CT- 891-551
ISS-168-CD
SIOC :NOG

Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional
Unidad Guadalajara

MINERIA DE PROCESOS DE EVENTOS DISCRETOS. SÍNTESIS DE LAS COMPONENTES OBSERVABLES DE REDES DE PETRI INTERPRETRADAS

Tesis que presenta:

Edelma Rodríguez Pérez

para obtener el grado de:
Maestra en Ciencias

en la especialidad de:
Ingeniería Eléctrica

Director de Tesis
Dr. Luis Ernesto López Mellado

CINVESTAV del IPN Unidad Guadalajara, Guadalajara, Jalisco, Diciembre de 2014

MINERIA DE PROCESOS DE EVENTOS DISCRETOS. SÍNTESIS DE LAS COMPONENTES OBSERVABLES DE REDES DE PETRI INTERPRETADAS

Tesis de Maestría en Ciencias
Ingeniería Eléctrica

Por:

Edelma Rodríguez Pérez

Ingeniero en Computación

Universidad de Guadalajara 2005-2010

Director de Tesis

Dr. Luis Ernesto López Mellado

CINVESTAV del IPN Unidad Guadalajara, Diciembre de 2014.

MINERIA DE PROCESOS DE EVENTOS DISCRETOS. SÍNTESIS DE LAS COMPONENTES OBSERVABLES DE REDES DE PETRI INTERPRETADAS

Resumen

Esta tesis trata sobre la identificación de procesos de eventos discretos a partir de observaciones de la evolución de entradas y salidas. Se presenta un método eficiente que permite construir de manera automatizada un modelo en redes de Petri interpretadas (RPI).

Siguiendo una estrategia de dos fases, en el cual se obtiene primero la parte observable de la RPI y después se infiere la parte no observable, el método propuesto se enfoca en la primera fase. Para ello se definieron conceptos básicos y se concibieron los algoritmos que constituyen cada una de las etapas del método. A partir de una secuencia w de vectores de entrada-salida, se obtiene un conjunto de eventos de entrada compuestos, los cuales determinan las transiciones del modelo. Con estos eventos y los eventos de salida se crea la RPI observable. Posteriormente se determina la secuencia de transiciones correspondiente a w.

Este método, el cual es más simple que otro similar previamente propuesto, fue implementado como una herramienta de software; ésta fue probada con secuencias de entrada-salida obtenidas experimentalmente de procesos de eventos discretos reales.

II

DISCRETE EVENT PROCESSES MINING. SYNTHESIS OF THE OBSERVABLE COMPONENTS OF INTERPRETED PETRI NETS

Abstract

This thesis deals with identification of discrete event processes from the observation of input-output evolution. An efficient method allowing building an interpreted Petri net (IPN) model, in an automated way, is presented.

Based on a two phase strategy, in which an observable part is first obtained, and then the non observable part is inferred, the proposed method focuses on the first phase. For this purpose basic concepts have been defined and several algorithms have been conceived for the metod's steps. From a single sequence w of input-output vectors, a set of composed input events are computed, which determine the transitions of the model. Using these events and the outputs events the observable IPN is created. Later, the transitions sequence corresponding to w is computed.

The method, which is simpler than other similar previously proposed, has been implemented as a software tool; it has been tested on several input-output sequences experimentally obtained from real discrete event processes.

IV

Agradecimientos

A mi madre y mi hermano por su apoyo.

A mi abuela por ser ejemplo en mi vida.

A Tonatiuh Tapia y Ana Paula Estrada por compartir su conocimiento sobre el tema.

A Ory Medina por su compresión, consejos y amor.

Al Dr. Luis Ernesto López-Mellado por su paciencia, asesoría, apoyo y por todo lo que aprendí trabajando con él.

A CONACYT por la beca recibida durante el programa de maestría.

VI

Índice

Introducción

En la actualidad muchos procesos dinámicos automatizados que están operación no tienen una documentación fidedigna sobre su funcionamiento; esto es porque las actualizaciones en los controladores no ha sido documentadas en el mejor de los casos, o bien porque no se cuenta con dichas especificaciones.

La identificación de sistemas trata de aproximar el comportamiento entrada-salida de una planta desconocida con un modelo apropiado. Algunos métodos y técnicas inteligentes como la lógica difusa y las redes neuronales entre otras, han sido usados para crear modelos de plantas continuas, y en consecuencia aplicados para resolver un gran número de problemas de identificación. En el caso de los Procesos de Eventos Discretos el problema es similar; el modelo matemático para representar comportamiento es generalmente un autómata finito o una red de Petri

En esta tesis nos interesamos en el problema de identificación de procesos de eventos discretos, el cual ha sido abordado en la literatura con diferentes nombres: inferencia gramatical o aprendizaje de lenguajes, identificación de procesos y minería de procesos. En general este problema se puede plantear como un problema de ingeniería de reversa. Los enfoques principales de las técnicas de identificación propuestas se resumen a continuación.

En el trabajo de [Gold, 1967] se aborda por primera vez el problema de identificar determinadas características sobre una secuencia de elementos pertenecientes a un alfabeto haciendo una inferencia gramatical, en donde un autómata finito se construye a partir de una muestra correcta de palabras que acepta. En el trabajo de [Angluin, 1987] por otra parte muestra un método de aprendizaje que obtiene un autómata finito que reproduce las palabras que acepta un leguaje.

En [Meda, 2002a] se describe un método para construir un modelo de RPI de manera incremental con una sola secuencia de vectores de salida. El SED considerado para identificar debe de detectarse su comportamiento por las señales de salida que posee. La aplicación de este método para la identificación de una secuencia de E/S daría lugar a modelos en los que mismos cambios de salida provocados por diferentes evoluciones en la entrada no se diferencian.

En el enfoque que se usa programación lineal entera, varios trabajos se han propuesto en variantes del problema. En [Cabasino, 2009] donde se obtiene un Sistema de red de Petri a partir del conocimiento de su lenguaje, es decir, el conjunto de secuencias de transición que puede ser disparado desde un marcado inicial. En [Cabasino, 2013], difieren del enfoque de la caja negra que es abordado en esta tesis porque se consideran transiciones que son desconocidas, es decir, la única información disponible acerca del sistema es la entrada y salida de señales, así como su evolución. Algunas de sus hipótesis no están bien adaptadas para un SED complejo, en particular con los supuestos de que se observa el lenguaje total y la existencia de contra ejemplos. Adicionalmente, la complejidad computacional de estos métodos es alta.

El método presentado en [Klein, 2005a] obtiene modelos de autómatas que presentan un conjunto de secuencias de E/S cíclicas. Este método también considera sistemas automatizados. Sin embargo, en los modelos que se obtienen, la concurrencia no puede ser expresada de manera explícita. Una extensión presentada de este trabajo en [Roth, 2010], permite dividir el Sistema en partes concurrentes. Incluso si subsistemas modelados representan paralelismo, el método se adapta firmemente para los propósitos detección de fallas.

En [Dotoli, 2008] se observa una secuencia de eventos, así como los correspondientes símbolos de salida de un SED para producir un modelo de RPI, en el que la secuencia y los vectores de entrada son reproducidos. Esta metodología requiere el conocimiento de una lista de eventos que no esta disponible en el contexto del problema de identificación de caja negra tratado en este trabajo. Una alternativa a esta falta de lista de eventos podría ser la consideración de todos los cambios de entrada que se han observado. Así, se construyeron modelos con varias trayectorias que describen los cambios de entrada, en la que no se observarían explícitamente algunas relaciones de entrada-salida.

En un trabajo reciente, se ha abordado el problema de identificación de caja negra, en donde se tiene como observaciones externas las entradas y salidas del proceso [Estrada, 2013]. En este enfoque se presenta un método para construir modelos de redes de Petri Interpretadas (RPI) a partir de observaciones de las señales que intercambian un proceso y un controlador. El método consta de dos etapas: la primer construye la parte observable del modelo y la segunda construye la parte no observable a partir de una secuencia de transiciones generada por una secuencia de entrada salida, en este trabajo se trata de mejorar la primera parte del trabajo antes mencionado.

En esta tesis se presenta una técnica alternativa más simple para realizar la primera etapa de la identificación de caja negra. El método propuesto en este trabajo permite construir de manera automatizada un modelo en redes de Petri Interpretadas (RPI). La entrada del método es una secuencia w de vectores E/S, la cual es procesada por un conjunto de algoritmos descritos en el capítulo 3, para obtener una secuencia reducida de eventos característicos que serán agrupados para formar un conjunto de eventos compuestos, los cuales determinan las transiciones del modelo observable. Con los eventos compuestos y los eventos de salida correspondientes se crea la red de Petri que representa el modelo observable. Posteriormente se determina la secuencia de transiciones que corresponde a w. Adicionalmente se ha implementado una herramienta de software a partir de los algoritmos del método, la cual fue probada con secuencias de E/S, algunas de ellas obtenidas de procesos de eventos discretos reales.

El presente documento de tesis está organizado de la siguiente manera.

- Capítulo 1. Se resumen los diferentes métodos de identificación de la literatura que fueron estudiados.
- Capítulo 2. Se presentan conceptos básicos, la definición del problema y se resume el trabajo propuesto en [Estrada, 2013] para contextualizar la contribución de esta tesis.
- Capítulo 3. Se presenta un método para la identificación de procesos de eventos discretos.
- Capítulo 4 se describe de manera general la herramienta de software realizada para probar el algoritmo propuesto y su aplicación a casos de estudio.

- **Finalmente, se enuncian algunas conclusiones y perspectivas de este trabajo.**

Capítulo 1
Minería de Procesos de Eventos Discretos

Resumen. En este capítulo se analizan las diversas técnicas de minería de procesos de eventos discretos más representativas, resaltando los aspectos más importantes de los enfoques que abordan este problema. Finalmente se plantea una discusión resultado de la comparación entre estos enfoques.

El problema de obtener un modelo matemático a partir de observaciones del comportamiento de un proceso de eventos discretos ha sido abordado desde diferentes enfoques, donde ha recibido los nombres de inferencia gramatical, identificación y minería de procesos. A continuación presentamos un panorama breve de los enfoques y técnicas más representativas que tratan sobre este problema.

1.1. Métodos propuestos en la Teoría de Lenguajes

Los primeros trabajos orientados a resolver el problema de minería de procesos, fueron presentados en el área de Teoría de Lenguajes, donde se pretende caracterizar el lenguaje a partir de una muestra de palabras del mismo, por medio del aprendizaje o haciendo inferencias sobre él.

La inferencia gramatical es un problema inductivo en el que el dominio es un lenguaje. El aprendizaje consiste en, dado un número finito de palabras, identificar una gramática correcta para ellas. En [Gold, 1967] se propuso una técnica de aprendizaje que procesa muestras las cuales contienen palabras incluidas en el lenguaje a identificar. A partir de estas muestras se hace una conjetura y se compara con el lenguaje a identificar, el procedimiento termina una vez que ambos lenguajes son iguales.

Basándose en los resultados anteriores, Gold (1972) desarrolla un método esencialmente no lineal y no paramétrico para el diseño de experimentos que determinarán directamente una representación en un espacio de estados para la identificación de sistemas de tipo caja negra (black-box). El enfoque se basa en la descripción del espacio de estados de acuerdo con la Teoría de Sistema Abstracto y la Teoría de Autómatas, que se construye usando un conjunto de funciones entrada-salida de los estados alcanzables. El resultado del método es presentado como una máquina de Moore o como una máquina de Mealy.

En [Angluin, 1987] se aborda el problema de identificar un lenguaje regular desconocido a partir de la pertenencia o ausencia de sus miembros, en el cual con la ayuda de "Minimally Adequate Teacher" es posible responder a las consultas de pertenecía al lenguaje o en caso contrario ofrecer un contraejemplo. El método representa el lenguaje por medio de un autómata finito determinista.

El método presentado en [Veelenturf, 1978] procesa simultáneamente una muestra de secuencias de palabras para producir paso a paso una máquina de Mealy, de tal manera que el comportamiento de cada nueva máquina incluye el comportamiento de la anterior. En cada paso, se analiza la máquina anterior y se añaden nuevas transiciones y estados posibles.

