.1 Event-detectability property

The property presented in this section correspond to the possibility that an event that had occurred in the system can be detected from the knowledge of the inputs given to the system or by the observation of its output symbols. This property was defined in [1][44].

Definition 2.22 *Let Q be an IPN. Q is event-detectable if any transition firing can be uniquely determined by the knowledge of the input given to Q and output signals that it produces.*

The following proposition provides an structural characterization of the IPN exhibiting event-detectability property.

Proposition 2.3 *Let Q be a live IPN. Q is event-detectable if and only if*

1. $\forall t_i, t_j \in T$ *such that* $\lambda(t_i) = \lambda(t_j)$ *or* $\lambda(t_i) = \varepsilon$ *it holds that* $\varphi C(\bullet, t_i) \neq \varphi C(\bullet, t_j)$ *and*
2. $\forall t_k \in T$ *it holds that* $\varphi C(\bullet, t_k) \neq \overrightarrow{0}$

Proof. The proof is sketched as follows:

(Sufficiency)

Assume that $(Q, \mathcal{M}_0)$ is an IPN where $\forall t_i, t_j \in T$ such that $\lambda(t_i) = \lambda(t_j)$ or $\lambda(t_i) = \varepsilon$ it holds that $\varphi C(\bullet, t_i) \neq \varphi C(\bullet, t_j)$ and $\forall t_k \in T$ it holds that $\varphi C(\bullet, t_k) \neq \overrightarrow{0}$. Let $M_m, M_n \in \mathcal{M}_0$ and a transition $t_p \in T$ such that $M_m \xrightarrow{t_p} M_n$ fire while the input symbol α is given to $(Q, \mathcal{M}_0)$. From state equation (2.2) $y_n - y_m$ can be computed as $y_n - y_m = \varphi(M_n) - \varphi(M_m) = \varphi(M_m + C(\bullet, t_p)) - \varphi(M_m) = \varphi C(\bullet, t_p)$. Since $\forall t_k \in T$ it holds that $\varphi C(\bullet, t_k) \neq \overrightarrow{0}$ the change in the output produced by the firing of t_p is not null, that is $y_n - y_m \neq \overrightarrow{0}$. Now there are two possibilities.

1. Suppose that the input symbol is ε, since $\forall t_i, t_j \in T$ such that $\lambda(t_i) = \lambda(t_j) = \varepsilon$ it holds that $\varphi C(\bullet, t_i) \neq \varphi C(\bullet, t_j)$ there is no transition $t_q \in T$ with $t_q \neq t_p$ such that $\lambda(t_q) = \varepsilon$ and $\varphi C(\bullet, t_q) = \varphi C(\bullet, t_p)$. Thus, the firing of transition t_p is the only one that could produce the change $y_n - y_m = \varphi C(\bullet, t_p)$ while the null input word $\alpha = \varepsilon$ was given to the system.

2. Suppose now that the input symbol is $\alpha \neq \varepsilon$, since $\forall t_i, t_j \in T$ such that $\lambda(t_i) = \lambda(t_j) = \alpha$ (or $\lambda(t_i) = \varepsilon, \lambda(t_j) = \alpha$) it holds that $\varphi C(\bullet, t_i) \neq \varphi C(\bullet, t_j)$ there is no transition $t_q \in T$ whit $t_q \neq t_p$ such that $\lambda(t_q) = \alpha$ (or $\lambda(t_q) = \varepsilon$) and $\varphi C(\bullet, t_q) = \varphi C(\bullet, t_p)$. Thus, the firing of transition t_p is the only one that could produce the change $y_n - y_m = \varphi C(\bullet, t_p)$ while the non-null input word α was given to the system.

Then, in both cases, the transition t_p that fired can be uniquely determined and $(Q, \mathcal{M}_0)$ is event-detectable.
(Necessity) Suppose first that there exist two transitions $t_i, t_j \in T$ such that $\lambda(t_i) = \lambda(t_j) = \alpha$ and $\varphi C(\bullet, t_i) = \varphi C(\bullet, t_j)$. Then for an input word α there are two transitions t_i, t_j that may fire, therefore the input symbol given to $(Q, \mathcal{M}_0)$ does not provide information to distinguish the firings of t_i and t_j. Since $\varphi C(\bullet, t_i) = \varphi C(\bullet, t_j)$ the changes in the output that those firings produce are equal and no further information is provided. Therefore, there is no way to distinguish the firings of t_i and t_j.
Now suppose that there exist two transitions $t_i, t_j \in T$ such that $\lambda(t_i) \neq \varepsilon, \lambda(t_j) = \varepsilon$ and $\varphi C(\bullet, t_i) = \varphi C(\bullet, t_j)$. Assume that $\lambda(t_i) = \alpha$. Then for an input word α both transitions t_i and t_j may fire, again the input symbol does not help to distinguish the firings of t_i and t_j. If also $\varphi C(\bullet, t_i) = \varphi C(\bullet, t_j)$ then both firings produce the same change in the output and once more the firings of those transitions cannot be distinguished.
Finally assume that $\exists t_k \in T$ such that $\varphi C(\bullet, t_k) \neq \overrightarrow{0}$ Then the firing of t_k has no effect in the output and no matter what is the input symbol α given to $(Q, \mathcal{M}_0)$ there is no way to determine if transition t_k fires.
Thus, in all those cases the firing of the transitions cannot be uniquely determined and if any of those conditions holds $(Q, \mathcal{M}_0)$ is not event-detectable. ■
The first condition of proposition 2.3 states that if two transitions have the same input signal assigned then their columns in φC matrix must to be different to can distinguish them, while the second condition refers to the fact that any change of marking allows to determine the occurrence of a transition.

Chapter 3

Identification of DES

This chapter overviews the works dealing with identification of DES. First the works on grammatical inference are summarized; then recent works that use PN as modelling formalism are presented.

3.1 Grammatical inference

The identification problem of DES addressed in this thesis is very related to the grammatical inference problem. The grammatical inference is an inductive problem where the target domain is a formal language and the representation class is a family of grammars. The learning or inference task consists in given a finite number of examples of an unknown target language, identify a correct grammar for it. Grammatical inference is a well-established research field in artificial intelligence and it dates back to the 60s. Gold [19] originated those studies and introduced the notion of identification in the limit.
This chapter is devoted to present several results of the inference or learning problem found out in the literature. The inference problem enclose the following three elements:

1. A class Ω of objects. One of the objects will be chosen, the learner will be presented information about it, and the learner is to figure out which one it is.

2. A method of information presentation. At each time the learner receives a unit of information i_t which is chosen of a set I. The method of information presentation consists in specify, for each $\omega \in \Omega$, which sequences of units of information, $i_1, i_2, \cdots$, are allowed. Let the set of allowable sequences be designated $I^{\infty}(\omega)$.

3. A naming relation. The learner is to identify the unknown object by finding one of its names. A naming relation consists of a set N of names and a function f which assigns an object to each name, $f : N \longrightarrow \Omega$.

The inference problem is to determine whether there is a rule the learner can use to accomplish the following: for any object $\omega \in \Omega$ and for any information sequence from $I^{\infty}(\omega)$, on the basis of that information sequence the rule will be yield a name n of ω, that is $f(n) = \omega$. In all works here presented it is considered as an unknown object $\omega \in \Omega$ the language that it is desired to identify.
Three variations of the identification problem are the following:

a) Identification in the limit. In this case the learner is to guess a name of the unknown object at each time. It is required that there be a finite time after which the guesses are all the same and are correct.

b) Finite identification is the type of identification problem usually considered. It is best known in automata theory. In finite identification, the learner is to stop the presentation of information at some finite time when it thinks it has received enough, and state the identity of the unknown object. This is not possible unless there is some finite time at which the information distinguishes the unknown object. That is, no other object satisfies the information.

c) Fixed-time identification. In this case the information sequence stops after some finite time which is specified a priori and which is independent of the object being described. The learner is to then state the identity of the unknown object.

The motivation for studying this problem is to construct a formal model of human language acquisition. Grammatical inference is applied to natural language processing [13][40] and computational biology [26][50], and has

been investigated, more or less independently, within many research fields including machine learning, computational learning theory, pattern recognition, computational linguistics, neural networks, formal language theory, information theory, and many others.

The research activities on grammatical inference have been stimulated by the new learning models proposed recently within computational learning theory framework: the query learning model of Angluin [8] and the PAC (probably approximately correct) learning model of Valiant [57]. These models put much more emphasis on the computational efficiency of the inference algorithm.

This review will begin with the problem of identifying deterministic finite automata (DFAs) from examples. DFAs are the bottom class of formal grammars in the Chomsky hierarchy, and the problem of identifying DFAs from examples has been studied quite extensively [7][42], where several interesting results are presented on the identification of DFAs: polynomial-time identification of DFAs from queries, identification of subclasses of DFAs from positive data, computationally hardness results and identification from erroneous examples. Also it is considered the problem of identifying context-free grammars (CFGs) because the questions of whether there are analogous results held for context-free grammars would be more interesting and important. The results contain identification of CFGs from examples in the form of structured strings, polynomial-time reduction to identification of finite automata, and efficient identifications of several subclasses of CFGs.

3.2 Learning models

Within computational learning theory, there are three major established formal models for learning from examples or inductive inference: Gold's model of identification in the limit [19], the query learning model by Angluin [9], and PAC learning model by Valiant [57]. Each model provides a learning protocol and a criterion for the success of learning. Identification in the limit views learning as an infinite process and provides a learning model where an infinite sequence of examples of the unknown grammar G is presented to the inference algorithm M and the eventual or limiting behavior of the algorithm is used as the criterion of its success. A complete presentation of the unknown grammar G is an infinite sequence of ordered pairs (w, l) from $\Sigma^* \times \{0, 1\}$ such that $l = 1$ if and only if w is generated by G, and such that every string w of Σ^* appears at least once as the first component of some pair in the sequence, where Σ is the terminal alphabet. An inference algorithm M take as input initial segments of a complete presentation of G, and outputs a next conjecture. If for every complete presentation of the unknown grammar G, M guesses a correct grammar which is equivalent to G at some point and never changes its guess after this, then M is said to *identify G in the limit from complete presentations.*

Angluin [7] has considered a learning situation in which a teacher is available to answer specific kind of queries on the unknown grammar G. In the query learning model, a teacher is a fixed set of oracles that can answer specific kind of queries made by the inference algorithm of the unknown grammar G. The following two types of queries are typical:

a. Membership. The input is a string $w \in \Sigma^*$ and the output is "yes" if w is generated by G and "no" otherwise.

b. Equivalence. The input is a grammar G' and the output is "yes" if G' is equivalent to G (i.e. G' generates the same language as G) and "no" otherwise. If the answer is no, a string w' in the symmetric difference

of the language $L(G)$ generated by G and the language $L(G')$ generated by G' is returned. The string w' is named counter example.

In this approach, an inference algorithm M runs with oracles for queries for the unknown grammar G, and eventually halts and outputs a correct grammar in a certain finite time. An important result is that the class of DFAs can be identified in polynomial time using equivalence queries and membership queries while it cannot be efficiently identified from equivalence queries only [9][4].

Valiant [57] has introduced the distribution-independent probabilistic model of learning from random examples, which is called probably approximately correct learning (PAC learning). In PAC learning model, it is assumed that random samples are drawn independently from the domain Σ^* whose probability distribution D may be arbitrary and unknown. The success of the identification is measured by two parameters: the accuracy parameter ε and the confidence parameter δ, which are given as inputs to the inference algorithm. The error of a grammar G' with respect to the unknown grammar G is defined to be the sum of probabilities $D(w)$ of the strings w in the symmetric difference of $L(G')$ and $L(G)$ with respect to D. A successful inference algorithm is one that *with high probability* (at least 1-δ) finds a grammar *whose error is small* (less than ε).

It is measured the efficiency of the inference algorithm with respect to relevant parameters: the size of the examples and the size of the unknown grammar. The size of the example in the form of string is the length of the string. The size of the unknown grammar is usually the number of states, in the case of finite automata, and the number of production rules, in the case of context free grammars.

3.3 Learning finite automata

In this section, it is presented a review of several important results and useful techniques related to computationally efficient identifications of deterministic finite automata. Good references are the early work of Trakhtenbrot and Barzdin [56], the work of Wiehagen concerning to the learnability from good examples [58], and an excellent survey by Pitt [42].

A deterministic finite (state) automaton (DFA) is defined by a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is an alphabet of input symbols, δ is the state transition function defined as $\delta : Q \times \Sigma \longrightarrow Q$, q_0 is the initial state, and $F \subseteq Q$ is a set of final states. The language accepted by a DFA A is denoted by $L(A)$.

3.3.1 Learning from representative samples

When it is needed to identify an unknown DFA $A = (Q, \Sigma, \delta, q_0, F)$ from examples, useful information about A is a representative sample S of A, that is, a finite subset of $L(A)$ that exercises every live transition in A. Taking the set $R(S)$ of all prefixes of strings in S, for every live state q of A, there must exist a string u in $R(S)$ such that $\delta(q_0, u) = q$. Further, for every state q and every transition $\delta(q, a)$ from q where $a \in \Sigma$, there exists a string va in $R(S)$ such that $\delta(q_0, v) = q$ and $\delta(q, a) = \delta(q_0, va) = q'$. Thus, every state and transition are represented by strings in $R(S)$. It remains to distinguish two states qu and qv represented by two strings in $R(S)$, i.e. $q_u = \delta(q_0, u)$ and $q_v = \delta(q_0, v)$, if q_u and q_v are different states in A.

Angluin [10] has given an efficient procedure to solve this problem using membership queries.

Theorem 3.1 *(Angluin [4]) The class of deterministic finite automata can be identified in polynomial time from a representative sample and using membership queries.*

3.3.2 Learning with teachers

Angluin [2] has considered a learning protocol which is based on what is called "minimally adequate teacher" This teacher can answer two types of queries about the unknown DFA A made by an inference algorithm: membership query and equivalence query. Angluin [9] has shown that equivalence queries compensate for the lack of representative samples, and presents an efficient inference algorithm for identifying DFAs using equivalence and membership queries.

Theorem 3.2 *(Angluin[9]. The class of deterministic finite automata can be identified in polynomial time using equivalence queries and membership queries.*

The important data structure used in Angluin´s algorithm is called observation table. An observation table is a two-dimensional matrix with rows and columns labelled by strings. The entry is 0 or 1, and the intended interpretation is that the entry for row s and column e is equal to 1 if and only if the string $s \cdot e$ is accepted by the unknown automaton. The rows are partitioned in two parts, the ones labelled by a non empty prefix-closed set S of strings and the others labelled by the set $S \cdot \Sigma$. Rows labelled by S are the candidates for states of the automaton being constructed and rows labelled by $S \cdot \Sigma$ are used to construct the state transition function. The columns labelled by a non empty suffix closed set E of strings play a role of witnesses to distinguish the candidates for representing states. The observation table has $S = E = \{\varepsilon\}$ (the set of only the empty string) at the beginning of learning, and is augmented as the algorithm runs. Two specific observation tables are defined, which are *closed* and *consistent*. When an observation table fulfill that it is a closed and consistent then a minimum DFA can be constructed in polynomial time of the size of the table which it is consistent with the data contained in the table. The algorithm is going to find a closed, consistent observation table by asking membership queries to fill entries. It has been shown in [9] that the algorithm ask at most $O(mn^2)$ membership queries and $n-1$ equivalence queries, and eventually terminates and outputs the minimum DFA which is equivalent to the unknown DFA, where m is the maximum length of any counter example returned by the teacher during the running algorithm and n is the number of states in the minimum DFA, equivalent to the unknown DFA. The idea of the observation table is also related to the state characterization matrix by Gold [20].
Yokomori [61] has studied efficient identification of non deterministic finite automata from equivalence and membership queries.

3.3.3 Learning from positive data

One interesting and important topic on Gold´s framework of identification in the limit for language learning is identification from positive data. A positive presentation of the unknown DFA A is any infinite sequence of examples such that the sequence contains all and only the strings in the language $L(A)$. Gold [19] has shown that there is a fundamental and important difference in what could be learned from positive versus complete presentations, and shown a negative result that no superfinite class of languages can be identified in the limit from positive presentation. A class of language is called superfinite if it contains all the finite languages and at least one infinite language. Since the class of regular languages is superfinite, it is needed to constrain DFAs somehow to subclasses to establish identifiability results from positive presentation.

The problem is to avoid overgeneralization, which means guessing a language that is a strict superset of the unknown language. Angluin [5] has introduced a series of subclasses of DFAs, called k-reversible automata for $k = 0, 1, 2, \ldots$, and shown that the existence of characteristics samples is sufficient to identify from positive presentation (to avoid overgeneralization) this kind of automata. A characteristic sample of a k-reversible automaton A is a finite sample $S \subset L(A)$ such that $L(A)$ is the smallest k-reversible language that contains S with respect to the set inclusion. Notice then that any characteristic sample is a representative sample for k-reversible automata.
Notice that a representative sample provides enough information for reconstructions of states and states transitions. By using the structural properties specific to k-reversible automata it could be accomplished the main task of state distinctions in identifying k-reversible automata without the use of membership queries. For example a zero-reversible automaton, is a DFA such that it has at most one final state and no two edges entering any state are labeled with the same symbol. Given a representative sample S for the unknown zero-reversible automaton, the prefix tree automaton A' is constructed such that it accepts the set S, and then merge the states in A' to satisfy the conditions for zero-reversible automata.

Theorem 3.3 *(Angluin [5]. The class of k-reversible automata, for $k = 0, 1, 2 \ldots$, can be identified in the limit from positive presentation.*

Furthermore, the inference algorithm updates a conjecture in time polynomial in the size of the inputs.
Another interesting class of DFAs which can be identified in the limit from positive presentation is the class of strictly deterministic automata investigated by Yokomori [62]. A strictly deterministic automaton is a DFA such that the set of labels W for state transition edges is extended to be a finite subset of strings over Σ, each edge has a unique label (no same label is attached to different edges), and for each symbol $a \in \Sigma$ there is at most one label in W starting with a.

Theorem 3.4 *(Yokomory [62]). The class of strictly deterministic automata can be identified in the limit from positive presentation.*

An inference algorithm can be constructed so that it not only runs in time polynomial in m to update the conjecture, where m is the maximum length of all positive examples provides so far, but also makes at most a polynomial number of implicit errors of prediction in m and n, where n is the size of the unknown strictly deterministic automaton. Pitt [42] has proposed the definition of implicit errors of prediction that after seeing ith example in the presentation, the inference algorithm M is said to make an implicit error of prediction at step i if the conjecture output by M is not consistent with the $(i+1)$th example.
Other interesting topics and results on identification from positive presentation which may not be directly related to DFAs are Angluin's characterization of identifiability from positive presentation [3], Angluin's pattern languages [2], Koshiba's extension to typed pattern languages [25], Shinohara's general result for identifiability from positive presentation [52], and Oncina et Al.'s subsequential transducers [37].

3.3.4 Hardness results

There are many computationally hardness results related to identifying DFAs. Gold [20] has shown that the problem of finding a DFA with a minimum number of states consistent with a given finite sample of positive and

negative examples is NP-hard. This result is generally interpreted as indicating that even a very simple case of grammatical inference, identifying DFAs from positive and negative examples, is computationally intractable. Further, Pitt and Warmuth [43] have proven a stronger result, namely that it is NP-hard to find a DFA of at most $n^{(1-\alpha)\log\log n}$ states consistent with a given finite sample of positive and negative examples for any constant $\alpha > 0$, where n is the number of states of a minimum DFA consistent with the given sample.
Angluin [10] has shown negative results for efficient identifications of various classes of grammars from equivalence queries only. In that work is developed useful technique of approximate fingerprints to obtain negative results for identification from equivalence queries only. As applications of the technique has shown that there is no polynomial time algorithm using only equivalence queries that identifies the class of DFAs, non deterministic finite automata, context free grammars, or disjunctive or conjunctive normal form boolean formulas.

3.3.5 Learning from erroneous examples

In practice, it is natural to assume that the example may contain some noise. There are fewer works to study the effect of noise on learning from queries in the Valiant's probabilistic framework of PAC-learnability.
Sakakibara [47] has defined a benign model for errors in the responses to membership queries where answers to queries are subject to random independent noise (i.e. for each query there is some independent probability to receive an incorrect answer and these errors are not persistent), and shown that these errors can be effectively removed by repeating the query until the confidence in the correct answer is high enough.
Ron and Rubinfeld [45] have considered a model of persistent noise in membership queries in which a fixed but randomly chosen fraction of the membership queries are answered incorrectly but any additional query on the same string is replied consistently with the same incorrect answer when queried again. They have shown by modifying Angluin's algorithm (theorem 3.2) for identifying DFA's using equivalence and membership queries that DFAs can be learned in polynomial time from membership queries with persistent noise under the uniform distribution on inputs.
Sakakibara and Siromoney [49] have studied a noise model which specific to language learning where the examples are corrupted by purely random errors affecting only the strings (and no labels). They have considered three types of errors on strings, called EDIT operation errors. EDIT operations consists of insertion, deletion, and change of a symbol in a string. They have shown efficient identification from random examples with EDIT noise for a small subclass of regular languages defined by containment decision lists.

3.4 Learning context free grammars

The question of weather there are analogous results for context-free grammars is interesting and important simply because context-free grammars are more expressive than the DFA.
A context-free grammar (CFG) is defined by the quadruple $G = (N, \Sigma, P, S)$, where N is an alphabet of non terminal symbols, Σ is an alphabet of terminal symbols such that $N \cap \Sigma = \emptyset$, P is a finite set of productions rules of the form $A \longrightarrow \alpha$ for $A \in N$ and $\alpha \in (N \cup \Sigma)^*$ and S is a special non terminal called start symbol. The language generated by a CFG G is denoted as $L(G)$.
Angluin [10] has shown that the whole class of CFG cannot be identified in polynomial time using equivalence queries only. Furthermore, Angluin and Kharitonov [11] have shown that the problem of identify the class of

CFGs from membership and equivalence queries is computationally as hard as the cryptographic problems for which there is currently no known polynomial time algorithm (e.g., inverting RSA encryption, or factoring Blum integers). Even the existence of these negative results, in the following sections are presented several positive results for identifying the whole class of CFGs with additional information or identifying subclasses of CFGs efficiently.

3.4.1 Learning from structural information

Here it is considered an identification problem for CFGs where, besides given examples, some additional information is available for the inference algorithm. Useful (and maybe reasonable) information would be on the grammatical structure of the unknown CFG. It is assumed example presentations in the form of strings with grammatical structure. Levy and Joshi [27] have already suggested the possibility of efficient grammatical inferences in terms of strings with grammatical structure.
A string with grammatical structure, called a structured string or a structural description (of string), is a string with some parentheses inserted to indicate the shape of the derivation tree of a CFG, or equivalently unlabeled derivation tree of the CFG, that is a derivation tree whose internal nodes have no labels. It is known that the set of derivation trees of a CFG constitutes a rational set of trees, where a rational set of trees is a set of trees which can be recognized by some tree automaton. Further, the set of unlabeled derivation trees of a CFG also constitutes a rational set of trees. Based on these observations, the problem of identifying CFGs from structures strings is reduced to the problem of identifying tree automata.
Sakakibara [46] has shown by extending Angluin's inference algorithm (theorem 3.2) for DFA's to tree automata that the class of CFGs can be identified in polynomial time using structural membership queries and structural equivalence queries.

Theorem 3.5 *(Sakakibara [46]). The class of context free grammars can be identified in polynomial time using structural equivalence queries and structural membership queries.*

Let $D(G)$ denotes the set of derivation trees of a CFG G and $s(D(G))$ denote the set of unlabeled derivation trees (structured strings) of G. A structural membership query for a structures string ask whether it is generated by the unknown CFG G, and a structural equivalence query returns "yes" if a queried CFG G' is structurally equivalent to the unknown CFC G and returns "no" with a counter example otherwise, that is, a structured string in the symmetric difference of $s(D(G))$ and $s(D(G'))$.
It was presented that Angluin's algorithm for identifying DFAs uses the observation table to organize the information about a finite collection of strings with the indication whether they are strings accepted by the unknown DFA. For the problem to identify a CFG it is extended the observation table to the one for tree automata. The extended observation table has rows labelled by structures strings and columns labelled by structures strings with a special symbol. The intended interpretation is that the entry for row s and column e is equal to 1 if and only if the structures string of the concatenation of s and e is a structured string generated by the unknown grammar G.
Since the class of CFGs is superfinite, Gold's negative result [19] on identifiability from positive presentation implies that the class of CFG's cannot be identified in the limit from positive presentation. Sakakibara [48] has demonstrated that certain information on the grammatical structure of the unknown CFG could help in the

inference. He has shown that there exists a class of CFGs called reversible context free grammars, which can be identified in the limit from positive presentations of structured strings, that is, all and only the unlabeled derivation trees of the unknown CFG, and shown that the reversible CFG is a normal form for CFGs, that is, reversible CFG can generate all the context free languages.
A reversible context free grammar is a CFG $G = (N, \Sigma, P, S)$ such that (1) $A \longrightarrow \alpha$ and $B \longrightarrow \alpha$ in P implies that $A = B$ and (2) $A \longrightarrow \alpha B \beta$ and $A \longrightarrow \alpha C \beta$ implies that $B = C$, where A, B and C are non terminals and $\alpha, \beta \in (N \cup \Sigma)^*$.

Theorem 3.6 *(Sakakibara [48]). The class of reversible context free grammars can be identified in the limit from positive presentation of structured strings provided such that the structured strings are generated with respect to a reversible context free grammar for the unknown context free language.*

Since the inference algorithm for reversible context free grammars is an extension of the Angluin's inference algorithm which identifies zero-reversible automata (theorem 3.3), the algorithm updates a conjecture in time polynomial in the size of the inputs. Note that the above result does not imply that the whole class of CFG can be identified from positive presentation of structured strings.
A related early work to identify CFGs from positive presentation of structures strings is presented in [15]. In this work it is described a constructive method for identifying a subclass of CFGs which is a different class from reversible CFGs, from positive samples of structured strings. The defined class of CFGs describes only a subclass on context free languages, called noncounting context free languages. Mäkinen [29] has refined Sakakibara's inference algorithm for reversible CFGs to gain more efficiency, and also investigated a subclass of reversible CFGs, called type invertible grammars, that can be identified from positive presentation structures strings in linear time in the size of the inputs.

3.4.2 Reductions to finite-automata learning problems

A well-known technique often used to establish identifiability results is a reduction technique that reduces an inference problem to some other inference problem whose result is known. Takada [55] has shown that the inference problem for even linear grammars can be solved by reducing it to one for DFAs, and presented a polynomial-time algorithm for the reduction. For example, the class of even linear grammars can be identified using equivalence and membership queries in polynomial time by employing Angluin's efficient algorithm for DFAs (theorem 3.2) via reduction.
An even linear grammar is a CFG that has productions only of the form $A \longrightarrow uBv$ or $A \longrightarrow w$ such that u and v have the same length. A and B are nonterminals and u, v and w are strings over Σ. Let $G = (N, \Sigma, P, S)$ be an even linear grammar. It is written $x \stackrel{\pi}{\Longrightarrow} y$ to mean that y is derived from x applying the production π in P, where $x, y \in (N \cup \Sigma)^*$ The derivation from x_0 to x_k obtained by applying a sequence $\gamma = \pi_1 \pi_2 \cdots \pi_k$ of productions is denoted by $x_0 \stackrel{\gamma}{\Longrightarrow} x_k$. γ is called associated word and a set of associated words is called control set on G. The language generated by G with a control set C is defined by $L(G, C) = \{w \in \Sigma^* \mid S \stackrel{\gamma}{\Longrightarrow} w$ and $\gamma \in C\}$. It can be shown that there is a universal even linear grammar G_U such that for any even linear grammar G, $L(G) = L(G_U, C)$ for some regular control set C.

Theorem 3.7 *(Takada [55]). The problem of identifying the class of even linear grammars is reduced to the problem of identifying the class of finite automata.*

Note that the class of even linear languages properly contains the class of regular languages and is a proper subclass of context free languages. By iteratively applying the above reduction technique, Takada [54] has further developed an infinite hierarchy of families languages whose identification problems are reduced to the identification problem of DFAs.

3.4.3 Learning subclasses of context free grammars

Because the whole class of CFGs seems to be hard to be identified efficiently without any additional information, there have been some attempts to design polynomial time algorithms for identifying subclasses of CFGs from examples.

Ishizaka [24] has investigated a subclass of CFGs called simple deterministic grammars and gave a polynomial time algorithm for exactly identifying it using equivalence and membership queries in terms of general CFGs. This inference algorithm may sometimes ask equivalence query for a CFG which is not simple deterministic.

A CFG $G = (N, \Sigma, P, S)$ in 2-standard form is called simple deterministic if $A \longrightarrow a\alpha$ and $A \longrightarrow a\beta$ in P implies that $\alpha = \beta$, where A and B are non terminals, a is terminal, and $\alpha, \beta \in (N \cup \Sigma)^*$

Theorem 3.8 *(Ishizaka [24]). The class of simple deterministic grammars can be identified in polynomial time using equivalence queries and membership queries in terms of general context free grammars.*

Notice that given any regular language L, the language $L\#$ is simple deterministic, where $\#$ is a special symbol not in Σ. In this sense, the class of simple deterministic languages properly contains the class of regular languages.

Yokomory [60] has considered a smaller class of simple deterministic grammars with the goal of finding a polynomial time algorithm to identify it in the limit from positive presentation. A CFG $G = (N, \Sigma, P, S)$ in Greibach normal form is called very simple if for each terminal symbol a in Σ there exists exactly one rule production starting with a (i.e., exactly one production rule of the form A$\longrightarrow a\alpha$, where $\alpha \in (N \cup \Sigma)^*$). He has shown that the class of very simple grammars can efficiently be identified in the limit from positive presentation, and this result has provided the first instance of languages class containing non regular languages that can be identified in the limit in polynomial time in a criterion proposed by Pitt [42], that is, the time for updating a conjecture is bounded by a polynomial in the size of the unknown grammar and the sum of lengths of examples provided, and the number of implicit errors of prediction made by the inference algorithm is bounded in a polynomial in n.

Theorem 3.9 *(Yokomori [60]). The class of very simple grammars can be identified in the limit from positive presentation in polynomial time.*

From this result, it immediately follows that the class of very simple grammars can be identified in polynomial time using only equivalence queries.

Related to identification of very simple grammars, Burago [12] has investigated the structurally reversible CFGs and shown that this class of CFG can be identified in polynomial time using equivalence queries and membership queries. A CFG is called structurally reversible if among all non terminal strings that might derive a given terminal string, no one is a extension of the other. The class of structurally reversible CFGs is a subclass of CFGs and the class of structurally context free languages properly contains the class of very simple languages.

3.5 Identification using Petri nets

In [22] is presented an algorithm for constructing Petri net models. The specification of the system are presented in the form of sequences of events, or concurrent languages such as partial languages. The proposed algorithm has two phases. In the first phase, each example is given to the algorithm until the language of the target system is identified in the form of DFA. The identified DFA has the property to be I-reversible. A DFA I-reversible has only one initial state and only one final sate and it is invariant consistent. The meaning of invariant consistency can be described as follows: each occurrence of events causes some change of state, and this change depends only on which events have occurred. The I-reversible languages is a subclass of the zero-reversible languages [6]. This phase is related to the inductive inference of regular languages such as those presented in previous section. In the second phase, the algorithm extracts the dependency relation from the language, and then guesses the structure of a Petri net that accepts the obtained language. The running time of the first phase is bounded by a polynomial function of given inputs. For the second phase it is presented a polynomial time algorithm for a subclass of live and safe free choice nets. In this work it is consider the case that the target system is represented by a safe marked net without self loops and with one final state.

In [31] is presented an identification method based on the least square estimator $\theta = (X^T X)^{-1} XY$ where X represents the inputs and Y represents the outputs of a system which is described as $Y_M = X\theta$. To describe a PN in terms of X, Y and θ it is considered that $X = v_k^T$ and $Y = \Delta M_k = M_{k+1} - M_k$, since the state equation of a PN can be reduced to $M_{k+1} - M_k = Cv_k$ having the form of $Y_M = X\theta$. Then using this method the relation of the between places and transitions is founded out.

Notice that to use this method it is needed to know the number of places and transitions of the PN that it is needed to be identified and also several evolutions of the system must to be considered in order to built the vectors X and Y

In [30] it is presented an algorithm to identify a DES building a PN. This algorithm does not use a positive sample of the system language, instead of the algorithm works on-line with the system processing its output symbols and computing a new model as new information of the system is detected. The systems that could be identified using that algorithm are those described by IPN in which the non measurable places are constrained to have only one input and only one output transition. To preserve the firing of two concurrent transitions computed consecutively in some evolution of the system, self-loops are added to the model which are removed when in a posterior evolution of the system that transitions occur in different order. This work gave the basis for the development of the work here introduced.

Chapter 4

On-line synthesis of PN models

In this chapter it is defined the asymptotic identification problem for DES. The identification approach herein proposed computes an IPN as the mathematical model describing the unknown DES; the construction of this model is incrementally performed by computing a new model from new on-line measurements of the system outputs. In this chapter are presented the strategies to update an already computed model as the system evolves.

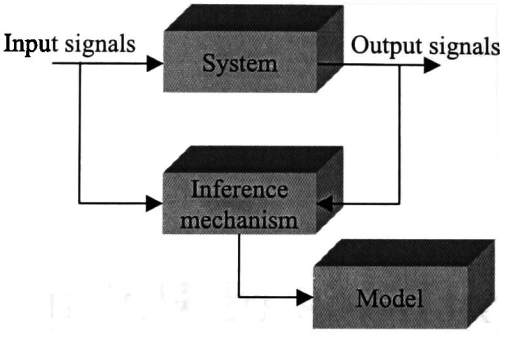


Figure 4.1: Identification scheme.

4.1 Introduction

As was introduced previously, the addressed problem in this work consists in computing a mathematical model for a system from the knowledge of its input and output signals. This problem is known as system identification. The identification scheme is illustrated on figure 4.1. In this work the proposed identification scheme fits with the following hypotheses:

H_1: The systems considered to be identified are those DES that can be described by a live, binary and cyclic IPN Q.

H_2: Q is an event-detectable IPN.

H_3: The transitions of Q are not fired simultaneously and Q has not self-loops.

H_4: The input and output signals will be sequences of input and output symbols respectively.

Given the characteristics of the identification problem addressed in this work, the inference mechanism will build the model processing the input and output signals of the system as it evolves. The computed model will be a live, binary, cyclic and event-detectable IPN.
In this chapter it is defined the asymptotic identification problem for DES and are described the procedures in which the proposed identification approach to solve this problem is based. This approach mainly differs from others identification approaches in the way of how the model of the system is computed and also in the class of computed model describing the system. This novel identification approach proposes to compute an IPN model describing the behavior of the unknown DES; the construction of this model is incrementally performed by computing a new model from new on-line measurements of the system outputs. One advantage of this approach is that it is not needed to have a priori information of the system (commonly called a positive sample of the system) because the inference mechanism will operate on-line processing the system information.
This chapter introduce the concepts, characterizations and procedures in which the identification approach is supported.

4.2 Alternative representation of a PN structure

The procedures for inferring a model Q_i from the observed behavior of the system, handle a simple structural unit called dependency. This section presents how an IPN can be represented by its dependencies.

Definition 4.1 *Let Q be an IPN. Two transitions t_i and t_j of Q form a dependency $[t_i, t_j]$ iff $\exists p_k$ such that $p_k \in t_i^\bullet$ and $p_k \in^\bullet t_j$. The notation $p_k = [t_i, t_j]$ is referred to the fact that the place p_k forms a dependency between t_i and t_j.*

On figure 4.2 is depicted a dependency in which p_k forms a dependency between the transitions t_i and t_j.

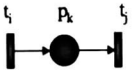


Figure 4.2: Graphical representation of the dependency $p_k = [t_i, t_j]$

Notice that the place p_k could form more than one dependency.

Definition 4.2 *If p_k is a measurable place, then $[t_i, t_j]$ will be called measurable dependency (MDep) and if p_k is a non measurable place then $[t_i, t_j]$ will be called non measurable dependency (NDep). On figure 4.3 are depicted a MDep and a NDep.*

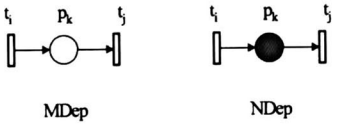


Figure 4.3: Measurable and non measurable dependencies.

The representation of a dependency $p_k = [t_i, t_j]$ in the incidence matrix C of Q is as follows: by definition of an incidence matrix C, p_k is represented by the kth row in C and t_i and t_j are represented by the ith and jth column of C, then $C[k,i] = 1$ since t_i is an input transition of the place p_k and $C[k,j] = -1$ since t_j is an output transition of the place p_k.

Definition 4.3 *The set of all dependencies in Q, is denoted as $Dep(Q) = Dep^m(Q) \cup Dep^u(Q)$, where $Dep^m(Q)$ is the set of all MDep and $Dep^u(Q)$ is the set of all NDep in Q. Hence, the set $Dep(Q)$ describes also the structure of the Q in terms of its dependencies.*

The set of all NDep formed with the place p_k will be denoted as $Dep^u_{p_k}$

Definition 4.4 *A place p_k in an IPN Q can describe two kinds of dependencies:*

1. *p_k forms a single dependency if: $|^\bullet p_k| = |p_k^\bullet| = 1$, where p_k forms just one dependency. The characteristic of this kind of dependency is that p_k has only one input transition t_i and only one output transition t_j and they belong to the same t-semiflows in Q.*

2. *p_k forms a complex dependency if $|{}^\bullet p_k| > 1$ and $|p_k^\bullet| \geq 1$ or $|{}^\bullet p_k| \geq 1$ and $|p_k^\bullet| > 1$, where p_k forms more than one dependency. Indeed p_k forms $a \times b$ dependencies, where $a = |{}^\bullet p_k|$ and $b = |p_k^\bullet|$. The main characteristic of this dependency is that p_k have more than one input transitions or more that one output transitions and the transitions t_x and t_y in a dependency $[t_x, t_y] = p_k$ could belong to different t-semiflows in Q.*

Figures 4.2 and 4.3 show single dependencies.
The dependency depicted on figure 4.4 describes a complex dependency. Notice that p_k has two input and also two output transitions, hence it forms 4 dependencies: $[t_i, t_j]$, $[t_i, t_y]$, $[t_x, t_y]$ and $[t_x, t_j]$. Consider the NDep $[t_i, t_j] = p_k$, notice that t_i belongs to another t-semiflow in which t_j does not belong because p_k also forms the dependency $[t_i, t_y] = p_k$.

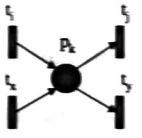


Figure 4.4: A complex dependency

- If the place p_k is a measurable place then it describes a single or a complex MDep and if p_k is a non measurable place then it describes a single or a complex NDep.

- A row representing a place forming a complex NDep has a -1s and b 1s.

Example. Consider the IPN Q depicted on figure 4.5. The decomposition of Q in its dependencies is the following:

MDep

$p_1 = [t_4, t_1]$ $p_3 = [t_2, t_4]$
$p_1 = [t_4, t_5]$ $p_4 = [t_1, t_3]$
$p_1 = [t_4, t_7]$ $p_6 = [t_5, t_6]$
$p_1 = [t_9, t_1]$ $p_7 = [t_7, t_8]$
$p_1 = [t_9, t5]$
$p_1 = [t_9, t_7]$

NDep

$p_2 = [t_1, t_2]$
$p_5 = [t_3, t_4]$
$p_8 = [t_6, t_9]$
$p_8 = [t_8, t_9]$

The places forming complex dependencies are the places p_1 and p_8.

p_1 forms complex MDep while p_8 forms complex NDep.

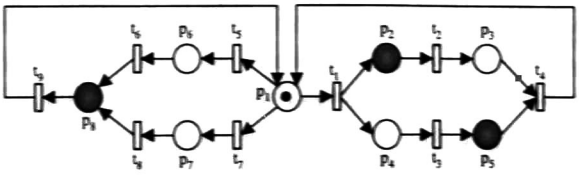


Figure 4.5: Interpreted Petri net.

Definition 4.5 *Let Q be an IPN and $\sigma = t_x t_a t_b \cdots t_c t_d t_y \in \mathcal{L}(Q)$ be a firing sequence, a dependency sequence from a transition t_x to a transition t_y is defined as $DSeq(t_x, t_y) = [t_x, t_a][t_a, t_b] \cdots [t_c, t_d][t_d, t_y]$ where each dependency $[t_i, t_j] \in Dep^m(Q) \cup Dep^u(Q)$.*

Remark. It is important to notice that a PN has a unique dependency decomposition, then the structure of a PN can be also represented or described by its dependencies. ■

4.3 Identification problem formulation

The identification problem addressed in this work deals with the computation of a model for an assumed unknown system when the only available information is its observed behavior (input and output signals), this fact leads to compute the model as the system evolves, hence the identification algorithm will work on-line with it. Since it is not possible to know if the observed information is enough to compute a model describing the entire behavior of the system, then a sequence of models is computed in such way that every model in the sequence is computed when new information of the system is observed; each new model is "better" than the previous one in the sense that it is more similar to the actual model of the system. The last model in the sequence will describe at least the observed behavior of the system. The identification strategy defined in this thesis is named *asymptotic identification* [32][33][34][35], due to the computed model of the system converges to the actual model of the system as more information is provided by itself. Given previous statements it is formally formulated the asymptotic identification problem.

