CT. 729-SS1
Don. - 2013

xx(209421.1)

Centro de Investigación y de Estudios Avanzados

del Instituto Politécnico Nacional

Unidad Guadalajara

Ubicación de sensores para observabilidad en Redes de Petri fluidificadas

Tesis que presenta:
Enrique Javier Aguayo Lara

para obtener el grado de:
Doctor en ciencias

En la especialidad de:
Ingeniería eléctrica

Directores de tesis:
Dr. Antonio Ramírez Treviño
Dr. José Javier Ruíz León

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Diciembre 2012.

Ubicación de sensores para observabilidad en Redes de Petri fluidificadas

Tesis de Doctorado en Ciencias en Ingeniería Eléctrica

Por:
Enrique Javier Aguayo Lara
Maestro en ciencias en Ingeniería eléctrica
Cinvestav Unidad Guadalajara 2005-2008

Becario de CONACYT, No. 47949.

Directores de tesis:
Dr. Antonio Ramírez Treviño
Dr. José Javier Ruíz León

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Diciembre 2012.

Centro de Investigación y de Estudios Avanzados

del Instituto Politécnico Nacional

Unidad Guadalajara

Sensor placement for observability in fluidified Petri nets

A thesis presented by:
Enrique Javier Aguayo Lara

to obtain the degree of:
Doctor in Science

in the subject of:
Electrical Engineering

Thesis Advisors:
Dr. Antonio Ramírez Treviño
Dr. José Javier Ruíz León

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, December 2012.

Sensor placement for observability in fluidified Petri nets

Doctor in Science thesis
in Electrical Engineering

By:
Enrique Javier Aguayo Lara
Master in sciences in Electrical Engineering
Cinvestav Unidad Guadalajara 2005-2008

Scholarship granted by CONACYT, No. 47949.

Thesis Advisors:
Dr. Antonio Ramírez Treviño
Dr. José Javier Ruíz León

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, December 2012.

Resumen Ejecutivo

Las redes de Petri continuas temporizadas con la semántica de servidores infinitos (*ContPN*) son una relajación de una red de Petri discreta, en la cual las transiciones pueden ser disparadas en valores reales; por esta razón, el marcado de la red (el estado del sistema) se vuelve un vector real no negativo. En este trabajo se muestra que las *ContPN* se pueden modelar como un sistema lineal conmutado (*SLS*), el cual consiste en una familia de sistemas lineales (*LS*) conmutando entre sí; para la conmutación de *LS*, dependiendo del marcado de la red, un *LS* se vuelve activo. Un resultado conocido es que un *SLS* autónomo es observable si y solo sí cada uno de los *LS* que lo conforman es observable y además cada par de ellos es distinguible entre sí. Desafortunadamente, aún cuando el problema de Observabilidad en *SLS* está resuelto, en las *ContPN* el problema no es trivial, puesto que el número de *LS* necesarios para la representación de la *ContPN* en *SLS* crece de manera exponencial con el número de transiciones de sincronización, haciendo que el enfoque ingenuo de verificar si cada uno de los *LS* sea observable y cada par de ellos sean distinguible, resulta imposible en términos prácticos. En cambio, este trabajo presenta dos principales aportaciones:

- Una estrategia para la colocación de sensores en una *ContPN*, la cual garantiza que el sistema sea observable. Para esto, algunos de los subespacios invariantes de la *ContPN* son caracterizados desde la estructura de la *ContPN*, eliminando la necesidad de la enumeración y cálculo de las matrices dinámicas de cada *LS* de la representación en *SLS* de la *ContPN*.
- El diseño de observadores de estado que permiten el cómputo del marcado del *ContPN* con un único observador para redes de la clase free choice.

Abstract

Continuous timed Petri nets with infinite servers semantics (*ContPN*) are a relaxation of a discrete Petri net. In this case, transitions may be fired in real amounts. For this reason, the marking of the net (the state of the system) becomes a non-negative real vector. In this work it is shown that *ContPN* can be modelled with a switched linear system (*SLS*), which is a family of linear systems (*LS*) switching among them; for the switching of *LS*, depending on the marking of the net, a *LS* is chosen. A known result is that an autonomous *SLS* is observable if and only if every *LS* of the family is observable and each pair of *LS* is distinguishable from each other. Unfortunately, even when the observability problem in *SLS* is solved, in *ContPN* the problem is not trivial. This is because the number of *LS* in the family of *SLS* increases exponentially with the number of join transitions; therefore the naive approach of verifying the observability in each *LS* and the distinguishability between each pair of *LS* becomes prohibited in practice. Instead, this work presents two main contributions:

1. An strategy for sensor placement in *ContPN*, which guarantees observability. In order to achieve this, some of the invariant subspaces of the *ContPN* are characterized from the *ContPN* structure, avoiding the need of the enumeration and computation of the dynamical matrices of each *LS* in the *SLS* representation of the *ContPN*.

2. An observer design, which allows the computation of the marking of the *ContPN* with a single observer structure for the free choice class of nets.

Agradecimientos

Al CONACYT.

A **Bere**, por tu incansable apoyo, guía, amor y comprensión. No tengo palabras que describan mi agradecimiento.

A mis asesores **Antonio Ramírez Treviño** y **José Javier Ruíz León**. Excelentes maestros y mejores personas. Gracias por todo su apoyo, guía y amistad. Mi respeto y admiración para toda la vida.

A mi **familia**. Papá, mamá y Carmen. Gracias por siempre estar conmigo.

Oto. ¡Ahí la llevamos!

Contents

CONTENTS

List of Figures

LIST OF FIGURES

1

Introduction

Discrete event systems (*DES*) framework is adequate to model in a very intuitive way a lot of human designed systems [1]. For instance, consider algorithms, traffic [2], transportation or manufacturing processes [3]; all these systems can be *explained* or modelled by a set of *steps*, creating sequences of *events* leading to the completion of an activity. This behaviour is easily captured using the *DES* concepts [4], [5], [1]. In order to represent these systems, many paradigms have been developed, such as process algebras [6], boolean equations [7] [8], finite automaton [9] and Petri nets [10] among many others. Each one of these paradigms provides a series of formal *objects* to capture the main *DES* characteristics, such as the state of the system, the events generating the state changes, event precedence relations and available resources in the system. In the same way, *DES* provide a wide variety of strategies to control the behaviour of the systems, such as te ones presented in [11], [12], [13]

Among the many modelling tools, Petri nets (*PN*), introduced by C. A. Petri in 1962 in his doctoral thesis [10], provides a compact graphical *DES* representation, as well as a sound mathematical background to analyze such models. The graphical representation includes two types of nodes: **places**, usually associated to the system's state and graphically represented by circles; and **transitions**, usually associated to events and represented by boxes or squares. These nodes are *linked* or connected by arcs, which are arrows pointing from a place to a transition or from a transition to a place, and which constitute the precedence relations. The system's available resources are represented by *tokens* or marks residing inside each place.

Some relevant properties of *PN* are :

- **Liveness**, which deals with the possibility of a transition to be eventually fired again from any reached state, i.e. the event can occur again.

- **Boundedness**, which determines if there exists a bound for the maximum number of tokens residing inside each place in any system's state.
- **Controllability**, which deals with the problem of leading the state of the *PN* from an initial state to a required one.
- **Observability**, which deals with the possibility to determine the initial state of the system (or the current state of the system) when not all the state is available for measurement.

In order to determine if any of the previous properties is present in a *PN*, there exist several approaches [14], [15]:

1. **Enumeration of the state space**, which in *PN* is represented by the *reachability graph.* Using this approach, it is always possible (theoretically in bounded *PN*) to determine if a property is present on a *PN*. Unfortunately, when dealing with highly populated systems, the state explosion problem appears, and therefore it becomes prohibited in practice to use this approach.
2. **Reduction techniques**, which are based on finding out an equivalent net in which the number of places or transitions is reduced, but guaranteeing the preservation of some properties of interest. This leads to the analysis of a *smaller PN*. Unfortunately, this approach does not provide solution for all possible cases [16].
3. **Analytical techniques**, which are based on the underlying structure of the *PN*, which is captured in a matrix called the *incidence matrix* of the *PN*. These mathematical techniques are usually dependent on the initial marking of the *PN* or provide only sufficient or necessary conditions, but not both [16].

In order to avoid the state explosion problem, *continuization* or *fluidization* (when also time is present) is an efficient relaxation technique used in many modelling paradigms, such as queueing networks [17], Markovian models [18] and stochastic process algebras [19], among many others. The continuization consists on approximating the discrete states by some continuous ones [20]. In the *PN* context, the continuous *PN* were first introduced in [21]. This is achieved by the *continuization* of the transition's firing, i.e. enabled transitions can be executed in real amounts.

Very often, the processes described by *DES* also have a time dependence [4] and many techniques have been developed to analyze them [22]. In order to include the time in the *PN* context, the timed *PN* were developed. This tool assigns to each transition a delay, i.e. a period of time that the system must remain in a state before an event can occur. As a *natural* evolution of timed *PN* continuous timed Petri nets (*ContPN*) were also developed (the fluidified model). The *ContPN* provide useful information of the behaviour of the real system under some considerations [23] [16] [24].

Even though *ContPN* were born as an approximation of the discrete *PN*, they have been successfully used to model many hybrid and complex systems [25] such as biological systems [26], traffic systems [27] and manufacturing systems [28], among many other. Besides the modelling, many properties characterizations have been developed in the continuous *PN* and *ContPN* framework [22]. Some relevant research areas in the continuous version of *PN* are: fault detection [29], diagnosis [28], [30], control [31] [32] [33] and controllability [34], [35], performance analysis [36] [37] and deadlock freeness [38].

Unfortunately, in many practical applications, the state of the system is not available for measurement, but only a part of it (only the *output* of the system). The knowledge of the state of the system is important for many applications, such as state feedback control, fault detection, etc. Then, it is necessary to have techniques to determine either the initial state of the system, and from its evolution, or the current state of the system [39], [40]. This is known as the *observability* problem [41]. In this way, it is important to structurally characterize if a *ContPN* presents the observability property [42], [43], [44], [45], [46] or to develop an strategy to determine the adequate instrumentation that the *ContPN* must possess to become observable. This work is concerned with two problems: the sensor placement for observability in continuous timed Petri nets and, once the observability is obtained, the observer design.

In this work it is shown that *ContPN* can be modelled with a switched linear system (*SLS*) [47], which is a family $\mathcal{F}$ of linear systems (*LS*) switching among them. The switching between *LS*, depends on the marking of the net. The number of *LS* in the family $\mathcal{F}$ depends (exponentially) on the number of *join* transitions, where a transition is named a join if it has more than one input arc.

A very well known result is that an autonomous *SLS* is observable if and only if every *LS* of the family is observable and each pair of *LS* is *distinguishable* from each other [48] [49]. The distinguishability property deals with the possibility of determining which *LS* is evolving only by measuring the output of the *SLS*.

Unfortunately, even when the observability problem in autonomous *SLS* is solved, in *ContPN* the observability problem is not trivial. This is because the number of *LS* in the family $\mathcal{F}$ increases exponentially with the number of join transitions, i.e. there is a huge number of *LS* in $\mathcal{F}$; therefore the naive approach of verifying the observability in each *LS* and the distinguishability between each pair of *LS* becomes prohibited in practice.

Once a system presents the observability property, it is possible to design an observer, which will allow the computation of the state of the system ([41], [40], [39] and [50]).

1.1 State of the art

In the literature there exist some results on observability in *ContPN*. The most relevant results on this topic are presented in [44], [45] and [43]. These results are developed in a very intuitive way using the underlying graph of the *ContPN*. With this approach, in [44] the first basic results on observability in *ContPN* are presented. First, consider a place with a sensor, i.e. the marking of such measured place is always known. Using the knowledge of the marking of a measured place, the authors show that it is possible to compute the marking of places located *upstream* as long as there does not exist any attribution place (a place with more than one input transition). Using this reasoning and a set of measured places, they also present an algorithm to determine all the places whose marking can be computed. Then, with the aim of extending these results for *ContPN* with join transitions, they provide some conditions under which the same results hold for the non-join free case.

Once the observability property has been proved, the authors provide an strategy for computing estimates in the non-free case of *ContPN*. In this case, the main drawback is that they propose to construct an observer for each *LS* and then filter out those observers which provide markings which are not coherent with the *ContPN* marking.

In [45], the previous results presented in [44] are extended and clarified. Since the number of *LS* in the *SLS* representation of the *ContPN* increases exponentially, the authors provide a linear programming problem (*LPP*) to determine which *LS* are redundant, i.e. those *LS* should not be analyzed since they are never active or they are included in the border of one of the convex polyedral regions of the system [51] [52].

In [45], the authors also provide a characterization for observability in *ContPN* which also contain attribution places. This result is a particular result of the ones provided in generic observability for structured linear systems [53], where an *associated directed graph* is constructed. The main drawback of this approach is that in order to determine if the *ContPN* is observable, it is necessary to search in the associated directed graph for a *contraction* [53], which is also an NP-algorithm.

Finally in [43] the authors provide an strategy to determine optimal cost sensor placement. It is based on assigning a cost of measuring each place. Then, they construct an output map such that the *ContPN* is observable at minimum cost. Unfortunately, these results are very restrictive in the classes of *ContPN* that can be solved with the algorithm that they present. This is because even though the authors use the structure of the *ContPN* to solve some classes of nets, the algorithm is still of complexity *NP* in the general case.

1.2 Objective and main contributions

The main objectives of this work are:

1. To find a procedure for the sensor placement in a general *ContPN* such that it becomes observable.

2. To develop an observer design such that with a single observer it is possible to compute the state of the *ContPN*.

The first objective deals with the sensor placement for the observability problem. As previously discussed, if a *ContPN* presents the observability property, then it is possible to determine the initial or the current state of the system. Unfortunately, as previously discussed, it is not possible in practice to determine the observability property of a *ContPN* using traditional methods (such as verifying the rank of the observability matrix [40], [39]), since they lead to complete *NP* algorithms. Instead, this work presents an strategy for sensor placement in *ContPN* which guarantees observability. In order to achieve this, some of the invariant subspaces of the *ContPN* are characterized directly from the *ContPN* structure. Using the *PN* structure, it is avoided the need of enumerating and computing all of the dynamical matrices (one per *LS* in the *SLS* representation of the *ContPN*).

The second objective deals with the observer design. An observer design for *ContPN* will be presented allowing the computation of the marking of the *ContPN* with a single observer for the free choice class of nets. This is relevant, since the classical approach is to design an observer for each *LS*; then it is decided which *LS* is actually active and the state is obtained from its observer. Unfortunately this approach would require a large number of observers to be designed, and therefore the existence of a single observer structure is of relevance.

1.3 Document organization

This document is organized as follows:

- Chapter 2 presents basic concepts on Petri nets and its extensions to continuous and continuous timed Petri nets (*ContPN*). In this chapter, a brief review of concepts in linear systems is included.
- Chapter 3 reviews the existing results on observability and observers design in *ContPN*.
- Chapter 4 introduces the A_k-invariant subspaces in the *ContPN* context. These A_k-invariant subspaces will be characterized using the structure of the *ContPN*.
- Chapter 5 deals with the sensor placement problem. In order to achieve this, some of the $A-$invariant subspaces of the *ContPN* are characterized. Based on these $A-$invariant subspaces, an algorithm is introduced to place sensors in the *ContPN* such that observability is guaranteed. Since the sensor placement algorithm does not guarantee the optimality in the number of sensors, an algorithm to reduce the number of sensors is also included.
- Chapter 6 is concerned with the observer design. For this, a dynamic separation of the state equation of the *ContPN* is introduced. With this dynamic separation, a single structure observer design is presented, which guarantees the asymptotic convergence of the estimate marking at any desired rate.
- Chapter 7 presents a case of study based on the cigarettes production process. In this chapter, the process will be represented as a *ContPN* and then the sensor placement algorithms will be used to guarantee that the model of the process is observable. Finally, an observer for the process will be designed.
- Chapter 8 provides the conclusions and future work.

2

Basic Concepts

This chapter contains the basic concepts and the notation used in this work. In Section 2.1 the Petri nets are presented. The basic concepts such as its structure, graphical representation, the incidence matrix and the discrete evolution is presented. Then, its extensions to continuous and continuous timed Petri nets (*ContPN*) are presented in Section 2.1.2. This section also includes the basic concepts of controllability and observability in *ContPN*. Finally, at the end of Section 2.1.2, it will be shown that a continuous timed Petri net can be represented by a switched linear system (*SLS*), which is a family of linear systems (*LS*) switching among them. Therefore, in the Sections 2.3 and 2.4, some basic concepts on linear systems and switched linear systems are briefly recalled.

2.1 Petri Nets

This section is devoted to formally define Petri nets (*PN*) and its relaxations to continuous *PN* and continuous timed *PN*. The Petri nets were first introduced by C. A. Petri in 1962 in his Ph. D. thesis [10]. Since then, many extensions to *PN* have been developed and nowadays, *PN*s represent a wide research area in the discrete event systems framework. *PN* and its relaxations have been intensively studied and are commonly used to model, analyze and control systems [1] in several fields such as communications systems, [54], [55], computer science [56], manufacturing [57], [3], traffic [58] and diagnosis [59], among other applications.

Basic definitions and relevant concepts of *PN*, continuous *PN* and continuous timed *PN* will be presented next. An interested reader on *PN* may consult [14], [56]; on continuous *PN*, its introduction and general concepts may read [60], [61]; on continuous timed Petri nets (*ContPN*) with the infinite servers semantics may consult [16].

2.1.1 Discrete Petri Nets

Definition 2.1.1 *The Petri net structure is a bipartite digraph formed by the four-tuple* $N = \langle P, T, \mathbf{Pre}, \mathbf{Post} \rangle$ *where* $P = \{p_1, p_2, \ldots, p_{|P|}\}$ *is a finite set of nodes called places;* $T = \{t_1, t_2, \ldots, t_{|T|}\}$ *is a finite set of nodes called transitions; The sets* P *and* T *are disjoin, i.e.* $P \cap T = \emptyset$. $\mathbf{Pre}$ *and* $\mathbf{Post}$ *are* $|P| \times |T|$ *matrices representing the weighted arcs going from places to transitions and from transitions to places respectively.*

Each node, either a place or a transition, and the directed arcs can be graphically represented as follows:

- Places are represented as circles
- Transitions are represented as boxes or squares
- Arcs are represented as arrows from the source node to the ending node.

Example 2.1.2 *In Figure 2.1.1, a simple PN structure with two places and two transitions is presented. The set of places is* $P = \{p_1, p_2, p_3\}$. *The set of transitions is* $T = \{t_1, t_2, t_3, t_4\}$. *Matrices* $\mathbf{Pre}$ *and* $\mathbf{Post}$ *are:*

$$\mathbf{Pre} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad \mathbf{Post} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

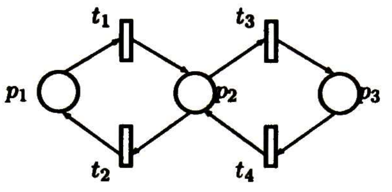


Figure 2.1.1: A simple *PN* structure.

The graphical *connection* among nodes, represented by the arcs, lead to the concept of input set and output set of a node. This is formally defined next:

Definition 2.1.3 *Let $n \in P \cup T$ be a node of N.*
The input set of n or Pre-set of n, denoted by $\bullet n$, is defined as:

$$\bullet n = \{n_i \in P \cup T | \text{ there exists an arc from } n_i \text{ to } n\}.$$

The output set of a node n or Post-set of n, denoted by $n\bullet$, is defined as:

$$n\bullet = \{n_i \in P \cup T | \text{ there exists an arc from } n \text{ to } n_i\}.$$

A sequence of nodes in which $n_1 \in n\bullet$ is named a path, formally defined next.

Definition 2.1.4 *[14] A path in a PN structure N is a non-empty sequence $n_1 n_2 ... n_g$ of nodes which $\forall h = 1, 2, ..., g$ satisfies:*

$$n_{h+1} \in n_h\bullet$$

i.e. there exists an arc between node n_h and n_{h+1}.

A path $\omega = n_1 n_2 ... n_g$ is said to lead from n_1 to n_g. It is clear that it starts in n_1 and ends in n_g.

Definition 2.1.5 *Let ω be a path in a PN.*

- *The notation $final(\omega)$ stands for the final node of the path ω.*
- *The notation $\vec{\omega} \in (\mathbb{N} \cup 0)^{|T|+|P|}$ is a vector which contains the number of times that the node n_i appears in the path ω, according to the indexation $\begin{bmatrix} p_1 & ... & p_{|P|} & t_1 & ... & t_{|T|} \end{bmatrix}^T$ Particularly $\vec{n_i}$ represents the vector for the path $\omega = n_i$.*
- *Let $\omega_1 = n_1 n_2 ... n_g$ be a finite path. The notation ωn_l stands for the new path $\omega_2 = n_1 n_2 ... n_g n_l$.*

Example 2.1.6 *Take the PN structure in Fig. 2.1.1. The sequence $\omega_1 = p_1 t_1 p_2$ is a path. However, the sequence $w_2 = p_1 p_2 p_3$ is not, since there are no arcs from p_1 to p_2.*

The final node of the path ω_1, $final(\omega_1) = p_2$.

The corresponding vector $\vec{\omega_1}$ is given by:

$$\vec{\omega_1} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^T$$

For the construction of a path, consider the path ω_1. The path $\omega_3 = \omega_1 t_3 = p_1 t_1 p_2 t_3$, with its corresponding vector:

$$\vec{\omega_3} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 1 & 0 \end{bmatrix}^T$$

Some nodes of particular interest are the *ending* nodes, formally defined next.

Definition 2.1.7 *A node $n \in P \cup T$ is named an ending node if $n\bullet = \emptyset$.*

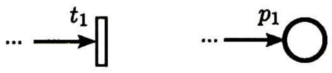


Figure 2.1.2: Ending nodes.

For instance, in Figure 2.1.1, $\bullet p_1 = \{t_2\}$, $\bullet t_2 = \{p_2\}$, $p_2 \bullet = \{t_2, t_3\}$ and $t_3 \bullet = \{p_3\}$. This *PN* does not have any ending node. The ending nodes are represented in Figure 2.1.2.

The transitions in a *PN* can be classified based on its input places, as shown in the next definition:

Definition 2.1.8 *Types of transitions based on its input places.*

1. ***Join transitions.*** *A transition is named a **join transition** if it has more than one input place. The set of join transitions is $T_J = \{t_i \in T | \, |\bullet t_i| > 1\}$.*

2. ***Single input transitions.*** *A transition is named a **single input transition** if it has only one input place. The set of single input transitions is $T_S = \{t_i \in T | \, |\bullet t_i| = 1\}$.*

It is clear from the previous definition that the set of join transitions and single input transitions are a partition of T, i.e. $T_J \cap T_S = \emptyset$ and $T_J \cup T_S = T$.

Similarly to transitions, places can also be classified based on its input transitions in:

Definition 2.1.9 *Types of places based on its input transitions.*

1. ***Attribution places.*** *A place is named an* ***attribution place*** *if it has more than one input transition. The set of attribution places is* $P_A = \{p_j \in P|\ |\bullet p_j| > 1\}$

2. ***Single input places.*** *A place is named a* ***single input place*** *if it has only one input transition. The set of single input places is* $P_S = \{p_j \in P|\ |\bullet p_j| = 1\}$

It is also clear that the set of attribution places and single input places are a partition of P, i.e. $P_A \cap P_S = \emptyset$ and $P_A \cup P_S = T$.

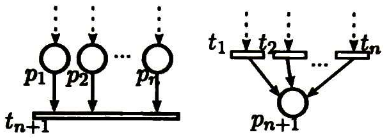


Figure 2.1.3: A join transition and an attribution place.

In Figure 2.1.3, transition t_{n+1} is a join transition. Similarly, place p_{n+1} is an attribution place.

Transitions can also be classified by its output places:

Definition 2.1.10 *Types of transitions based on its output places.*

1. ***Ending transitions.*** *A transition is named an* ***ending transition*** *if it does not have any output place. The set of ending transitions is* $T_E = \{t_i \in T|\ t_i\bullet = \emptyset\}$

2. ***Attributing transitions.*** *A transition is named* ***attributing transition*** *if at least one of its output places is an attribution place. The set of attributing transitions is* $T_A = \{t_i \in T|\ \exists p_j \in t_i\bullet,\ p_j \in P_A\}$

Finally, places can also be classified by its output transitions:

Definition 2.1.11 *Types of places based on its output transitions.*

1. ***Ending place.*** *A place is named an* ***ending place*** *if it does not have any output transitions. The set of ending places is* $P_E = \{p_j \in P|\ p_j\bullet = \emptyset\}$

2. ***Join-input place.*** *A place is named a* ***join-input place*** *if at least one of its output transitions is a join transition. The set of join-input places is* $P_J = \{p_j \in P|\ \exists t_i \in p_j\bullet,\ t_i \in T_J\}$

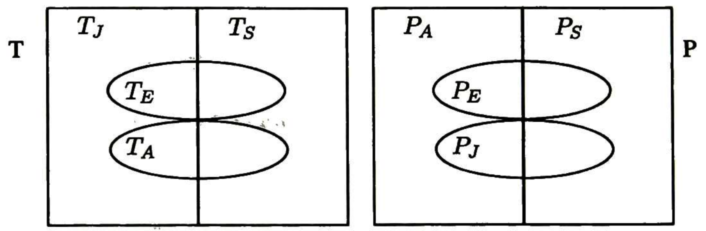


Figure 2.1.4: Classification of transitions and places.

The classification of transitions and places is presented as a Venn diagram in Figure 2.1.4. There exists some basic classes of *PN*. These classes are defined based on its structure.

Definition 2.1.12 *A PN is said to be*

- ***Join Free (JF)*** *if:*

$$\forall t \in T, |\bullet t_i| \leq 1$$

- ***Attribution-Free (AF)*** *if:*

$$\forall p \in P, |\bullet p_i| \leq 1$$

- ***Join-Attribution-Free (JAF)*** *if:*

$$\forall p \in P, |\bullet p_i| \leq 1 \,\&\, \forall t \in T, |\bullet t_i| \leq 1$$

- ***State machine*** *or* ***S-net*** *[14] if:*

$$\forall t \in T, |\bullet t| = |t\bullet| = 1$$

  Marked graph *or* ***T-net*** *[14] if:*

$$\forall p \in P, |\bullet p| = |p\bullet| = 1$$

- ***Free choice (FC)*** *[14] if:*

$$\forall p_1, p_2 \in P \text{ either } p_1 \bullet \cap p_2 \bullet = \emptyset \text{ or } p_1 \bullet = p_2 \bullet$$

In Figure 2.1.5 a), the dashed arc makes the *PN* not to be a state machine. If it was removed, then the *PN* would be a state machine. Similarly in Figure 2.1.5 b), the dashed arc makes the *PN* not to be a marked graph. Similarly if it was removed, the *PN* would be a marked graph.

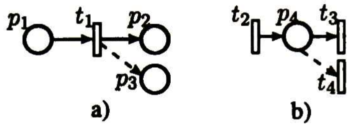


Figure 2.1.5: The state machine and marked graph case.

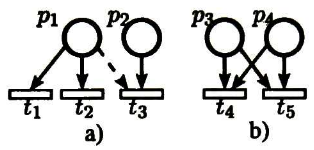


Figure 2.1.6: The Free-Choice case.

It is clear that S-nets and T-nets are subclasses of Free choice (*FC*). The *PN* in Figure 2.1.6, a) is not a Free choice while b) is. If the dashed arc in Fig. 2.1.6 a) was removed, then both cases would be *FC*.

Definition 2.1.13 *A Petri net system or simply a Petri net (PN) is the tuple* $\langle N, \mathbf{m}_0 \rangle$ *where* N *is the Petri net structure and* $\mathbf{m}_0$ *is the initial token distribution or initial marking, where a marking* $\mathbf{m} : P \to \mathbb{Z}_{\geq 0}^{|P|}$ *is a vector representing the number of tokens inside each place. The notation* $\mathbf{m}(p_j)$ *denotes the marking in the place* p_j.

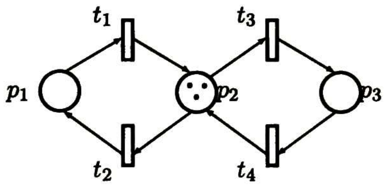


Figure 2.1.7: A simple *PN* system.

A transition t_i is said to be enabled at a marking $\mathbf{m}_k$ if for every place $p_j \in \bullet t_i$, $\mathbf{m}_k(p_j) \geq \mathbf{Pre}(p_j, t_i)$; if the transition t_j is enabled at a marking $\mathbf{m}_k$, then t_j can be *fired* reaching a new marking $\mathbf{m}_{k+1}$. The notation $\mathbf{m}_k$ represents the $k-$th (*ordered*) element in a sequence of markings. The firing of a transition t_i removes $\mathbf{Pre}(i, j)$ tokens from each place $p_j \in \bullet t_i$ and adds $\mathbf{Post}(i, h)$ tokens in each place $p_h \in t_i \bullet$.

In a similar way, let $\sigma = t_{i_1}t_{i_2}...t_{i_k}$ be a firing transitions sequence leading from $\mathbf{m}_0$ to some $\mathbf{m}_{k+1}$ in the following way:

$$\mathbf{m}_0 \xrightarrow{t_{i_1}} \mathbf{m}_1 \xrightarrow{t_{i_2}} \mathbf{m}_2 \xrightarrow{t_{i_3}} ... \xrightarrow{t_{i_k}} \mathbf{m}_{k+1}$$

where $t_{i_1}, t_{i_2}, ..., t_{i_k} \in T$ and the notation $\mathbf{m}_k \xrightarrow{t_{i_k}} \mathbf{m}_{k+1}$ means that from the marking $\mathbf{m}_k$ transition t_{i_k} is fired, reaching a new marking $\mathbf{m}_{k+1}$. This discrete evolution can also be computed by the equation

$$\mathbf{m}_{k+1} = \mathbf{m}_0 + \mathbf{C}\vec{\sigma}$$

which is known as the fundamental *PN* equation, where $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$ is the incidence matrix and $\vec{\sigma}$ is an $|T|$-dimensional vector with the i-th entry representing the number of occurrences of t_i in σ. Vector $\vec{\sigma}$ is known as the parikh vector of a sequence σ.

For instance, in the *PN* in Figure 2.1.7, transitions t_2 and t_3 are enabled 3 times, while transitions t_1 and t_4 are not enabled. If transition t_2 is fired once, then the marking $\mathbf{m}_1 = \begin{bmatrix}1 & 2 & 0\end{bmatrix}^T$ is reached. With marking $\mathbf{m}_1$ transition t_1 is now enabled and therefore it is capable of being fired. The firing of transition t_2 and the marking $\mathbf{m}_1$ can be computed with equation (2.1.1) as follows:

$$\mathbf{m}_1 = \begin{bmatrix}0\\3\\0\end{bmatrix} + \begin{bmatrix}-1 & 1 & 0 & 0\\1 & -1 & -1 & 1\\0 & 0 & 1 & -1\end{bmatrix}\begin{bmatrix}0\\1\\0\\0\end{bmatrix} = \begin{bmatrix}1\\2\\0\end{bmatrix}$$

Left and right rational annulers of $\mathbf{C}$ are called P-and T-flows respectively. When the elements of the annuler are either positive or zero they are called P-Semiflow and T-Semiflow.

Definition 2.1.14 *A **T-Semiflow** is a rational solution $X \geq 0$ for*

$$\mathbf{C} \cdot X = 0$$

Definition 2.1.15 *A **P-Semiflow** is a rational solution $Y \geq 0$ for*

$$Y^T \cdot \mathbf{C} = 0.$$

The notation $Y > 0$ ($Y \geq 0$) means that the vector Y is greater than (greater or equal than) zero in each entry, i.e. the i-th entry $Y(i) > 0$ ($Y(i) \geq 0$), $\forall i = 1, 2, ..., |Y|$.

Definition 2.1.16 *If there exists a T-Semiflow $X > 0$, then the PN is said to be consistent.*

If there exists a P-Semiflow $Y > 0$, then the PN is said to be conservative.

Another important structure of the *PN* are siphons and traps, formally defined next.

Definition 2.1.17 *A set of places $P_S \subseteq P$ of a PN is a **siphon** if:*

$$\bullet P_S \subseteq P_S \bullet$$

*A siphon P_S is called **proper siphon** if $P_R \neq \emptyset$, i.e. it is not the empty set.*

Definition 2.1.18 *A set of places $P_T \subseteq P$ of a PN is a **trap** if:*

$$P_T \bullet \subseteq \bullet P_T$$

*A trap P_T is called **proper trap** if $P_T \neq \emptyset$, i.e. it is not the empty set.*

The characterization of every siphon and trap of a *PN* has been proved of NP complexity, however, an interesting approach for generating a family of siphons and traps is presented in [62]. It is based on the synthesis of a generator of siphons or traps.

The following proposition shows a trap generator.

Proposition 2.1.19 *[62]*

Let $\langle N, \mathbf{m_0} \rangle$ be a PN, where $\mathbf{Pre} = W^-$ and $\mathbf{Post} = W^+$. Also, let W_θ^- and W_θ^+ be two non negative $|P| \times |T|$ matrices, such that:

$$\begin{array}{c} W_\theta^+[p,t] = 0 \Longleftrightarrow W^+ = 0 \\ W_\theta^-[p,t] = 0 \Longleftrightarrow W^- = 0. \end{array} \tag{2.1}$$

Also, let $W_\theta = W_\theta^+ - W_\theta^-$. If $\mathbf{y}$ is an integer non-negative solution of

$$\mathbf{y}^T \cdot W_\theta \geq 0$$

then $\langle \mathbf{y} \rangle$ is a trap of N.

When infinite servers semantics is used, if the input places $\bullet t_i$ of a transition t_i have enough tokens, the transition may be fired as many times as possible at once. The number of times that a transition can be fired at a given marking is known as its *enabling degree*. This concept will be formally defined later.

As a notation, for any vector $\mathbf{y} \in \mathbb{R}^{|P|}$ ($\mathbf{x} \in \mathbb{R}^{|T|}$), $\mathbf{y}(p_j)$ ($\mathbf{x}(p_j)$) represents the value of the vector in its corresponding j-th (i-th) entry associated to place p_j (transition t_i).

Definition 2.1.20 *[14] The* ***Support*** *of a vector* $\mathbf{x} \in \mathbb{R}^{|T|}$ $(\mathbf{y} \in \mathbb{R}^{|P|})$*, denoted by* $\langle \mathbf{x} \rangle$ $(\langle \mathbf{y} \rangle)$ *is the set of all the transitions (places) satisfying* $\mathbf{x}(t_i) \neq 0$ $(\mathbf{y}(p_j) \neq 0)$*, where* $\mathbf{x}(t_i)$ $(\mathbf{y}(p_j))$ *is the value of* $\mathbf{x}$ $(\mathbf{y})$ *in the corresponding position of* t_i (p_j)

$$\langle \mathbf{x} \rangle = \{t_i \in T | \mathbf{x}(t_i) \neq 0\}$$
$$(\langle \mathbf{y} \rangle = \{p_j \in P | \mathbf{y}(p_j) \neq 0\}).$$

For instance, let $\mathbf{m} = \begin{bmatrix} 0 & 1 & 2 \end{bmatrix}^T$ be a marking of the *PN* in Figure 2.1.7. The support of $\mathbf{m}$ is $\langle \mathbf{m} \rangle = \{p_2, p_3\}$.

As a notation, an **elementary vector** is a vector $\mathbf{e}_h \in \{0,1\}^n$ having $n-1$ zeros and only one entry equal to one in the $h-th$ position, of the form $\mathbf{e}_h = \begin{bmatrix} 0 & ... & 0 & 1 & 0 & ... & 0 \end{bmatrix}^T$ The elementary vectors are important since they can represent in vectorial form a place or a transition. This is formally defined next.