Se ha demostrado en [Takada, 1998] que el problema de inferencia gramatical, incluso para idiomas lineales, se puede reducir en tiempo polinomial para la inferencia de lenguajes regulares.

Otros trabajos relacionados utilizan el formalismo de red de Petri. En [Hiraishi, 1992] se presenta un algoritmo para la síntesis de los modelos de redes de Petri. El algoritmo propuesto tiene dos fases: en la primera fase, el lenguaje del sistema se identifica bajo la forma de un autómata finito determinista; en la segunda fase, se genera una red de Petri que acepta el mismo lenguaje que el autómata obtenido en la primera fase.

1.2. Métodos para la identificación de Sistemas de Eventos Discretos

Durante la última década la comunidad científica ha propuesto métodos de identificación de sistemas de eventos discretos (basados en redes de Petri o autómatas) para obtener modelos aproximados de los sistemas cuyo comportamiento es desconocido o poco conocido.

1.2.1. Enfoques de identificación

El enfoque de identificación de Sistemas de Eventos Discretos (SED) propone calcular un modelo de red de Petri Interpretada (RPI) que describe el comportamiento del sistema desconocido basándose en las señales de entrada y salida, eventos, mensajes, tareas, secuencias de operaciones, etc., asegurando la reproducción del comportamiento observado.

En la actualidad, existen varios enfoques para la identificación de SED automatizados. El enfoque de síntesis incremental presentado por Meda [Meda, 2005], se basa en la construcción de un modelo de Red de Petri Interpretada, a partir de la observación de las señales de salida. Se genera progresivamente una secuencia de modelos de tal manera que el modelo actual se acerca cada vez al modelo real del sistema. Algunos de los supuestos que se consideran en este trabajo son: el sistema a ser identificado puede ser descrito por una RPI viva, 1-acotada y cíclica, el mismo cambio en las salidas no puede ser provocado por diferentes transiciones, y las transiciones no se disparan simultáneamente.

Diferentes algoritmos para la construcción de Redes de Petri Interpretadas que han sido propuestos permiten la identificación de SED concurrentes a partir de las secuencias de salida, a pesar de ser métodos eficientes, los modelos que se obtienen pueden representar más comportamientos que los descritos en el sistema.

En [Klein, 2005] se realiza la construcción de un autómata finito no determinista a partir de una muestra de secuencias entrada-salida del sistema. El autómata obtenido genera exactamente las secuencias de entrada-salida que fueron observadas. El método de [Roth, 2009] usa este método para la detección de faltas y extiende el método para la detección de faltas distribuidas en [Roth, 2012].

1.2.2. Método de Identificación por medio de la Programación Lineal Entera

Los trabajos presentados por Dotoli en [Dotoli, 2006a], [Dotoli, 2006b], [Dotoli, 2007], [Dotoli, 2008] se presentan métodos que extienden los trabajos de [Giua, 2005] en los cuales además de obtener una secuencia de eventos, la secuencia de salida del SED se utiliza para hacer la inferencia de un modelo de Red de Petri.

El método supone que todos los eventos SED se pueden detectar y distinguir. El algoritmo recibe una secuencia de eventos con sus vectores de salida correspondientes. Además, se necesita conocimiento del número de lugares no observables presentes en la red a identificar, el algoritmo devuelve una red de Petri con lugares observables que representan los actuadores del sistema, los lugares no observables y transiciones marcadas que representan la ejecución de la secuencia de eventos observados; regresa un 0 (cero) cuando no existe solución posible o el número de lugares no observables.

La estrategia del algoritmo es generar un problema de programación lineal entera de manera similar a la del método de [Cabasino, 2009]: un sistema de red es una solución del

problema de identificación si y sólo si satisface el siguiente conjunto de restricciones algebraicas lineales:

$$\xi(w,Y,\lambda,T,m)=\begin{cases} Pre, Post \in N^{m\times n} \\ M_i \in N^m \quad con \quad i=0,\ldots,h \\ Post^T \vec{1}_{m\times 1} + Pre^T \vec{1}_{m\times 1} \geq \vec{1}_{n\times 1} \\ Post\vec{1}_{n\times 1} + Pre\vec{1}_{n\times 1} \geq \vec{1}_{m\times 1} \\ \forall t_{\beta i}^{\alpha i} \in \sigma \quad con \quad \lambda(\sigma)=w, Pre\vec{t}_{\beta i}^{\alpha i} \leq M_{i-1} \\ \forall t_{\beta i}^{\alpha i} \in \sigma \quad con \quad \lambda(\sigma)=w, (Post-Pre)\vec{t}_{\beta i}^{\alpha i} = M_i - M_{i-1} \end{cases}$$

Se pueden agregar algunas restricciones si las propiedades estructurales son conocidas en el modelo de Red de Petri a identificar. Por ejemplo, si no existe un lugar sin transición sucesora se puede agregar: $Pre \cdot \vec{1}_{n\times 1} \geq \vec{1}_{m\times 1}$. Si no hay transición sin lugar sucesor: $Post^T \cdot \vec{1}_{m\times 1} \geq \vec{1}_{n\times 1}$. Si no hay transiciones fuente: $Pre^T \cdot \vec{1}_{m\times 1} \geq \vec{1}_{n\times 1}$.

Se utiliza un indicador del tamaño de Red de Petri como una función lineal.

$$\phi(Pre, Post, M_0) = \vec{a}^T Pre\vec{b} + \vec{c}^T Post\vec{d} + \vec{e}M_0$$

Se presenta el algoritmo que resuelve el problema de identificación mencionado anteriormente. Se da una mejor explicación de la solución y del algoritmo de identificación en [Dotoli, 2008].

Algoritmo 1.1

1. Inicialización de las variables. Se obtiene el primer vector de salida y se inicializa el conjunto de etiquetas, de transiciones y de vectores de salida. Estos vectores se van actualizando a lo largo de la ejecución.
2. Se obtiene un nuevo evento y su vector de salida correspondiente.
3. Se le asocia una transición a cada evento.
   3.1. Si se observa por primera vez el evento. Una nueva transición es creada se asocia al evento y el cambio de marcado observado.
   3.2. Si el evento ya había ocurrido previamente.
      3.2.1. Una nueva transición se debe asociar con el evento (si no hay cambio en el marcado asociado a alguna transición).
      3.2.2. Una transición de disparo se asocia al evento.
4. Se resuelve el problema de Programación Lineal Entera (PLE), $min\phi(Pre_w, Post_w, M_{0w})$s.t. $\xi(w,Y,\lambda_w,T_w,m')$, tantas veces como se necesite, a partir de un número m de lugares observables y se va incrementando, hasta que se encuentre una solución o hasta que m sea igual al límite superior del número de lugares.
5. 5. Regresar al punto 2.

Ejemplo 1.1. Se considera un SED con $y \in \mathbb{N}^5$ y $\bar{m} = q = 5$. Se asume que la salida inicial es $y_0 = [00102]^T$ y la secuencia observada es $w = e_{\alpha_1}, e_{\alpha_2}, e_{\alpha_3}, e_{\alpha_4} = e_1, e_2, e_2, e_1$ con la salida correspondiente $y_1 = [40101]^T$ $y_2 = [31001]^T, y_3 = [01011]^T$ and $y_4 =$

$[00102]^T$, ejemplo tomado de [Dotoli, 2008]. Cada vez que aparece un evento se aplica el algoritmo de identificación, añadiendo las restricciones para obtener una Red de Petri sin transiciones ni lugares sin sucesores. Sin embargo, la solución se obtiene hasta que el último evento es procesado. La PLE a resolver es:

Minimizar

$$[1 \quad 1 \quad 1 \quad 1 \quad 1](Pre + Post)\begin{bmatrix}1\\1\\1\\1\end{bmatrix} + [1 \quad 1 \quad 1 \quad 1 \quad 1]M_0$$

Depende de:

1) $Pre, Post \in N^{m \times n}$

$$Pre = \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}, Post = \begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix}$$

2) $M_i \in N^m$ con $i = 0, \ldots, h$

$$M_0 = \begin{bmatrix} m_{0_1} \\ m_{0_2} \\ m_{0_3} \\ m_{0_4} \\ m_{0_5} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 2 \end{bmatrix}, M_1 = \begin{bmatrix} m_{1_1} \\ m_{1_2} \\ m_{1_3} \\ m_{1_4} \\ m_{1_5} \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix}, M_2 = \begin{bmatrix} m_{2_1} \\ m_{2_2} \\ m_{2_3} \\ m_{2_4} \\ m_{2_5} \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}, M_3 = \begin{bmatrix} m_{3_1} \\ m_{3_2} \\ m_{3_3} \\ m_{3_4} \\ m_{3_5} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}, M_4 = \begin{bmatrix} m_{4_1} \\ m_{4_2} \\ m_{4_3} \\ m_{4_4} \\ m_{4_5} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 2 \end{bmatrix}$$

3) $Post^T \vec{1}_{m \times 1} + Pre^T \vec{1}_{m \times 1} \geq \vec{1}_{n \times 1}$

$$\begin{bmatrix} post_{11} & post_{12} & post_{13} & post_{14} & post_{15} \\ post_{21} & post_{22} & post_{23} & post_{24} & post_{25} \\ post_{31} & post_{32} & post_{33} & post_{34} & post_{35} \\ post_{41} & post_{42} & post_{43} & post_{44} & post_{45} \end{bmatrix}\begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} + \begin{bmatrix} pre_{11} & pre_{12} & pre_{13} & pre_{14} & pre_{15} \\ pre_{21} & pre_{22} & pre_{23} & pre_{24} & pre_{25} \\ pre_{31} & pre_{32} & pre_{33} & pre_{34} & pre_{35} \\ pre_{41} & pre_{42} & pre_{43} & pre_{44} & pre_{45} \end{bmatrix}\begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix} \geq \begin{bmatrix}1\\1\\1\\1\end{bmatrix}$$

4) $Post\vec{1}_{n \times 1} + Pre\vec{1}_{n \times 1} \geq \vec{1}_{m \times 1}$

$$\begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} + \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\begin{bmatrix}1\\1\\1\\1\end{bmatrix} \geq \begin{bmatrix}1\\1\\1\\1\\1\end{bmatrix}$$

5) $\forall t_{\beta_i}^{\alpha_i} \in \sigma$ con $\lambda(\sigma) = w$, $Pre\vec{t}_{\beta_i}^{\alpha_i} \leq M_{i-1}$

$$\begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\begin{bmatrix} t_{\beta_1 1}^{\alpha_1} \\ t_{\beta_1 2}^{\alpha_1} \\ t_{\beta_1 3}^{\alpha_1} \\ t_{\beta_1 4}^{\alpha_1} \end{bmatrix} \leq \begin{bmatrix}0\\0\\1\\0\\2\end{bmatrix}, \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\begin{bmatrix} t_{\beta_2 1}^{\alpha_2} \\ t_{\beta_2 2}^{\alpha_2} \\ t_{\beta_2 3}^{\alpha_2} \\ t_{\beta_2 4}^{\alpha_2} \end{bmatrix} \leq \begin{bmatrix}4\\0\\1\\0\\1\end{bmatrix},$$

$$\begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\begin{bmatrix} t_{\beta_3 1}^{\alpha_3} \\ t_{\beta_3 2}^{\alpha_3} \\ t_{\beta_3 3}^{\alpha_3} \\ t_{\beta_3 4}^{\alpha_3} \end{bmatrix} \leq \begin{bmatrix}3\\1\\0\\0\\1\end{bmatrix}, \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\begin{bmatrix} t_{\beta_4 1}^{\alpha_4} \\ t_{\beta_4 2}^{\alpha_4} \\ t_{\beta_4 3}^{\alpha_4} \\ t_{\beta_4 4}^{\alpha_4} \end{bmatrix} \leq \begin{bmatrix}0\\1\\0\\1\\1\end{bmatrix}$$