4.3.1 Problem definition

Let S_f be a DES that can be modeled by an IPN Q; and $\mathcal{M} = \{Q_0, Q_1, ...\}$ be the non empty set of all IPN. Then the asymptotic identification problem is defined as follows:

1. Select a similitude function $f : \{Q\} \times \mathcal{M} \longrightarrow \mathcal{R}^+$ indicating the similitude between Q and $Q_j \in \mathcal{M}$. A lower value of $f(Q, Q_j)$ indicates that Q and Q_j are more similar.

2. Find out a model sequence $Q_0, Q_1, ..., Q_k$, where $Q_i \in \mathcal{M}$ such that $f(Q, Q_i) \leq f(Q, Q_{i-1})$.

3. Each model Q_i in the sequence will be computed as new information of the system is detected.

The notion of a model sequence indicates that the actual model of the system will be computed in an incremental way: once a new information of the system is detected, a new IPN model can be computed such that it is more approximated to the actual model of the system than its predecessor model in the sequence. Furthermore, each model in the sequence fulfills that it describes at least the observed behavior of the system at the moment of its computation. The procedures used to update each model in the sequence is presented in section 4.4.

4.3.2 Similitude function

The similitude function that estimates how much the computed model Q_i is approached to the actual model Q of the system S_f is defined as follows.

Definition 4.6 *Let $\{\varphi C_Q\}$ be the set of columns of matrix φC of the system model Q, $\{\varphi C_{Q_i}\}$ be the set of columns of matrix φC of the computed model Q_i and, let $Dep^u(Q)$ and $Dep^u(Q_i)$ be the sets of NDep of the system Q and the identified model Q_i respectively. The identification error is defined as:*

$$f(Q, Q_i) = |\{\varphi C_Q\} - \{\varphi C_{Q_i}\}| + |Dep^u(Q) - Dep^u(Q_i)| + |Dep^u(Q_i) - Dep^u(Q)| \tag{4.1}$$

The terms $|\{\varphi C_Q\} - \{\varphi C_{Q_i}\}|$ and $|Dep^u(Q) - Dep^u(Q_i)|$ represent the number of elements (columns of φC_Q and $NDep$) of Q which are missed in the computed model Q_i because they are not still computed, while the term $|Dep^u(Q_i) - Dep^u(Q)|$ represents the wrong computed NDep in Q_i, these wrongly NDep are computed to preserve the behavior of Q in Q_i. Even a NDep $[t_i, t_j] = p_k$ belongs to Q, if p_k is not fully computed then it will be considered also as an error since it does not belong to Q; this fact is reflected in the term $|Dep^u(Q) - Dep^u(Q_i)|$ due to it represents the number of missed NDep of Q in Q_i. In this case p_k will not be removed in posterior steps of the identification procedure instead of it will be updated until the k-th row of γC_{Q_i} matrix belongs to γC_Q matrix, i.e. until all its NDep will be computed.

The last two terms of the equation 4.1 form the symmetric difference of the sets of non measurable dependencies $Dep^u(Q)$ and $Dep^u(Q_i)$ of the system and the model respectively, considering in this case as an error the number of missing NDep of Q in Q_i plus the number of NDep computed in the model Q_i which do not belong to Q. The symmetric difference of the set of columns of the output matrices φC_Q and φC_{Q_i} was not considered since, as it will be presented in section 4.4, all the computed columns in φC_{Q_i} matrix are columns of φC_Q matrix. Thus, for the measurable part of a system Q it is only considered as an error the number of columns of φC_Q matrix which are not still computed in the model Q_i.

The equation 4.1 determines the identification error in terms of the missed elements of Q (columns of φC and NDep) in Q_i and also in the number of NDep wrongly computed in Q_i. The error $f(Q, Q_i)$ will be zero when all columns of φC_Q and all NDep of Q be computed and the wrong NDep in Q_i be removed; i.e. when $C_Q = C_{Q_i}$.

Example. Consider the IPN depicted on figure 4.6.a and 4.6.b be the system Q that need to be identified and its computed model Q_i when the transition sequence $t_1t_2t_3t_4$ was fired respectively; then the error is $f(Q, Q_i) = 5$, which is computed as follows:

$|\{\varphi C_Q\} - \{\varphi C_{Q_i}\}| = 2$ because the transitions t_5 and t_6 are not computed in Q_i,

$|Dep^u(Q) - Dep^u(Q_i)| = 3$ because the NDep $[t_4, t_5]$, $[t_6, t_5]$ and $[t_6, t_1]$ are not still computed in Q_i.

$|Dep^u(Q_i) - Dep^u(Q)| = 0$ because the computed NDep $[t_2, t_3]$ and $[t_4, t_1]$ belong to Q.

This error will decrease when the transitions t_5 and t_6 and the NDep $[t_4, t_5]$, $[t_6, t_5]$ and $[t_6, t_1]$ be computed inferring completely the place p_4 of Q.

It is important consider the following:

Even though the function f describes the computation error, it cannot be used to guide the algorithm to compute a new model since the system S_f is unknown, however f will be used to prove the convergence of the computed model Q_n to the actual model of the system Q.

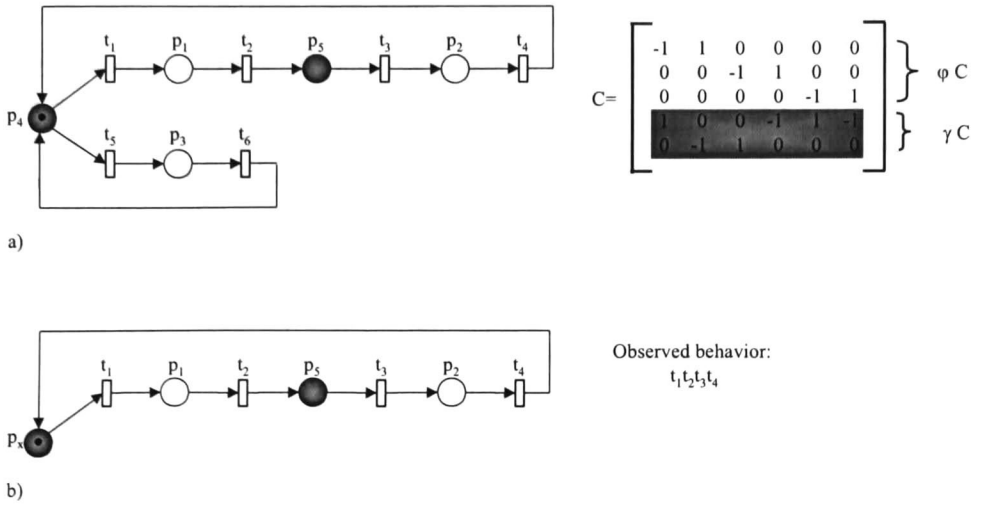


Figure 4.6: a) System model Q that need to be identified. b) Computed model Q_i for Q when the observed behavior is the firing transition sequence $t_1t_2t_3t_4$.

- Even more, the IPN model Q of a DES S_f is unknown in principle; the hypothesis of Q known is held just for proving the convergence of the proposed identification technique.

- For a better explanation of the identification procedure a DES S_f will be considered as its IPN representation Q, which also will be called *system model* or simply system.

4.4 Incremental modelling

The asymptotic identification problem states the computation of a sequence of models fulfilling that any model Q_i in this sequence is better than the previous model Q_{i-1} in the sense that Q_i is more approached to the actual model of the system than Q_{i-1} (problem definition, statement 2). Each model Q_i will be computed updating the previous computed model Q_{i-1} as new information of the system is detected (problem definition, statement 3). The proposed method to compute a model Q_i will consist in compute its incidence matrix. As introduced in previous chapter the incidence matrix C_Q of an IPN Q can be decomposed as $C_Q = \begin{bmatrix} \varphi C_Q \\ \gamma C_Q \end{bmatrix}$, where φC_Q and γC_Q are the matrices representing the measurable and non measurable part of Q respectively.
The only information that it is possible to detect from direct measurement of a system model Q is its change of state over the measurable places, this leads to compute the measurable part of its transitions represented by the columns of φC matrix. However the non measurable places must to be inferred from the observed behavior of Q forming dependencies between any two transitions t_i and t_j computed consecutively represented by the columns i and j of φC matrix, thus the γC matrix will be computed by rows instead of by columns like φC matrix.
Hence the identification process will consist in compute the columns of φC_Q matrix and infer the rows of γC_Q matrix as the system evolves.

Identification considering only the output information of the system

For a clearer explanation of the identification procedure, in the remaining of this chapter will be considered that every change of state of a system model Q can be detected from its output information, hence a model Q_i will be built using only the output information of Q. The identification procedure will require that all columns of φC can be detected and uniquely determined from its output information. Hence in this chapter will be considered a particular case of the event detectability property presented in section 2.3 in which the events can be uniquely determined by the output information of the system.

Definition 4.7 *Let Q be an IPN. Q is event-detectable by the output if every transition firing can be uniquely determined by the knowledge of its output signals.*

The characterization of the event-detectability property considering only the output information of the system is that all the columns of φC matrix be non null and different from each other. This is stated in the next proposition.

Proposition 4.1 *Let Q be an IPN. Q is event-detectable by the output iff:*

1. $\forall t_i \in T$ *it is fulfilled that* $\varphi C\overrightarrow{t_i} \neq \overrightarrow{0}$ *and*
2. $\forall t_i \neq t_j \in T$ *it holds that* $\varphi C\overrightarrow{t_i} \neq \varphi C\overrightarrow{t_j}$

Proof. The proof follows from the proof of proposition 2.3. However as in this case are not considered the input signals then to distinguish any two transitions it is required that all columns of φC matrix be different from each others since in other case the transition represented by equal columns cannot be distinguished even if they have different input signal assigned. ■

In chapter 6 is outlined the case in which the inputs are used in the identification procedure.

This section provides the related concepts and procedures needed to compute each model in the sequence in order to identify incrementally a desired DES.

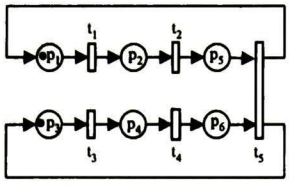
4.4.1 Computation of measurable part: φC matrix

Since a column i of the incidence matrix C_Q of an IPN Q represents the number of tokens removed or added from its input places to its output places respectively when the transition t_i is fired, then a column of C_Q matrix can be computed from two consecutive markings of Q.

Consider the IPN state equation $M_{k+1} = M_k + Cv_k$, assume that the marking M_{k+1} is reached from M_k firing the transition t_j, $M_k \xrightarrow{t_j} M_{k+1}$, the firing vector in this case will be $v_k(i) = \begin{cases} 1 & \text{if } i = j \\ 0 & \forall i \neq j \end{cases}$ since only t_j is fired, the product Cv_k is the $j-th$ column of C representing the transition t_j, this column will be represented as $C\overrightarrow{t_j}$. The state equation can be rewritten as $M_{k+1} = M_k + C\overrightarrow{t_j}$, thus

$$C\overrightarrow{t_j} = M_{k+1} - M_k \tag{4.2}$$

Example. Consider the IPN Q depicted on figure 4.7, assume that the transition t_1 is fired from the



a) Interpreted Petri net Q

$$C=\begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

b) Incidence matrix of Q

Figure 4.7: IPN Q with its incidence matrix.

initial marking M_0 reaching the marking M_1: $\overset{M_0}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \xrightarrow{t_1} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ The transition t_1 could be computed using the equation 4.2 as: $\overset{M_1}{\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} - \overset{M_0}{\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} = \overset{C\overrightarrow{t_1}}{\begin{bmatrix} -1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}}$. This computed column is the first column of C matrix depicted on figure 4.7.b.

The following proposition states how to compute a column of an incidence matrix C.

Proposition 4.2 *Let Q be an IPN and σ_i and σ_j be any two sequences which contain the transition t_k such that $M_x^i \xrightarrow{t_k} M_y^i$ and $M_x^j \xrightarrow{t_k} M_y^j$, then the column $C\overrightarrow{t_k}$ can be computed from any two consecutive markings as $C\overrightarrow{t_k} = M_y^i - M_x^i = M_y^j - M_x^j$*

Proof. $M_y^i = M_x^i + C\overrightarrow{t_k} \rightarrow M_y^i - M_x^i = C\overrightarrow{t_k}$ similarly $M_y^j = M_x^j + C\overrightarrow{t_k} \rightarrow M_y^j - M_x^j = C\overrightarrow{t_k}$
Hence the column k of C matrix representing the transition t_k can be computed as $C\overrightarrow{t_k} = M_y^i - M_x^i = M_y^j - M_x^j$
■

Consider the class of completely measurable system models (in which all the places are measurable places), then the previous proposition states a procedure to identify any model of the system model of this class since an output symbol φM_k of a completely measurable system model is the entire marking M_k of Q. However, more interesting systems for the identification problem are those in which the entire information about the system state is not accessible from its output measurement, and hence only partial information of the system state is available. The aim of this thesis is to state an identification procedure for this class of DES. The remaining of this section is devoted to compute the φC matrix of a system model Q from the observed output symbols.
By definition of the output function φ (definition 2.10), an output symbol $y_k = \varphi M_k$ (equation 2.2) is the marking of a system model Q over its measurable part, i.e. is the marking of the measurable places of Q in the marking M_k.

Example. Consider the IPN Q depicted on figure 4.8. The φ matrix of Q is $\varphi_Q = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}$

Notice that there exists not any row in φ_Q in which there exists a 1 at positions 5 or 6 since the places p_5 and p_6 are non measurable places.

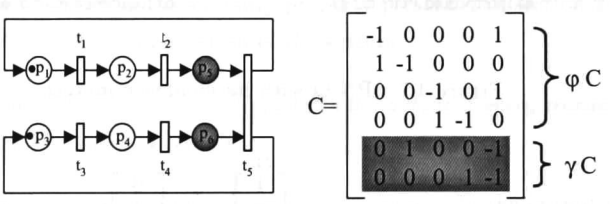


Figure 4.8: System model Q and its incidence matrix.

The resulting matrix φC of Q is a submatrix of C in which only are considered the rows of the measurable places. On figure 4.8 it is presented the matrix φC of the system model Q.

By definition 2.3 of the φ matrix of an IPN Q, an output symbol φM_k generated by Q, will be the marking M_k of Q considering only its measurable places.

The first output symbol φM_0 generated by the IPN Q depicted on figure 4.8 is

$\varphi M_0 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}^{\varphi} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}^{M_0} = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}^{\varphi M_0}$. When the event related with the transition t_1 is executed the output symbol generated by Q is $\varphi M_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{bmatrix}^{\varphi} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}^{M_1} = \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}^{\varphi M_1}$. Notice then that φM_0 and φM_1 are the markings of the measurable places in the markings M_0 and M_1 respectively.

Based on the procedure to compute a column of the incidence matrix of a completely measurable IPN is presented how to compute the matrix φC of an IPN.

Proposition 4.3 *Let Q be an event detectable IPN. Each column $\varphi(C\overrightarrow{t_i})$ of the φC matrix can be computed from the output symbols of Q as $\varphi(C\overrightarrow{t_i}) = \varphi(M_{k+1}) - \varphi(M_k)$, where $\varphi(M_{k+1})$ and $\varphi(M_k)$ are any two consecutive output symbols generated by Q.*

Proof. The proof follows from proposition 4.2, then each column i of φC_Q matrix can be computed from any two consecutive output symbols $\varphi(M_i)$ and $\varphi(M_{i-1})$ as follows:

Assume that M_k and M_{k+1} are any two consecutive markings of Q such that M_{k+1} is reached from M_k firing the transition t_i, i.e.$M_k \xrightarrow{t_i} M_{k+1}$.

$$\begin{array}{ll} M_{k+1} = M_k + C\overrightarrow{t_i} & \text{(state equation)} \\ \varphi(M_{k+1}) = \varphi(M_k + C\overrightarrow{t_i}) & \text{(applying } \varphi \text{ to both sizes)} \\ \varphi(M_{k+1}) = \varphi(M_k) + \varphi(C\overrightarrow{t_i}) & \text{(Due to } \varphi \text{ is a linear function)} \\ \varphi(C\overrightarrow{t_i}) = \varphi(M_{k+1}) - \varphi(M_k) & \text{(solving for } \varphi(C\overrightarrow{t_i})) \end{array}$$

where $C\vec{t_i}$ is the column i of C matrix representing the transition t_i and then $\varphi(C\vec{t_i})$ is the column i of φC matrix representing the measurable part of the transition t_i.
Since Q is assumed to be event-detectable by the output (proposition 4.1) then all columns of φC matrix can be detected and uniquely determined as a difference of consecutive markings, due to all of them are not null and different from each other. ■
This proof gives a strategy to compute the measurable part (φC matrix) of an IPN Q using only its output symbols.

Proposition 4.4 *Let Q be a system model and let Q_i be the computed model for Q. If a new column of φC matrix is computed then* $|\{\varphi C_Q\} - \{\varphi C_{Q_i}\}| < |\{\varphi C_Q\} - \{\varphi C_{Q_{i-1}}\}|$

Proof. By proposition 4.3 each computed column $\varphi(C\overrightarrow{t_j})$ belongs to φC matrix of Q, hence if $\varphi(C\overrightarrow{t_j})$ is a new computed column of φC_Q matrix the term $|\{\varphi C_Q\} - \{\varphi C_{Q_i}\}|$ of the identification error equation 4.1 is reduced in one unit and hence $|\{\varphi C_Q\} - \{\varphi C_{Q_i}\}| < |\{\varphi C_Q\} - \{\varphi C_{Q_{i-1}}\}|$. ■

Notice that if the terms $|Dep^u(Q) - Dep^u(Q_i)|$ and $|Dep^u(Q_i) - Dep^u(Q)|$ remains without change i.e. the non measurable places were not modified in Q_i or are reduced because either a new non measurable place was correctly computed or a wrong non measurable place was removed in Q_i then it is fulfilled that $f(Q, Q_i) < f(Q, Q_{i-1})$ when a new transition is computed.

Let Q be an IPN, if there exists the following sequence of reachable markings $M_i \xrightarrow{t_a} M_j \cdots M_u \xrightarrow{t_b} M_x \xrightarrow{t_c} M_y$ then Q could generates the output word $w_0 = \varphi(M_i)\varphi(M_j)\cdots\varphi(M_u)\varphi(M_x)\varphi(M_y)$, each transition in the transition sequence $w = t_a \cdots t_b t_c$ generating w_0 is computed using proposition 4.3 as: $t_a = \varphi(M_j) - \varphi(M_i)$, $t_b = \varphi(M_x) - \varphi(M_u)$ and $t_c = \varphi(M_y) - \varphi(M_x)$. Notice then that each transition t_i is a column of the matrix φC_Q.

Example. Consider the IPN Q depicted on figure 4.8, the IPN to be identified. The non measurable part of Q is computed following the procedure described above as:

$$\overset{\varphi M_1}{\begin{bmatrix}1\\0\\1\\0\end{bmatrix}} \xrightarrow[\varphi M_2 - \varphi M_1]{t_1} \begin{bmatrix}-1\\1\\0\\0\end{bmatrix} \overset{\varphi M_2}{\begin{bmatrix}0\\1\\1\\0\end{bmatrix}} \xrightarrow[\varphi M_3 - \varphi M_2]{t_3} \begin{bmatrix}0\\0\\-1\\1\end{bmatrix} \overset{\varphi M_3}{\begin{bmatrix}0\\1\\0\\1\end{bmatrix}} \xrightarrow[\varphi M_4 - \varphi M_3]{t_2} \begin{bmatrix}0\\-1\\0\\0\end{bmatrix} \overset{\varphi M_4}{\begin{bmatrix}0\\0\\0\\1\end{bmatrix}} \xrightarrow[\varphi M_5 - \varphi M_4]{t_4} \begin{bmatrix}0\\0\\0\\-1\end{bmatrix} \overset{\varphi M_5}{\begin{bmatrix}0\\0\\0\\0\end{bmatrix}} \xrightarrow[\varphi M_5 - \varphi M_1]{t_5} \begin{bmatrix}1\\0\\1\\0\end{bmatrix} \overset{\varphi M_1}{\begin{bmatrix}1\\0\\1\\0\end{bmatrix}}$$

where all the elements $\varphi(M_i) - \varphi(M_{i-1})$ are the columns of the matrix φC_Q.

Remark. *Using this approach it is not possible to detect the event* $t_j \| t_k$ (the simultaneous firing of t_j and t_k); thus *the detection of a multiple change in the entries of the output symbol is assumed to be caused by the firing of one transition.* ■

Notice that each column i of φC_Q represents only the input and output measurable places of the transition t_i. The following sections are devoted to state when and how to infer the input and output non measurable places of a computed transition.

4.4.2 Condition for inferring the matrix γC

Until now a procedure to compute the φC matrix of an IPN Q has been presented. Before to define the procedure to compute the non measurable places of Q it is needed to define the conditions under which they can be computed.

Since the systems considered in this thesis are those that can be described by a live and bounded IPN, then they must exhibit a cyclic behavior. A cycle has the property to be a repetitive part of the system and hence it can be detected when a sequence of events in the system allows to reach a state (marking) previously reached. In IPN terms the t-semiflows represent the cycles of a system, due to they induce the repetitive components of an IPN. Consider the sequence of markings $M_i \xrightarrow{t_x} M_j \xrightarrow{t_y} \cdots \xrightarrow{t_z} M_i$ reached when each transition of the transition sequence $\sigma = t_x t_y \cdots t_z$ is fired, $\overrightarrow{\sigma}$ is a t-semiflow since the marking M_i is reached again after the firing of $\sigma = t_x t_y \cdots t_z$ from M_i.

Example. In the IPN depicted on figure 4.9 there are two cycles which are described by the t-semiflows $X_1 = t_1 t_2 t_3 t_4 t_5 t_6$ and $X_2 = t_7 t_8 t_9$.

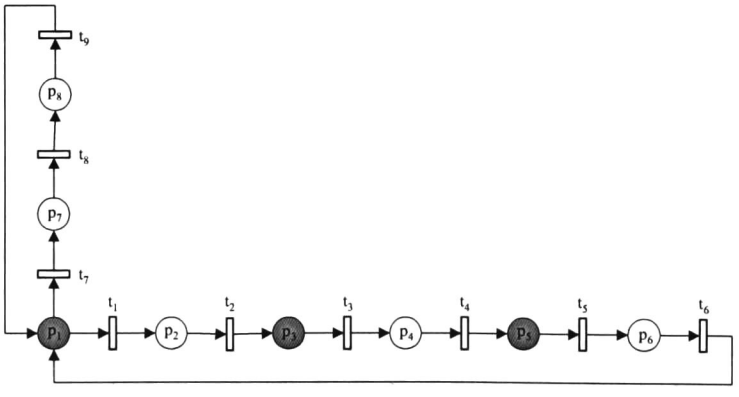


Figure 4.9: Petri net with two t-semiflows and four m-words.

The measurable part of an IPN Q can be computed directly from any two observed consecutive output symbols of Q as stated in proposition 4.3. However, the inference of the non measurable part of Q will be made when a t-semiflow is detected. Hence it is important to state how to detect a t-semiflow from the measurement of the output symbols of the system. Thus, the notion of a m-word is presented to specify that a possible cycle is detected by the output.

Definition 4.8 *Let Q be an IPN, $w_o = \varphi(M_i)\cdots\varphi(M_j)$ be an output word generated by Q and $\sigma = t_1\cdots t_k$ be the firing sequence detected when w_o is observed, such that each transition t_i is computed using proposition 4.3, then σ is a m-word iff $\varphi(M_i) = \varphi(M_j)$.*

By definition of t-semiflow, when a marking M_i in an IPN has been already reached then a t-semilflow is detected. However, it could occurs that when an output symbol is repeated the underlying firing sequence is not a t-semiflow of the IPN.

Consider the IPN Q depicted on figure 4.10. The initial marking $M_0 = \begin{bmatrix} 0\\0\\0\\1\\0\\1 \end{bmatrix}$ produces the output symbol $\varphi M_0 = \begin{bmatrix} 0\\0\\0 \end{bmatrix}$; applying the firing sequence t_1t_2 the marking $M_i = \begin{bmatrix} 0\\0\\0\\0\\1\\1 \end{bmatrix}$ is reached, which produces the output symbol $\varphi M_i = \begin{bmatrix} 0\\0\\0 \end{bmatrix}$. According with definition 4.8 a m-word $w_k = t_1t_2$ is detected since $\varphi M_0 = \varphi M_i$, however $M_0 \neq M_i$, hence t_1t_2 is not a t-semiflow of Q.

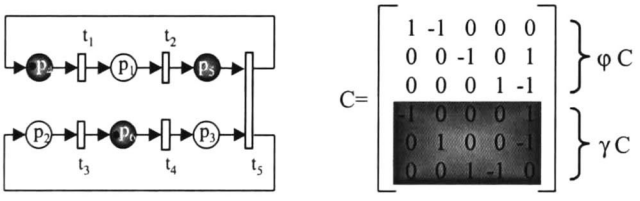


Figure 4.10: IPN Q and its incidence matrix.

Notice then that a m-word is not always a t-semiflow of an IPN. However a m-word is the approximation of a t-semiflow that can be detected from the output of an IPN; indeed a m-word m_i of an IPN Q fulfills that $\varphi C_Q \bullet \overrightarrow{m_i} = 0$.

As illustrated on figure 4.10, $\varphi C_Q = \begin{bmatrix} 1 & -1 & 0 & 0 & 0\\ 0 & 0 & -1 & 0 & 1\\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$ and $\overrightarrow{m_i} = \begin{bmatrix} 1\\1\\0\\0\\0 \end{bmatrix}$ then $\varphi C_Q \bullet \overrightarrow{m_i} = \begin{bmatrix} 0\\0\\0 \end{bmatrix}$.

Consider again the IPN Q depicted on figure 4.9. Notice that there exist two t-semiflows in Q: $X_1 = t_1t_2t_3t_4t_5t_6$ and $X_2 = t_7t_8t_9$, and four m-words $w_1 = t_1t_2$, $w_2 = t_3t_4$, $w_3 = t_5t_6$ and $w_4 = t_7t_8t_9$. Notice

also that X_1 is the concatenation of the m-words w_1, w_2 and w_3 i.e., $X_1 = w_1 w_2 w_3$, while the m-word w_4 is a t-semiflow of Q, $X_2 = w_4$.

Hence will be considered that every t-semiflow of an IPN Q have an m-word decomposition.

Remark. *The m-word decomposition of a t-semiflow will depend on the allocation of the non measurable places in the underlying t-component and also in the marking of the system in which its measurement begin.* ■

The non measurable places of a system model Q will begin to be inferred when a m-word is detected. Next are presented the procedures to infer non measurable places.

4.4.3 Inference of non measurable part: γC matrix

The computation of the non measurable places (rows of matrix γC) is not as straight as the computation of the measurable places (rows of matrix φC); since as was presented in section 4.4.1, all columns of φC_{Q_i} matrix are computed correctly and directly from the output symbols of the system, however some non measurable places need to be inferred from several evolutions of the system. The non measurable places will be computed according to: 1) preserve the firing order of the transitions in the current m-word and 2) preserve the order in which the m-words have been observed. This is to preserve in the computed model the observed behavior of the system. Hence at the beginning of the identification procedure some non measurable places could be wrongly computed, however, with new information provided by the system a wrong non measurable place will be progressively corrected yielding models that represent the observed system behavior, by this reason it was stated that a model Q_i for a system model Q will be computed incrementally as Q evolves.
This section is devoted to introduce the needed procedures to add, update (adding or removing input or output arcs to an already computed place or merging places) or remove NDep in order to compute a new IPN model which agrees with the observed system behavior; with these procedures the matrix γC_Q is inferred when a m-word is detected.
The following situations may arise when a dependency in a computed model Q_{n-1} need to be updated in order to identify a new model Q_n.
For each NDep $[t_i, t_j] = p_k$, a vector $u_k = \begin{bmatrix} v_1 & \cdots & v_i & \cdots & v_j & \cdots & v_r \end{bmatrix}$ is computed; where r is the number of detected transitions and $v_i = 1$, $v_j = -1$ and $v_x = 0\ \forall x \neq i, j$. In order to obtain the incidence matrix of the computed model Q_i, the matrix $\varphi C_{Q_i}(\cdot, t_i)$ and u_{ij} vectors could be arranged as follows:

$$C = \begin{bmatrix} \varphi C(\cdot, t_1) & \cdots & \varphi C(\cdot, t_r) \\ & u_j & \\ & \vdots & \\ & u_k & \end{bmatrix}$$

where the u_i vectors are the rows of the matrix γC_{Q_i}.
The updating of a non measurable place p_k is made when a new NDep is computed using p_k, when some NDep formed with p_k are removed or when two places p_i and p_j are merged forming a new place p_k. All updating operations are performed on the incidence matrix C_{Q_i} of the computed model Q_i as follows:

- If there exists a NDep $[t_i, t_j] = p_k$ and it is needed to form another NDep $[t_x, t_y]$ using the same non measurable place p_k, then an arc is added from t_x to p_k and another arc is added from p_k to t_y, while in

the incidence matrix C_{Q_i} it will be added the elements -1 and 1 in the positions $C_{Q_i}[k,x]$ and $C_{Q_i}[k,y]$ respectively.

- If a NDep $[t_i,t_j] = p_k$ must to be removed and p_k belongs to another NDep, then the input and the output arcs of p_k related with t_i and t_j are removed; only in the case when p_k belongs to a single NDep the place p_k can be removed, while in the incidence matrix C_{Q_i} the elements $C_{Q_i}[k,i]$ and $C_{Q_i}[k,j]$ are set in zero, if p_k belongs to only the NDep $[t_i,t_j]$ then the row k of C_{Q_i} is removed.

- If p_i and p_j are two non measurable places to be merged, then a new non measurable place p_k is computed such that ${}^\bullet p_k = {}^\bullet p_i \cup {}^\bullet p_j$ and $p_k^\bullet = p_i^\bullet \cup p_j^\bullet$. In the incidence matrix C_{Q_i} the rows i and j are added forming the row k, and the rows i and j are removed from C_{Q_i}.

4.4.4 Non measurable places classification

In order to compute the matrix γC_Q, the non measurable places are classified according the membership of its input and output transitions of a given m-word.

Definition 4.9 *Let p_k be a non measurable place of an IPN Q. If $\forall\ [t_i,t_j] \in Dep^u_{p_k}$ t_i and t_j belong to the same m-word then p_k is a non measurable place of Class A, otherwise p_k is a non measurable place of Class B.*

The characterization of the classes given in previous definition is given below.

Class A: *The non measurable places forming NDep between transitions belonging to the same t-component which underlying t-semiflow does not have a m-word decomposition.*

The underlying t-semiflows of t-components in which the places of this class belong do not have a m-word decomposition, hence when a m-word is determined is also determined an actual t-semiflow of the system model. Hence in a NDep $[t_i,t_j] = p_k$ such that p_k belongs to Class A, t_i and t_j belong to the same t-semiflow (m-word).

Two kind of non measurable places can be distinguished into this class:

A.i. The non measurable places of Class A forming a single NDep.

A.ii. The non measurable places of Class A forming a complex NDep.

These non measurable places can be computed from any two consecutive transitions t_i, t_j in a m-word w_n to preserve the firing order in which such transitions are computed.

An example of these classes of non measurable places is given on figure 4.11. In figure 4.11.a, the non measurable place p_2 belongs to Class A.i, while in figure 4.11.b, the non measurable places p_3 and p_6 belong to Class A.ii.

Class B: *The non measurable places forming NDep between transitions of different m-words.*

The underlying t-semiflows of t-components in which the places of this class belong have a m-word decomposition, hence when a m-word is determined it is possible that an actual t-semiflow has not been determined. Hence in a NDep $[t_i,t_j] = p_k$ such that p_k belongs to Class B, t_i and t_j belong to different m-words.

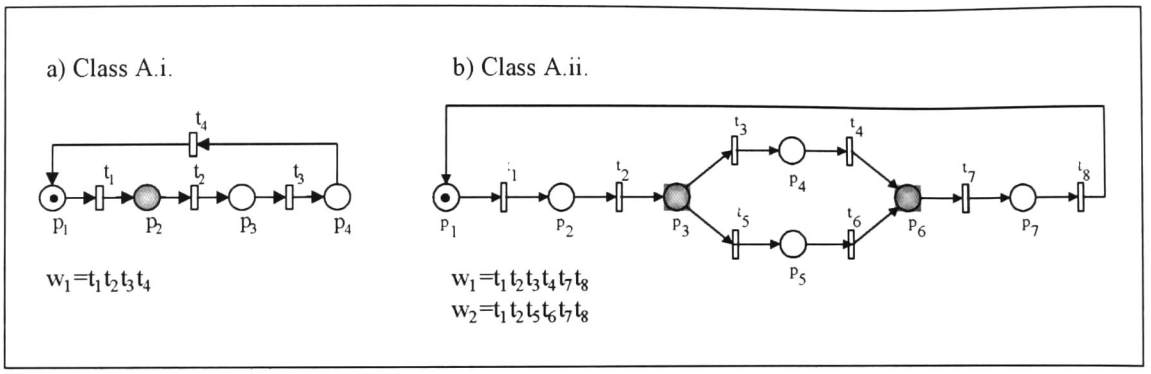


Figure 4.11: Non measurable places of Class A.

Two kind of non measurable places into this class can be distinguished:

B.i. Non measurable places of Class B forming a single NDep between transitions belonging to different m-words in a t-semiflow decomposition.

B.ii. Non measurable places of Class B forming a complex NDep

- **a.** between transitions belonging to different m-words of the same t-semiflow decomposition.
- **b.** between the last and the first transitions of different t-semiflows of Q.

These non measurable places are computed to preserve the occurrence order of the computed m-words concatenating the previous and the current m-words. The updating of this class of non measurable places will be made when new m-words be detected, also it is possible that as the system evolves new occurrence order of the m-words could be detected, in this case a non measurable place is updated merging the output and the input non measurable places of last and the first transitions of the previous and the current m-word respectively.

An example of non measurable belonging to these classes is given on figure 4.12. In figure 4.12.a, the non measurable places p_1, p_3 and p_5 belong to Class B.i, In figure 4.12.b, the non measurable places p_3 and p_6 belong to Class B.ii.a, while in figure 4.12.c, the non measurable place p_5 belong to Class B.ii.b.
The previous classification of non measurable places is given according to the m-word decomposition of the t-semiflows into the system model Q. Thus, the Class A represents the non measurable places in which their inputs and output transitions belong to the same m-word, indeed this m-word is an actual t-semiflow of Q. The Class B represents the non measurable places in which their input and output transitions could belong to different m-words. The input and output transition of a non measurable place of Classes B.i.a and B.ii.a belong to different m-words of a same t-semiflow, while the input and output transitions of a non measurable place of Class B.ii.b are the last and the first transitions of actual t-semiflows of Q. These places are computed by sequencing the last and the first transitions of the previous and the current m-words of different t-semiflows. It is important to notice that the places belonging to subclass i are those forming single NDep, while the places belonging to subclass ii are those forming complex NDep.

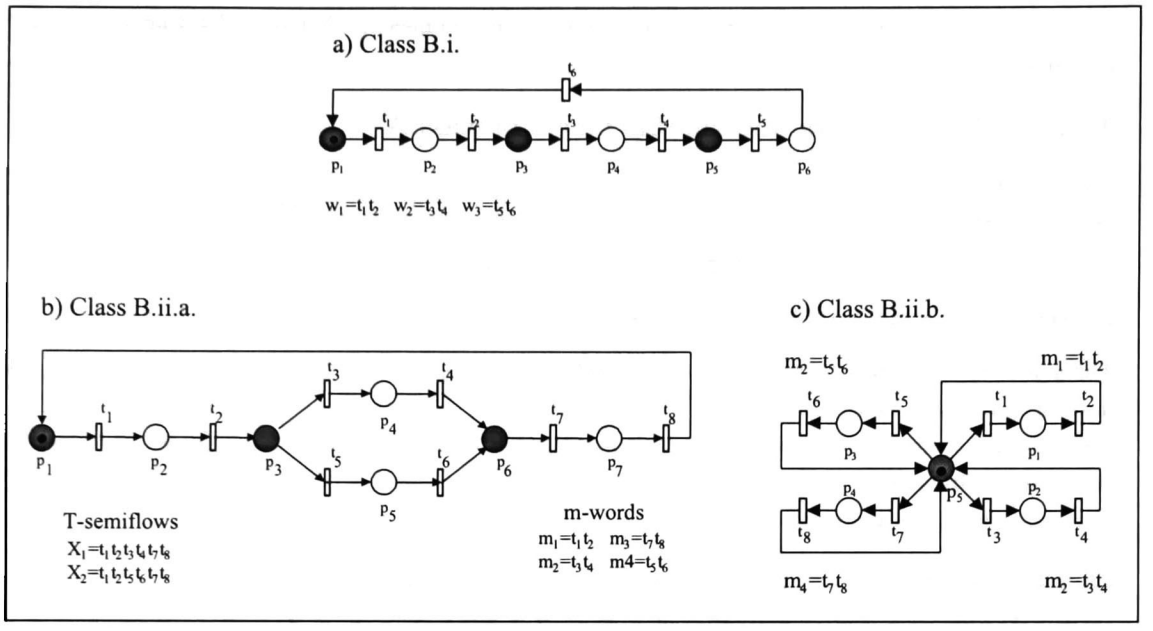


Figure 4.12: Non measurable places of Class B.

If two IPN have the same structure but different interpretation or initial marking then it is possible that their non measurable places do not belong to the same class.

Example. Consider the IPN Q_1 and Q_2 depicted on figure 4.13. Although these two IPN have the same structure, the location of their non measurable places is different. This fact leads to a different m-word decomposition of their t-semiflows. These IPN only have one t-semiflow $X_1 = t_1t_2t_3t_4$.

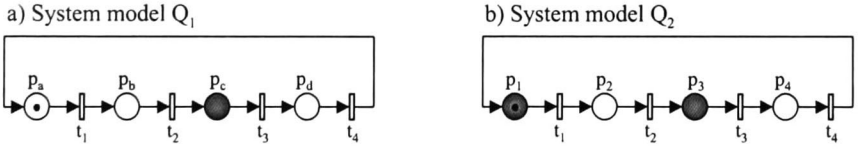


Figure 4.13: Different allocation of the non measurable places in two structurally equal IPNs.

The m-word decomposition of Q_1 is $w_1 = t_1t_2t_3t_4$, while the m-word decomposition of Q_2 is $w_1 = t_1t_2$ and $w_2 = t_3t_4$. Then the m-word decomposition of Q_1 is not the same m-word decomposition of Q_2 and their respective non measurable places belong to different classes:

Consider the non measurable place $p_c = [t_2, t_3]$ of Q_1; since t_2 and t_3 belong to the same m-word, then p_c belong to Class A. In turn for the non measurable place $p_3 = [t_2, t_3]$ of Q_2, t_2 and t_3 belong to different m-words, thus p_3 belongs to the class B.

For every subclass it is defined a procedure used to infer the non measurable places. Next are described these procedures.

4.4.5 Inference of non measurable places of class A

The main features of the non measurable places belonging to this class is that their input and output transitions belong to the same m-word and this m-word is an actual t-semiflow of a system model Q, i.e. this t-semiflow is composed by only one m-word.

Inference of non measurable places of Class A.i

The places belonging to this class form a single NDep $p_k = [t_i, t_j]$ between two transitions t_i and t_j of a same m-word of a system model Q.
The strategy to compute these non measurable places consists in preserve the firing order of the transitions to the observed firing order in the current computed m-word w_n forming a NDep between any two consecutive transitions in w_n which are not connected.

Identification Step 4.1 *Let Q be a system model, Q_n be the computed model for Q, $w_n = \cdots t_i t_j \cdots$ be the current computed m-word from the observed behavior of Q and t_i and t_j be any two consecutive transitions in w_n. If there exists not a dependency $[t_i, t_j]$ in Q_n then the non measurable place $p_x = [t_i, t_j]$ can be computed to preserve the firing of t_i before t_j in Q_n.*

Proposition 4.5 *Let Q be a system model. A non measurable place p_k of Class A.i can be computed using identification step 4.1.*

Proof. A non measurable place of Class A.i fulfills that 1) it forms a single NDep i.e. p_k only has one input and one output transition 2) its input and output transitions belong to same m-word. A non measurable place computed using identification step 4.1 is computed between any two consecutive transitions t_i and t_j in the current computed m-word $w_n = \cdots t_i t_j \cdots$ such that the dependency $[t_i, t_j]$ does not exist in $Dep(Q_i)$. If p_k is the computed place then $p_k = t_i^\bullet =^\bullet t_j$, hence p_k fulfills both conditions of places belonging to Class A.i. ■

The underlying procedure described by identification step 4.1 is the computation of a t-component by sequencing all the non connected consecutive transitions in w_n, because it is assumed that a detected m-word is a t-semiflow of a system model. Indeed the last and the first transitions of w_n are considered as consecutive transitions. The next example illustrate this fact.

Example. Consider the IPN Q depicted on figure 4.14. Assume that the m-word $w_1 = t_1 t_2 t_3 t_4 t_5$ is computed from an output sequence of Q, then the columns of φC representing each transition of w_1 are the following: $\overset{t_1}{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \overset{t_2}{\begin{bmatrix} -1 \\ 1 \\ 0 \\ 0 \end{bmatrix}} \overset{t_3}{\begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}} \overset{t_4}{\begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \end{bmatrix}} \overset{t_5}{\begin{bmatrix} 0 \\ -1 \\ 0 \\ -1 \end{bmatrix}}$. Notice that t_2 and t_3, and t_5 and t_1 are consecutive transitions in w_1; however there exist not a place forming any dependency between these transitions. This fact can be detected since there exists not a row in φC matrix in which there exists 1 in the column of t_2 and a -1 in the column of t_3 and either there exist a row in which there exists 1 in the column of t_5

and a -1 in the column of t_1. Hence the NDep $p_x = [t_2, t_3]$ and p_y=$[t_5, t_1]$ are computed. On figure 4.15 is depicted the computed model Q_1.