Definition 2.1.21 *A place* p_j *is said to be associated to an elementary vector* $\mathbf{e}_j \in \{0,1\}^{|P|}$ *and represented by* $\mathbf{e}_j \sim p_j$*, since*

$$\langle \mathbf{e}_j \rangle = \{p_j\}$$

A transition t_i *is said to be associated to an elementary vector* $\mathbf{e}_i \in \{0,1\}^{|T|}$ *and represented by* $\mathbf{e}_i \sim t_i$*, since*

$$\langle \mathbf{e}_j \rangle = \{t_i\}.$$

2.1.2 Continuous Petri nets

A continuous Petri net, introduced in [21] and further studied in [60], [63], [64], [61] and [65], is a relaxation of the discrete *PN* model, where a transition can be fired in any real amount between zero and its *enabling degree*. As a consequence, the number of tokens in each place can be a positive real number. This model is formally defined below.

Definition 2.1.22 *A continuous Petri net, is the tuple* $\langle N, \mathbf{m}_0 \rangle$ *where* N *is the PN structure and* $\mathbf{m}_0$ *is the initial token distribution, where* $\mathbf{m} : P \to \mathbb{R}^{|P|}_{\geq 0}$ *is a vector representing the number of tokens inside each place.*

In a continuous Petri net a transition $t_i \in T$ is enabled if and only if for every $p_j \in \bullet t_i$, $\mathbf{m}(p_j) > 0$. The enabling degree of a continuous transition t_i is given by the equation:

$$enab(t_i, \mathbf{m}) = \min_{p_j \in \bullet t_i} \left\{ \frac{\mathbf{m}(p_j)}{\mathbf{Pre}[p_j, t_i]} \right\}$$

An enabled transition t_i at a marking $\mathbf{m}$ can be fired in any real amount $0 \leq \sigma_i \leq enab(t_i, \mathbf{m})$ leading to a marking $\mathbf{m}'$ that can be computed with the fundamental continuous *PN* equation

$$\mathbf{m}' = \mathbf{m} + \mathbf{C}\vec{\sigma} \qquad (2.2)$$

where $\vec{\sigma} = \begin{bmatrix}\sigma_1 & \dots & \sigma_{|T|}\end{bmatrix}^T$, represents the firing amounts of each transition.

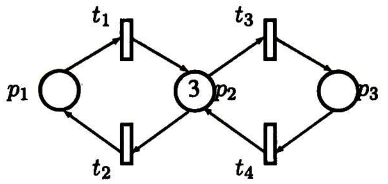


Figure 2.1.8: A continuous *PN* system.

In figure 2.1.8 a continuous Petri net system (*CPN*) is presented. Transitions t_2 and t_3 are $3-$ *enabled.* If transition t_2 is fired 1.3 times and transition t_3 is fired 0.9 times, the *CPN* reaches a new marking

$$\mathbf{m}_1 = \begin{bmatrix}0\\3\\0\end{bmatrix} + \begin{bmatrix}-1 & 1 & 0 & 0\\1 & -1 & -1 & 1\\0 & 0 & 1 & -1\end{bmatrix}\begin{bmatrix}0\\1.3\\0.9\\0\end{bmatrix} = \begin{bmatrix}1.3\\0.8\\0.9\end{bmatrix}$$

2.1.3 Continuous timed Petri nets

In order to include the notion of time in the continuous *PN* models, a function $\lambda : T \rightarrow \mathbb{R}_{\geq 0}$ is introduced. This function assigns to each transition a positive value representing the maximum number of tokens that can flow through the transition per time unit and per server [51] and it is called the firing rate of the transition. Hence, a continuous timed Petri net (*ContPN*) is formally defined as:

Definition 2.1.23 *A continuous timed Petri net system is the tuple* $\langle N, \lambda, \mathbf{m}_0\rangle$ *where* N *is the Petri net structure;* $\lambda : T \rightarrow \mathbb{R}_{\geq 0}$ *are the firing rates of the transitions and* $\mathbf{m}_0$ *is the initial marking of the net.*

The marking evolution of the *ContPN* is now modelled with the equation

$$\mathbf{m}(\tau) = \mathbf{m}_0 + \mathbf{C}\vec{\sigma}(\tau) \qquad (2.3)$$

where $\vec{\sigma}(\tau)$ represents the firings amounts of each transition at time τ.

Since the marking evolution is now time dependent, it is possible to obtain the derivative of Eq. (2.3) with respect to time τ to obtain the dynamic equation

$$\dot{\mathbf{m}}(\tau) = \mathbf{C}\dot{\vec{\sigma}}(\tau) = \mathbf{C}f(\tau), \quad \mathbf{m}(0) = \mathbf{m}_0. \tag{2.4}$$

The relaxation of the discrete model used in this work considers infinite server semantics, therefore the flow $f(\tau)$ through a transition t_i is computed as the product of $\lambda(t_i)enab(t_i, \mathbf{m})$.

As a notation, the firing rate of a transition is represented by $\lambda(t_i) = \lambda_i$, and the vector $\lambda = \begin{bmatrix}\lambda_1 & \dots & \lambda_{|T|}\end{bmatrix}^T$

The computation of a transition's flow requires the computation of its enabling degree function: $enab(t, \mathbf{m})$; however, this function requires the min operator, which leads to the concept of *configurations*.

Definition 2.1.24 *A configuration $\mathcal{C}_k$ of a net N is a set of (p,t) arcs, one per transition, covering the set T of transitions [16].*

A configuration is represented by the set of $|T|$ arcs covering T, and denoted by $\mathcal{C}_k$. If an arc $(p_j, t_i) \in \mathcal{C}_k$, then it is said that the place p_j constrains transition t_i or that transition t_i is constrained by the place p_j.

The flow through the transitions can be written as $f(\mathbf{m}) = \Lambda\Pi(\mathbf{m})\mathbf{m}$ [51] where $\Lambda = diag(\lambda)$ and $\Pi(\mathbf{m})$ is the configuration matrix at marking $\mathbf{m}$, defined by

$$\Pi(\mathbf{m})[i,j] = \begin{cases} \dfrac{1}{\mathbf{Pre}[p_j, t_i]} & \text{if } p_j \text{ constrains } t_i \\ 0 & \text{otherwise.} \end{cases} \tag{2.5}$$

If more than one place is constraining the flow of a transition at a given marking, any of them can be used, but only one is taken. The number of configurations in a *ContPN* is upper-bounded by $\prod_{t_i \in T} |\bullet t_i|$, i.e. there exists an exponential number of configurations.

Example 2.1.25 *Consider the ContPN in Figure 2.1.9, transition t_3 is a join transition. Its flow can be either constrained by place p_2 or place p_4. With the marking $\mathbf{m}_0 = \begin{bmatrix}0 & 3 & 0 & 0\end{bmatrix}^T$ its flow is actually constrained by place p_4, since $\mathbf{m}_0(p_4) < \mathbf{m}_0(p_2)$. Let $\mathbf{m}_1 = \begin{bmatrix}0.5 & 0.5 & 0 & 1.5\end{bmatrix}^T$ be the marking of the ContPN after the firing sequence $\vec{\sigma} = \begin{bmatrix}1.5 & 2 & 0\end{bmatrix}^T$ Now, $\mathbf{m}(p_2) < \mathbf{m}(p_4)$ and the flow of transition t_3 is now constrained by place p_2. The configuration matrices for this example are:*

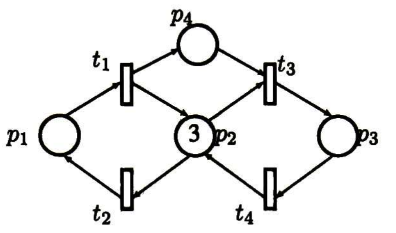


Figure 2.1.9: A continuous *PN* system with join transitions.

$$\Pi(\mathbf{m}_0) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \quad \Pi(\mathbf{m}_1) = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

A *ContPN* with infinite servers semantics (or just *ContPN*), as previously discussed, can be modelled by a *SLS* with polyhedral regions determined by the configurations [51], [52]. A region, denoted by $\mathcal{R}_k$, is a set of markings reachable from the initial marking $\mathbf{m}_0$, such that they have the same configuration matrix.

The configuration matrix for the region $\mathcal{R}_k$ will be denoted by Π_k. Thus, (2.4) can be rewritten for the k-th region as:

$$\dot{\mathbf{m}} = \mathbf{C}\Lambda\Pi_k\mathbf{m}. \tag{2.6}$$

Source transitions are not defined in the infinite servers semantics, because no place is constraining its flow. One way to define constant flow transition (fixed flow source transition) is to model a transition t_i together with a place p_j such that $\bullet t_i = t_i \bullet = p_j$ and $\mathbf{Pre}(p_j, t_i) = \mathbf{Post}(t_i, p_j) = 1$ as in Fig. 2.1.10. Next definition is introduced to ensure that (2.6) can model the flow $\forall t_i \in T$.

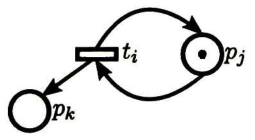


Figure 2.1.10: A well defined source transition.

Definition 2.1.26 *In a ContPN, a transition $t_i \in T$ is named **well defined** if $|\bullet t_i| \geq 1$, i.e. it has at least one input place.*

Definition 2.1.27 *A ContPN is named **well defined** if $\forall t_i \in T$, t_i is well defined.*

Through this work the *ContPN* are considered to be well defined.

2.2 Controlability and Control Laws

The results presented in this section are taken from [35], [34] and [66].

In order to apply a control action in Equation (2.6), a subtracting term u_i, $0 \leq u_i \leq f_i$, is added to every transition t_i to indicate that its flow can be reduced. This is adequate because it captures the real behaviour in which the maximum machine's throughput can only be reduced. Thus the controlled flow of transition t_i, becomes $w_i = f_i - u_i$. In this sense, f_i is named the natural flow of the transition t_i and w_i is its controlled flow.

Now, introducing $f = \Lambda\Pi_k \cdot \mathbf{m}$ and u in (2.6) the forced state equation is

$$\begin{gathered}\overset{\bullet}{\mathbf{m}} = \mathbf{C}\left[f - u\right] = \mathbf{C}w \\ 0 \leq u_i \leq f_i.\end{gathered}$$

Definition 2.2.1 *If the inequality*

$$0 \leq u_i \leq f_i \tag{2.7}$$

*holds, then the control action is named **suitably bounded**.*

A simplified version of the state equation can be obtained writing the input vector as

$$u = I_u \Lambda \Pi_k \cdot \mathbf{m},$$

where $I_u = diag\left(I_{u1}, \ldots, I_{u|T|}\right)$ and $0 \leq I_{u_i} \leq 1$. Then the matrix $I_c = I - I_u$ is constructed and the state equation can be rewritten as

$$\overset{\bullet}{\mathbf{m}} = \mathbf{C} I_c f = \mathbf{C}w. \tag{2.8}$$

Notice that $0 \leq I_{c_i} \leq 1$ is the equivalent condition of Eq 2.7 (suitably bounded). In this way, the diagonal values I_{c_i} represent the opening proportion of a transition's natural flow.

As in the case of the discrete *PN*, the set of all reachable markings from $\mathbf{m}_0$ is called the reachability set [67] and it is denoted by $RS\left(N, \mathbf{m}_0\right)$.

2.2.1 Observability in *ContPN*

Some important definitions regarding the observability in *ContPN* will be provided next. An interested reader can find more information in [44], [45], [46], [43], [16].

The output matrix of a *ContPN* is represented by the matrix S. This matrix S is composed only by transposed elementary vectors, showing which places of the *ContPN* have a sensor. This is formally defined next.

Definition 2.2.2 *Let S be the output matrix of a ContPN. A place $p_j \in P$ is said to be measured if there exists a row h in S, represented by $S(h, \bullet)$ such that:*

$$S(h, \bullet) = \mathbf{e}_j$$

where $\mathbf{e}_j \sim p_j$ (i.e. $\mathbf{e}_j \in \{0, 1\}^{|P|}$ and $\langle \mathbf{e}_j \rangle = p_j$)

Definition 2.2.3 *Let S be the output matrix of a ContPN.*

$$P_M(S) = \{p_j | \mathbf{e}_j^T \textit{is a row of matrix } S\}$$

is the set of measured places with the output matrix S.

Definition 2.2.4 *[46] A ContPN system $\langle N, \mathbf{m}_0, \lambda \rangle$ is observable in infinitesimal time if it is always possible to compute its initial state $\mathbf{m}_0$ in any time interval $\tau \in [0, \epsilon]$, $\forall \epsilon > 0$, by measuring only a set of places $P_M \subseteq P$.*

It is important to recall that the aim of this work is to develop a strategy to construct an output matrix S such that the *ContPN* is observable in polynomial time.

2.2.2 Controllability in *ContPN*

Once it is guaranteed that a *ContPN* is observable with an appropriate sensor placement, the second objective of this work is to deal with the construction of observers. Then, it is also interesting to use the observer integrated with a control strategy. Therefore, in this subsection some basic concepts of controllability in *ContPN* will be briefly recalled.

Since *ContPN* are positive *SLS* then the *LS* controllability concept cannot be applied to this case. Instead, the controllability definition presented in [66] is used.

Definition 2.2.5 *The equivalence relation β is defined as $\mathbf{m}_1 \beta \mathbf{m}_2$ iff $B^T\mathbf{m}_1 = B^T\mathbf{m}_2$, where B is an algebraic basis of P-flows.*

The system admissible states set is the equivalent class of the initial marking $Class\,(\mathbf{m}_0)$ under the relation β. The set of interior points of $Class\,(\mathbf{m}_0)$ is defined as

$$intClass(\mathbf{m}_0) = \{\mathbf{m} \in Class(\mathbf{m}_0) \wedge \forall p_i \in P,\ \mathbf{m}(p_i) \neq 0\}.$$

Definition 2.2.6 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a ContPN. The system is said to be fully controllable with bounded input ($BIFC$) if there is an input such that for any two markings $\mathbf{m}_1, \mathbf{m}_2 \in Class\,(m_0)$, it is possible to transfer the marking from $\mathbf{m}_1$ to $\mathbf{m}_2$ in finite or infinite time, and the input fulfils the suitably bounded condition along the trajectory.*

The system is said to be controllable with bounded input (BIC) over $S \subseteq Class\,(\mathbf{m}_0)$ if there is an input such that for any two markings $\mathbf{m}_1, \mathbf{m}_2 \in S$, it is possible to transfer the marking from $\mathbf{m}_1$ to $\mathbf{m}_2$ in finite or infinite time, and the input fulfils the suitably bounded condition along the trajectory.

It is important to remark that controllability is a structural property for the *ContPN*. The following theorem is valid only when all transitions in a *ContPN* are controllable.

Theorem 2.2.7 *A ContPN is BIFC iff the structure of the net N is consistent and there do not exist empty siphons at any marking in $Class(\mathbf{m}_0)$.*

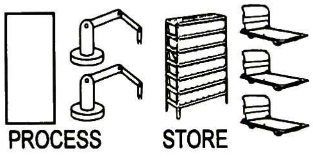


Figure 2.2.1: A manufacturing process.

Example 2.2.8 *Consider the manufacturing process depicted in Figure 2.2.1. The ContPN of this manufacturing process is presented in Figure 2.2.2.*

Places p_1 and p_2 represent a production machine, places p_3 and p_4 represent a store (buffer) and places p_5 and p_6 represent a consumption machine. Marking m_1 represents inactive production resources, m_2 represents active production resources, m_3 represents free cells of storage (10 of them), m_4 represents occupied cells of storage, m_5 represents free carriers and m_6 represents occupied carriers.

Transition t_1 puts a free production resource to work, transition t_2 produces a product if there is a free cell and release a production resource, transition t_3 consumes a product from the store if there is a free carrier, releasing a cell, and transition t_4 releases a carrier.

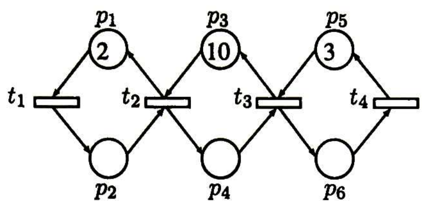


Figure 2.2.2: *ContPN* model of a manufacturing process.

This ContPN has four different configurations and therefore there are four different propagation matrices. The configurations index corresponds to the following sets of places and transitions:

$\mathfrak{R}_1 = \{(p_1, t_1), (p_2, t_2), (p_4, t_3), (p_6, t_4)\}$
$\mathfrak{R}_2 = \{(p_1, t_1), (p_2, t_2), (p_5, t_3), (p_6, t_4)\}$
$\mathfrak{R}_3 = \{(p_1, t_1), (p_3, t_2), (p_4, t_3), (p_6, t_4)\}$
$\mathfrak{R}_4 = \{(p_1, t_1), (p_3, t_2), (p_5, t_3), (p_6, t_4)\}$.

The incidence matrix of this ContPN is given by:

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

Since there exists $x = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \end{bmatrix}^T$ such that $\mathbf{C}x = 0$, *then the ContPN is controllable.*

Another important concept is the equilibrium point, formally presented next.

Definition 2.2.9 *Let* $\langle N, \lambda, \mathbf{m}_0 \rangle$ *be a ContPN. Also, let* $\mathbf{m}_{eq} \in RS(N, \mathbf{m}_0)$ *and* $0 \leq I_{c_r}[i, i] \leq 1$. *If*

$$\overset{\bullet}{\mathbf{m}} = \mathbf{C} I_{c_{eq}} \Lambda \Pi (\mathbf{m}_{eq}) \cdot \mathbf{m}_{eq} = 0,$$

then $\left(\mathbf{m}_{eq}, I_{c_{eq}}\right)$ *is called an equilibrium point and* $\mathbf{m}_{eq}$ *an equilibrium marking. Also, the steady state flow for* $\left(\mathbf{m}_{eq}, I_{c_{eq}}\right)$ *is* $w_{ss}\left(\mathbf{m}_{eq}, I_{c_{eq}}\right) = I_{c_{eq}} \Lambda \Pi (\mathbf{m}_{eq}) \cdot \mathbf{m}_{eq}$.

An equilibrium point represents a state in which the system can be maintained using the defined control action. According to [51], these points can represent states of maximum system throughput. Thus, an important problem is that given an initial marking $\mathbf{m}_0$ and a required target marking $\mathbf{m}_r$ (for instance, markings allowing maximum system throughput), to obtain a control law leading the *ContPN* state from $\mathbf{m}_0$ to $\mathbf{m}_r$. This problem is formally defined as follows.

Definition 2.2.10 *Let* $(N, \lambda, \mathbf{m}_0)$ *be a ContPN. Then the* **Regulation Control Problem** *in* $(\mathbf{m}_r, I_{c_r})$ *(written as* $RCP(\mathbf{m}_r, I_{c_r})$ *deals with the computation of a control law* $I_c(\tau)$, $0 \leq \tau < \tau_f$ *feasible in the ContPN such that* $\mathbf{m}(\tau_{ss}) = \mathbf{m}_r$ *and* $I_c(\tau_{ss}) = I_{c_r}$, $\forall \tau_{ss} \geq \tau_f$.

2.2.3 Control laws

Control laws for *ContPN* are widely studied in the literature. For instance in [68] a fuzzy proportional controller is introduced. However, it only guarantees a bounded error between the target and the actual final marking of the system. In [32] implicit and explicit Model Predictive Control are presented as well as a comparison between them. Unfortunately the computational time becomes prohibited when the system is large on the number of nodes.

In [33] a control strategy is presented, assigning piecewise constant flows to transitions in order to reach the target state. The idea presented is to use linear programming to drive the system through a linear trajectory and then to include intermediate states to improve time performance. Nevertheless, an important drawback of this approach is that the problem of defining the intermediate states is exponential.

In this work, the idea presented in [26] and [69] will be briefly reviewed.

In order to deal with the *ContPN* it is necessary to measure the cumulative transition's flow, therefore some extra places are added to the *ContPN*. The following definition shows how these places are included in an extended *ContPN*.

Definition 2.2.11 *Let* $\langle N, m_0, \lambda \rangle$ *be a ContPN, where* $N = (P, T, \mathbf{Pre}, \mathbf{Post})$. *Its extension is defined by xContPN* $= \langle N_x, \mathbf{m}_{0_x}, \lambda \rangle$, *where:*

$N_x = (P \cup P_a, T, \mathbf{Pre}, \mathbf{Post} \cup Post_a)$,

$|P_a| = |T|$, $\mathbf{m}_{0_x} = \begin{bmatrix} \mathbf{m}_0 & 0_{|T|} \end{bmatrix}^T$

$Post_a = \{(t_i, p_{a_i}) \, | \forall t_i \in T \text{ and } \forall p_{a_i} \in P_a\}$.

Then the incidence matrix of xContPN is $C_x = \begin{bmatrix} C & I_{|T|} \end{bmatrix}^T$

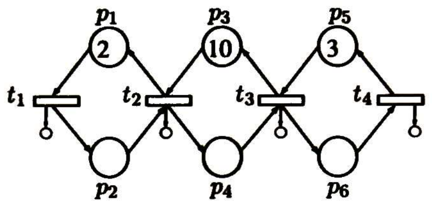


Figure 2.2.3: An extended *ContPN*.

Since $\Pi_x(\mathbf{m}_x) = \begin{bmatrix} \Pi(\mathbf{m}) & 0_{|T|\times|T|} \end{bmatrix}$, then the state equation of *xContPN* is

$$\overset{\bullet}{\mathbf{m}}_x = \begin{bmatrix} \overset{\bullet}{\mathbf{m}} \\ \overset{\bullet}{\mathbf{m}}_a \end{bmatrix} = \begin{bmatrix} Cw \\ w \end{bmatrix} \qquad (2.9)$$
$$\mathbf{m}(0) = \mathbf{m}_0, \mathbf{m}_a(0) = 0.$$

Notice that the extension has the same dynamics over the marking as the original system and the marking over extra places is the transition flow integration, then it can only increase. In fact, if the *ContPN*is live, then by construction the *xContPN* is also live.

As an illustration about the extension concept refer to Figure 2.2.3.

Another concept that will be used herein is the minimum Parikh vector. A Parikh vector σ, $\forall i\ \sigma[i] > 0$, due to $\mathbf{m}$ must fulfil $\mathbf{m} = \mathbf{m}_0 + \mathbf{C}\sigma$, where $\mathbf{m}_0$ is the initial marking and $\mathbf{m}$ is the reached marking. When a *ContPN* is live it contains right annulers of $\mathbf{C}$ (T-flows), therefore σ has several solutions such that $\mathbf{m} - \mathbf{m}_0 = C\sigma$. In order to fix a unique solution it is chosen, for convenience, the one which involves less marking transit, i.e. the smallest vector solution. This can be done by solving the next linear programming problem:

$$\sigma_{min} = \min \sigma \text{ such that } \mathbf{C}\sigma = \mathbf{m}_r - \mathbf{m}_0 \text{ and } \sigma \geq 0 \qquad (2.10)$$

where $\mathbf{m}_r \in RS(N, \mathbf{m}_0)$ is a marking that is required to reach.

2.2.3.1 A solution to the regulation control problem

A solution to the regulation control problem for *ContPN*s is presented next [[26]].

Theorem 2.2.12 *Let* $(N, \mathbf{m}_0, \lambda)$ *be a* BIC *over* $int\,(Class\,(\mathbf{m}_0))$ *ContPN and* $xContPN =$ $(N_x, \mathbf{m}_{0_x}, \lambda)$ *be its extension. If* $(I_{c_r}, \mathbf{m}_r)$ *is an arbitrary equilibrium point such that* $\mathbf{m}_r \in$

int $(Class\,(\mathbf{m}_0))$, *then there exists* $I_c(\tau)$, $0 \le \tau \le \tau_f$, *feasible in the ContPN such that* $\mathbf{m}(\tau_{ss}) = \mathbf{m}_r$, $I_c(\tau_{ss}) = I_{c_r}, \forall \tau_{ss} \ge \tau_f$.

$$I_{c_i} = \begin{cases} 1 & \textit{if}\, \mathbf{m}_a\,[i] < \sigma_r\,[i] \\ 0 & \textit{otherwise} \end{cases} \tag{2.11}$$

where $\mathbf{C}\sigma_r = \mathbf{m}_r - \mathbf{m}_0$.

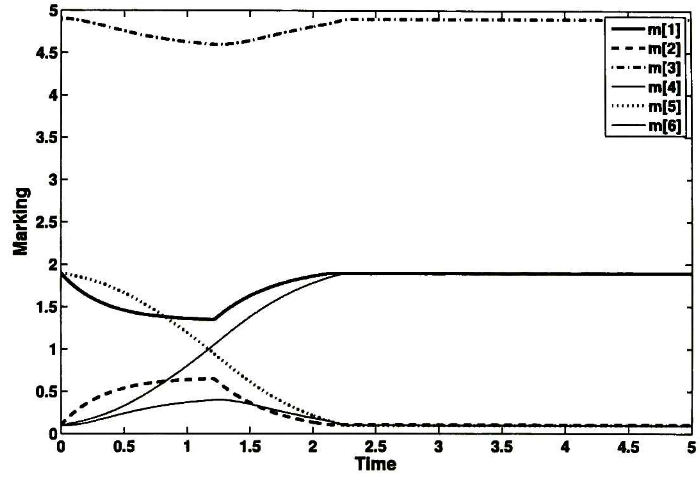


Figure 2.2.4: The controlled marking evolution of a *ContPN.*

Figure 2.2.4 shows the marking evolution of the system in Figure 2.2.2 using the control law presented in Theorem 2.2.12 and with a target marking $\mathbf{m}_r = \begin{bmatrix}2 & 0 & 10 & 0 & 0 & 3\end{bmatrix}^T$ This is computed considering that the marking of all places is known $\forall\tau$.

2.3 Linear systems

Since each configuration in the *ContPN* defines a linear system, some basic concepts that will be used through this work will be briefly recalled. An interested reader may consult [40] and [39].

A time invariant linear system (*LS*) $\Sigma(A, B, S)$ is represented by

$$\overset{\bullet}{x}(\tau) = Ax(\tau) + Bu(\tau), \;\; y(\tau) = Sx(\tau)$$

where $x \in \mathbb{R}^a$ is the state vector, $u \in \mathbb{R}^b$ is the control input, $y \in \mathbb{R}^s$ the output signal (sensors in the systems which are a linear combination of the state vector), and A, B and S are constant matrices of appropriate dimensions. A *LS* is a differential equation and its evolution is completely defined with an initial condition $x(\tau_0) = x_0$, where τ_0 is the initial time (usually 0).

The evolution of the state of the *LS* $\Sigma(A, B, S)$ is given by the equation [40]

$$x(\tau) = e^{A\tau}x_0 + \int_0^{\tau} e^{A(\tau-\psi)}Bu(\psi)d\psi.$$

It is clear from the previous equation that if the initial condition $x(\tau_0)$ is known, then it is possible to determine the value for the state vector x $\forall\tau$. Unfortunately, in practical problems, the initial condition cannot always be exactly known. This leads to the problem of determining the initial condition of the state x of the *LS* only by the knowledge (or measurement) of the output variables within a finite time period. This problem is known as the **observability** problem. The observability property in a *LS* is formally defined next.

Definition 2.3.1 *[40] The system $\Sigma(A, B, S)$ is said to be observable at time τ_0 if there exists a finite time $\tau_1 > \tau_0$ such that for any state x_0 at time τ_0, the knowledge of the input $u_{[\tau_0,\tau_1]}$ and the output $y_{[\tau_0,\tau_1]}$ over the time interval $[\tau_0, \tau_1]$ suffices to determine the state x_0. Otherwise the system $\Sigma(A, B, S)$ is said to be unobservable at time τ_0.*

This problem has been addressed and is completely solved for *LS* [40], [39], [41]. The characterization for the observability problem in *LS* is presented in the next subsection.

2.3.1 Observability in *LS*.

With the following theorems it is possible to determine if a *LS* is observable.

Theorem 2.3.2 *The linear time-invariant system $\sum(A, B, S)$ is observable iff the observability matrix*

$$\mathbb{O} = \begin{bmatrix} S^T & (SA)^T & \dots & (SA^k)^T \end{bmatrix}^T \tag{2.12}$$

has full rank.

Example 2.3.3 *Take for instance a LS $\Sigma(A, B, S)$ with the following matrices*

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad S = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}$$

The observability matrix of the LS $\Sigma(A, B, S)$ is

$$\mathcal{O} = \begin{bmatrix} 1 & 0 & 0 \\ 1 & 2 & 3 \\ 30 & 36 & 42 \end{bmatrix}$$

which has full rank, therefore the LS $\Sigma(A, B, S)$ is observable.

It is also possible to determine if a *LS* is observable by analysing its $A-$invariant subspaces. Let $\mathcal{V}$ be a subspace of $\mathbb{R}^n$. $\mathcal{V}$ is said to be $A-$invariant if $A\mathcal{V} \subset \mathcal{V}$.

In Example 2.3.3, $span\{\begin{bmatrix}0.2320 & 0.5253 & 0.8187\end{bmatrix}^T\}$ is an $A-$invariant subspace, because

$$A\begin{bmatrix} 0.2320 \\ 0.5253 \\ 0.8187 \end{bmatrix} = \begin{bmatrix} -3.7386 \\ -8.4665 \\ -13.1944 \end{bmatrix} = -16.1168 \begin{bmatrix} 0.2320 \\ 0.5253 \\ 0.8187 \end{bmatrix}$$

The supremal A–invariant subspace contained in $\ker S$ is

$$\mathcal{N} = \bigcap_{i=1}^{n} \ker(SA^{i-1}).$$

The following theorem gives necessary and sufficient condition for observability in *LS* [70].

Theorem 2.3.4 *The LS $\Sigma\,(A, B, S)$ is observable if and only if the supremal A–invariant subspace contained in* $\ker S$ *is the trivial subspace, i.e.* $\mathcal{N} = 0$.

The subspace $\mathcal{N}$ is known as the unobservable subspace of the *LS* $\Sigma(A, B, S)$ and it is closely related to the observability matrix of the *LS* since it holds that $\mathcal{N} = \ker(\mathcal{O})$.

2.3.2 Observer design in *LS*.

Once that a *LS* shows the observability problem, it is possible to create a mathematical entity called observer, which actually computes the state vector of the *LS*.

There exist several approaches for the observers design in the literature. The most commonly used observer is the so called Luenberger observer [41] which is a copy of the *LS* $\Sigma(A, B, S)$ with an error correction term as described by the equation

$$\begin{aligned} \overset{\bullet}{\bar{x}}(\tau) &= A\bar{x}(\tau) + Bu(\tau) + L(y - \bar{y}) \\ \bar{y}(\tau) &= S\bar{x}(\tau). \end{aligned}$$

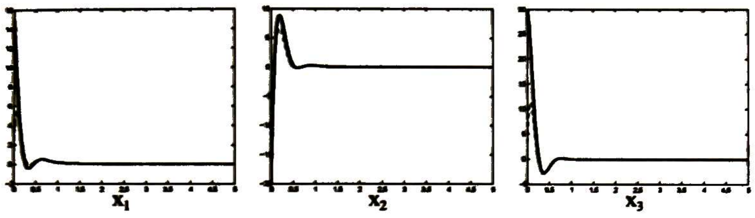


Figure 2.3.1: State evolution and its observed state.

It is now possible to compute the error between the *LS* $\Sigma(A,B,S)$ and its observer, denoted by $\bar{\Sigma}(A,B,S)$. Let $\tilde{x} = x - \bar{x}$ be the observer's error. Hence, the dynamics of the observer's error is

$$\dot{\tilde{x}}(\tau) = (A - LS)\tilde{x}(\tau). \tag{2.13}$$

It is known that the set of eigenvalues of the matrix $A - LS$ in Eq. (2.13) can be arbitrarily assigned by a suitable choice of L if and only if, the *LS* $\Sigma(A,B,S)$ is observable. Thus, by a suitable choice of L the observer's error $\tilde{x}$ converges asymptotically to zero at a desired rate [40], [39].

Example 2.3.5 *Consider the LS determined by matrices*

$$A = \begin{bmatrix} -1 & -2 & -3 \\ 4 & -5 & 6 \\ 7 & -8 & -9 \end{bmatrix} \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \quad S = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}.$$

The observer for this example is given by $\bar{\Sigma}(A,B,S)$*:*

$$\dot{\bar{x}}(\tau) = \begin{bmatrix} -1 & -2 & -3 \\ 4 & -5 & 6 \\ 7 & -8 & -9 \end{bmatrix} \bar{x}(\tau) + \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} u(\tau) + \begin{bmatrix} 12 \\ -4 \\ 23.33 \end{bmatrix} (y - \bar{y})$$
$$\bar{y}(\tau) = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \bar{x}(\tau).$$

The matrix $L = \begin{bmatrix} 12 & -4 & 23.33 \end{bmatrix}^T$ *in the previous equation leads to an observer's error dynamic with poles located in -9, -10 y -8, which is asymptotically stable. In Figure 2.3.1 the evolution of the states and its observed state (in dashed line) is shown.*

2.4 Switched linear systems.

A switched linear system is a collection of *LS* switching among them, formally defined next.

Definition 2.4.1 *A switched linear system (SLS) $\Sigma_{\phi(\tau)} = \langle \mathcal{F}, \phi \rangle$ is a hybrid system where $\mathcal{F} = \{\Sigma_1, \Sigma_2, ..., \Sigma_n\}$ is a collection of LS and $\phi : [0, \infty) \rightarrow \{1, 2, ..., m\}$ is a switching signal determining, at each time instant, the evolving LS $\Sigma_i \in \mathcal{F}$.*

In the general case of the *SLS*, the switching signal is external and it can be either known or unknown. For the particular case of *ContPN*s, the switching signal is dependent on the state of the *ContPN*, i.e. the *ContPN* marking.

2.4.1 Observability in *SLS*.

The observability problem in *SLS* is more complex than the one in *LS*. This is because, in order to determine the state of the *SLS*, it is necessary to determine which *LS* is actually evolving as well as the state vector x from the output measurements. Once the evolving *LS* has been uniquely identified, then the state vector x of the system should be computed. However, it is possible that two different *LS* generate the same output trajectory with an input. If this happens, the two *LS* are said to be indistinguishable from each other. Therefore it is necessary that each pair of *LS* are distinguishable. The distinguishability property deals with the possibility to determine which *LS* is evolving at any given time τ, only by the knowledge of the output of the *SLS* over a finite time period. The concept of distinguishability is presented next.

Definition 2.4.2 *[71] Let Σ_1 and Σ_2 be two LS of the same dimension and with the same number of inputs. Then, the LS Σ_1, Σ_2 are said to be distinguishable from each other if for every initial condition x_0 and input $u(\tau)$, the knowledge of the input $u(\tau)$ and the outputs y_1 and y_2 over the finite time interval $\tau \in [\tau, \tau_1]$ suffices to determine which LS is evolving, Σ_1 or Σ_2.*

Theorem 2.4.3 *([48], [49]) The autonomous LS $\Sigma_i(A_i, 0, S_i)$ and $\Sigma_j(A_j, 0, S_j)$ are distinguishable from each other if and only if the extended LS $\Sigma_{ij}(A_{ij}, 0, S_{ij})$ with matrices*

$$A_{ij} = \begin{bmatrix} A_i & 0 \\ 0 & A_j \end{bmatrix} \quad S_{ij} = \begin{bmatrix} S_i & -S_j \end{bmatrix} \tag{2.14}$$

is observable.

Clearly, the distinguishability between *LS* implies the observability of the individual *LS*, but the converse is not true in general.

The characterization of the indistinguishability subspace is given next.

Definition 2.4.4 *[49] Given any two LS Σ_i, Σ_j, the indistinguishability subspace of Σ_i, Σ_j, denoted by $\bar{\mathcal{W}}_{i,j}$, is*

$$\bar{\mathcal{W}}_{i,j} = \left\{ \begin{bmatrix} x_0^i \\ x_0^j \end{bmatrix} : \exists u(\tau)\ \forall \tau \geq 0 \textit{ such that } y_i(x_0^i, u(t)) = y_j(x_0^j, u(t)) \right\}$$

The previous definition shows that the indistinguishability subspace contains the state trajectories which are equal under the same input, and therefore it is not possible to determine to which *LS* it belongs (it could be to any of them).