6) $\forall t_{\beta_i}^{\alpha_i} \in \sigma$ con $\lambda(\sigma) = w$, $(Post - Pre)\vec{t}_{\beta_i}^{\alpha_i} = M_i - M_{i-1}$

$$\left(\begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix} - \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\right) \begin{bmatrix} t_{\beta_1 1}^{\alpha_1} \\ t_{\beta_1 2}^{\alpha_1} \\ t_{\beta_1 3}^{\alpha_1} \\ t_{\beta_1 4}^{\alpha_1} \end{bmatrix} = \begin{bmatrix} 4 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 2 \end{bmatrix},$$

$$\left(\begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix} - \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\right) \begin{bmatrix} t_{\beta_2 1}^{\alpha_2} \\ t_{\beta_2 2}^{\alpha_2} \\ t_{\beta_2 3}^{\alpha_2} \\ t_{\beta_2 4}^{\alpha_2} \end{bmatrix} = \begin{bmatrix} 3 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} - \begin{bmatrix} 4 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix},$$

$$\left(\begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix} - \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\right) \begin{bmatrix} t_{\beta_3 1}^{\alpha_3} \\ t_{\beta_3 2}^{\alpha_3} \\ t_{\beta_3 3}^{\alpha_3} \\ t_{\beta_3 4}^{\alpha_3} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix} - \begin{bmatrix} 3 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix},$$

$$\left(\begin{bmatrix} post_{11} & post_{21} & post_{31} & post_{41} \\ post_{12} & post_{22} & post_{32} & post_{42} \\ post_{13} & post_{23} & post_{33} & post_{43} \\ post_{14} & post_{24} & post_{34} & post_{44} \\ post_{15} & post_{25} & post_{35} & post_{45} \end{bmatrix} - \begin{bmatrix} pre_{11} & pre_{21} & pre_{31} & pre_{41} \\ pre_{12} & pre_{22} & pre_{32} & pre_{42} \\ pre_{13} & pre_{23} & pre_{33} & pre_{43} \\ pre_{14} & pre_{24} & pre_{34} & pre_{44} \\ pre_{15} & pre_{25} & pre_{35} & pre_{45} \end{bmatrix}\right) \begin{bmatrix} t_{\beta_4 1}^{\alpha_4} \\ t_{\beta_4 2}^{\alpha_4} \\ t_{\beta_4 3}^{\alpha_4} \\ t_{\beta_4 4}^{\alpha_4} \end{bmatrix} \leq \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{bmatrix}.$$

La Red de Petri Interpretada que se obtiene es la de la Figura 1.1.

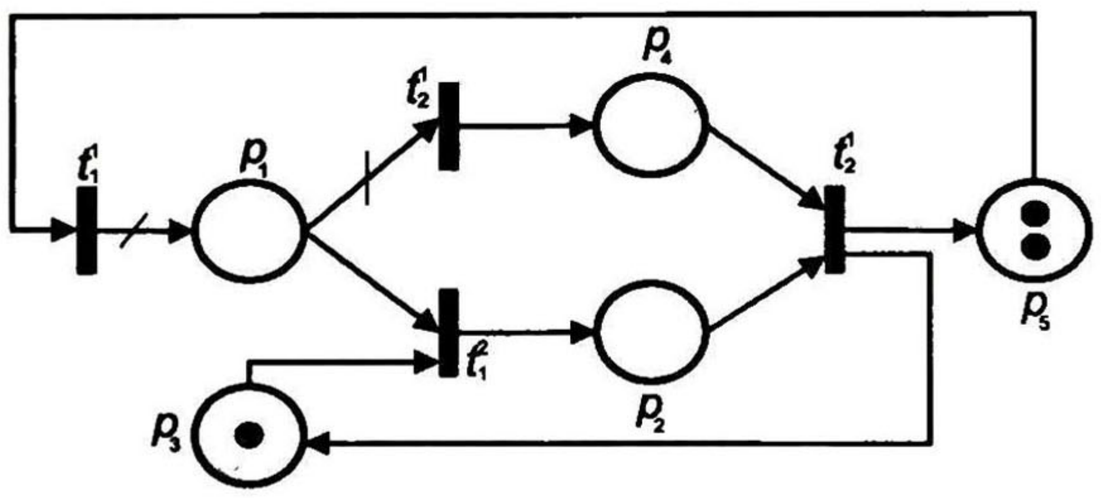


Figura 1.1 Solución al problema de identificación del ejemplo 1.1

En el peor de los casos el número de incógnitas es lineal con el número de lugares, de las transiciones y con la longitud h de la secuencia de disparo. En los casos que se examinan por los autores, se puede obtiene la solución en un corto período de tiempo de ejecución. Sin embargo, cuando se hace la ejecución en línea, lo dinámico del SED tiende a ser más lento con respecto al tiempo que se resuelve el problema de PLE en cada ocurrencia.

1.2.3. Identificación de Red de Petri Interpretada Paramétrica

El método usado en [Estrada, 2009] construye modelos de Red de Petri Interpretados para las observaciones de las señales de entrada y salida de SED's. Consiste en varias etapas que construyen sistemáticamente un modelo de RPI a partir de las secuencias de entrada y salida representando el comportamiento externo parcialmente observable del SED. Se presenta una

herramienta de software que implementa algoritmos desarrollados en el método; se procesa un conjunto de secuencias de vectores de entrada-salida produciendo el modelo de red de Petri interpretada.

Los datos de entrada para realizar la identificación son un conjunto de palabras de entrada-salida que pueden incluir un comportamiento cíclico. Basado en la precisión del parámetro κ, $\Phi = \{\varphi_1, \varphi_2, \ldots\}$ el objetivo del proceso de identificación es obtener un modelo seguro de RPI (Q, M_0) tal que $\pounds^{\kappa}_{out}(Q, M_0) = \mathcal{L}^{\kappa}(S)$. El parámetro κ se usa para ajustar la precisión del modelo a identificar.

A partir del vector de palabras de entrada-salida se obtiene la secuencia de eventos la cual se transforma en de sub-secuencias de eventos de tamaño κ, cada sub-secuencia está asociada a una transición de la red de Petri, la cual describe la relación de precedencia entre las sub-secuencia de eventos; esta red de Petri está formada por los lugares no observables. Se realiza un procedimiento de transformación basado en las transformaciones de concurrencia. Finalmente, los cambios en las salidas provocados por eventos son descritos por los cambios en los marcados en los lugares observables y se le asocian las transiciones pertinentes de la red de Petri; los cambios en las entradas están asociados a cada transición. Los lugares implícitos son eliminados.

Se puede resumir las etapas del método para la identificación del modelo de RPI como:

Algoritmo 1.2

Entradas: Un SED y un parámetro κ

Salidas: (Q, M_0) un modelo RPI

---

1. Obtener los símbolos de entrada y las secuencias cíclicas de los vectores de salida observados.
2. Procesar la secuencia de eventos de los vectores observados.
3. Para cada secuencia de eventos, crear la traza de tamaño κ.
4. Crear el modelo RPI del comportamiento no observable y simplificarlo.
5. Complementar el modelo RPI añadiendo el comportamiento observable y eliminar los lugares implícitos.

1.2.4. Identificación entrada-salida

El método desarrollado en [Estrada, 2010] considera un sistema de control de lazo cerrado (Figura 1.2); consiste en una planta y su controlador industrial (en la mayoría de los casos se trata de un Controlador Lógico Programable). El comportamiento de tales sistemas se puede observar tomando muestras de las señales que interactúan entre el controlador y la planta.

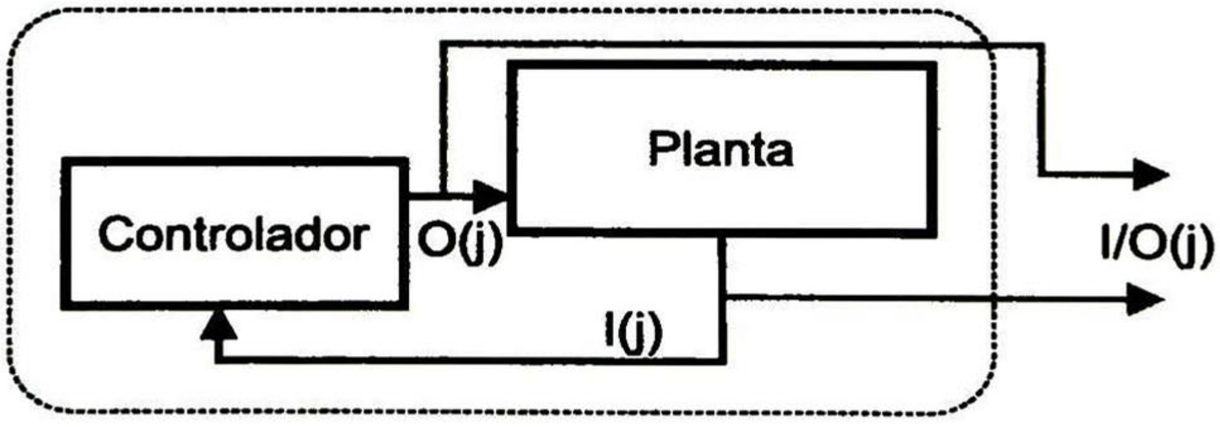


Figura 1.2 SED de lazo cerrado controlador-planta

La complejidad puede incrementar por algunas situaciones entre la interacción del controlador y la planta. Sin embargo, al identificar el SED se debe de tomar en cuenta que:

- El cambio en la entrada no siempre provoca un cambio en la salida. En la práctica, pocos cambios en las entradas provocan cambios en las salidas.
- No se observan eventos de entrada-salida (E/S) simultáneos.
- Cuando la salida que cambia es provocada por un cambio en las entradas, esta relación causal no es necesariamente capturada simultáneamente.

Ciclo de funcionamiento PLC (Controlador Lógico Programable)

Los PLC son máquinas secuenciales que ejecutan las tareas indicadas, generando señales de mando a partir de las señales de entrada leídas de la planta: cuando se detectan cambios en las señales, el sistema reacciona según las señales de entrada hasta obtener las salidas correspondientes. Esta secuencia se ejecuta continuamente para conseguir el control actualizado del proceso.

El método se puede dividir en tres fases: lectura de señales de entrada, procesamiento de las señales de entrada para obtener las señales de salida correspondiente (ejecución del programa) y la escritura de las señales.

Al finalizar la fase de ejecución del programa los valores actuales de las entradas y salidas (E/S) se envían al PLC para su tratamiento posterior con el algoritmo de identificación.

La estrategia general del método de identificación consiste en varias etapas que construyen sistemáticamente una Red de Petri Interpretada segura, representando exactamente el lenguaje de salida de tamaño $\kappa+1$ del SED.

Algoritmo 1.3

Entradas: Un sistema de eventos discreto y un parámetro κ.

Salidas: Un modelo de Red de Petri Interpretada

---

1. Obtener la secuencia cíclica w_i de los vectores E/S observados.
2. Procesar la secuencia de vectores evento τ_i y la función de entrada λ' de los vectores observados.
3. Para cada secuencia de vectores de eventos τ_i, se crea la traza τ_i^{κ} de tamaño κ.
4. Crear el modelo de RPI del comportamiento no observable y simplificar la red.
5. Completar la RPI añadiendo el comportamiento observado eliminando lugares implícitos.