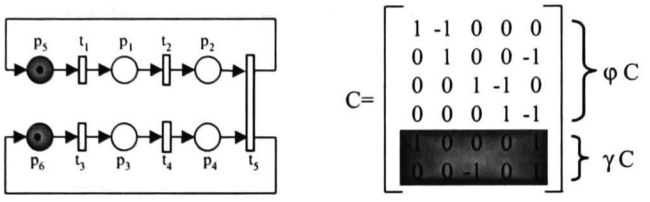


Figure 4.14: IPN Q and its incidence matrix.

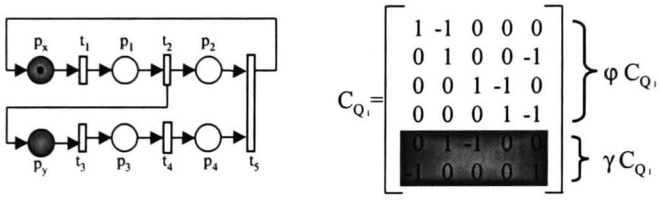


Figure 4.15: Computed model Q_1 from the m-word $w_1 = t_1t_2t_3t_4t_5$.

In previous example, the only non measurable place computed correctly is the place p_x, since the place p_y does not belong to Q, however it is computed to preserve the firing order of the transition t_2 and t_3 as stated in w_1. Notice then that p_x belongs to Class *A.i.*

It is important consider the following when a non measurable place is computed using identification step 4.1:

1. In order to identify a NDep $p_k = [t_i, t_j]$ of Class A.i, it is needed that the transitions t_i and t_j occurs consecutively in a m-word. For example the place p_6 of the IPN Q depicted on figure 4.14 is a non measurable place belonging to Class A.i., however it could not be computed from w_1 since t_5 and t_3 did not occur consecutively in w_1.

   - However the missing NDep $[t_i, t_j]$ will be computed when in a new m-word t_i and t_j occur consecutively.

2. Since the procedure stated in identification step 4.1 computes a t-component with the transitions of the current m-word w_n, if w_n is not a t-semiflow of Q then the NDep computed between the last and the first transitions of w_n is wrongly computed. However the firing order of the transitions in w_n is preserved. In this case the term $|Dep^u(Q_i) - Dep^u(Q)|$ of the identification error equation is augmented.

   - The wrongly NDep computed as above is removed when a future m-word is detected, computing a non measurable place of Class B.i concatenating the previous and the current m-word. The concatenation procedure is presented in next section.

3. As it is shown in figure 4.16, the transitions t_2 and t_3 are concurrent transitions and hence there could not exist a non measurable place constraining their firing, however at early stages of the identification process these kind of NDep need to be computed to preserve the observed behavior of the system. In this case the term $|Dep^u(Q_i) - Dep^u(Q)|$ of the identification error equation is augmented because the computed place does not belong to Q.

   - The wrongly computed NDep between concurrent transitions will be removed in a posterior stage of the identification procedure when in another m-word the concurrent transitions change its firing order. The procedure to remove this kind of NDep is presented at the end of this section.

4. Only when the input and output transitions t_i and t_j of the computed non measurable place p_k belong to the same m-word which is also a t-semiflow of Q the term $|Dep^u(Q) - Dep^u(Q_i)|$ of the identification error equation is reduced.

   - Indeed if $p_k = [t_i, t_j]$ forms a single NDep then it is completely computed. However when t_i or t_j be shared transition by another t-semiflow, the computed NDep $[t_i, t_j] = p_k$ belongs to Q but the non measurable place p_k is not inferred completely. It is needed to compute another m-word (t-semiflow) to compute new NDep of the place p_k. In this case p_k is a place of Class A.ii. This procedure is presented below.

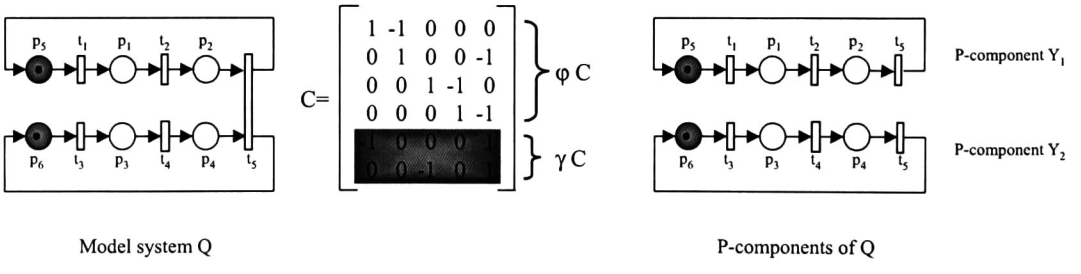


Figure 4.16: Undelying P-components of the p-semiflows $Y_1 = [11001]$ and $Y_2 = [00111]$ of Q.

Inference of non measurable places of Class A.ii.

If a place $p_k = [t_i, t_j]$ belongs to Class A.ii., then p_k forms a complex NDep such that t_i and t_j belong to a same t-semiflow X_p such that they occur consecutively in a transition sequence of a system model Q, however t_i or t_j belongs to at least another t-semiflow X_q. Hence the place p_k is a shared place at least by the underlying t-components of X_p and X_q and t_i or t_j are shared transition by the t-semiflows X_p and X_q.
In the IPN depicted on figure 4.17 the places p_3 and p_6 form NDep of class A.ii. In this example $p_3 = [t_2, t_3] = [t_2, t_5]$. The transitions t_3 and t_5 belong to different t-semiflows $X_1 = [\ 1\ \ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ ]^T$ and $X_2 = [\ 1\ \ 1\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 1\ ]^T$ respectively, however t_2 belongs to both t-semiflows, hence it is a shared transition by X_1 and X_2. Notice that the non measurable place p_3 belongs also to the t-components related with X_1 and X_2.

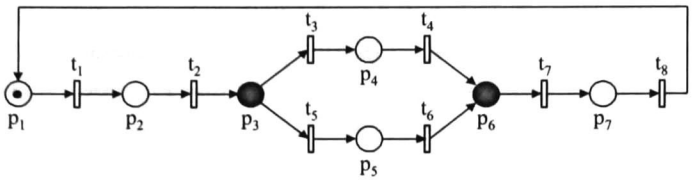


Figure 4.17: System model.

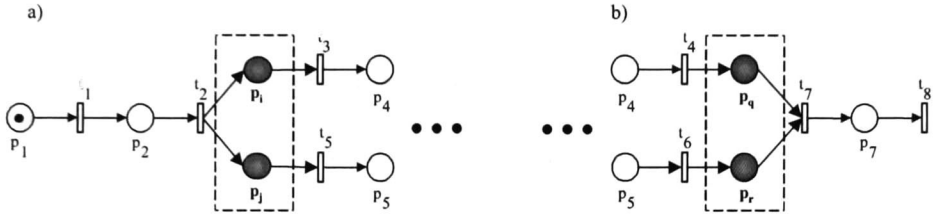


Figure 4.18: Non measurable places computed using Step identification 4.1.

Consider the system model Q depicted on figure 4.17, its m-words are $w_a = t_1t_2t_3t_4t_7t_8$ and $w_b = t_1t_2t_5t_6t_7t_8$ which are also its t-semiflows.

- Let $p_x = p_3$, $t_w = t_2$, $t_i = t_3$ and $t_r = t_5$.

  The transitions t_2 and t_3 are consecutive transitions in w_a while the transitions t_2 and t_5 are consecutive transitions in w_b. If it is used the identification step 4.1 to compute these non measurable dependencies then two non measurable places p_i and p_j are computed as depicted on figure 4.18.a. Notice that in this model t_3 and t_5 belongs to the same t-semiflow which is a contradiction since t_3 belongs to w_a and t_5 belongs to w_b, hence p_i and p_j must to be the same place.

  Let $p_y = p_6$, $t_j = t_4$, $t_s = t_6$ and $t_v = t_7$.

  The transitions t_4 and t_7 are consecutive transitions in w_a and the transitions t_6 and t_7 are consecutive transitions in w_b. If it is used the identification step 4.1 to compute these non measurable dependencies then two non measurable places p_q and p_r are computed as depicted on figure 4.18.b. Notice that in this model t_4 and t_6 belongs to the same t-semiflow which is a contradiction since t_4 belongs to w_a and t_6 belongs to w_b, hence p_q and p_r must to be the same place.

In order to cope with this situation it is stated how to compute a NDep using an already computed non measurable place to infer a non measurable place of Class A.ii.

Identification Step 4.2 *Let Q be a system model, Q_i be the computed model for Q, $w_n = \cdots t_it_j \cdots$ be the current computed m-word from the observed behavior of Q, and t_i and t_j be any two consecutive transitions in w_n. If there exists not a dependency $[t_i, t_j]$ in Q_i and*
Case 1: there exists a NDep $[t_i, t_k] = p_x$ or

Case 2: there exists a NDep $[t_k, t_j] = p_y$
such that t_k *does not belong to* w_n *then the NDep* $[t_i, t_j]$ *can be computed using an already computed place adding an arc from* p_x *to* t_j *or adding an arc from* t_i *to* p_y *for case 1 or case 2 respectively, forming the NDep* $[t_i, t_j] = p_x$ *or* $[t_i, t_j] = p_y$.

Proposition 4.6 *Let* Q *be a system model. A non measurable place of Class A.ii can be inferred using the identification step 4.2.*

Proof. A non measurable place of Class A.ii fulfills next conditions 1) it forms a complex NDep and 2) its input and output transitions belong to the same m-word which is also a t-semiflow of Q. Using identification step 4.2 a non measurable place is updated when two transitions t_i and t_j are consecutive transitions in the current m-word w_n, such that there exists a NDep $[t_i, t_k] = p_k$ or a NDep $[t_k, t_j] = p_y$ in Q_i and t_k does not belong to w_n. This two cases are next summarized:
case 1: If the NDep $[t_i, t_k] = p_x$ exists in Q_i then an arc is added from the place $p_x = [t_i, t_k]$ to the transition t_j to compute the NDep $[t_i, t_j] = p_x$, the place p_x forms a decision between t_j and t_k, since t_k does not belong to w_n then it is possible that Q_n generates the m-words: w_n and the m-word in which t_i and t_k were consecutive transitions.
or
case 2: If the NDep $p_y = [t_k, t_j]$ exists in Q_i then an arc is added from the transition t_i to the place $p_y = [t_k, t_j]$ to compute the NDep $[t_i, t_j] = p_y$, the place p_y forms an attribution between the transitions t_i and t_k, since t_k does not belong to w_n then it is possible that Q_n generates the m-words: w_n and the m-word in which t_k and t_j were consecutive transitions.
The fact that there exist the NDep $[t_i, t_k]$ or $[t_j, t_k]$ in Q_i such that t_k does not belong to the current m-word w_n in which t_i and t_j are consecutive transitions, implies that there exist another m-word in which t_i and t_k or t_k and t_j are consecutive transitions fulfilling the condition that t_i or t_j are shared transitions of at least two t-semiflows while the updated non measurable place p_x or p_y is a shared place by their underlying t-components. Hence a non measurable place of Class A.ii. is inferred using identification step 4.2 as new NDep $[t_i, t_j]$ are computed adding input or output arcs to an already computed non measurable place which is an output or an input place of t_i or t_j. ■

The underlaying procedure of identification step 4.2 consists in compute a NDep between any two consecutive transitions using an already computed non measurable place: If there exists not a dependency $[t_i, t_j]$ between any two consecutive transitions in the current m-word w_n but t_i has an output non measurable place p_x or t_j has and input non measurable place p_y i.e. there exists the NDep $[t_i, t_k] = p_x$ or the NDep $[t_k, t_j] = p_y$ in the computed model Q_i, then the NDep $[t_i, t_j]$ need to be computed using p_x or p_y. With this new computed NDep the place p_x or the place p_y is approached to an actual place of Q and the term $|Dep^u(Q) - Dep^u(Qi)|$ of the identification error equation is reduced.
The next algorithm presents how to infer a single or a complex NDep belonging to class A.i and A.ii respectively preserving the firing order of the transitions in the current computed m-word w_n.

Algorithm 4.1 *Computing NDep to constrain the firing order of the transitions in the current m-word w_n. Inference non measurable places belonging to class A.i and A.ii.*

Input: The $Dep(Q_n) = Dep(Q_{n-1})$ set and the current computed m-word $w_n = t_m \cdots t_n$.
Output: an updated $Dep(Q_n)$ set

---

1. If there exists not a MDep $[t_n, t_m]$ in $Dep(Q_n)$ then
   (a) add a NDep $[t_n, t_m] = p_k$ to $Dep^u(Q_n)$ set. In this case, the new non measurable place $p_k = [t_n, t_m]$ must contain one token in the initial marking.
2. Let t_i and t_j any two consecutive transitions in w_n
3. If there exist not a dependency $[t_i, t_j]$ in $Dep(Q_n)$ then
   (a) If there exists a NDep $[t_i, t_k] = p_x$ and also there exists a NDep $[t_k, t_j] = p_y$ in $Dep(Q_n)$ then form the NDep $[t_i, t_j]$ merging the non measurable places p_x and p_y.
   (b) If there exists a NDep $[t_i, t_k] = p_x$ in $Dep(Q_n)$ then add the NDep $[t_i, t_j]$ using the already computed non measurable place p_x such that $[t_i, t_j] = p_x$. (Identification step 4.2 case 1).
   (c) If there exists a NDep $[t_k, t_j] = p_y$ in $Dep(Q_n)$ then add the NDep $[t_i, t_j]$ using the already computed non measurable place p_y such that $[t_i, t_j] = p_y$. (Identification step 4.2 case 2).
   (d) else add the NDep $[t_i, t_j]$ to $Dep(Q_n)$. (Identification step 4.1)

A key feature of a non measurable place $p_k = [t_i, t_j]$ belonging to Class A is that its input and output transitions t_i and t_j belong to the same t-component. The m-word $w_n = \cdots t_i t_j \cdots$ in which t_i and t_j appear consecutively is also a t-semiflow of the system. The subclasses of non measurable places in Class A are the Class A.i in which p_k forms a single NDep implying that t_i and t_j belong to the same t-semiflow and the Class A.ii. in which p_k forms a complex NDep implying that t_i or t_j belongs to at least another t-semiflow and p_k is a shared non measurable place by more than one t-component. The procedure to compute this class of non measurable places consists in constrain the firing of any two consecutive transitions in the current computed m-word w_n to the order in which they are computed in w_n forming the t-component associated with this m-word. The non measurable places of Class A.i are computed using a new non measurable place while the non measurable places of the Class A.ii are computed using an already computed non measurable place adding input or output arcs to p_k.

◊ Dealing with concurrent transitions

Parallel activities are those that can occur independently during the execution of a cycle in a DES. Concurrence of a DES can be easily expressed in terms of Petri nets. In general two transitions are said to be concurrent if one transition may fire before, after or in parallel with other transition, this occurrence is considered in a determined t-component. The transitions t_1 and t_3 of the IPN depicted on figure 4.16 are an example of concurrent transitions.

Case 2: there exists a NDep $[t_k, t_j] = p_y$
such that t_k *does not belong to* w_n *then the NDep* $[t_i, t_j]$ *can be computed using an already computed place adding an arc from* p_x *to* t_j *or adding an arc from* t_i *to* p_y *for case 1 or case 2 respectively, forming the NDep* $[t_i, t_j] = p_x$ *or* $[t_i, t_j] = p_y$.

Proposition 4.6 *Let* Q *be a system model. A non measurable place of Class A.ii can be inferred using the identification step 4.2.*

Proof. A non measurable place of Class A.ii fulfills next conditions 1) it forms a complex NDep and 2) its input and output transitions belong to the same m-word which is also a t-semiflow of Q. Using identification step 4.2 a non measurable place is updated when two transitions t_i and t_j are consecutive transitions in the current m-word w_n, such that there exists a NDep $[t_i, t_k] = p_k$ or a NDep $[t_k, t_j] = p_y$ in Q_i and t_k does not belong to w_n. This two cases are next summarized:
case 1: If the NDep $[t_i, t_k] = p_x$ exists in Q_i then an arc is added from the place $p_x = [t_i, t_k]$ to the transition t_j to compute the NDep $[t_i, t_j] = p_x$, the place p_x forms a decision between t_j and t_k, since t_k does not belong to w_n then it is possible that Q_n generates the m-words: w_n and the m-word in which t_i and t_k were consecutive transitions.
or
case 2: If the NDep $p_y = [t_k, t_j]$ exists in Q_i then an arc is added from the transition t_i to the place $p_y = [t_k, t_j]$ to compute the NDep $[t_i, t_j] = p_y$, the place p_y forms an attribution between the transitions t_i and t_k, since t_k does not belong to w_n then it is possible that Q_n generates the m-words: w_n and the m-word in which t_k and t_j were consecutive transitions.
The fact that there exist the NDep $[t_i, t_k]$ or $[t_j, t_k]$ in Q_i such that t_k does not belong to the current m-word w_n in which t_i and t_j are consecutive transitions, implies that there exist another m-word in which t_i and t_k or t_k and t_j are consecutive transitions fulfilling the condition that t_i or t_j are shared transitions of at least two t-semiflows while the updated non measurable place p_x or p_y is a shared place by their underlying t-components. Hence a non measurable place of Class A.ii. is inferred using identification step 4.2 as new NDep $[t_i, t_j]$ are computed adding input or output arcs to an already computed non measurable place which is an output or an input place of t_i or t_j. ■

The underlaying procedure of identification step 4.2 consists in compute a NDep between any two consecutive transitions using an already computed non measurable place: If there exists not a dependency $[t_i, t_j]$ between any two consecutive transitions in the current m-word w_n but t_i has an output non measurable place p_x or t_j has and input non measurable place p_y i.e. there exists the NDep $[t_i, t_k] = p_x$ or the NDep $[t_k, t_j] = p_y$ in the computed model Q_i, then the NDep $[t_i, t_j]$ need to be computed using p_x or p_y. With this new computed NDep the place p_x or the place p_y is approached to an actual place of Q and the term $|Dep^u(Q) - Dep^u(Qi)|$ of the identification error equation is reduced.
The next algorithm presents how to infer a single or a complex NDep belonging to class A.i and A.ii respectively preserving the firing order of the transitions in the current computed m-word w_n.

Algorithm 4.1 *Computing NDep to constrain the firing order of the transitions in the current m-word w_n. Inference non measurable places belonging to class A.i and A.ii.*

Input: The $Dep(Q_n) = Dep(Q_{n-1})$ set and the current computed m-word $w_n = t_m \cdots t_n$.
Output: an updated $Dep(Q_n)$ set

---

1. If there exists not a MDep $[t_n, t_m]$ in $Dep(Q_n)$ then
   (a) add a NDep $[t_n, t_m] = p_k$ to $Dep^u(Q_n)$ set. In this case, the new non measurable place $p_k = [t_n, t_m]$ must contain one token in the initial marking.
2. Let t_i and t_j any two consecutive transitions in w_n
3. If there exist not a dependency $[t_i, t_j]$ in $Dep(Q_n)$ then
   (a) If there exists a NDep $[t_i, t_k] = p_x$ and also there exists a NDep $[t_k, t_j] = p_y$ in $Dep(Q_n)$ then form the NDep $[t_i, t_j]$ merging the non measurable places p_x and p_y.
   (b) If there exists a NDep $[t_i, t_k] = p_x$ in $Dep(Q_n)$ then add the NDep $[t_i, t_j]$ using the already computed non measurable place p_x such that $[t_i, t_j] = p_x$. (Identification step 4.2 case 1).
   (c) If there exists a NDep $[t_k, t_j] = p_y$ in $Dep(Q_n)$ then add the NDep $[t_i, t_j]$ using the already computed non measurable place p_y such that $[t_i, t_j] = p_y$. (Identification step 4.2 case 2).
   (d) else add the NDep $[t_i, t_j]$ to $Dep(Q_n)$. (Identification step 4.1)

A key feature of a non measurable place $p_k = [t_i, t_j]$ belonging to Class A is that its input and output transitions t_i and t_j belong to the same t-component. The m-word $w_n = \cdots t_i t_j \cdots$ in which t_i and t_j appear consecutively is also a t-semiflow of the system. The subclasses of non measurable places in Class A are the Class A.i in which p_k forms a single NDep implying that t_i and t_j belong to the same t-semiflow and the Class A.ii. in which p_k forms a complex NDep implying that t_i or t_j belongs to at least another t-semiflow and p_k is a shared non measurable place by more than one t-component. The procedure to compute this class of non measurable places consists in constrain the firing of any two consecutive transitions in the current computed m-word w_n to the order in which they are computed in w_n forming the t-component associated with this m-word. The non measurable places of Class A.i are computed using a new non measurable place while the non measurable places of the Class A.ii are computed using an already computed non measurable place adding input or output arcs to p_k.

◊ Dealing with concurrent transitions

Parallel activities are those that can occur independently during the execution of a cycle in a DES. Concurrence of a DES can be easily expressed in terms of Petri nets. In general two transitions are said to be concurrent if one transition may fire before, after or in parallel with other transition, this occurrence is considered in a determined t-component. The transitions t_1 and t_3 of the IPN depicted on figure 4.16 are an example of concurrent transitions.

It is possible that in any m-word $w_i = \cdots t_i t_j \cdots$ two concurrent transitions t_i and t_j occurs consecutively, for example in the m-word $w_1 = t_1 t_3 t_2 t_4 t_5$ of the same IPN t_1 and t_3 occurs consecutively.
At the beginning of the identification procedure it is not possible to know if any two consecutive transitions are concurrent transitions or not, then if there exists not a dependency between any two consecutive transitions t_i and t_j in the current m-word $w_n = \cdots t_i t_j \cdots$ and there exists not evidence that t_j has occurred before t_i in another m-word, then to constrain the firing of t_i before t_j it is computed a NDep $p_k = [t_i, t_j]$ as stated in identification step 4.1. If t_i and t_j are concurrent transitions then $p_k = [t_i, t_j]$ is a wrong NDep, i.e. this NDep does not belong to the system model to be identified. This wrong NDep will be removed when another m-word $w_m = \cdots t_j \cdots t_i \cdots$ in which t_j occurs before t_i is computed.
Next it is presented the definition of concurrent transitions, the needed properties to detect such transitions and the procedures to update a previous model when concurrent transitions are detected.

Definition 4.10 *Let Q be an IPN and X_k be a t-component of Q, two transitions t_p and t_q of X_k are concurrent transitions if there exists two transition sequences $\sigma_i \neq \sigma_j \in \pounds(X_k)$, such that $\sigma_i = \cdots t_p t_q \cdots$ and $\sigma_j = \cdots t_q t_p \cdots$. If there exists two transition sequences $\sigma_x = \cdots t_p \cdots t_q \cdots$ and $\sigma_y = \cdots t_p \cdots t_q \cdots$ t_p and t_q are considered pseudoconcurrent.*

Since the two sequences $\sigma_i \neq \sigma_j$ of previous definition are sequences generated by the same t-component they fulfill that $\overrightarrow{\sigma_i} = \overrightarrow{\sigma_j}$ (i.e. they have the same characteristic vector).
The set of all concurrent transitions in an IPN Q will be denoted as $Concurrent(Q)$, two concurrent transitions t_i and t_j are denoted as $c[t_i, t_j]$ which is equivalent to $c[t_j, t_i]$.
Let t_i and t_j any two consecutive transitions in the current m-word w_n and $\mathcal{T}$ be the set of all computed m-words. To detect if t_i and t_j are concurrent transitions or not, it is searched if there exists a m-word $w_k \neq w_n$ such that $\overrightarrow{w_k} = \overrightarrow{w_n}$ in which t_j occurs before t_i. The existence of the m-word $w_k = \cdots t_j \cdots t_k \cdots$ implies that t_i and t_j are concurrent transitions (by definition 4.10). However the absence of $w_k = \cdots t_j \cdots t_k \cdots$ in $\mathcal{T}$ does not implies that t_i and t_j are not concurrent, it only indicates that it has not been already computed a m-word in which t_j occurs before t_i.

Example. Consider the system model Q depicted on figure 4.19, the transitions t_1 and t_3 are concurrent transitions, however when Q had only generated the m-words $w_1 = t_1 t_3 t_2 t_4 t_5$ and $w_2 = t_1 t_2 t_3 t_4 t_5$ it is not possible to determine that t_1 and t_3 are concurrent transitions since from the behavior observed of Q, t_1 occurs always before than t_3.

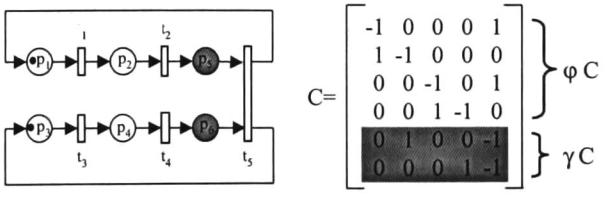


Figure 4.19: System model and its incidence matrix.

Proposition 4.7 *Let Q be a system model, $w_n = \cdots t_i t_j \cdots$ be the current computed m-word from the last output word produced by a t-component X_k of Q and $\mathcal{T}$ be the set of all computed m-words from Q. If there exist a m-word $w_k = \cdots t_j \cdots t_i \cdots$ in $\mathcal{T}$ such that $w_k \neq w_n$ and $\overrightarrow{w_k} = \overrightarrow{w_n}$, then t_i and t_j are concurrent transitions.*

Proof. By hypothesis if w_k exists in $\mathcal{T}$ implies that w_k and w_n belong to the same t-component, from definition 4.10 t_i and t_j are concurrent transitions due to Q had produced two m-words $w_k = \cdots t_j \cdots t_i \cdots$ and $w_n = \cdots t_i t_j \cdots$ in which the transitions t_i and t_j occurs in different order. ■

Corollary 4.1 *Let Q be a system model, Q_i be the computed model for Q and $w_n = \cdots t_i \cdots t_x \cdots$ be the current m-word computed from the last output word generated by the t-semiflow X_k of Q. If there exists a NDep $[t_x, t_i]$ in Q_i, then t_i and t_x are concurrent transitions.*

Proof. By identification step 4.1 a NDep is built to form a sequence between any two consecutive transitions of a computed m-word, hence if there exists a NDep $[t_x, t_i]$ in a t-component of the computed model Q_i implies that from the observed behavior of Q was already computed a m-word $w' = \cdots t_x t_i \cdots$ in which t_x occurs before t_i, by proposition 4.7 t_i and t_x are concurrent transitions since in the current m-word $w_n = \cdots t_i \cdots t_x \cdots$, t_i occurs before t_x. ■

Identification Step 4.3 *Let Q be an IPN, Q_i be the computed model for Q and t_i and t_x be any two concurrent transitions detected from the current computed m-word $w_n = \cdots t_i \cdots t_x \cdots$ If there exists a NDep $[t_x, t_i]$ in $Dep(Q_i)$, then the NDep $[t_x, t_i]$ is removed from $Dep(Q_i)$.*

Proposition 4.8 *Let Q be a system model and Q_i be the computed model for Q. If a NDep $[t_x, t_i]$ is removed using step 4.3, then the term $|Dep^u(Q_i) - Dep^u(Q)|$ of the error equation 4.1 is reduced.*

Proof. By corollary 4.1 t_i and t_x are concurrent transitions, hence they could not form any dependency and the NDep $[t_x, t_i]$ must to be removed from $Dep(Q_i)$ as stated in identification step 4.3. Since the NDep $[t_x, t_i]$ is detected that does not belong to Q the term $|Dep^u(Q_i) - Dep^u(Q)|$ of the identification error is reduced after remove a NDep. ■

Notice that after remove in the computed model a NDep $[t_i, t_j]$ could occur that t_i or t_j remain as a sink or as a source transition respectively such that $t_i^\bullet = \emptyset$ or $^\bullet t_j = \emptyset$. Assume that the NDep $[t_i, t_j] = p_k$ is removed from $Dep^u(Q_n)$: The transition t_i could become a sink transition when the only NDep formed with t_i is using the non measurable place p_k and t_i has not another output place. Similarly, t_j could become a source transition when the only NDep formed with t_j is using the non measurable place p_k and t_j has not another input place. Will be considered that a transition remain as a sink or as a source transition after the evaluation of all transitions in the current m-word w_n due to it is possible that in w_n t_i has another successor transition different to t_j or t_j has another predecessor transition different to t_i computing a new NDep.

Example. Consider the system model Q depicted on figure 4.20.a. The model Q_1 depicted on figure 4.20.b. is computed when the first m-word $w_1 = t_1 t_2 t_3 t_4 t_5$ is detected. A NDep $[t_2, t_3]$ is computed (among others) using identification step 4.1 because to t_2 and t_3 are consecutive transitions in w_1 and there exist not any dependency between these two transitions. Assume now that $w_2 = t_3 t_1 t_2 t_4 t_5$ is the next m-word computed. Then the transitions t_2 and t_3 are determined to be concurrent transitions because t_2

occurs before t_3 in w_1(proposition 4.7 and corollary 4.1). Since there exists a NDep $[t_2, t_3]$ in $Dep(Q_2)$, it is removed to allow the firing of t_3 before t_2 as stated in w_2 (identification step 4.3). After remove the NDep $[t_2, t_3]$ from $Dep(Q_2)$, t_2 remains as a sink transition and t_3 remains as a source transition; this is illustrated on figure 4.20.c.

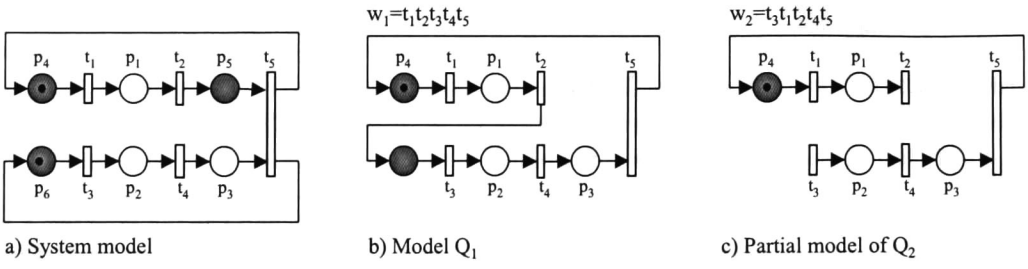


Figure 4.20: System model and its computed models.

However, the existence of sink or source transitions after removing a wrong NDep will depend on the computed m-words, i.e. in the observed behavior of the system, and also in how the non measurable places are allocated into the net. This fact is illustrated in the next two examples.

Example. Consider again the system model Q depicted on figure 4.20.a and its first model Q_1 depicted on figure 4.20.b computed when the m-word $w_1 = t_1t_2t_3t_4t_5$ was detected. If the second m-word computed is $w_2' = t_3t_4t_1t_2t_5$ (instead of $w_2 = t_3t_1t_2t_4t_5$ like in previous example) then the computed model Q_2' is equal to Q, because even the NDep $[t_2, t_3]$ is removed; the NDep $[t_2, t_5]$ is computed due to t_2 and t_5 are consecutive transitions in w_n and hence t_2 does not appear as a sink transition after evaluate w_2'. The NDep $[t_5, t_3]$ is computed since t_3 and t_5 are the first and the last transitions of w_n and as was stated previously they are considered as consecutive transitions, hence t_3 does not remain as a source transition.

Example. Consider now the system model Q depicted on figure 4.21 which has the same structure as the IPN depicted on figure 4.20, however they have a different interpretation since the number and allocation of the non measurable places in these two IPN is different. The model Q_1 depicted on figure 4.21.b is computed when the m-word $w_1 = t_1t_2t_3t_4t_5$ is detected. When the m-word $w_2 = t_3t_1t_2t_4t_5$ is computed it is removed the NDep $[t_2, t_3]$ since in this m-word t_3 occurs before t_2, notice that t_3 remains as a source transition as shown in the partial model Q_2 depicted on figure 4.21.c. In this example the transition t_2 does not remain as a sink transition as in the previous example.

In order to rebuilt the t-component associated to w_n, when a transition t_i remains as a sink or as a source transition it is needed to compute a NDep, this new NDep will be computed considering the predecessor or the successor of such a transition in the current m-word w_n, if t_i is a source or a sink transition respectively. This NDep is computed using the following identification step.

Identification Step 4.4 *Let w_n be the current computed m-word and Q_i be the computed model for a system model Q,*

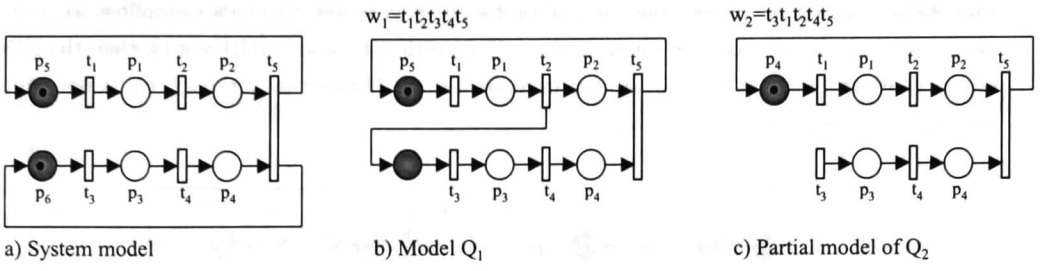


Figure 4.21: System model and its computed models.

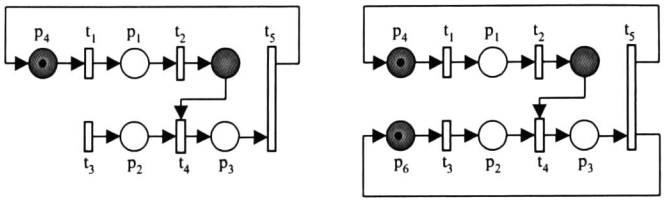
Case 1: If t_i remains as a sink transition in Q_i then a NDep $[t_i, t_x]$ is added to $Dep(Q_i)$ such that t_x is the successor transition of t_i in w_n. If t_i and t_x are concurrent transitions then t_x is now taken as the next successor of t_i in w_n.
Case 2: If t_j remains as a source transition in Q_i then a NDep $[t_y, t_j]$ is added to $Dep(Q_i)$ such that t_y is the predecessor transition of t_j in w_n. If t_y and t_j are concurrent transitions then t_y is now taken as the previous predecessor of t_j in w_n.

Proposition 4.9 *Let Q be a system model, Q_i be the computed model for Q, w_n be the current computed m-word and t_i be a sink transition or t_j be a source transition. If the identification step 4.4 is used to connect a sink or a source transition in Q_i, then it is fulfilled that the observed behavior of Q is preserved in Q_i.*

Proof. Case 1: Since t_i is a sink transition implies that $t_i^\bullet = \emptyset$, in order to compute an output non measurable place it is searched a successor t_x of t_i in w_n. Since t_i and t_x are consecutive transitions in w_n then it is possible to compute the NDep $[t_i, t_x]$, however if t_i and t_x are concurrent transitions this NDep cannot be computed because there exists another m-word in which t_x occurred before t_i; then the next successor of t_i in w_n it is chosen to form the NDep $[t_i, t_x]$. Hence when t_i is connected with a successor, the behavior of Q_i correspond to the observed behavior of Q since t_i has occurred before t_x in all the m-words computed.
Case 2: Since t_j is a source transition implies that ${}^\bullet t_j = \emptyset$, in order to compute an input non measurable place it is searched a predecessor t_y of t_j in w_n. Since t_y and t_j are consecutive transitions in w_n then it is possible to compute the NDep $[t_y, t_j]$, however if t_y and t_j are concurrent transitions this NDep cannot be computed because there exists another m-word in which t_j occurred before t_y, then the previous predecessor of t_j in w_n is chosen to form the NDep $[t_y, t_j]$. Hence when t_j is connected with a predecessor t_y the behavior of Q_i correspond to the observed behavior of Q since t_j had occurred after t_y in all the m-words computed. ■

Example. Consider the partial model Q_2 depicted on figure 4.20.c, after remove the NDep $[t_2, t_3]$ from $Dep(Q_2)$, t_2 remains as a sink transition and t_3 remains as a source transition. So t_2 need to be reconnected with its successor (identification step 4.4, case 1) in $w_2 = t_3t_1t_2t_4t_5$, the successor transition of t_2 in w_2 is t_4, as t_2 and t_4 have not been detected to be concurrent because neither in $w_1 = t_1t_2t_3t_4t_5$ nor in $w_2 = t_3t_1t_2t_4t_5$, t_4 occurs before t_2 (thus $[t_4, t_2]$ does not belong to $Concurrent(Q_2)$), then the NDep $[t_2, t_4]$ could be added to $Dep(Q_2)$ as presented in figure 4.22.a. Now the transition t_3 need to be reconnected with its predecessor (identification step 4.4 case 2) in $w_2 = t_3t_1t_2t_4t_5$, since t_3 is the first transition in w_2

then its predecessor is the last transition of w_2 due to w_2 is assumed to be a t-semiflow, as t_5 and t_3 have not been detected to be concurrent transitions (proposition 4.7 and corollary 4.1), then the NDep $[t_5, t_3]$ could be added to $Dep(Q_2)$ as presented in figure 4.22.b. Notice that although Q_2 is not equal to Q it can generate the m-words w_1 and w_2.



a) Partial model Q_2 b) Model Q_2

Figure 4.22: Computed models.

The remaining sink or source transitions will be connected after the current m-word is evaluated by pairs of consecutive transitions detecting new NDep or detecting concurrent transitions.

Remark. Notice that if a new NDep $[t_x, t_y] = p_k$ *is computed such that* t_x *is the last transition of a m-word then* p_k *must be marked..* ■

So, in previous example when the NDep $[t_5, t_3] = p_6$ is computed (figure 4.22.b) it is added one token to the place p_6 because the transition t_3 cannot be fired in Q_2 if p_6 is not marked.

The next algorithm removes wrongly computed NDep constraining the firing order of any two consecutive concurrent transitions and reconnecting the transitions remaining as a sink or as a source transition.

Algorithm 4.2 *Procedure to update a wrong single NDep*

Input: The $Dep(Q_n) = Dep(Q_{n-1})$ set, $Concurrent(Q_n) = Concurrent(Q_{n-1})$ and the current computed m-word $w_n = t_m \cdots t_n$
Output: The updated set $Dep(Q_n)$

---

1. Let t_i and t_j be any two consecutive transitions in $w_n = t_m \cdots t_n$

   (a) If there exist a NDep $[t_x, t_i]$ or $[t_y, t_j]$ (where $t_x \neq t_n$ and $t_y \neq t_n$) such that t_x or t_y occurs after t_i or t_j in the m-word $w_n = t_m \cdots t_n$ then remove $[t_x, t_i]$ and/or $[t_y, t_j]$ from $Dep^u(Q_n)$ and add it to $Concurrent(Q_n)$. (Corollary 4.1, identification step 4.3).

   (b) If $[t_i, t_j] \notin Dep(Q_n)$ and $[t_j, t_i] \notin Concurrent(Q_n)$ then

      i. If there exist a NDep $[t_j, t_i] \in Dep^u(Q_n)$ then remove $[t_j, t_i]$ from $Dep^u(Q_n)$ and add it to $Concurrent(Q_n)$. (Corollary 4.1, identification step 4.3).

      ii. If a m-word $w' = \cdots t_j \cdots t_i \cdots$ in which t_j occurred before t_i has been already computed, then add $[t_j, t_i]$ to $Concurrent(Q_n)$. (Proposition 4.7).

iii. If there exists not a MDep $[t_i, t_j]$ in $Dep(Q_n)$ then add the NDep $[t_i, t_j]$ to $Dep(Q_n)$. (Identification step 4.1).

2. After remove a wrong NDep any transition t_a could remain as a source or as a sink transition (Identification step 4.4)

   (a) If t_a remains as a *source* transition then add a NDep $[t_b, t_a]$ to $Dep^u(Q_n)$ such that t_b precedes t_a in the m-word w_n. If t_a is the first transition of $w_n = t_m \cdots t_n$ (i.e. $t_a = t_m$) then t_b will be the last transition of $w_n = t_m \cdots t_n$.

   (b) If t_a remains as a *sink* transition then add a NDep $[t_a, t_b]$ to $Dep^u(Q_n)$ such that t_b follows t_a in the m-word w_n. If t_a is the last transition of $w_n = t_m \cdots t_n$ (i.e. $t_a = t_n$) then t_b will be the first transition of $w_n = t_m \cdots t_n$ ($t_b = t_m$).

   (c) To form the NDep $[t_b, t_a]$ or $[t_a, t_b]$ we use the step 1 where $t_i = t_b$ and $t_j = t_a$ if t_a is a source transition, else in the case that t_a be a sink transition then $t_i = t_a$ and $t_j = t_b$. If $[t_i, t_j]$ belongs to $concurrent(Q_n)$ then repeat the step 2 such that $t_b = {}^\bullet({}^\bullet t_b)$ if t_a is a source transition, or $t_b = (t_b^\bullet)^\bullet$ if t_a is a sink transition until $[t_a, t_b]$ does not belong to $Concurrent(Q_n)$.

Example. Consider the system model Q depicted on figure 4.23.

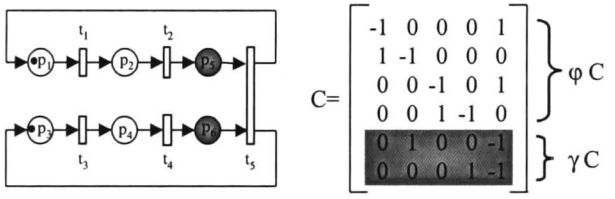


Figure 4.23: System model and its incidence matrix.