For the autonomous case of *SLS*, the observability problem has been completely characterized in [48] and [49]. Even though *SLS* are non linear systems, [48] and [49] provide a simple and linear characterization for the distinguishability property.

2.4.2 *SLS* and *ContPN*s

A *ContPN* can be modelled by a switched linear system *SLS* with polyhedral regions, which are determined by the configurations or regions of the *ContPN*([16], [51], [52]).

Consider the family of *LS*

$$\mathcal{F} = \{\overset{\bullet}{\mathbf{m}} = A_k \mathbf{m}; \quad y = S\mathbf{m}\} \tag{2.15}$$

where $A_k = \mathbf{C}\Lambda\Pi_k$ and Π_k is the configuration matrix introduced in Eq. (2.5). The switching signal for the case of *ContPN* is state dependent, where the configuration matrix Π_k is selected depending on the *ContPN* marking. Notice that in *ContPN*s $B_i = 0, S_i = S, \forall i = 1, ..., n$, i.e. the *LS* are autonomous and the output matrix is the same for each *LS* $\Sigma_k \in \mathcal{F}$.

As a notation, the observability matrix and the unobservable subspace for the configuration $\mathcal{R}_k$ represented by the *LS* $\Sigma(A_k, 0, S)$ are denoted as $\mathcal{O}_k$ and $\mathcal{N}_k$ respectively.

It is important to recall that the *ContPN*s are a special class of autonomous *SLS*. First, because the marking is always non-negative. Second, the switching among *LS* is state dependent. Finally, the number of *LS* in the *SLS* representation of the *ContPN* grows exponentially with the number of join transitions. Take for instance the *ContPN* in Fig. 2.4.1-A. This *ContPN* has 13 places and 4 transitions and there exists 81 *LS* in the family $\mathcal{F}$. However, in the *ContPN*

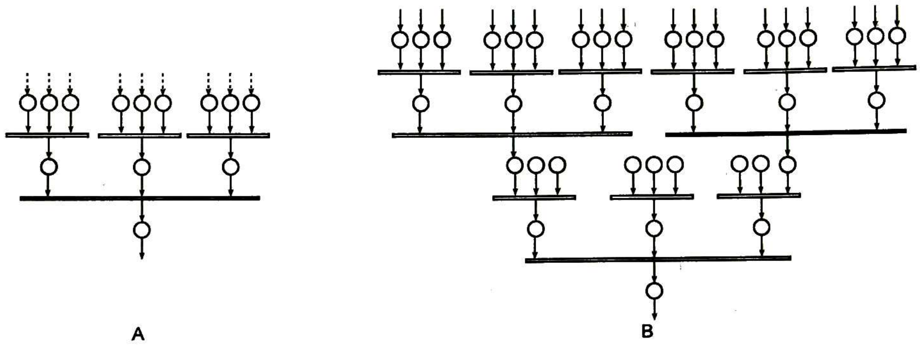


Figure 2.4.1: Explosion on the number of *LS*.

presented in Figure 2.4.1-B, there are 37 places and 12 transitions, but 531,441 *LS* in the family $\mathcal{F}$.

It seems that since *ContPN*s are a *SLS*, then the previously explained approach to determine the observability in *SLS* suffices to determine if a *ContPN* is observable. However, as previously discussed, the number of *LS* in the family $\mathcal{F}$ grows exponentially on the number of join transitions in the *ContPN*. Therefore, the naive approach of verifying the observability of each $\Sigma_k \in \mathcal{F}$ and the distinguishability between each pair of *LS* becomes prohibited in practice. For instance, to determine if the *ContPN* in Figure 2.4.1-A is observable, it is necessary to verify that each of the 81 *LS* are observable and distinguishable from each other. To achieve that, it is necessary to verify the rank of 81 observability matrices to determine if each one is observable and 3,240 extended observability matrices to determine if each pair of *LS* is distinguishable from each other. In the *ContPN* in Fig. 2.4.1-B, the number of observability matrices to verify are 531,441 for observability in each *LS* + 141,214,502,520 for the distinguishability between each pair of *LS*.

3

Observability and observers design. State of the art.

This chapter provides the most relevant results on observability of *ContPN*. Since a *ContPN* can be modelled by a *SLS*, the observability in *SLS* is an important approach for the solution of the observability problem in *ContPN*. Unfortunately, due to the exponential number of *LS* required to model a *ContPN* with join transitions, this approach results infeasible in practice. Therefore, a different approach is required. The existing results on observability for *ContPN* will be provided in this chapter. These results are based on analysing the underlying graph of the *ContPN* (i.e. the structure of the *ContPN*), which is very convenient, since it provides algorithms in polynomial time to verify if a *ContPN* is observable or not. Unfortunately the existing results are valid only for the JF, *AF* and JAF classes of *ContPN*. In the general case, the current results also require the explicit enumeration of every *LS* of the *SLS* representation of the *ContPN*.

3.1 Results on observability of *ContPN*

In this section, the existent results on observability of *ContPN* will be described. The work herein presented was mainly developed by the GISED research group, from the Universidad de Zaragoza.

The work presented in [45] is based on the previously known concept of observability in *SLS*: A *SLS* is observable iff

1. $\forall \Sigma_i \in \mathfrak{F}$, Σ_i is observable.

2. $\forall \Sigma_i, \Sigma_j, i \neq j$, the pair Σ_i, Σ_j are distinguishable from each other.

One of their main contributions is that in *ContPN* there exist redundant *LS*, therefore the number of *LS* to analyze can be reduced. In this work, the authors state that a place $p_h \in P$ is implicit for any reachable marking from an initial marking $\mathbf{m}_0$, i.e.

$$\frac{\mathbf{m}(p_j)}{Pre(p_j, t_i)} \leq \frac{\mathbf{m}(p_h)}{Pre(p_h, t_i)}$$

with $p_j \in \bullet t_i \setminus \{p_h\}$ is satisfied $\forall t_i \in p_h \bullet$. This implies that the regions where

$$\frac{\mathbf{m}(p_h)}{Pre(p_h, t_i)} \leq \frac{\mathbf{m}(p_j)}{Pre(p_j, t_i)}$$

are either empty or reduced to their borders. Therefore such region should not be considered. Unfortunately this approach is based on the initial marking $\mathbf{m}_0$, which is actually unknown.

In order to avoid the dependence of $\mathbf{m}_0$, they remove the ***structurally redundant modes***, i.e. the modes which for every initial marking $\mathbf{m}_0$ are redundant. This is solved by solving a linear programming problem (*LPP*) where the objective function is $\max \epsilon$ and there is a restriction of the form:

$$\frac{\mathbf{m}(p_j)}{Pre(p_j, t_i)} + \epsilon \leq \frac{\mathbf{m}(p_h)}{Pre(p_h, t_i)}$$

for each inequality determining the region. The main drawback of this approach is that it requires the enumeration of the inequalities which determine the regions of the *ContPN*, i.e. an exponential number of restrictions.

Once the redundant modes are removed, they address the distinguishability problem with the following linear programming problem (*LPP*).

Proposition 3.1.1 *Let $k(k = 1, 2)$ be a mode, $\mathcal{O}_k$ and $\mathcal{R}_k$ the corresponding observability matrix and region. If the LPP*

$$\begin{aligned}
&\max \epsilon \\
&s.t. \\
&-\epsilon \cdot \mathbf{1} \leq \mathbf{m}_1 - \mathbf{m}_2 \leq \epsilon \cdot \mathbf{1} \\
&\mathcal{O}_1 \cdot \mathbf{m}_1 - \mathcal{O}_2 \cdot \mathbf{m}_2 = 0 \\
&\mathbf{m}_1 \in \mathcal{R}_1 \\
&\mathbf{m}_2 \in \mathcal{R}_2 \\
&\mathbf{m}_1, \mathbf{m}_2 \geq 0
\end{aligned}$$

has solution $\epsilon = 0$ then the modes 1 and 2 are distinguishable.

The previous *LPP* looks for a marking $\mathbf{m}_1$ in $\mathcal{R}_1$ which can be *confused* with a marking $\mathbf{m}_2$ in $\mathcal{R}_2$. If such pair of markings exist ($\epsilon > 0$), then the pair of modes will be undistinguishable. Otherwise, when $\epsilon = 0$, it means that there does not exist any similar marking in both regions, thus the pair of regions are distinguishable from each other. Unfortunately, this approach also needs the computation of a *LPP* for each pair of regions in the *ContPN*.

For the observability of the *ContPN*, the problem is addressed in the graph level of the *ContPN*. This is a great advantage since it does not require the enumeration of neither the constrains which determine each region of the *ContPN* nor the actual *LS* structure ($\Sigma_i(A_i, 0, S)$).

The most important contribution of [45] is based on the next definition:

Definition 3.1.2 *A place p_j is output connected if there exists a path, denoted γ_j from place p_j to a measured place $p_h \in P_M$.*

The output connectedness characterizes the existence of a path from a place p_j to another place p_h which is measured. This path does not have any restrictions, but the authors show that in some classes of *ContPN*s, the output connectedness leads to the computation of the marking $\mathbf{m}(p_j)$ from the knowledge of $\mathbf{m}(p_h)$ in infinitesimal time.

Proposition 3.1.3 *[45] Let N be a JAF ContPN. A place p_j is structurally observable iff p_j is output connected.*

The previous propositions states that in *JAF ContPN*s, it is necessary and sufficient that from each non-measured place p_j there exists a path to a measured place.

Also, Proposition 3.1.3 can be interpreted as the following idea: if there exists a $JAF-$path from a place p_j to a measured place p_h, then the marking of place p_j can be computed in infinitesimal time, using the back propagation algorithm presented in [44].

In *JAF-ContPN*s they also propose to analyze the terminal components. A set of places F is said to define a strongly connected component of N, named N', which is the subnet N' generated by the set of places F and the set of transitions $T' = \bullet F \cup F \bullet$.

Definition 3.1.4 *[45] A strongly connected component $N' = \langle F, T', Pre', Post' \rangle$ of a ContPN is said to be terminal if there is no path from a place belonging to F to a place not in F.*

The general solution for the observability in *JAF ContPN*s is given by the following proposition:

Proposition 3.1.5 *[45] Let N be a JAF ContPN. N is structurally observable iff at least one place from each terminal strongly connected component is measured.*

For the attribution free (AF) case, they propose the following:

Proposition 3.1.6 *[45] Let N be a ContPN. Then, construct N' from N by removing every join transition and its input and output arcs. Then, N is structurally observable iff N' is structurally observable.*

With previous proposition, a *AF-ContPN* is *broken* into many pieces, each of them a *JAF-ContPN*. With this strategy, the previous results apply and each strongly connected component should have at least one measured place. Unfortunately, the previous results do not apply if the *ContPN* is not AF.

For the non AF case, the observability problem addressed is the *generic* observability, formally defined next.

Definition 3.1.7 *Let $\langle N, \lambda, \mathbf{m}_0 \rangle$ be a JF ContPN and P_M the set of measured places. $\langle N, \lambda, \mathbf{m}_0 \rangle$ is weakly structural or generically observable from P_M if $\langle N, \lambda, \mathbf{m}_0 \rangle$ is observable for all values of λ outside a proper algebraic variety of the parameter space.*

In [45], it is also presented an algorithm to determine if a *ContPN* is generically observable when there exist attribution places but only for the join free (JF) case. The idea is based on [53], and it consists on building a directed digraph $\mathcal{G}(N)$ for the *ContPN*. The directed digraph $\mathcal{G}(N)$ is constructed in the following way:

1. The vertex set Z is given by the set P of places ($Z = P$).

2. The edge set is computed as:

$$W = \{(p_i, p_j)|p_j \in (p_i\bullet)\bullet \wedge p_u \neq p_j\} \cup \{(p_i, p_i)|\exists t \in p_i\bullet, Pre(p_i, t) \neq Post(p_i, t)\}$$

In the edges computation, the first set adds an edge from a place p_i to all places $(p_i\bullet)\bullet$ since the dynamic matrix has a non-null entry and prevents adding an edge in the case of a self-loop. The second subset will add a self-loop in the associated graph for any place with $Pre(p_i, t) \neq Post(p_i, t)$ i.e., the marking of pi will change firing t, implying that the dynamical matrix has a non-zero entry.

Definition 3.1.8 *Let N be a ContPN and $\mathcal{G}(N)$ its associated digraph with vertex set Z and edge set W. Consider a set W_S made of k_s vertices. Denote by $E(W_S)$ the set of vertices w_i, for $i = 1, 2, ..., l_s$ of Z such that there exists an edge $(x_j, w_i) \in W$ with $x_j \in W_S$. W_S is said to be a contraction if $k_s - l_s > 0$.*

Using the previous definition, the next proposition determines if a *JF-ContPN* is generically observable.

Proposition 3.1.9 *Let N be a JF-ContPN and $\mathcal{G}(N)$ be its associated graph. N is generically observable iff:*

- *N is output connected*
- *$\mathcal{G}(N)$ contains no contraction*

Unfortunately, for this procedure it is needed to compute many combinations of sets W_S and its possible contractions $E(W_S)$ in order to verify if a contraction exists.

3.2 Observer design

The work presented in [44] introduces an algorithm to compute the marking of a $JF-$*ContPN* from a given set of measured places P_M. It is important to notice that a $JF-$*ContPN* has only one *LS* $\Sigma(A, 0, S)$ which determines its behaviour. Then, they use such single structure captured in the dynamical matrix A to compute the marking of every place in the *ContPN*. This is executed as explained in the following example.

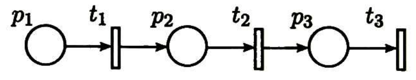


Figure 3.2.1: A simple *ContPN* structure.

Example 3.2.1 *The ContPN in Figure 3.2.1 (taken from [44]) has a very simple structure. Let the rate of transitions be* $\lambda = \begin{bmatrix}\lambda_1 & \lambda_2 & \lambda_3\end{bmatrix}^T$. *The set of measured places is* $P_M = \{p_3\}$. *This ContPN has the following incidence matrix and dynamical matrix:*

$$\mathbf{C} = \begin{bmatrix} -1 & 0 & 0 \\ 1 & -1 & 0 \\ 0 & 1 & -1 \end{bmatrix} \qquad A = \begin{bmatrix} -\lambda_1 & 0 & 0 \\ \lambda_1 & -\lambda_2 & 0 \\ 0 & \lambda_2 & -\lambda_3 \end{bmatrix} \tag{3.1}$$

Then the dynamics of the system is modelled by:

$$\begin{aligned} \dot{\mathbf{m}}_1 &= -\lambda_1 \mathbf{m}(p_1) \\ \dot{\mathbf{m}}_2 &= \lambda_1 \mathbf{m}(p_1) - \lambda_2 \mathbf{m}(p_2) \\ \dot{\mathbf{m}}_3 &= \lambda_2 \mathbf{m}(p_2) - \lambda_3 \mathbf{m}(p_3) \end{aligned}$$

Since the marking of place p_3 *is known, then also its derivative can be computed with the evolution of the ContPN. Then,* $\mathbf{m}(p_2) = (\dot{\mathbf{m}}_3 + \lambda_3 \mathbf{m}(p_3))/\lambda_2$. *Once* $\mathbf{m}(p_2)$ *is computed, its derivative can also be computed and* $\mathbf{m}(p_1) = (\dot{\mathbf{m}}_2 + \lambda_2 \mathbf{m}(p_2))/\lambda_1$.

The algorithm explained in the previous example will be further referred as the backward computation algorithm (*BCA*).

This paper also provides a definition of observable places and structurally observable places.

Definition 3.2.2 *A place* $p_j \in P$ *is observable from* P_M *iff it is possible to compute its initial marking* $\mathbf{m}_0(p_j)$ *by measuring the marking evolution of the places in the set* P_M. *A place* p_j *is structurally observable from* P_M *iff it is observable from* P_M *for any* $\lambda > 0$.

They also provide an algorithm that, given a measured set of places P_M, returns a set of structurally observable places Ω. With the assumption that every input place to join transitions, the authors claim that the same algorithm can be executed for non $JF-$*ContPN*.

For the state estimation, they propose the following.

1. Compute an estimate for every structural $PT-set$ of the *ContPN*.
2. Rule out the $PT-sets$ of the *ContPN* which are not ruling the behaviour of the *ContPN*:

(a) An estimate marking $\bar{\mathbf{m}}$ for a $PT - set\ W$ is *infeasible* if

$$\begin{bmatrix} y(0) \\ \overset{\bullet}{y}(0) \\ \vdots \\ y^{|P|-1}(0) \end{bmatrix} = \mathcal{O}_k \mathbf{m}_0$$

where $\mathcal{O}_k$ is the observability matrix of the configuration k.

(b) An estimate marking $\bar{\mathbf{m}}$ for a $PT - set\ W$ is *incoherent* if $W \not\subseteq PT - Set(\bar{\mathbf{m}})$, i.e. W is not a $PT - set$ of $\bar{\mathbf{m}}$.

3. The remaining estimates represent every potential marking of the *ContPN*.

However, there are two relevant drawbacks in the previous approach. The first one is the need of the computation of every $PT - set$ or configuration of the *ContPN*, which grows exponentially with the join transitions, as previously discussed. The second drawback is that the computation of the estimate marking is very sensitive to noise in the output derivatives.

In order to improve the estimate marking, they propose to use a Luenberger observer for each configuration, and then, rule out the $PT - sets$ as previously discussed. Unfortunately, there is still needed the computation of every dynamical matrix of the *SLS* representation of the *ContPN* (an exponential number of them) in order to be able to design its observer, which becomes prohibited in practice.

3.3 Optimal observability in *ContPN*

The work presented in [43] deals with the sensor placement problem. It is considered that all places are measurable, i.e. it is possible to add a sensor to every place, therefore its marking is always known. Assuming that there exists an associated cost to add a sensor to every place, i.e. $\forall p_j \in P$ the cost of adding a sensor is $\varsigma(p_j)$. This concept is extended to a set of measured places $\mathcal{D}_i$, where the cost of measuring the places in $\mathcal{D}_i$ is $\varsigma(\mathcal{D}_i) = \sum_{p_j \in \mathcal{D}_i} \varsigma(p_j)$. Based on these definitions, the problem is to find a set $\mathcal{D}$ of minimum cost from which a *ContPN* $\langle N, \mathbf{m}_0, \lambda \rangle$ is observable.

In this work, they define the set of *observable* places as the following:

Definition 3.3.1 *A place $p_j \in P$ is observable from $\mathcal{D}_i$ if the marking $\mathbf{m}(p_j)$ can be computed from the knowledge of the marking of the places in $\mathcal{D}_i$.*

The set of all the places which are observable from the measured places $\mathcal{D}_i$ is named the observable places from $\mathcal{D}_i$ and it is denoted by $\mathbf{O}_i$.

It is assumed that it is possible to compute $\mathbf{O}_i$ from $\mathcal{D}_i$ using the tools provided in [44].

Unfortunately it has been shown in [46], [44] that if $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \ldots \cup \mathcal{D}_i$, the observable places from $\mathcal{D}$, denoted by $\mathbf{O}$ fulfils the following

$$|\mathbf{O}| \geq |\mathbf{O}_1 \cup \mathbf{O}_2 \cup \ldots \cup \mathbf{O}_i|$$

i.e. the places that can be computed from a union of disjoin sets of measured places $\mathcal{D}_i$ can be greater than the union of the observable places $\mathbf{O}_i$ from each set of measured places $\mathcal{D}_i$. This can be seen in the following example.

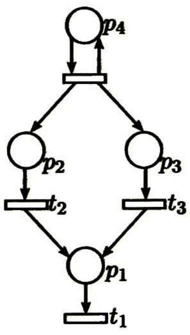


Figure 3.3.1: A *ContPN* structure with an attribution.

Example 3.3.2 *Consider the Figure 3.3.1 with $\lambda_2 = \lambda_3$. Let $\mathcal{D}_1 = \{p_1\}$. The set of observable places from $\mathcal{D}_1$ is $\mathbf{O}_1 = \{p_1\}$. Also, let $\mathcal{D}_2 = \{p_2\}$. The set of observable places from $\mathcal{D}_2$ is $\mathbf{O}_2 = \{p_2, p_4\}$. However, if $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 = \{p_1, p_2\}$, then the net becomes observable, i.e. $\mathbf{O} = \{p_1, p_2, p_3, p_4\}$.*

The previous example shows that the optimal observability problem can be seen as a Set Covering Problem, which is NP-hard in the strong sense [43], then the authors show that it is possible to determine the optimal sensor placement for some subclasses or, in the worse case, to reduce considerably the number of covers to consider with an algorithm that they introduce. This algorithm can be further studied in [43].

4

On invariant subspaces and the *ContPN* structure

This chapter deals with the A_k−invariant subspaces characterization of the *ContPN*. As previously discussed, *ContPN* are a special class of autonomous *SLS*. Also, in Subsection 2.4.2 it was shown that the number of configurations in a *ContPN*, and equivalently the number of *LS* in its *SLS* representation, increases exponentially with the number of join transitions. Even when obtaining an A_k-invariant subspace of a particular dynamical matrix A_k is possible, and actually simple, the problem of obtaining them for all of the matrices A_k would also lead to NP-complete algorithms. Fortunately, the complexity problem can be avoided by characterizing these invariant subspaces directly from the structure of the *ContPN*.

It will be shown in this chapter that the *ContPN* structure allows the computation of the A_k−invariant subspaces associated to $\ker(A_k)$, but also some other A_k−invariant subspaces which do not belong to $\ker(A_k)$. Even though not every A_k−invariant subspace is characterized, it will also be shown that every A_k−invariant subspace which does not belong to $\ker(A_k)$ is contained in the image of the incidence matrix $\mathbf{C}$ ($Im(\mathbf{C})$).

The results obtained in this chapter will later be used to propose a strategy to construct the output mapping such that the *ContPN* becomes observable.

4.1 Invariant subspaces

In this section, some of the invariant subspaces of the *ContPN* will be characterized using the *PN* structure.

As discussed in Subsection 2.4.2, the *ContPN* systems have several regions due to join transitions, and each region has a different dynamical matrix $A_k = \mathbf{C}\Lambda\Pi_k$ (Π_k is different in each region $\mathcal{R}_k$). The naive approach of verifying the observability property by computing each observability matrix $\mathcal{O}_k$ leads to NP-complete algorithms since the number of regions in a *ContPN* grows exponentially with the join transitions.

As previously defined, a subspace $\mathcal{V}$ is named A_k-invariant iff $A_k\mathcal{V} \subseteq \mathcal{V}$.

A particular case of subspaces are those of dimension one, generated by the eigenvectors. An eigenvector is a non-zero vector v satisfying

$$A_k v = \beta v \tag{4.1}$$

i.e. the eigenvector generates an A_k-invariant subspace of dimension one. The vector v is said to be an eigenvector of A_k associated to the eigenvalue β.

Particularly, it is known that a subspace $\mathcal{V} \subseteq \ker A_k$ iff $\forall v \in \mathcal{V},\ \ A_k v = 0$. In this case, all vectors $v \in \mathcal{V}$ are eigenvectors associated to the eigenvalue $\beta = 0$.

Consider now a subspace $\mathcal{V}$ such that $\mathcal{V} \not\subseteq \ker(A_k)$. Then, since $\mathcal{V} \subseteq Im(A_k)$ and $A_k = \mathbf{C}\Lambda\Pi_k$ then $\mathcal{V} \subseteq Im(\mathbf{C}\Lambda\Pi_k)$. Then, it is possible to see that $\mathcal{V} \subseteq Im(\mathbf{C})$. It is an important fact, since the *ContPN* has only one incidence matrix $\mathbf{C}$. However, if $\mathcal{V} \subseteq \ker(A_k)$ then there are two options:

1. $\mathcal{V} \subseteq \ker(\Pi_k)$ or
2. $\mathcal{V} \subseteq \ker(\mathbf{C}) \cap Im(\Lambda\Pi_k)$

i.e. $\mathcal{V}$ may not be contained in $Im(\mathbf{C})$. Therefore $\ker(A_k)$ should be characterized differently, as shown in the next section.

4.1.1 The kernel A_k- invariant subspaces

The problem of finding $\ker(A_k)$ is equivalent to find the eigenvectors for a matrix A_k associated to an eigenvalue $\beta = 0$. In this way, Equation (4.1) must hold and it becomes $\mathbf{C}\Lambda\Pi_k v = 0$. Since Λ has full rank, the only two possibilities are $\Lambda\Pi_k v \in \ker(\mathbf{C})$ or $v \in \ker(\Pi_k)$. Because of the form of matrix Π_k, it can be seen that this matrix only looses rank when a place is not constraining any transition's flow. The following proposition characterizes vectors $v \in \ker(\Pi)$.

Proposition 4.1.1 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a ContPN. A vector $v \in \ker(\Pi_k)$ iff $v = \sum_j \alpha_j \mathbf{e}_j$, where $\alpha_j \in \mathbb{R}$ and the associated places $p_j \sim \mathbf{e}_j$ do not constrain any transition in the region $\mathcal{R}_k$.*

Proof:

$\leftarrow$ When a place p_j does not constrain any transition, the j-th column of Π_k is a zero column. Clearly a vector $e_j \in \ker(\Pi_k)$, since $\Pi_k e_j = 0$. Even more, any linear combination of such vectors also belongs to $\ker \Pi_k$.

$\rightarrow$ Each row of Π_k is $\pi_{ki} = e_h^T$, meaning that a place p_h is constraining the flow of transition t_i. Clearly if a vector $v \in \ker(\Pi_k)$, $\pi_{ki} v = 0\ \forall i,\ i = 1, 2, ..., |T|$. Therefore every π_{ki} is orthogonal to e_j^T, i.e. $\pi_{ki} \neq e_j^T\ \forall i$ which proves that place p_h does not constrain any transition. ■

One of the cases when a place p_j does not constrain any transition is when $p_j \in P_E$, where P_E is the set of ending places, i.e. $p_j\bullet = \emptyset$. The following proposition characterizes those ending places.

Proposition 4.1.2 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a ContPN. If $p_j \in P_M$, i.e. p_j is an ending place, then $\forall k\ \mathbf{e}_j \in \ker(A_k)$.*

Proof: Let p_j be an ending place. Then any configuration matrix Π_k has its $j-th$ column equal to zero, because p_j is not constraining the flow of any transition. Therefore $\Pi_k \alpha_j e_j = 0 \rightarrow \alpha_j e_j \in \ker \Pi_k \rightarrow \alpha_j e_j \in \ker \mathbf{C}\Lambda\Pi_k$. ■

It is important to notice that Proposition 4.1.2 applies for every class of *ContPN*.

The other possibility for a place p_j not to constrain any transition is when $p_j\bullet \neq \emptyset$ but some other place, say p_h constrains the flow of those transitions in $p_j\bullet$. When this occurs, clearly the transitions in $p_j\bullet$ are join transitions. The following proposition characterizes the null space corresponding to join transitions.

Proposition 4.1.3 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a ContPN with a set $T_J \neq \emptyset$ of join transitions. If $p_j\bullet \subseteq T_J$ then $\exists \Pi_k$ such that $\mathbf{e}_j \in \ker(\mathbf{C}\Lambda\Pi_k)$.*

Proof: Let t_i be a join transition with $p_j, p_i \in \bullet t_i$. Now, $\forall t_i$ let p_i be the place constraining the flow of transition t_i. Then p_j does not constrain any transitions' flow in such configuration and therefore $\exists \Pi_k$ such that $\mathbf{e}_j \in \ker \Pi_k$. ■

The previous result can be particularized for the class of free choice *ContPN* (*FC-ContPN*).

Corollary 4.1.4 *Let t_i be a join transition of a FC-ContPN. Then for each vector $e_j \sim p_j$, where $p_j \in \bullet t_i$, there exist some dynamical matrices $\mathrm{C}\Lambda\Pi_k$ such that e_j is an eigenvector associated to the eigenvalue $\beta = 0$.*

Proof: A PN is FC when:

a) For any two places $s, r \in P$, $s\bullet \cap r\bullet = \emptyset$. In this case, there are not Join transitions.

b) For any two places $s, r \in P$, $s\bullet = r\bullet$. Let $\bullet t_i = \{p_1, p_2, ..., p_n\}$, $i = 1, 2, ..., m$. Let p_1 constrain the flow of t_i. It is clear that places $p_2, ..., p_n$ are not constraining the flow of any other transition and every matrix Π_k for which p_1 is constraining the flow of t_i has $n-1$ zero columns associated to places $p_2, ..., p_n$. It is also clear that those zero columns are also zero columns of $\mathrm{C}\Lambda\Pi_k$ and every vector $e_2, ..., e_n$ are eigenvectors for some $\mathrm{C}\Lambda\Pi_k$. Following this reasoning, $\forall p_j \in \bullet t_i$, e_j is an eigenvector associated to some $\mathrm{C}\Lambda\Pi_k$. ■

Corollary 4.1.4 is important, since it states that for every input place p_j to a join transition t_i, the vector $\mathrm{e}_j \sim p_j$ is contained in $\ker(A_k)$ for some k.

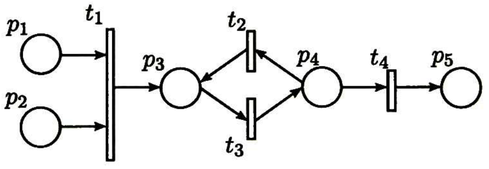


Figure 4.1.1: A *ContPN* system with join transitions and ending places.

Example 4.1.5 *The ContPN in figure 4.1.1 has two regions. One of them is when place p_1 constrains the flow of transition t_1 (say, represented by configuration matrix Π_1); the second one is when p_2 does, represented by configuration matrix Π_2.*

$$\Pi_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix} \qquad \Pi_2 = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}$$

Assume that the transition's rates are $\lambda = \begin{bmatrix} 1 & 2 & 3 & 4 \end{bmatrix}^T$ With these configurations and transition's rates, the dynamical matrices are:

$$A_1 = \mathbf{C}\Lambda\Pi_1 = \begin{bmatrix} -1 & 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 1 & 0 & -3 & 2 & 0 \\ 0 & 0 & 3 & -6 & 0 \\ 0 & 0 & 0 & 4 & 0 \end{bmatrix} \qquad A_2 = \mathbf{C}\Lambda\Pi_2 = \begin{bmatrix} 0 & -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 \\ 0 & 1 & -3 & 2 & 0 \\ 0 & 0 & 3 & -6 & 0 \\ 0 & 0 & 0 & 4 & 0 \end{bmatrix}$$

In Figure 4.1.1, the PN structure shows an ending place (p_5). Therefore, the vector $e_5 \in \ker \Pi_k$, $k = 1, 2$ (e_5 associated to place p_5); also, for each of the dynamical matrices of the ContPN A_k, $e_5 \in \ker A_k$, $k = 1, 2$.

Similarly, vectors e_1 and e_2, associated to places p_1 and p_2 respectively, are input places to join transitions as in Lemma 4.1.4. Therefore $e_1 \in \ker \Pi_2$ and $e_2 \in \ker \Pi_1$. In the same way $e_1 \in \ker A_2$ and $e_2 \in \ker A_1$.

Propositions 4.1.2 and 4.1.3 characterize all vectors in $\ker(\Pi)$. Now, as previously discussed, Equation (4.1) also holds when $\Lambda\Pi_k v \in \ker(\mathbf{C})$. The solution are those vectors

$$v \in Im(\Lambda\Pi_k) \cap \ker(\mathbf{C}). \tag{4.2}$$

Clearly those vectors v are T-Semiflows, but not every T-Semiflow belongs to $Im(\Lambda\Pi_k) \cap \ker(\mathbf{C})$. A T-Semiflow belonging to $Im(\Lambda)$ will be called a T-Semiflow agreeing with λ, formally defined next.

Definition 4.1.6 *A T-Semiflow agreeing with λ is a vector $x \in \ker(\mathbf{C})$ such that for every decision place p_j with $p_j\bullet = \{t_1, t_2, ..., t_n\}$, either every transition $t_1, ..., t_n$ belongs to a different minimum T-Semiflow or for those transitions in the same minimum T-Semiflow, the ratio $x_i/x_h = \lambda_i/\lambda_h$ holds $\forall t_i, t_h \in p_j\bullet$.*

The following proposition characterizes the conditions for which a T-Semiflow becomes an annuler for $\mathbf{C}\Lambda\Pi_k$ in a join free (*JF*) *ContPN*, i.e. a T-Semiflow that also belongs to $Im(\Lambda\Pi_k)$.

Proposition 4.1.7 *Let $\langle N, \lambda, \mathbf{m}_0 \rangle$ be a JF ContPN and let X be a T-Semiflow agreeing with λ. Then $\exists v \notin \ker(\Pi)$ such that $\mathbf{C}\Lambda\Pi_k v = 0$ iff $(\bullet\langle X \rangle)\bullet = \langle X \rangle$. v will be called* ***Annuler vector****. The structure of v is given by*

$$v = \begin{bmatrix} \gamma_1 & \gamma_2 & \dots & \gamma_{|P|} \end{bmatrix}^T$$

where $\gamma_h = 0$ if $p_i \notin \bullet X$ or $\gamma_h \neq 0$ if $p_i \in \bullet X$.

Proof:

Necessity. *Let v be a vector such that $\mathbf{C}\Lambda\Pi_k v = 0$. Clearly, $\Lambda\Pi_k v = x$ is a T-Semiflow. However, $\Lambda\Pi_k$ is a function that maps from places in $\langle v\rangle$ that constrain the flow of some transitions, to all transitions in $\langle v\rangle\bullet$. Let $t_i \in \langle v\rangle\bullet$ such that $t_i \notin \langle x\rangle$. The $i-th$ component of $\Lambda\Pi_k v$ is nonzero and the $i-th$ component of x is zero. This is a contradiction and for v to exist, $(\bullet\langle x\rangle)\bullet = \langle x\rangle$ must hold.*

Sufficiency. *Let $(\bullet\langle x\rangle)\bullet = \langle x\rangle$. Then every non-zero (zero) component in $\Lambda\Pi_k v$ is also non-zero (zero) in x. The $i-th$ component of $\Lambda\Pi_k v$ can be written as $\lambda_i v_j$, where v_j corresponds to the place p_j that is constraining transition t_i. If each transition is constrained by a different place, equation $\lambda_i v_j = x_i$ can always be solved with $v_j = x_i/\lambda_i$. When a place p_j constrains the flow of more than one transition (i.e. p_j is a decision place), let $p_j\bullet = \{t_1, t_2, ..., t_n\}$. There are n equations for place p_j, $\lambda_1 v_j = x_1$,..., $\lambda_n v_j = x_n$. Since x agrees with λ, there are two cases:*

a) When each transition in $p_j\bullet$ belongs to a different T-Semiflow. Let $v_j = x_{a1}/\lambda_1$ and compute $\alpha_a x_{ai} = \lambda_i v_j$ for $i = 2, ..., n$ (clearly $\alpha_a x_{ai}$ is also a T-Semiflow for $\alpha_a \in \mathbb{R}$).

b) When some of the transitions in $p_j\bullet$ belong to the same minimum T-Semiflow. Let $t_2, t_3 \in \langle x_a\rangle$, x_a be a minimum T-Semiflow. Since v_j can be fixed with some other transition let $\alpha_a = \lambda_2 v_j/x_{a2}$. Clearly, equation $\alpha_a x_{a3} = \lambda_3 v_j$ holds. ■

Example 4.1.8 *In Figure 4.1.1, transitions t_2 and t_3 are the basis of the only T-Semiflow, i.e. $\mathbf{x}_1 = \begin{bmatrix}0 & 1 & 1 & 0\end{bmatrix}^T \in \ker\mathbf{C}$. However, $\langle\mathbf{x}\rangle = \{t_1, t_2\}$ and $(\bullet\langle x\rangle)\bullet = \{p_3, p_4\}\bullet = \{t_3, t_2, t_4\}$. Therefore $\nexists v \ker\mathbf{C}\Lambda\Pi_k$.*

However, in the ContPN in Fig. 4.1.2 with $\lambda = \begin{bmatrix}1 & 2 & 3 & 4\end{bmatrix}^T$ the T-Semiflow $\mathbf{x}_1 = \begin{bmatrix}0 & 1 & 1\end{bmatrix}^T$ fulfils the conditions in Proposition 4.1.7. Therefore there is a vector:

$$v = \begin{bmatrix}0 & 0 & \gamma_1 & \gamma_2\end{bmatrix}^T$$

such that $v \in \ker\mathbf{C}\Lambda\Pi_k$. The dynamical matrices are:

$$A_1 = \mathbf{C}\Lambda\Pi_1 = \begin{bmatrix}-1 & 0 & 0 & 0\\ -1 & 0 & 0 & 0\\ 1 & 0 & -3 & 2\\ 0 & 0 & 3 & -6\end{bmatrix} \qquad A_2 = \mathbf{C}\Lambda\Pi_2 = \begin{bmatrix}0 & -1 & 0 & 0\\ 0 & -1 & 0 & 0\\ 0 & 1 & -3 & 2\\ 0 & 0 & 3 & -6\end{bmatrix}$$

In order to obtain v, solving $A_1 v$ and $A_2 v$, two simultaneous equations are obtained: $2\gamma_2 - 3\gamma_1 = 0$ and $3\gamma_1 - 2\gamma_2 = 0$. Clearly any value such that $\gamma_2 = \frac{3}{2}\gamma_1$ fulfils the conditions. However, it is important to notice that the actual values of γ_i are not required, only the structure where the vector v is non-zero.