Ejemplo 1.2. Se considera un SED con n = 4 señales de salida, $\Phi = \{A, B, C, D\}$, y $m = 3$ señales de salida $\Sigma = \{a, b, c\}$. Tres secuencias de E/S fueron observadas; las entradas del vector corresponden a la siguiente distribución $[a, b, c|A, B, C, D]^T$.

$$w_1 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\1\\0\\0\end{bmatrix} \right) \Bigg| w_2 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\1\\0\\0\\1\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\1\\0\\0\\1\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\1\end{bmatrix} \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \right) \Bigg| w_3 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\1\\1\\0\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\1\end{bmatrix} \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \right)$$

Dónde:

$$w_1 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \overset{e_1}{\rightarrow} \begin{bmatrix}1\\0\\0\\-1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \overset{e_2}{\rightarrow} \begin{bmatrix}0\\0\\0\\0\\1\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\1\\0\\0\end{bmatrix} \right), \tau_1 = e_1 e_2 \qquad \lambda'(e_1) = a\_1, \lambda'(e_2) = \varepsilon$$

$$w_2 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \overset{e_1}{\rightarrow} \begin{bmatrix}1\\0\\0\\-1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \overset{e_3}{\rightarrow} \begin{bmatrix}0\\1\\1\\0\\0\\1\\0\end{bmatrix} \begin{bmatrix}1\\1\\0\\0\\1\\0\\0\end{bmatrix} \overset{e_4}{\rightarrow} \begin{bmatrix}0\\-1\\0\\0\\0\\0\\1\end{bmatrix} \begin{bmatrix}1\\0\\1\\0\\0\\1\\1\end{bmatrix} \overset{e_5}{\rightarrow} \begin{bmatrix}0\\0\\-1\\0\\0\\-1\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\1\end{bmatrix} \overset{e_6}{\rightarrow} \begin{bmatrix}-1\\0\\0\\1\\0\\0\\-1\end{bmatrix} \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \right), \tau_2 = e_1 e_3 e_4 e_5 e_6$$

$$\lambda'(e_3) = b\_1\, c\_1\,, \lambda'(e_4) = b\_0, \lambda'(e_5) = c\_0, \lambda'(e_6) = a\_0$$

$$w_3 = \left(\begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \overset{e_1}{\rightarrow} \begin{bmatrix}1\\0\\0\\-1\\0\\0\\0\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\end{bmatrix} \overset{e_3}{\rightarrow} \begin{bmatrix}0\\1\\1\\0\\0\\1\\0\end{bmatrix} \begin{bmatrix}1\\1\\0\\0\\1\\0\\0\end{bmatrix} \overset{e_5}{\rightarrow} \begin{bmatrix}0\\0\\-1\\0\\0\\-1\\0\end{bmatrix} \begin{bmatrix}1\\1\\0\\0\\0\\0\\0\end{bmatrix} \overset{e_4}{\rightarrow} \begin{bmatrix}0\\-1\\0\\0\\0\\0\\1\end{bmatrix} \begin{bmatrix}1\\0\\0\\0\\0\\0\\1\end{bmatrix} \overset{e_6}{\rightarrow} \begin{bmatrix}-1\\0\\0\\1\\0\\0\\-1\end{bmatrix} \begin{bmatrix}0\\0\\0\\1\\0\\0\\0\end{bmatrix} \right), \tau_2 = e_1 e_3 e_5 e_4 e_6$$

Las secuencias de trazas de evento con $\kappa = 2$ son:

$\tau_1^2 = \varepsilon e_1, e_1 e_2$ para τ_1

$\tau_2^2 = \varepsilon e_1, e_1 e_3, e_3 e_4, e_4 e_5, e_5 e_6$ para τ_2

$\tau_3^2 = \varepsilon e_1, e_1 e_3, e_3 e_5, e_5 e_4, e_4 e_6$ para τ_3

Usando el Algoritmo 1.3, la red de Petri correspondiente a las 3 secuencias de trazas de eventos del Ejemplo 1.2. corresponde a la Figura 1.3. Nos podemos dar cuenta que una de las secuencias no es cíclica.

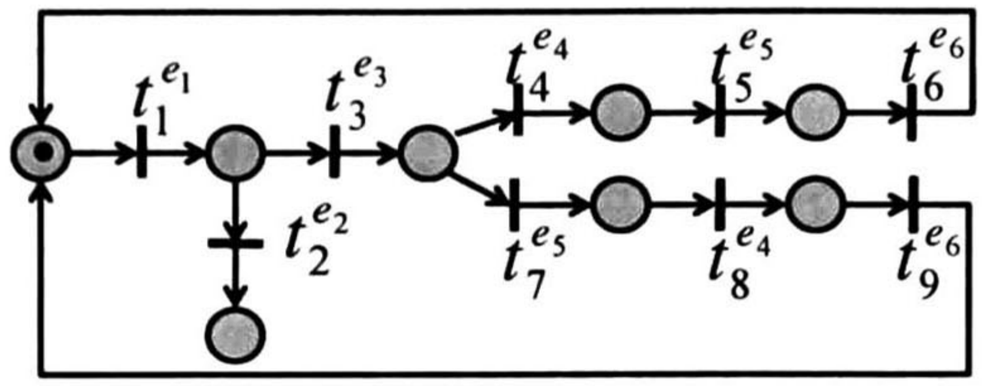


Figura 1.3 Modelo inicial de Red de Petri del Ejemplo 1.2. [Estrada, 2009]

Simplificando la estructura básica

Se realizan operaciones adicionales a la estructura básica con la finalidad de obtener un modelo equivalente reducido. Se consideran las siguientes reglas de transformación:

Regla 1: $\forall t_a^{e_j}, t_b^{e_j} \in p_{ini}\bullet | a \neq b \qquad merge(t_a^{e_j} t_b^{e_j})\, merge(t_a^{e_j}\bullet, t_b^{e_j}\bullet)$

Regla 2: $\forall t_a^{e_j}, t_b^{e_j} \in \bullet p_{ini} | a \neq b \qquad merge(t_a^{e_j} t_b^{e_j})\, merge(\bullet t_a^{e_j}, \bullet t_b^{e_j \bullet})$

En la Figura 1.4 (a) nos damos cuenta que entre la transición asociada a e_3 y la transición asociada a e_6 existen caminos con todas las posibles permutaciones de e_4 y e_5. Entonces se puede convertir en un componente concurrente y se obtiene la red de la Figura 1.4 (b).

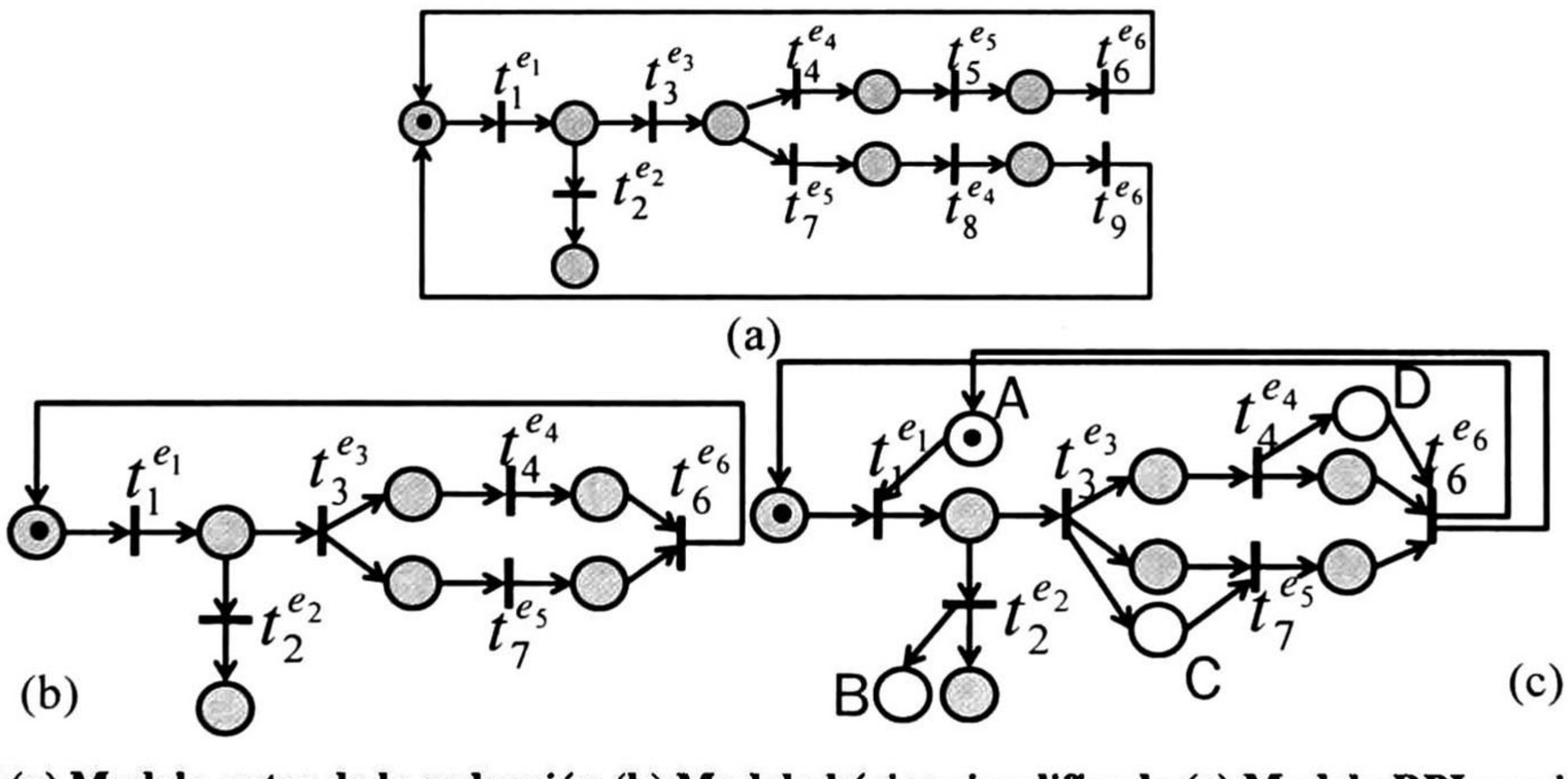


Figura 1.4 (a) Modelo antes de la reducción (b) Modelo básico simplificado (c) Modelo RPI que incluye los lugares observables. [Estrada, 2009]

Este modelo conserva la misma secuencia de vectores de eventos que la anterior. La simplificación por el análisis de la concurrencia no es estrictamente necesaria para representar la secuencia; sin embargo, el modelo equivalente con transiciones concurrentes puede ser más sencillo. Aunque el análisis puede ser ineficiente cuando el número de caminos en la subred es grande, usualmente este número es reducido.

Una vez que se añaden los lugares observables y los lugares implícitos son eliminados, solo queda añadir la información de entrada para completar el modelo de RPI. La información de entrada está asociada con etiquetas para las transiciones que se obtienen de las funciones simbólicas de entrada de los eventos.

El modelo final para ilustrar este ejemplo se muestra en la figura 1.5. Las entradas asociadas para las transiciones dadas son: $\lambda'(e_1)$ = a_1, $\lambda'(e_2) = \varepsilon$, $\lambda'(e_3)$ = b_1 c_1, $\lambda'(e_4)$ = b_0, $\lambda'(e_5)$ = c_0, $\lambda'(e_6)$ = a_0.

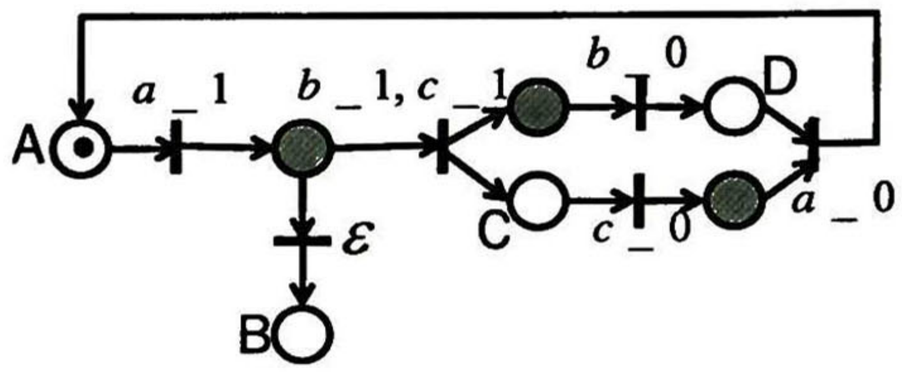


Figura 1.5 Modelo simplificado de RPI. [Estrada, 2009]

La red representa la secuencia de vectores evento de tamaño κ, el lenguaje de tamaño κ+1 de la red es igual al lenguaje observados.

1.2.5. Método de identificación estadístico

El método presentado en [Estrada,2012] construye una Red de Petri Interpretada (RPI), en la que se describen las relaciones causales y concurrentes observadas entre los eventos basándose en un enfoque estadístico en el cual se puede realizar el análisis de secuencias de entrada-salida muy grandes [Estrada, 2012]. El método consta de dos pasos generales:

1. Construcción de la parte observable de la RPI.
2. Se determinan los lugares no observables y la relación entre los eventos.

Antes de presentar el algoritmo es necesario explicar algunos conceptos tomados de [Estrada, 2012]. Se definen los tipos de eventos.