Let $w_1 = t_1 t_3 t_2 t_4 t_5$ be the first m-word computed using proposition 4.3. The measurable part of Q computed when w_1 is detected is depicted on figure 4.24.a. To preserve the observed behavior of the system the NDep $[t_1, t_3]$, $[t_3, t_2]$, $[t_2, t_4]$ and $[t_4, t_5]$ are computed according to identification step 4.1. This is illustrated in step 1.b.iii of the algorithm 4.2; the computed model Q_1 is depicted on figure 4.24.b.

Consider $w_2 = t_3 t_4 t_1 t_2 t_5$ the next m-word computed, the updating of the computed model is as follows: Evaluating the consecutive transitions t_3 and t_4; since there exist the NDep $[t_1, t_3]$ and $[t_2, t_4]$ in the previous computed model Q_1 and t_3 occurs before t_1 and t_4 occurs before t_2 in the m-word w_2, then t_1 and t_3 and t_2 and t_4 are concurrent transitions; hence the NDep $[t_1, t_3]$ and $[t_2, t_4]$ must be removed from $Dep(Q_2)$ (corollary 4.1, identification step 4.3) and added to $Concurrent(Q_2)$; this procedure is described in step 1.a. The NDep $[t_3, t_4]$ is not added to the new model Q_2 since there exists the MDep $[t_3, t_4]$ in $Dep(Q_2)$ (step 1.b.iii). The following consecutive transitions in w_2 are t_4 and t_1 but they are concurrent transitions because in w_1 t_1 occurs before t_4, this verification is made in step 1.b.ii (based on

proposition 4.7) and then the pair $[t_1, t_4]$ is added to $Concurrent(Q_2)$. Since there exist not dependencies constraining the firing of t_2 before t_1 and in w_1 t_1 occurs before t_2, where t_1 and t_2 are the following consecutive transitions in w_2, then they are not concurrent transitions however the NDep $[t_1, t_2]$ is not added to $Dep(Q_2)$ because there exists the MDep $[t_1, t_2]$ in $Dep(Q_2)$ (step 1.*b.iii*). The next computation concerns to the NDep $[t_2, t_5]$ since t_2 and t_5 are consecutive transitions in w_2 and they are not concurrent transitions; this procedure is specified in step 1.*b.iii* of the previous algorithm.

The computed model after evaluate w_2 is depicted on figure 4.24.c. Notice that this model describes the observed behavior of the system; however in order to compute a model for the system Q, it is necessary to compute from future measurements, a m-word in which t_2 occurs before t_3 for example the m-word $w_3 = t_1t_2t_3t_4t_5$ induces the remotion of the wrong NDep $[t_3, t_2]$.

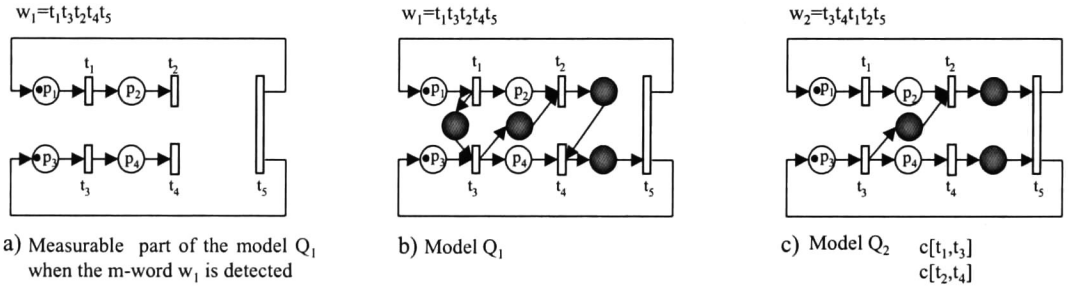


Figure 4.24: Iterations of the algorithm 4.2.

4.4.6 Inference of non measurable places of class B

The non measurable places of Class B are those in which their input and output transitions belong to different m-words, hence the procedures to compute these places consist in concatenate the current and the previous m-words forming a new t-semiflow in the computed model. Only the places belonging to the class B.i are computed directly using the concatenating procedure since they form a single NDep. Since the places of classes B.ii.a and B.ii.b form complex NDep they will be inferred when new information of the system is detected. However when the computed m-word is an actual t-semiflow of the system, the procedure proposed to update the non measurable place related with the previous and current m-word is merging two non measurable places. The places computed with the last procedure are those belonging to Class B.ii.b.

The non measurable places of Class A are inferred computing NDep between consecutive transitions t_i, t_j in the computed m-word $w_n = \cdots t_i t_j \cdots$ Now the procedures to infer the non measurable places of Class B will consist in compute NDep between the last and the first transitions of the previous and the current m-word. Let $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the previous and the current m-word respectively, now it is searched the existence of the NDep $[t_j, t_m]$ in the computed model Q_i in order to rebuilt the t-component of the system model Q.

To rebuilt the t-semiflows of a system it is defined the set $\mathcal{W} = \{W_k\}$ containing all computed t-semiflows W_k; where each W_k is a concatenation of selected computed m-words. Notice that each W_k is a t-semiflow (not

necessarily elemental) at least in the φC matrix since every m-word is a t-semiflow of this matrix. Also it is defined a sequence of m-words $\mathcal{T}$ composed by all detected m-words preserving the order in which they are computed. A computed transition will be labeled as t_F if it is detected to be the last transition of an actual t-semiflow of a system model Q.

Consider the next algorithm to relate the previous model with the current computed m-word.

Algorithm 4.3 *Initialization*

Input: The previous and the current computed m-words $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m ... t_n$ respectively and the set of computed t-semiflows $\mathcal{W}$.
Output: None

---

1. If w_n has been already computed
   (a) If w_n is the first m-word of any W_i then
      i. If t_j is not marked as t_F then mark t_j as t_F which implies that w_{n-1} is the last t–semiflow (m-word) of some W_i in which w_{n-1} belongs
      ii. $W_{k+1} = w_n$.
      iii. Restart the algorithm with a new m-word
   (b) If there exists a NDep $[t_j, t_m]$ then
      i. $W_k = W_k w_n$.
      ii. Restart the algorithm with a new m-word
2. If t_j is marked as t_F and w_n is computed by first time, then remove the NDep $[t_n, t_m]$ and form the NDep $[t_n, t_m] = {}^\bullet t_i$, where t_i is the first transition of W_j where w_{n-1} belongs. $W_{k+1} = w_n$.
3. If $w_n = t_m \cdots t_n$ is a new m-word and there exists a NDep $[t_j, t_m]$ and $t_j \notin w_n$ then W_k=$W_k w_n$.

Inference of non measurable places of Class B.i

A non measurable place $p_k = [t_i, t_j]$ of Class B.i form a single NDep and its input and output transition belong to different m-words $w_a = \cdots t_i$ and $w_b = t_j \cdots$ respectively such that w_a and w_b are m-words of a same t-semiflow in Q, i.e. there exists a t-semiflow X_k in Q such that $X_k = \cdots w_a w_b \cdots$, notice then that t_i and t_j are consecutive transitions in the transition sequence generated by X_k.
The IPN depicted on figure 4.25 has only one t-semiflow X_i, the m-word decomposition of X_i is $w_1 = t_1 t_2$, $w_2 = t_3 t_4$ and $w_3 = t_5 t_6$, an example of a place belonging to this class is the place $p_x = [t_6, t_1]$ since t_6 belongs to w_3 and t_1 belongs to w_1; indeed all the non measurable places in this IPN are places belonging to the class B.i.
The procedure to compute the non measurable places of Class B.i from the detected information of the system is concatenating the previous and the current m-word forming a t-semiflow in the computed model since it is assumed that the t-semiflows of the systems have a m-word decomposition.

Identification Step 4.5 *Let Q be a system model, Q_i be the computed model for Q and $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the previous and the current computed m-words. If there exists not a NDep $[t_j, t_m]$ in $Dep^u(Q_i)$ then add the non measurable place $p_k = [t_j, t_m]$ to Dep Q_i and concatenate w_n to the t-semiflow W_i which w_{n-1} belongs such that $W_i = W_i w_n$.*

Proposition 4.10 *Let Q be a system model. A non measurable place of Class B.i, can be computed using identification step 4.5.*

Proof. Let p_k be a non measurable place of Class B.i, then p_k belongs to a t-component X_q such that its underlying t-semiflow have a m-word decomposition $\cdots w_a w_b \cdots$ and p_k only form a single NDep $[t_j, t_m]$ such that t_j is the last transition of a m-word different to the m-word in which t_m is the first transition. Let w_a and w_b be the previous and the current m-words computed $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ respectively, then when w_{n-1} and w_n are concatenated as stated in identification step 4.5 a non measurable of class B.i. is computed. ■

The computed t-semiflow W_i using identification step 4.5 will be an actual t-semiflow of Q if w_n is the last m-word in the m-word decomposition of a t-semiflow X_j of Q. Then the only non measurable places computed correctly are those places forming dependencies between m-words belonging to an actual t-semiflow of the system, an example of this class of places are the places p_x, p_y and p_z of the PN depicted on figure 4.25.

The computed t-semiflows W_i of a model Q_i are approaching to the actual t-semiflows of a system Q as new m-words are detected.

Example. Consider the system model Q depicted on figure 4.25.

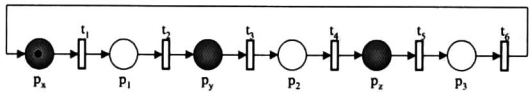


Figure 4.25: System model Q.

When the transitions t_1 and t_2 are fired it is detected a m-word $w_1 = t_1 t_2$ using the procedure to constrain the firing order of the transitions computed to the order stated in w_1 illustrated in algorithm 4.1 it is computed the model Q_1, depicted on figure 4.26.

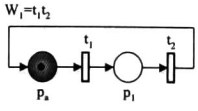


Figure 4.26: Computed model Q_1 for the detected m-word $w_1 = t_1 t_2$.

Notice that the NDep $p_a = [t_2, t_1]$ does not exists in Q however the model Q_1 describes the observed behavior of Q. This wrong NDep is computed since the computed m-word w_1 is not a t-semiflow of Q. The t-semiflow computed associated with w_1 is $W_1 = w_1$

Assume that the transitions t_3 and t_4 are fired in Q, then the computed m-word is $w_2 = t_3t_4$. The computed t-component associated to this m-word is depicted on figure 4.27.

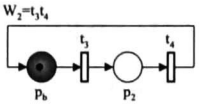


Figure 4.27: Computed t-component for w_2.

Using the identification step 4.5 to concatenate the previous and the current computed m-word, the correctly computed non measurable place is the place p_y since its input and output transitions t_2 and t_3 respectively are the last and the first transitions of different m-words of a same m-word decomposition of a t-semiflow. Since w_1 and w_2 were concatenated then the computed t-component in Q_2 is $W_1 = w_1w_2$. The model for the observed behavior is the model Q_2 depicted on figure 4.28.

When the m-word $w_3 = t_5t_6$ is detected the computed t-semiflow of the computed model Q_3 will be $W_1 = w_1w_2w_3$ at this moment an actual t-semiflow of Q is computed.

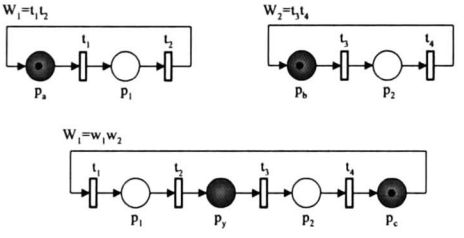


Figure 4.28: Computed model Q_2 for the observed output words $w_1 = t_1t_2$ and $w_2 = t_3t_4$.

Then, the first approach to compute an actual t-semiflow of a system model, is concatenating the previous and the current computed m-words $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ respectively, in the computed model Q_n when there exists not a NDep $[t_j, t_m]$ in $Dep^u(Q_n)$ and t_m has not an input measurable place. Using identification step 4.5 the non measurable places p_y and p_x that relates the transitions t_j and t_m and the transitions t_n and t_i respectively forming a new t-semiflow in Q_n are computed.

It is important consider the following:

If a non measurable place forms a complex NDep between transitions of different m-words then to infer it, is needed that all the m-words of the t-components in which p_k belongs have been computed, this kind of places belong to Class B.ii.a.

Also it is possible that an actual t-semiflow be concatenated wrongly in the cases where w_{n-1} or w_n be an actual t-semiflow of a system model Q, in this case the wrongly computed NDep will be updated when

it is observed that w_{n-1} and w_n occur in a non consecutively form in a future evolution of Q, then the place forming the NDep $[t_j, t_m]$ between w_{n-1} and w_n will be computed merging the places $t_j^\bullet$ and ${}^\bullet t_m$ computing hence the non measurable places of Class B.ii.b.

Inference of non measurable places of Class B.ii.a

The non measurable places of Class B.ii.a form a complex NDep $p_k = [t_i, t_j]$ such that t_i and t_j belong to different m-words $w_a = \cdots t_i$ and $w_b = t_j \cdots$ respectively, and w_a and w_b are m-words of a same m-word decomposition of a t-semiflow in Q, i.e. there exists a t-semiflow X_k in Q such that a sequence generated by a X_k is $X_k = \cdots w_a w_b \cdots$, notice then that t_i and t_j are consecutive transitions in X_k. However a difference of the Class B.i, w_a or w_b belongs also to another t-semiflow X_j.

Example. Consider again the system model depicted on figure 4.17 but now assume that the place p_1 is also a non measurable place, such system structure with the new interpretation is depicted on figure 4.29.

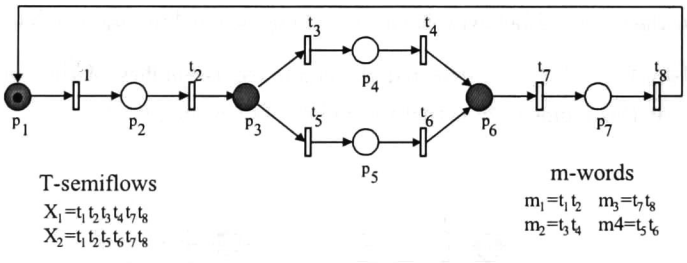


Figure 4.29: System model.

For this new system model Q it could be computed four $m-$words: $w_1 = t_1t_2$, $w_2 = t_3t_4$, $w_3 = t_7t_8$ and $w_4 = t_5t_6$. The place p_3 belongs to the class B.ii.a since it forms a complex NDep $p_3 = [t_2, t_3] = [t_2, t_5]$ and the input and output transitions of p_k belong to different m-words: in $[t_2, t_3]$ t_2 belongs to w_1 and t_3 belong to w_2, while in $[t_2, t_5]$ t_2 belongs to w_1 and t_5 belongs to w_4. The place p_6 belongs also to the class B.ii.a, however the place p_1 belongs to the class B.i due to it forms a single NDep between transitions of different m-words of a same t-semiflow. Notice that the m-words w_1 and w_3 belong to both t-semiflows of Q : $X_1 = t_1t_2t_3t_4t_7t_8$ and $X_2 = t_1t_2t_5t_6t_7t_8$

Notice that the transitions t_2 and t_3, and t_2 and t_5 are not shared transitions in any m-word, thus the identification step 4.2 cannot be used to compute the non measurable place p_3. Also to compute the non measurable place p_6 using identification step 4.2, the transitions t_4 and t_7, and t_6 and t_7 must to be shared transitions in any m-word and it is not the case.
The NDep belonging to Class B.ii.a can be computed using next identification step.

Identification Step 4.6 *Let Q be a system model, Q_i be the computed model for Q, $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the previous and the current computed m-words, $W_j \in \mathcal{W}$ be the computed t-semiflow in where w_{n-1} belongs and $W_k \in \mathcal{W}$ be the current computed t-semiflow.*

Case 1: Selection. If w_{n-1} belongs to W_k and also to another $W_j \neq W_k$ and w_n is computed by first time then, based on identification step 4.2 case 1, to form the NDep $[t_j, t_m]$ it is needed to add an arc from p_x to t_m and to form the NDep $[t_n, t_y]$, where t_y is the first transition of W_j, it suffices to add an arc from t_n to ${}^\bullet t_y$. The current compute t-semiflow is updated as $W_k = W_k w_n$.
Case 2: Attribution. If w_n belongs to another $W_j \neq W_k$ and w_{n-1} belongs only to W_k then, there must exist a NDep $[t_k, t_m] = p_y$, remove the arc from t_j to ${}^\bullet t_y$, where t_y is the first transition of W_k, to form the NDep $[t_j, t_m]$ based on identification step 4.2 case 2 it is only need to add an arc from t_j to p_y. The current compute t-semiflow is updated as $W_k = W_k w_n$.
Notice that in both cases t_k belongs to another m-word $w_a \neq w_n$

Proposition 4.11 *Let Q be a system model. Using identification step 4.6, a non measurable place of Class B.ii.a can be inferred.*

Proof. Case 1: Selection. Since w_{n-1} belongs besides to W_k to another $W_j \neq W_k$ implies that there exists a NDep $[t_j, t_k] = p_x$, notice that t_j and t_m are consecutive transitions in the t-semiflow in which w_{n-1} and w_n belongs, hence based on proposition 4.2 case 1, the computation of the NDep $[t_j, t_m] = p_x$ using the already computed non measurable place $p_x = [t_j, t_k]$ forms a decision place between the transitions t_k and t_m. Since also is computed the NDep $[t_n, t_y]$ then a t-semiflow is computed in Q_n, this t-semiflow is $W_k = W_k w_n$.
Case 2: Attribution. Since w_n belongs to another $W_j \neq W_k$ then there must exist a NDep $[t_k, t_m] = p_y$, notice that t_j and t_m are consecutive transitions in the t-semiflow in which w_{n-1} and w_n belongs, hence based on proposition 4.2 case 2, the computation of the NDep $[t_j, t_m] = p_y$ using the already computed non measurable place $p_x = [t_k, t_m]$ forms an attribution place between the transitions t_k and t_m. Since w_{n-1} belongs to W_k and w_n is concatenated with w_{n-1}, when the NDep $[t_j, t_m]$ is formed also is computed the NDep $[t_n, t_y]$ and a t-semiflow is computed in Q_n, this t-semiflow is $W_k = W_k w_n$.
Notice that in Q_n the t-semiflows W_k and W_j can be executed. Since the computed places p_x and p_y forms a complex NDep and t_j and t_m belongs to different m-words of the computed t-semiflow W_k then using identification step 4.6 the decision and the attribution non measurable places of Class B.ii.a can be inferred. ■

Example. Consider the system model depicted on figure 4.29.

Let $w_1 = t_1 t_2$, $w_2 = t_3 t_4$ and $w_3 = t_7 t_8$ be the computed m-words in the order $\mathcal{T} = w_1 w_2 w_3$. The computed model for this observed behavior was computed concatenating any two consecutive m-words (identification step 4.5), it is depicted on figure 4.30.

The t-semiflow computed is $W_1 = w_1 w_2 w_3$.

The next computed m-word is w_1, $\mathcal{T} = w_1 w_2 w_3 w_1$. Since w_1 is an already computed m-word and it is the first m-word of W_1 then $W_2 = w_1$ implying that the m-word w_1 could belong to another t-semiflow of the system. t_8 is marked as t_F implying that another m-word could not be concatenated with w_3. This is made using algorithm 4.3.

The next m-word computed is $w_4 = t_5 t_6$, $\mathcal{T} = w_1 w_2 w_3 w_1 w_4$. The t-component associated with this m-word is computed using the algorithm 4.1, it is depicted on figure 4.31.

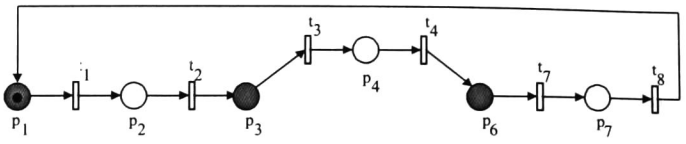


Figure 4.30: Computed model Q_1.

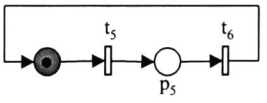


Figure 4.31: Computed t-component associated with $w_4 = t_5 t_6$.

In order to concatenate this component with the previous model is used the case 1 of the identification step 4.6. The computed model Q_2 is depicted on figure 4.32.

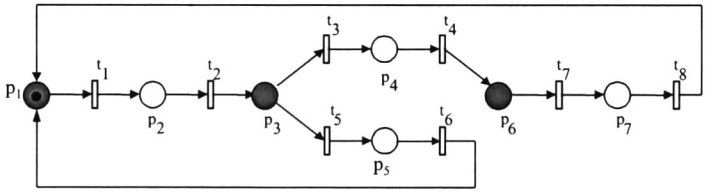


Figure 4.32: Computed model Q_2.

$W_2 = w_1 w_4$. Notice that W_2 is a t-semiflow in Q_2, and that Q_2 describes the observed behavior.

The next m-word computed is $w_3 = t_7 t_8$, $\mathcal{T} = w_1 w_2 w_3 w_1 w_4 w_3$. Since w_3 is an already computed m-word and it is not the first m-word of some t-semiflow W_x in Q_2 (verification using algorithm 4.3) then it suffices to concatenate w_3 with the previous m-word w_4. To do that is used case 2 of identification step 4.6. The resulting model Q_3 is the same as the system model depicted on figure 4.29 and $W_2 = w_1 w_4 w_3$.

Notice that the t-semiflows W_1 and W_2 of the computed model are the same t-semiflows of the system model Q, hence the computed model Q_3 describes the entire behavior of Q.

Inference of non measurable places of Class B.ii.b

A non measurable place p_k that belongs to Class B.ii.b, forms complex NDep such that each transition t_i and t_j of a NDep $[t_i, t_j] = p_k$ are the last and the first transitions of a t-semiflow of a system model Q. The inference of these non measurable places can be made using a procedure based on next proposition.

Proposition 4.12 *Let Q be an IPN and let $K_i = t_i \cdots t_j$, $K_j = t_k \cdots t_l$ and $K_m = t_m \cdots t_n$ be three t-semiflows of Q, if there exist two transition sequences $\sigma_i = K_i K_j K_m$ and $\sigma_j = K_i K_m$ in Q (σ_i and σ_j are also t-semiflows) then there exists a place $p_x = t_j^\bullet =^\bullet t_k =^\bullet t_m = t_l^\bullet$.*

Proof. If σ_i can be generated by Q then there exists two dependencies $[t_j, t_k] = p_i$, and $[t_l, t_m] = p_j$ hence $t_j^\bullet =^\bullet t_k = p_i$ and $t_l^\bullet =^\bullet t_m = p_j$. As σ_j also can be generated by Q then t_k and t_m are in conflict due to after the firing of t_j either t_k can be fired in the transition sequence σ_i or t_m can be fired in the transition sequence σ_j then there exist a decision place in Q such that ${}^\bullet t_k =^\bullet t_m$, since ${}^\bullet t_k = p_i$ and ${}^\bullet t_m = p_j$ then there exists a place $p_x = p_i \cup p_j$ such that $p_x = t_j^\bullet =^\bullet t_k =^\bullet t_m = t_l^\bullet$. ■

Example. Consider the system model Q depicted on figure 4.33. The transition sequences stated in the above proposition could be $\sigma_i = \overbrace{t_1 t_2}^{K_i} \overbrace{t_3 t_4}^{K_j} \overbrace{t_5 t_6}^{K_m}$ and $\sigma_j = \overbrace{t_1 t_2}^{K_i} \overbrace{t_5 t_6}^{K_m}$, notice that these two sequences can be generated by Q. Since $t_i = t_1$, $t_j = t_2$, $t_k = t_3$, $t_l = t_4$, $t_m = t_5$ and $t_n = t_6$ then the place p_x of Q fulfills previous proposition due to $p_x = t_2^\bullet =^\bullet t_3 =^\bullet t_5 = t_4^\bullet$

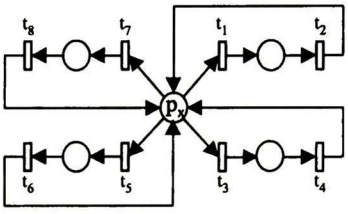


Figure 4.33: System model Q.

The places belonging to the class B.ii.b are computed when some non measurable places are merged. The selected places to be merged are those sequencing m-words, since as stated in the identification step 4.1, the other non measurable places are those belonging to the same m-word, hence they are computed correctly.

Identification Step 4.7 *Let Q_i be the computed model for a system model Q, $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the previous and the current computed m-words respectively. If there exists a $DSeq(t_j, t_m)$ in Q_i then the NDep $p_k = [t_j, t_m]$ is computed merging the places $t_j^\bullet$ and ${}^\bullet t_m$.*

Proposition 4.13 *Let Q be a system model. A non measurable place of Class B.ii.b can be inferred using identification step 4.7.*

Proof. If there exists a $DSeq(t_j, t_m)$ in Q_i implies that there exists a m-word sequence $W_p = w_{n-1} w_j w_n \cdots$. Since w_{n-1} and w_n are also consecutive m-words then there exists a m-word sequence $W_q = w_{n-1} w_n$. If W_p and W_q are the transition sequences stated in proposition 4.12 then there must exist a place $p_x = t_j^\bullet =^\bullet t_k =^\bullet t_m = t_l^\bullet$ in Q allowing that the sequences W_p and W_q can be generated. Let t_k and t_l be the first and the last transitions of w_j, from sequence W_p it is detected that $t_j^\bullet = [t_j, t_k]$ and ${}^\bullet t_m = [t_l, t_m]$, then merging the places $t_j^\bullet$ and ${}^\bullet t_m$ the non measurable place p_k is computed and indeed the NDep $[t_j, t_m]$.

If p_k belongs to Class B.ii.b its input and output transitions correspond to actual t-semiflows of Q then using identification step 4.7 the non measurable places of Class B.ii.b can be inferred. ■

The procedures for merging places computes a decision place allowing the firing of the m-words in the order as they are observed. The merged places are those allowing to form the NDep $[t_j, t_m]$ between the previous and the current computed m-words w_{n-1} and w_n respectively such that t_j is the last transition of w_{n-1} and t_m is the first transition of w_n.

Identification Step 4.8 *Let Q_i be the computed model after merge the non measurable places $t_j^\bullet$ and ${}^\bullet t_m$ as stated in previous propositions, $\mathcal{W}$ be the set of all computed t-semiflows and w_j be the t-semiflow formed when $t_j^\bullet$ and ${}^\bullet t_m$ of the sequence $w_{n-1}w_jw_n$ were merged. After merge $t_j^\bullet$ and ${}^\bullet t_m$ all the t-semiflows $W_i \in \mathcal{W}$ of Q_i which contain the transitions of w_j are updated as follows: $W_i = W_i/w_j$ (where W_i/w_j means that w_j is removed from W_i), $W_{k+1} = w_j$, where k is the subindex of the current computed t-semiflow and if there exists a t-semiflow W_x in Q_i in which w_{n-1} belongs but no w_n then $W_x = W_x w_n$.*

Proposition 4.14 *Let Q_i be the computed model after merge two places, using identification step 4.8 the t-semiflows of Q_i can be updated.*

Proof. Consider the m-word sequence $w_{n-1}w_jw_n$, notice that after merge the places $t_j^\bullet$ and ${}^\bullet t_m$ w_j becomes a t-semiflow in Q_i since the computed non measurable place p_x forms the NDep $[t_l, t_k]$ (by identification step 4.7), where t_k and t_l are the first and the last transitions of w_j, then w_j is a new computed t-semiflow in Q_i, hence $W_{k+1} = w_j$.
Let W_i be any t-semiflow in $\mathcal{W}$ containing w_j, as w_j is a new t-semiflow then W_i forms a t-semiflow without the transitions of w_j due to the computed place p_k forms the NDep $[t_j, t_m]$ (by identification step 4.7), hence each t-semiflow W_i containing w_j will be updated as W_i/w_j implying that w_j is removed from each W_i. Assume that W_x contains the m-word w_{n-1}, when the places $t_j^\bullet$ and ${}^\bullet t_m$ are merged the NDep $[t_j, t_m]$ is formed such that t_j is the last transition of w_{n-1} and t_m is the first transition of w_n (as stated in identification step 4.7), then also is formed the t-semiflow $W_x = W_x w_n$. Hence using identification step 4.8 the t-semiflows of the computed model Q_i are updated. ■

Corollary 4.2 *Let Q_i be the computed model after merge the places $t_j^\bullet$ and ${}^\bullet t_m$ as stated in identification step 4.7. Then the last transition t_l of w_j is detected as a final transition of some t-semiflow of the system model Q.*

Proof. Since t_l is the last transition of w_j then another m-word cannot be concatenated with w_j, due to Q can generate the transition sequence $w_{n-1}w_jw_n$ and $w_{n-1}w_n$ implying that t_l is the last transition of a m-word decomposition of an actual t-semiflow of Q. ■
The transitions detected as final transitions are labeled as t_F, implying that they are the last transitions of a t-semiflow in the system Q, this label is needed to avoid the concatenation of consecutive m-words forming a wrong NDep in the computed model Q_i.

Example. Consider the IPN Q depicted on figure 4.34. The m-words of Q are: $w_1 = t_1t_2$, $w_2 = t_3t_4$, $w_3 = t_5t_6$, $w_4 = t_7t_8$ and $w_5 = t_9t_{10}$.

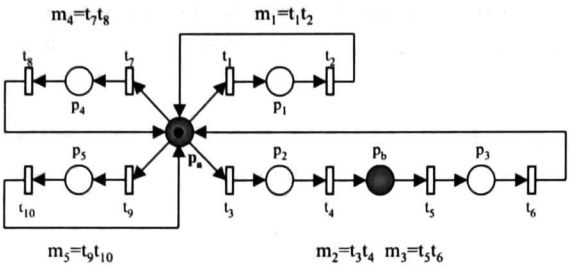


Figure 4.34: System model without shared transitions by any t-semiflow.

Assume that were computed the next sequence of m-words $\mathcal{T} = w_1w_2w_3w_4w_5$, then using the concatenating procedure (identification step 4.5), the model Q_i depicted on figure 4.35 is computed. The t-semiflow of Q_i is $W_1 = w_1w_2w_3w_4w_5$

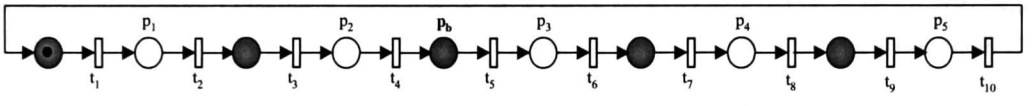


Figure 4.35: Computed model Q_i.

The fully identified non measurable place is the place p_b since it belong to the class B.i. All the NDep computed exists in the model, however the places forming these NDep does not belong to Q; these non measurable places will be updated in posterior observations of the system behavior.

Now consider the observation of $w_4 = t_7t_8$ as the current m-word $w_n = t_m \cdots t_n$, hence $w_5 = t_9t_{10}$ is the previous computed m-word $w_{n-1} = t_i \cdots t_j$, then $\mathcal{T} = w_1w_2w_3w_4w_5w_4$. $W_p = w_5w_1w_2w_3w_4$ and $W_q = w_5w_4$ Notice that in the computed model Q_i there exists a $DSeq(t_{10}, t_7) = \overbrace{t_9t_{10}\overbrace{t_1t_2\,t_3t_4\,t_5t_6}^{w_j}\,t_7t_8}^{w_{n-1}\ w_1\ w_2\ w_3\ w_n}$; in this case $w_j = w_1w_2w_3$, the transitions t_j, t_k, t_l and t_m are the following: $t_j = t_{10}$, $t_k = t_1$, $t_l = t_6$ and $t_m = t_7$

Merging the places $t_j^\bullet = t_{10}^\bullet$ and ${}^\bullet t_m = {}^\bullet t_7$, the model Q_j depicted on figure 4.36 is computed.

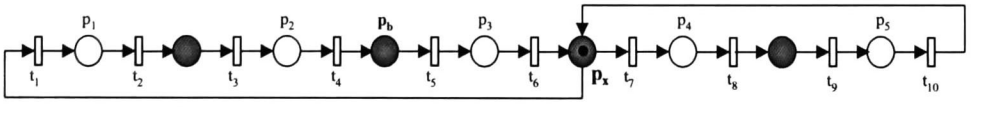


Figure 4.36: Computed model Q_j.

The place p_x fulfill the conditions of proposition 4.12, since $p_x = [t_{10}, t_7] = [t_6, t_7] = [t_6, t_1] = [t_{10}, t_1]$ then $p_x =^{\bullet} t_7 =^{\bullet} t_1 = t_{10}^{\bullet} = t_6^{\bullet}$ and hence $p_x =^{\bullet} t_m =^{\bullet} t_k = t_j^{\bullet} = t_l^{\bullet}$. Hence the NDep $[t_j, t_m] = [t_{10}, t_7]$ is computed.

The new t-semiflows computed are $W_2 = w_j = w_1 w_2 w_3$ and $W_1 = W_1 / w_j = w_4 w_5$. (Corollary 4.8).

The final transitions t_l and t_j (corollary 4.2) are t_6 and t_{10} respectively since t_6 is the last transition of $w_j = w_1 w_2 w_3$ and t_{10} is the last transition of W_1.

The next identification step states how to compute the NDep $[t_j, t_m]$ to concatenate the previous and the current m-words w_{n-1} and w_n respectively, when w_n is a new m-word and t_j is marked as t_F.

Identification Step 4.9 *Let Q_i be the computed model for a system model Q, $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the previous and the current computed m-words such that w_n is a new m-word. If t_j is labeled as t_F and w_n is a new computed m-word then form the NDep $[t_n, t_m] = t_j^{\bullet}$, where t_i is the first transition of W_j in which w_{n-1} belongs.*

Proposition 4.15 *Let Q be a system model and Q_i be the computed model for Q, then using identification step 4.9 a non measurable place of Class B.ii.b can be inferred.*

Proof. Since w_n is a new m-word then does not exists the transition sequence $DSeq(t_j, t_m)$ in Q and then the identification step 4.7 cannot be used to compute the NDep $[t_j, t_m]$ relating the previous model with the current computed m-word. Since t_j is labeled as t_F, then the current m-word cannot be concatenated with the previous model using identification step 4.5 because t_j is detected as the last transition of an actual t-semiflow of Q. Notice that when the NDep $[t_n, t_m] = t_j^{\bullet}$ is computed as stated in identification step 4.9, the NDep $[t_j, t_m]$ also is formed relating the current m-word w_n with the transitions of w_{n-1} in Q_i. Hence $t_j^{\bullet}$ forms the NDep between transitions of different m-words of different t-semiflows. Since w_n is added to Q_i as a new t-semiflow, then a new t-semiflow $W_{k+1} = w_n$ is computed in Q_i. ■

Example. Consider again the model system depicted on figure 4.34.

Let $\mathcal{T} = w_1 w_4 w_5$ be the sequence of m-words computed, the model Q_i is depicted on figure 4.37.

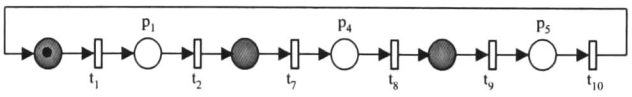


Figure 4.37: Computed model Q_i.

The computed t-semiflow of Q_i is $W_1 = w_1 w_4 w_5$.

Let $w_1 = t_1 t_2$ be the next m-word computed, $\mathcal{T} = w_1 w_4 w_5 w_1$. As w_1 is the first m-word of a t-semiflow of Q, then $W_2 = w_1$ and t_{10} is marked as t_F (corollary 4.2). And as there exists the NDep $[t_j, t_m] = [t_{10}, t_1]$, then there is nothing to update in Q_i. Algorithm 4.3.

Let $w_5 = t_9t_{10}$ be the next m-word computed, $\mathcal{T} = w_1w_4w_5w_1w_5$. Hence $w_1 = t_1t_2$ and $w_5 = t_9t_{10}$ are the previous and the current m-words w_{n-1} and w_n respectively. Since there exists a $DSeq(t_2, t_9) = \overbrace{\overbrace{t_1t_2}^{w_1}\overbrace{t_7t_8}^{w_4}\overbrace{t_9t_{10}}^{w_5}}^{w_{n-1}\ w_j\ w_n}$ in Q_i, then the NDep $[t_j, t_m] = [t_2, t_9]$ can be computed merging the places $t_2^\bullet$ and ${}^\bullet t_9$ as stated in identification step 4.7. The computed model Q_2 is depicted on figure 4.38.

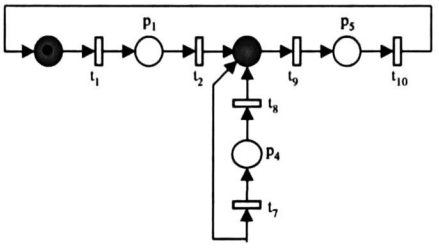


Figure 4.38: Computed model Q_i.

The t-semiflows of Q_2 are $W_1 = W_1/w_j = w_1w_5 = W_2 = W_2w_5$ and $W_3 = w_j = w_4$, $W_1 = W_2$ since in previous model $W_2 = w_1$. (Corollary 4.8). And t_7 is marked as t_F (corollary 4.2).

Let $w_2 = t_3t_4$ be the current computed m-word. Now $w_5 = w_{n-1}$ and $w_2 = w_n$ are the previous and the current computed m-words respectively. As w_2 is a new m-word and $t_j = t_{10}$ is marked as t_F then the NDep $[t_j, t_m] = [t_{10}, t_4]$ cannot be computed concatenating the previous and the current m-word using identification step 4.5; instead of this, it is used the proposition 4.9, such that $[t_4, t_3] = t_{10}^\bullet$. The updated model Q_i is depicted on figure 4.39. The new t-semiflow computed in Q_i is $W_4 = w_2$.

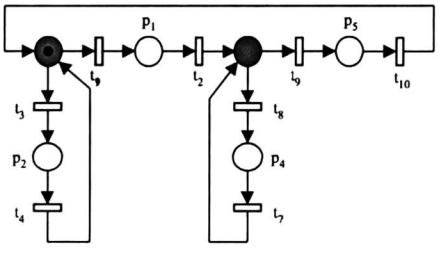


Figure 4.39: Computed model Q_i

Notice that the following m-word produced by the system must be $w_3 = t_5t_6$ since it belong to the same m-word decomposition as w_2. Let w_3 be the current computed m-word. The procedure to compute the NDep $[t_j, t_m]$ is concatenating w_3 with the transitions of w_2 in the computed model Q_i. The identification step 4.5 is used to concatenate w_2 with w_3 since w_3 is a new m-word and there exists not a $DSeq(t_j = t_4, t_m = t_5)$ and t_4 is not marked as t_F. The computed model Q_i is depicted on figure 4.40.

The updated t-semiflow in Q_i is $W_4 = W_4w_3$

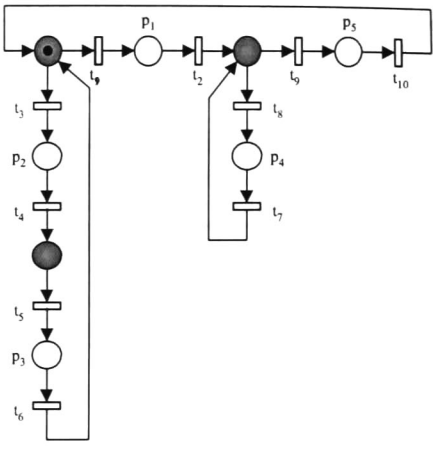


Figure 4.40: Computed model Q_i.

If the next m-word computed is $w_5 = t_9 t_{10}$, then the places $t_6^\bullet$ and ${}^\bullet t_9$ are merged since w_3 and w_5 are the previous and the current m-words w_{n-1} and w_n respectively and there exists a m-word sequence $w_3 w_1 w_5$. The updating of the t-semiflows of Q_i after merge $t_6^\bullet$ and ${}^\bullet t_9$ is $W_1 = W_2 = w_5$, $W_3 = w_4$, $W_4 = w_2 w_3$ and $W_5 = w_1$, notice that these t-semiflows are the t-semiflows of the system model Q depicted on figure 4.34, hence the computed model Q_i generates the entire behavior of Q.

The next algorithm computes non measurable places of Class B preserving the order in which the m-words have been computed.

Algorithm 4.4 *Computing non measurable places of Class B*

Input: The $Dep(Q_n) = Dep(Q_{n-1})$ set, the previous and the current computed m-words $w_{n-1} = t_i \cdots t_j$ $w_n = t_m \cdots t_n$, the set of all computed t-semiflows $\mathcal{W}$. The current t-semiflow is $W_k \in \mathcal{W}$

Output: a model Q_n in which non measurable places of Class B are computed, also a t-semiflow W_i in φC

---

1. SELECTION. If w_{n-1} belongs to W_k and also to another $W_j \neq W_k$ and w_n is computed by first time then

   (a) Since w_{n-1} belongs to another $W_j \neq W_k$ this implies that there exists a NDep $[t_j, t_k] = p_x$ such that t_k does not belong to w_n. Remove the NDep $[t_n, t_m]$. To form the NDep $[t_j, t_m]$ based on identification step 4.2 case 1, it is needed to add an arc from p_x to t_m and to form the NDep $[t_n, t_y]$, where t_y is the first transition of W_j, it is needed to add an arc from t_n to ${}^\bullet t_y$.

   (b) $W_k = W_k w_n$.

   (Identification step 4.6 Case 1).