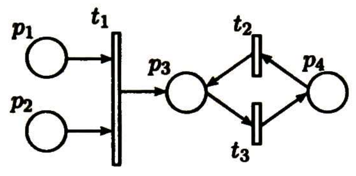


Figure 4.1.2: A *ContPN* system with a T-Semiflow and an annuler.

4.1.2 A_k–invariant subspaces contained in $Im(\mathbf{C})$

The following definitions will be used to characterize the A_k–invariant subspaces from the structure of the *ContPN*.

Definition 4.1.9 *Let $T_x \subseteq T$ be a set of transitions. The set of places constraining T_x in a given configuration $\mathcal{C}_k$ is*

$$\mathcal{P}(T_x|\mathcal{C}_k) = \{p_j \in P|(p_j, t_i) \in \mathcal{C}_k,\ t_i \in T_x\}.$$

Definition 4.1.10 *Let $P_x \subseteq P$ be a set of places. The set of transitions constrained by P_x in a given configuration $\mathcal{C}_k$ is*

$$\mathcal{T}(P_x|\mathcal{C}_k) = \{t_i \in T|(p_j, t_i) \in \mathcal{C}_k,\ p_j \in P_x\}.$$

Definition 4.1.11 *Let $T_x \subseteq T$ be a set of transitions and $P_x = \mathcal{P}(T_x|\mathcal{C}_k)$ be the set of places that constrain T_x in the configuration $\mathcal{C}_k$. If*

$$\mathcal{T}(P_x|\mathcal{C}_k) = T_x,$$

*i.e. the places that constrain the transitions in T_x only constrain transitions in T_x, then T_x is a **self-contained** set of transitions.*

Definition 4.1.12 *Let T_x be a self-contained set of transitions. If $T_x\bullet \subseteq \mathcal{P}(T_x|\mathcal{C}_k)$ then T_x is a **fully self-contained** set of transitions.*

As a notation, let a vector $x \in \mathbb{R}^{|T|}$. If $\langle x \rangle$ is a self-contained set of transitions, then it is said that x is self-contained.

In this subsection, the A_k–invariant subspaces contained in $Im(\mathbf{C})$ will be characterized. This characterization will be made for the *JF-ContPN*. This is because it has been proved in [44] that if all the join transitions are removed from a *ContPN*, and all the remaining *JF* nets are observable, then the *ContPN* is observable.

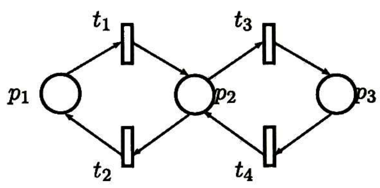


Figure 4.1.3: A simple *ContPN* with two T-Semiflows.

In a *JF-ContPN* a single dynamical matrix $A = \mathbf{C}\Lambda\Pi$ determine the behaviour of the *ContPN*. Then, the traditional methods to verify the observability property in *LS*, such as verifying the rank of the observability matrix or computing the non observable subspace $\mathcal{N}$, can be made in polynomial time. However, the characterization herein presented is relevant because it will not need the computation of matrix A, i.e. it depends only on the structure N of the *ContPN*. This characterization can be further extended to the non-*JF* case, where the number of *LS* increases exponentially, and the naive approach of using the traditional methods fail to provide an answer in polynomial time.

The analysis will start with $\ker(A) \subseteq Im(\mathbf{C})$ and the eigenvectors of the dynamical matrix $A = \mathbf{C}\Lambda\Pi$ associated to eigenvalues $\beta \neq 0$, since they are A−invariant subspaces of dimension one. Then, the general A−invariant subspaces will be characterized.

4.1.3 The *JF* case - Eigenvectors

This section is devoted to analyse the underlying structure of the eigenvectors, i.e. the sets of places and transitions in the underlying graph which are related to the eigenvector.

Take for instance the *ContPN* in Figure 4.1.3 with $\lambda = \begin{bmatrix}\lambda_1 & \lambda_2 & \lambda_3 & \lambda_4\end{bmatrix}^T$ The incidence matrix and the dynamical matrix of this *ContPN* are

$$\mathbf{C} = \begin{bmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 \\ 0 & 0 & 1 & -1 \end{bmatrix} \qquad A = \begin{bmatrix} -\lambda_1 & \lambda_2 & 0 \\ \lambda_1 & -\lambda_2 - \lambda_3 & \lambda_4 \\ 0 & \lambda_3 & -\lambda_4 \end{bmatrix}$$

First, let us analyse the vectors $v \in \ker(A)$. In Section 4.1.1 it was proved that $v \in \ker(A_k)$ if $v \in \ker(\Pi_k)$ or if there exists a vector $x \in \ker(\mathbf{C}) \cap Im(\Lambda\Pi)$. Since in the *ContPN* in Fig. 2.1.1 is *JF*, there are not any join transitions. Also this *ContPN* does not have any ending places; then, there does not exist any vector $v \in \ker(\Pi)$. However the vector $v = \begin{bmatrix}(\lambda_2/\lambda_1) & 1 & (\lambda_3/\lambda_4)\end{bmatrix}^T \in \ker(A)$. Then, the vector $x = \Lambda\Pi v = \begin{bmatrix}\lambda_2 & \lambda_2 & \lambda_3 & \lambda_4\end{bmatrix}^T$

is a T-flow. On the other hand, let B_λ be a matrix whose columns are a basis of $\ker(C\Lambda)$. In this example,

$$B_\lambda = \begin{bmatrix} \lambda_2/\lambda_1 & 0 \\ 1 & 0 \\ 0 & 1 \\ 0 & \lambda_3/\lambda_4 \end{bmatrix}$$

Clearly the linear combinations of the columns of B_λ generate T-flows $x \in Im(\Lambda)$. However, not every column of B_λ, $B_{\lambda i}$ generates an annuller v because x must also fulfil $x \in Im(\Lambda)\Pi$. In Proposition 4.1.7 it was proved that in a *JF-ContPN*, a T-flow x generates a vector $v \in \ker(A)$ iff $(\bullet\langle x\rangle)\bullet \subseteq \langle x\rangle$. Let $B_{\lambda i}$ stand for the $i-th$ column of B_λ. The support sets $\langle x\rangle$, $\bullet\langle x\rangle$ and $(\bullet\langle x\rangle)\bullet$ are:

$$\begin{array}{ccc} \langle B_{\lambda 1}\rangle = \{t_1,t_2\} & & \langle B_{\lambda 2}\rangle = \{t_3,t_4\} \\ \bullet\langle B_{\lambda 1}\rangle = \{p_1,p_2\} & , & \bullet\langle B_{\lambda 2}\rangle = \{p_2,p_3\} \\ \multicolumn{3}{c}{(\bullet\langle B_{\lambda 1}\rangle)\bullet = (\bullet\langle B_{\lambda 2}\rangle)\bullet = \{t_1,t_2,t_3\}} \end{array}$$

showing that none of the columns of B_λ generates an annuller. The vector $x = \Lambda\Pi v = \begin{bmatrix}\lambda_2 & \lambda_2 & \lambda_3 & \lambda_4\end{bmatrix}^T$ does fulfil $(\bullet\langle x\rangle)\bullet \subseteq \langle x\rangle$. Actually, $\langle x\rangle$ is self-contained (see Def. 4.1.11).

The following proposition shows that in a *JF-ContPN* a subset of transitions T_x is a self-contained set of transitions iff $(\bullet T_x)\bullet \subseteq T_x$.

Proposition 4.1.13 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JF-ContPN and $T_x \subseteq T$ be a subset of transitions*

$$(\bullet T_x)\bullet \subseteq T_x \textit{ iff } T_x \textit{ is a self-contained set of transitions.}$$

Proof: ($\Rightarrow$) Let $(\bullet T_x)\bullet \subseteq T_x$. Since N is JF, then $\mathcal{P}(T_x|\mathcal{R}) = \bullet T_x$. Now, since $T_x\bullet \subseteq T_x$, then every transition constrained by $\bullet T_x$ is contained in T_x.

($\Leftarrow$) Let T_x be a self-contained set of transitions. Again, since N is JF, then $\mathcal{P}(T_x|\mathcal{R}) = \bullet T_x$ and every transition in $\mathcal{T}(\bullet T_x|\mathcal{R}) \subseteq T_x$ (definition of self-contained set of transitions). Since N is JF, if $t_i \in p_j\bullet$ then p_j constrains t_i, then $(\bullet T_x)\bullet \subseteq T_x$. ■

Proposition 4.1.14 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JF-ContPN and $T_x \subseteq T$ be a self-contained subset of transitions. If $T_x\bullet \subseteq \bullet T_x$ then T_x is a fully self-contained set of transitions.*

Proof: It is clear since $T_x\bullet \subseteq \bullet T_x$. ■

It can be deduced from previous the proposition and Proposition 4.1.7 that in the case of the annuller eigenvectors, there exists a fully self-contained set of transitions T_x associated.

Proposition 4.1.15 *Let $\langle N, \mathbf{m}_0, \lambda\rangle$ be a JF-ContPN. If $v \in \ker(C) \cap Im(\Lambda\Pi_k)$ then $\bullet v$ is a fully self-contained set of transitions.*

Proof: $v \in \ker(\mathbf{C}) \cap Im(\Lambda\Pi)$. *In this case, v is an annuller of the matrix* $\mathbf{C}\Lambda\Pi$ *and conditions of Propositions 4.1.13 and 4.1.14 hold for $\langle\bullet v\rangle$.* ■

Now for the eigenvectors of a dynamical matrix $A = \mathbf{C}\Lambda\Pi$ associated with an eigenvalue $\beta \neq 0$, the underlying structure will be analysed.

From (4.1) it can be seen that $v \in Im(\mathbf{C}\Lambda\Pi)$. Also, when $\beta \neq 0$, $v \in Im(\mathbf{C})$, i.e. $\exists x$, such that $\mathbf{C}x = y$. The entries of the vector $x \in \mathbb{R}^{|T|}$ are the coefficients associated to a linear combination of the columns of the incidence matrix; these columns are directly associated to the transitions of the *ContPN*.

Proposition 4.1.16 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JF-ContPN.If v is an eigenvector of the dynamical matrix A, with an eigenvalue $\beta \neq 0$, then $\exists x$ such that $v = \mathbf{C}x$ and $\bullet\langle x\rangle = \langle v\rangle$.*

Proof: Since v is an eigenvector, then $\mathbf{C}\Lambda\Pi v = \beta v$, *therefore* $v \in Im(\mathbf{C}\Lambda\Pi)$. *Also, since* $\beta \neq 0 \rightarrow v \in Im(\mathbf{C})$. *Then* $\exists x \in Im(\Lambda\Pi)$ *such that* $\mathbf{C}x = v$. *Particularly,* $x = (1/\beta)\Lambda\Pi v$ *fulfils* $\mathbf{C}x = v$. *It is clear that* $\Lambda\Pi v$ *maps to every output transition of* $\langle v\rangle$. *Since N is a JF-PN, then* $\langle v\rangle = \bullet\langle x\rangle$, *i.e.* $\langle v\rangle$ *contains every input place to the transitions in* $\langle x\rangle$. ■

The converse of the previous proposition is not true in general. Take for instance the *ContPN* in Fig. 2.1.1. A vector $x = \begin{bmatrix}\gamma_1 & \gamma_2 & 0 & 0\end{bmatrix}^T$ with $\gamma_1, \gamma_2 \neq 0$ has $\langle x\rangle = \{t_1, t_2\}$. Its input set is $\bullet\langle x\rangle = \{p_1, p_2\}$. The vector $y = \mathbf{C}x = \begin{bmatrix}-\gamma_1 + \gamma_2 & \gamma_1 - \gamma_2 & 0\end{bmatrix}^T$ has the support $\langle y\rangle = \{p_1, p_2\}$ which actually fulfils the conditions of Proposition 4.1.16. However y is not an eigenvector of A with $\beta \neq 0$ since

$$A\begin{bmatrix}-\gamma_1+\gamma_2\\ \gamma_1-\gamma_2\\ 0\end{bmatrix} = \begin{bmatrix}-\lambda_1(-\gamma_1+\gamma_2)+\lambda_2(\gamma_1-\gamma_2)\\ \lambda_1(-\gamma_1+\gamma_2)+(-\lambda_2-\lambda_3)(\gamma_1-\gamma_2)\\ \lambda_3(\gamma_1-\gamma_2)\end{bmatrix}^T$$

Proposition 4.1.17 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JF-ContPN. If v is an eigenvector of the dynamical matrix A, with an eigenvalue $\beta \neq 0$, then $\exists x$ such that $v = \mathbf{C}x$ and $(\bullet\langle x\rangle)\bullet \subseteq \langle x\rangle$.*

Proof: Based on Proposition 4.1.16, there exists x such that $\langle y\rangle = \bullet\langle x\rangle$. In addition, because of the mapping $\Lambda\Pi$, *every output transition of $\langle v\rangle$ is contained in $\langle x\rangle$. Then $(\bullet\langle x\rangle)\bullet \subseteq \langle x\rangle$.* ■

The converse of the previous proposition is not true. Take for instance the *ContPN* in Fig. 2.1.1 . Transition t_1 fulfils $(\bullet\langle x\rangle)\bullet \subseteq \langle x\rangle$. However the vector $\mathbf{e}_1$, associated to $\bullet t_1 = \{p_1\}$, is not an eigenvector since $A\mathbf{e}_1 = -\lambda_1\mathbf{e}_1 + \lambda_1\mathbf{e}_2$.

Propositions 4.1.16 and 4.1.17 show that if v is an eigenvector associated to an eigenvalue $\beta \neq 0$, then $\exists x \in \mathbb{R}^{|T|}$ such that $\mathbf{C}x = v$, $\langle v \rangle \bullet = \langle x \rangle$ and $\langle v \rangle = \bullet \langle x \rangle$ are fulfilled. This means that there exists a set of self-contained transitions T_x associated to the eigenvector v (in this case, given by $\langle x \rangle$).

Now, it is easy to see that if a transition $t_i \in \langle x \rangle$, then the marking in the places $p_j \in \bullet t_i \cup t_i \bullet$ is affected, i.e. there is a marking change in these places due the firing of transition t_i. Take for instance the *ContPN* example in Figure 2.1.1. A vector $x_1 = \gamma_1 \mathbf{e}_1$ with $\gamma_1 \in \mathbb{R}$ has $\mathbf{C}x_1 = \begin{bmatrix} -\gamma_1 & \gamma_1 & 0 \end{bmatrix}^T$, i.e. it removes marks from place p_1 and adds marks to place p_2. Similarly the vector $x_4 = \gamma_4 \mathbf{e}_4$ affects the marking in places p_3 and p_2. Now, let $x_a = x_1 + x_4$. Then $\mathbf{C}x_a = \begin{bmatrix} -\gamma_1 & \gamma_1 + \gamma_4 & -\gamma_4 \end{bmatrix}^T$ If $\gamma_1 = -\gamma_4$, then $\mathbf{C}x_a = \begin{bmatrix} \gamma_4 & 0 & -\gamma_4 \end{bmatrix}^T$, i.e. it only changes the marking of places p_1 and p_3, but not p_2's. However, the existence of a linear combination of the columns of the incidence matrix $\mathbf{C}$ (in the example $\langle x \rangle = \{t_1, t_4\}$) does not guarantee that there exists an eigenvector associated to places $p_j \in \langle v \rangle = \bullet \langle x \rangle$ (in the example, $\bullet \langle x \rangle = \langle v \rangle = \{p_1, p_3\}$). For the example in Fig. 2.1.1, let the transition's firing rates be $\lambda = \begin{bmatrix} \lambda_1 & \lambda_2 & \lambda_3 & \lambda_4 \end{bmatrix}^T$. Now, to find an eigenvector v such that $p_1, p_3 \in \langle v \rangle$ but $p_2 \notin \langle v \rangle$, let $v = \begin{bmatrix} v_1 & 0 & v_3 \end{bmatrix}^T$. Such eigenvector is given by the solution of the equation:

$$Av = v_1 \begin{bmatrix} -\lambda_1 \\ \lambda_1 \\ 0 \end{bmatrix} + v_3 \begin{bmatrix} 0 \\ \lambda_4 \\ -\lambda_4 \end{bmatrix} = \beta \begin{bmatrix} v_1 \\ 0 \\ v_3 \end{bmatrix}$$

In this example, the vector $\begin{bmatrix} 1 & -1 & 0 \end{bmatrix}^T$ is an eigenvector when $\lambda_1 = \lambda_4$, associated to the eigenvalue $-\lambda_1$. However, if $\lambda_1 \neq \lambda_4$, then it is not possible to find an eigenvector v associated to places p_1 and p_3 exclusively, i.e., an eigenvector v such that $p_1, p_3 \in \langle v \rangle$ and $p_2 \notin \langle v \rangle$. This is showed formally in the next proposition. The next proposition is generalized even for the *FC–ContPN* class of nets, but clearly it also holds for the *JF* case.

Proposition 4.1.18 *Let $\langle N, \lambda, \mathbf{m}_0 \rangle$ be a FC-ContPN and p_j be an attribution place. Also, let $\forall t_i \in \bullet p_j$, $|\bullet t_i| = 1$, i.e. each t_i has only one input place p_i such that $p_i \bullet = \{t_i\}$. If $\exists \lambda_{i_1} = \lambda_{i_2}$ for some transition $t_{i_1}, t_{i_2} \in \bullet p_j$ and $(\bullet p_j) \bullet = \{p_j\}$ then a vector $v_{a_i} = \mathbf{e}_{l_1} - \mathbf{e}_{l_2}$, associated to places $\{p_{l_1}\} = \bullet t_{i_1}$ and $\{p_{l_2}\} = \bullet t_{i_2}$ is an eigenvector of all matrices $\mathbf{C}\Lambda\Pi_k$. The vector v_{a_i} is called* ***Attribution eigenvector****.*

Proof: The $i_1 - th$ column of $\mathbf{C}$ is $\mathbf{C}_{i_1} = \mathbf{e}_j^T - \mathbf{e}_{l_1}^T$, i.e. it adds one token to place p_j and removes one from p_{l_1}. Similarly, the $i_2 - th$ column $\mathbf{C}_{i_2} = \mathbf{e}_j^T - \mathbf{e}_{l_2}^T$. Since $\forall t_i \in \bullet p_j$, $|\bullet t_i| = 1$ then t_{i_1} is constrained by p_{l_1} and t_{i_2} is constrained by p_{l_2} and neither p_{l_1} nor p_{l_2} constrain the *flow of any other transition. Then in matrix* $\mathbf{C}\Lambda\Pi_k$ *its* $l_1 - th$ *column will be* $\lambda_{i_1}\mathbf{C}_{i_1}$ *and its* $l_2 - th$ *column will be* $\lambda_{i_2}\mathbf{C}_{i_2}$. *Then* $\mathbf{C}\Lambda\Pi_k\mathbf{e}_{l_1} = \lambda_{i_1}\mathbf{C}_{i_1}$ *and* $\mathbf{C}\Lambda\Pi\mathbf{e}_{l_2} = \lambda_{i_2}\mathbf{C}_{i_2}$.

Let $v_{a_i} = \mathbf{e}_{l_1} - \mathbf{e}_{l_2}$, $\mathbf{C}\Lambda\Pi v_{a_i} = \mathbf{C}\Lambda\Pi(\mathbf{e}_{l_1} - \mathbf{e}_{l_2}) = \lambda_{i_1}\mathbf{C}_{i_1} - \lambda_{i_2}\mathbf{C}_{i_2} = \lambda_{i_1}(e_j^T - e_{l_1}^T) - \lambda_{i_2}(\mathbf{e}_j^T - \mathbf{e}_{l_2}^T)$. *Since* $\lambda_{i_1} = \lambda_{i_2}$, $\mathbf{C}\Lambda\Pi v_{a_i} = -\lambda_{i_1}\mathbf{e}_{l_1}^T + \lambda_{i_1}\mathbf{e}_{l_2}^T = -\lambda_{i_1}(\mathbf{e}_{l_1} - \mathbf{e}_{l_2})$ *which is the definition of eigenvector.* ■

In order to find the eigenvectors associated to attribution places, it suffices to find those attribution places and their attributing transitions. It must be verified that each attribution transition does not have any output place other than the attribution place itself. Also, each input place to the attribution transition should only be connected with the attributing transition. Then, for each pair of attributing transitions $t_i, t_j \in \bullet p_k$ with $\lambda_i = \lambda_j$ and $\bullet t_i = \{p_{l_i}\}$ and $\bullet t_j = \{p_{l_j}\}$, the vector $v_a = \mathbf{e}_{l_i} - \mathbf{e}_{l_j}$ will be an eigenvector of each $\mathbf{C}\Lambda\Pi_k$. This is important, since matrices $\mathbf{C}\Lambda\Pi_k$ are not needed to compute the attributing eigenvector v_a.

For the case of attribution eigenvectors, there is also a self-contained set of transitions associated, as shown in the next proposition.

Proposition 4.1.19 *Let* $\langle N, \lambda, \mathbf{m}_0\rangle$ *be a JF-ContPN. Let* v *be an attribution eigenvector. Then* $\bullet v$ *is a self-contained set of transitions.*

Proof: It is clear since the effects of transitions in $\bullet v$ *only affect places in* $\bullet\bullet v$ *but they do not affect the attribution place.* ■

The previous proposition shows that when there exist attribution eigenvectors, then the associated set of transitions is only self-contained, but not fully self-contained.

Besides eigenvectors associated to attributions and similarly to the eigenvectors associated to ending places, it is possible to characterize eigenvectors associated to ending transitions when $(\bullet t_i)\bullet = t_i$ (it will be assumed that if $\forall t_i \in p_j\bullet$, $t_i\bullet = \emptyset$, those ending transitions are represented as only one ending transition). The next proposition is generalized for every class of *ContPN*, therefore clearly holds for the *JF* case.

Proposition 4.1.20 *Let* t_i *be an ending transition of a ContPN. If* $(\bullet t_i)\bullet = \{t_i\}$, *then a vector corresponding to each place* $p_j \in \bullet t_i$, $\mathbf{e}_j$ *is an eigenvector of matrix* $\mathbf{C}\Lambda\Pi_k$. *The vector* $\mathbf{e}_j \sim p_j$ *will be named ending transition eigenvector.*

Proof: There are two cases:

a) Let $\{p_j\} = \bullet t_i$. *Then the* $i - th$ *column of* $\mathbf{C}$ *has only one negative element corresponding to the arc from* p_j *to* t_i. *In addition, every matrix* Π_k *has its* $i - th$ *row equal to some elementary vector* $\mathbf{e}_j$. *Then, the matrix* $\mathbf{C}\Lambda\Pi_k = \begin{bmatrix}\mathbf{a}_1 & \lambda_i\mathbf{C}_i & \mathbf{a}_2\end{bmatrix}$ *where* $\mathbf{C}_i$ *is the* $i - th$

column of $\mathbf{C}$ *and* $\mathbf{a}_1$ *and* $\mathbf{a}_2$ *are two matrices with* n *rows and* $j-1, n-j$ *columns respectively. Clearly the elementary vector* $\mathbf{e}_j$ *is an eigenvector of* $\mathbf{C}\Lambda\Pi_k$ *since* $\mathbf{C}\Lambda\Pi_k \cdot \mathbf{e}_j = -\lambda_i \cdot \mathbf{e}_j$*, where* $-\lambda_i$ *is the eigenvalue associated to* $\mathbf{e}_j$.

If $|\bullet t_i| > 1$ *then each place* $p_l \in \bullet t_i | p_l \bullet = t_i$ *is either constraining the flow of* t_i *and the previous reasoning applies or it is not constraining it and the place* p_l *is not constraining any transition. If so, then the* $l - th$ *column* $\mathbf{C}\Lambda\Pi_k$ *is zero and* $\mathbf{e}_l$ *is also an eigenvector of* $\mathbf{C}\Lambda\Pi_k$.

■

For the case of ending transition eigenvectors, there is also a fully self-contained set of transitions associated, as shown in the next proposition.

Proposition 4.1.21 *Let* $\langle N, \lambda, \mathbf{m}_0\rangle$ *be a JF-ContPN. Let* $\mathbf{e}_j$ *be an ending transition eigenvector. Then* $\mathbf{e}_j\bullet$ *is a fully self-contained set of transitions.*

Proof: *Since* $\mathbf{e}_j$ *is an ending transition eigenvector, then* $(\bullet t_i)\bullet = \{t_i\}$ *holds. Also, since* $T_i\bullet = \emptyset$*, then the conditions of Proposition 4.1.14 also hold.* ■

4.1.4 The *JAF* case. A_k–invariant subspaces

Based on the previous examples and propositions, there is the intuition that the A–invariant subspaces $\mathcal{V} \subseteq Im(\mathbf{C})$ have a special structure on the *ContPN*. In this section, this intuition will be proved for the join attribute free class of *ContPN*.

Definition 4.1.22 *Let* $\mathcal{V}_B = \{v_1, ..., v_n\}$ *be a basis of the A–invariant subspace* $\mathcal{V}$*. The support of an invariant subspace* $\mathcal{V}$*, denoted by* $\langle\mathcal{V}\rangle$ *is*

$$\langle\mathcal{V}\rangle = \bigcup_{v_i \in \mathcal{V}_B} \langle v_i\rangle \tag{4.3}$$

Proposition 4.1.23 *Let* $\langle N, \lambda, \mathbf{m}_0\rangle$ *be a JF-ContPN and* $\mathcal{V} \subseteq Im(\mathbf{C})$ *be an* A–*invariant subspace. Then* $\forall v \in \mathcal{V}$, $\exists x \in \mathbb{R}^{|T|}$ *such that* $\mathbf{C}x = v$ *and* $\langle v\rangle\bullet = \langle x\rangle$.

Proof: Clearly $v \in Im(\mathbf{C}\Lambda\Pi)$ and $v \notin \ker(\mathbf{C}\Lambda\Pi)$. Then $v \in Im(\mathbf{C})$. Now, since v is A–invariant, then $\mathbf{C}\Lambda\Pi v = v_1 \in \mathcal{V}$. Let $x = \Lambda\Pi v$. It is clear that $\Lambda\Pi$ maps to every output transition of $\langle v\rangle$ because N is *JF*. Then $\langle v\rangle\bullet = \langle x\rangle$. ■

Clearly since previous the propositions hold for the *JF* class of *ContPN*, then they also hold fort he *JAF* class. Proposition 4.1.23 states that for every vector $v \in \mathcal{V}$ it is possible to make $\mathbf{C}x = v$ with a linear combination of columns of $\mathbf{C}$ associated to output transitions of $\langle v\rangle$.

Proposition 4.1.24 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JF-ContPN and $\mathcal{V} \subseteq Im(\mathbf{C})$ be an $A-$invariant subspace. Then $\exists T_v$, a set of self-contained transitions associated to the $A-$invariant subspace $\mathcal{V}$, i.e.*

1. $\bullet T_v = \langle \mathcal{V} \rangle$.
2. $\langle \mathcal{V} \rangle \bullet = T_v$.

Proof: 2 is a consequence of Proposition 4.1.23. 1 is a consequence of the fact that N if *JF*. ■

Propositions 4.1.23 and 4.1.24 show that for any invariant subspace $\mathcal{V}$, there exists a set of self-contained transitions T_v, i.e. $\bullet T_v = \langle \mathcal{V} \rangle$ and $\langle \mathcal{V} \rangle \bullet = T_v$; this means that the *effects* of the transitions in T_v in the marking of the *ContPN* is restricted only to the places $p_j \in \langle \mathcal{V} \rangle$.

It is important to notice that in Proposition 4.1.24, the *effects* of the transitions $t_i \in T_v$ are contained only in places $p_j \in \bullet T_v$, but it states nothing about the output places of the transitions in T_v.

Now, the following proposition shows that if $\mathcal{V}$ is an $A-$invariant subspace in a *JAF-ContPN* and T_v is its associated set of self-contained set of transitions then $\forall p_j \in T_v\bullet$, the place $p_j \in \langle \mathcal{V} \rangle$.

Proposition 4.1.25 *Let $\langle N, \lambda, \mathbf{m}_0\rangle$ be a JAF-ContPN and $\mathcal{V} \subseteq Im(\mathbf{C})$ be an $A-$invariant subspace. Also, let T_v be the self-contained set of transitions associated to $\mathcal{V}$. Then $\bullet T_v \cup T_v \bullet \subseteq \langle \mathcal{V} \rangle$.*

Proof: From Proposition 4.1.24, $\bullet T_v \subseteq \langle \mathcal{V} \rangle$. Then it will only be proved that $\forall p_j \in T_v\bullet$, $p_j \in \langle \mathcal{V} \rangle$. Let $p_j \in \bullet t_i$ where $t_i \in T_v$, such that p_h is the place constraining t_i, i.e. $\mathcal{P}(t_i|\mathcal{C}) = p_h$. Let $v \in \mathcal{V}$ such that $p_h \in \langle v \rangle$. Clearly $Av = v_1 \in \mathcal{V}$ contains the effects of every transition $t_g \in p_h\bullet$, particularly, it contains the effect of $t_i \in \bullet p_j$, since it cannot be cancelled by the effect of another transition (the net is *JAF*). Therefore $p_j \in \langle v_1 \rangle$.

Let $v_1, v_2 \in \mathcal{V}$ such that $p_h \in \langle v_1 \rangle \cap \langle v_2 \rangle$. Clearly $\alpha_1 v_1 + \alpha_2 v_2 = v_3 \in \mathcal{V}$, with $\alpha_i \in \mathbb{R}$, $i = 1, 2$. ■

Now it will be proved that if such fully self-contained set T_v of transitions exists, then there exists an invariant subspace $\mathcal{V} \subseteq Im(\mathbf{C})$ associated to places in $\bullet T_s$.

Definition 4.1.26 *A set of transitions T_v is said to generate an $A-$invariant subspace $\mathcal{V}$ if an $A-$invariant subspace $\mathcal{V}$ exists such that $\langle \mathcal{V} \rangle \subseteq \bullet T_v$.*

Proposition 4.1.27 *Let $\langle N, \lambda, \mathbf{m}_0 \rangle$ be a connected JAF-ContPN and $T_v \subseteq T$ be a fully self-contained set. Then T_v generates an $A-$invariant subspace $\mathcal{V} \in Im(\mathbf{C})$.*

Proof: Let the index of the transitions and places be assigned in such a way that $T_v = \{t_1, t_2, ..., t_n\} \subseteq T$ and $P_v = \bullet T_v = \{p_1, p_2, ..., p_m\} \subseteq T$. The incidence matrix has the form:

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_v & \mathbf{C}_v^+ \\ 0 & \mathbf{C}_{\bar{v}} \end{bmatrix}$$

where $\mathbf{C}_v$ is the incidence matrix of the subnet $N_v = \langle P_v, T_v, \mathbf{Pre_v}, \mathbf{Pre_v} \rangle$. It is clear that $\forall t_i \in T_v$, $t_i \bullet \subseteq P_v$ and $\bullet t_i \subseteq P_v$, i.e. t_i has only input and output places $p_j \in P_v$. Also, since N is *JF* and T_v is fully self-contained, $\forall p_j \in P_v$, $p_j \bullet \subseteq T_v$, i.e. the output transitions of the places $p_j \in P_v$ are contained in the set T_v. $\mathbf{C}_v^+$ represents the input transitions to the places $p_j \in P_v$. It is clear that at least one place $p_j \in P_v$ has an input transition $t_h \in T - T_v$, otherwise the net would not be connected. Finaly $\mathbf{C}_{\bar{v}}$ represents the rest of the incidence matrix. Using the same reasoning, the configuration matrices Λ and Π have the following structure:

$$\Lambda = \begin{bmatrix} \Lambda_v & 0 \\ 0 & \Lambda_{\bar{v}} \end{bmatrix} \quad \text{and} \quad \Pi = \begin{bmatrix} \Pi_v & 0 \\ 0 & \Pi_{\bar{v}} \end{bmatrix}$$

where Λ_v and $\Lambda_{\bar{v}}$ are the transitions' firing rates associated to the transitions $t_i \in T_v$ and the transitions $t_h \notin T_v$ respectively. The configuration matrix Π can be similarly separated since the transitions $t_i \in T_v$ are constrained by places in P_v and the places $p_j \in P_v$ only constrains transitions $t_i \in T_v$. Then, the dynamical matrix $A = \mathbf{C}\Lambda\Pi$ is given by:

$$A = \begin{bmatrix} \mathbf{C}_v \Lambda_v \Pi_v & \mathbf{C}_v^+ \Lambda_{\bar{v}} \Pi_{\bar{v}} \\ 0 & \mathbf{C}_{\bar{v}} \Lambda_{\bar{v}} \Pi_{\bar{v}} \end{bmatrix}.$$

Now, let $\mathcal{V} = \{v | v_v \in \mathbb{R}^m$, and $v_v(h) = 0$, $h = m+1, m+2, ..., |P|\}$, i.e. $v = \begin{bmatrix} v_v & 0 \end{bmatrix}^T \subseteq \mathcal{V}$. Clearly $Av \subseteq \mathcal{V}$. ■ ■

Theorem 4.1.28 *Let $\langle N, \lambda, \mathbf{m}_0 \rangle$ be a JAF-ContPN. $T_x \subseteq T$ is fully self-contained iff T_x generates an invariant subspace $\mathcal{V}$.*

Proof: It follows from Propositions 4.1.25 and 4.1.27. ■

4. ON INVARIANT SUBSPACES AND THE *CONTPN* STRUCTURE

5

Sensor placement for observability in *ContPN*

This chapter presents a sensor placement strategy which guarantees observability of a *ContPN*. The problem consists on determining which places should be measured so that later on, the marking of every place $p_j \in P$ can be computed trough an observer.

Even when the observability in infinitesimal time in autonomous *SLS* has been completely characterized in [48] and [49], it has been pointed out that this approach is not feasible in practice. Similarly, the observability characterization for general *ContPN* provided in [46] also leads to high complexity algorithms.

However, using the results in [48] and [49] it is known that a *SLS* is observable iff every *LS* is observable and distinguishable from each other.