La secuencia de vectores de eventos observados es derivada de la palabra de entrada-salida w, $E(j) = w(j+1) - w(j)$; cada uno de éstos se divide, a su vez, vectores de entrada y de salida.

$$E(j) = \begin{bmatrix} IE(j) \\ OE(j) \end{bmatrix}, \text{ donde } IE(j) = \begin{bmatrix} IE_0(j) \\ IE_1(j) \\ \vdots \\ IE_{m-1}(j) \end{bmatrix} \text{ y } OE(j) = \begin{bmatrix} OE_0(j) \\ OE_1(j) \\ \vdots \\ OE_{m-1}(j) \end{bmatrix}$$

De éstos se determinan los eventos elementales que se definen de la siguiente manera

$$IE(j) = IE_{j1} \bullet IE_{j2} \bullet \ldots = \prod IE_{ji} \text{ t.q. } I_{ji}(j+1) - I_{ji}(j) \neq 0$$

$$OE(j) = OE_{j1} \bullet OE_{j2} \bullet \ldots = \prod OE_{ji} \text{ t.q. } O_{ji}(j+1) - O_{ji}(j) \neq 0$$

Si ningún evento curre en $E(j)$, se denota como

$$IE(j) = \varepsilon \ (OE(j) = \varepsilon).$$

De manera similar se puede representar cada vector de entrada

$$I(j) = I_{j1} \bullet I_{j2} \bullet \ldots = \prod I_{ji} \quad \text{t.q} \quad I_{ji} = \begin{cases} I_i = 1 \, if \; I_i(j) = 1 \\ I_i = 0 \, if \; I_i(j) = 0 \end{cases}$$

1.2.6. Otros enfoques

1.2.6.1. La identificación mediante la programación evolutiva

La programación evolutiva fue originalmente concebida por Lawerence J. Fogel en 1960 y consiste en un mecanismo que hace evolucionar un conjunto de máquinas de estados finitos. Esta técnica desarrolla un conjunto de soluciones las cuales muestran un comportamiento óptimo con respecto a un ambiente y a una función deseada.

El método presentado en [Aguilar, 2007] estudia la identificación de SED desde un nuevo enfoque: programación evolutiva. La programación evolutiva tiene sus bases en la Computación Evolutiva, y no es más que un método de búsqueda y optimización. El planteamiento es hacer evolucionar un conjunto de individuos, cada uno representando un modelo SED, que a su vez sea una posible solución del sistema que se desea identificar. El modelo es representado por medio de una Máquina de Estados Finito (MEF).

Así, a través de un número finito de generaciones estos sistemas se deben irse ajustando al entorno, para dar como resultado un conjunto de máquinas de estados finitos que se lograrán acoplar a una secuencia de eventos, eventos obtenidos del sistema real y que sirven para reproducir el comportamiento general del modelo.

Para llevar a cabo la identificación de una MEF se establece un conjunto de condiciones que deben ser satisfechas para poder determinar las propiedades de la máquina desconocida. Primero se asume que todo el conjunto de símbolos de entrada está definido, se debe poseer esta información para poder definir completamente las funcionès de transición (salida y estado siguiente). La condición final que se debe de cumplir para identificar una MEF es que ésta debe ser fuertemente conexa.

Cuando la salida de la máquina es independiente de la entrada, se trata de una máquina de Moore, de lo contrario se asume que la máquina se comporta como el modelo de Mealy.

La inicialización de las MEF se puede hacer de dos maneras: completamente aleatoria o desde una población ya existente.

La función de costos posee dos atributos (dos cadenas de caracteres: la secuencia de entrada y la secuencia de salida), que son compartidos para todos los individuos de la población. La capacidad de cada máquina se mide comparando la secuencia de salida que genera el individuo con la secuencia de salida que se tiene como referencia, cuando es procesada la secuencia de entrada.

El procedimiento se ejecuta de la siguiente manera: se aplica el primer símbolo de entrada al individuo que se encuentra en un estado inicial, si ese símbolo no es reconocido por la MEF se penaliza con un valor positivo.

De lo contrario, si el símbolo es reconocido se compara el símbolo de salida que genera la MEF con el símbolo de salida de referencia, si los símbolos son diferentes el individuo es

penalizado con un valor positivo, pero si los símbolos de salida coinciden la máquina es premiada con una valor negativo.

Luego, se verifica hacia qué estado transita el autómata y se repite el proceso para el siguiente símbolo de entrada-salida correspondiente hasta que se haya alcanzado el final de las secuencias.

Además se incorpora en la función de costos la capacidad de recompensar a aquellas MEF que poseen menos estados, esto es opcional y es con la finalidad de obtener MEF's que no sólo traducen una secuencia de entrada, sino que también sean simples (desde el punto de vista del tamaño).

Se describe el proceso para buscar MEF de los sistemas reales. Se considera un conjunto de medidas que van a permitir la identificación de un sistema real.

Algoritmo 1.4

1. Obtener la sucesión de acontecimientos de los SED a identificar. Es el paso fundamental de todo proceso de identificación, la secuencia debe ser la más representativa posible.
2. Analizar la complejidad del sistema a identificar con el fin de determinar si es necesario para dividir en diferentes fases del proceso de identificación.
3. Empezar a generar prototipos que son posibles resultados, de fases o de la totalidad del experimento.
4. Realizar las primeras pruebas que son de nuestro interés. Una ventaja es que el individuo reconoce la totalidad de la sucesión de eventos (señales de entrada), sin preocuparse por la cantidad de estados que esta posee.
5. Inicializar la población con los individuos que resultaron del paso anterior, pero en esta nueva ejecución del algoritmo se corrige la maquina con menos estados la cual será premiada, eliminando así estados redundantes.
6. Comprobar los modelos resultantes, si no presentan comportamientos satisfactorios volver al paso 3 o analizar las secuencias de entrada y salida para determinar la posibilidad de realizar la evolución con los nuevos datos (paso 1).
7. Si los individuos son lo suficientemente capaces, en este paso deben enfrentarse a una secuencia de validación, con la que se puede analizar el comportamiento del modelo, y de esta manera generar una función de error. Si las máquinas no aprueban con éxito esta fase, se deben estudiar nuevos prototipos.

Ejemplo 1.3. El caso a identificar es un sistema compuesto por un tanque que almacena un líquido [Aguilar, 2007], con un par de válvulas que regulan la entrada y salida. Antes de comenzar con el proceso de identificación, es indispensable tener la secuencia de eventos de entrada y de salida.

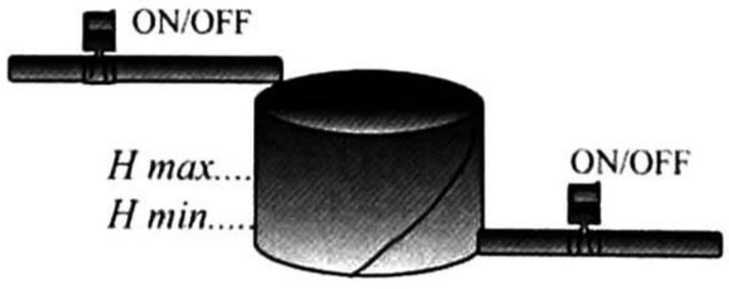


Figura 1.6 Sistema del caso de estudio

Se usa la técnica de Programación Evolutiva para iniciar el proceso de identificación. El comportamiento interno de la planta es descrito por un grupo de doce (12) estados, un alfabeto de entrada compuesto por ocho (8) símbolos, y un alfabeto de salida de tres (3) símbolos.

Entrada	*Descripción*
α	Para vaciar el tanque, el líquido solo sale pero no entra.
β	Para llenar el tanque, el líquido solo entra pero no sale.
γ	Para dar líquido, se busca un flujo normal donde el líquido continuamente entra y sale.
δ	Para reiniciar o apagar el sistema, se cierran las dos válvulas.
ε	La altura del líquido en el tanque pasa del nivel mínimo a normal.
ζ	La altura del líquido disminuye, pasando del nivel normal al mínimo.
η	El nivel de líquido en el tanque pasa la altura máxima permitida.
θ	El líquido pasa de la altura máxima al nivel normal de operación.

Tabla 1.1 Símbolos del alfabeto de entrada

Salida	*Descripción*
A	Después de generar el evento, la altura del líquido en el tanque pasa a un nivel mínimo.
B	Después de generar el evento, la altura del líquido en el tanque pasa a un nivel normal.
C	Después de generar el evento, la altura del líquido en el tanque pasa a un nivel máximo.

Tabla 1.2 Símbolos del alfabeto de salida

Se supone que el depósito puede llegar a tres estados:

- Bajo Nivel. Ocurre cuando el líquido en el tanque está por debajo de una marca de referencia *H(tanque)<H(mínimo)*.
- Nivel Normal. La altura del líquido en el depósito esta entre dos límites de referencia, lo que indica que está por debajo del nivel máximo permitido, pero por encima del nivel mínimo, *H(mínimo)<H(tanque)<H(máximo)*.
- Alto Nivel. En este estado, el nivel de líquido en el tanque ha sobre pasado la altura máxima permitida, *H(tanque)>H(máximo)*.

Las válvulas son del tipo (ON/OFF), por lo tanto dos estados son posibles. De acuerdo con esto, se obtiene tres estados para el tanque y dos para cada válvula, siendo doce estados posibles en que la planta puede ser en un momento dado (véase la Fig. 3).

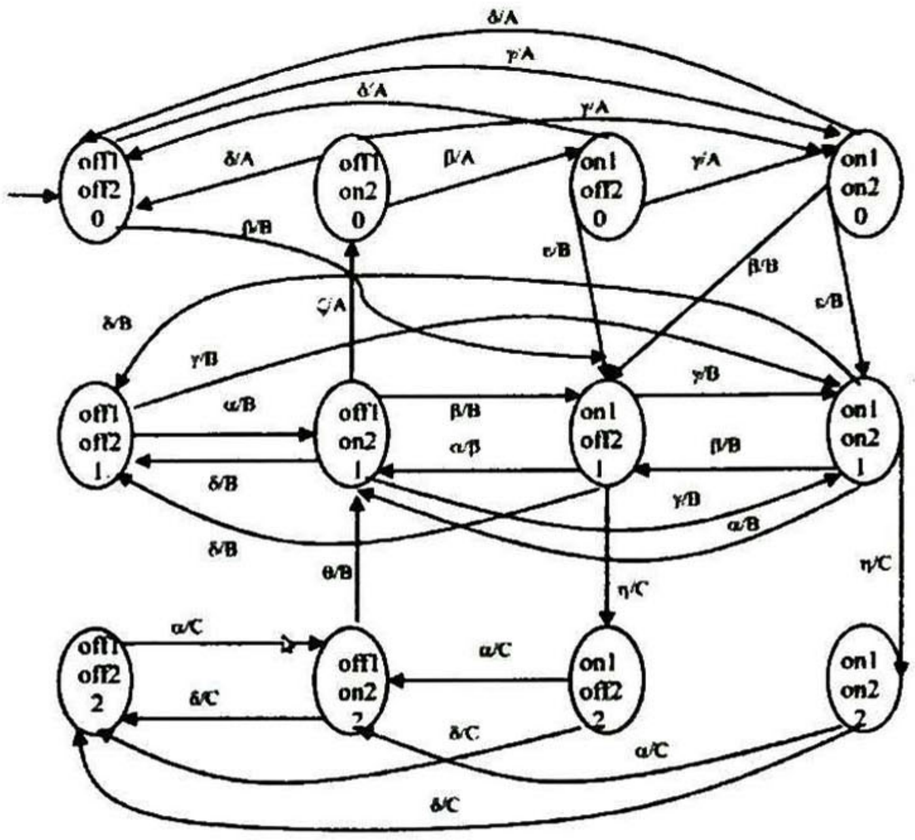
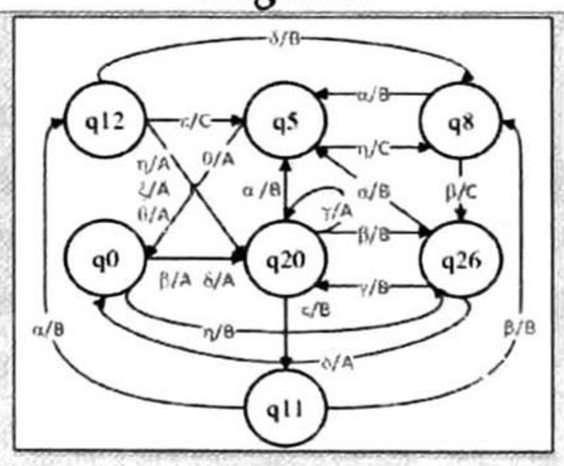
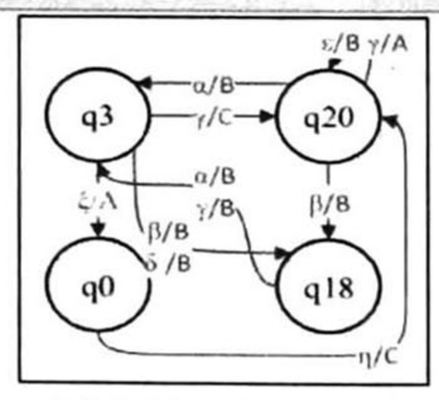


Figura 1.7 MEF que modela el sistema real [Aguilar,2007]

Se considera que el flujo que pasa por la válvula de entrada es más grande que el flujo de salida; esta diferencia se utiliza para permitir que la MEF tenga eventos que son causados internamente, como los eventos de cambio de nivel.