2. ATTRIBUTION. If w_n belongs to another $W_j \neq W_k$ and w_{n-1} belongs to W_k then

(a) Since w_n belongs to another $W_j \neq W_k$ then there must exist a NDep $[t_k, t_m] = p_y$, such that t_k does not belong to w_n. Remove the arc from t_j to ${}^\bullet t_y$, where t_y is the first transition of W_k. To form the NDep $[t_j, t_m]$ based on identification step 4.2 case 2 it is only needed to add an arc from t_j to p_y.

(b) $W_k = W_k w_n$.

(Identification step 4.6 Case 2).

3. MERGING. If there exists a $DSeq(t_j, t_m)$

Case a. $w_n = w_{n-1}$

If $t_l^\bullet$ forms a complex NDep, where t_l is the last transition of w_j and w_j is the transition sequence $DSeq(t_j, t_m)$ without considering t_j and t_m, then remove the NDep $[t_j, t_x]$, where t_x is the first transition of W_k and form the NDep $[t_j, t_m] = t_l^\bullet$. Make $W_k = w_n$.

else if merge the non measurable places $t_j^\bullet$ and ${}^\bullet t_m$ as stated in Case b.ii

Case b. w_{n-1} and w_n belong to the same computed t-semiflow W_k

(a) i. If w_n occurs before w_{n-1} in W_k then

a. Remove the NDep $[t_j, t_k]$ and add the NDep $[t_j, t_m] =^\bullet t_m$ and the NDep $[t_n, t_k] = t_n^\bullet$ to $Dep(Q_n)$ and mark $t_n^\bullet$.

b. The t-semiflows are updated as $W_k = w_j w_n$ and $W_{k+1} = w_q w_n - 1 w_n$, where w_j is the transition sequence $DSeq(t_j, t_m)$ without considering transitions t_j and t_m and wq is the transition sequence $DSeq(t_n, t_i)$ without considering the transitions t_n and t_i.

ii. Else if w_n occurs after w_{n-1} in W_k then

a. Merge $t_j^\bullet$ and ${}^\bullet t_m$ to compute a complex NDep. (Proposition 4.12). If $t_j^\bullet$ or ${}^\bullet t_m$ is marked then the new computed non measurable place is also marked.

b. Mark as t_F the last transition of w_j, where w_j is the transition sequence $DSeq(t_j, t_m)$ without considering t_j and t_m.

c. The t-semiflows of Q_n are updated as follows: $W_i = W_i / w_j$ and $W_{k+1} = w_j$. (Identification step 4.8).

(Identification step 4.7).

Case c. w_{n-1} and w_n belong to different computed t-semiflows

(a) i. If t_l and t_m are the last and the first transition of consecutive computed m-words, where t_l is the last transition of the transition sequence $DSeq(t_j, t_m)$ then merge the places $t_j^\bullet$ and ${}^\bullet t_m$ following step Case b.ii.

ii. Else if t_l and t_m are not the last and the first transition of consecutive computed m-words then

a. Remove the arc form $t_l^\bullet$ to t_i and the NDep $[t_j, t_k]$ where t_k is the first transition of $DSep(t_j, t_m)$.

b. Compute the NDep $[t_l, t_i] = p_x$ such as p_x is a new non measurable place and the NDep $[t_j, t_k] =^{\bullet} t_m$.

c. The computed t-semiflows are updated as follows: $\forall Wi$ such that $w_n \in W_i$, $W_i = w_i w_{n-1}$ and $W_k = W_k / w_{n-1}$ and $W_k = W_k w_{n-1}$.

4. CONCATENATION. w_n is a new m-word

a. If the last transition of W_k is not marked as t_F

i. Remove the NDep $[t_n, t_m]$ and the NDep $[t_y, t_x]$, such that t_x and t_y are the first and the last transitions of W_k respectively.

ii. Compute the NDep $[t_n, t_x] = p_k$, if t_x has an input non measurable place then $p_k =^{\bullet} t_x$ else p_k is a new non measurable place and p_k will be marked.

iii. compute the NDep $[t_y, t_m] = p_k$, if t_j has an input non measurable place then $p_k = t_j^{\bullet}$ else p_k is a new non measurable place.

iv. $W_k = W_k w_n$

Identification step 4.5.

b. Else if the last transition of W_k is marked as t_F

(a) Remove the NDep $[t_n, t_m]$ and form the NDep $[t_n, t_m] = {}^{\bullet}t_i$, where t_i is the first transition of W_j where w_{n-1} belongs.

(b) $W_{k+1} = w_n$

Identification step 4.9.

In a non measurable dependency $p_k = [t_i, t_j]$ of Class B t_i and t_j may belong to different m-words. In the case that p_k belongs to Class B.i.a or Class B.ii.a, t_i and t_j belong to different m-words of a same t-semiflow X, while in the case that p_k belongs to Class B.ii.b, t_i and t_j are the last and the first transition of an actual t-semiflow of the system. The non measurable places of Class B.i forms a single NDep and they are computed concatenating the previous and the current computed m-words w_{n-1} and w_n respectively. The non measurable places of Class B.ii.a form complex NDep, as well as the non measurable places of Class B.i; these non measurable places are computed concatenating previous and current m-words, however in this case the concatenation is made with an already computed non measurable place. The non measurable places of Class B.ii.b, also forms complex NDep, however these places are computed merging the output non measurable place of the last transition of w_{n-1} with the input non measurable place of the first transition of w_n.
These places cannot be computed directly when a m-word is computed even if all m-words have been computed, they must be inferred as more information of the system is detected in terms of the occurrence order of the m-words.

Using the strategies provided by propositions previously introduced, it can be computed models describing the behavior of DES that exhibit certain behavior such as decisions, synchronizations and concurrence. Then it is

possible to compute marked graphs, states machines, free choice, and simple Petri nets as a model for a DES. However, there exist specific structures that cannot be computed, this problem is addressed in section 6.2.

◊ Concurrent-independent transitions

There exist also another kind of relationship between places and transitions describing more general PN than the PN that can be identified using the procedures previously introduced.
In an ordinary PN there is no restriction on how the places and transitions are related. However it can be distinguished two subclasses of ordinary PN according to how the transitions of each t-semiflow are related with the transitions of another t-semiflow in the ordinary PN.

Definition 4.11 *Let Q be an IPN, two transitions t_i and t_j are independent transitions if the firing of t_i is not constrained to the firing of t_j and viceverse.*

The set of all independent transitions in an IPN Q will be denoted as $Independent(Q)$, two independent transitions t_i, t_j are denoted as $i[t_i, t_j]$ which is equivalent to $i[t_j, t_i]$.

Proposition 4.16 *Let Q be an IPN and t_i and t_j be any two transitions of Q. If t_i and t_j belong to different t-semiflows then t_i and t_j are independent transitions.*

Proof. The proof follows from definition 4.11. ■

Proposition 4.17 *Let Q be an IPN and t_i and t_j be any two transitions of Q. If*

a) t_i and t_j are concurrent transitions belonging to the same t-semiflow X_i and

b) t_i belongs to another t-semiflow X_j such that t_j does not belong to X_j, or t_j belongs to another t-semiflow X_k in which t_i does not belong

then t_i and t_j are independent transitions.

Proof. Notice that if t_i and t_j belongs to a t-semiflow X_i and t_i (t_j) belongs to another t-semiflow X_j (X_k) in which t_j (t_i) does not appear, then by definition 4.11 t_i and t_j are independent transitions since there exists at least one t-semiflow X_j (X_k) in which the firing of t_i (t_j) is not constrained to the firing of t_j (t_i). However if t_i and t_j fulfill only the condition b t_i and t_j could be dependent transitions in the case where t_i or t_j are shared by two or more t-semiflows since they always would occur in the same order. ■

Example. Even the transitions t_{11} and t_{12} of the PN depicted on figure 4.41.a belong to the same t-semiflow $X_1 = t_8t_9t_{10}t_{11}t_{12}$ and t_{12} belongs also to another t-semiflow $X_2 = t_8t_9t_{12}t_{13}t_{14}$, these two transitions do not fulfill the condition a of proposition 4.17 since they are consecutive transitions, i.e. they are not concurrent transitions and hence they are not independent transitions. The transitions t_2 and t_{10} of the PN depicted on figure 4.41.b fulfill both conditions of proposition 4.17, since there exists a t-semiflow $X_i = t_2t_3t_5t_6t_7t_8t_{10}t_{11}$ in which they are concurrent transition and there exist another t-semiflow $X_j = t_1t_2t_3t_4t_6t_7$ in which t_{10} does not belong and also another t-semiflow $X_k = t_6t_7t_9t_{10}t_{11}t_{12}$ in which t_2 does not belong, hence t_2 and t_{10} are independent transitions.

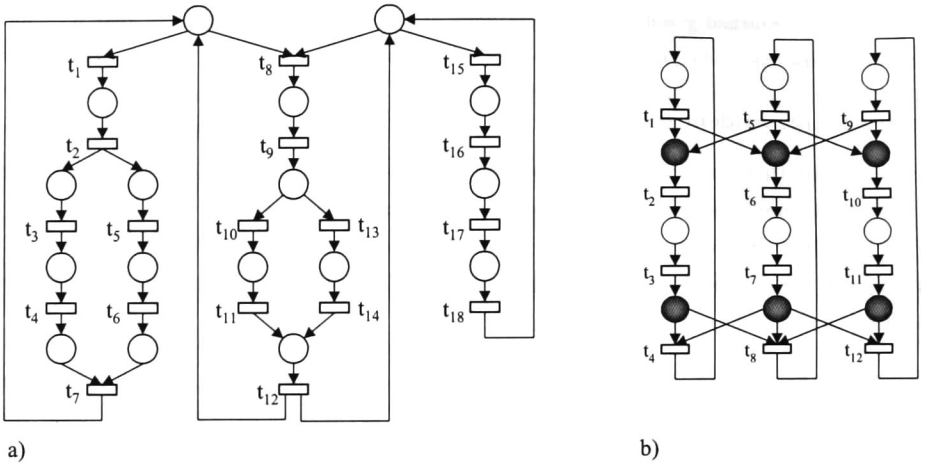


Figure 4.41: a) Ordinary PN fulfilling only condition b of proposition 4.17. b) Ordinary PN fulfilling both conditions of proposition 4.17.

The ordinary PN fulfilling conditions *a* and *b* of proposition 4.17, have transitions which are both concurrent and independent; this is the class of ordinary PN studied in this section, since for the ordinary PN defined in proposition 4.16, a combination of procedures presented in previous section can be used to identify them.

Proposition 4.18 *Let Q be a system model, Q_n be the computed model for Q, $w_n = t_x \cdots t_y$ be the current m-word computed from the last output word generated by Q, and t_i and t_j be any two consecutive transitions in w_n. If t_i and t_j are independent transitions and there exists not a dependency $[t_i, t_j]$ in $Dep(Q_n)$, then the NDep $[t_i, t_j] = p_k$ is not added to $Dep(Q_n)$, where p_k is a new non measurable place as stated in identification step 4.1.*

Proof. Since t_i and t_j are independent transitions, the NDep $[t_i, t_j] = p_k$ cannot not be added to $Dep(Q_n)$ because it would not allow the firing of t_j without the firing of t_i which contradicts the definition of independent transitions. ■

Previous proposition states that it is not possible to add a new non measurable place to form a NDep between any two independent transitions. Next two procedures state when it is possible to compute a NDep formed with this kind of transitions.

Identification Step 4.10 *Let Q be a system model, Q_n be the computed model for Q, $w_n = \cdots t_i t_j \cdots$ be the m-word computed from the last output sequence produced by Q, and t_i and t_j be any two independent consecutive transitions in w_n. If there exists not a dependency $[t_i, t_j]$ in $Dep(Q_n)$ and there exists a NDep $[t_i, t_k] = p_k$ in $Dep(Q_n)$ such that t_k does not belong to w_n then the NDep $[t_i, t_j] = p_k$ will be added to $Dep(Q_n)$.*

By proposition 4.18, a NDep $[t_i, t_j]$ cannot be computed using a new non measurable place when two consecutive transitions t_i and t_j are independent transitions. However, if there exists a NDep $[t_i, t_k] = p_k$ in $Dep(Q_n)$ it can be computed the NDep $[t_i, t_j]$ using the same non measurable place p_k, forming a decision between t_k and t_j. Since t_j and t_k belong to different t-semiflows $w_n = \cdots t_i t_j \cdots$ can be generated by Q_n because when t_i is

fired the place p_k is marked and as t_j is fired after t_i in $w_n = \cdots t_i t_j \cdots$, then the marks into the place p_k are removed. Notice that if t_k belongs to $w_n = \cdots t_i t_j \cdots$ and the NDep $[t_i, t_j] = p_k$ is added to $Dep(Q_n)$ then the model Q_n will not be live since after the firing of t_i, t_j or t_k could be fired and hence the firing of t_j disables the firing of t_k or the firing of t_k disables t_j according which is the first fired transition, not allowing the occurrence of the sequence $w_n = \cdots t_i t_j \cdots$ in Q_n.
The above identification step states that a NDep $[t_i, t_j]$ can be computed, where t_i and t_j are any two consecutive independent transitions in the current computed m-word w_n; just in the case t_i has an output non measurable place $t_i^\bullet = [t_i, t_k]$ and t_k does not belong to w_n.

Identification Step 4.11 *Let Q be a system model, Q_n be the computed model for Q, $w_n = \cdots t_i t_j \cdots$ be the m-word computed from the last output sequence produced by Q and t_i and t_j be any two independent consecutive transitions in w_n. If there exists not a dependency $[t_i, t_j]$ in $Dep(Q_n)$ and there exists a NDep $[t_k, t_j] = p_k$ in $Dep(Q_n)$ such that t_k does not belong to $w_n = \cdots t_i t_j \cdots$, then a NDep $[t_i, t_j] = p_k$ will be added to $Dep(Q_n)$.*

By identification step 4.18, a NDep $[t_i, t_j]$ cannot be computed using a new non measurable place when two consecutive transitions t_i and t_j are independent transitions. However, if there exists a NDep $[t_k, t_j] = p_k$ in $Dep(Q_n)$ it can be computed the NDep $[t_i, t_j]$ using the same non measurable place p_k, forming an attribution between t_k and t_i. Since t_j and t_k belong to different t-semiflows $w_n = \cdots t_i t_j \cdots$ can be generated by Q_n due to when t_i is fired the place p_k is marked and as t_j is fired after t_i in $w_n = \cdots t_i t_j \cdots$ then the place p_k is unmarked and hence never could be more than one mark into the place p_k. Notice that if t_k belongs to $w_n = \cdots t_i t_j \cdots$ and the NDep $[t_i, t_j] = p_k$ is added to $Dep(Q_n)$, then $\overrightarrow{w_n}$ would not be a t-semiflow in Q_n due to each transition t_i and t_k add one mark to p_k when they are fired, hence after the firing of t_j one mark remains into p_k; this leads to the situation in which after the firing of all transitions in w_n, Q_n reaches a marking which is different to the marking in which the first transition of w_n was fired.
The above identification step states that a NDep $[t_i, t_j]$ can be computed, where t_i and t_j are any two consecutive independent transitions in the current computed m-word w_n; just in the case t_j has an input non measurable place ${}^\bullet t_j = [t_k, t_j]$ and t_k does not belong to w_n.
The next algorithm presents how to detect independent transitions and how they can form a complex NDep.

Algorithm 4.5 *Computing complex NDep for independent transitions*

Input: $Dep(Q_{n-1})$, the current m-word $w_n = t_m \cdots t_n$ and the set of computed t-semiflows $\mathcal{W}$.
Output: The model Q_n describing the observed behavior of a system model Q

---

1. $Dep(Q_n) = Dep(Q_{n-1})$
2. If there exists a t-semiflow X_j in Q_n fulfilling that $\overrightarrow{w_n} = X_j$ then update the possible wrong single NDep computed. Algorithm 4.2.
3. Let t_i and t_j be any two consecutive transitions in $w_n = t_m \cdots t_n$
4. If there exist a NDep $[t_x, t_i]$ and/or $[t_y, t_j]$ such that t_x and t_y are computed after t_i and t_j respectively in $w_n = t_m \cdots t_n$, then remove this NDep from $Dep(Q_n)$ and add $[t_x, t_i]$ and/or $[t_y, t_j]$ to $Concurrent(Q_n)$.

5. If a NDep is removed from $Dep(Q_n)$ then two cases could arise for any transition t_x:

   (a) If t_x remains as a *source* transition, then add a NDep $[t_y, t_x]$ to $Dep^u(Q_n)$ such that t_y is the predecessor of t_x in the previous m-word in which the transitions of the removed NDep were consecutive (or in the current m-word w_n in the case that this step had been reached from step 7).

   (b) If t_x remains as a *sink* transition then add a NDep $[t_x, t_y]$ to $Dep^u(Q_n)$ such that t_y is the successor of t_x in the previous m-word in which the transitions of the removed NDep were consecutive (or in the current m-word w_n in the case that this step had been reached from step 7).

   It is possible that the transition t_y, predecessor or the successor respectively of t_x in m-word be concurrent or independent with t_x :

   i. If $[t_x, t_y] \in Concurrent(Q_n)$ then select the following predecessor or successor of t_x in w_n i.e. $t_y = {}^\bullet({}^\bullet t_y)$ or $t_y = (t_y^\bullet)^\bullet$ respectively until the NDep $[t_x,t_y] \notin Concurrent(Q_n)$ or t_y does not appear before t_x in w_n and then add the NDep $[t_y, t_x]$ or $[t_x, t_y]$ to $concurrent(Q_n)$.

   ii. If $[t_x, t_y] \in Independent(Q_n)$ then select the following predecessor or successor of t_x in w_n i.e. $t_y = {}^\bullet({}^\bullet t_y)$ or $t_y = (t_y^\bullet)^\bullet$ respectively until the NDep $[t_x,t_y] \notin Independent(Q_n)$ and then go to step 6 such that $t_i = t_x$ and $t_j = t_y$ if t_x is a sink transition or $t_i = t_y$ and $t_j = t_x$ if t_x is a source transition. If $[t_x, t_y]$ is added to $independet(Q_n)$ set then select the following predecessor or successor of t_x i.e. $t_y = {}^\bullet({}^\bullet t_y)$ or $t_y = (t_y^\bullet)^\bullet$ respectively and repeat this step.
   Else use the algorithm 4.6 to reconnect the removed arcs verifying only the remaining input place of t_x.

6. If $[t_i, t_j] \notin Dep(Q_n)$ and $[t_i, t_j] \notin Concurrent(Q_n) \cup Independent(Q_n)$ then

   (a) If $[t_i,t_j] \notin Independent(Q_n)$ but they fulfill the conditions of independence (Definition 4.11 and Proposition 4.17) then

   i. If t_i has an output non measurable place p_k forming the NDep $[t_i, t_k] = p_k$
   if t_k is not a transition of w_n then add the NDep $[t_i, t_j] = p_k$ to $Dep(Q_n)$ and also add $[t_i, t_k]$ and $[t_j, t_k]$ to $Independent(Q_n)$.
   else add $[t_k, t_j]$ to $Independent(Q_n)$
   (Identification step 4.10).

   ii. If t_j has an input non measurable place p_k forming the NDep $[t_k, t_j] = p_k$
   if t_k is not a transition of w_n then add the NDep $[t_i, t_j] = p_k$ to $Dep(Q_n)$ and also add $[t_k, t_i]$ and $[t_k, t_j]$ to $Independent(Q_n)$
   else add $[t_k, t_i]$ to $Independent(Q_n)$
   (Identification step 4.11).

   (b) If the NDep $[t_i, t_j]$ was not added to $Dep(Q_n)$ in step 6.a.i or 6.a.ii then add $[t_i, t_j]$ to $Independent(Q_n)$.

   Else add the NDep $[t_i, t_j]$ to $Dep(Q_n)$. (Proposition 4.1).

7. If after evaluate $w_n = \cdots t_i t_j \cdots$ remain source or sink transitions in Q_n then go to step 5.a or 5.b respectively for each source or sink transition.

8. Use the algorithm 4.6 to verify that every non measurable place be correctly connected allowing that all transition sequences computed from the output of the system can be generated by Q_n.

When an arc is removed (step 4 of the above algorithm) to form the t-semiflow computed from the last output word observed could be possible that others t-semiflows be disconnected. The next procedure is useful to verify that in the updated model Q_n all the computed t-semiflows are correctly connected.

Algorithm 4.6 *Checking the t-semiflows of* Q_n.

Input: The γC_{Q_n} matrix and the set of computed t-semiflows $\mathcal{W}$.
Output: The updated (if it is necessary) model Q_n.

---

1. Multiply each row r_i of the matrix γC_{Q_n} by all the computed t-semiflows $W_k \in \mathcal{W}$
2. If any row r_i of γC_{Q_n} does not fulfill that $\overrightarrow{r_i} \cdot W_k = 0$ for some t-semiflow W_k then
   (a) Select the last m-word σ such that $\overrightarrow{\sigma} \in W_k$
   (b) Detect the column k of γCQ_n, in which there exists a data in the row r_i making that $r_i \cdot W_k \neq 0$

   case 1 $\gamma C_{Q_n}(i,k) = 1$: this implies that a -1 is missed in the row r_i, then
   1. Add a -1 in the position $\gamma C_{Q_n}(i,m)$, where m is the column representing the transition t_m of Q_n such that t_m is a successor of t_k in the m-word σ
   2. If $[t_k, t_m] \in Dep(Q_n)$ or $[t_k, t_m] \in Concurrent(Q_n) \cup Independent(Q_n)$ then make $t_m = (t_m^{\bullet})^{\bullet}$ (the successor of t_m in σ) and repeat this step until $[t_k, t_m] \notin Dep(Q_n)$ or $[t_k, t_m] \notin Concurrent(Q_n) \cup Independent(Q_n)$

   case 2 $\gamma C_{Q_n}(i,k) = -1$: this implies that a 1 is missed in the row r_i.
   1. Add a 1 in the position $\gamma C_{Q_n}(i,n)$, where n is the column representing the transition t_n of Q_n such that t_n is a predecessor of t_k in the m-word σ
   2. If $[t_n, t_k] \in Dep(Q_n)$ or $[t_n, t_k] \in Concurrent(Q_n) \cup Independent(Q_n)$ then make $t_n = {}^{\bullet}({}^{\bullet}t_n)$ and repeat this step until $[t_n, t_k] \notin Dep(Q_n)$ or $[t_n, t_k] \notin concurrent(Q_n) \cup independent(Q_n)$ Notice that when it is added a -1or a 1 as stated in steps 2.b case 1 and 2.b case 2 it is computed a NDep $[t_k, t_m]$ or $[t_n, t_k]$ respectively, this new NDep need to be computed according to step 5 of algorithm 4.5, where $t_i = t_k$ and $t_j = t_m$ or $t_i = t_n$ and $t_j = t_k$, if it is used the step 2.c or the step 2.d respectively.

Notice that this algorithm only perform the test over the non measurable places of Q_n since the measurable places are computed correctly (Proposition 4.3) from the output of the system, hence the algorithm 4.6 check that every row in γC_{Q_n} matrix multiplied by all the computed t-semiflows be equal to zero to guarantee that each non measurable place is connected correctly in every t-semiflow, if this condition is not fulfilled in some row r_i for some t-semiflow X_k then a one (1) or a minus one (-1) is added in the position (i,j) of the matrix γC_{Q_n}, where j is the column representing the transition in which the t-semiflow is not satisfied by the place i, according if t_j is a predecessor or a successor of the transition $t_m \in X_k$ appearing in the row r_i respectively. The addition of a one (1) or a minus (-1) to the γC_{Q_n} matrix is an input or an output arc respectively of the non measurable place p_i.

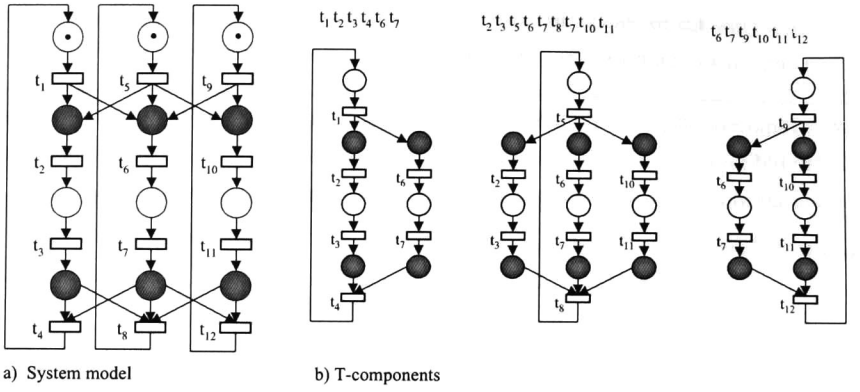


Figure 4.42: System model and its t-components.

Example. Consider the system model Q depicted on figure 4.42.a; the next table summarizes the steps of algorithm 4.5 to update the non measurable places of each model computed to identify Q.

	m-word	identification step Algorithm 4.5	Figure
1	w_1=$t_1t_6t_2t_3t_7t_4$	5	4.43.1
2	w_2=$t_5t_2t_{10}t_6t_3t_{11}t_7t_8$	Proposition 4.3	4.43.2
3	t_5t_2	4 5.a.i and 6.a.ii $c[t_2,t_6]$ NDep $[t_5,t_2]$ and $i[t_1,t_5]$, $i[t_1,t_2]$	4.43.3 and 4.44.4
4	t_2t_{10}	6.b- $i[t_2,t_{10}]$	
5	$t_{10}t_6$	6.a.ii-NDep $[t_{10},t_6]$ and $i[t_1,t_6]$, $i[t_1,t_{10}]$	4.44.5
6	t_6t_3	6.b- $i[t_6,t_3]$	
7	t_3t_{11}	6.a.i- $i[t_{11},t_7]$ and 6.b- $[t_3,t_{11}]$	
8	$t_{11}t_7$	$[t_{11},t_7]\in$ *Independent*(Q_n)	
9	t_7t_8	6ai- NDep $[t_7,t_8]$ and $i[t_4,t_7]$, $i[t_4,t_8]$	4.44.6
10	t_{10}	7-5aii-6- NDep $[t_5,t_{10}]$	4.45.7
11	t_{11}	7-5bii-6- NDep $[t_{11},t_8]$	4.45.8
12	w_3=$t_9t_6t_{10}t_7t_{11}t_{12}$	Proposition 4.3	4.45.9
13	t_9t_6	4-c$[t_6,t_{10}]$ and 6.a.ii.- NDep $[t_9,t_6]$ and $i[t_1,t_9]$	4.46.10
14	t_6t_{10}	$[t_6,t_{10}]\in$ *Concurrent*(Q_n)	
15	$t_{10}t_7$	6.b- $i[t_{10},t_7]$	
16	t_7t_{11}	$[t_{11},t_7] \in$ *Independent*(Q_n)	
17	$t_{11}t_{12}$	6.a.i-NDep $[t_{11},t_{12}]$ and $i[t_8,t_{11}]$, $[t_8,t_{12}]$	4.46.11
18		7 using the algorithm 4.6	4.47.12
19		The updated model Q_n	4.47.13

A new IPN model is computed when the current m-word is completely evaluated by the algorithm, hence the IPN model sequence stated in section 4.3 for the system model Q depicted on figure 4.42 is composed by the models 1, 8 and 13 depicted on figures 4.43, 4.45 and 4.47 respectively, which were computed when the m-words w_1, w_2 and w_3 were processed respectively by algorithm 4.5. Notice that each one of these models describe the observed behavior of the system, however the last computed model is not equal to Q, this implies that it is needed to obtain more information from Q to update the wrong NDep computed. Notice that with new m-words in which the transitions occurs in different order than in previous m-words computed, the wrong

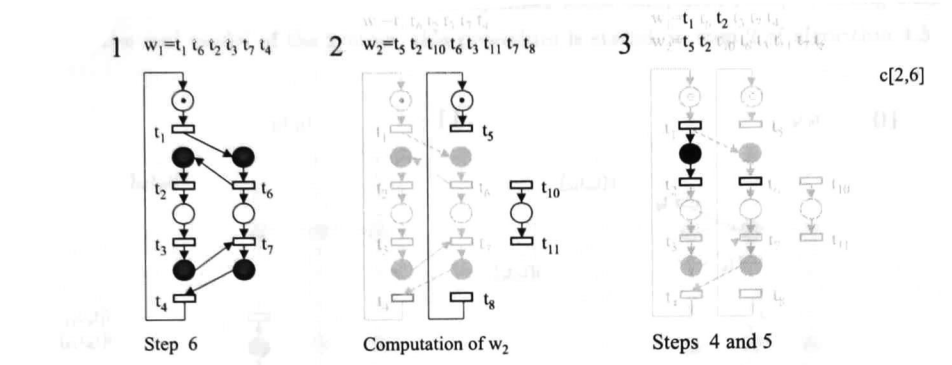


Figure 4.43: Computed models.

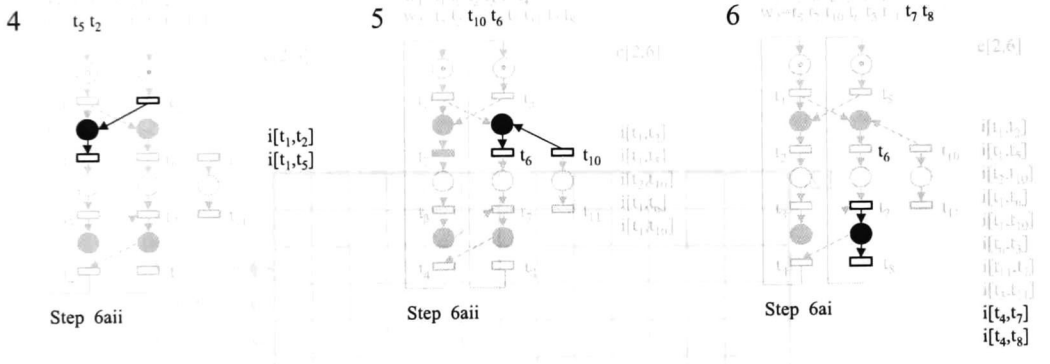


Figure 4.44: Computed models.

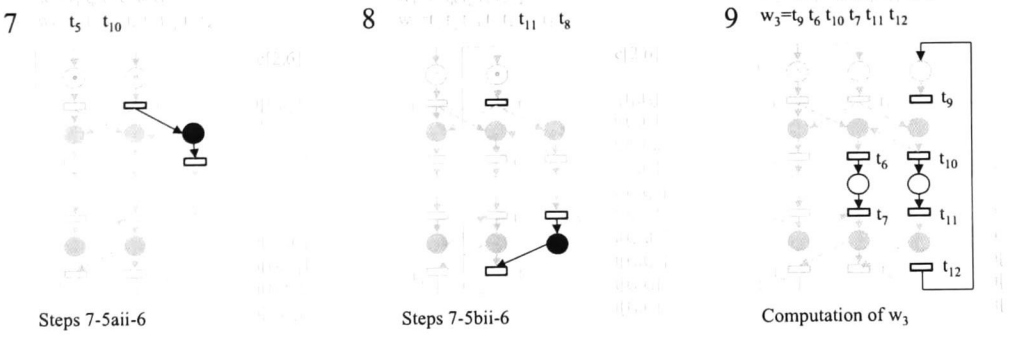


Figure 4.45: Computed models.

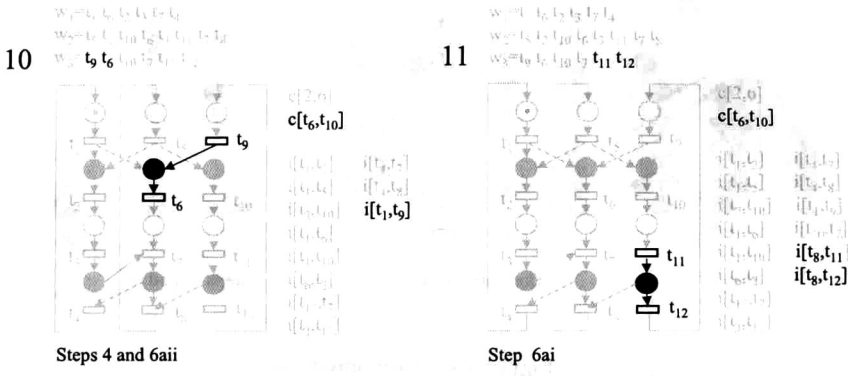
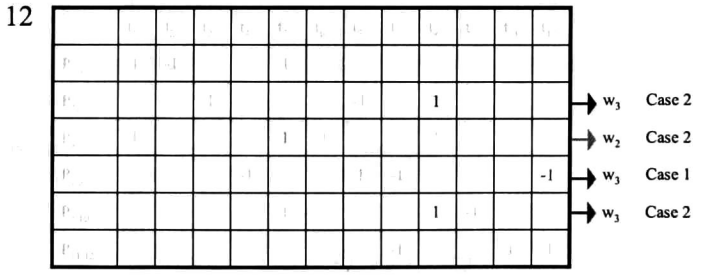


Figure 4.46: Computed models.



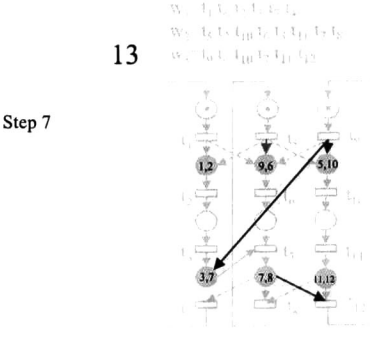


Figure 4.47: Computed model.

computed NDep will be removed and also will be updated some computed NDep building then a new model approaching to the real model of the system, this procedure is stated on step 2 of algorithm 4.5.

Chapter 5

Asymptotic identification algorithm

Based on procedures to update a computed model, in this chapter is presented the asymptotic identification algorithm that computes a new IPN model when new information of the system is detected. Each model is computed as the system evolves and it is fulfilled that the current model is more approached to the actual model of the system than the previous computed model.

5.1 Introduction

The previous chapter presented the procedures to update a model when a m-word is computed. Based on the analysis of model updating, the aim of this chapter is to present the complete identification process to compute a new model Q_i when new information of the system model Q is provided.
Also, here is stated that when a new model Q_i is computed updating the previous computed model Q_{i-1} using the proposed identification process it is fulfilled that $f(Q, Q_i) \leq f(Q, Q_{i-1})$, implying that Q_i is better approached to Q than Q_{i-1}. Both, identification process and convergence criterion are presented in a theorem. The information provided by the system is represented by the m-words computed. As new information is considered:

1. A new computed m-word
2. A new occurrence order of the transitions in a m-word
3. A new occurrence order of the computed m-words

The next algorithm summarizes the identification process.

Algorithm 5.1 *Asymptotic identification approach*

---

1. Read the output symbol $\varphi(M_n)$ generated by the system Q.
2. Detect the output word $o_i = \varphi(M_i)\varphi(M_k)\cdots\varphi(M_j)$, such that $\varphi(M_i) = \varphi(M_j)$. (Definition 4.8).
3. For any two consecutive output symbols compute a transition $t_i = \varphi(M_i) - \varphi(M_{i-1})$. (Proposition 4.3).
4. Compute the m-word $w_i = t_i t_j \cdots$ adding each computed transition in the step above. (Proposition 4.3).
5. Compute the non measurable places. (Identification Step 4.1)
   (a) to constrain the firing order of the transitions in w_i to the order in which they were computed
   (b) to compute the t-component associated with w_i.
6. Infer an actual t-semiflow W_i of Q concatenating the previous and the current m-words w_{n-1} and w_n, due to it is possible that the computed m-word w_i in step 4 could not be a t-semiflow of Q. Identification step 4.5.
7. Observe the next output word O_n and compute its associated m-word w_n. (Definition 4.8 and Proposition 4.3).
8. Update the computed IPN model (computing a new model) with the information provided by the m-word w_n, allowing the firing of all computed m-words w_j. (Identification step 4.2, algorithm 4.1, identification step 4.3, algorithm 4.2, algorithm 4.6, identification step 4.7, algorithm 4.5, identification step 4.10, identification step 4.11, algorithm 4.5).
   (a) computing new measurable places and transitions

(b) removing or adding NDep

i. updating the computed actual t-semiflows

5.2 Asymptotic identification procedure

Theorem 5.1 *Let Q be a system model and Q_{n-1} be the proposed model for Q when the m-word w_{n-1} was computed. If w_n is the current computed m-word then a new model Q_n can be computed updating the previous model Q_{n-1} such that $f(Q,Q_n) \leq f(Q,Q_{n-1})$ is fulfilled.*

Proof. Let $f(Q,Q_n) = |\{\varphi C_Q\} - \{\varphi C_{Q_n}\}| + |Dep^u(Q) - Dep^u(Q_n)| + |Dep^u(Q_i) - Dep^u(Q)|$ be the identification error of the computed model Q_n with respect to the system model Q. At the beginning of the identification procedure the identification error is bounded by the number of columns of the φC matrix of Q and the number of NDep of Q, then the initial error when the computed model is $Q_0 = \emptyset$ is defined as:

$$f(Q,Q_0) = |\{\varphi C_Q\}| + \sum_{\forall p_i \in \gamma C_Q} a^i * b^i \tag{5.1}$$

The term $\sum_{\forall p_i \in \gamma C_Q} a^i * b^i$ is the number of all NDep of Q, where a^i is the number of input transitions of the non measurable place p_i and b^i is the number of output transitions of the non measurable place p_i, notice then that $|Dep^u(Q)| = \sum_{\forall p_i \in \gamma C_Q} a^i * b^i$.
The equation 5.1 represents the number of elements of Q in terms of columns of φC matrix and NDep of Q that are missed in the computed model. Since $Q_0 = \emptyset$ then the term $|Dep^u(Q_0) - Dep^u(Q)| = 0$. Hence it is considered as initial error the number of all elements of Q.

Let $\mathcal{W} = \{W_i\}$ be the set of computed t-semiflows when the xth m-word is computed, such that every W_j is a concatenation of some selected m-words.

Let $\mathcal{T}$ be the set of all computed m-words; initially $\mathcal{T} \leftarrow \emptyset$

Let $Concurrent(Q_n)$ be the set of all computed concurrent transitions; initially $Concurrent(Q_0) \leftarrow \emptyset$

Let $Independent(Q_n)$ be the set of all computed independent transitions; initially $Independent(Q_0) \leftarrow \emptyset$

A transition t_i will be marked as t_F when is detected that it is the last transition of any t-semiflow of the system Q.

Let k be the subindex of the t-semiflows on $\mathcal{W}$. $k \leftarrow 1$.

By induction on the number x of computed m-words, it is presented how the xth IPN model is computed

- Let $x = 1$.

1. Computing the φC matrix representing the measurable part (measurable places) of Q_1. Use proposition 4.3 to compute the first m-word $w_1 = t_1 t_2 ... t_r$.

2. $\mathcal{T} \leftarrow \mathcal{T} \cup \{w_1\}$

3. Computing the γC matrix representing the non measurable part (non measurable places) of Q.

   (a) Constrain the firing order of the transitions in w_1 using algorithm 4.1

4. $W_k = w_1$.

- A new model Q_1 is determined, where the incidence matrix of Q_1 is C_{Q_1} computed as illustrated in identification step 4.1.

- The error $f(Q, Q_1)$ fulfills that $f(Q, Q_1) \leq f(Q, Q_0)$ since it is ensured that the columns $\varphi C(\cdot, t_i)$ exist in the system by proposition 4.3. So $f(Q, Q_1) = (|\{\varphi C_Q\}| - |\{\varphi C_{Q_1}\}|) < (|\{\varphi C_Q\}| + |Dep(Q)|) = f(Q, Q_0)$, assuming in the worst case that no one of the computed NDep is correct.

  By the induction hypothesis, $\forall x < n$, $f(Q, Q_x) \leq f(Q, Q_{x-1})$.

  Let $x = n$. Updating the model using the current computed m-word w_n

1. Let $Dep(Q_n) = Dep(Q_{n-1})$ and $Q_n = Q_{n-1}$

2. Compute the $n - th$ m-word $w_n = t_m \cdots t_n$. Proposition 4.3

3. $\mathcal{T} \leftarrow \mathcal{T} \cup \{w_n\}$

   Inferring the non measurable places of Q the γC matrix representing the non measurable part (non measurable places) of Q.

4. Let $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m ... t_n$ be the previous and the current computed m-words.

   Case 1: w_n is an already computed m-word

   (a) If w_n is the first m-word of any $W_i \in \mathcal{W}$ then

      i. If t_j is not marked as t_F then mark t_j as t_F which implies that w_{n-1} is the last m-word of some $W_i \in \mathcal{W}$ in which w_{n-1} belongs.

      ii. $W_{k+1} = w_n$.

      iii. Restart the algorithm. Compute the next m-word

   (b) else if there exists a NDep $[t_j, t_m]$ then

      i. $W_k = W_k w_n$.

      ii. Restart the algorithm. Compute the next m-word

   Case 2: $w_n = t_m \cdots t_n$ is a new m-word.

   Case a: All or some columns of w_n are computed by first time

   (a) If there exists not independent transitions. (Definition 4.11 and Proposition 4.17)

i. Compute NDep to preserve the firing order of the transitions in w_n using algorithm 4.1. Computing places belonging to the classes A.i and A.ii respectively.

else use algorithm 4.5 to preserve the firing of consecutive transitions which are detected as independent transitions.

Case b: All transitions of w_n have been already computed

(a) If $w_n \in W_i$ and there exists a $W_j \neq W_i$ such that $\overrightarrow{W_i} = \overrightarrow{W_j}$ and $w_n = t_m...t_n$ is different of any w_t of W_j then: Use the algorithm 4.2 to detect the concurrent transitions and to remove the wrong NDep computed in the model Q_n.