Therefore, the strategy herein presented will determine where to place sensors to guarantee observability in each *LS* of the *SLS* representation of the *ContPN* and the distinguishability among them. This approach, however, may lead to observability by excess, i.e. it may add more sensors than the strictly necessary. Then a sensor reduction strategy is also presented.

For the distinguishability between each pair of *LS*, it will be shown that for some classes of *ContPN* observability in each *LS* implies distinguishability as well, but also that the classical test for distinguishability in *SLS* fails to provide a correct answer in the *ContPN* context, since the information provided by the *ContPN* is greater than the one available in the general *SLS*.

5.1 Sensor placement for Observability in each *LS*

Since every A_k−invariant subspace of each matrix A_k must not be in $\ker(S)$ for the *ContPN* to be observable, it is possible to design S such that neither of the eigenvectors and subspaces previously characterized in 4 is contained in $\ker(S)$. For every other A_k−invariant subspace which is not characterized, it is known that it is also contained in $Im(\mathbf{C})$. The $Im(\mathbf{C})$ contains more elements than those invariants, but if $Im(\mathbf{C})$ is not in $\ker(S)$, clearly neither the A_k−invariant subspaces are. The following definitions will allow us to define an *invariant matrix* and to propose an algorithm to determine where to place sensors in a *ContPN* so that observability is guaranteed.

Definition 5.1.1 *Let a place $p_i \in P \cap P_E$ (i.e. p_i is an ending place). Let $E_{Ep} = \{e_i | e_i$ is an elementary vector associated to the ending place $p_i\}$ be the set of elementary vectors associated to the ending places. Then, the* ***Ending Places Matrix*** *is defined as*

$$V_{Ep} = \begin{bmatrix} e_i & ... & e_j \end{bmatrix} \tag{5.1}$$

where $e_i, ..., e_j \in E_E$.

Definition 5.1.2 *Let $p_i \in P$ such that $p_i\bullet = \{t_j\}$ for some $t_j \in T_E$ (i.e. t_j is an ending transition and p_i constrains its flow). Let $E_{Et} = \{e_i | e_i$ is an elementary vector associated to the place p_i constraining the flow of an ending transition $t_j\}$ be the set of elementary vectors associated to the places constraining the ending transitions $t_j \in T_E$.*

The ***Ending Transitions Matrix*** *is defined as*

$$V_{Et} = \begin{bmatrix} e_i & ... & e_j \end{bmatrix} \tag{5.2}$$

where $e_i, ..., e_j \in E_{Et}$.

Definition 5.1.3 *Let v_{a_i} be an Attribution eigenvector as the ones computed with Proposition 4.1.18. Let $E_A = \{v_{a_i} | \ v_{a_i}$ is an Attribution eigenvector$\}$. Then, the* ***Attribution Eigenvector Matrix*** *is*

$$V_A = \begin{bmatrix} v_{a_1} & ... & v_{a_n} \end{bmatrix} \tag{5.3}$$

where $v_{a_1}, ..., v_{a_n} \in E_A$.

Definition 5.1.4 *Let v_i be an Annuler vector as the ones computed with Proposition 4.1.7. Let $E_{AP} = \{v_i | v_i$ is an Annuler vector$\}$. Then, the* ***Annuler Matrix*** *V_{AP} is a matrix with all Annuler Vectors*

$$V_{AP} = \begin{bmatrix} v_1 & ... & v_n \end{bmatrix} \tag{5.4}$$

where $v_1, ..., v_n \in E_{AP}$.

Definition 5.1.5 *Let $p_i \in P \cap \bullet t_j$ where t_j is a Join transition. Let $E_{SP} = \{e_i^j | e_i^j$ is an elementary vector associated to a place $p_i \in \bullet t_j$ and t_j is a Join transition$\}$ be the set with all elementary vectors associated to places $p_i \in \bullet t_j$, where t_j is a Join transition. Then, the* ***Synchronized Places Annuler Matrix*** *V_{SP} is*

$$V_{SP} = \begin{bmatrix} e_i^j & \dots & e_h^g \end{bmatrix} \tag{5.5}$$

$e_i^j, \dots, e_h^g \in E_{SP}$.

With the previous definitions, some important matrices will be defined, which will be further used to compute an output matrix such that the observability of the *ContPN* is guaranteed.

Definition 5.1.6 *Let the matrices V_{Ep}, V_{AP} and V_{SP} be as defined in (5.1), (5.4) and (5.5) respectively. The* ***Invariant Kernel Matrix*** *M_K is defined as*

$$M_K = \begin{bmatrix} V_{Ep} & V_{AP} & V_{SP} \end{bmatrix}. \tag{5.6}$$

Definition 5.1.7 *Let the matrices M_K, V_{Et} and V_A, be as defined in (5.6), (5.2) and (5.3) respectively. The* ***Invariant Matrix*** *M_I is defined as*

$$M_I = \begin{bmatrix} M_K & V_{Et} & V_A \end{bmatrix}. \tag{5.7}$$

It is important to notice that M_I has every invariant subspace characterized in this work. V_{Ep} are the invariant subspaces associated to ending places, V_{Et} are the invariant subspaces associated to ending transitions, V_A are the invariant subspaces associated to attribution places, V_{AP} are the invariant subspaces associated to T-Semiflows and they are annulers for some matrix $\mathbf{C}\Lambda\Pi_k$ and V_{SP} are the invariant subspaces associated to Join transitions. Clearly, for the *ContPN* to be observable, each of these invariant subspaces must not belong to $\ker(S)$. However, there are still some invariant subspaces associated to some nonzero eigenvalues which are not represented in M_I. Nevertheless, these invariant subspaces are in $Im(\mathbf{C})$ and as previously discussed, if $Im(\mathbf{C})$ is not in $\ker(S)$, neither the invariant subspaces of $\mathbf{C}\Lambda\Pi_k$ are. In order to consider $Im(\mathbf{C})$ to determine sensor placement in a *ContPN*, the next definition is introduced.

Definition 5.1.8 *Let M_I be the Invariant Matrix of a ContPN and $\mathbf{C}_I = \begin{bmatrix} \mathbf{c}_{I1} & \mathbf{c}_{I2} & \dots & \mathbf{c}_{In} \end{bmatrix}$ be a matrix with n linearly independent columns of $\mathbf{C}$. The* ***Extended Invariant Matrix*** *M_{IE} is defined as*

$$M_{IE} = \begin{bmatrix} M_I & \mathbf{C}_I \end{bmatrix}. \tag{5.8}$$

Using the extended invariant matrix of a *ContPN*, it is now possible to present an algorithm to determine sensor placement in a *ContPN* that guarantees observability. The next algorithm will be used with the extended invariant matrix M_{IE} as an input; however, it is presented in terms of a general input matrix M_{input}, because it will later be used with another input matrix. As a notation $S(i, \bullet)$ represents the $i-th$ row of the output matrix S, and $M(\bullet, j)$ represents the $j-th$ column of matrix M.

Algorithm 5.1.9 ***Sensor placement algorithm.***

```
Function: $S = SensorPlacement(M_{input})$.
Inputs: A invariant matrix $M_{input}$.
Outputs: The output matrix $S$ such that $S \cdot M_{input} \neq 0$.
Initialize: The matrix of orthogonal columns to $S$ as zero: $OS = 0$. Initialize the counter sensor $num = 1$. Initialize the temporary variable $Temp = 0$.

1. Build the linearly independent elementary vector set $LIS$ as the set of all columns of $M_I$ which are linearly independent from each other and these columns are scaled elementary vectors, i.e. $v \in LIS$ if $v = \delta_i e_i$ for some $\delta \in \mathbb{R}$ (a scaled elementary vector).
2. $\forall \delta_i e_i \in LIS$, add a new sensor for place $p_i$, i.e. a new row $num-th$, i.e. $S(num, \bullet) = e_i^T$ and increment $num$ ($num = num + 1$) – Vectors in $LIS$ are no longer in $\ker(S)$.
3. While there exist columns in $M_{input}$
    (a) $Temp = M_{input}$.
    (b) For $i = 1$ to the number of columns of $Temp$
        i. If $S \cdot Temp(\bullet, i) = 0$ then
           $OS = \begin{bmatrix} OS & Temp(\bullet, i) \end{bmatrix}$
           else remove column $M_{input}(\bullet, i)$ from $M_{input}$.
           – If the $i-th$ column of $M_{input}$ is in $\ker(S)$ this column is added to $OS$ otherwise it is removed from $M_{input}$.
    (c) Find the row $r_i$ in $OS$ with the largest number of elements different from zero.
    (d) Add a new row $num$ to $S$ equal to $e_i^T$ and increment $num$ ($num = num + 1$) – at least one column of $M_{input}$ is now removed from $\ker(S)$.
   End While
4. Return $S$.
```

The previous algorithm ensures that none of the columns of M_{input} is in $\ker(S)$. Step 1 and 2, remove every elementary vector in M_{input} from $\ker(S)$. By removing these elementary vectors, some other columns of M_{input} may also be removed from $\ker(S)$. Step 3 removes from M_{input} those columns which no longer belong to $\ker(S)$ and puts in a matrix OS all columns which still are in $\ker(S)$. Then, it chooses the row of OS with more non-zero values, say the $i-th$ row, and places a sensor in place p_i. The algorithm is repeated until every column in M_{input} is not in $\ker(S)$ and finally returns the value of the computed output matrix S.

Using Algorithm 5.1.9 with the invariant matrix M_{IE} as an input, i.e.

$$S = SensorPlacement(M_{IE}),$$

it will return an output matrix such that each column of matrix M_{IE} is removed from $\ker(S)$. However, some A_k-invariant subspaces may still be in $\ker(S)$. In order to avoid this problem, the following algorithm will verify if $Im(\mathbf{C})$ is removed from $\ker(S)$. Similarly to Algorithm 5.1.9, it is presented in terms of a more general input matrix M_{input}.

Algorithm 5.1.10 ***Validation algorithm***

```
  Function: $S = Validation(M_{input}, S)$
  Inputs: An output matrix $S = \begin{bmatrix} S_1^T & \dots & S_{num}^T \end{bmatrix}^T$ obtained from Algorithm 5.1.9.
  A matrix $M_{input}$ with only linearly independent columns and with $k = rank(M_{input})$.
  Outputs: The output matrix $S$ such that $Im(M_{input}) \not\subseteq \ker(S)$.
  Initialize: $f = \begin{bmatrix} 1 & \dots & 1 \end{bmatrix} \in 1^k$, $j = 1$, $i = n$ and $z = 1$.

  1. While $z \neq 0$
     (a) Solve the LPP:
     $\max\ z = f\alpha,$
     $s.t.$
     $S \cdot M_{input}\alpha = 0$
     $\alpha_i \leq 1$
     (a) If $z \neq 0$ then
         i. compute $Q = M_{input} \cdot \alpha$.
         ii. choose any row $h$ of $Q$ such that $Q_h \neq 0$.
```

iii. add the $(num + 1) - th$ row equal to $\mathbf{e}_h$ to S and increment the value num ($num = num + 1$) – places a sensor in place p_h.

Let $S = SensorPlacement(M_{IE})$, which is an output matrix such that none of the columns of M_{IE} is contained in $\ker(S)$. However, a linear combination of the kernel invariant matrix M_K, which also belongs to $\ker(A_k)$ may still be in $\ker(S)$. Then, it should be computed $S_0 = Validation(M_k, S)$, such that neither the columns of M_K or any of its linear combinations are contained in $\ker(S_1)$. Then, in order to obtain an output matrix such that the *ContPN* is observable, it is necessary to compute $S_1 = Validation(\mathbf{C}_I, S_0)$, which guarantees observability of the *ContPN*. However, S_1 guarantees observability by excess, i.e. it has more sensors than the strictly needed for the *ContPN* to be observable. This occurs since Algorithm 5.1.10 designs S_1 such that $Im(\mathbf{C}) \not\subseteq \ker(S_1)$, but as previously discussed, $Im(\mathbf{C})$ is larger than the A_k−invariant subspaces.

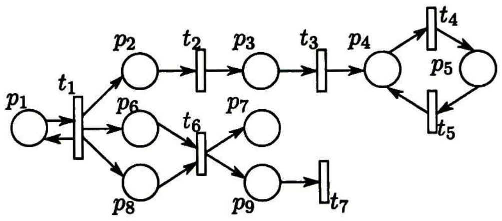


Figure 5.1.1: An illustrative example of a *ContPN* for observability.

Example 5.1.11 *Let us consider the ContPN in Figure 5.1.1. Let* $\lambda = \begin{bmatrix}1 & 2 & 3 & 4 & 3 & 5 & 6\end{bmatrix}^T$

The matrix M_{IE}, composed by one ending place vector, one annuler vector and two synchronized place annuler vectors is:

$$M_K = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & \gamma_1 & 0 & 0 \\ 0 & \gamma_2 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

where $\gamma_1, \gamma_2 \neq 0$ (the actual values are not relevant). The complement of matrix M_{IE}, composed by one ending transition vector, one attribution eigenvector and the linearly independent columns of the incidence matrix $\mathbf{C}$, *represented by* $\mathbf{C}_I$ *is*

$$[V_{Et}\ V_A\ \mathbf{C}_I] = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 1 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 & 0 & 0 \\ 0 & -1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 & -1 \end{bmatrix}$$

The output matrix obtained from Algorithm 5.1.9 with the function

$$S = SensorPlacement(M_{IE})$$

is $S = [\mathbf{e}_7\ \ \mathbf{e}_6\ \ \mathbf{e}_8\ \ \mathbf{e}_9\ \ \mathbf{e}_3\ \ \mathbf{e}_4]^T$

In order to validate that there exists no other vector $v \in \ker(A_k)$ contained in $\ker(S)$, it is computed $S_0 = Validation(M_k, S)$. In this case, $S_0 = S$, i.e. there is not any linear combination of the columns of M_K contained in $\ker(S)$. Finally, to make sure that $Im(\mathbf{C})$ is not contained in $\ker(S_0)$, it is computed $S_1 = Validation(\mathbf{C}_I, S_0)$. In this example, there exists a linear combination of the columns of $\mathbf{C}_I \in \ker(S_0)$. The vector $Q = \begin{bmatrix} 0 & -1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}^T \in \ker(S_0)$. Then a sensor in place p_2 is added and $S_1 = [\mathbf{e}_7\ \ \mathbf{e}_6\ \ \mathbf{e}_8\ \ \mathbf{e}_9\ \ \mathbf{e}_3\ \ \mathbf{e}_4\ \ \mathbf{e}_2]^T$ S_1 guarantees observability of each LS in the SLS representation of the ContPN.

5.1.1 Sensor reduction

As previously discussed, Algorithms 5.1.9 and 5.1.10 are used to compute an output matrix S_1 which guarantees, by excess, that the *ContPN* is observable. Now, in [46] it was proved that if there exists a *JAF-path* from a place p_i to a measured place p_j, then the marking of place p_i can be computed in infinitesimal time. Then, it is possible to reduce the number of sensors and still guarantee observability of the *ContPN*.

In order to reduce the number of sensors in a *ContPN* the following definitions and algorithms are presented.

For the following algorithm let $\omega = n_1 n_2 ... n_g$ be a finite path. The notation ωn_h stands for the new path $\omega = n_1 n_2 ... n_g n_h$. Also, the notation $final(\omega)$ stands for the final node of the path ω. Finally the notation $\vec{\omega} \in (\mathbb{N} \cup 0)^{|T|+|P|}$ is a vector which contains the number of times that the node n_i appears in the path ω, according to the indexation $\begin{bmatrix} p_1 & ... & p_{|P|} & t_1 & ... & t_{|T|} \end{bmatrix}^T$ Particularly $\vec{n_i}$ is the vector for the path $\omega = n_i$.

A *JAF-path* from an arbitrary place p_j to a measured place p_i can be constructed in the following way:

Algorithm 5.1.12 *JAF-path from a given place p_j to a measured place $p_i \in P_M$.*

Function: *$\Omega = JAF(p_j, P_M(S))$.*

Inputs: *Initial place p_j. The measured set places $P_M(S)$ from a given output matrix S.*

Outputs: *If there exists, a set Ω of the shortest JAF-paths from p_j to a measured place $p_i \in P_M$. Otherwise $\Omega = \emptyset$.*

Initialize: *Set $Q = \Omega = \{p_j\}$ and $\Omega_a = \emptyset$. - Q is a search set, Ω is a paths set and Ω_a is an auxiliary paths set.*

Compute: *Compute the auxiliary set $Aux = Q\bullet$.*

WHILE $Aux \neq \emptyset$

1. *$\forall n_i \in Aux$ if $|\bullet n_i| > 1$, then remove n_i from Aux - removes every join transition and attribution place from the set Aux.*

2. *$\forall n_i \in Aux$ if $n_i\bullet = \emptyset$ and $n_i \in T$, then remove n_i from Aux - removes every ending transition from the set Aux.*

3. *Make $Q = Aux$.*

4. *If $|Aux \cap P_M| > 1$ then*

   (a) *$\forall n_i \in Aux$ if $n_i \in P_M$ then.*

      i. *$\forall \sigma_h \in \Omega$, if $n_i \in final(\omega_h)\bullet$ then add $\omega_h n_i$ to Ω_a - adds the node n_i to each path in Ω and the resulting path is added to Ω_a.*

      ii. *Make $\Omega = \Omega_a$ and EXIT While.*

5. *Else $\forall n_i \in Aux$*

   (a) *$\forall \omega_h \in \Omega$, if $n_i \in final(\omega_h)\bullet$ and $\vec{\omega_h}^T \vec{n_i} = 0$ then add $\omega_h n_i$ to Ω_a - adds the node n_i to each path in Ω for which n_i is an output node to the ending node of ω and n_i is not already in the path. The resulting path is added to Ω_a.*

```
    (b) Make $\Omega = \Omega_a$ and $\Omega_a = \emptyset$.
  6. Make $Aux = Q\bullet$.
END WHILE
$\forall \omega_i \in \Omega$ If $final(\omega_i) \neq P_M$ then remove $\omega_i$ from $\Omega$ - Removes every path which does not end in a measured place.
Return $\Omega$.
```

The previous algorithm starts in a given input place p_j. Then, it creates a set Ω of paths starting on p_j and containing only single input transitions and single input places, i.e. *JAF-paths*. If a path is a loop, i.e. a node n_i is visited more than once, such path is not further analyzed. When one of the paths reaches a measured place, then it removes from the set Ω all the paths in which $final(\omega)$ is not a measured place and returns Ω.

Now, in order to reduce the number of sensors in a *ContPN* guaranteeing the observability in each of the *LS* of the *ContPN* the idea is the following: If there exists a $JAF-path$ from a measured place p_i to a different measured place p_j, then the value of $\mathbf{m}(p_i)$ can be computed in infinitesimal time from the knowledge of $\mathbf{m}(p_j)$; therefore it is not necessary to measure p_i and the sensor from p_i can be removed.

Definition 5.1.13 *A sensor in a place p_i is named **redundant** if there exists a JAF-path from p_i to another measured place p_j.*

Algorithm 5.1.14 *Sensor reduction.*

```
Function: $S = SensorReduction(\langle N, \lambda, \mathbf{m}_0\rangle, S_1)$
Inputs: A ContPN $\langle N, \lambda, \mathbf{m}_0\rangle$. $S_1$, an output matrix which guarantees observability by excess.
Outputs: An output matrix $S$ such that the ContPN $\langle N, \lambda, \mathbf{m}_0\rangle$ is observable and with $|P_M(S)| \leq |P_M(S_1)|$.
Initialize $S = S_1$.
  1. $\forall p_j \in P_M(S)$ compute $\Omega = JAF(p_j, P_M(S) - \{p_j\})$.
     If $\Omega \neq \emptyset$ then remove the sensor from place $p_j$, i.e. remove the row $S(i,\bullet) = \mathbf{e}_j$ from $S$.
  2. Return $S$.
```

The previous algorithm computes an output matrix which still guarantees observability of the *ContPN* but removes redundant sensors.

Example 5.1.15 *From the Example 5.1.11, the algorithm 5.1.14 will be applied. The output matrix such that the ContPN in Fig. 5.1.1 is observable, is given by*

$$S_1 = \begin{bmatrix} \mathbf{e}_7 & \mathbf{e}_6 & \mathbf{e}_8 & \mathbf{e}_9 & \mathbf{e}_3 & \mathbf{e}_4 & \mathbf{e}_2 \end{bmatrix}^T$$

Then, $P_M(S_1) = \{p_7, p_6, p_8, p_9, p_3, p_4, p_2\}$.
$\Omega = JAF(p_7, P_M(S_1) - \{p_7\}) = \emptyset$.
$\Omega = JAF(p_6, P_M(S_1) - \{p_6\}) = \emptyset$.
$\Omega = JAF(p_8, P_M(S_1) - \{p_8\}) = \emptyset$.
$\Omega = JAF(p_9, P_M(S_1) - \{p_9\}) = \emptyset$.
$\Omega = JAF(p_3, P_M(S_1) - \{p_3\}) = \emptyset$.
$\Omega = JAF(p_4, P_M(S_1) - \{p_4\}) = \emptyset$.
$\Omega = JAF(p_2, P_M(S_1) - \{p_2\}) = p_2, t_2, p_3$.
Therefore the sensor in place p_2 *is redundant and it can be removed.*

$$S = SensorReduction(\langle N, \lambda, \mathbf{m}_0 \rangle, S_1) = \begin{bmatrix} \mathbf{e}_7 & \mathbf{e}_6 & \mathbf{e}_8 & \mathbf{e}_9 & \mathbf{e}_3 & \mathbf{e}_4 \end{bmatrix}^T$$

5.2 Distinguishability

As mentioned in previous chapters, the observability in *ContPN* requires the observability of each *LS* and the distinguishability of every pair of *LS*. In this work it is assumed that each *LS* of the *SLS* representation of the *ContPN* is already observable, for instance, by choosing output matrix S as proposed in Section 5.1.

The naive approach of verifying the distinguishability property for each pair of *LS* using Eq. (2.14) also becomes prohibited, since the number of *LS* increases exponentially with the join transitions. Even more, testing the distinguishability for every pair of *LS* using the methods proposed by [48] and [49] may lead to a wrong conclusion, as shown in the next section.

5.2.1 Classical testing for Distinguishability

The following result provides sufficient conditions for indistinguishability in *SLS*, i.e. there exists a pair of *LS* which are not distinguishable from each other. It will be proved that if two given *LS* have the same output matrix and there exists a common eigenvector with the same eigenvalue for the two *LS*, then those *LS* will not be distinguishable from each other. Notice that this result is relevant since a *ContPN* fulfils the condition of having the same output matrix for all *LS*.

Theorem 5.2.1 *Let Σ_i and Σ_j be two LS of the same dimension and with the same output matrix, $S_i = S_j$. Let $v \in \mathbb{R}^{|P|}$ be a vector such that $A_i v = A_j v = \beta v$. Then the LS Σ_i and Σ_j are not distinguishable from each other.*

Proof: The extended *LS* for system Σ_i and Σ_j is

$$A_{ij} = \begin{bmatrix} A_i & 0 \\ 0 & A_j \end{bmatrix}$$

Now, let $v_{ij} = [v^T \; v^T]^T$. Then v_{ij} is an eigenvector of A_{ij} since $A_{ij}v_{ij} = [(A_i v)^T \; (A_j v)^T]^T = [(\beta v)^T \; (\beta v)^T]^T$ Now, since $S_i = S_j$, then $S_{ij} = [S_i \; - S_i]$ and then $S_{ij}v_{ij} = 0$, therefore $v_{ij} \in \ker(S_{ij})$. Then, there exists an A_{ij}-invariant subspace contained in $\ker(S_{ij})$ and A_{ij} is not observable. Therefore Σ_i and Σ_j are not distinguishable from each other. ■

It can be seen from the previous theorem that if there exists a common eigenvector with the same eigenvalue for two *LS*, then this pair of *LS* are not distinguishable from each other and therefore the *SLS* is not observable.

Particularly for FC-*ContPN* when dealing with join transitions, it has been proved that in FC-*ContPN* each input place p_j to a join transition t_i has associated an elementary eigenvector $\mathbf{e}_j$ for some matrix $\mathbf{C}\Lambda\Pi_k$ (see Proposition 4.1.3).

When all the input places to join transitions are measured, the *ContPN* becomes trivially distinguishable. Indeed, using this information, the evolving configuration (hence the actual evolving *LS*) is known. However, using the classical distinguishability tests, the indistinguishability of the *ContPN* is obtained, which is a failure of this test. This problem is shown in the next proposition.

Proposition 5.2.2 *Let a FC-ContPN having all the input places to its join transition measured and every LS of the ContPN be observable. Let T_J be the set of join transitions of a FC-ContPN. If $| \bullet t_i| \geq 3$ for some $t_i \in T_J$ or $|T_J| > 1$, then there exists two indistinguishable LS.*

Proof:

Case $| \bullet t_i| \geq 3$ for some $t_i \in T_J$.

Let $\bullet t_i = \{p_1, p_2, p_3\}$ and Π_h be the configuration matrix in which place $p_h \in \bullet t_i$ constrains the transition's flow, $h = 1, 2, 3$. Then $\mathbf{e}_1 \in \ker(\mathbf{C}\Lambda\Pi_2) \cap \ker(\mathbf{C}\Lambda\Pi_3)$, since p_1 does not constrain the flow of transition t_i when configuration Π_2 or Π_3 are active. Therefore $\exists \mathbf{e}_1$ which fulfills conditions of Theorem 5.2.1 and $\mathbf{C}\Lambda\Pi_2$ and $\mathbf{C}\Lambda\Pi_3$ systems are indistinguishable from each other. It occurs similarly with place p_2 and p_3 and its associated vectors $\mathbf{e}_2$ and $\mathbf{e}_3$.

Case $|T_J| > 1$.

Let $T_J = \{t_1, t_2\}$. If t_i is a join transition, then t_i has at least two input places. If one of them has more than two input places, then the previous case applies. Let $\bullet t_1 = \{p_1, p_2\}$ and $\bullet t_2 = \{p_3, p_4\}$. Let Π_{gh} be the configuration matrix where place p_g constrains the flow for transition t_1 and p_h constrains the flow for transition t_2. Then $\mathbf{e}_1 \in \ker(\mathbf{C}\Lambda\Pi_{23}) \cap \ker(\mathbf{C}\Lambda\Pi_{24})$. Therefore $\exists \mathbf{e}_1$ which fulfils the conditions of Theorem 5.2.1 and therefore $\mathbf{C}\Lambda\Pi_{23}$ and $\mathbf{C}\Lambda\Pi_{24}$ systems are indistinguishable from each other. It occurs similarly with the other combinations for the places constraining transitions t_1 and t_2. ■

There are other cases when the classical tests for distinguishability in *SLS* fail to determine if a *ContPN* is distinguishable. To cope with this problem, some A_k−invariant subspaces for the non FC-*ContPN* will be characterized.

Proposition 4.1.3 shows that when the output transitions of a place p_j are join transitions, then there exists an invariant subspace generated $\mathbf{e}_j$ of configuration matrices where place p_j is not constraining a transition's flow.

The next proposition shows that if a place p_j is input place to only join transitions and ending transitions, then the associated elementary vector $\mathbf{e}_j$ generates an A_k−invariant subspace.

Proposition 5.2.3 *Let a ContPN having a set T_J of join transitions and a set T_E of ending transitions. Let p_j be a place such that $p_j\bullet \subseteq T_J \cup T_E$. Then $\exists \Pi_k$ such that $\mathbf{C}\Lambda\Pi_k\mathbf{e}_j = \beta\mathbf{e}_j$, i.e. $\mathbf{e}_j$ generates an A_k−invariant subspace.*

Proof: Similarly to Proposition 4.1.3, let $\forall t_i \in p_j \bullet \cap T_J$ be join transitions with some place $p_i \in \bullet t_i$, $p_i \neq p_j$ constraining its flow. Then p_j does not constrain any flow for its output join transitions. However, p_j does constrain the flow for all the ending transitions $t_h \in p_j\bullet\cap T_E$. Then, as in Proposition 16 in [72] (Proposition 4.1.20 in this work), $\mathbf{C}\Lambda\Pi_k\mathbf{e}_j = \beta\mathbf{e}_j$. ■

The next corollary summarizes the two previous propositions.

Corollary 5.2.4 *Let p_j be a place such that $p_j\bullet \subseteq T_J \cup T_E$. Then $\exists \Pi_k$ such that $\mathbf{C}\Lambda\Pi_k\mathbf{e}_j = \beta\mathbf{e}_j$, i.e. $\mathbf{e}_j$ generates an A_k−invariant subspace.*

Proof: The result is a consequence of Propositions 4.1.3 and 5.2.3. ■

The previous corollary shows that if a place p_j is input place to only join transitions and ending transitions, then $\mathbf{C}\Lambda\Pi_k\mathbf{e}_j = \beta\mathbf{e}_j$ for some k and some β. These A_k−invariant subspaces will be used to determine if there exist two indistinguishable *LS* in a *ContPN*.

Definition 5.2.5 *Let P_K be the places fulfilling the conditions of Corollary 5.2.4. Each place $p_j \in P_K$ will be named a* ker *−place.*

Definition 5.2.6 *Let $t_i \in T_J$. The* ker *−places index for transition t_i, denoted by $\iota(t_i)$,*

$$\iota(t_i) = |P_K \cap \bullet t_i|$$

i.e. is the cardinality of the set $P_K \cap \bullet t_i$, the number of input ker *−places to transition t_i.*

The next propositions show that even if a *ContPN* is distinguishable, the classical test shows that there exist two indistinguishable *LS*.

Proposition 5.2.7 *Let a ContPN having every input place to join transitions measured. Let $T_J \neq \emptyset$ be the set of join transitions of the ContPN. If any of the following conditions is fulfilled, then there exist two indistinguishable LS*

1. $\exists t_a, t_b \in T_J$ *with* $\iota(t_a) > 1$ *and* $\iota(t_b) > 1$.
2. $\exists t_i \in T_J$ *with* $\iota(t_i) \geq 3$.

Proof:

1) Let t_a and t_b be two join transitions with $\iota(t_a) = 2$ and $\iota(t_b) = 2$. Also, let $p_1, p_2 \in P_K \cap \bullet t_a$ and $p_3, p_4 \in P_K \cap \bullet t_b$. Let Π_{gh} be the configuration matrix where place p_g constrains transition t_a and place p_h constrains transition t_b. Then $\mathbf{e}_1 \in \ker(\mathbf{C}\Lambda\Pi_{23}) \cap \ker(\mathbf{C}\Lambda\Pi_{24})$ and the *LS* with dynamical matrices $\mathbf{C}\Lambda\Pi_{23}$ and $\mathbf{C}\Lambda\Pi_{24}$ are not distinguishable from each other.

2) Let $\bullet t_i \cap T_K = \{p_1, p_2, p_3\}$. Now, let Π_j stand for the configuration matrix where p_j constrains the flow of transition t_i. Then, $\mathbf{e}_1 \in \ker(\mathbf{C}\Lambda\Pi_2) \cap \ker(\mathbf{C}\Lambda\Pi_3)$ and the *LS* with dynamical matrices $\mathbf{C}\Lambda\Pi_2$ and $\mathbf{C}\Lambda\Pi_3$ are not distinguishable from each other. ■

For the case of ending transitions, let t_i be an ending transition and places $p_j \in \bullet t_i$ are its input places. It will be assumed that if $\forall t_i \in p_j\bullet$ such that $t_i\bullet = \emptyset$, those ending transitions are represented as only one ending transition. Proposition 4.1.20 characterize the ending transitions eigenvectors.

With this proposition it can be seen that if $|T_J| \neq \emptyset$ and $\exists t_i$, $t_i\bullet = \emptyset$ with $(\bullet t_i)\bullet = \{t_i\}$, i.e. its input places only have t_i as output transition, then $\forall \Sigma_k$, $A_k\mathbf{e}_j = \beta\mathbf{e}_j$ and every pair of *LS* are indistinguishable from each other.

Similarly to ending transitions, ending places also have A_k-invariant subspaces associated, as shown in Proposition 4.1.2.

Again, if a *ContPN* has an ending place and $|T_J| \neq \emptyset$, then every pair of *LS* are indistinguishable from each other.

In the proposition 4.1.18, the attribution eigenvector is an eigenvector for all matrices $\mathbf{C}\Lambda\Pi_k$, therefore if such vector exists in a *ContPN* and $|T_J| \neq \emptyset$, then every pair of *LS* are indistinguishable from each other.

Now, as discussed in [72], there exist annulers of $\mathbf{C}\Lambda\Pi_k$. For that, it is required that $\exists v \in Im(\Lambda\Pi_k) \cap \ker(\mathbf{C})$. Such vector v is contained in $Im(\Lambda\Pi_k)$ when the next definition is fulfilled.

Definition 5.2.8 *A T-Semiflow agreeing with λ is a vector $x \in \ker(\mathbf{C})$ such that for every decision place p_j with $p_j\bullet = \{t_1, t_2, ..., t_n\}$, either every transition $t_1, ..., t_n$ belongs to a different minimum T-Semiflow or for those transitions in the same minimum T-Semiflow, the ratio $x_i/x_h = \lambda_i/\lambda_h$ holds $\forall t_i, t_h \in p_j\bullet$.*

Proposition 4.1.7 characterizes the conditions for which a T-Semiflow becomes an annuler for $\mathbf{C}\Lambda\Pi_k$.

The next proposition shows that if a T-Semiflow does not contain a join transition, then the associated annuler vector v is a common eigenvector to every *LS*.

Proposition 5.2.9 *Let x be a T-Semiflow in which $\langle x\rangle \cap T_J = \emptyset$, i.e. there is not any join transition in the T-Semiflow. Let v be the annuler of $\mathbf{C}\Lambda\Pi_k$ associated to x. Then $\forall k,\ v \in \ker(\mathbf{C}\Lambda\Pi_k)$.*

Proof: It is clear that if x does not contain any join transition, then each transition's flow is constrained by the same place in every configuration Π_k. Then, the $i-th$ column in each matrix $A_k(\bullet, i)$ is the same. Therefore $\forall k,\ v \in \ker(\mathbf{C}\Lambda\Pi_k)$. ■

The previous proposition shows that if $T_J \neq \emptyset$ and if there exists a T-Semiflow which does not contain any join transition, then every pair of *LS* are not distinguishable from each other.

5.2.2 Sensor placement for distinguishability

Now, this subsection addresses the problem of finding out the places in a *ContPN* that must be measured for the *ContPN* to exhibit the distinguishability property. In this subsection we assume that there exists a set of measured places such that every *LS* is observable. This set of measured places can be constructed with the algorithms of the previous section. Thus the problem consists in adding (removing) sensors to (from) places to guarantee distinguishability.

As previously discussed, it is clear that if every input place to join transitions are measured, then the *ContPN* becomes trivially distinguishable. However, it may not be necessary to measure every input place to join transitions for distinguishability (nets where the *LS* are observable but the are indistinguishable from each other is presented next [44]).

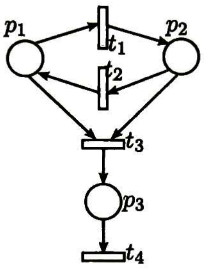


Figure 5.2.1: An indistinguishable *ContPN* system.

Now, in [46] it was proved that if there exists a *JAF-path* from a place p_j to a measured place p_i, then the marking of place p_j can be computed in infinitesimal time. Thus the problem deals with verifying if every input place p_j to a join transition t_i is either measured or if there exists a *JAF-path* from p_j to a measured place. If none of these conditions occur, then a sensor must be added to p_j. If both conditions hold, then the sensor of p_j should be removed. Otherwise no sensor is added or removed.

Since adding/removing sensors as explained before leads to *ContPN* where the marking of all input places to join is known or can be computed in infinitesimal time, then the net becomes distinguishable. The adding/removing sensors procedure is implemented with the following two algorithms.

Let S be the output matrix of the *ContPN* and $P_M = \{p_j | e_j^T \text{is a row of matrix } S\}$ be the set of measured places.