Los parámetros principales del experimento son:

- Tamaño de la población = 10
  Tamaño del grupo de entrada = 8
  Tamaño del grupo de salida = 3
  Símbolos.

Fase	*Datos*	*Figura*
Primera	***Entrada:*** γ, ε, β, α, ζ, δ, γ, ε, β, α, ζ, δ, β, γ, β, α, ζ, δ, β γ, β, α, ζ, δ, β, α, ζ, β, ε, α, γ, α, ζ, β, ε, α, δ ***Salida:*** A, B, B, B, A, A, A, B, B, B, A, A, B, B, B, B, A, A, B, B, B, B, A, A, B, B, A, A, B, B, B, B, A, A, B, B, B, B, A, A, B, B	 **Figura 1.8 Máquina generada en la primera fase**
Segunda	***Entrada:*** γ, β, α, δ, α, β, γ, β, α, δ, γ, δ, γ, δ, α, ζ, δ, γ, ε, β, α, δ, α, ζ, δ, γ, ε, β, α, δ ***Salida:*** B, B, B, B, B, B, B, B, B, B, B, B, B, B, B, B, A, A, A, B, B, B, B, B, B, A, A, A, B, B, B	 **Figura 1.9 Máquina generada en la segunda fase**

Tercera ***Entrada:***
α, β, δ, α, β, δ, α, ζ, δ, β, α, δ, γ, β, δ, γ, β, δ, γ, η, α, θ, δ, γ, η, α, θ, δ
Salida:
B, B, B, B,B, B, B, A, A, B, B, B, B, B, B,B, B, B, B,C, C, B, B, B, C, C, B, B

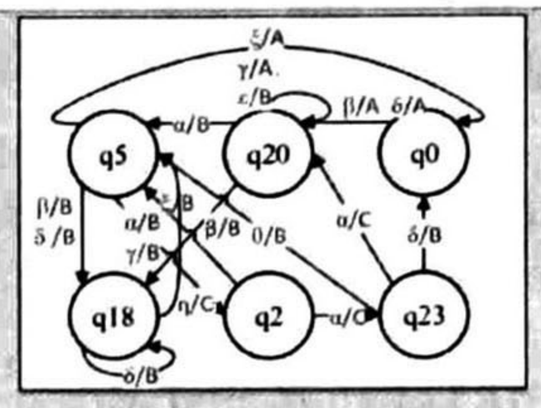


Figura 1.10 Máquina generada en la tercera fase

Cuarta ***Entrada:***
α, β, γ, α, β, γ, δ, α, β, γ, α, β, γ, δ
Salida:
B,B,B,B,B,B,B,B,B,B,B,B,B,B

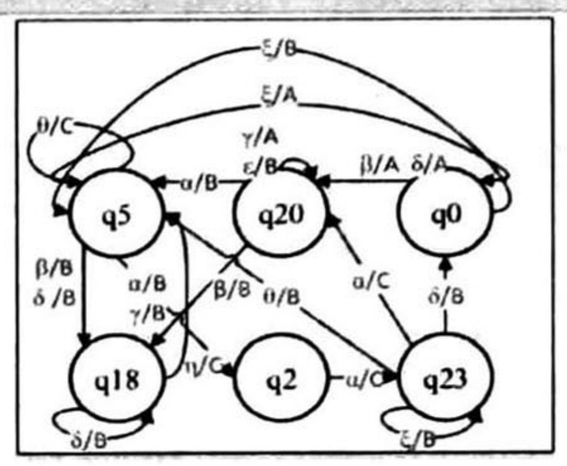


Figura 1.11Resultado de la cuarta fase de identificación

Quinta ***Entrada:***
γ, η, δ, α, θ, δ, γ, η, δ, α, θ, δ
Salida:
B, C, C, C, B

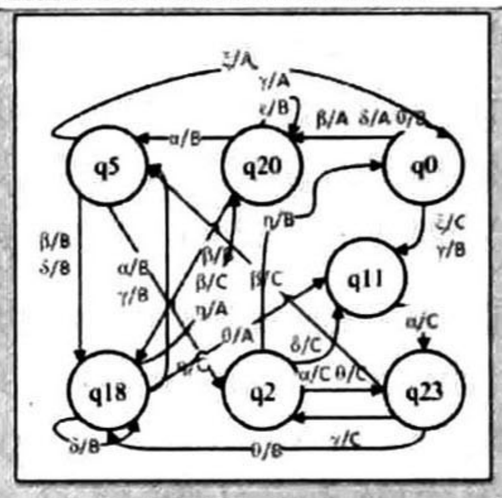


Figura 1.12 Máquina generada en la quinta fase

Sexta ***Entrada:***
γ, η, α, θ, β, η, α, θ, β, η, α, θ, δ, γ, η, α, θ, β, η, α, θ, β, η, α, θ, δ
Salida:
B, C, C, B, B, C, C, B, B, C, C, B, B, B, C, C, B, B, C, C, B, B, C, C, B, B

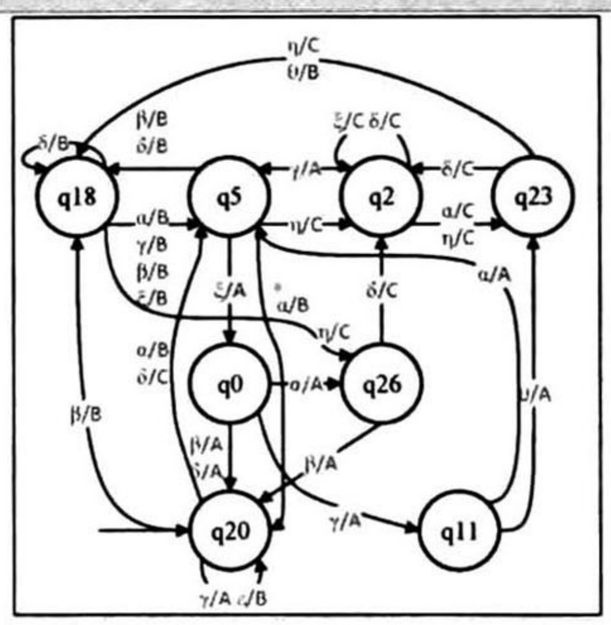


Figura 1.13 Máquina identificada del Sistema real

1.3. Métodos de minería de procesos

Avances recientes en las investigaciones de minería de procesos han hecho posible descubrir y analizar procesos de negocios basados en registros de eventos. Actividades

ejecutadas por personas, máquinas y herramientas de software dejan marca en los llamados "registros de eventos" (*event logs*). Estos eventos (p.e. como entrar en un pedido de un cliente en SAP, registrar el vuelo a un pasajero, un médico que cambia la dosis de medicina de un paciente, una agencia de construcción rechaza un permiso) tienen en común que están registrados en los sistemas de información.

El volumen de datos y la capacidad de almacenamiento han crecido espectacularmente en la década pasada. Por lo tanto los procesos de negocios deben ser administrados, respaldados y mejorados basados en datos de eventos y no en opiniones personales.

La importancia de la minería de procesos y los interesantes retos científicos relacionados hacen de la minería de procesos un tema importante en la gestión de procesos de negocios (BPM: Business Process Management).

El objetivo de la minería de procesos es descubrir, monitorear y mejorar los procesos a través de la obtención de conocimiento a partir de los registros de eventos que podemos encontrar en los sistemas de información que tenemos en la actualidad.

Consideramos aquí dos técnicas de minería de procesos.

1.3.1. Minería de procesos por grafo dirigido

Un proceso de flujo de trabajo especifica la forma en que las actividades se realizan en una empresa y los recursos que utilizan.

Para analizar el proceso, hay que realizar el conjunto o subconjunto de actividades que lo compondrán. Pueden existir dependencias entre las diferentes actividades. Un método presentado por Rakesh Agrawal [Agrawal, 1998], identifica las diferencias entre estas actividades y el enfoque principal es modelar el proceso como un grafo dirigido.

El *registro de ejecución* (log) de un proceso es una simple secuencia de eventos que inicia y termina con los mismos eventos, llamados START y END respectivamente y posee como atributos el nombre del proceso, la actividad, el tiempo en el que ocurre cada evento y el cambio en la salida.

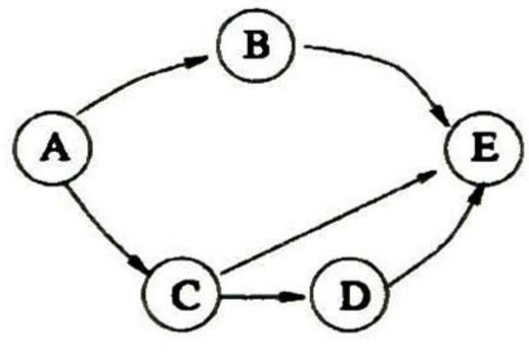


Figura 1.14 Ejemplo de flujo de trabajo

ABCE, ACDBE, ACDE son ejemplos de registros de ejecución para el flujo de trabajo de la figura 1.14

Dado un *log* de ejecuciones del mismo proceso, la actividad B sigue de A si cualquier actividad B inicia después de que A termina en cada ejecución en la que aparecen ambas actividades, o si existe una actividad C tal que C sigue a A y B sigue a C, a esto se le conoce como *siguiente*.

Si existe una dependencia entre dos actividades en un proceso real, entonces estas dos actividades deben aparecer en el mismo orden en cada ejecución.

Dado un *log* de la ejecución de un proceso se dice que la actividad B *sigue* a la actividad A, si la actividad B comienza después de que la actividad A termina. Si B *sigue* a la actividad A pero A no *sigue* a la actividad B se dice entonces que B *depende* de A.

Entonces considerando el siguiente *log* de ejecuciones de un proceso determinado *{ABCE, ACDE, ADBE}*; La actividad B *sigue* de A. Pero A no *sigue* de B, entonces B *depende* de A. La actividad B *sigue* de D, y D *sigue* de B, porque *sigue* C *sigue* de B, entonces B, y D son independientes.

Sea un conjunto de actividades V y un registro de ejecuciones L, del mismo proceso, el *grafo de dependencia*, es aquel que tiene un camino de la actividad u a la actividad v si y solo si, v depende de u.

El grafo de dependencia no es único. Grafos con la misma cerradura transitiva representan la misma dependencia.

Algoritmo 1.5

Entrada: L (Log de ejecuciones).

Salida: G (Grafo obtenido).

---

1. Comienza con el grafo G = (V, E) con V inicializado con el conjunto de actividades del proceso y E igual a vacío.
2. Para cada proceso en ejecución en L, y para cada par de actividades u, v tal que u termina antes que v comience, añade un arista (u, v) a E.
3. Remueve las aristas que aparecen en ambas direcciones.
4. Para cada componente fuerte conexo de G, remueve de E todas las aristas entre vértices en el mismo componente fuertemente conexo.
5. Para cada proceso en ejecución en L.
   a) Encuentra el subgrafo del G inducido.
   b) Procesa la reducción transitiva del subgrafo.
   c) Marca esas aristas en E que representen la reducción transitivas.
6. Elimina las aristas no marcadas en E.
7. En el grafo que obtenemos, unir los vértices que pertenecen a la misma instancia de una actividad.
8. Regresa (V, E).