(b) If $w_n \in W_x$ and W_x is a subset of a t-semiflow W_y such that w_n is different of any w_t of W_y then replace W_x by W_i and W_y by W_j and use the algorithm 4.2 to detect the concurrent transitions and to remove the wrong NDep computed in the model Q_n.

5. Validating the order in which w_n has occurred with respect to the others computed m-words to rebuilt the real t-semiflows of Q. Using algorithm 4.4.

A new model Q_n is determined: the matrix φC is computed in step 2 while the matrix γC is computed incrementally from step 4 to step 5.

When new transitions are computed the term $|\{\varphi C_Q\} - \{\varphi C_{Q_n}\}|$ is reduced while when new NDep are computed the term $|Dep^u(Q) - Dep^u(Q_n)|$ is reduced, however when a computed NDep does not belong to Q the term $|Dep^u(Q_n) - Dep(Q)|$ increases.

There exist two cases in where the number $|Dep^u(Q_n) - Dep^u(Q)|$ is increased:

Case 1. When consecutive transitions in a m-word are concurrent transitions and a NDep between them is (wrongly) computed.

In this case the total number of wrong computed NDep is bounded by:

$$\sum_{\forall T_i}((c^i - 1) * d^i) + (d^i - 1) \tag{5.2}$$

where T_i is a component of Q having concurrent transitions, c^i is the number of p-components of T_i and d_i is the greater number of transitions in any p-component of T_i. The number of wrongly computed NDep of this case can remain even if a new m-word is computed, this number will be reduced when m-words of the same t-component T_i are computed.

Case 2. When the current m-word w_n is concatenated with previous m-word w_{n-1} and w_n is not the last m-word of the m-word decomposition of its underlaying t-semiflow.

In this case the term $|Dep^u(Q_n) - Dep^u(Q)|$ is increased in one unity since the NDep $[t_n, t_x]$ computed between the last transition of w_n and the first transition of the computed t-semiflow in which w_{n-1} belongs t_n and t_y respectively does not belong to Q.

Hence, the worst case considered is when the current m-word w_n has concurrent transitions and it is not the last m-word of its underlying t-semiflow.

The greater number of wrong computed NDep when a new m-word is computed is bounded by:

$((c^i-1)*d^i)+(d^i-1)+1$. However the value of this term never makes that the global error be greater than the identification error of previous model because 1) the number of computed transitions of a m-word is greater than the number of the NDep that can be wrongly computed i.e. $[((c^i-1)*d^i)+(d^i-1)] < (c*d)$ and 2) the wrongly computed NDep $[t_n, t_x]$ is removed when the next m-word is computed.

Thus, the error fulfills that $f(Q, Q_n) \leq f(Q, Q_{n-1})$ and the theorem holds.

■

5.3 Example

Consider the system model Q depicted on figure 5.1 the system to be identified. The next table summarizes the steps of the identification procedure to build the system model. Figures from 5.2 to 5.13 show the succesive computed models.

	Output word	m_word	Identification step (Theorem 5.1)	Figure
1.	$w_1 = \epsilon p_1 \epsilon$	$m_1 = t_1 t_2$	4 Case 2-Case a	5.2
2.	$w_2 = \epsilon p_2 \epsilon$	$m_2 = t_3 t_4$	a) 4 Case 2-Case a, b) 5-4	5.3
3.	$w_3 = \epsilon p_4 \epsilon$	$m_3 = t_6 t_7$	a) 4 Case 2-Case a, b) 5-4	
4.	$w_4 = \epsilon p_3 \epsilon$	$m_4 = t_5 t_8$	a) 4 Case 2-Case a, b) 5-4	5.4
5.	$w_5 = \epsilon p_5 \epsilon$	$m_5 = t_9 t_{10}$	a) 4 Case 2-Case a, b) 5-4	
6.	$w_6 = \epsilon p_6 \epsilon$	$m_6 = t_{11} t_{12}$	a) 4 Case 2-Case a, b) 5-4	
7.	$w_7 = \epsilon p_7 \epsilon$	$m_7 = t_{13} t_{14}$	a) 4 Case 2-Case a, b) 5-4	5.5
8.	$w_1 = \epsilon p_1 \epsilon$	$m_1 = t_1 t_2$	4 Case 1-a	
9.	$w_2 = \epsilon p_2 \epsilon$	$m_2 = t_3 t_4$	4 Case 1-b	
10.	$w_{3-4} = \epsilon p_3 \begin{matrix} p_3 \\ p_4 \end{matrix} p_3 \epsilon$	$m_{3-4} = t_5 t_6 t_7 t_8$	4 Case 2-Case b-a	5.6
11.	$w_7 = \epsilon p_7 \epsilon$	$m_7 = t_{13} t_{14}$	5-3	5.7
12.	$w_8 = \epsilon p_8 \epsilon$	$m_8 = t_{15} t_{16}$	Case 2-Case a and 5-5	5.8
13.	$w_9 = \epsilon p_9 \epsilon$	$m_9 = t_{17} t_{18}$	Case 2-Case a and 5-4	
14.	$w_{10} = \epsilon p_{11} \epsilon$	$m_{10} = t_{21} t_{22}$	Case 2-Case a and 5-4	5.9
15.	$w_7 = \epsilon p_7 \epsilon$	$m_7 = t_{13} t_{14}$	Case 2-Case a and 5-3	5.10
16.	$w_8 = \epsilon p_8 \epsilon$	$m_8 = t_{15} t_{16}$	Case 1-a	
17.	$w_{11} = \epsilon p_{10} \epsilon$	$m_{11} = t_{19} t_{20}$	Case 2-Case a and 5-1	5.11
18.	$w_{10} = \epsilon p_{11} \epsilon$	$m_{10} = t_{21} t_{22}$	Case 2-Case a and 5-1	5.12
19.	$w_1 = \epsilon p_1 \epsilon$	$m_1 = t_1 t_2$	Case 1-a	
20.	$w_{2-3-4} = \epsilon p_2 \begin{matrix} p_2 \\ p_4 \end{matrix} p_2 \epsilon / p_3 \epsilon$	$m_{2-3-4} = t_3 t_6 t_7 t_4 / t_5 t_8$	Case 1-Case b	5.13

For these output words observed, the computed model is equal to the system.

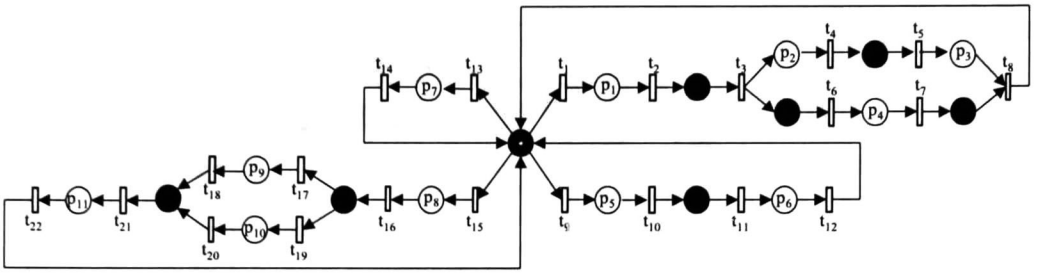


Figure 5.1: IPN model of the system.

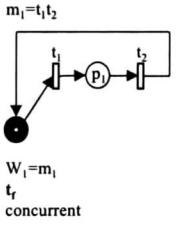


Figure 5.2: Computed model for the first observed ouput word w_1.

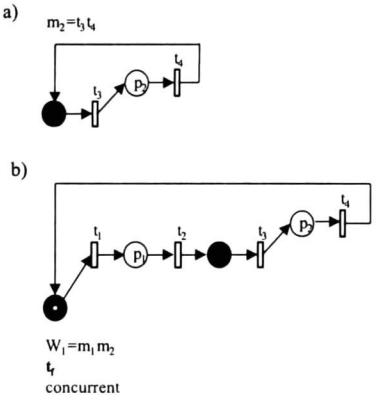


Figure 5.3: Computed model when the output word w_2 is observed.

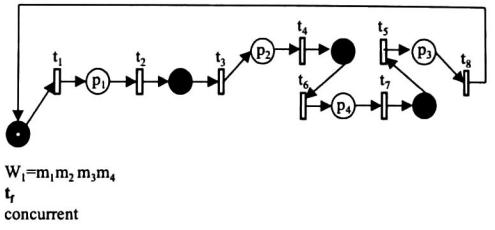


Figure 5.4: Computed model when the output words w_3 and w_4 were observed.

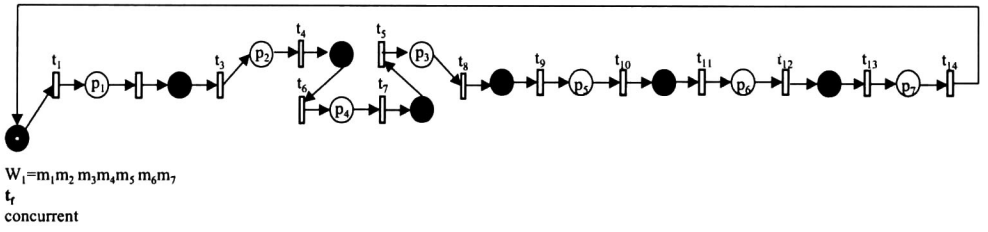


Figure 5.5: Computed model when the ouput words w_5, w_6 and w_7 were observed.

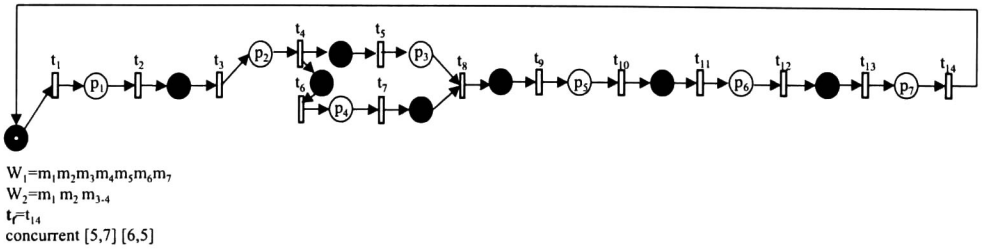


Figure 5.6: Computed model when the output words w_1, w_2 and w_{3-4} were observed.

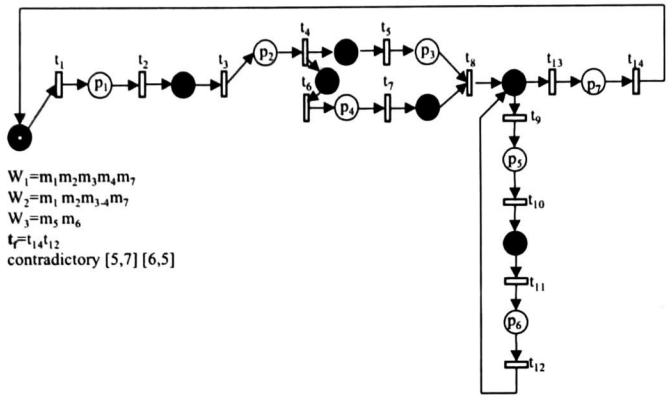


Figure 5.7: Computed model when the output word w_7 is observed.

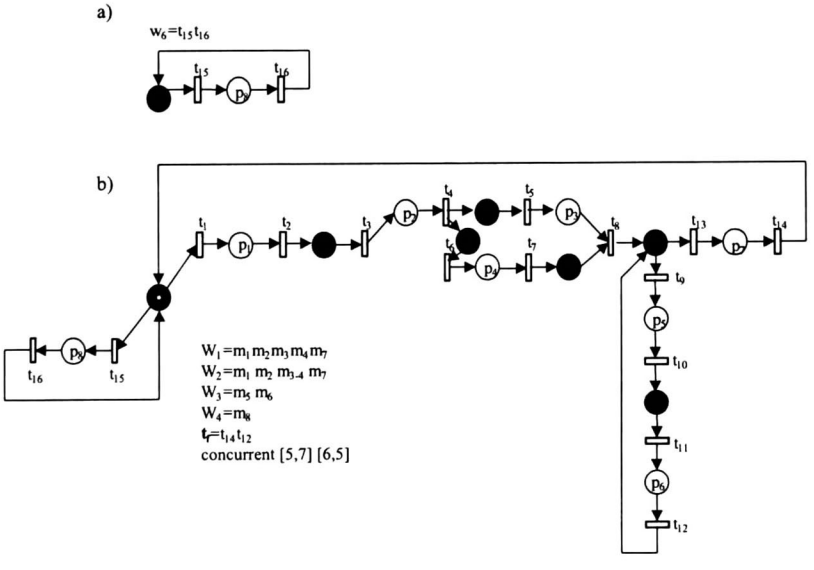


Figure 5.8: Computed model when the output word w_8 was observed.

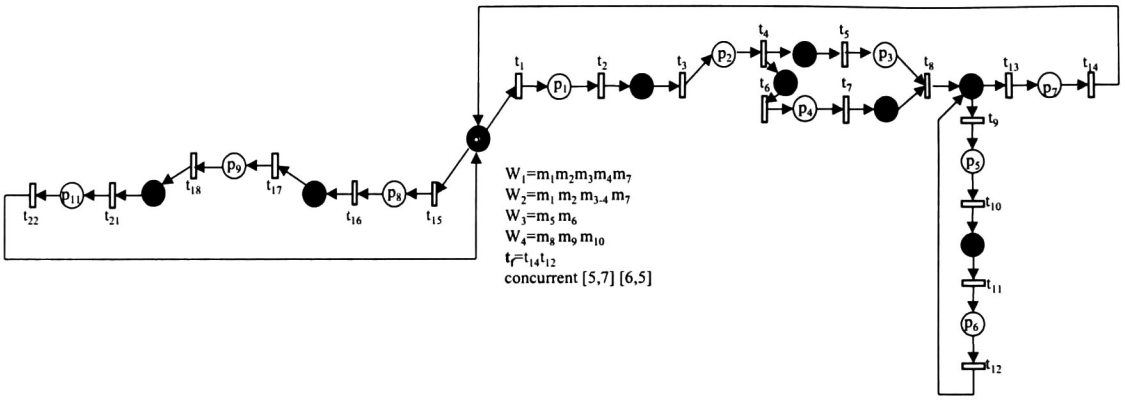


Figure 5.9: Computed model when the output words w_9 and w_{10} were observed.

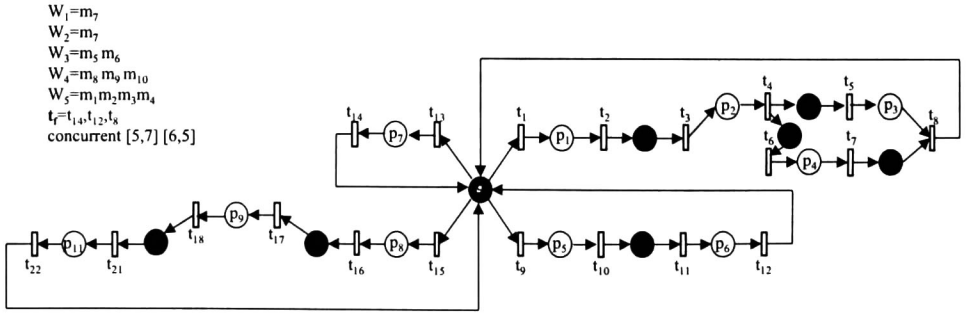


Figure 5.10: Computed model when the output word w_7 is observed.

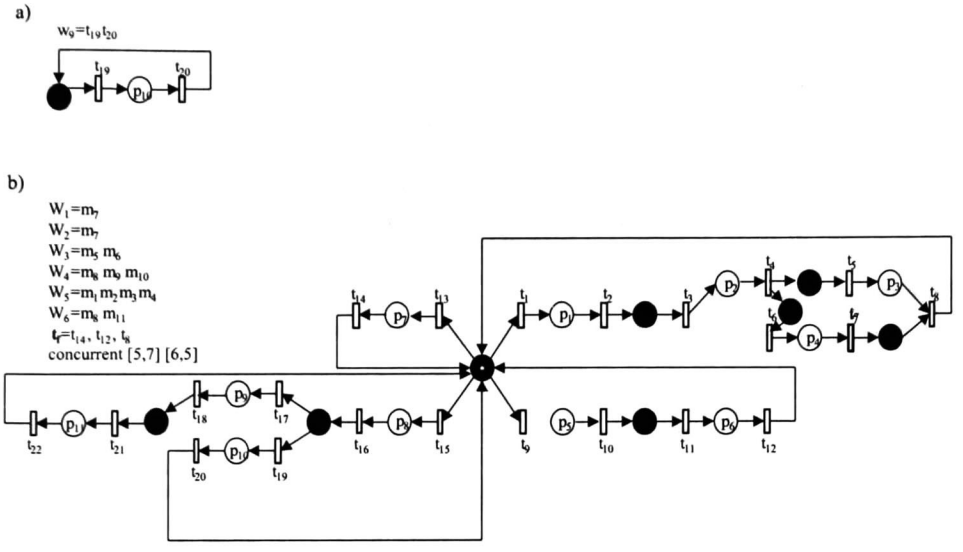


Figure 5.11: Computed model when the output word w_8 and w_{11} were observed.

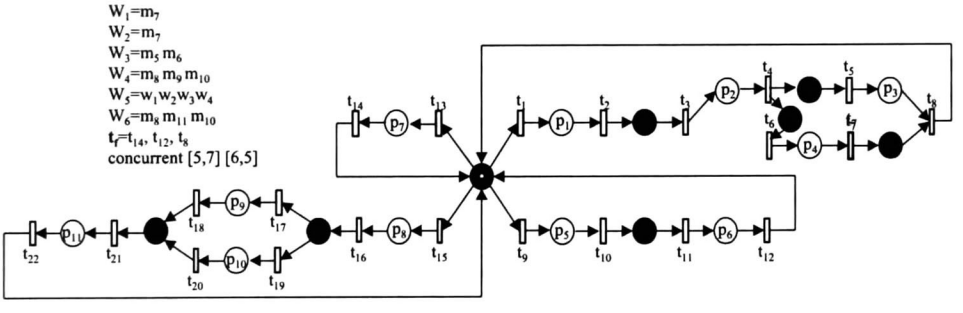


Figure 5.12: Computed model when the output word w_{10} was observed.

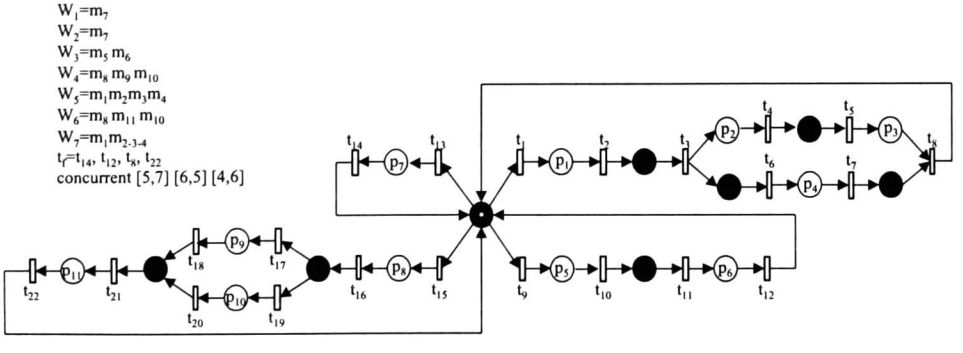


Figure 5.13: Computed model when the output words w_1 and w_{2-3-4} were observed.

5.4 Complexity analysis

Let Q be a system model and Q_i be the computed model from the observed behavior of Q. Next the complexity of each procedure needed to compute a new model are presented.

1. To compute a model first is needed to compute the current m-word w_n. The complexity of this procedure is linear in the number of the output symbols observed, due to each transition t_i of w_n is computed from any two consecutive output symbols $\varphi(M_{i-1})$ and $\varphi(M_i)$ as $t_i = \varphi(M_i) - \varphi(M_{i-1})$. Proposition 4.3.

2. Once w_n is computed it is classified as "new" or as "already computed" The procedure to detect if w_n is new or it is already computed is linear in the number of m-words computed.

3. When a new m-word is processed two cases are considered : Case 1) when all transitions of w_n or some of them are new transitions, and Case 2) when all transitions of w_n were already computed in another m-word w_m, however the order in which they were computed is different. In this case implies that there exists a m-word $w_m \neq w_n$ such that $\overrightarrow{w_m} = \overrightarrow{w_n}$ indicating that there exists concurrent transitions. The procedure to detect if w_n is of Case 1 or Case 2 is linear in $\sum_{\forall w_i \in \mathcal{T}} |T_i|$ where $|T_i|$ is the number of transitions of the m-word w_i.

   Case 1: In this case it is computed the associated t-component of w_n, forming NDep $[t_i, t_j]$ between consecutive transitions t_i and t_j of w_n such that $[t_i, t_j]$ does not belong to $Dep(Q_i)$. This NDep can be computed using a new non measurable place or using an already non measurable place. A new non measurable place will be added in the case that there exists not a NDep $[t_i, t_k]$ or a NDep $[t_k, t_j]$ in $Dep(Q_i)$ such that t_i and t_j are not independent transitions. Computing the NDep $[t_i, t_j]$ is an unitary operation.

   (a) Detecting if $[t_i, t_j]$ does not belong to $Dep(Q_i)$ has complexity linear in the number of dependencies (MDep and NDep) of Q_i.

   (b) Detecting that t_i and t_j are not independent transition. To detect that t_i and t_j are independent transitions it is searched that there exists a t-semiflow in which t_i has occurred but not t_j and also find out a t-semiflow in which t_j has occurred but not t_i. The complexity of this procedure is linear in the order of $\sum_{\forall w_i \in \mathcal{T}} |T_i|$ where $|T_i|$ is the number of transitions of the m-word w_i and $\mathcal{T}$ is the set of all computed m-words.

   (c) Detect that if t_i has an output non measurable place or if t_j has an input non measurable place. This procedure consists on find out a NDep $[t_i, t_k]$ or a NDep $[t_k, t_j]$ in $Dep^u(Q_i)$. Its complexity is linear in the number of NDep in $Dep^u(Q_i)$.

   Case 2: In this case it is needed to remove the wrong NDep formed between concurrent transitions. The procedure to remove wrong NDep is an unitary operation. Let t_i and t_j be consecutive transition in w_n, the procedure to detect the wrong NDep consist in determine if there exists a NDep $[t_x, t_i]$ or $[t_y, t_j]$ such that t_x and or t_y occurs after t_i and t_j respectively in w_n. This procedure is linear in the number of NDep in $Dep^u(Q_i)$.

In both cases w_n need to be concatenated with the previous m-word w_{n-1} to preserve the firing order of the m-words. Since in the case 2 all transitions have been already computed then it is possible that w_{n-1} and w_n are already concatenated. Then to concatenate w_{n-1} with w_n it is searched that:

(a) the NDep $[t_j, t_m]$ does not exist in $Dep(Q_i)$, where t_j is the last transition of w_{n-1} and t_m is the first transition of w_n. This searching is of linear complexity in the number of NDep in $Dep^u(Q_i)$. If the NDep $[t_j, t_m]$ does not exists in Q_i then the m-word w_{n-1} and

(b) the last transition of w_{n-1} is not marked as t_F. This procedure is linear in the number of marked transition as t_F.

4. If w_n is an already computed m-word then it is needed to verify that the computed model Q_i preserve the order in which the m-words are computed. Let $w_{n-1} = t_i \cdots t_j$ and $w_n = t_m \cdots t_n$ be the current and the previous computed m-words. Since w_n is an already computed m-word then it is possible that it occurs in different order than the order in which it was previously computed, then to verify that in Q_i w_n can occur as it has been observed it is searched the relation with the previous computed m-word w_{n-1} :

   (a) If there exists the NDep $[t_j, t_m]$ in $Dep^u(Q_i)$ then implies that w_n has occurred in the same order than its previous computation and hence the model Q_i remains without change. The procedure to find out the NDep $[t_j, t_m]$ has linear complexity in the number of NDep in $Dep^u(Q_i)$.

   (b) However if there exists not a NDep $[t_j, t_m]$ in Q_i, then implies that now w_n has occurred in different order than its previous computation. To compute the NDep $[t_j, t_m]$ to preserve the behavior of Q in Q_i, it is needed to merge $t_j^\bullet$ and ${}^\bullet t_m$. This procedure is linear in the number of the NDep in $Dep^u(Qi)$.

From previous analysis can be stated that the procedure to update a model is of polynomial complexity given by the number of transitions of the current computed m-word, the number of m-words computed and also in the number of computed dependencies. However to compute an actual model Q_n for a system model Q it is required that:

1. all the m-words of Q be computed and

2. the m-words be computed in a right order to update correctly the current model.

This requirements makes that the identification problem have the form of a non deterministic polynomial (NP) problem, since even the updating of a model is made in polynomial time, the process to obtain the good m-words in the right sequences in the right order can be viewed as a non deterministic process, this fact is illustrated on figure.5.14.

5.5 Completeness of the identification algorithm

This section is devoted to the study of the completeness of the identification procedure presented in this work. The strategy adopted for updating a model Q_i depends on how the current computed m-word w_n is related with the previous computed m-word w_{n-1} in other evolutions of the system model Q. The number of relationships

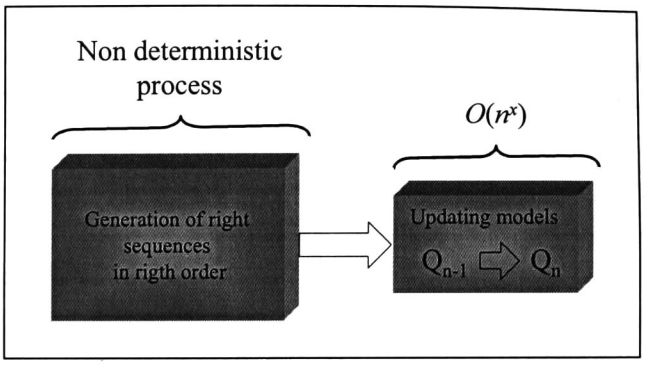


Figure 5.14: Complexity of the identification problem.

between w_n and w_{n-1} is reduced and there exists a procedure to obtain a new model Q_{i+1}. Thus it is possible to state the following proposition.

Proposition 5.1 *The identification algorithm of theorem 5.1 is complete.*

Proof. Let Q be a system model, Q_i be the computed model and w_{n-1} and w_n be the previous and the current computed m-words. The possible cases in which w_n may be related with w_{n-1} can be determined and there exists a procedure to update Q_i when w_n is computed according to previous evolutions of Q. The table depicted on figure 5.15 summarizes the relations between w_{n-1} and w_n.

1. w_n is a new m-word					
	Computed		Example	Procedure	Class of non measurable place
w_{n-1}	a. Once or consecutively		$w_a w_b$	Concatenation 4.a	B.i or B.ii.b
	b. Several times		$w_a w_b w_a w_c$	Selection 1	B.ii.a
	c. Consecutively		$w_a w_a w_b$	Concatenation 4.b	B.ii.b
2. w_n is an already computed m-word					
	Computed		Example	Procedure	Class of non measurable place
w_{n-1}	a. Once		$w_a w_b w_a w_c w_b$	Attribution 2	B.ii.a
	b. Several times	i. In the same order	$w_a w_b w_a w_b$	----	
		ii. In another order	$w_a w_b w_c w_a w_c$	Merging 3.b	B.ii.b
	c. Consecutively		$w_a w_b w_b w_a$	---	
3. $w_{n-1} = w_n$			$w_a w_a$	Merging 3.a	B.ii.b

Figure 5.15: Situations of w_{n-1} with respect to w_n.

Since it is considered the order in which the m-words can be computed, the non measurable places related with these situations are the non measurable places of Class B, indeed the procedures stated in table 5.15 are the procedures of algorithm 4.4. The non measurable places of Class A are computed when a new m-word is detected. ■

5.6 Obtaining timed models

This section deals with the computation of timed models for DES in which all places of the system model Q are measurable. Here, it is presented the procedure to compute the durations of the activities of the system. Each marking M_i has associated a time α_i being the instant in which M_i is detected, i.e. the firing instant of the transition t_j that leads to M_i. Now, an output symbol ω_0 will be considered as the pair $[\varphi(M_i), \alpha_i]$, where $\varphi(M_i)$ is the output generated by the system and α_i is the instant in which it was generated.
The time elapse α_{p_k} of each place p_k will be computed as:

$$\alpha_{p_k} = \alpha^f_{p_k} - \alpha^0_{p_k}$$

where $\alpha^0_{p_k}$ is the instant in which p_k is marked and $\alpha^f_{p_k}$ is the time in which p_k is unmarked. Then $\alpha^0_{p_k} = \alpha_i$ if p_k is marked in M_i and $\alpha^f_{p_k} = \alpha_j$ if p_k is unmarked in M_j. This time elapse is the duration of the activity described by the place p_k. Each computed time α_{p_k} is stored in the set d_{p_k}, this set will contain all the durations computed for the place p_k.
The time associated to each place p_k can be computed as the average of the times in d_{p_k} or can be computed using a distribution function.
The next algorithm computes an IPN for a DES from the observation of its output symbols and computes the set of durations t_{p_k} associated to each place p_k.

Algorithm 5.2 *Computing a timed IPN model for a DES.*

Input: $C_{Q_{n-1}}$, t_{p_k}
Output: The updated timed model Q_n

---

1. $C_{Q_n} = C_{Q_{n-1}}$
2. $\varphi(M) = \varphi(M_0)$, $i = 1$
3. for $k = 1 \cdots n$ (n is the number of places)
   (a) If $\varphi(M(p_k)) = 1$ then $\alpha^0_{p_k} = \alpha_0$
4. **while** there exists a marking $\varphi(M_i) \neq \varphi(M)$ **until** $\varphi(M_i) = \varphi(M_0)$
   For $k = 1 \cdots n$ (n is the number of places)
   (a) If $\varphi(M_{i-1}(p_k)) = 0$ and $\varphi(M_i(p_k)) = 1$ then $\alpha^0_{p_k} = \alpha_i$
   (b) If $\varphi(M_{i-1}(p_k)) = 1$ and $\varphi(M_i(p_k)) = 0$ then
      i. $\alpha^f_{p_k} = \alpha_i$
      ii. $\alpha_{p_k} = \alpha^f_{p_k} - \alpha^0_{p_k}$
      iii. add αp_k to d_{p_k}
      iv. reset $\alpha^0_{p_k}$, $\alpha^f_{p_k}$ and α_{p_k}

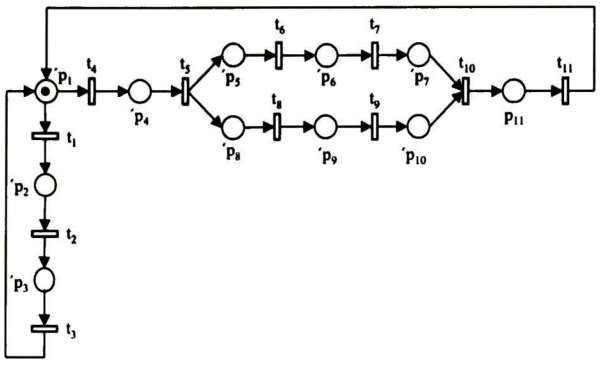


Figure 5.16: System model

(c) Computing a column of the incidence matrix $C(\bullet, t_k) = \varphi(M_i) - \varphi(M)$

(d) If $C(\bullet, t_k)$ does not exists in C_Q then add it to C_Q

(e) $\varphi(M) = \varphi(M_i)$

(f) $i = i + 1$

Theorem 5.2 *Let Q be a fully observable IPN and Q_{n-1} be the proposed model for Q. If w_n is the current m-word then a model Q_n can be built using the algorithm 5.2 such that $f(Q, Q_n) \leq f(Q, Q_{n-1})$.*

Proof. Since Q is a completely measurable IPN, then Q has not non measurable places, hence the term $|Dep^u(Q) - Dep^u(Q_n)|$ is zero. Thus the error equation 4.1 is reduced to $f(Q, Q_n) = |\{\varphi C_Q\} - \{\varphi C_{Q_n}\}|$. Since the algorithm in the step 4.b computes a column of φC_{Q_n} then $|\{\varphi C_Q\} - \{\varphi C_{Q_n}\}| < |\{\varphi C_Q\} - \{\varphi C_{Q_{n-1}}\}|$ in the case when $C(\bullet, t_k)$ is a new transition or $|\{\varphi C_Q\} - \{\varphi C_{Q_n}\}| = |\{\varphi C_Q\} - \{\varphi C_{Q_{n-1}}\}|$ in the case when $C(\bullet, t_k)$ is an already computed transition, thus it is fulfilled that $f(Q, Q_n) \leq f(Q, Q_{n-1})$. ■

The algorithm 5.2 can be used on-line with the system; its programming is straightforward quickly. The complexity of this algorithm is linear.

Notice that every time a t-semiflow is detected (step 4 of the algorithm) a new model Q_i is computed.

Example. Consider Q the system model depicted on figure 5.16 to be identified. The identification steps when the t-semiflow $X_1 = [\ 1\ \ 1\ \ 1\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ \ 0\ ]^T$ is detected, are presented in figure 5.17. At this moment the model Q_1 depicted in figure 5.18 is identified.

The time elapses computed for the places in the occurrence of X_1 are $\alpha_{p_1} = 2$, $\alpha_{p_1} = 3$ and $\alpha_{p_1} = 4$.

The identification steps when the t-semiflow $X_2 = [\ 0\ \ 0\ \ 0\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ ]^T$ is detected, are summarized in figures 5.19, 5.20 and 5.21.

The structure of the IPN depicted on figure 5.16 has been computed since all its transitions were computed. The associated durations of the places in the occurrence of X_1 and X_2 are the following $d_{p_1} = \{2, 5\}$, $d_{p_2} = \{3\}$, $d_{p_3} = \{4\}$, $d_{p_4} = \{6\}$, $d_{p_5} = \{2\}$, $d_{p_6} = \{8\}$, $d_{p_7} = \{3\}$, $d_{p_8} = \{10\}$, $d_{p_9} = \{4\}$, $d_{p_{10}} = \{3\}$ and $d_{p_{11}} = \{3\}$.

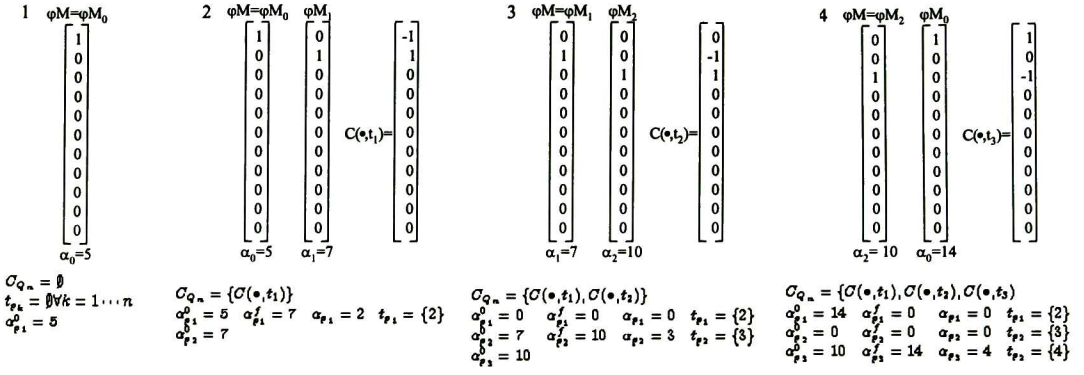


Figure 5.17: Evaluation of the first observed t-semiflow.

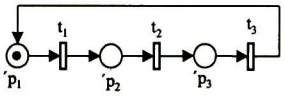


Figure 5.18: Computed model Q_1.

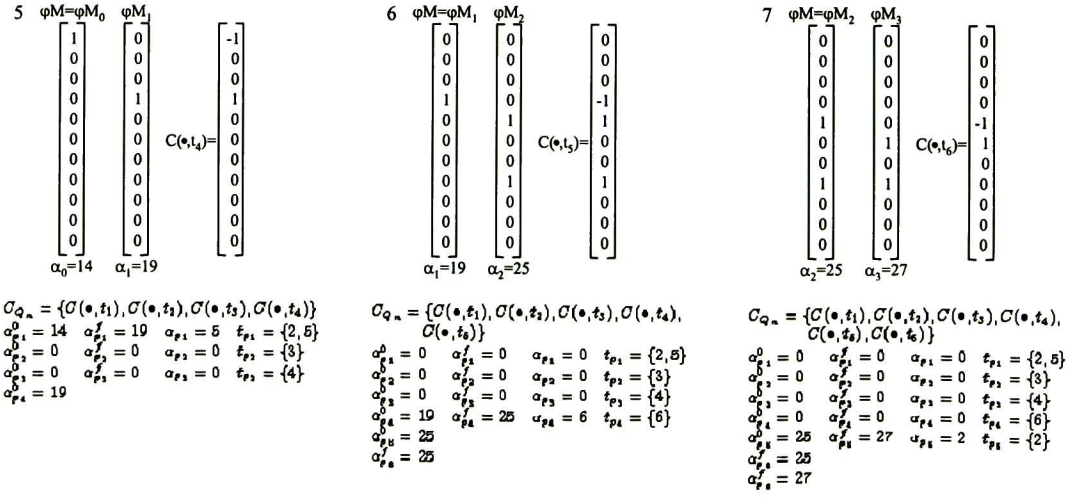


Figure 5.19: Identification steps when the t-semiflow X_2 is evaluated.

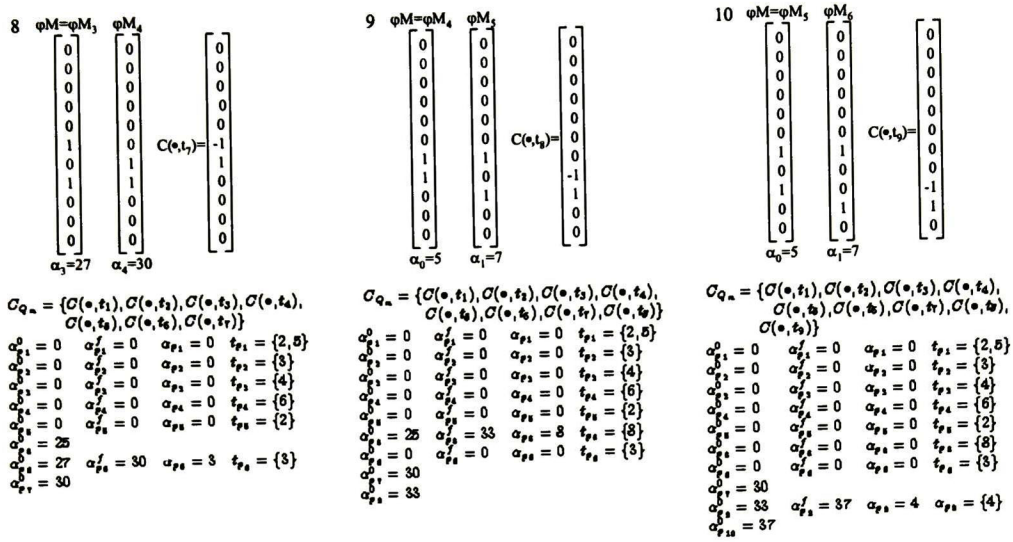


Figure 5.20: Identification steps.

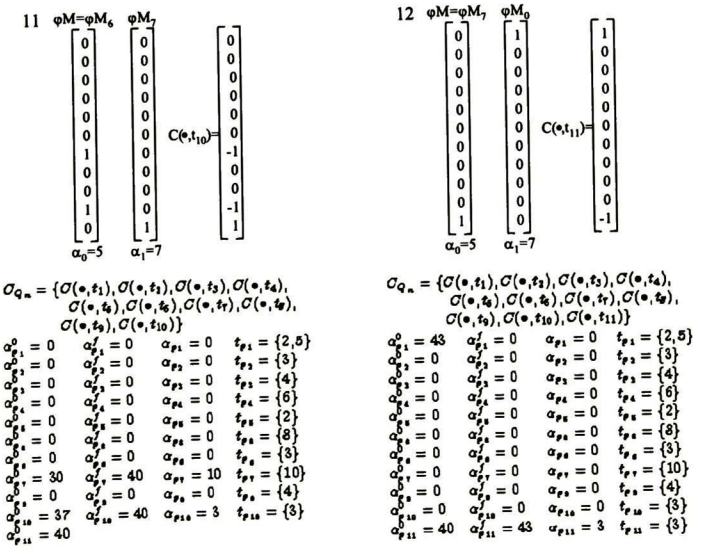


Figure 5.21: Identification steps.

Chapter 6

Identifiable DES

This chapter studies the conditions for a system to be identifiable. It includes a description of the structures which cannot be computed using the proposed algorithms, then it is analyzed when non measurable places cannot be computed. A characterization of the sequences allowing to compute all the dependencies of the model is proposed. Finally, an analysis of the influence of event detectability property for identification is presented.

6.1 Introduction

The procedure presented in the previous chapter allows obtaining a Petri net model for an unknown DES. Even if the hypothesis of event detectability is not fulfilled (actually it cannot be determined on an actual unknown system), the obtained model represents the observed behavior of the system; the on-line approach supports the updating of the inferred model if a "new" behavior is observed. Nevertheless it is interesting to analyze the conditions in which a DES can be fully identified by the measurement of its outputs; besides to provide a better understanding of the problem, some results can be useful for addressing other problems.
First at all one must to state which specific structures cannot be detected, such as the self-loops and others that concern the placement of the non measurable places. Furthermore it is important to know the characteristics of the output sequences that guarantee that the whole behavior of a system has been exhibited allowing to compute φC matrix and to infer γC matrix; one of them is event-detectability by the output, a starting hypothesis.