Based on Algorithm 5.1.12, the next algorithm is introduced to determine weather or not to put a sensor in a place $p_j \in \bullet t_i$, where t_i is a join transition.

Algorithm 5.2.10 *Sensor placement for distinguishability.*

Inputs: *$\bullet T_J$, the set of input places to join transitions. P_M, the set of measured places.*
Outputs: *The output matrix S for which the ContPN is distinguishable.*
Initialize: *$S_1 = S$. $P_{M1} = P_M$. – Initializes matrix S_1 as the output matrix S and the set of measured places $P_{M1} = P_M$.*

1. $\forall p_j \in \bullet T_J$

   (a) Compute $\Omega \doteq JAF(p_j, P_{M1})$.

   If $\Omega = \emptyset$ *and* p_j *is not measured, then add a sensor to place* p_j*, i.e.,* $S_1 = \begin{bmatrix} S_1 \\ \mathbf{e}_j^T \end{bmatrix}$ *and add* p_j *to the set* P_{M1}.

   Else If $\Omega \neq \emptyset$ *and* p_j *is measured, then remove sensor from place* p_j*, i.e. remove the row* $\mathbf{e}_j \in \{0,1\}^{|T|}$ *from matrix* S_1.

2. Return S_1.

The previous algorithm adds a sensor on a place $p_j \in \bullet t_i$, where t_i is a join transition, when p_j is not measured and there does not exist a $JAF-$path from p_j to a measured place. On the other hand, if place $p_j \in \bullet t_i$ is measured but there exists $JAF-$path to another measured place p_i, then p_j can be computed in infinitesimal time; therefore this sensor in p_j is not necessary, and it can be removed. Since in infinitesimal time it is possible to determine all the marking $\mathbf{m}(p_j)$, $\forall p_j \in \bullet T_J$, i.e. the marking of all the input places to join transitions, then the actual *LS* evolving is known.

Algorithm 5.2.10 guarantees that the *ContPN* is distinguishable.

6

Observer design

In this chapter, the observers design problem is addressed. Once a *ContPN* shows the observability property, then a mathematical entity called observer may be designed. This observer will allow the computation of the state of the *ContPN*. In Subsection 2.3.2 it was presented the strategy for observers design in *LS*. The same idea, based on a Luenberger observer, will be applied to *ContPN*. For the observer design in *ContPN*s, a different state equation representation than the one in Eq. (2.6) will be introduced so that a single observer for the *ContPN* can be designed. Finally, a general *ContPN*-observer structure will be presented. If the extended transitions of such *ContPN*-observer structure are adequately controlled, then the marking (state) of the *ContPN*-observer will converge to the *ContPN* marking.

6.1 State space representation for *ContPN*

In this section the dynamics of the join transitions will be separated from the dynamics of those transitions with a single input place. Then, it will be shown that with this representation, a single observer may be designed.

First, consider the sets of single input and join transition, as defined in Definition 2.1.8. The set of single input transitions can also be represented with the vector

$$\mathbf{t}_S \in \{0,1\}^{|T|}$$

such that $\langle \mathbf{t}_S \rangle = T_S$.

Similarly, the set of join transitions can be represented by the vector

$$\mathbf{t}_J \in \{0,1\}^{|T|}$$

such that $\langle \mathbf{t}_J \rangle = T_J$.

Clearly $T_S \cap T_J = \emptyset$ and $T_S \cup T_J = T$. Using vectors $\mathbf{t}_S$ and $\mathbf{t}_J$ matrices $\mathbf{C}$ and Λ will be now rewritten.

Definition 6.1.1 *The $|P| \times |T|$ Join (Single Input) Incidence Matrix is*

$$\mathbf{C}_J = \mathbf{C} \cdot diag(\mathbf{t}_J) \quad (\mathbf{C}_S = \mathbf{C} \cdot diag(\mathbf{t}_S))$$

which is a $|P| \times |T|$ matrix where its j-th column corresponds to the j-th column of $\mathbf{C}$ if $\mathbf{t}_J(j) = 1$ $(\mathbf{t}_S(j) = 1)$ else it is a zero column.

It is also clear that $\mathbf{C}_S + \mathbf{C}_J = \mathbf{C}$. Similarly to Definition 6.1.1, matrices Λ_S and Λ_J are $\Lambda_S = \Lambda \cdot diag(\mathbf{t}_S)$ and $\Lambda_J = \Lambda \cdot diag(\mathbf{t}_J)$

For the configuration matrices, if a transition $t_i \in T_S$ then there exists only one place that can constrain its flow. Therefore there exists a unique matrix Π_S for which its $i-$th row (associated to transition t_i) is an elementary transposed vector $\mathbf{e}_h^T$, associated to the place p_h that constrains its flow. Otherwise the $i-$th row of Π_S is a zero row when $t_i \in T_J$. Then, it is possible to rewrite every matrix $\Pi_k = \Pi_S + \Pi_{J_k}$ where Π_{J_k} contains every switching representation for the $enab(t_i, p_j)$ function.

Using the previously defined matrices, it is possible to rewrite (2.4) as follows:

$$\dot{\mathbf{m}} = (\mathbf{C}_S + \mathbf{C}_J) \cdot f \tag{6.1}$$

where the flow vector is $f = f_S + f_J$, f_S and f_J are the flow vectors corresponding to single input and join transitions respectively. Now, it is possible to represent the flow of the single input transitions as:

$$f_S = \Lambda_S \cdot \Pi_S \mathbf{m} \tag{6.2}$$

and substituting (6.2) in (6.1) we obtain:

$$\overset{\bullet}{\mathbf{m}} = (\mathbf{C}_S + \mathbf{C}_J) \cdot (\Lambda_S \cdot \Pi_S \cdot \mathbf{m} + f_J)\,. \tag{6.3}$$

The product $\mathbf{C}_S f_J = 0$ since every column corresponding to a join transition in $\mathbf{C}_S$ is a zero column and for those nonzero columns, the corresponding entry in f_J is zero. Similarly $\mathbf{C}_J f_S = 0$. Therefore, the *ContPN* dynamics can be represented with:

$$\overset{\bullet}{\mathbf{m}} = \mathbf{C}_S \Lambda_S \Pi_S \mathbf{m} + \mathbf{C}_J f_J \tag{6.4}$$

where $\mathbf{C}_S \Lambda_S \Pi_S \mathbf{m}$ will be named single input dynamics and $\mathbf{C}_J f_J$ will be named join dynamics.

6.2 Luenberger observer for *ContPN*

Using (6.4) and the observer presented in Subsection 2.3.2, it can be introduced a *ContPN* observer given by:

$$\begin{aligned} \overset{\bullet}{\bar{\mathbf{m}}} &= \mathbf{C}_S \Lambda_S \Pi_S \bar{\mathbf{m}} + \mathbf{C}_J \bar{f}_J + L(y - \bar{y}) \\ \bar{y} &= S \bar{\mathbf{m}} \end{aligned} \tag{6.5}$$

where $\mathbf{C}_J \bar{f}_J$ are the Join transitions' dynamics of the observer.

Now, consider the case of *FC-ContPN*s. In this case, vector f_J is known since in a *FC-ContPN* every input place to join transitions must be measured and it can always be made $\bar{f}_J = f_J$. Proposing an error system with $\tilde{\mathbf{m}} = \mathbf{m} - \bar{\mathbf{m}}$, the dynamics of the error system is given by $\overset{\bullet}{\tilde{\mathbf{m}}} = \overset{\bullet}{\mathbf{m}} - \overset{\bullet}{\bar{\mathbf{m}}}$ from which the following equation is obtained:

$$\overset{\bullet}{\tilde{\mathbf{m}}} = \mathbf{C}_S \Lambda_S \Pi_S \tilde{\mathbf{m}} - L(y - \bar{y}). \tag{6.6}$$

However, for the *ContPN* $y = Sm$ and $\bar{y} = S\bar{m}$ and Equation (6.6) becomes

$$\overset{\bullet}{\tilde{\mathbf{m}}} = (\mathbf{C}_S \Lambda_S \Pi_S - LS)\, \tilde{\mathbf{m}} \tag{6.7}$$

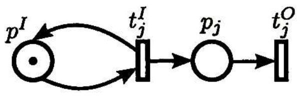


Figure 6.3.1: Observer structure for a place.

which is a standard Luenberger observer design as in Subsection 2.3.2 [39] and if L is designed properly, the error system is asymptotically stable and the state vector $\bar{\mathbf{m}} \to \mathbf{m}$ at any desired rate.

The main advantage of this observer is that with a single structure it is possible to observe a *ContPN* no matter the number of *LS* in the family $\mathcal{F}$ representing it. The only problem is that it is not guaranteed that the estimated marking $\bar{\mathbf{m}}$ remains non-negative during the transient state. In such cases when this observer is used to obtain a control action, the negative markings should be considered zero.

The initial marking for this observer can be any marking, but it can be used the following:

$$\bar{\mathbf{m}}_0(p_j) = \begin{cases} \mathbf{m}_0(p_j) & \text{if } p_j \text{ is a measured place} \\ 0 & \text{if } p_j \text{ is not a measured place} \end{cases}$$

6.3 A *ContPN* general observer

The observer for a *ContPN* $\langle N, \mathbf{m}_0 \rangle$ can also be implemented as a *ContPN* with a structure similar to N extended with one place and $2|P|$ transitions.

For each place $p_j \in P$ in a *ContPN*, its observer will have two additional transitions: An input transition t_j^I and an output transition t_j^O with arc weights equal to 1. In order to guarantee that every transition in the *ContPN* observer (*ContPN-O*) is also well defined, one additional place p^I connected as input and output to each and every transition t_j^I is included, as shown in Fig. 6.3.1. In this way, the *ContPN*-Observer graphical representation (*ContPN-O*) is the *ContPN* $\langle N^O, \mathbf{m}_0^O \rangle$ where $N^O = \langle P^O, T^O, \mathbf{Pre^O}, \mathbf{Post^O} \rangle$ and $\mathbf{m}_0^O$ is the initial marking of the *ContPN-O*. The set $P^O = P \cup \{p^I\}$. The set $T^O = T \cup T_{in} \cup T_{out}$ where $T_{in} = \{t_j^I \forall p_j \in P\}$ and $T_{out} = \{t_j^O \forall p_j \in P\}$. In order to modify the *ContPN-O* dynamics each transition $t \in T_{in} \cup T_{out}$ will be Controllable [73], meaning that it is possible to slow down its flow. Let now the transition's index assignation be accordingly to the following vector: $\mathbf{t^O} = [\mathbf{t}^T \;\; \mathbf{t}_{in}^T \;\; \mathbf{t}_{out}^T]^T$ where $\langle \mathbf{t} \rangle = T$, $\langle \mathbf{t}_{in} \rangle = T_{in}$ and $\langle \mathbf{t}_{out} \rangle = T_{out}$. Matrix $\mathbf{Post^O}$ is defined as:

$$\mathbf{Post}^O(p_j,t_i)=\begin{cases} 1 & \text{if } p_j = p^I \text{ and } t_i \in T_{in} \\ 1 & \text{if } t_i = t_j^I \in T_{in} \\ & \text{and } p_j \in P \\ \mathbf{Post}(p_j,t_i) & \text{if } t_i \in T \text{ and } p_j \in P \\ 0 & \text{otherwise} \end{cases}$$

Similarly, matrix $\mathbf{Pre^O}$ is defined as:

$$\mathbf{Pre}^O(p_j,t_i)=\begin{cases} 1 & \text{if } p_j = p^I \text{ and } t_i \in T_{in} \\ 1 & \text{if } t_i = t_j^O \in T_{out} \\ & \text{and } p_j \in P \\ \mathbf{Pre}(p_j,t_i) & \text{if } t_i \in T \text{ and } p_j \in P \\ 0 & \text{otherwise} \end{cases}$$

The Observer Incidence matrix is $\mathbf{C}^O = Post^O - Pre^O$. This matrix $\mathbf{C}^O$ has always the form:

$$\mathbf{C}^O = \begin{bmatrix} \mathbf{C} & \mathbf{I}_{|P|} & -\mathbf{I}_{|P|} \\ & \mathbf{0}^T & \end{bmatrix}$$

where $\mathbf{I}_{|P|}$ is the identity matrix of $dim|P|$ and $\mathbf{0}$ is a zero vector of $dim(|T|+2|P|)$. Since $\forall t_j \in (T_{in} \cup T_{out}), |{\bullet}t_j| = 1$ then the number of configurations in the *ContPN-O* is the same than in the *ContPN*. Therefore, the configuration matrices for the *ContPN* are:

$$\Pi_k^O = \begin{bmatrix} \Pi_k & \mathbf{0} \\ \mathbf{0} & \mathbf{1} \\ I_{|P|} & \mathbf{0} \end{bmatrix}$$

The speed vector is $\lambda^{\mathbf{O}} = \begin{bmatrix} \lambda^T & \lambda_{in}^T & \lambda_{out}^T \end{bmatrix}$, where λ_{in} and λ_{out} are the speed vectors associated to transitions in T_{in} and T_{out} respectively. Finally, the initial marking vector is given by:

$$\mathbf{m}^O(p_i) = \begin{cases} m(p_i) & \text{if } p_i \text{ is a measured place} \\ 0 & \text{if } p_i \text{ is not a measured place} \\ 1 & \text{if } p_i = p^I. \end{cases} \tag{6.8}$$

The state equation for the *ContPN-O* is given by:

$$\overset{\bullet}{\mathbf{m}}^O = \mathbf{C}^O \Lambda^O \Pi_k^O \mathbf{m}^O$$

Using $\mathbf{m}^O = \begin{bmatrix} \bar{\mathbf{m}}^T & \mathbf{m}^I \end{bmatrix}^T$ and substituting the previous matrix definitions, it can be obtained:

$$\begin{bmatrix} \overset{\bullet}{\bar{\mathbf{m}}} \\ \overset{\bullet}{\mathbf{m}}^I \end{bmatrix} = \begin{bmatrix} \mathbf{C}\Lambda\Pi_k \cdot \bar{\mathbf{m}} \\ 0 \end{bmatrix} + \begin{bmatrix} \Lambda_{in}\mathbf{1} \\ 0 \end{bmatrix} - \begin{bmatrix} \Lambda_{out} \cdot \bar{\mathbf{m}} \\ 0 \end{bmatrix} \tag{6.9}$$

Since the dynamics for place p^I is equal to zero, its marking is always $m^I = 1$ and it is possible to eliminate the last row of (6.9). Now, using the Single Input and Join Dynamics representation given in (6.4), the dynamics for the *ContPN-O* can also be represented as follows:

$$\dot{\bar{\mathbf{m}}} = \mathbf{C}_S\Lambda_S\Pi_S \cdot \bar{\mathbf{m}} + \mathbf{C}_J\bar{f}_J + \Lambda_{in}\mathbf{1} - \Lambda_{out} \cdot \bar{\mathbf{m}}.$$

From the previous equation, it can be seen that $\mathbf{C}_J\bar{f}_J$ should be dependent on the state vector $\bar{\mathbf{m}}$. However, in *FC-ContPN*, f_J is known and it can be considered as an external input for the *ContPN-O*.

Let a transition $t_i \in (T_{in} \cup T_{out})$ be named an extended transition. Every extended transition is controllable since they are designed to that end, therefore it is possible to include a u_{in} and a u_{out} control vectors to slow down the flow of those extended transitions. Then, the flow for the extended transitions can be written as:

$$f_e = (\Lambda_{in}\mathbf{1} - u_{in}) - (\Lambda_{out}\bar{\mathbf{m}} - u_{out}) \tag{6.10}$$

where $0 \leq u_{in} \leq \Lambda_{in}\mathbf{1}$ and $0 \leq u_{out} \leq \Lambda_{out}\bar{\mathbf{m}}$. It is possible to represent u_{in} and u_{out} as a proportion of the flow $\Lambda_{in}\mathbf{1}$ and $\Lambda_{out}\bar{\mathbf{m}}$ respectively:

$$\begin{array}{l} u_{in} = I^u_{in}\Lambda_{in}\mathbf{1} \\ u_{out} = I^u_{out}\Lambda_{out}\bar{\mathbf{m}} \end{array} \tag{6.11}$$

where I^u_{in} and I^u_{out} are diagonal matrices with elements $I^u_{in_j}, I^u_{out_j} \in [0,1]$. Substituting (6.11) in (6.10) it is obtained:

$$\begin{aligned} f_e &= (\Lambda_{in}\mathbf{1} - I^u_{in}\Lambda_{in}\mathbf{1}) - (\Lambda_{out}\bar{\mathbf{m}} - I^u_{out}\Lambda_{out}\bar{\mathbf{m}}) \\ &= (I^c_{in}\Lambda_{in}\mathbf{1}) - (I^c_{out}\Lambda_{out}\bar{\mathbf{m}}) \end{aligned}$$

where $I^c_{in} = \mathbf{I}_{|P|} - I^u_{in}$ and $I^c_{out} = \mathbf{I}_{|P|} - I^u_{out}$.

It is clear that the speed vectors λ_{in} and λ_{out} are design parameters for the *ContPN-O*. Proposing λ_{in} and λ_{out} values sufficiently big and with matrices I^c_{in}, I^c_{out} it is possible to add or remove marks at any place and with an adequate control strategy for f_e, marking $\bar{\mathbf{m}} \to \mathbf{m}$.

Let the observer's error be defined as $\mathbf{e} = \mathbf{m} - \bar{\mathbf{m}}$. The error system therefore is $\dot{\mathbf{e}} = \dot{\mathbf{m}} - \dot{\bar{\mathbf{m}}}$, where

$$\dot{\mathbf{e}} = \mathbf{C}_S\Lambda_S\Pi_S(\mathbf{m} - \bar{\mathbf{m}}) - (I^c_{in}\Lambda_{in}\mathbf{1}) + (I^c_{out}\Lambda_{out}\bar{\mathbf{m}})$$

and it is possible to make $(I^c_{in}\Lambda_{in}\mathbf{1}) = (I^c_{out}\Lambda_{out}\bar{\mathbf{m}}) + \mathbf{f}_{in}$, i.e. the input flow for each place in the observer is greater than its output flow, obtaining

$$\dot{\mathbf{e}} = \mathbf{C}_S\Lambda_S\Pi_S(\mathbf{m} - \bar{\mathbf{m}}) - \mathbf{f}_{in} \tag{6.12}$$

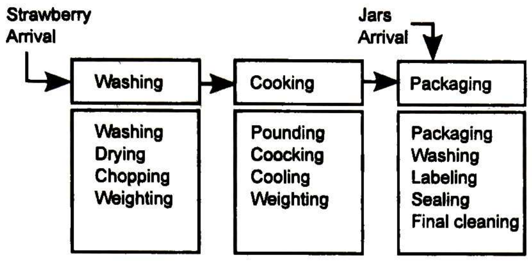


Figure 6.3.2: Illustrative example flow diagram.

where $\mathbf{f}_{in} \geq 0$.

The following control action for the extended input transitions is now introduced

$$\mathbf{f}_{in_j} = \begin{cases} \kappa_j \mathbf{e}(p_j) & \text{if } p_j \in P_m \\ 0 & \text{otherwise.} \end{cases} \tag{6.13}$$

The stability demonstration for this observer's strategy is currently being developed.

6.3.1 Illustrative example

Consider the flow diagram in Figure 6.3.2. It represents a simple process for the production of strawberry jam. The process consists of three steps: Washing, Cooking and Packaging. The Washing process consists of Washing the strawberries, drying them, chopping them to simplify the cooking and finally weighting them in standard lots. The cooking process consists on pounding the strawberries. Then the actual cooking of the strawberries take place, where the sugar is added. Afterwards, for the pasteurization process the jam is cooled. Finally, the jam is again weighted and ready to be packaged. After the packaging, the jars are washed to ensure no spills are in the jar. Then, the full clean jars are labelled and sealed. The final step before they are ready to be shipped is a final cleaning and visual inspection. The simplified *ContPN* model is also presented in Figure 6.3.3.

The firing rates for the transitions are given by

$\lambda = [\ 2\ \ 1\ \ 1\ \ 1\ \ 1\ \ 3\ \ 2\ \ 2\ \ 2\ \ 1\ \ 1\ \ 1\ \ 1\ \ 1\ \ 2\ ]^T$

This *ContPN* is observable with an output matrix

$$S = \begin{bmatrix} e_5^T & e_6^T & e_{10}^T & e_{11}^T \end{bmatrix}^T$$

The Single Input and Join transitions sets are $T_S = \{t_1, t_2, t_3, t_4, t_6, t_7, t_8, t_{10}, t_{11}, t_{12}, t_{13}\}$ and $T_J = \{t_5, t_9\}$ respectively.

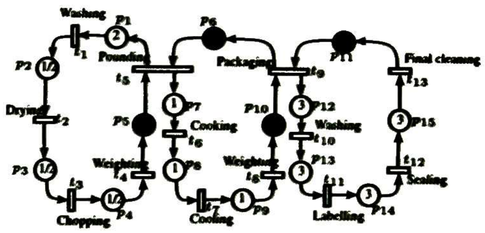


Figure 6.3.3: *ContPN* model for the strawberry jam production.

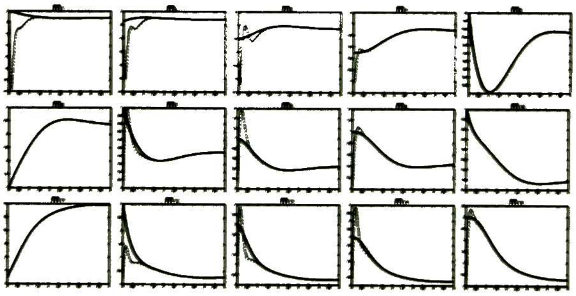


Figure 6.3.4: Luenberger observer estimates.

In the Luenberger observer, the output error feedback matrix L, can be computed to obtain any desired dynamics for the error system. Particularly for this example, L is designed in order to obtain an asymptotically stable error system with eigenvalues

$eig = \{-3, -4, -5, ..., -15, -16, -17\}$.

In Fig. 6.3.4 the marking evolution for the places of *ContPN* (continuous line) and its Luenberger observer (dashed line) are shown. It is important to notice that with the observer design previously introduced, it is not relevant to know the minimum dwell time of the *ContPN*. In fact, since the active configuration is always known, it is possible to determine the continuous state even when switching occurs in the *ContPN*.

In Figure 6.3.5 the *ContPN*-O for the *ContPN* inf Fig. 6.3.3 is shown. The results for the *ContPN*-O are shown in figure 6.3.6. For this example $\kappa_j = 675, \forall p_j \in P_m$. It can be seen that the observed marking in each place is always non-negative and reaches asymptotically the marking of the *ContPN*. Therefore, if the results are used for controlling the marking of a *ContPN* it is not necessary to filter the value. Instead, the separation principle should be provided.

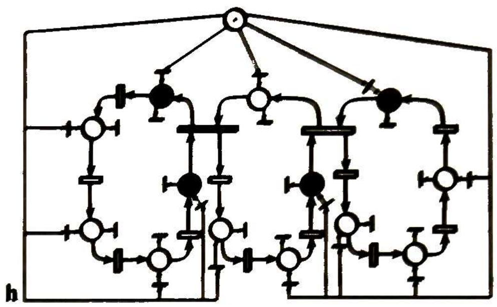


Figure 6.3.5: *ContPN*-O for the *ContPN* in Fig. 6.3.3.

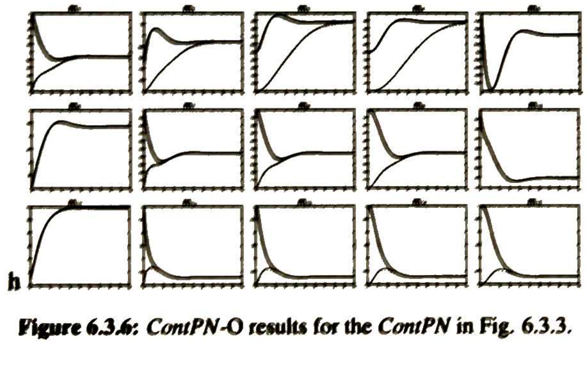


Figure 6.3.6: *ContPN*-O results for the *ContPN* in Fig. 6.3.3.

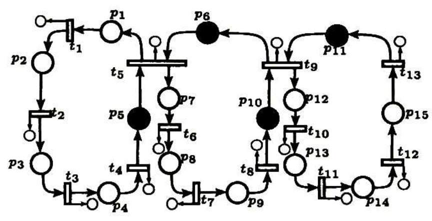


Figure 6.4.1: The *xContPN* for figure 6.3.3.

6.4 Observer and control integration

The control law introduced in (2.11) requires, for the computation of the minimum Parikh vector, the initial marking $\mathbf{m}_0$ of the *ContPN* as well as the value of the transitions flow. However, $\mathbf{m}_0$ is actually unknown, otherwise it would not be necessary to have an observer. It is only known the output marking, i.e. $S\mathbf{m}_0$. Fortunately, the observer can be designed in such a way that the steady state is reached within any desired $\tau > 0$. Then the control and observer integration strategy will be the next:

1. Design a Luenberger observer as in Eq. (6.5) to guarantee $|\mathbf{m}(\tau_1) - \bar{\mathbf{m}}(\tau_1)| = \epsilon$ where ϵ is a very close to zero value reached in designed time τ_1.

2. Design the *xContPN* for the original *ContPN*with an initial marking as in Eq (6.8). The *xContPN* represents the observer, where the marking $\mathbf{m}(p_i) = \bar{\mathbf{m}}_i \; \forall p_i \in P$. If $\exists \tau$ such that $\bar{\mathbf{m}}_i(\tau) < 0$, then $\mathbf{m}(p_i) = 0$.

3. Once $|\mathbf{m} - \bar{\mathbf{m}}| = \epsilon$, then compute the min Parikh vector σ_{min} as in Eq. (2.10).

4. Let $\mathbf{m}_a(\tau_2)$ for a time $\tau_2 > \tau_1$, be the marking in the places P_a of the *xContPN* at time τ_2. Then apply the control law introduced in Eq. (2.11) with $\sigma_r = \sigma_{min} + \mathbf{m}_a(\tau_2)$.

Notice that, since there exists an observer's error $\mathbf{m} - \bar{\mathbf{m}} = \epsilon$ at time τ_1, then the marking evolution of the *ContPN* will also have an error in the reached steady state. This error depends on the design parameter ϵ, so the smaller the convergence observer error ϵ is, the smaller the steady state error $\mathbf{m} - \mathbf{m}_r$ will be.

6.4.1 Illustrative example

The initial marking of the *ContPN* in Figure 6.3.3 is taken as $\mathbf{m}_0(p_1) = 2$, $\mathbf{m}_0(p_2) = \mathbf{m}_0(p_3) = \mathbf{m}_0(p_4) = \mathbf{m}_0(p_5) = 1/2$, $\mathbf{m}_0(p_6) = \mathbf{m}_0(p_7) = \mathbf{m}_0(p_8) = \mathbf{m}_0(p_9) = \mathbf{m}_0(p_{10}) = 1$, $\mathbf{m}_0(p_{11}) = \mathbf{m}_0(p_{12}) = \mathbf{m}_0(p_{13}) = \mathbf{m}_0(p_{14}) = \mathbf{m}_0(p_{15}) = 3$.

For the observer, its initial marking is considered as $\bar{\mathbf{m}}_0(p_5) = 1/2$, $\bar{\mathbf{m}}_0(p_6) = \mathbf{m}_0(p_{10}) = 1$, $\mathbf{m}_0(p_{11}) = 3$ and zero for all the other places.

The target marking for this example is $\mathbf{m}_r(p_1) = 1.6$, $\mathbf{m}_r(p_2) = \mathbf{m}_r(p_3) = \mathbf{m}_r(p_4) = \mathbf{m}_r(p_5) = 0.6$, $\mathbf{m}_r(p_6) = 3$, $\mathbf{m}_r(p_7) = \mathbf{m}_r(p_8) = \mathbf{m}_r(p_9) = \mathbf{m}_r(p_{10}) = 0.5$, $\mathbf{m}_r(p_{11}) = 5$, $\mathbf{m}_r(p_{12}) = \mathbf{m}_r(p_{13}) = \mathbf{m}_r(p_{14}) = \mathbf{m}_r(p_{15}) = 2.5$.

The firing rates for the transitions are given by

$$\lambda = [\ 1\ \ 1\ \ 1\ \ 1\ \ 2\ \ 2\ \ 2\ \ 2\ \ 3\ \ 3\ \ 3\ \ 3\ \ 3\ ]^T$$

This *ContPN* is observable with an output matrix

$$S = \begin{bmatrix} e_5^T & e_6^T & e_{10}^T & e_{11}^T \end{bmatrix}^T$$

The Single Input and Join transitions sets are $T_S = \{t_1, t_2, t_3, t_4, t_6, t_7, t_8, t_{10}, t_{11}, t_{12}, t_{13}\}$ and $T_J = \{t_5, t_9\}$ respectively.

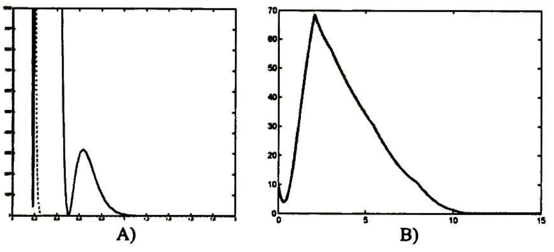


Figure 6.4.2: A) Quadratic Luenberger observer error. B) Quadratic regulation error.

In the Luenberger observer, the correction gain matrix L can be computed to obtain any desired dynamics for the error system. Particularly for this example, two correction gain matrices L were designed. The first one was designed in order to make the quadratic error of the Luenberger observer $\tilde{\mathbf{m}}^T \tilde{\mathbf{m}} \rightarrow 0$ in 0.25 time units. The second one was designed to make $\tilde{\mathbf{m}}^T \tilde{\mathbf{m}} \rightarrow 0$ in 1.25 time units. In Figure 6.4.2 A) the quadratic observer error for both simulations is shown.

Once $\tilde{\mathbf{m}}^T \tilde{\mathbf{m}} \to 0$, in time $\tau = 2$ time units, the control action (2.11) is applied in both simulations (each with a different gain matrix L). In Fig. 6.4.2 B) the quadratic regulation error $(\mathbf{m} - \mathbf{m}_r)^T(\mathbf{m} - \mathbf{m}_r)$ is shown. It can be seen that in proximately 11 time units the target marking is reached. Of course, in both simulations, the marking evolution for the *ContPN* is the same.

7

Case of study

This chapter is devoted to present a case of study to illustrate the previously presented theory about sensor placement for observability in *ContPN* and observer design (see chapters 4 and 5). The case of study is about the cigarettes production and packaging. In Section 7.1 a brief description of the system will be presented. Then, in Section 7.2, an abstraction of the system (a model) will be obtained as a *ContPN*. Then, the concepts presented in previous chapters will be used to determine a proper instrumentation to guarantee that the *ContPN* is observable. Then, the model will be analyzed and simplified based on the knowledge of the system dynamics, leading to a simpler model. This simplification will be presented in Section 7.3. Again, this work's theoretical results will be used to determine a sensor placement selection such that the simplified model is also observable. With this simplified model, an observer will be constructed as presented in Chapter 6. Finally, in Section 7.4, the simplified model will be simulated and the results for the observer will be presented.

7.1 Process description

The cigarette production and packaging is separated into three main steps:

1. **Cigarette formation.** In this process the cigarette itself is made. It includes the rod formation, the filter assembly and the delivery of the cigarette to a buffer machine.

2. **Packing.** A standard pack of cigarettes contain 20 cigarettes. The packs are carton made and covered with polypropylene. The pack also has an *auto strip*, which is a polymer strip that allows the polypropylene straight ripping.

3. **Packaging process.** A standard package contains 10 packs, which are put together and then covered with polypropylene.

The whole production is made within a production module, also named *link-up*. In Figure 7.1.1 a link-up is presented. A standard link-up, such as the one shown in Figure 7.1.1 requires two operators and an assistant operator per two modules. In an eight hour shift, a standard link-up is expected to produce 130,000 packs of cigarettes, with the equivalent of 2,600,000 cigarettes plus scrap production. These values are based on a statistics of machines of fifth generation (medium speed) as the ones modelled.

Figure 7.1.1: A cigarette production module.

7.1.1 Cigarette formation.

The cigarette formation is a process which includes several machines connected one to the other as presented in the schematic (Figure 7.1.2).

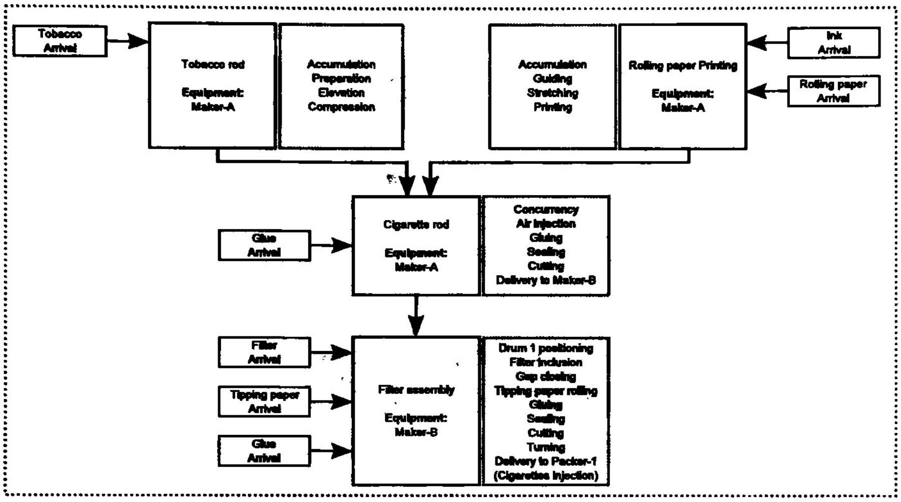


Figure 7.1.2: Cigarette formation schematic.

The process begins with the tobacco arrival. The tobacco is placed in a chute. To lead the tobacco to the chute there exists a pneumatic system, which ensures a ***regular*** amount of tobacco in the chute. The chute and the tobacco arrival can be seen in Figure 7.1.3. The tobacco arrives from the left of the chute. On the right of the chute there is also a pipe with an arrow. This pipe removes tobacco powder and maintains the chute with an appropriate pressure.

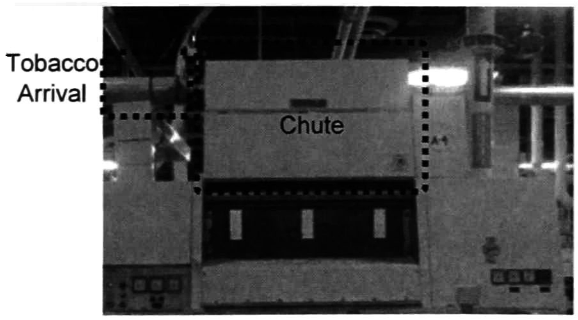


Figure 7.1.3: Tobacco arrival and the chute.

Once in the chute, the tobacco is prepared with an air curtain trough a brushing device from the top of the machine to its bottom. After the preparation, the tobacco is once again elevated with air to a compression device. At this point, the tobacco has already the rod shape, which will be assembled with the rolling paper. This part of the process is made in the Maker-A machine, as shown in Figure 7.1.4.

Figure 7.1.4: Tobacco preparation and compression.

On the other hand and in a parallel process, the rolling paper (cigarette paper) is processed in another part of the Maker-A machine. The paper is first accumulated so it can be pulled with a constant speed. The paper is prepared trough a guide. Trough this guide, the paper will be stretched and printed. This part of the process is made in the Maker-A machine as shown in Figure 7.1.5

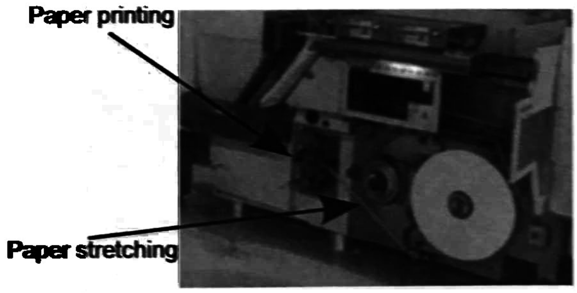


Figure 7.1.5: Paper stretching and printing.