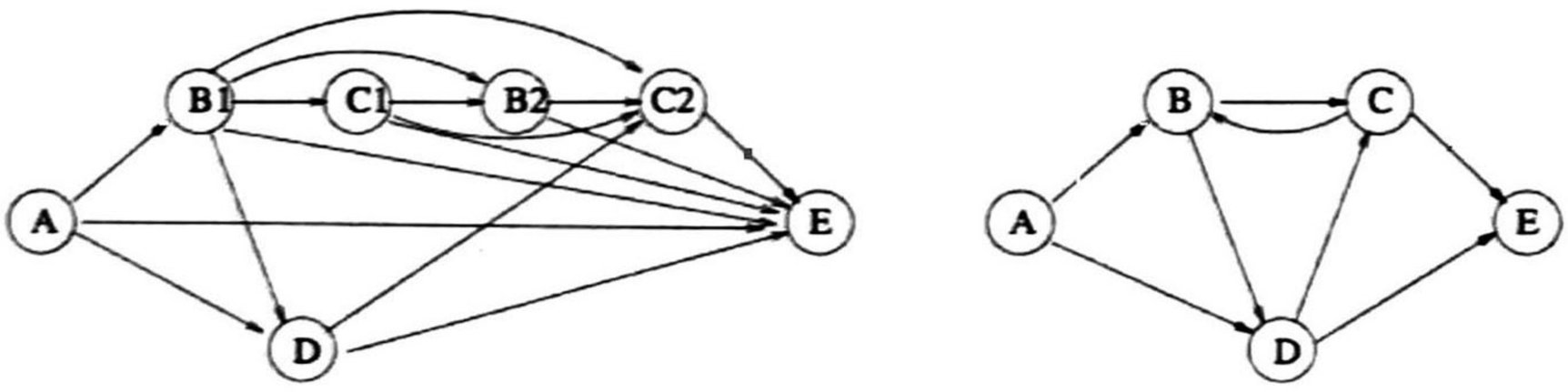


Figura 1.15 Ejemplo

Ejemplo 1.3. Considere el *log* = {ABDCE, ABDCBCE, ABCBDCE, ADE}. El primer grafo de la Figura 1.15 Ejemplo surge después de haber procesado el paso 4 del Algoritmo 1.6. No hay aristas entre D y C1 porque D es una ejecución donde D aparece después de C1, y antes. Lo mismo sucede con D y B2.

Las aristas (A, E), (B1, B2), (B1, C2), (B1, E), (C1, C2), y (B2, E) son eliminadas en el paso 6.

Dando como resultado el segundo grafo de la Figura 1.15 Ejemplodespués de haber realizado el paso 8 del algoritmo. Este grafo muestra el ciclo de las actividades B y C.

1.3.2. Minería de procesos basada en la inferencia de componentes repetitivas

En este método [Tapia, 2013] se extienden los resultados obtenidos en [Estrada, 2013], para la determinación de la parte no observable de una red de Petri a partir de una secuencia de transiciones S. Uno de los conceptos retomados de este enfoque es la dependencia repetitiva, la cual es la base para determinarlas componentes repetitivas de un modelo.

La *Dependencia Repetitiva* de una transición t_i contiene el conjunto de transiciones que ocurren siempre repetitivamente para poder observar a t_i nuevamente; es decir, es un conjunto de transiciones que forma parte de un invariante del sistema. Sin embargo, esta definición no es suficiente para caracterizar todos los invariantes del sistema.

En general el método de minado de procesos de flujo de trabajo presentado consta de 4 pasos los cuales se explican a continuación.

Algoritmo 1.6

1. Detectar las dependencias repetitivas entre los eventos. En este paso se retoma y se extiende la definición de dependencia repetitiva con el fin de detectar eventos que pertenecen a esta dependencia. En este paso también se calculan conjuntos maximales que se acercan mucho a ser los soportes de los t-invariantes de la red.
2. Determinación de las componentes repetitivas de la red. A partir de los conjuntos maximales se propone una técnica para verificar que efectivamente sean los soportes de t-invariantes de la red.
3. Construcción del modelo. Con el conocimiento de las relaciones causales y concurrentes analizadas en el capítulo 2 y los t-invariantes de la red se expone una técnica para construir una primera aproximación de la red.
4. Ajuste final al modelo. Si los t-invariantes de modelo obtenido en el paso 3 no coinciden con los t-invariantes del análisis en el paso 2, es necesario realizar un ajuste al modelo. Si por el contrario coinciden estos, el modelo del paso 3 puede presentarse como resultado. En este paso se describe un método para ajustar el modelo y dar como resultado la red que representa exactamente el lenguaje en S.

1.4. Caracterización de los métodos analizados

Para fines de comparación, se analizan los diferentes enfoques que consideran varias características.

Estas funciones se estructuran en 4 categorías: las que caracterizan al SED, los que describen el proceso de identificación, de aquellos que cumplen el modelo identificado, y aquellos que están considerando las características generales del algoritmo.

Características SED

- Tipo de entradas/salidas. En el caso general, las entradas y salidas de los SED a ser identificados son discretos (que pueden tomar un número finito de valores). Si todas las entradas y salidas sólo pueden tomar dos valores (on/off), el SED es llamado lógico.
- Comportamiento iterativo. Un SED es cíclico si llega de forma iterativa el estado inicial durante su funcionamiento. Si itera sobre el mismo comportamiento revisitar un estado que no es el inicial se llama repetitivo.

Las características del proceso de identificación

- Información a priori. Si no hay conocimiento disponible sobre el SED distinta de sus entradas y salidas de la evolución, la identificación es absoluta (comúnmente llamado caja negra). De lo contrario, la identificación es relativa.
- Modelo de actualización. Cuando el modelo es actualizado incrementalmente, el método actualiza progresivamente el modelo de la información observada; de lo contrario, el procedimiento de identificación es global: debe ser ejecutado en la totalidad de las secuencias observadas cada vez nuevas secuencias se recogen.

Características modelo identificado

- Concurrencia. Esta característica considera si el modelo obtenido puede representar el comportamiento de manera explícita concurrente observada desde el sistema.
- Precisión. Este término está relacionado con la exhaustividad del modelo identificado. Si este modelo representa exactamente el comportamiento observado, es completa.

Características del algoritmo

- Datos considerados. El algoritmo de identificación construye un modelo identificado a partir de datos experimentales que pueden ser entradas y/o salidas del sistema observado.
- Estrategia. Si el algoritmo de identificación devuelve todos los posibles modelos que representan el comportamiento observado, el algoritmo se llama enumerativo. Si sólo se da uno de los modelos posibles, es constructivo.
- Ejecución. Si la construcción del modelo se puede realizar durante el funcionamiento del sistema mediante el cálculo de un nuevo modelo de nuevas mediciones de las entradas del sistema y / o salidas, la ejecución se realiza en línea. De lo contrario, la ejecución está fuera de línea; el algoritmo no es capaz de correr al mismo tiempo que el sistema.
- Complejidad. Este término se refiere a la complejidad computacional del algoritmo de identificación. Procedimientos de tiempo polinomio son mejores que los exponenciales para hacer frente a los grandes sistemas que presentan una gran cantidad de secuencias de entrada-salida.

Las principales características de los métodos considerados se resumen en la Tabla 1.3.

Enfoque de Identificación Criterios de Comparación	Enfoque progresivo	Enfoque Programación Lineal Entera	Enfoque RPI Paramétrica	Enfoque de identificación estadístico	Programación Evolutiva	Enfoque de Minería de procesos
Características del SED						
Tipos de entradas	Lógicas	Discreta	Lógica	Discreta	Discreta	Discreta
Comportamiento iterativo	Repetitivas	Ninguna	Cíclica	Cíclica	Ninguno	Ciclica
Las características del proceso de identificación						
Información a priori	Absoluto	Relativo	Absoluta	Absoluta	Absoluta	Relativo
Modelo de actualización	Incremental	Global	Incremental	Incremental	Incremental	Global
Características del modelo identificado						
Concurrencia	Explicita	Explicita	Explicita	Explicita	Explicita	Explicita
Precisión	No completo	No completo	Completo	Completo	Completo	No completo
Características del Algoritmo						
Datos considerados	Salidas	Eventos y salidas	Entradas y Salidas	Entradas y Salidas	Población	Eventos
Estrategia	Constructiva	Enumerativo	Constructiva	Constructiva	Constructiva	Constructiva
Ejecución	En línea	En línea y fuera de línea	Fuera de línea	En línea	En línea	Fuera de línea
Complejidad	Polinomial	Exponencial	Polinomial	Polinomial	Exponencial	Polinomial

Tabla 1.3 Caracterización de los métodos analizados

Capítulo 2
Síntesis de Redes de Petri Interpretadas

Resumen. En este capítulo se incluyen los antecedentes necesarios para la presentación de la propuesta en esta tesis. Por un lado se presentan los conceptos básicos sobre redes de Petri, en particular las redes de Petri interpretadas. Por otro lado, se define el problema de identificación de procesos de eventos discretos y se da un resumen de una técnica reciente [Estrada, 2012] para tratar dicho problema abordado: la obtención de la parte observable de un modelo en redes de Petri para procesos de eventos discretos, a partir de secuencias de vectores entrada-salida.

2.1. Modelado con redes de Petri

Existen diferentes formalismos para modelar y analizar un SED. Uno de estos formalismos son las Redes de Petri (RP). RP representa la mayoría de las características propias de un SED como decisiones, sincronizaciones, exclusiones mutuas, concurrencia, relaciones causales entre otras características.

Una de las ventajas de una RP es que tienen una representación gráfica y matemática que hace su uso más amigable. La herramienta grafica de una RP puede ser usada como un diagrama de flujo o diagrama de bloques. Y como una herramienta matemática es posible establecer un una ecuación de estado que controla el comportamiento del sistema.

En este trabajo se utiliza una clase de RP llamada Redes de Petri Interpretadas (RPI). Una RPI tiene la propiedad de relacionar las señales de entrada y salida del sistema, con la estructura de una RP dándole un significado físico.

En este capítulo se introducen los conceptos básicos y las propiedades de RP, también se introducen los conceptos de RPI y las definiciones relacionadas al modelado de SED usados en este trabajo.

2.1.1. Conceptos Básicos

Una *Red de Petri* es un grafo bipartita dirigido, que consiste de dos tipos de vértices o nodos: los lugares y las transiciones, los cuales están representados por círculos y barras respectivamente. Los arcos, representados gráficamente por flechas, van de un lugar a una transición y viceversa. Los arcos pueden estar etiquetados por un peso expresado enteros positivos, donde un arco k-ponderado puede ser interpretado como un conjunto de k arcos paralelos. Si el peso del arco es uno usualmente se omite. La definición formal de la RP con arcos con peso unitario, llamada RP ordinaria, es la siguiente.

Definición 2.1 Una estructura de Red de Petri ordinaria denominada G es un grafo bipartita dirigido representado por la cuádrupla $G = (P, T, I, O)$, donde:

- $P = (p_1, p_2, \dots, p_n)$ es un conjunto finito de vértices llamados lugares.
- $T = (t_1, t_2, \dots, t_m)$ es un conjunto finito de vértices llamados transiciones.
- $Pre: P \times T \rightarrow \{0,1\}$ es una función que representa los arcos que van de lugares a las transiciones.
- $Post: T \times P \rightarrow \{1,0\}$ es una función que representa los arcos que van de transiciones a los lugares.