6.2 Non computable structures

This section presents several structures that cannot be computed using the proposed algorithms and discuss this characteristic. First, specific structures are analyzed, then the analysis of implicit non measurable places is presented.

6.2.1 Specific structures

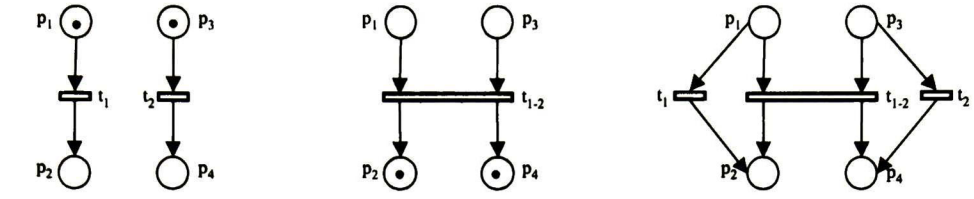
◊ Simultaneous transition firing

Claim 6.1 *Let Q be an IPN and t_i and t_j be any transitions of Q that can be enabled simultaneously. If t_i and t_j are fired simultaneously then their firing is detected as the firing of only one transition.*

Proof. As presented in proposition 4.2, a transition t_k is computed as the difference of consecutive markings $M_{k+1} - M_k$, however notice that a change of marking can be generated by the firing of several transitions. Thus, the difference of consecutive markings $M_{k+1} - M_k$ only represents the relation of which places must to be dismarked and which places must to be marked to reach M_{k+1} from M_k. Hence $M_{k+1} - M_k$ is the sum of the columns of C matrix representing the transitions fired from M_k leading to reach M_{k+1}. In the case that M_{k+1} is reached with the firing of only one transition t_k then $M_{k+1} - M_k$ represents the column k of C matrix representing t_k. In the identification procedure it is not possible to determine if a change of marking is generated by the firing of one or several transitions because the system is unknown and by difference of consecutive markings the simultaneous firing of t_i and t_j ($t_i||t_j$) will be considered as the firing of only one transition. ■
Previous proposition is illustrated in figure 6.1. If the transitions t_1 and t_2 are fired simultaneously in the PN depicted on figure 6.1.a, then will be computed the transition t_{1-2} of the PN depicted on figure 6.1.b. The computed model considering also that t_1 and t_2 are not fired simultaneously is depicted on figure 6.1.c.

◊ Self loops

Claim 6.2 *Let Q be an IPN. A self-loop formed with the place p_i and the transition t_j cannot be computed.*

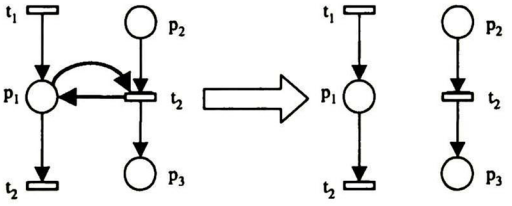


a) Simultaneous enabling b) Computed transition from simultaneuos firing c) Computed model

Figure 6.1: Simultaneous firing.

Proof. The proof follows from the fact that a self-loop cannot be described by the incidence matrix of a PN. An element c_{ij} of C is computed as $c_{ij} = c_{ij}^{+} - c_{ij}^{-}$ such that c_{ij}^{+} is the weight of the arc from t_j to p_i and c_{ij}^{-} is the weight of the arc to the transition t_j from the place p_i. Since p_i is an output place of t_i then $c_{ij}^{+} = 1$ and since t_j is an input transition of p_i then $c_{ij}^{-} = 1$, hence $c_{ij} = c_{ij}^{+} - c_{ij}^{-} = 0$ and then the relationship between p_i and t_j and viceverse cannot be represented in the incidence matrix of Q and hence it is not possible compute a self-loop. ■

The situation stated in previous proposition is illustrated in figure 6.2; the dependency $[t_2, t_2]$ is not computed.



a) Self-loop b) Computed model

Figure 6.2: PN with a self-loop.

◊ Mutual exclusion

The PN depicted on figure 6.3 is an example of a mutual exclusion scheme. In such model the processes 1 and 2 are executed in parallel sharing the resource r, the place p_r represents this shared resource. The activities a and b represented by the places p_a and p_b respectively, require the resource r, once the transition t_a is fired the activity a is performed and the activity b can be performed when the activity a is finished and the resource r is released. A similar situation can be found when the transition t_b is fired, i.e. when r is assigned to the activity b. Notice then that the activities a and b cannot be executed simultaneously. This structure can be computed using the proposed algorithm when p_r is a measurable place. Unfortunately when p_r is a non measurable place it cannot be computed.

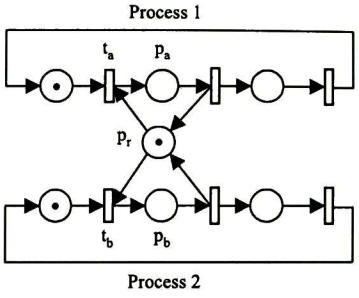


Figure 6.3: Mutual exclusion scheme.

Claim 6.3 *Let Q be an IPN and let p_r be a place of Q describing a shared resource. If p_r is a non measurable place then it cannot be computed.*

Proof. The places describing shared resources are included in mutual exclusions schemes. Hence, consider that Q describes two concurrent processes: proc A and proc B which share the resource r described by the place p_r. Consider also that $T_A = \{t_{a_1}, t_{a_2}, t_{a_3}, \cdots, t_{a_m}\}$ are the transitions of proc A that cannot be executed in parallel with respect to the transitions of proc B, while $T_B = \{t_{b_1}, t_{b_2}, t_{b_3}, \cdots, t_{b_n}\}$ are the transitions of proc B that cannot be executed in parallel with respect to the transitions of the proc A, because its related activities are those requiring the resource r. Since the resource r can be assigned to each process without any order then in a evolution σ_1 when r is assigned to proc A before to proc B of Q the transitions $t_{a_1}, t_{a_2}, t_{a_3}, \cdots, t_{a_m}$ of proc A occur before the transitions $t_{b_1}, t_{b_2}, t_{b_3}, \cdots, t_{b_n}$ of proc B, while in another evolution σ_2 of Q, r could be assigned to proc B before to proc A and hence in σ_2 the transitions $t_{b_1}, t_{b_2}, t_{b_3}, \cdots, t_{b_n}$ of proc B occur before to the transitions $t_{a_1}, t_{a_2}, t_{a_3}, \cdots, t_{a_m}$ of proc A.
Notice that $\sigma_1 = \cdots t_{a_1} \cdots t_{a_m} \cdots t_{b_1} \cdots t_{b_n} \cdots$ and $\sigma_2 = \cdots t_{b_1} \cdots t_{b_n} \cdots t_{a_1} \cdots t_{a_m} \cdots$ and hence the place p_r cannot be computed because if a NDep is computed to preserve the firing order of the transitions of proc A before the transitions of proc B when σ_1 is detected then it will be removed with the detection of σ_2. ■

6.2.2 Implicit non measurable places

The identification problem is related with the inference of the non measurable places of the system model Q. This inference is made by sequencing consecutive transitions and/or consecutive m-words, when new information of the system is obtained; then it is needed to remove, merge, or add new non measurable places in order to allow the firing of all computed m-words. However not all the non measurable places of the system can be computed since the information of the measurable places is enough to preserve the order in which the transitions are computed in a m_word.
Consider the system model Q depicted on figure 6.4.a, notice that there exists two t-semiflows represented by the vectors $[\ 1\ \ 1\ \ 0\ \ 0\ \ 1\ ]^T$ and $[\ 0\ \ 0\ \ 1\ \ 1\ \ 1\ ]^T$; their associated t-components are depicted on figure 6.4.b.
The next situations show how this system is identified when different places on the net are assumed to be non measurable places.

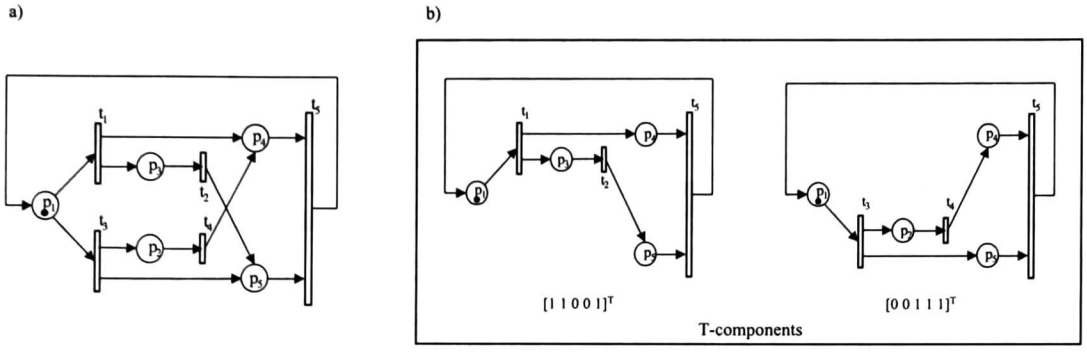


Figure 6.4: a) System model Q. b) T-components of Q

- *Case 1* Consider the system of figure 6.4.a, where the place p_5 is a non measurable place as depicted on figure 6.5.a. When the output word $w_1 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ is observed, the m-word $m_1 = t_1 t_2 t_5$ is computed using proposition 4.3. Using the identification theorem 5.1 the only NDep computed is $p_5 = [t_2, t_5]$ since for another pair of consecutive transitions t_i and t_j there exists a MDep $p_x = [t_i, t_j]$, i.e. $p_1 = [t_5, t_1]$ and $p_3 = [t_1, t_2]$. The computed model for this output word is depicted on figure 6.5.b. When the output word $w_2 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$ is observed, the $m-$word $m_2 = t_3 t_4 t_5$ is computed using proposition 4.3. Using the identification theorem 5.1 no one NDep is computed since for any two consecutive transitions t_i and t_j in the m-word m_2 there exists a MDep $p_x = [t_i, t_j]$, i.e., $p_1 = [t_5, t_3]$, $p_2 = [t_3, t_4]$ and $p_4 = [t_4, t_5]$. The computed model for these output words is depicted on figure 6.5.c.

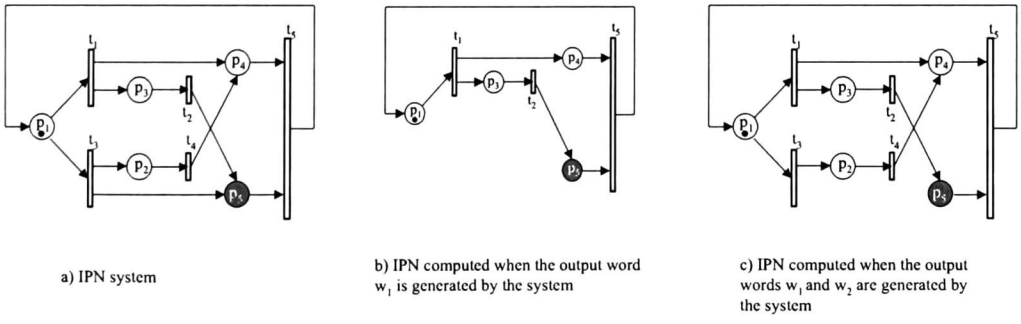
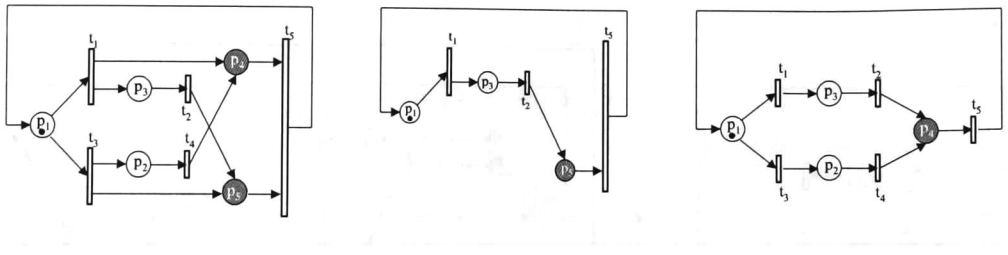


Figure 6.5: IPN system and the computed models when the ouput words w_1 and w_2 are generated by the system.



a) IPN system

b) IPN computed when the output word w_1 is generated by the system

c) IPN computed when the output words w_1 and w_2 are generated by the system.

Figure 6.6: IPN system and the computed models when the ouput words w_1 and w_2 are generated by the system.

The computed model for the system depicted on figure 6.5 is not live. In this case the place p_5 could not be computed correctly since there is no information to form the NDep $[t_3, t_5] = p_5$.

Case 2 Consider the same system 6.4.a where p_4 and p_5 are non measurable places as depicted on figure 6.6.a. When the output word $w_1 = \begin{bmatrix}1\\0\\0\end{bmatrix}\begin{bmatrix}0\\0\\1\end{bmatrix}\begin{bmatrix}0\\0\\0\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix}$ is observed, the m-word $m_1 = t_1t_2t_5$ is computed using proposition 4.3. Using the identification theorem 5.1 the only NDep computed is $p_5 = [t_2, t_5]$ since for another pair of consecutive transitions t_i and t_j there exists a MDep $p_x = [t_i, t_j]$, i.e. $p_1 = [t_5, t_1]$ and $p_3 = [t_1, t_2]$. The computed model for this output word is depicted on figure 6.6.b. When the output word $w_2 = \begin{bmatrix}1\\0\\0\end{bmatrix}\begin{bmatrix}0\\1\\0\end{bmatrix}\begin{bmatrix}0\\0\\0\end{bmatrix}\begin{bmatrix}1\\0\\0\end{bmatrix}$ is observed, the m-word $m_2 = t_3t_4t_5$ is computed using the proposition 4.3. Using the identification theorem 5.1 the only NDep computed is $p_4 = [t_4, t_5]$ since for another pair of consecutive transitions there exists a MDep, i.e. $p_1 = [t_5, t_3]$ and $p_2 = [t_3, t_4]$. However as t_5 is a shared transition by the two t-semiflows, then the non measurable places p_4 and p_5 must be the same by identification step 4.2 case 2. The computed model for this output word is depicted on figure 6.6.c. Notice that the arcs (t_1, p_4) and (t_3, p_5) could not be computed and hence the places p_4 and p_5 could not be identified correctly.

The computed model 6.6.c describes the same behavior as the target system depicted on figure 6.6.a, however this model is different to the system model Q, the difference can be found in the γC matrix of the system and the model.

In both cases the non measurable places could not be computed correctly since the dependencies formed by the measurable places is enough to preserve the order in which the transitions were computed. Notice that the non measurable places in the systems above are implicit places. This fact lead to the following proposition.

Proposition 6.1 *Let p_i be an implicit place of an IPN Q describing a DES, if p_i is a non measurable place then it cannot be computed.*

Proof. As stated in [53], if p_i is an implicit place implies that: 1) its marking can be computed from the marking of others places and 2) never is the place avoiding the firing of its output transitions. This last

condition implies that if there exists a dependency $[t_x, t_y] = p_i$ also there exists a dependence transition sequence $B = [t_x, t_r][t_r, t_s] \cdots [t_u, t_y]$ and t_y cannot be fired until the firing of all the transitions of B, hence the only NDep that could be computed related with t_x and t_y are $[t_x, t_r] = p_s$ and/or $[t_u, t_y] = p_r$ in the case where p_s and/or p_r be non measurable places, and then p_i cannot be computed. ■

Proposition 6.2 *A DES described by an IPN Q in which some transition need to be fired simultaneously or Q contains self-loops, non measurable places describing shared resources or implicit non measurable places cannot be fully identified.*

Proof. The proof follows from claims 6.1, 6.2 and 6.3 and proposition 6.1. ■

6.3 Properties of the output sequences

In order to compute a model Q_i for a system model Q it is needed that the observed behavior of Q be enough to detect all the transitions and places of Q.
Since the identification procedure is directly related with the computed transition sequences, in this section is presented which are the transition sequences needed to identify a DES.
As introduced in previous chapter the difficulty to identify a model system Q lays on the inference of its non measurable places, i.e. in the inference of its γC matrix. Basically, there exists two main problems that must to be solved to compute the γC matrix:

1. Distinguish which transitions are concurrent transitions.

   As introduced in chapter 4, to detect if any two transitions t_i and t_j are concurrent it is needed to compute transition sequences in which t_i and t_j occur in different order, in transition sequences generated by the same t-component of Q. If from the observed behavior of Q it is not possible to distinguish that two transitions t_i and t_j are concurrent then it is possible that there exists in the computed model Q_i a NDep $[t_i, t_j]$ that does not belong to Q.

2. Determine how the non measurable places of Q are connected.

   The procedures to infer the non measurable places show the importance to observe several occurrences of a same transition sequence (m-word) in order to extract the exact information of how the transitions of a m-word are connected and also to determine how the m-words are related with each other; this is due to the non measurable places are computed to preserve the firing order of the transitions in a m-word and also to preserve the occurrence order of the m-words in which they have occurred. If the non measurable places are not computed correctly then the t-components of Q are not rebuilt correctly causing that the computed model Q_i does not generate the complete behavior of Q.

Notice that the possibility to solve these problems depends on the transition sequences fired in Q : if it is possible to determine the dependency relation between places and transitions from the computed transition sequences, then a correct model Q_i for Q could be computed. Thus, if the entire behavior of Q is observed then a correct model for Q is computed since the entire dependency relation of Q could be detected. However it is possible to characterize a reduced set of transition sequences that allows to extract the entire information of the relation

dependency between transitions and places of Q. The analysis to find out these transition sequences is based on the t-component decomposition of Q.

6.3.1 Reduced set of transition sequences needed in the identification procedure

Let Q be a system model and let X be the set of all t-components of Q, in order to detect what is the relation between the transitions of Q (i.e. to identify what kind of dependency they are forming), the t-components of X must to be "stimulated" in two ways: 1) in a particular way: for determining how the transitions of a t-component X_i are related (if they are sequential or concurrent) and 2) in a general way: for detecting what is the relation of a t-component with respect to the others t-components of the system model Q.
The reduced set of transition sequences needed in the identification procedure allows to find out the entire dependency relation between the transitions of Q.

◊Individual analysis of a t-component

A t-component could be of sequential or concurrent nature, depending how its transitions are related. If a t-component X_i is of sequential nature then its transitions are dependent since they represent causal relationships. In this class of t-components there exists only one possible transition sequence in which the transitions can be fired. However if a t-component X_i is of concurrent nature, some transitions could occur in different order with respect to another transitions in X_i. In this class of t-components there could exist several transition sequences. As a t-component X_i is also a net, then it could be decomposed on its p-components (the conservative part of X_i). Let Y^i be the set of all p-components of X_i; by definition the transitions in a p-component are dependent i.e. that there exists a specific order in which they can occur, hence if the t-component X_i only has one p-component then X_i only can generate one transition sequence, however if X_i has more than one p-component then it could generate more than one transition sequence since the transition belonging to different p-components could be concurrent.
These two facts are illustrated below.

> **Example**. Consider the t-components of a system model depicted on figure 6.7. Let X_1 be the t-component composed by the transitions of the t-semiflow $\overrightarrow{X_1} = t_9t_{10}t_{13}t_{14}$, X_2 be the t-component composed by the transitions of the t-semiflow $\overrightarrow{X_2} = t_{11}t_{12}t_{13}t_{14}$, and X_3 be the t-component composed by the transitions of the t-semiflow $\overrightarrow{X_3} = t_1t_2t_3t_4t_5t_6t_7t_8$. These components are depicted separately on figures 6.8.a, 6.8.b and 6.8.c respectively.
>
> The t-components X_1 and X_2 have only one p-component Y_1^1 and Y_1^2, which are the same that their t-components X_1 and X_2, hence the only possible transition sequence generated by X_1 is $\sigma_1 = t_9t_{10}t_{13}t_{14}$, while the only transition sequence generated by X_2 is $\sigma_2 = t_{11}t_{12}t_{13}t_{14}$. The t-component X_3 has two p-components: Y_1^3 generated by the places of the p-semiflow $\overrightarrow{Y_1^3} = p_1p_2p_3p_4p_8p_9$ and $\overrightarrow{Y_2^3} = p_1p_5p_6p_7p_8p_9$, these p-components are depicted on figure 6.9. Notice then that X_3 can generate the following 6 transition sequences: $\sigma_3 = t_1t_2t_3t_4t_5t_6t_7t_8$, $\sigma_4 = t_1t_4t_5t_2t_3t_6t_7t_8$, $\sigma_5 = t_1t_2t_4t_3t_5t_6t_7t_8$, $\sigma_6 = t_1t_4t_2t_5t_3t_6t_7t_8$, $\sigma_7 = t_1t_2t_4t_5t_3t_6t_7t_8$, $\sigma_8 = t_1t_4t_2t_3t_5t_6t_7t_8$.

From previous example it is possible to conclude that

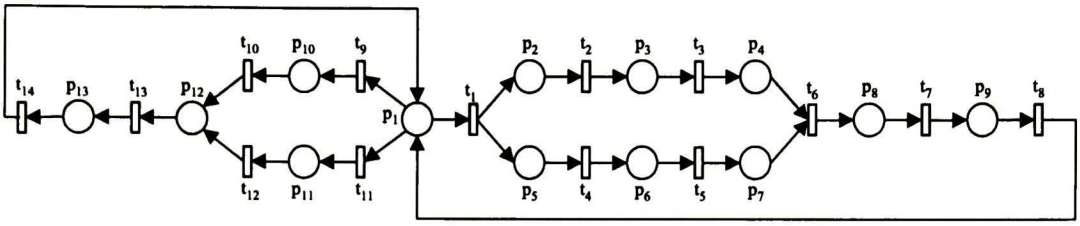


Figure 6.7: System model Q

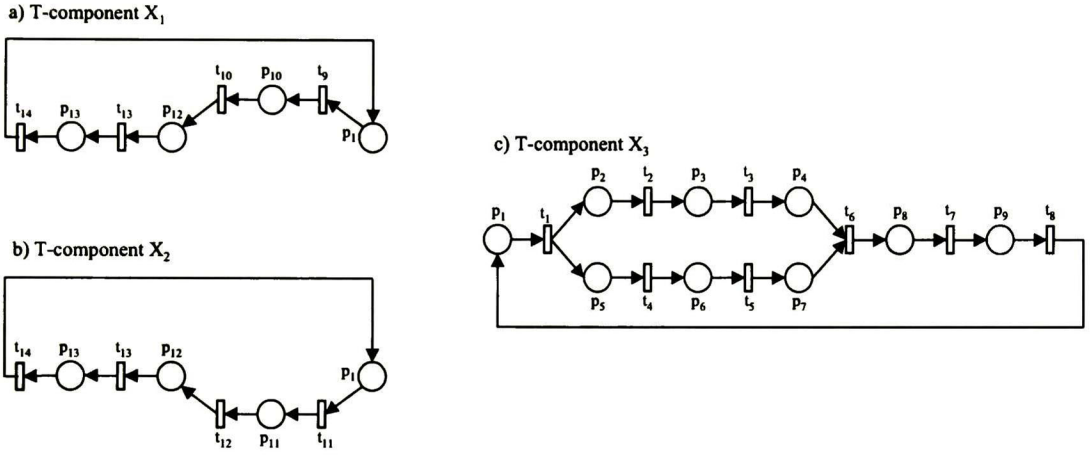


Figure 6.8: T-components of the system model Q depicted on figure 6.7.

1. the t-components having more than one p-component could generate more than one transition sequences and
2. the concurrent transitions belongs to different p-components

However if any two transitions belongs to different p-components it does not implies that they are concurrent transitions.

Example. Consider the t-component X_1 depicted on figure 6.10, it has tree p-components: Y_1^1 generated by the p-semiflow $\overrightarrow{Y_1^1} = p_1p_2p_3p_4$, Y_2^1 generated by the p-semiflow $\overrightarrow{Y_2^1} = p_5p_6p_7p_8$ and Y_3^1 generated by the p-semiflow $\overrightarrow{Y_3^1} = p_2p_3p_4p_6p_7p_8p_9p_{10}$. These p-components are depicted on figures 6.11.a, 6.11.b and 6.11.c respectively. Notice that the transitions t_1 and t_5 belong two different p-components Y_1^1 and Y_2^1 respectively, however they are not concurrent transitions since either t_1 occurs before t_5 or t_5 occurs before t_1 depending on the initial marking of X_1. Given the initial marking of this example t_1 occurs before t_5, hence X_1 can only generate the transition sequence $\sigma_1 = t_1t_2t_3t_4t_5$.

This is related with the fact that there exists a p-component in X_1 containing both transitions, this p-component is $\overrightarrow{Y_3^1}$ (figure 6.11.c).

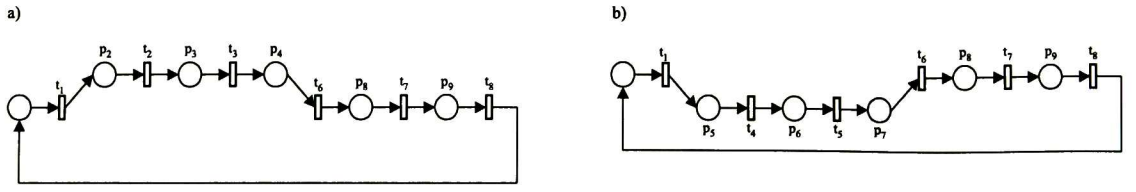


Figure 6.9: P-components of the t-component X_3 depicted on figure 6.8.c.

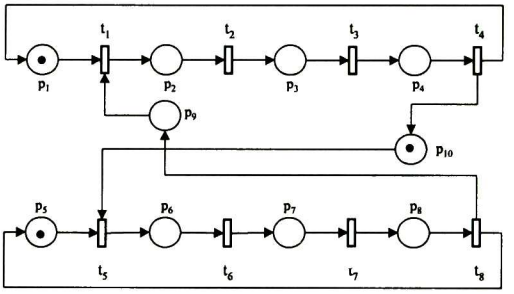


Figure 6.10: T-component X_1.

Then it is possible to state that two transitions t_i and t_j belonging to a t-component X_i are concurrent if they belong to different p-components and there exist not another p-component containing them; otherwise case t_i and t_j will be dependent transitions. Also, the number of transitions sequences generated by a t-component depend on how the transitions in a p-component are related with the transitions of another p-component of X_i. In order to detect that any two transitions t_i, t_j are concurrent there must exist at least two sequences σ_1 and σ_2 generated by the same t-component X_i such that t_i occurs before t_j in $\sigma_1$and t_j occurs before t_i in σ_2. Based on the fact that the concurrent transitions belong to different p-components of X_i, the next procedure is introduced to find out the transition sequences σ_1 and σ_2.

Algorithm 6.1 *Extracting the dependency relation between transitions of a same t-component* X_i

---

Inputs: The p-components of X_i

Output: The transition sequences σ_1 and σ_2

---

1. Enumerate the p-components of X_i
2. Compute σ_1 as the firing of the transitions of each p-component following the enumeration.
3. Compute σ_2 as the firing of the transitions of each p-component following the reverse order of the enumeration.
4. If any transition t_x cannot be fired in the *ith* p_component then fire the transitions of the p-component $i+1$ or $i-1$ (depending of which transition sequence is being computed) until t_x can be fired.

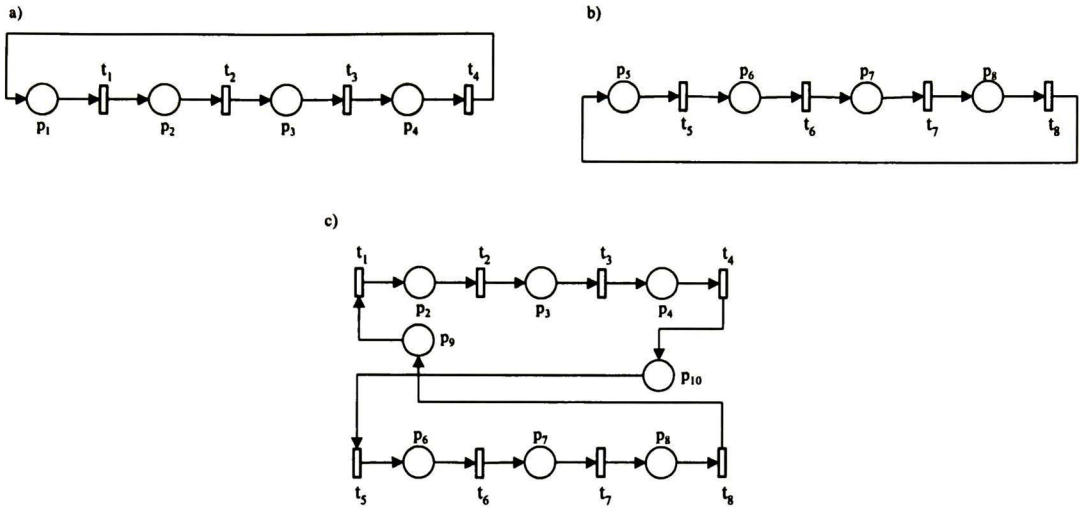


Figure 6.11: P-components of the t-component X_1 depicted on figure 6.10.

Notice that even the t-component X_i would have more than two p-components, the number of transition sequences computed using algorithm 6.1 always will be two: σ_1 and σ_2.

Example. Consider the t-component X_i depicted on figure 6.12. In order to compute the transition sequences σ_1 and σ_2, consider only the p-components Y_1^i, Y_2^i and Y_3^i generated by the p-semiflows: $\overrightarrow{Y_1^i} = p_1p_2p_3p_4$, $\overrightarrow{Y_2^i} = p_5p_6p_7p_8$ and $\overrightarrow{Y_3^i} = p_9p_{10}p_{11}p_{12}$ respectively. Assume that the enumeration of the p-components is the same as the order in which they are presented. As stated in step 2 of the algorithm 6.1 the transition sequence σ_1 is computed firing the transitions of Y_1^i before the transitions of Y_2^i, firing the transitions of Y_2^i before the transitions of Y_3^i and finally firing the transition of Y_3^iwhenever is possible. To fire t_1 it is needed to fire the transition t_4, then t_4 is the first transition fired, following with the transitions of Y_1^i the transitions t_1, t_2 and t_3 can be fired without the firing of transitions belonging to another p-component. The next transitions that need to be fired are the transitions of the p-component Y_2^i, since the transition t_4 was already fired then the transitions t_5 and t_6 are fired; notice that the transition t_5 can be fired because the transition t_3 was already fired. Next, the transitions of Y_3^i are fired: the transitions t_7, t_8, t_9 can be fired without restrictions; finally the transition t_{10} is fired. Hence the first transition sequence is $\sigma_1 = t_4t_1t_2t_3t_5t_6t_7t_8t_9t_{10}$.

The transition sequence σ_2 is computed as stated in the step 3 of the algorithm 6.1 firing the transitions of Y_3^i before the transitions of Y_2^i, firing the transitions of Y_2^i before the transitions of Y_1^i and finally firing the transitions of Y_1^i whenever is possible. The transitions t_7, t_8 and t_9 can be fired before any other transition belonging to other p-component, the next transitions to be fired are the transitions of Y_2^i: t_4 can be fired but not the transition t_5 since the transition t_3 must be fired before t_5, then the transitions t_1, t_2 and t_3 are fired before t_5; next the transition t_6 can be fired. Notice that all the transitions of Y_1^i were already fired then it is needed to fire the transition t_{10}. Hence, the second transition sequence is

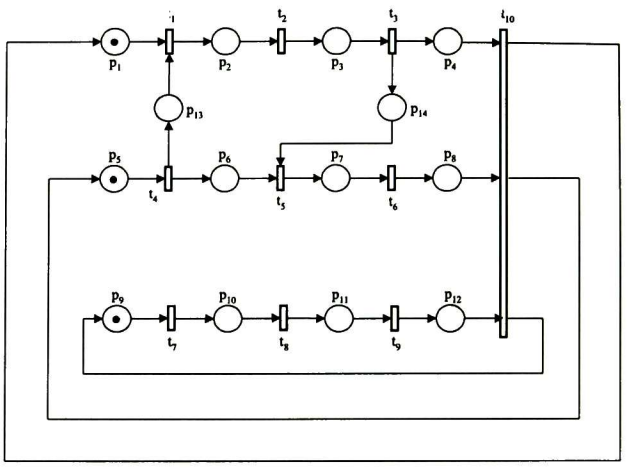


Figure 6.12: T-component X_i.

$\sigma_2 = t_7 t_8 t_9 t_4 t_1 t_2 t_3 t_5 t_6 t_{10}$.

Notice that in previous example to compute the transition sequences σ_1 and σ_2 were not used all the p-components of X_i The p-components selected were those forming the minimum set covering all the transitions of X_i. The other p-components are those describing the possible order in which the transitions can occur. Then the computation of these transition sequences is related to the problem to find out the minimum set of p-components covering all the transitions of X_i.

Since the transitions shared by p-components are sequential transitions it is needed to focus on the segments of a t-component in which the transitions could occur in different order.

The analysis to determine the minimum set of p-components covering all the transitions of a t-component X_i is based on the decomposition of a t-component X_i in its fork-joint subnets $t_j - t_k$ for each pair of fork-joint transitions t_j and t_k.

There could exists several kind of subnets generated by a pair of fork-joint transitions depending on how the external transition of this pair are related. Here will be considered those pairs such that if they are merged, all the transitions in the resulting net remain connected.

Example. Consider the t-component depicted on figure 6.13. The first pair of fork-joint transitions detected is the pair $t_3 - t_{18}$. Inside of this pair there exists the pairs: $t_5 - t_{10}$, $t_{12} - t_{17}$, $t_{12} - t_{10}$ and $t_3 - t_{17}$, however the only pair considered is the pair $t_5 - t_{10}$ since if the transition t_5 and t_{10} are merged the transitions remain connected. However for the others pairs this is not possible: consider the pair $t_{12} - t_{17}$ if they are merged, the transitions t_4 and t_{11} result in disconnected transitions. These facts are illustrated on figures 6.14.a and 6.14.b respectively.

A procedure to select the p-components containing disjoint transitions between the pairs fork-joint, is given below:

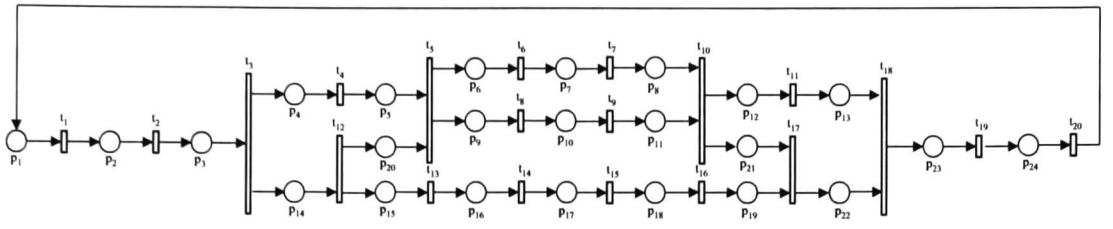


Figure 6.13: T-component X_i

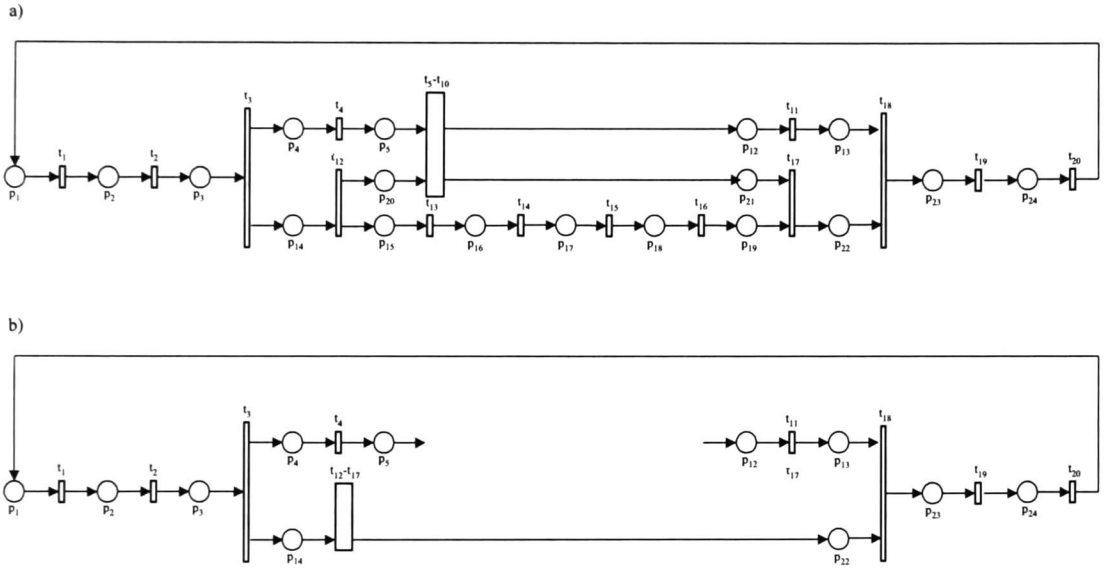


Figure 6.14: Merging fork-joint transitions.

Algorithm 6.2 *Computing the minimum set of p-components of a t-component X_i covering all transition of X_i*

---

1. Detect the subnets $t_i - t_j$ delimited by the fork-joint transitions t_i and t_j in X_i

2. If a subnet $t_i - t_j$ contains another subnet $t_x - t_y$ then assume that $t_x - t_y$ is just a transition inside of $t_i - t_j$ and select the p-components containing the disjoint transitions in $t_i - t_j$, next select from those p-components including $t_x - t_y$, the p-components in which the transitions of $t_x - t_y$ are disjoint.

   else select the p-components with disjoint transitions in $t_i - t_j$

Example. Consider again the t-component X_i depicted on figure 6.13. The transitions of each p-component of X_i are presented in the next table.

P-components	Transitions
Y_1^i	$t_1\ t_2\ t_3\ t_4\ t_5\ t_6\ t_7\ t_{10}\ t_{11}\ t_{18}\ t_{19}\ t_{20}$
Y_2^i	$t_1\ t_2\ t_3\ t_4\ t_5\ t_8\ t_9\ t_{10}\ t_{11}\ t_{18}\ t_{19}\ t_{20}$
Y_3^i	$t_1\ t_2\ t_3\ t_{12}\ t_{13}\ t_{14}\ t_{15}\ t_{16}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$
Y_4^i	$t_1\ t_2\ t_3\ t_4\ t_5\ t_6\ t_7\ t_{10}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$
Y_5^i	$t_1\ t_2\ t_3\ t_4\ t_5\ t_8\ t_9\ t_{10}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$
Y_6^i	$t_1\ t_2\ t_3\ t_{12}\ t_5\ t_6\ t_7\ t_{10}\ t_{11}\ t_{18}\ t_{19}\ t_{20}$
Y_7^i	$t_1\ t_2\ t_3\ t_{12}\ t_5\ t_8\ t_9\ t_{10}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$
Y_8^i	$t_1\ t_2\ t_3\ t_{12}\ t_5\ t_6\ t_7\ t_{10}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$
Y_9^i	$t_1\ t_2\ t_3\ t_{12}\ t_5\ t_8\ t_9\ t_{10}\ t_{17}\ t_{18}\ t_{19}\ t_{20}$

The subnet $t_3 - t_{18}$ is the first subnet detected, this subnet includes the subnet $t_5 - t_{10}$, then this transitions are merged resulting the t-component depicted on figure 6.14.a. The p-components having disjoint transitions in $t_3 - t_{18}$ are the p-components Y_1^i, Y_2^i and Y_3^i. Notice that Y_1^i and Y_2^i are same p-component in the t-component where the transitions t_5 and t_{10} were merged (figure 6.14.a) by this reason these two p-components are considered.

Considering this enumeration the transitions sequences σ_1 and σ_2 computed using algorithm 6.1 are the following:

$\sigma_1 = t_1t_2t_3t_4t_{12}t_5t_6t_7t_8t_9t_{10}t_{11}t_{13}t_{14}t_{15}t_{16}t_{17}t_{18}t_{19}t_{20}$

$\sigma_2 = t_1t_2t_3t_{12}t_{13}t_{14}t_{15}t_{16}t_4t_5t_8t_9t_6t_7t_{10}t_{17}t_{11}t_{18}t_{19}t_{20}$

Proposition 6.3 *Let X_i be a t-component of an IPN Q. The transition sequences σ_1 and σ_2 computed using algorithm 6.1 allows to determine which transitions are concurrent transitions.*

Proof. By construction of σ_1 and σ_2, if t_i and t_j are concurrent transition in X_i then in any of these transition sequences t_i occurs before t_j and in the other t_j occurs before t_i because they belong to different p-components of X_i. ■

◇Global analysis of t-components of the system model

In previous section it was studied the relation between transitions of a same t-component, now it is important to determine the correct relation between the t-components of a system model Q to compute correctly the places forming complex NDep in Q; the single NDep are computed correctly when two transitions occurs consecutively. During the identification process it could happen that:

1. some NDep formed with a non measurable place p_k of Q has not been computed, resulting a non measurable place computed incompletely in the model Q_i because even the NDep computed belong to Q this place is not a place of Q since are missed some 1s or -1s in the row i of the incidence matrix of Q_i.

2. the m-words which are actual t-semiflows of the system are concatenated with the previous m-word leading to compute a wrong non measurable place since it does not belong to Q.

The solution of both problems is related with the information provided by the system, i.e. with the computed transitions sequences. Consider that w_n is the current computed m-word. In the first case the missed NDep has not been computed because some m-words sharing transitions with m-words already computed have not

been detected. Then it is needed to compute the transition sequence related with the missed m-words. While in the second case it is needed that in a posterior evolution of Q, w_n occurs before w_{n-1} to compute the NDep indicating that after w_{n-1}, w_n can occur, and also that w_n can occur without the occurrence of w_{n-1} since w_n is an actual t-semiflow of Q.