Finally the paper already printed will be assembled together with the tobacco rod in the *former* of the Maker-A machine. In order to make the assembly, both the cigarette rod and the printed paper should join together in the Maker-A machine in a concurrency step. When the paper is being folded, air is injected in order to avoid the presence of tobacco particles in the cigarette's seem (this is considered a very relevant non-conformity of quality, since it can make the cigarette to open because of the lack of glue). Then, the rolling paper is completely folded around the tobacco rod, gluing the seem. After the gluing, the cigarette rod is sealed with a heating device. This process delivers a continuous cigarette rod. This process is made in the Maker-A machine as shown in Figure 7.1.6

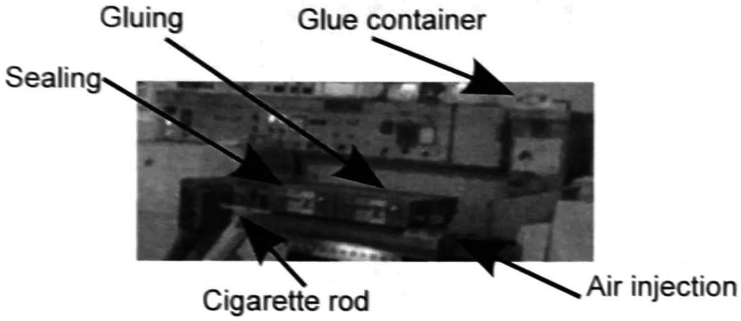


Figure 7.1.6: Cigarette assembly (cigarette rod formation).

Once a continuous cigarette rod is obtained, it is cut into smaller pieces of the appropriate dimension in the cutter of the Maker-A machine. The process of the Maker-A machine delivers up to 7,000 cigarettes per minute. In Figure 7.1.7 it is shown where the cutting process is made in the Maker-A machine.

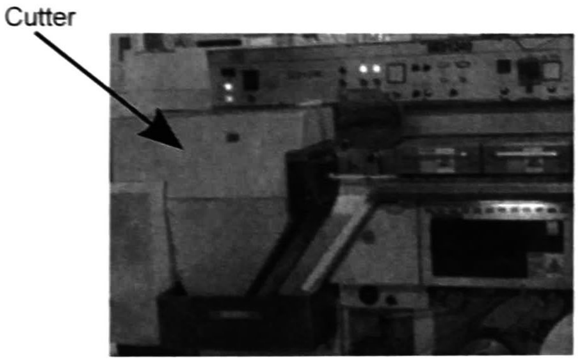


Figure 7.1.7: Cigarette cutting.

After the rod is cut into pieces, the pieces are delivered to the Maker-B machine. The

Maker-B machine is composed of several drums in which the filter assembled with the cigarette rod. The first step is devoted to place properly two cigarette rods with an appropriate distance between them. In the second step, a filter is added in between them. Then, a different drum closes the existing gap between both the cigarettes rods and the filter (Step 3). When the gap is closed, the tipping paper is added (Step 4). Right after that, the tipping paper is glued (Step 5) and rolled (Step 6). Then, with a heating drum, the complete rod is sealed (Step 7). Then, in a cutting drum, the rod is cut in half (Step 8) in order to finally turn it around (Step 9) so all the cigarettes are delivered to the buffer in the same position. This process, step by step, can be seen in Figure 7.1.8.

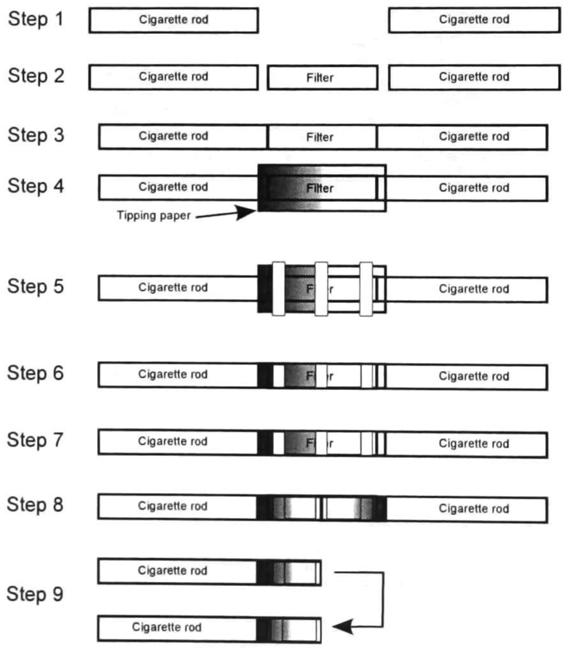


Figure 7.1.8: Filter assembly process.

The equipment which is devoted to make the previous process is a Maker-B machine. In Figure 7.1.9 it is possible to see the Maker-B in operational mode.

In Figure 7.1.10, the Maker-B is empty and clean, so all its components are visible.

Similarly to the Maker-A, the Maker-B is also capable to deliver up to 7,500 cigarettes per minute. Once the cigarettes are ready and aligned, they are sent into a buffer where cigarettes

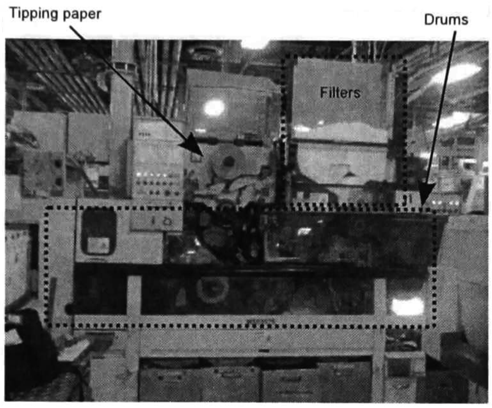


Figure 7.1.9: Maker-B.

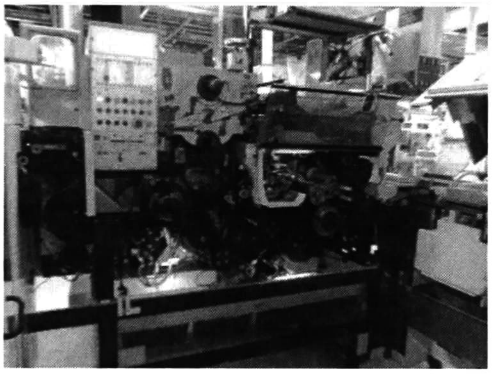


Figure 7.1.10: Maker-B. Open machine.

can be extracted in containers or they can be sent to continue into the Packer-1 machine. This part of the equipment can be seen in Figure 7.1.11. This part of the process is important since it ensures a minimum amount of cigarettes available for the Packer-1 machine. Also, if it was necessary, it is possible to obtain cigarettes from the maker and take them into a different packer, since there are several packers which can use the same cigarette for different pack's presentation.

In order to guarantee the cigarettes' availability, the cigarettes collected in the buffer or made on a different maker machine, can be inserted to continue into the packing process with the help of a cigarettes tray turner machine. This equipment can be seen in Figure 7.1.12

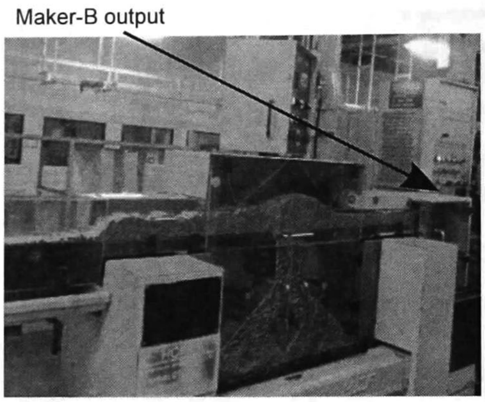


Figure 7.1.11: Cigarette tray filler. A cigarettes buffer.

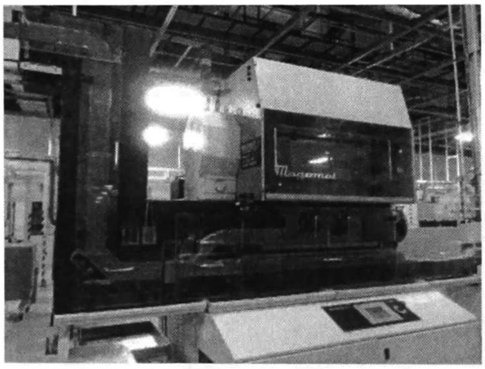


Figure 7.1.12: Cigarette tray turner machine.

With this equipment, the cigarette formation process ends. The complete cigarettes' production is usually also named the making process. It is important to mention that the making process can be understood as a *supplier* process for the packing and packaging process. However, the making process can be also a part of a link-up process, as explained in this work. Therefore the cigarette tray filler machine and the cigarette tray turner machine will not be included in the model, since they only represent an increase of cigarettes' capacity in the equipment.

Now, in the next subsection, the packing process will be described.

7.1.2 Packing process.

The packing is made in the second set of equipment on a link-up. An standard set of cigarettes (20 of them) will be covered by an aluminium layer, a carton layer and a polypropylene film. This step concludes the packing process. An schematic of this process is presented in Figure 7.1.13.

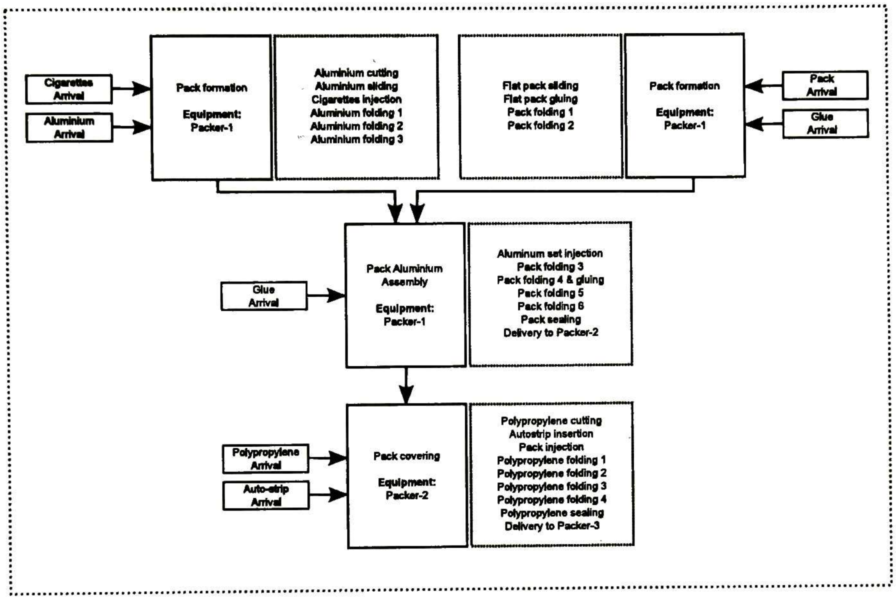


Figure 7.1.13: Packing process schematic.

The packing process starts with the accumulation of the cigarettes on a guided chute. The cigarettes are guided into a set of channels, where an injector *pushes* them into the folded aluminium.

The aluminium is cut and prepared to *receive* the 20 cigarettes set in a slider component. The cigarettes are then pushed into the cut aluminium. The set aluminium-cigarettes is then folded in such a way that the cigarettes are completely covered by the aluminium paper.

Then on a parallel process, the pack is being sided, glued and folded in order to assemble it with the aluminium set. In the aluminium set assemble process with the pack, the pack goes trough several folding steps and a sealing step. All these steps but the sealing are made in a Packer-1 machine. This machine can be seen in Figure 7.1.14.

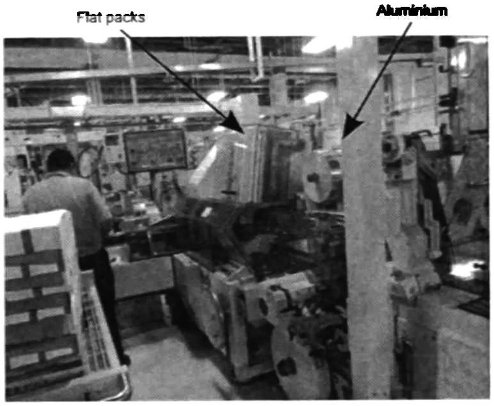


Figure 7.1.14: Packer-1 machine.

The sealing step is made on the last part of the Pack-1 machine. It is made trough a heating device which allows the glue to quickly dry out. The sealing device can be seen in Figure 7.1.15.

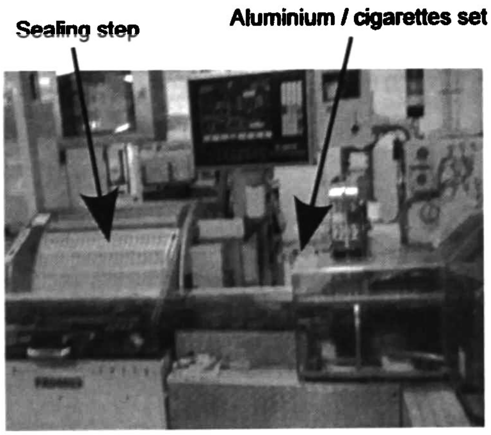


Figure 7.1.15: Sealing device. Packer-1 machine.

Finally, the polypropylene film is added to the pack. The polypropylene film has an auto-strip that allows the film to rip aligned to the top *opening-part* of the pack. This process is made on a Packer-2 machine, which can be seen in Figure 7.1.16.

The packing process ends with this step. In the next subsection, the packaging process will be described..

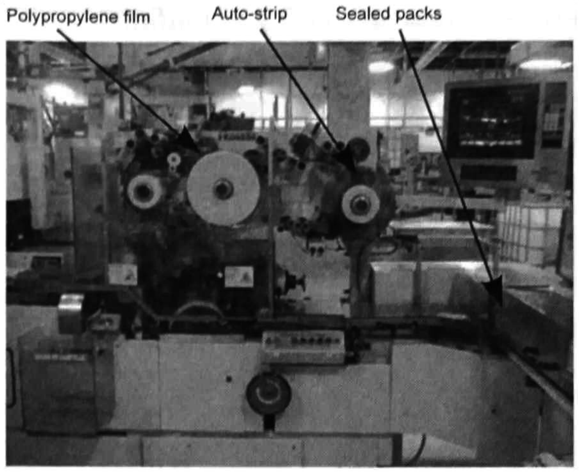


Figure 7.1.16: Packer-2 machine.

7.1.3 Packaging process.

In the packaging process, 10 packs are assembled into one package. This process consists of the injection of two layers of five packs which are covered by a polypropylene film. Then the film is properly folded and delivered as a final product. The schematic of this process can be seen in Figure 7.1.17.

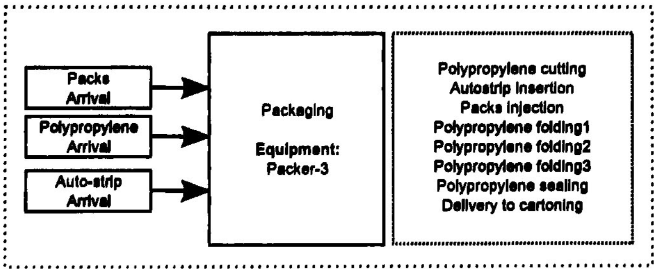


Figure 7.1.17: Packaging process schematic.

This process is made on a Packer-3 machine. This machine can be seen in Figure 7.1.18.

The packages, consistent on 10 packs will be later included into a carton. This process is not included since it is made in a different location, i.e. out of the link-up. Actually, the packages are taken from the link-up to be inserted on the carton box all around the factory

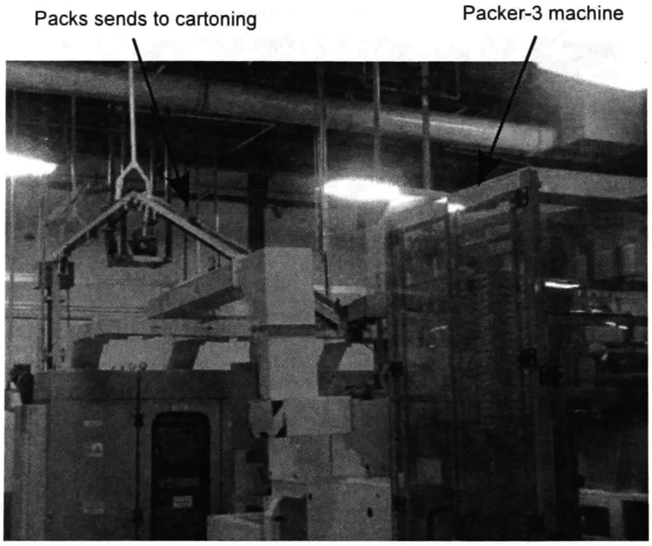


Figure 7.1.18: Packer-3 machine.

trough a belt conveyor. Each carton contains 50 packages, i.e. 500 packs. This step of the process is known as cartoning.

In the final step, the carton goes into a wrapper and a labelling machine. Then, the cartons are taken into the finished goods warehouse.

7.2 Process representation as a *ContPN*

In this section the *ContPN* representation of the previously described processes will be presented.

7.2.1 Cigarette formation.

The cigarette formation process' representation as a *ContPN* can be seen in Figure 7.2.1.

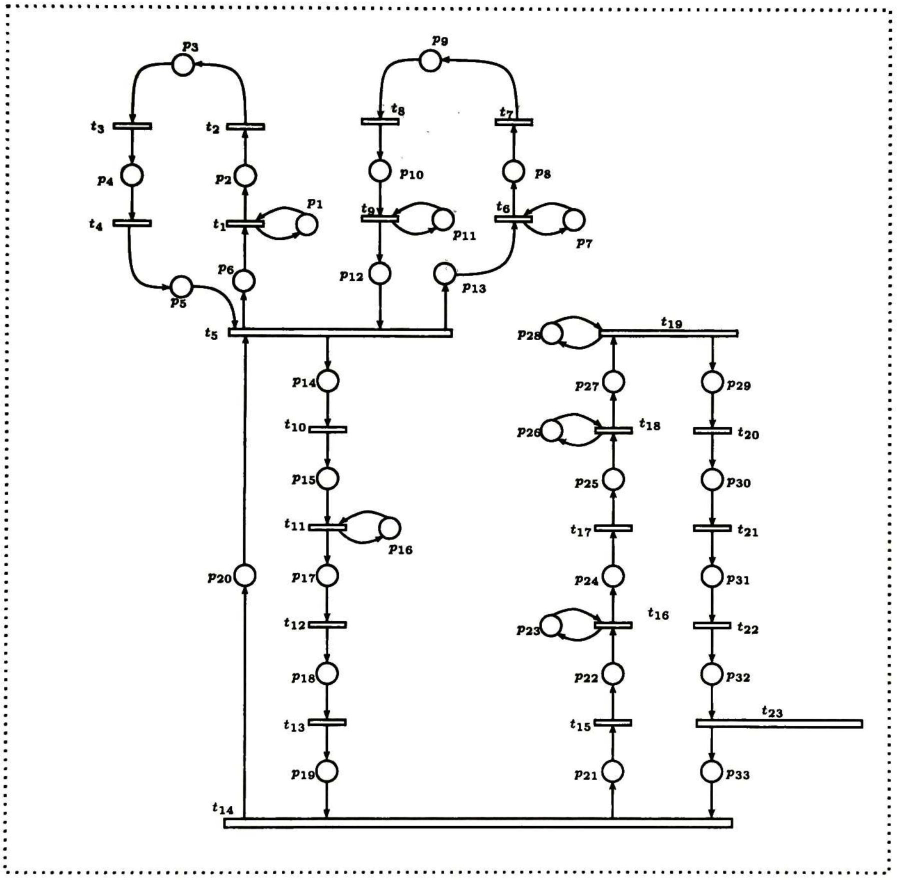


Figure 7.2.1: Cigarette formation *ContPN* representation.

The interpretation of the nodes in the *ContPN* presented in Figure 7.2.1 is given in the Figure 7.2.2 (refer to Figure 7.1.2 for the schematic process).

p	Interpretation	*t*	Interpretation
1	Tobacco (infinite) warehouse	1	Tobacco arrival to chute
2	Tobacco in the chute	2	Tobacco preparation
3	Tobacco in preparation compartment	3	Tobacco elevation
4	Tobacco in the elevator	4	Tobacco compression
5	Compressed tobacco	5	Concurrency with rolling paper
6	Tobacco compartment free capacity	6	Paper accumulation
7	Rolling paper (infinite) buffer	7	Paper guiding
8	Accumulated paper	8	Paper stretching
9	Guided paper	9	Paper printing
10	Stretched paper	10	Air injection
11	Ink (infinite) buffer	11	Gluing
12	Printed paper	12	Sealing
13	Printer free capacity	13	Cutting
14	Tobacco-paper (TP) rod	14	Delivery to Maker-B
15	Small particles free TP rod	15	Drum positioning
16	Glue (infinite) container	16	Filter inclusion
17	Cigarette rod	17	Gap closing
18	Sealed cigarette rod	18	Tipping paper rolling
19	Cut cigarette rod	19	Gluing
20	Maker-A free capacity	20	Sealing
21	Cigarette rods in Maker-B	21	Cutting
22	Positioned cigarette rods	22	Turning
23	Filter (infinite) buffer	23	Delivery to buffer
24	Cigarette-Filter-Cigarette (spaced)		
25	Cigarette-Filter-Cigarette rod		
26	Tipping paper (infinite) buffer		
27	Double cigarette rod		
28	Glue (infinite) container		
29	Glued double cigarette rod		
30	Sealed double cigarette rod		
31	Cut double cigarette rod		
32	Maker-B free capacity		

Figure 7.2.2: Nodes interpretation of the cigarette making.

7.2.2 Packing process.

The *ContPN* which represents the packing process can be seen in Figure 7.2.3.

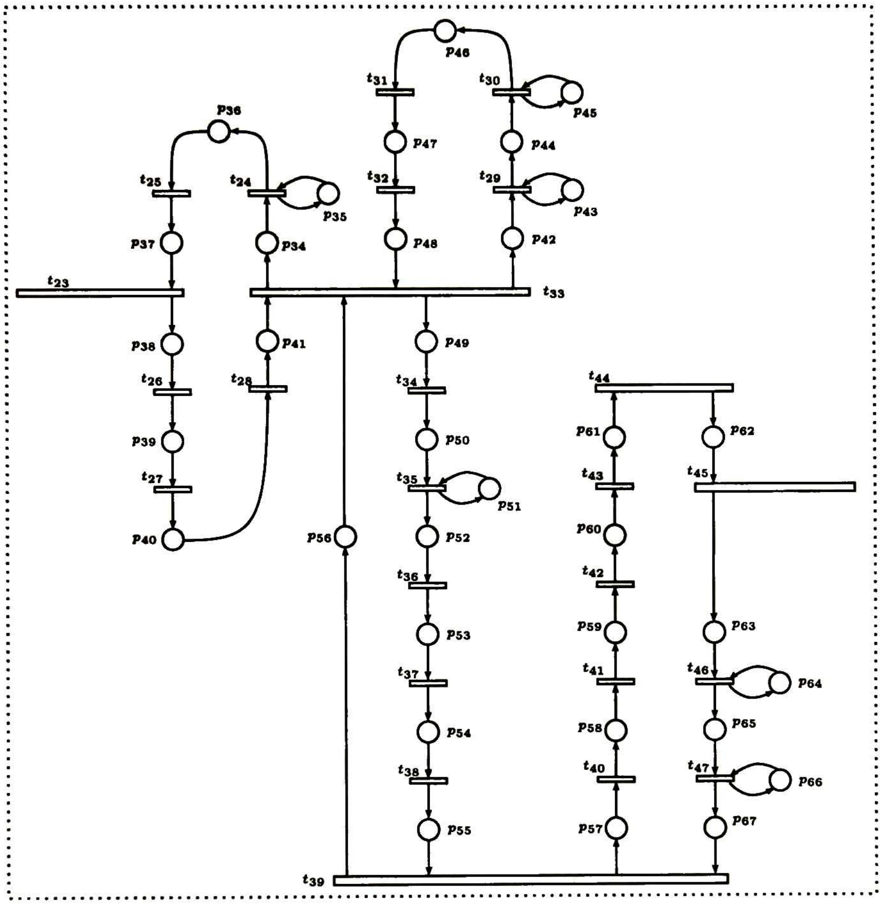


Figure 7.2.3: Packing process *ContPN* representation.

The interpretation of the nodes in the *ContPN* presented in Figure 7.2.3 is given in the Figure 7.2.4 (refer to Figure 7.1.13 for the schematic process).

p	Interpretation	*t*	Interpretation
33	Packer-1 aluminium free capacity	23	Cigarettes injection in Packer-1
34	Aluminium paper (infinite) buffer	24	Aluminium cutting
35	Cut aluminium	25	Aluminium sliding
36	Slided (ready) aluminium	26	Aluminium folding 1
37	Aluminium-cigarette (A&C) set	27	Aluminium folding 2
38	Folded 1 A&C set	28	Aluminium folding 3
39	Folded 2 A&C set	29	Flat pack sliding
40	Folded 3 A&C set	30	Flat pack gluing
41	Packer-1 flat pack free capacity	31	Pack folding 1
42	Flat pack (infinite) buffer	32	Pack folding 2
43	Slided (ready) flat packs	33	Aluminium set injection
44	Glue (infinite) buffer	34	Pack folding 3
45	Glued flat packs	35	Pack folding 4
46	Folded 1 flat packs	36	Pack folding 5
47	Folded 2 flat packs	37	Pack folding 6
48	Pack & A&C set	38	Pack sealing
49	Folded 3 pack & A&C set	39	Delivery to Packer-2
50	Glue (infinite) buffer	40	Pack & polypropylene folding 1
51	Glued and folded 4 pack & A&C set	41	Pack & polypropylene folding 2
52	Folded 5 pack & A&C set	42	Pack & polypropylene folding 3
53	Folded 6 pack & A&C set	43	Pack & polypropylene folding 4
54	Sealed pack & A&C set	44	Pack & polypropylene sealing
55	Paker-1 Pack folding free capacity	45	Delivery to Packer-3
56	Packs & polypropylene	46	Polypropylene cutting
57	Folded 1 packs & polypropylene	47	Autostrip insertion
58	Folded 2 packs & polypropylene		
59	Folded 3 packs & polypropylene		
60	Folded 4 packs & polypropylene		
61	Sealed packs & polypropylene		
62	Packer-2 free capacity		
63	Polypropylene (infinite) buffer		
64	Cut polypropylene		
65	Autostrip (infinite) buffer		
66	Polypropylene & autostrip set		

Figure 7.2.4: Nodes interpretation of the packing process.

7.2.3 Packaging process.

The *ContPN* which represents the packaging process can be seen in Figure 7.2.5.

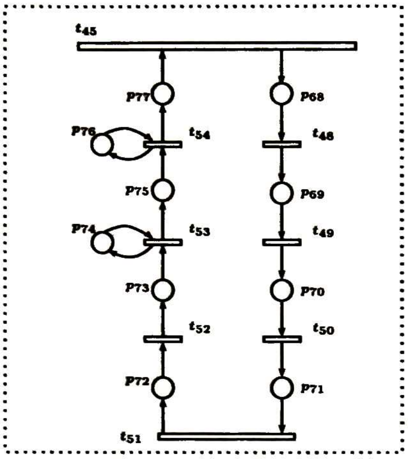


Figure 7.2.5: Packaging process *ContPN* representation.

The interpretation of the nodes in the *ContPN* presented in Figure 7.2.5 is given in the Figure 7.2.6 (refer to Figure 7.1.17 for the schematic process).

p	Interpretation	*t*	Interpretation
67	Package with polypropylene	45	From Packer-2 (packs insertion)
68	Folded 1 package	48	Polypropylene folding 1
69	Folded 2 package	49	Polypropylene folding 2
70	Folded 3 package	50	Polypropylene folding 3
71	Sealed package	51	Polypropylene sealing
72	Packer 3 free capacity	52	Delivery to cartoning
73	Polypropylene (infinite) buffer	53	Polypropylene cutting
74	Cut polypropylene	54	Autostrip insertion
75	Autostrip (infinite) buffer		
76	Polypropylene & autostrip set		

Figure 7.2.6: Nodes interpretation of the packaging process.

7.2.4 Complete model.

The complete model is shown in Figure 7.2.7. For simplicity of the representation, all the labels were removed but the synchronizations between processes.

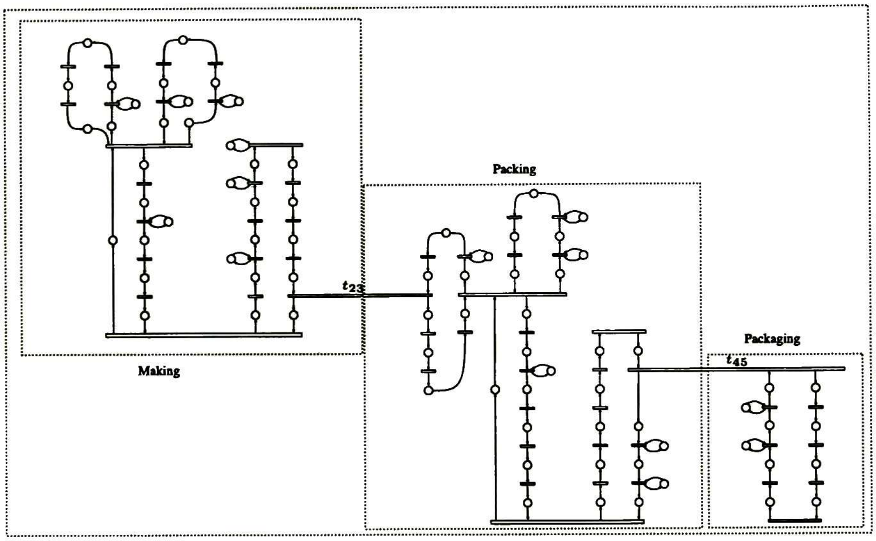


Figure 7.2.7: Packaging process *ContPN* representation.

7.2.5 Observability of the complete model.

The *ContPN* in Figure 7.2.7 has 20 join transitions. The join transitions set is

$$T_J = \{t_1, t_5, t_6, t_9, t_{11}, t_{14}, t_{16}, t_{18}, t_{19}, t_{23}, t_{24}, t_{29}, t_{30}, t_{33}, t_{35}, t_{39}, t_{45}, t_{46}, t_{47}, t_{53}, t_{54}\}$$

In order to represent it as a *SLS* it is necessary to model a family $\mathcal{F}$ of *LS* with 4,718,592 *LS*, due to the join transitions. In order to compute the number of *LS* needed to represent the *ContPN*, it is necessary to compute.

$$\prod_{i=1}^{|T|} |\bullet t_i|.$$

Also, in order to make the *ContPN* using Algorithms 5.1.9 and 5.1.10 returns an output such that the set of measured places is:

$$P_M = \left\{ \begin{array}{c} p_1, p_3, p_5, p_6, p_7, p_8, p_{10}, p_{11}, p_{12}, p_{13}, \\ p_{15}, p_{16}, p_{17}, p_{19}, p_{20}, p_{22}, p_{23}, p_{25}, p_{26}, p_{27}, \\ p_{28}, p_{30}, p_{32}, p_{33}, p_{34}, p_{35}, p_{37}, p_{39}, p_{41}, p_{42}, \\ p_{43}, p_{44}, p_{45}, p_{47}, p_{48}, p_{50}, p_{51}, p_{53}, p_{55}, p_{56}, \\ p_{58}, p_{60}, p_{62}, p_{63}, p_{64}, p_{65}, p_{66}, p_{67}, p_{69}, p_{71}, \\ p_{73}, p_{74}, p_{75}, p_{76}, p_{77} \end{array} \right\}$$

i.e. 55 measured places in order to compute the marking of 77 places. The distinguishability in this case is guaranteed, since every input place to a join transition is measured.

Using the sensor reduction algorithm (see Algorithm 5.1.14), the set of measured places is reduced to the set:

$$P_{M2} = \left\{ \begin{array}{c} p_1, p_5, p_6, p_7, p_{10}, p_{11}, p_{12}, p_{13}, \\ p_{15}, p_{16}, p_{19}, p_{20}, p_{22}, p_{23}, p_{25}, p_{26}, p_{27}, \\ p_{28}, p_{32}, p_{33}, p_{34}, p_{35}, p_{37}, p_{41}, p_{42}, \\ p_{43}, p_{44}, p_{45}, p_{48}, p_{50}, p_{51}, p_{55}, p_{56}, \\ p_{62}, p_{63}, p_{64}, p_{65}, p_{66}, p_{67}, \\ p_{73}, p_{74}, p_{75}, p_{76}, p_{77} \end{array} \right\}$$

which only contains 44 places.

7.3 Model simplification

The previously presented model can be simplified under the following consideration:

The material buffers (tobacco, rolling paper, tipping paper, glue, etc.) are infinite.

With previous consideration, it is possible to see that the evolution of the *ContPN* does not depend on such material buffers. Therefore, the marking of the set of places

$$P_{source} = \{p_1, p_7, p_{11}, p_{16}, p_{23}, p_{26}, p_{28}, p_{35}, p_{43}, p_{45}, p_{51}, p_{64}, p_{66}, p_{74}, p_{76}\}$$

can be considered as known $\forall\tau$.

Now, let the source transitions set be defined as $P_{source}\bullet = T_{source}$. Consider a transition $t_i \in T_{source}$. Each of these transitions has two input places, one which belongs to P_{source} and the other which does not. Then, the transition t_i's flow is always constrained by the place $p_j \in \bullet t_i$ such that $p_j \notin P_{source}$. Then, it is not longer necessary to measure the set of places

$$P_{M3} = \left\{ \begin{array}{c} p_1, p_6, p_7, p_{10}, p_{11}, p_{13}, p_{15}, p_{16}, p_{22}, p_{23}, \\ p_{25}, p_{26}, p_{27}, p_{28}, p_{34}, p_{35}, p_{42}, p_{43}, p_{44}, p_{45}, \\ p_{50}, p_{51}, p_{63}, p_{64}, p_{65}, p_{66}, p_{73}, p_{74}, p_{78}, p_{76}, \end{array} \right\}$$

i.e. 30 places can be removed from the set of measured places. It is important to remark that the distinguishability property is also preserved, since every input place to a join transition (which has a non zero dynamics) is measured; and, for the ones with zero dynamics, its marking is assumed to be known and those places never constrain the flow of their output join transitions.. In this way, the set of measured places that guarantees observability of the *ContPN* in Figure 7.2.7 and its output matrix is given by:

$$P_{M4} = P_{M2} - P_{M4} = \left\{ \begin{array}{l} p_5, p_{12}, p_{19}, p_{20}, p_{32}, p_{33}, p_{37}, \\ p_{41}, p_{48}, p_{55}, p_{56}, p_{62}, p_{67}, p_{77} \end{array} \right\}$$

$$S = \begin{bmatrix} \mathbf{e}_5^T \\ \mathbf{e}_{12}^T \\ \mathbf{e}_{19}^T \\ \mathbf{e}_{20}^T \\ \mathbf{e}_{32}^T \\ \mathbf{e}_{33}^T \\ \mathbf{e}_{37}^T \\ \mathbf{e}_{41}^T \\ \mathbf{e}_{48}^T \\ \mathbf{e}_{55}^T \\ \mathbf{e}_{56}^T \\ \mathbf{e}_{62}^T \\ \mathbf{e}_{67}^T \\ \mathbf{e}_{77}^T \end{bmatrix}$$

7.4 Model simulation

Using the approach proposed in 6, a single observer was designed.

The *ContPN* and its observer were simulated on Simulink ®, based on Matlab ®.

The Simulink ®diagram can be seen in Figure 7.4.1.

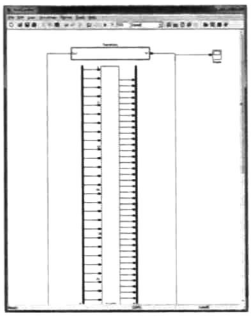


Figure 7.4.1: Simulink ®diagram.

The main component is a *differential equation editor*, which contains the *ContPN* described on Section 7.3. In the differential equation editor, the dynamics of the *ContPN* is captured, together with its initial state and the output variables. In order to have the state marking $\mathbf{m}$ as a reference, all the markings were available in the output. The differential equation editor can be seen in Figure 7.4.2.

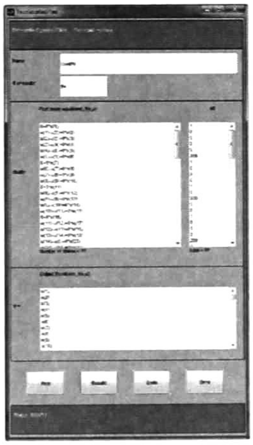
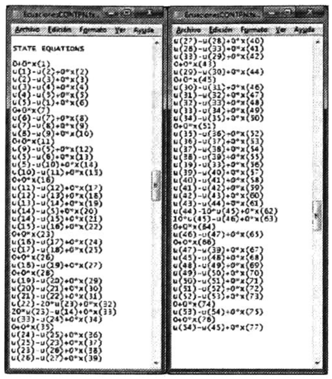


Figure 7.4.2: Differential equation editor.

The state equations used in the differential equation editor can be seen in Figure 7.4.3. It is important to recall that these equations represent the component $\dot{\mathbf{m}} = \mathbf{C}_S\Lambda_S\Pi_S\bar{\mathbf{m}}$, i.e. the single input dynamics.


Archivo Edición Formato Ver Ayuda
Archivo Edición Formato Ver Ayuda

```
STATE EQUATIONS

0+0*x(1)
u(1)-u(2)+0*x(2)
u(2)-u(3)+0*x(3)
u(3)-u(4)+0*x(4)
u(4)-u(5)+0*x(5)
u(5)-u(1)+0*x(6)
0+0*x(7)
u(6)-u(7)+0*x(8)
u(7)-u(8)+0*x(9)
u(8)-u(9)+0*x(10)
0+0*x(11)
u(9)-u(5)+0*x(12)
u(5)-u(6)+0*x(13)
u(5)-u(10)+0*x(14)
u(10)-u(11)+0*x(15)
0+0*x(16)
u(11)-u(12)+0*x(17)
u(12)-u(13)+0*x(18)
u(13)-u(14)+0*x(19)
u(14)-u(5)+0*x(20)
u(14)-u(15)+0*x(21)
u(15)-u(16)+0*x(22)
0+0*x(23)
u(16)-u(17)+0*x(24)
u(17)-u(18)+0*x(25)
0+0*x(26)
u(18)-u(19)+0*x(27)
0+0*x(28)
u(19)-u(20)+0*x(29)
u(20)-u(21)+0*x(30)
u(21)-u(22)+0*x(31)
u(22)-20*u(23)+0*x(32)
20*u(23)-u(14)+0*x(33)
u(33)-u(24)+0*x(34)
0+0*x(35)
u(24)-u(25)+0*x(36)
u(25)-u(23)+0*x(37)
u(23)-u(26)+0*x(38)
u(26)-u(27)+0*x(39)
```

```
u(27)-u(28)+0*x(40)
u(28)-u(33)+0*x(41)
u(33)-u(29)+0*x(42)
0+0*x(43)
u(29)-u(30)+0*x(44)
0+0*x(45)
u(30)-u(31)+0*x(46)
u(31)-u(32)+0*x(47)
u(32)-u(33)+0*x(48)
u(33)-u(34)+0*x(49)
u(34)-u(35)+0*x(50)
0+0*x(51)
u(35)-u(36)+0*x(52)
u(36)-u(37)+0*x(53)
u(37)-u(38)+0*x(54)
u(38)-u(39)+0*x(55)
u(39)-u(33)+0*x(56)
u(39)-u(40)+0*x(57)
u(40)-u(41)+0*x(58)
u(41)-u(42)+0*x(59)
u(42)-u(43)+0*x(60)
u(43)-u(44)+0*x(61)
u(44)-10*u(45)+0*x(62)
10*u(45)-u(46)+0*x(63)
0+0*x(64)
u(46)-u(47)+0*x(65)
0+0*x(66)
u(47)-u(39)+0*x(67)
u(45)-u(48)+0*x(68)
u(48)-u(49)+0*x(69)
u(49)-u(50)+0*x(70)
u(50)-u(51)+0*x(71)
u(51)-u(52)+0*x(72)
u(52)-u(53)+0*x(73)
0+0*x(74)
u(53)-u(54)+0*x(75)
0+0*x(76)
u(54)-u(45)+0*x(77)
```

Figure 7.4.3: State equations for the *ContPN*.

Since the marking depends on the flow of the transitions, the Simulink ®block named Transitions computes the flow of each transition. The implementation of this computation can be seen in Figure 7.4.4.

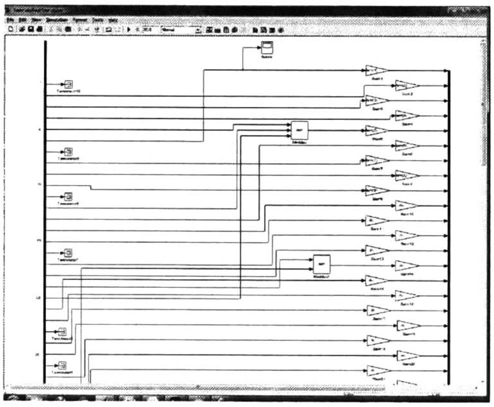


Figure 7.4.4: Transition's flow computation.

The observer for the *ContPN* was also implemented in Simulink ®. Its implementation can be seen in Figure 7.4.5.

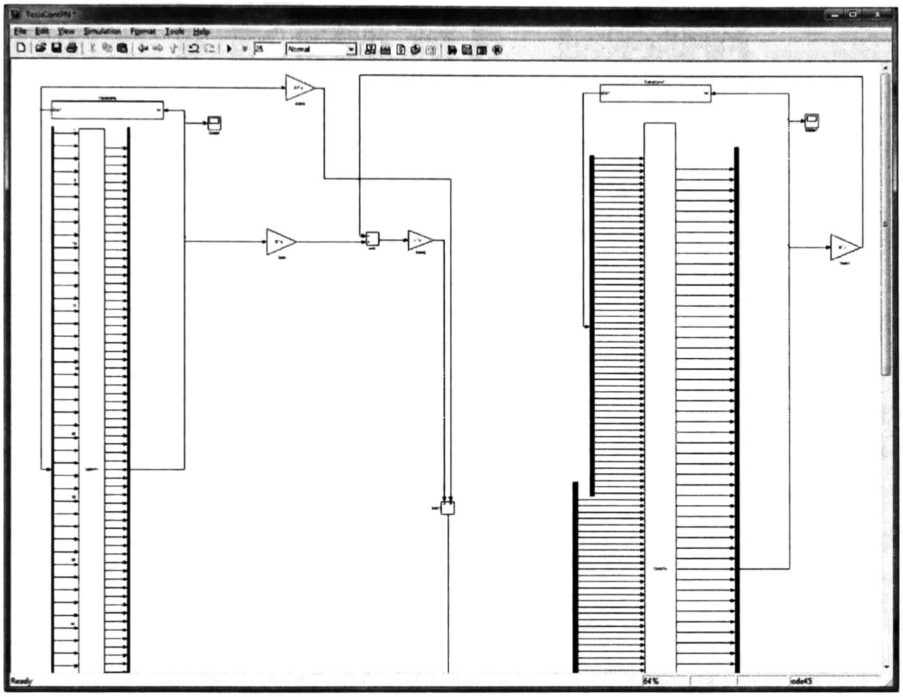


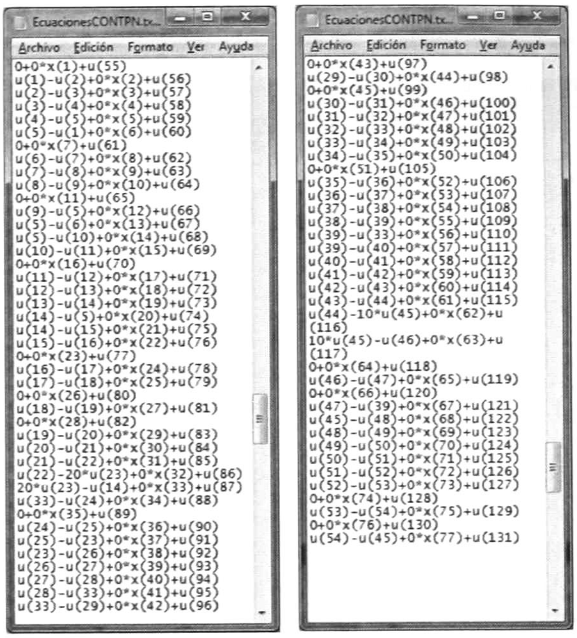
Figure 7.4.5: Simulink ®diagram for the observer.

The observer also includes as an input the component of the join transitions dynamics and the correction factor, given by the term

$$\mathbf{C}_J f_J + LS(y - \bar{y})$$

The state equations for the observer are presented in Figure 7.4.6.

As a result of the simulation, the quadratic error over time of the observed marking (defined by $\tilde{\mathbf{m}} = \mathbf{m} - \bar{\mathbf{m}}$) can be seen in Figure 7.4.7.


EcuacionesCONTPN.tx...
Archivo Edición Formato Ver Ayuda
EcuacionesCONTPN.tx...
Archivo Edición Formato Ver Ayuda

```
0+0*x(1)+u(55)
u(1)-u(2)+0*x(2)+u(56)
u(2)-u(3)+0*x(3)+u(57)
u(3)-u(4)+0*x(4)+u(58)
u(4)-u(5)+0*x(5)+u(59)
u(5)-u(1)+0*x(6)+u(60)
0+0*x(7)+u(61)
u(6)-u(7)+0*x(8)+u(62)
u(7)-u(8)+0*x(9)+u(63)
u(8)-u(9)+0*x(10)+u(64)
0+0*x(11)+u(65)
u(9)-u(5)+0*x(12)+u(66)
u(5)-u(6)+0*x(13)+u(67)
u(5)-u(10)+0*x(14)+u(68)
u(10)-u(11)+0*x(15)+u(69)
0+0*x(16)+u(70)
u(11)-u(12)+0*x(17)+u(71)
u(12)-u(13)+0*x(18)+u(72)
u(13)-u(14)+0*x(19)+u(73)
u(14)-u(5)+0*x(20)+u(74)
u(14)-u(15)+0*x(21)+u(75)
u(15)-u(16)+0*x(22)+u(76)
0+0*x(23)+u(77)
u(16)-u(17)+0*x(24)+u(78)
u(17)-u(18)+0*x(25)+u(79)
0+0*x(26)+u(80)
u(18)-u(19)+0*x(27)+u(81)
0+0*x(28)+u(82)
u(19)-u(20)+0*x(29)+u(83)
u(20)-u(21)+0*x(30)+u(84)
u(21)-u(22)+0*x(31)+u(85)
u(22)-20*u(23)+0*x(32)+u(86)
20*u(23)-u(14)+0*x(33)+u(87)
u(33)-u(24)+0*x(34)+u(88)
0+0*x(35)+u(89)
u(24)-u(25)+0*x(36)+u(90)
u(25)-u(23)+0*x(37)+u(91)
u(23)-u(26)+0*x(38)+u(92)
u(26)-u(27)+0*x(39)+u(93)
u(27)-u(28)+0*x(40)+u(94)
u(28)-u(33)+0*x(41)+u(95)
u(33)-u(29)+0*x(42)+u(96)
```

```
0+0*x(43)+u(97)
u(29)-u(30)+0*x(44)+u(98)
0+0*x(45)+u(99)
u(30)-u(31)+0*x(46)+u(100)
u(31)-u(32)+0*x(47)+u(101)
u(32)-u(33)+0*x(48)+u(102)
u(33)-u(34)+0*x(49)+u(103)
u(34)-u(35)+0*x(50)+u(104)
0+0*x(51)+u(105)
u(35)-u(36)+0*x(52)+u(106)
u(36)-u(37)+0*x(53)+u(107)
u(37)-u(38)+0*x(54)+u(108)
u(38)-u(39)+0*x(55)+u(109)
u(39)-u(33)+0*x(56)+u(110)
u(39)-u(40)+0*x(57)+u(111)
u(40)-u(41)+0*x(58)+u(112)
u(41)-u(42)+0*x(59)+u(113)
u(42)-u(43)+0*x(60)+u(114)
u(43)-u(44)+0*x(61)+u(115)
u(44)-10*u(45)+0*x(62)+u
(116)
10*u(45)-u(46)+0*x(63)+u
(117)
0+0*x(64)+u(118)
u(46)-u(47)+0*x(65)+u(119)
0+0*x(66)+u(120)
u(47)-u(39)+0*x(67)+u(121)
u(45)-u(48)+0*x(68)+u(122)
u(48)-u(49)+0*x(69)+u(123)
u(49)-u(50)+0*x(70)+u(124)
u(50)-u(51)+0*x(71)+u(125)
u(51)-u(52)+0*x(72)+u(126)
u(52)-u(53)+0*x(73)+u(127)
0+0*x(74)+u(128)
u(53)-u(54)+0*x(75)+u(129)
0+0*x(76)+u(130)
u(54)-u(45)+0*x(77)+u(131)
```

Figure 7.4.6: Observer's state equation.

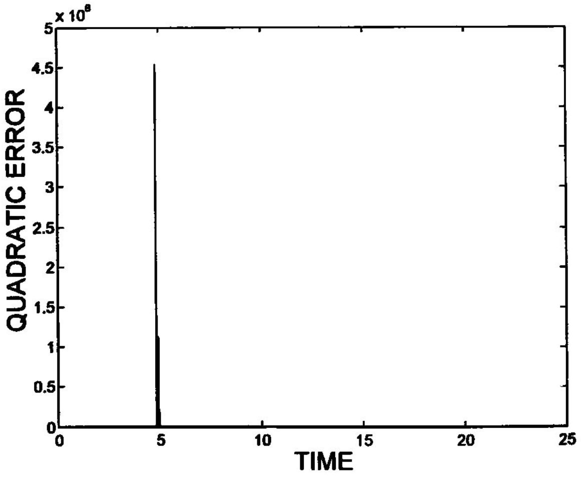


Figure 7.4.7: Quadratic error of the observer.

8

Conclusions and future work

The observability problem in *ContPN*, as in many other applications is relevant to determine the state of the system only by measuring its output. The knowledge of the state of the system is quite relevant in *ContPN*since it makes possible to determine control strategies, isolate faults or supervise the system. Even though *ContPN* can be seen as an autonomous *SLS* and the observability problem in autonomous *SLS* is solved [48] [49], the observability of *ContPN* is not trivial. This is because the observability characterization in autonomous *SLS* requires as a necessary and sufficient condition the observability of each *LS* and the distinguishability between each pair of *LS*. This becomes prohibited in practice since the number of *LS* necessary to represent the *ContPN* as a *SLS* increases exponentially with the number of join transitions. Instead, this work presented two main contributions:

1. An strategy for sensor placement in *ContPN* which guarantees observability. In order to achieve this, some of the invariant subspaces of the *ContPN* are characterized from the *ContPN* structure, i.e. the underlying graph of the *PN*. This avoids the need of the enumeration and computation of the dynamical matrices of each *LS* in the *SLS* representation of the *ContPN*. The A_k-invariant subspaces are divided into two disjoin sets: $\ker(A_k)$ and all the other A_k-invariant subspaces. In this work $\ker(A_k)$ is completely characterized. For the rest of the A_k-invariant subspaces, it is shown that they are also contained in $Im(\mathbf{C})$. Then, it is proposed an algorithm to place sensors by ensuring that neither $\ker(A_k)$ nor $Im(\mathbf{C})$ is contained in $\ker(S)$. Therefore there is not any A_k-invariant subspace contained in $\ker(S)$, i.e. each *LS* is observable. However, the number of sensors placed on the *ContPN* may not be minimum, since $Im(\mathbf{C})$ is

greater than the A_k−invariant subspaces. It is also shown that for the free choice class of *ContPN*, distinguishability is a consequence of observability in each *LS*.

2. An observer design, which allows the computation of the marking of the *ContPN* with a single observer structure for the free choice class of nets. This result is relevant since no matter the number of *LS* required to represent the *ContPN*, only one observer is required.

The future work consists on dealing with the following topics:

1. **Distinguishability**. It has been shown that distinguishability is a consequence of observability in each *LS* for the free choice class of *ContPN*. However, this is not true in the general case. Then, it is relevant to provide results for the distinguishability property in the general case of *ContPN*, or an strategy for sensor placement which guarantees the distinguishability property.

2. **Closed loop Control and observer**. It is important to determine an strategy for the integration of the presented observer with a control strategy, which allows to lead the marking of the *ContPN* from an initial marking to a required one only by the knowledge of its output.

3. **Optimal sensor placement**. Another open discussion is to determine the minimum number of sensors required (and their locations) in order to guarantee observability in *ContPN*.

References

[1] David L. Pepyne and Christos G. Cassandras. **Modeling, Analysis, and Optimal Control of a Class of Hybrid Systems.** *Discrete Event Dynamic Systems*, 8(2):175–201, 1998. 1, 8

[2] Yanfeng Geng and Christos G. Cassandras. **Multi-intersection Traffic Light Control Using Infinitesimal Perturbation Analysis.** *CoRR*, abs/1204.1910, 2012. 1

[3] Christos G. Panayiotou and Christos G. Cassandras. **Optimization of kanban-based manufacturing systems.** *Automatica*, 35(9):1521–1533, 1999. 1, 8

[4] Christos G. Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. Springer US, 2008. 1, 3

[5] Christos G. Cassandras, Jung-Im Lee, and Yu-Chi Ho. **Efficient Parametric Analysis of Performance Measures for Communication Networks.** *IEEE Journal on Selected Areas in Communications*, 8(9):1709–1722, 1990. 1

[6] Wan Fokkink. *Introduction to Process Algebra*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 1st edition, 2000. 1

[7] Richard Johnsonbaugh. *Matemáticas discretas*. Prentice Hall, 4ta edition, 1997. 1

[8] Bernd Steinbach and Christian Posthoff. *Logic Functions and Equations*. Springer, 1st edition, 2009. 1

[9] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation (3rd Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2006. 1

[10] Carl Adam Petri. *Kommunikation mit Automaten*. PhD thesis, Darmstadt University of Technology, Germany, 1962. 1, 8

[11] Lei Miao and Christos G. Cassandras. **Optimality of static control policies in some discrete-event systems.** *IEEE Trans. Automat. Contr.*, 50(9):1427–1431, 2005. 1

[12] Hervé Marchand, Olivier Boivineau, and Stéphane Lafortune. **On the Synthesis of Optimal Schedulers in Discrete Event Control Problems with Multiple Goals.** *SIAM J. Control and Optimization*, 39(2):512–532, 2000. 1

[13] Hervé Marchand, Olivier Boivineau, and Stéphane Lafortune. **On optimal control of a class of partially observed discrete event systems.** *Automatica*, 38(11):1935–1943, 2002. 1

[14] Jörg Desel and Javier Esparza. *Free Choice Petri Nets*. Cambridge University Press, 1995. 2, 8, 9, 12, 16

[15] Tadao Murata. **Petri nets: Properties, analysis and applications.** *Proceedings of the IEEE*, 77(4):541 –580, April 1989. 2

[16] Manuel Silva, Jorge Júlvez, Cristian Mahulea, and C. Vázquez. **On fluidization of discrete event models: observation and control of continuous Petri nets.** *Discrete Event Dynamic Systems*, 21:427–497, 2011. 10.1007/s10626-011-0116-9. 2, 3, 8, 18, 21, 31

[17] Yong Liu and Weibo Gong. **Perturbation Analysis for Stochastic Fluid Queueing Systems.** *Discrete Event Dynamic Systems*, 12(4):391–416, October 2002. 2

[18] K. Qian and D. McDonald. **An approximation method for complete solutions of Markov-modulated fluid models.** In *Global Telecommunications Conference, 1995. GLOBECOM '95., IEEE*, 2, pages 1406 –1411 vol.2, nov 1995. 2

[19] J. Hillston. **Fluid flow approximation of PEPA models.** In *Quantitative Evaluation of Systems, 2005. Second International Conference on the*, pages 33 – 42, sept. 2005. 2

[20] M. Silva and C. Mahulea. **Fluidization and fluid views of discrete event systems.** In *System Theory, Control, and Computing (ICSTCC), 2011 15th International Conference on*, pages 1 –10, oct. 2011. 2

[21] René David and Hassane Alla. **Autonomous And Timed Continous Petri Nets.** In *Applications and Theory of Petri Nets*, pages 71–90, 1991. 2, 16

[22] D. Lefebvre. **About numerical methods for timed and continuous Petri nets.** In *Systems, Man and Cybernetics, 2002 IEEE International Conference on*, 4, page 6 pp. vol.4, oct. 2002. 3

[23] C.R. Vazquez and M. Silva. **Stochastic Continuous Petri Nets: An Approximation of Markovian Net Models.** *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 42(3):641 –653, may 2012. 3

[24] L. Recalde and M. Silva. **Petri Nets fluidification revisited Semantics and steady state.** *APII JESA*, 35:435–449, 2001. 3

[25] R. David. **Modeling of hybrid systems using continuous and hybrid Petri nets.** In *Petri Nets and Performance Models, 1997., Proceedings of the Seventh International Workshop on*, pages 47 –58, jun 1997. 3

[26] Roberto Ross-León, Antonio Ramirez-Treviño, José Alejandro Morales, and Javier Ruiz-León. **Control of Metabolic Systems Modeled with Timed Continuous Petri Nets.** In *International Workshop on Biological Processes & Petri Nets*, pages 85–100, 2010. 3, 24, 25

[27] Jorge Júlvez and René Boel. **Modelling and controlling traffic behaviour with continuous Petri nets.** In *Proceedings of the 16th IFAC World Congress*, 2005. 3

[28] M.P. Cabasino, C. Seatzu, C. Mahulea, and M. Silva. **Fault diagnosis of manufacturing systems using continuous Petri nets.** In *Systems Man and Cybernetics (SMC), 2010 IEEE International Conference on*, pages 534 –539, oct. 2010. 3

[29] C. Mahulea, C. Seatzu, M.P. Cabasino, and M. Silva. **Fault Diagnosis of Discrete-Event Systems Using Continuous Petri Nets.** *Systems, Man and Cybernetics, Part A: Systems and Humans, IEEE Transactions on*, 42(4):970 –984, july 2012. 3

[30] Antonio Ramírez-Treviño, Elvia Ruiz-Beltrán, Jesús Arámburo-Lizárraga, and Ernesto López-Mellado. **Structural Diagnosability of DES and Design of Reduced Petri Net Diagnosers.** *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 42(2):416–429, 2012. 3

[31] C. MAHULEA, A. GIUA, L. RECALDE, C. SEATZU, AND M. SILVA. **Optimal Model Predictive Control of Timed Continuous Petri Nets.** *Automatic Control, IEEE Transactions on*, 53(7):1731 -1735, aug. 2008. 3

[32] A. GIUA, C. MAHULEA, L. RECALDE, C. SEATZU, AND M. SILVA. **Optimal control of continuous Petri nets via model predictive control.** In *Discrete Event Systems, 2006 8th International Workshop on*, pages 235 -241, july 2006. 3, 24

[33] HANIFE APAYDIN-OZKAN, JORGE JÚLVEZ, CRISTIAN MAHULEA, AND MANUEL SILVA. **An efficient heuristics for minimum time control of continuous Petri nets.** In *3rd IFAC Conference on Analysis and Design of Hybrid Systems*, 2009. 3, 24

[34] C. RENATO VÁZQUEZ, ANTONIO RAMÍREZ, LAURA RECALDE, AND MANUEL SILVA. **On Controllability of Timed Continuous Petri Nets.** In *Proceedings of the 11th international workshop on Hybrid Systems: Computation and Control*, HSCC '08, pages 528–541, Berlin, Heidelberg, 2008. Springer-Verlag. 3, 20

[35] MANUEL SILVA SUÁREZ AND CARLOS RENATO VAZQUEZ TOPETE. *Fluidization, Controllability and Control of Timed continuous Petri Nets*. PhD thesis, Zaragoza, Universidad de Zaragoza, Zaragoza, Jun 2011. Presentado: 24 06 2011. 3, 20

[36] CRISTIAN MAHULEA, LAURA RECALDE, AND MANUEL SILVA. **On Performance Monotonicity and Basic Servers Semantics of Continuous Petri Nets.** In *2006 8th International Workshop on Discrete Event Systems*, pages 345 – 351, July 2006. 3

[37] MANUEL SILVA MANUEL AND LAURA RECALDE. **Continuization of Timed Petri Nets: From Performance Evaluation to Observation and Control.** In GIANFRANCO CIARDO AND PHILIPPE DARONDEAU, editors, *Applications and Theory of Petri Nets 2005*, 3536 of *Lecture Notes in Computer Science*, pages 832–833. Springer Berlin / Heidelberg, 2005. 3

[38] J. JULVEZ, L. RECALDE, AND M. SILVA. **Deadlock-Freeness Analysis of Continuous Mono-T-Semiflow Petri Nets.** *Automatic Control, IEEE Transactions on*, 51(9):1472 -1481, sept. 2006. 3

[39] THOMAS KAILATH. *Linear Systems*. Prentice Hall, 1980. 3, 4, 5, 26, 27, 29, 76

[40] CHI-TSONG CHEN. *Linear System Theory and Design*. Saunders College Publishing, Philadelphia, PA, USA, 1984. 3, 4, 5, 26, 27, 29

[41] D. LUENBERGER. **An introduction to observers.** *Automatic Control, IEEE Transactions on*, 16(6):596 – 602, dec 1971. 3, 4, 27, 28

[42] C. MAHULEA, M.P. CABASINO, A. GIUA, AND C. SEATZU. **A state estimation problem for timed continuous Petri nets.** In *Decision and Control, 2007 46th IEEE Conference on*, pages 1770 -1775, dec. 2007. 3

[43] CRISTIAN MAHULEA, LAURA RECALDE, AND MANUEL SILVA. **Optimal Observability For Continuous Petri Nets.** page 65, 2005. 3, 4, 5, 21, 39, 40

[44] J. JÚLVEZ, E. JIMENEZ, L. RECALDE, AND M. SILVA. **On Observability and Design of Observers in Timed Continuous Petri Net Systems.** *IEEE Transactions on Automation Science and Engineering*, 5(3):532 -537, 2008. 3, 4, 21, 35, 37, 38, 40, 47, 71

[45] CRISTIAN MAHULEA, LAURA RECALDE, AND MANUEL SILVA. **Observability of continuous Petri nets with infinite server semantics.** *Nonlinear Analysis: Hybrid Systems*, 4(2):219 – 232, 2010. 3, 4, 21, 34, 35, 36

[46] CRISTIAN MAHULEA, LAURA RECALDE, AND MANUEL SILVA. **Observability of Timed Continuous Petri Nets: A Class of Hybrid Systems.** In *17th IFAC World Congress*, 2008. 3, 21, 40, 57, 63, 71

[47] DANIEL LIBERZON. *Switching in Systems and Control*. Springer, 2003. 3

[48] RENÉ VIDAL, ALESSANDRO CHIUSO, STEFANO SOATTO, AND SHANKAR SASTRY. **Observability of Linear Hybrid Systems.** In *Hybrid Systems: Computation and Control*, LNCS, pages 526–539. Springer, 2003. 3, 30, 31, 57, 66, 109

[49] DAVID GÓMEZ-GUTIÉRREZ, ANTONIO RAMÍREZ-TREVIÑO, JAVIER RUIZ-LEÓN, AND STEFANO DI GENNARO. **Observability of Switched Linear Systems: A Geometric Approach.** In *Proceedings of the 49th IEEE Conference on Decision and Control*, pages 5636 – 5642, 2010. 3, 30, 31, 57, 66, 109

[50] DAVID GOMEZ-GUTIERREZ, SERGEJ CELIKOVSKÝ, ANTONIO RAMÍREZ-TREVIÑO, JOSÉ JAVIER RUIZ-LEON, AND STEFANO DI GENNARO. **Sliding mode observer for Switched Linear Systems.** In *CASE* [74], pages 725–730. 4

[51] CRISTIAN MAHULEA, ANTONIO RAMIREZ-TREVINO, LAURA RECALDE, AND MANUEL SILVA. **Steady-State Control Reference and Token Conservation Laws in Continuous Petri Net Systems.** *Automation Science and Engineering, IEEE Transactions on*, 5(2):307 -320, April 2008. 4, 17, 18, 19, 23, 31

[52] JORGE JÚLVEZ, LAURA RECALDE, AND MANUEL SILVA. **Steady-state performance evaluation of continuous mono-T-semiflow Petri nets.** *Automatica*, 41(4):605 – 616, 2005. 4, 19, 31

[53] "JEAN-MICHEL DION, CHRISTIAN COMMAULT, AND JACOB VAN DER WOUDE". **"Generic properties and control of linear structured systems: a survey".** *"Automatica"*, "39"("7"):"1125 – 1144", "2003". 5, 36

[54] TIANHUA XU AND TAO TANG. **The modeling and Analysis of Data Communication System (DCS) in Communication Based Train Control (CBTC) with Colored Petri Nets.** In *Autonomous Decentralized Systems, 2007. ISADS '07. Eighth International Symposium on*, pages 83 -92, march 2007. 8

[55] C. JANCZURA AND I. COAT. **Modelling communication systems in a resource allocation process using coloured Petri nets.** In *Knowledge-Based Intelligent Electronic Systems, 1998. Proceedings KES '98. 1998 Second International Conference on*, 3, pages 23 -27 vol.3, apr 1998. 8

[56] MANUEL SILVA. *Las redes de Petri en la automática y la informática*. Editorial AC, 1985. 8

[57] J.M. PROTH, LIMING WANG, AND XIAOLAN XIE. **A class of Petri nets for manufacturing system integration.** *Robotics and Automation, IEEE Transactions on*, 13(3):317 -326, jun 1997. 8

[58] M. DOS SANTOS SOARES AND J. VRANCKEN. **Road Traffic Signals Modeling and Analysis with Petri nets and Linear Logic.** In *Networking, Sensing and Control, 2007 IEEE International Conference on*, pages 169 -174, april 2007. 8

[59] J. ARAMBURO-LIZARRAGA, A. RAMIREZ-TREVINO, AND E. LOPEZ-MELLADO. **Optimal communication distributed Petri net based diagnosers of Discrete event systems.** In *Electrical Engineering Computing Science and Automatic Control (CCE), 2011 8th International Conference on*, pages 1 -7, oct. 2011. 8

[60] RENÉ DAVID AND HASSANE ALLA. **Autonomous and timed continuous Petri nets.** 674:71–90, 1993. 8, 16

[61] RENÉ DAVID AND HASSANE ALLA. *Discrete, Continuous, And Hybrid Petri Nets*. Springer, 2nd edition, 2005. 8, 16

[62] J. EZPELETA, J.M. COUVREUR, AND M. SILVA. **A new technique for finding a generating family of siphons, traps and st-components. Application to colored Petri nets.** In GRZEGORZ ROZENBERG, editor, *Advances in Petri Nets 1993*, 674 of *Lecture Notes in Computer Science*, pages 126–147. Springer Berlin Heidelberg, 1993. 15

[63] GRZEGORZ ROZENBERG, editor. *Advances in Petri Nets 1993, Papers from the 12th International Conference on Applications and Theory of Petri Nets, Gjern, Denmark, June 1991*, 674 of *Lecture Notes in Computer Science*. Springer, 1993. 16

[64] RENÉ DAVID AND HASSANE ALLA. **Petri nets for modeling of dynamic systems: A survey.** *Automatica*, 30(2):175 – 202, 1994. 16

[65] MANUEL SILVA AND LAURA RECALDE. **Petri nets and integrality relaxations: A view of continuous Petri net models.** *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 32(4):314 –327, Nov. 2002. 16

[66] VAZQUEZ C., RAMIREZ ANTONIO, RECALDE LAURA, AND SILVA MANUEL. **On Controllability of Timed Continuous Petri Nets.** In *Hybrid Systems: Computation and Control*, 4981 of *Lecture Notes in Computer Science*, pages 528–541. Springer Berlin Heidelberg, 2008. 20, 21

[67] RENÉ DAVID AND HASSANE ALLA. **Reachability Graph for Autonomous Continuous Petri Nets.** In LUCA BENVENUTI, ALBERTO DE SANTIS, AND LORENZO FARINA, editors, *Positive Systems*, 294 of *Lecture Notes in Control and Information Sciences*, pages 735–736. Springer Berlin / Heidelberg, 2003. 20

[68] S. HENNEQUIN, D. LEFEBVRE, AND A. EL MOUDINI. **Fuzzy Control of Variable Speed Continuous Petri Nets.** 2, pages 1352–1356. IEEE, December 1999. 24

[69] L. WANG, C. MAHULEA, J. JULVEZ, AND M. SILVA. **Minimum-Time Control for Structurally Persistent Continuous Petri Nets.** In *CDC10: 49th IEEE Conference on Decision and Control*, Atlanta, Georgia, USA, 12/2010 2010. IEEE, IEEE. 24

[70] W. MURRAY WONHAM. *Linear Multivariable Control : A Geometric Approach*. Springer-Verlag, New York :, 2d ed. edition, 1979. 28

[71] D. GOMEZ-GUTIERREZ, G. RAMIREZ-PRADO, A. RAMIREZ-TREVIO, AND J. RUIZ-LEON. **Observability of Switched Linear Systems.** *Industrial Informatics, IEEE Transactions on*, 6(2):127 –135, may 2010. 30

[72] ENRIQUE AGUAYO-LARA, ANTONIO RAMÍREZ-TREVIÑO, AND JOSÉ JAVIER RUIZ-LEON. **Invariant subspaces and sensor placement for observability in Continuous Timed Petri Nets.** In *Int. Conference on Automation Science and Engineering CASE 2011* [74], pages 607–612. 68, 70

[73] CARLOS-RENATO VAZQUEZ AND MANUEL SILVA. **Piecewise-linear constrained control for timed continuous Petri nets.** In *Decision and Control, 2009 held jointly with the 2009 28th Chinese Control Conference. CDC/CCC 2009. Proceedings of the 48th IEEE Conference on*, pages 5714 –5720, December 2009. 76

[74] *IEEE Conference on Automation Science and Engineering, CASE 2011, Trieste, Italy, Aug. 24-27, 2011*. IEEE, 2011. 112, 113

CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS DEL I.P.N.
UNIDAD GUADALAJARA

El Jurado designado por la Unidad Guadalajara del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional aprobó la tesis

Ubicación de sensores para observabilidad en redes de Petri fluidificadas/Sensor placement for observability in fluidified Petri nets

del (la) C.

Enrique Javier AGUAYO LARA

el día 13 de Diciembre de 2012.

[illegible]
Dr. Luis Ernesto López Mellado
Investigador CINVESTAV 3C
CINVESTAV Unidad Guadalajara

[illegible]
Dr. José Luis Alejandro Naredo Villagrán
Investigador CINVESTAV 3C
CINVESTAV Unidad Guadalajara

A. Ramírez
Dr. Antonio Ramírez Treviño
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara

[illegible]
Dr. Andrés Méndez Vázquez
Investigador CINVESTAV Guadalajara 2C
CINVESTAV Guadalajara

[illegible]
Dr. Carlos Renato Vazquez Topete
Profesor Investigador
Centro Universitario de los Valles, UdG