The non measurable places related with these problems belong to the class B.ii.a and B.ii.b respectively.

The reduced set of transition sequences needed to identify the non measurable places of the classes A.i, A.ii and B.i is formed by sequences in which each transition is fired once. However for the non measurable places of classes B.ii.a and B.ii.b, the reduced set of transition sequences required to identify such places is that in which is enough to compute all the t-components of Q.

Considering X the set of t-components of Q, one transition sequence allowing the detection of the t-components could be: $\overrightarrow{X_1}\overrightarrow{X_2}\cdots\overrightarrow{X_n}\overrightarrow{X_n}\overrightarrow{X_{n-1}}\cdots\overrightarrow{X_2}\overrightarrow{X_1}$, where each X_i is a t-component of Q. Notice that in this sequence each t-component evolves in different order with respect to the others t-components of Q.

Example. Consider the system model Q depicted on figure 6.15. Since the t-semiflows of Q are the same as its m-words, then a reduced transition sequence $\overrightarrow{X_1}\overrightarrow{X_2}\cdots\overrightarrow{X_n}\overrightarrow{X_n}\overrightarrow{X_{n-1}}\cdots\overrightarrow{X_2}\overrightarrow{X_1}$ of Q is $w_1w_2w_3w_4w_4w_3w_2w_1$. On figures 6.16, 6.17, 6.18 and 6.19 are depicted the models computed when the m-words of in the reduced transition sequence are computed. The model of Q is computed after the m_word w_2 is already computed.

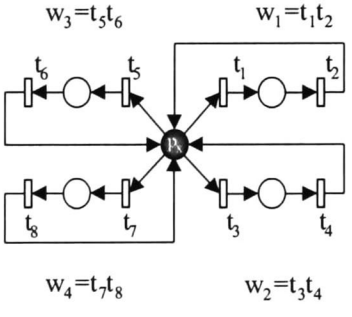


Figure 6.15: System model Q.

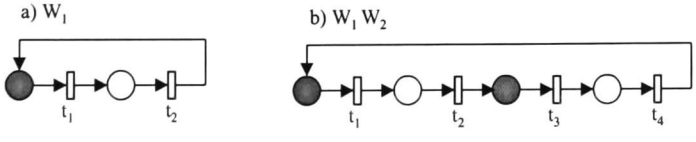


Figure 6.16: Computed models when the m_words w_1 and w_2 were computed.

- The reduced transition sequence is computed with the following algorithm.

Algorithm 6.3 *Computing the reduced set of transition sequences needed to identify a system model Q*

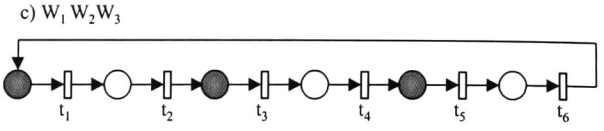


Figure 6.17: Computed model when the m_word w_3 is computed.

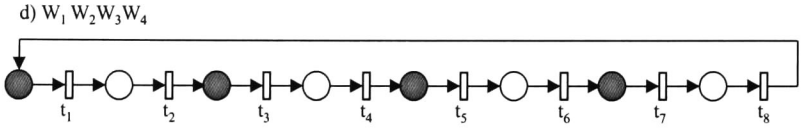


Figure 6.18: Model computed when the m_word w_4 is computed.

Input: The set of t-components of Q.

Output: The reduced set of transition sequences needed to identify a model system Q

---

1. Compute the transition sequences σ_1^i and σ_2^i using algorithm 6.1 for each t-component X_i of Q.
2. Then the sequence $\overrightarrow{X_1}\overrightarrow{X_2}\cdots\overrightarrow{X_n}\overrightarrow{X_n}\overrightarrow{X_{n-1}}\cdots\overrightarrow{X_2}\overrightarrow{X_1}$ will be equal to $\sigma_1^1\sigma_1^2\cdots\sigma_1^n\sigma_2^n\sigma_2^{n-1}\cdots\sigma_2^2\sigma_2^1$

Then, the reduced set of transition sequences needed to identify a system model denoted as $TSeq$ is

$$TSeq = \sigma_1^1\sigma_1^2\cdots\sigma_1^n\sigma_2^n\sigma_2^{n-1}\cdots\sigma_2^2\sigma_2^1$$

This transition sequence provides the needed information to detect the concurrent transitions and the non measurable places.

Proposition 6.4 *Let Q be a system model, the set TSeq allows to determine the relation of the transitions of each t-component X_i of Q and also the inference of the non measurable places.*

Proof. Since in $TSeq$ are considered the sequences σ_1^i and σ_2^i of each t-component X_i then the relation between transitions of each t-component can be determined, this follows from proposition 6.3. Notice also that in TSeq, each t-component occurs in different order with respect to the others t-components Q, allowing to update the NDeps formed sequencing consecutive m-words. ■

Proposition 6.5 *Let Q be a system model and $S = \overrightarrow{X_i}\sigma_1^1\overrightarrow{X_i}\sigma_1^2\overrightarrow{X_i}\cdots\overrightarrow{X_i}\sigma_1^n\overrightarrow{X_i}\sigma_2^n\overrightarrow{X_i}\sigma_2^{n-1}\overrightarrow{X_i}\cdots\overrightarrow{X_i}\sigma_2^2\overrightarrow{X_i}\sigma_2^1\overrightarrow{X_i}$ be a transition sequence computed from the observed behavior of Q, where X_i is any t-component of Q. Then S has the same property than TSeq.*

Proof. Since in S is preserved the occurrence order of the t-components in TSeq then in S the transitions of a t-component occurs in different order and also all the transitions of any t-component occur before and after the transitions of the others t-components allowing the detection of all dependencies of Q. ■

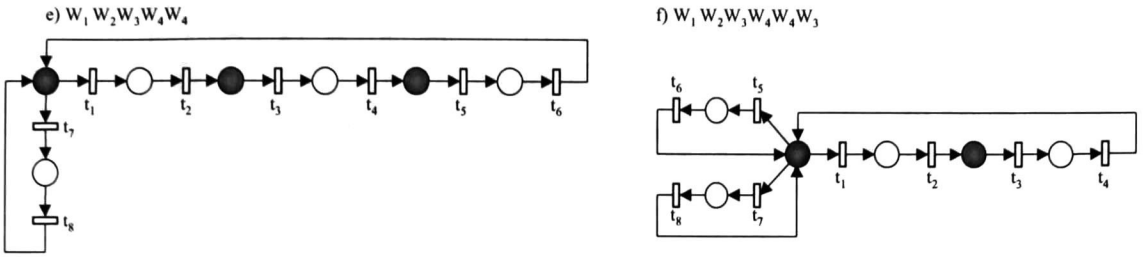


Figure 6.19: Computed model for the sequence $w_1w_2w_3w_4w_4$.

However it is possible that another set of transition sequences not including $TSeq$ allows to compute a correct model for Q, implying that a combination of the transition sequences in this set also provide the entire information about the dependency relation of the transitions in Q.

Example. Consider the system model Q depicted on figure 6.20. The $TSeq$ of Q is $TSeq = \sigma_1\sigma_2$, where $\sigma_1 = t_1t_2t_3t_4t_5$ and $\sigma_2 = t_3t_4t_1t_2t_5$ are the transition sequences computed using algorithm 6.1. With TSeq a model for Q could be computed since the concurrent transitions occur in different order. However consider the set of transition sequences $STSeq = \sigma_a\sigma_b\sigma_c$ where $\sigma_a = t_1t_2t_3t_4t_5$, $\sigma_b = t_1t_3t_4t_2t_5$ and $\sigma_c = t_3t_1t_2t_4t_5$. Notice that the sequences of $STSeq$ also allow to compute a model for Q. The transitions t_1 and t_4 are concurrent transitions, however it could seem that they are dependent transitions since in the transitions sequences of STSeq t_4 appears always after t_1, however notice that a dependence between t_1 and t_4 never is computed since they never occur consecutively.

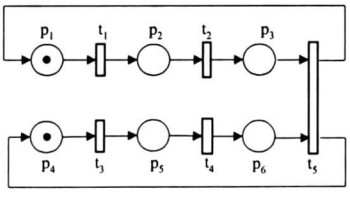


Figure 6.20: System model Q.

If the input signal given to a system model Q is not an excitation persistent input signal then it could occur that:

1. Some t-component is not excited at all

   In this case neither the columns of the φC matrix related with the transitions of this t-component nor the rows of γC matrix related with the non measurable places of this t-component can be computed because it is not observed the behavior generated by this cycle.

   Example. Consider the system model Q depicted on figure 6.21. Assume that the input signal allows to generate the transition sequences $w_1 = t_7t_8$, $w_2 = t_9t_{10}$, $w_3 = t_{11}t_{12}$, $w_2 = t_9t_{10}$ and

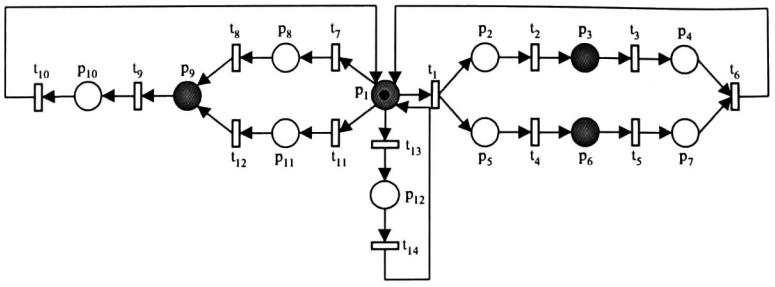


Figure 6.21: System model Q.

$w_4 = t_{13}t_{14}$. The computed model is depicted on figure 6.22. Even the computed model generates the observed behavior of Q (figure 6.21), the model is not complete since the input signal did not excite the t-component containing the transitions t_1, t_2, t_3, t_4, t_5 and t_6.

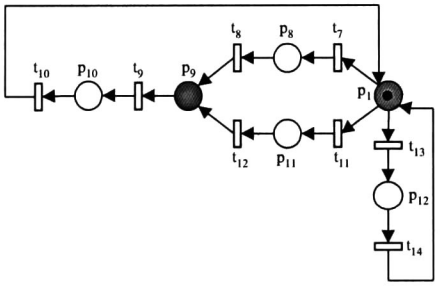


Figure 6.22: Computed model.

2. A t-component is partially excited.

   In this case some NDep have not been computed since a t-component that shares transitions with an already computed t-component has not been computed.

   Example. Consider the system depicted on figure 6.21. Assume that the next sequence of m-words has been detected: $w_1 = t_1t_2t_4t_3t_5t_6$, $w_2 = t_1t_4t_5t_2t_3t_6$, $w_3 = t_{13}t_{14}$, $w_4 = t_7t_8$, $w_5 = t_9t_{10}$, $w_3 = t_{13}t_{14}$. Then the computed model is depicted on figure 6.23. Notice that the non measurable places p_1 and p_9 are not computed correctly since the transition sequence of the t-component $X_k = t_{11}t_{12}t_9t_{10}$ has not been computed.

3. Not all the t-components are excited correctly

   In this case it is possible that all the columns of φC matrix can be computed (if they were computed at least once), however it could occur that some non measurable places are computed wrongly because the

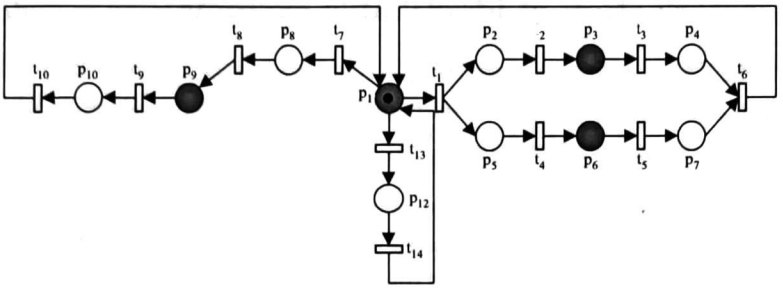


Figure 6.23: Computed model.

information captured from the evolution of the system is not enough to remove, update or compute a new NDep.

Example. Consider again the system model Q depicted on figure 6.21. Assume that the input signal allows to compute the next sequence of m-words: $w_1 = t_1t_2t_4t_3t_5t_6$, $w_2 = t_{13}t_{14}$, $w_3 = t_7t_8$, $w_4 = t_9t_{10}$, $w_5 = t_{11}t_{12}$, $w_4 = t_9t_{10}$ and $w_3 = t_7t_8$. The computed model is depicted on figure 6.24. Notice that this model describes the observed behavior of Q.

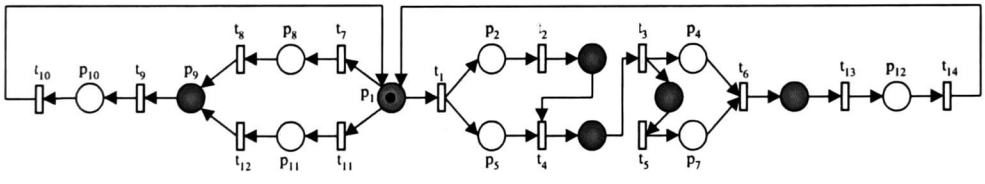


Figure 6.24: Computed model.

Even the matrix φC is computed correctly due to all transitions of the system model were fired, it could occurs that some non measurable places are wrongly computed. These non measurable places are described below.

a. The non measurable places forming NDep between concurrent transitions such as the places forming the NDep $[t_2, t_4]$, $[t_4, t_3]$ and $[t_3, t_5]$ in the computed model depicted on figure 6.24. This occurs because it has not been computed another m-word w_x such that the concurrent transitions occur in different order, if the m-word $w_x = t_1t_4t_5t_2t_3t_6$ is computed then this wrong NDep will be removed.

b. The non measurable places concatenating t-components of Q that forms only one t-component in the computed model. In this example, the t-component $X_k = t_{13}t_{14}$ has not been connected correctly, notice that the place ${}^\bullet t_{13}$ does not belong to Q. This occurs because it has not been computed the m-word w_2 before the m-word w_1.

6.4 Characterization of identifiable systems

Definition 6.1 *A DES is said to be identifiable by the output iff all the internal changes performed into the system are detected through the outputs and the measured output sequences are enough to compute all dependencies and to remove all the NDep wrongly inferred.*

This definition involves the topics addressed in previous discussions; the internal changes can be detected through the output if the hypothesis H_2 and H_3 stated in section 4.1 hold and there are not non-computable structures as described in section 6.2. The second condition refers to the sequences allowing to compute correctly all the measurable and non-measurable dependencies and to eliminate all the wrong NDep inferred during the identification process. This way it is immediate to state the following characterization.

Proposition 6.6 *If a system model Q is identifiable then a) Q is event-detectable by the output b) no transitions of Q are fired simultaneously, Q does not contain self-loops and neither non measurable places describing shared resources nor non measurable implicit places exist in Q, and c) the computed transition sequence detected from the behavior of Q is $S = \overrightarrow{X_i}\sigma_1^1\overrightarrow{X_i}\sigma_1^2\overrightarrow{X_i}\cdots\overrightarrow{X_i}\sigma_1^n\overrightarrow{X_i}\sigma_2^n\overrightarrow{X_i}\sigma_2^{n-1}\overrightarrow{X_i}\cdots\overrightarrow{X_i}\sigma_2^2\overrightarrow{X_i}\sigma_2^1\overrightarrow{X_i}$*

Proof. a) Follows from hypothesis of work H_2 and from proposition 4.1. b) Follows from hypothesis of work H_3 and from proposition 6.2 and c) Follows from proposition 6.5. ■

The previous proposition provides only sufficient conditions for identifiable DES. A complete characterization could be possible if one could characterize all the non-computable structures concerning the placement of non measurable places. This a difficult issue that deserves a more long and depth study on redundancy of the structures (such as implicit non measurable places).

6.5 Identification of non event detectable DES

As presented in chapter 4 the first stage of the identification procedure is to compute the measurable part of the system model Q as it is evolving, the measurable part is represented by the φC matrix of Q, each column of this matrix is computed from any two consecutive output symbols generated by Q as stated in proposition 4.3. However it is possible that some transitions have not effect over the measurable places and hence there could not be detected any change of state at the output of Q and hence those transitions cannot be computed from the output of Q, also it is possible that some transitions have the same measurable part being impossible to distinguish them from the output. Hence, for the identification procedure it is desirable that each transition in the system model Q be distinguishable from another transition from the output information of Q. The condition under which it is possible to detect and distinguish any transition of a system model Q is that it fulfills the event-detectability property introduced in [1].

Definition 6.2 *Let Q be an IPN. Q is event-detectable iff all columns of φC_Q matrix are not null and different from each other.*

As stated in previous definition, there exist two cases for a system model Q does not fulfill the event-detectability property: case 1) that there exists a null column in φC_Q or case 2) that two or more columns in φC_Q be equal. A deeper analysis of these two cases are presented below.

6.5.1 Null columns in φC matrix

The incidence matrix C of an IPN Q describes the relation between the places and the transitions of Q. Thus the elements in the incidence matrix have the following meaning:

$$C[i,j] = \begin{cases} -1 & p_i \text{ is an input place of } t_j, \text{ and hence } t_j \text{ removes marks from } p_i \\ 1 & p_i \text{ is an output place of } t_j, \text{ and hence the firing of } t_j \text{ add marks to } p_i \\ 0 & p_i \text{ and } t_j \text{ are not related, and hence } t_j \text{ does not affect the marking of } p_i \end{cases}$$

Assume that there exists a null column i in φC, i.e. $\varphi C[k,i] = 0$ for every non measurable place p_k of Q, then the firing of the transition t_i has no effect over the measurable places of Q since t_i does not remove or add marks to some measurable place.

The effect produced by a null column i of the φC matrix in the identification problem is that the transition t_i represented by that column cannot be computed using the output information of Q. This claim is based on the fact that a transition is computed from the output symbols of Q as the difference of any two consecutive output symbols (proposition 4.3), however if the firing of a transition t_i from a marking M_k has no effect over the measurable places, then its firing does not modify the output since the output symbol generated by the reached marking M_{k+1} is the same as the output symbol generated by the marking M_k.

Consider that the transition t_i of Q fulfills $\varphi C\vec{t_i} = 0$, i.e. i is a null column of φC matrix, and $M_k \xrightarrow{t_i} M_{k+1}$. In order to compute the transition t_i as $\varphi C\vec{t_i} = \varphi M_{k+1} - \varphi M_k$ notice that:

$M_{k+1} = M_k + C\vec{t_i}$ (state equation)
$\varphi M_{k+1} = \varphi(M_k + C\vec{t_i})$ (applying φ to both sides)
$\varphi M_{k+1} = \varphi M_k + \varphi C\vec{t_i}$ (since φ is linear)
$\varphi M_{k+1} = \varphi M_k$ (since $C\vec{t_i} = 0$)

Then the transition t_i cannot be computed from the output symbols of Q since it is not detected any change of marking in Q due to $\varphi M_{k+1} = \varphi M_k$ i.e. the observed output symbol generated by the marking reached with the firing of t_i is the same as the output symbol generated by the previous marking

Example. Consider the system model depicted on figure 6.25. The output symbol generated by the initial marking $M_0 = \begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T$ of Q is $\varphi M_0 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ since no measurable place is marked in M_0. Assume that the transition t_1 is fired; then Q reaches the marking $M_1 = \begin{bmatrix} 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}^T$ The generated output symbol is $\varphi M_1 = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$ due to the new marked place is p_5 and it is a non measurable place. Notice that $\varphi M_0 = \varphi M_1$ then it is not possible to detect that a change of state had occurred in Q from its output symbols. Notice also that the column 1 of the φC matrix depicted on figure 6.25 is a null column and hence t_1 does not add or remove marks from the measurable places of Q.

- The computed model in this case will be as that depicted on figure 6.26.

By previous illustration it is possible to state that if there exists a null column in the φC matrix of a model system Q then:

- there exist at least two consecutive markings generating the same output symbol since the transition represented by the null column has no effect over the measurable places, like the markings M_0 and M_1 of previous example due to $M_0 \xrightarrow{t_1} M_1$ and $\varphi M_0 = \varphi M_1$

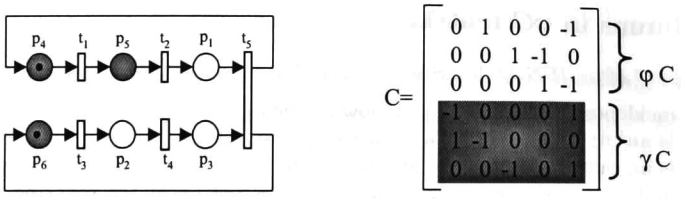


Figure 6.25: System model Q.

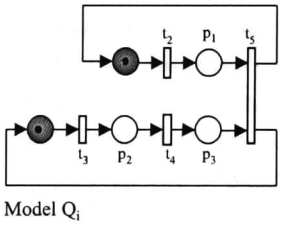


Figure 6.26: Computed model for the system model Q depicted on figure 6.25.

- all the input and output places of the transition t_i represented by the null column in φC matrix are non measurable places.

6.5.2 Equal columns in φC matrix

If there exist two or more columns j and k in the φC matrix of a system model Q such that they are equal columns, i.e. $\varphi C\overrightarrow{t_j} = \varphi C\overrightarrow{t_k}$, implies that the firing of t_j and the firing of t_k have the same effect over the measurable places of Q. The effect caused by this problem in the identification procedure is that these two transitions will be considered as the same transition due to they cannot be distinguished from the output of Q. Consider that $\varphi C\overrightarrow{t_i} = \varphi C\overrightarrow{t_j}$, i.e. i and j are equal columns in φC matrix, and $M_i \xrightarrow{t_i} M_i'$ and $M_j \xrightarrow{t_j} M_j'$. In order to compute the transitions t_i and t_j from the output symbols notice that:

$$
\begin{array}{ll}
M_i' = M_i + C\overrightarrow{t_i} & \\
M_j' = M_j + C\overrightarrow{t_j} & \text{(state equation)} \\
\varphi M_i' = \varphi(M_i + C\overrightarrow{t_i}) & \\
\varphi M_j' = \varphi(M_j + C\overrightarrow{t_j}) & \text{(applying } \varphi \text{ to both sizes)} \\
\varphi M_i' = \varphi M_i + \varphi C\overrightarrow{t_i} & \\
\varphi M_j' = \varphi M_j + \varphi C\overrightarrow{t_j} & \text{(since } \varphi \text{ is linear)} \\
\varphi M_i' - \varphi M_i = \varphi M_j' - \varphi M_j & \text{(since } C\overrightarrow{t_i} = C\overrightarrow{t_j}\text{)}
\end{array}
$$

Then the transitions t_i and t_j cannot be distinguished from the output symbols since $\varphi M_i' \overset{C\overrightarrow{t_i}}{-} \varphi M_i = \varphi M_j' \overset{C\overrightarrow{t_j}}{-} \varphi M_j$, i.e. they are computed as the same transition.

Example. Consider the system model Q depicted on figure 6.27. The t-components of Q are depicted on figure 6.28. The generated output symbols by the t-component X_1 of Q when the transitions t_1, t_2 and t_5 are fired are: $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ and the m-word computed from this output symbols

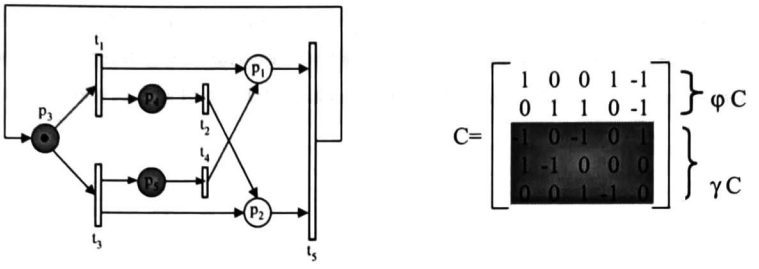


Figure 6.27: System model Q with equal columns in φC matrix.

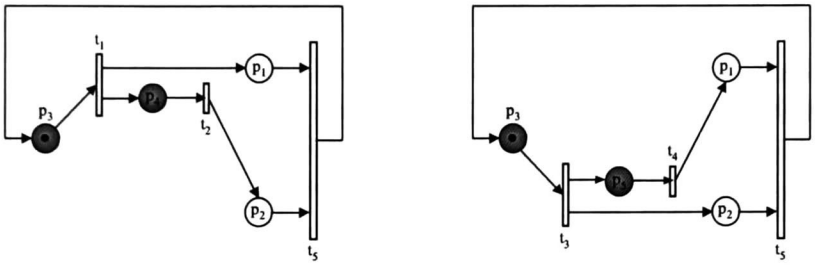


Figure 6.28: T-components of the system model Q depicted on figure 6.27.

is $w_1 = \begin{bmatrix} t_1 \\ 1 \\ 0 \end{bmatrix}\begin{bmatrix} t_2 \\ 0 \\ 1 \end{bmatrix}\begin{bmatrix} t_5 \\ -1 \\ -1 \end{bmatrix}$, the computed model for this m-word is depicted on figure 6.29.a. The t-component X_2 of Q generates the following output symbols $\begin{bmatrix} 0 \\ 0 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \end{bmatrix}\begin{bmatrix} 0 \\ 0 \end{bmatrix}$ when the transitions t_3, t_4 and t_5 are fired, the m-word computed is $w_2 = \begin{bmatrix} t_3 \\ 0 \\ 1 \end{bmatrix}\begin{bmatrix} t_4 \\ 1 \\ 0 \end{bmatrix}\begin{bmatrix} t_5 \\ -1 \\ -1 \end{bmatrix}$. Notice that $\varphi C\overrightarrow{t_3} = \varphi C\overrightarrow{t_2}$ and $\varphi C\overrightarrow{t_4} = \varphi C\overrightarrow{t_1}$. Since $w_1 = t_1t_2t_5$ then $w_2 = t_2t_1t_5$. The computed model is depicted on figure 6.29.b.

Based on previous analysis it can be stated that if a system model Q does not fullfil the event detectability property, then the correct structure of Q cannot be computed using only its output information since a) the

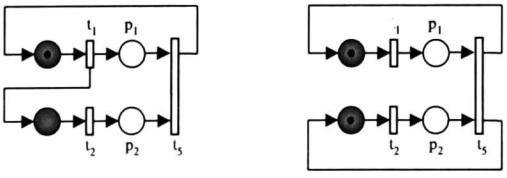


Figure 6.29: Computed models Q_1 and Q_2 for the m-words w_1 and w_2 respectively.

transitions cannot be detected (in the case that φC_Q has a null column) or b) some transitions cannot be distinguished and considered as the same transition (in the case that φC_Q have repeated columns). However, notice that in both cases the computed models describe the behavior of the system models.

Chapter 7

Conclusions

The work herein presented addresses the identification problem of DES; it is devoted to obtain a mathematical model from the observation of the system behavior, which is described by the evolution of its input and output signals. The class of systems dealt in this work are those in which not all the states of the system can be determined from the measurement of its outputs, however these systems can be described by IPN exhibiting the event-detectability property.

The adopted identification approach is a passive one (in which the input signals of the system are not manipulated) and the strategies proposed are translated into procedures executed on-line, building the models progressively. These models represent the observed behavior of the system and they approach asymptotically to the actual model of the system. Every sequence of events is analyzed and if it provides new information on the system then the computed model is updated.

The strategy of the identification procedure is suggested by the cyclic behavior of live and bounded systems. It is based on the reconstruction of the t-components of the system model, by processing the cyclic sequences of transitions computed from the observed output symbols (m-words). During the on-line operation of the identification process, the m-words are computed and then the new model is built adding, removing, or updating non measurable places.

The model synthesis procedure performs mainly two tasks: the computation of the measurable part of the system and the inference of the non measurable part of the system, which is related with the dependencies formed by the non measurable places with respect to the computed transitions. The first task is made directly from the observation of the output system signals, while the second task, rather difficult, derived a more detailed study about the dependencies formed by a non measurable places into a model. The proposed algorithms to updated the non measurable places are of linear complexity in the number of the transitions computed and the m-words detected. The general algorithm to update a model that incorporates all the updating procedures of non measurable places is also executed in polynomial time.

Although the execution of the identification procedure is efficient, the convergence to the actual system model using the passive approach depends on the evolution of the system itself. If the system evolution presents sequences that include a persistent excitation sequence then the model is fast computed, otherwise it is needed to wait until the system exhibits such behavior; however in each evolution the identification procedure generates a model describing the observed system behavior. The identification problem is complex by nature; the expectation of a certain order of the transition sequences can be seen as a non deterministic problem of m-words

generation; then the identification problem of DES belongs to the class of NP problems. This inconvenient is partially overcame by the on-line operation of the identification procedure as the system evolves, because the computed model describes the observed behavior; also because the behavior of a system is not always chaotic or biased and its activity sequences are presented in a reasonable time elapse.

Another identification approach that it is not considered in this thesis to dismiss the complexity problem is the active identification approach, in which it is possible to manipulate the input signals of the system to validate the computed model. This procedure would consist in capture the observed behavior of the system in a model computed using the passive approach, then commute to the active operation applying sequences that allow to validate the computed dependencies formed with non measurable places.

An approach more realistic than the black box approximation is one in which it is considered the existence of a legacy system for which there exists available information, the problem in this case is to characterize the situations or structures that must to have the available information that helps the validation of some computed dependencies in the on-line identification procedure.

This work is a first approximation of the identification problem in DES and it can be considered as a basis for future works on the matter, possibly oriented towards the verification of systems, hardware or software, or it can be extended to address problems of reverse engineering.

Bibliography

[1] L. Aguirre, A. Ramírez & O. Begovich. *Design of Asymptotic Observers for Discrete Event Systems.* Proceedings of the IASTED International Conference on Intelligent Systems and Control. Santa Barbara Ca., USA 1999.

[2] D. Angluin. *Finding Patterns Common to a Set of Strings.* Journal of Computational System Science. Vol. 21, pp. 46-62. 1980.

[3] D. Angluin. *Inductive Inference of Formal Languages from Positive Data.* Information and Control. Vol. 45, pp. 117-135. 1980.

[4] D. Angluin. *A Note on the Number of Queries Needed to Identify Regular Languages.* Information Control. Vol. 51, pp. 76-87. 1981.

[5] D. Angluin. *Inference of Reversible Languages.* Journal of the ACM. Vol. 29, pp. 741-765. 1982.

[6] D. Angluin. *Inference of Reversible Languages.* Journal of the Association for Computing Machinery, Vol. 29, No. 3, pp. 741-765, July 1982.

[7] D. Angluin and C.H. Smith. *Inductive Inference: Theory and Methods.* ACM Comput. Surveys. Vol 15, pp 237-269. 1983..

[8] D. Angluin. *Learning regular sets from Queries and Counter-Examples.* Inform. Comput. Vol 75, pp. 87-106. 1987.

[9] D. Angluin. *Queries and Concept of Learning. Machine Learning.* Vol. 2, pp. 319-342. 1988.

[10] D. Angluin. *Negative Results for Equivalence Queries.* Machine Learning. Vol. 5, pp. 121-150. 1990.

[11] D. Angluin and M. Kharitonov. *When won't membership queries help?.* 23th Annual ACM Symposium on Theory of Computing, ACM Press, New York. pp 444-454. 1991.

[12] A. Burago. *Learning Structurally Reversible Context-Free Grammars form Queries and Counter examples in Polynomial Time.* 6th Workshop on Computational Learning Theory, ACM Press, New York. pp. 140-146. 1994.

[13] R.C. Carrasco, J. Oncina. Proc. International Coll. on Grammatical Inference (ICGI-94). Lecture Notes in Artificial Intelligence, Vol. 862, Springer, Berlin, 1994.

[14] Chen, Chi-Tsong. *Linear System Theory and Design.* Harcourt Brace Jovanovich Inc. Sanders College Publishing. 1970.

[15] S. Crespi-Reghizzi, An effective model for grammar inference, Information Processing Letters, Vol. 71, pp. 236-245, Amsterdam, 1972.

[16] C.A. De-Jesús and A. Ramírez-Treviño. *Controller and Observer Synthesis in Discrete Event Systems Using Stability Concepts.* 2001 IEEE International Conference on Systems, Man & Cybernetics, pp 664-668,Tucson Arizona USA, October 2001.

[17] J. Esparza and J. Desel. *Free Choice Petri Nets.* Cambridge University Press 1995.

[18] A.F. Fahmy and A.W. Biermann. *Synthesis of Real Time Acceptors.* Symbolic Computing Journal. Vol. 15, pp. 807-842. 1993.

[19] E.M. Gold. *Language Identification in the Limit.* Information and Control, Vol. 10, pp. 447-474. 1967.

[20] E. M. Gold. *Complexity of Automaton Identification from Given Data.* Information and Control, Vol. 37, pp. 302-320. 1978.

[21] A. Guia. *Petri Nets as Discrete Event Models for Supervisory Control.* Ph. D. thesis, Rensselaer Polytechnic Institute. Troy, New York, 1992.

[22] K. Hiraishi. *Construction of Safe Petri Nets by Presenting Firing Sequences.* Lectures Notes in Computer Sciences, 616, pp. 244-262. 1992.

[23] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley Publishing Company. 1979.

[24] H. Ishizaka. *Polynomial Time Learnability of Simple Deterministic Languages.* Machine Learning. Vol. 5, pp. 151-164. 1990.

[25] T. Koshiba, *Typed Pattern Languages and their Learnability.* Proceedings of the 2nd European Conference on Computational Learning Theory. Lectures Notes in Artificial Intelligence. Vol. 904, pp. 367-379. Springer,Berlin. 1995.

[26] A. Krogh, M. Brown, I.S. Mian, K. Sjölander and D. Haussler. *Hidden Markov model in Computational Biology: Applications to Protein modeling*, Journal of Molecular Biology, vol. 235 (1994) pp 1501-1531.

[27] L.S. Levy and A.K. Joshi. *Skeletal structural descriptions.* Information and Control. Vol. 39, pp. 192-211. 1978.

[28] Yong Li and W.M. Wonham. *Controllability and Observability in the state-feedback Control of Discrete-Event Systems.* Proceedings of the 27th Conference on Decision and Control. Austin, Texas, pp. 203-208, December 1988.

[29] E. Mäkinen. *On Structural Grammatical Inference Problem for some Classes of Context-Free Grammars.* Information Processing Letters. Vol. 42, pp. 193-199. 1992.

[30] M.E.Meda, A. Ramírez and A. Malo. *Identification in Discrete Event Systems.* IEEE-Systems Man and Cybernetics Conference, San Diego Ca., USA, pp. 740-745. October 1998.

[31] M.E. Meda, DES identification using Interpreted Petri nets, International Symposium on Robotics and Automation, pp 353-357. December 1998.

[32] M.E. Meda-Campaña, A. Ramírez-Treviño and E. López-Mellado. *Dynamical local properties for estimation and control of discrete event systems modeled by interpreted Petri.* 2000 IEEE International Conference on Systems, Man and Cybernetics. Nashville TN, USA, pp 2150-2155. October 2000.

[33] M.E. Meda-Campaña, A. Ramírez-Treviño and E. López-Mellado. *Asymptotic Identification for DES.* IEEE International Conference on Decision and Control. Sydney, Australia. pp 2266-2271. December 2000.

[34] M.E. Meda-Campaña and E. López-Mellado. *A passive method for on-line identification of Discrete Event Systems.* IEEE International Conference on Decision and Control. pp 4990-4995 Orlando Fl. USA. December 2001.

[35] M.E. Meda-Campaña and E. López-Mellado. 41th IEEE International Conference on Decision and Control. Las Vegas, Nevada. USA. December 2002.

[36] T. Murata. *Petri nets: Properties, Analysis and Applications.* Proceedings of the IEEE, Vol. 77, No. 4, pp. 541-580, 1989.

[37] J. Oncina, P. Garcia and E. Vidal. *Learning subsequential Transducers for Pattern Recognition Interpretation Tasks.* IEEE Transactions, Pattern Analysis Machine Intelligence. Vol. 15, pp. 448-458. 1993.

[38] Cüneyt M. Özveren and Alan S. Willsky. *Observability of Discrete Event Dynamic Systems.* IEEE Transactions on Automatic Control, Vol. 35, No. 7, pp. 797-806, July 1990.

[39] K.M. Passino, A.N. Michel and P.J. Antsaklis. *Stability analysis of a class of discrete event systems.* IEEE Transactions On Automatic Control, Vol. 39, No. 2, pp. 269-279, 1994.

[40] F. Pereira and Y. Schabes. *Inside-outside reestimation for Partially bracketed corpora.* Proc. Ann. Meting of the Association for Computational Linguistics, pp. 128-135. 1992.

[41] J.L. Peterson. *Petri net theory and the modelling of systems*, Prentice hall inc., Englewood Clifs, New Jersey 07632.

[42] L. Pitt. *Inductive Inference, DFAs, and Computational Complexity.* AII-89 Workshop on Analogical and Inductive Inference. Lecture Notes in Computer Science. Vol. 397, pp. 18-44. Springer, Berlin. 1989.

[43] L. Pitt and M.K. Warmuth. *The Minimum Consistent DFA Problem Cannot Be Approximated within any Polynomial.* ACM Symposium on Theory of Computing, ACM Press, New York, 1989.

[44] A. Ramírez-Treviño, I. Rivera-Rangel and E. López-Mellado. *Observer Design for Discrete Event Systems Modeled by Interpreted Petri Nets.* IEEE-International Conference on Robotics and Automation, San Francisco Ca., USA, pp 2871-2876, April 2000.

[45] D. Ron and R. Rubinfeld. *Learning Fallible Deterministic Finite Automata*, Machine Learning. Vol. 18, pp, 149-185. 1995.

[46] Y. Sakakibara. *Learning Context-Free Grammars from Structural Data in Polynomial Time.* Theoretical Computing Science. Vol. 76, pp. 223-242. 1990.

[47] Y. Sakakibara, *On learning from queries and counter examples in the presence of noise*, Information Processing Letters, Vol. 37, pp. 279-284, 1991.

[48] Y. Sakakibara. *Efficient Learning of Context-Free Grammars from Structural Examples.* Information and Computing. Vol. 97, pp. 23-60. 1992.

[49] Y. Sakakibara and R. Siromoney. *A Noise Model on Learning Sets of Strings.* 5th Workshop on Computational Learning Theory, ACM Press, New York. pp. 295-302. 1992.

[50] Y. Sakakibara, M. Brown, R. Hughey, I.S. Mian, K. Sjölander, R.C. Underwood and D. Haussler. *Stochastic Context Free Grammars for tRNA Modeling*, Nucleic Acids Res. vol. 22, pp 5112-5120. 1994.

[51] Y. Sakakibara. *Recent Advances of Grammatical Inference.* Theoretical Computer Science. Vol. 185, pp. 15-45. 1997.

[52] T. Shinohara, *Rich Classes Inferable from Positive Data: Length-Bounded Elementary Formal Systems.* Information and Computation. Vol. 108, pp. 175-186. 1994.

[53] M. Silva. *Las Redes de Petri: en la Automática y la Informática.* Editorial AC. 1985.

[54] Y. Takada. A *Hierarchy of Languages Families Learnable by Regular Languages Learners.* Information and Computing Vol. 123, pp. 138-145. 1995.

[55] Y. Takada. *Grammatical Inference for Even Linear Languages Based on Control Sets.* Information Processing Letters. Vol. 28, pp. 193-199. 1998.

[56] B.A. Trakhtenbrot and J. Barzdin. *Finite Automata Behavior and Synthesis.* Fundamental Studies in Computer Science, North-Holland, Amsterdam, 1973.

[57] L.G. Valiant. *A theory of the Learnable.* Comm. ACM. Vol. 27, pp1134-1142. 1984.

[58] R. Wiehagen. *From Inductive inference to Algorithmic Learning Theory.* New Generation Computation. Vol 12, pp 321-335. 1994

[59] W. M. Wonham. *Notes on Control of Discrete-Event Systems.* Systems Control Group, Dept. of Electrical & Computer Engineering. University of Toronto, 1996-97.

[60] T. Yokomori. *Polynomial-Time Learning of Very Simple Grammars from Positive Data.* 4th Workshop On Computational Learning Theory. pp. 213-227. 1991.

[61] T. Yokomori. *Learning Non Deterministic Finite Automata from Queries and Counter examples.* Machine Intelligence. Vol. 13. Oxford University Press, pp. 169-189. 1994.

[62] T. Yokomori. *On Polynomial-Time Learnability in the Limit of Strictly Deterministic Automata.* Machine Learning. Vol. 19, pp. 153-179. 1995.

CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, aprobó la tesis titulada: "Identificación en Línea de Sistemas de Eventos Discretos: Fundamentos y Algoritmos para la Síntesis de Modelos en Redes de Petri", que presentó **María Elena MEDA CAMPAÑA** el día 22 de Noviembre de 2002.

Dra. Ofelia BEGOVICH MENDOZA
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara
Guadalajara, Jal.

Dr. Luis Ernesto LÓPEZ MELLADO
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara
Guadalajara, Jal.

Dr. Antonio RAMÍREZ TREVIÑO
Investigador CINVESTAV 2A
CINVESTAV Unidad Guadalajara
Guadalajara, Jal.

Dra. Xiaoou LI
Investigador CINVESTAV 3A
CINVESTAV-México
México, D.F.

Dr. Alejandro Justo MALO TAMAYO
Investigador CINVESTAV 2A
CINVESTAV-México
México, D.F

Dr. Victor Manuel LARIOS ROSILLO
Investigador Titular
CUCEA
Universidad de Guadalajara
Zapopan, Jal.