Ejemplo 2.1 Considera la RP de la Figura 2.1. El conjunto de P, T, I y O, son los siguientes:

$$P = \{p_1, p_2, p_3, p_4, p_5, p_6\} \qquad T = \{t_1, t_2, t_3, t_4, t_5\}$$

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

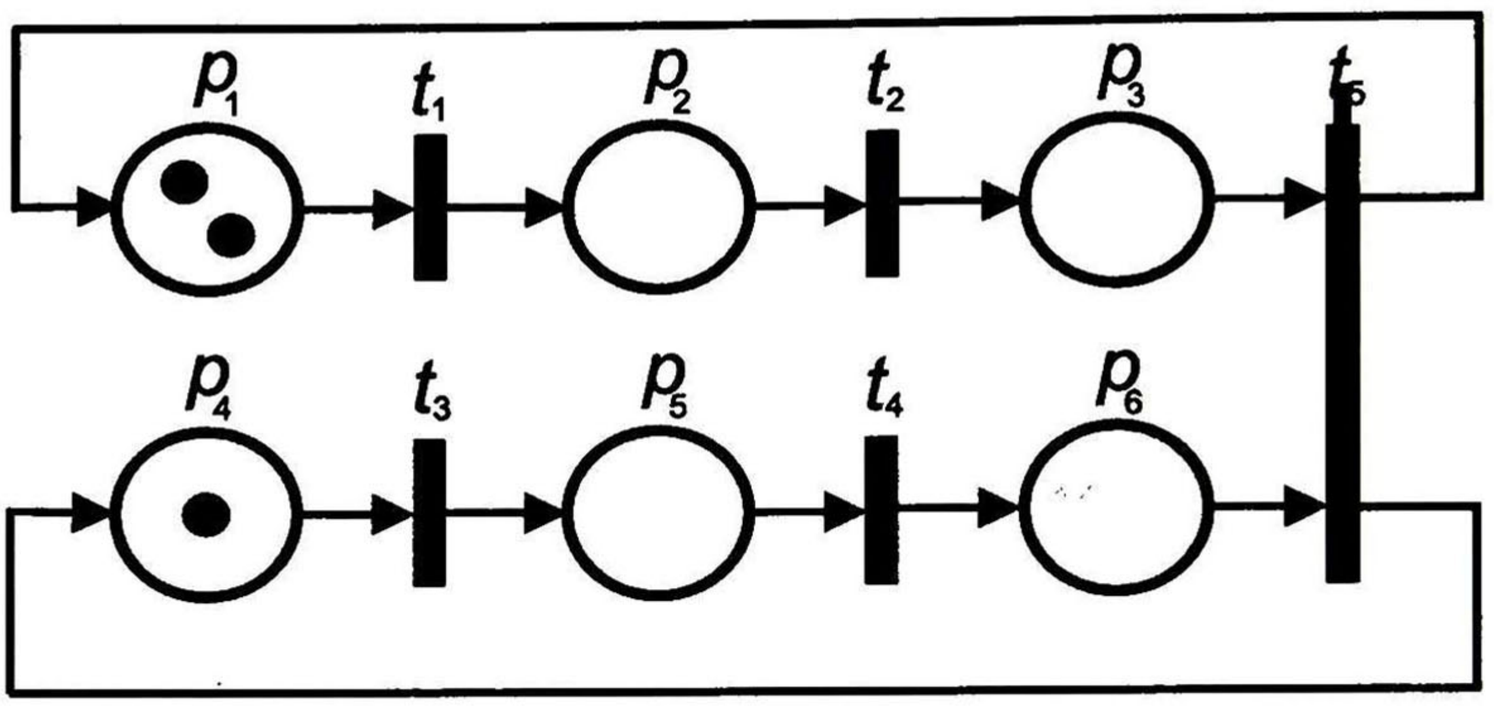


Figura 2.1 Representación gráfica de una Red de Petri

Sea un nodo x de G, el conjunto:

$\bullet x = \{y / (y, x) \in F\}$ es el pre-conjunto de x

$x \bullet = \{y / (x, y) \in F\}$ es el post-conjunto de x

Los elementos en el pre-conjunto (post-conjunto) de un lugar son las transiciones de entradas (salidas).

Los elementos en el pre-conjunto (post-conjunto) de una transición son los lugares de entradas (salidas).

Ejemplo 2.2 Considera la RP de la figura 2.1. El $\bullet t_1$, $t_4 \bullet$, $\bullet p_1$ y $p_2 \bullet$ es:

$\bullet t_1 = p_1, t_4 \bullet = p_6, \bullet p_1 = t_5,$ y $p_2 \bullet = t_2$

2.1.2. Ecuación de estado de una RP

El comportamiento de un SED puede ser descrito en términos de estados y sus cambios. En una RP los estados de un sistema pueden ser representados por marcados. Siguiendo la representación básica y los conceptos necesarios para representar el comportamiento de un SED usando una red de Petri.

Definición 2.2 Para una RP con n lugares y m transiciones, la matriz de incidencia de G denotada por $C = C^+ - C^-$, donde $C^- = [c_{ij}^-]$; $c_{ij}^- = I(p_i, t_j)$; y $C^+ = [c_{ij}^+]$; $c_{ij}^+ = O(p_i, t_j)$ son el pre y el post de la matriz de incidencia respectivamente.

Ejemplo 2.3 Considera la matriz de incidencia para la RP de la Figura 2.1. El pre y post son procesados para obtener C.

$$C = \begin{bmatrix} 0&0&0&0&1 \\ 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&0&0&1 \\ 0&0&1&0&0 \\ 0&0&0&1&0 \end{bmatrix} - \begin{bmatrix} 1&0&0&0&0 \\ 0&1&0&0&0 \\ 0&0&0&0&1 \\ 0&0&1&0&0 \\ 0&0&0&1&0 \\ 0&0&0&0&01 \end{bmatrix}$$

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

2.1.3. Marcado de una Red de Petri

Definición 2.3 La función de marcado $M: P \rightarrow \mathbb{Z}^+$ representa el número de marcas que se encuentran en cada lugar; el cual es usualmente representado con el vector $|P|$-entrada. $\mathbb{Z}^+$ es el conjunto de enteros no negativos.

Por definición un marcado asigna a cada lugar p un entero no negativo k, entonces el lugar p se dice que está marcado con k *marcas* (*tokens*). Las *marcas* son representadas como puntos negros dentro del lugar p.

Un marcado M_k está representado por un vector columna $n \times 1$, donde $n = |P|$. La i-ésima entrada de M_k se representa como $M_k(p_i)$ es el número de *marcas* en el lugar p_i inmediatamente después del k-ésimo disparo en alguna secuencia de transiciones.

El marcado para la RP de la figura 2.1 es $[2 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]^T$ donde el lugar p_1 y p_3 tienen 2 y 1 *marcas* respectivamente.

Definición 2.4 Un sistema de red de Petri o simplemente res de Petri (RP) se define como $N = (G, M_0)$, donde G es una estructura RP y M_0 es un marcado inicial dado.

El marcado $[2 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]^T$ es el marcado inicial de la RP de la figura 2.1.

Definición 2.5 El vector de control v_k es un vector columna $m \times 1$, donde $m = |T|$. La j-ésima entrada de v_k está definida como:

$$v_k(t_j) = \begin{cases} 1 & si\ t_j\ se\ activa\ en\ el\ k-ésimo\ disparo \\ 0 & en\ otro\ caso \end{cases}$$

En un sistema de red de Petri, una transición t_j está habilitada para el marcado M_k si $\forall p_i \in P$, $M_k(p_i) \geq I(p_i, t_j)$; una transición habilitada t_j puede ser disparada generando un nuevo marcado M_{k+1}.

Definicion 2.6 La ecuación de estado de una Red de Petri está definida por: $M_{k+1} = M_k + Cv_k$, donde $v_k(i) = 0$, i≠j, v_k(j) = 1. Este comportamiento es representado por la función $M_k \xrightarrow{t_j} M_{k+1}$.

Ejemplo 2.4: Considere la Figura 2.1, y $M_1 = [2 \quad 0 \quad 0 \quad 1 \quad 0 \quad 0]^T$. El marcado M_2 es alcanzado desde el marcado M_1 cuando la transición t_1 es disparada, usando la ecuación de estado se tiene que:

$$\underset{M_1}{\begin{bmatrix}2\\0\\0\\1\\0\\0\end{bmatrix}} + \underset{C}{\begin{bmatrix}-2 & 0 & 0 & 0 & 2\\ 1 & -1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0 & -1\\ 0 & 0 & -1 & 0 & 1\\ 0 & 0 & 1 & -1 & 0\\ 0 & 0 & 0 & 3 & -3\end{bmatrix}} \underset{v_k}{\begin{bmatrix}1\\0\\0\\0\\0\end{bmatrix}} = \underset{M_1}{\begin{bmatrix}2\\0\\0\\1\\0\\0\end{bmatrix}} + \underset{Cv_k}{\begin{bmatrix}-2\\1\\0\\0\\0\\0\end{bmatrix}} = \underset{M_2}{\begin{bmatrix}0\\1\\0\\1\\0\\0\end{bmatrix}}$$

Definición 2.7 El conjunto de alcanzabilidad de una RP es el conjunto de todos los marcados alcanzables a partir del marcado inicial M_0 con el disparo solo transiciones habilitadas; este conjunto es representado por $R(G,M_0)$.

Definición 2.8 El vector característico o vector de Parikh de una secuencia de transiciones σ es un vector $\vec{\sigma} \in \{\mathcal{N}^+\}^m$, donde $m = |T|$. $\vec{\sigma}(i)$ representa el número de disparos de t_i en σ.

Ejemplo 2.5. Considera la secuencia $\sigma_1 = t_1t_2t_3t_4t_5t_3t_1t_2$ disparada en la RP de la Figura 2.1, el vector de Parikh de σ_1 es $\overrightarrow{\sigma_1} = [2 \quad 2 \quad 2 \quad 1 \quad 1]$.

Se tiene que tomar en cuenta que el vector de Parikh no proporciona información acerca del orden de ocurrencia de las transiciones en la secuencia σ.

2.1.4. Propiedades de las Redes de Petri

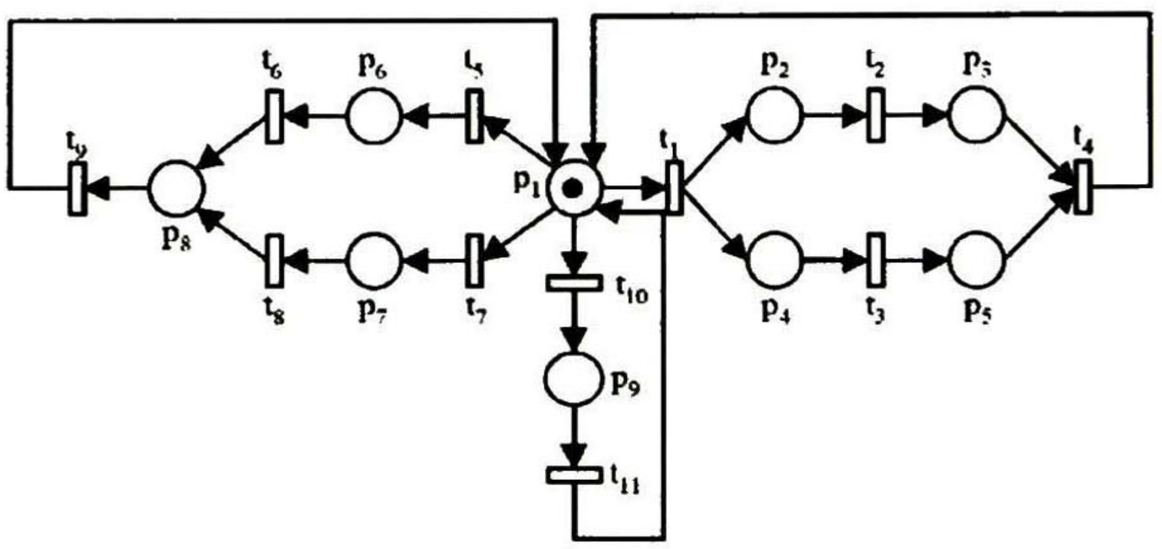


Figura 2.2 Red de Petri

Definición 2.9 Un p-invariante de una RP es un vector no negativo Y que cumple con $Y^T C = 0$. $\langle Y \rangle = \{p_i | Y(i) > 0\}$ es llamado el soporte de Y. Un p-componente es una subred generada por el soporte del p-invariante Y considerando también las transiciones de entrada y salida de cada lugar en $\langle Y \rangle$.

Ejemplo 2.6 El vector $Y_1 = [1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1]$ es un p-invariante de la Red de Petri G de la Figura 2.2. El soporte de Y_1 es $\langle Y_1 \rangle = \{p_1, p_2, p_3, p_6, p_7, p_8, p_9\}$. La subred G_1 generada por el p-invariante Y_1 es:

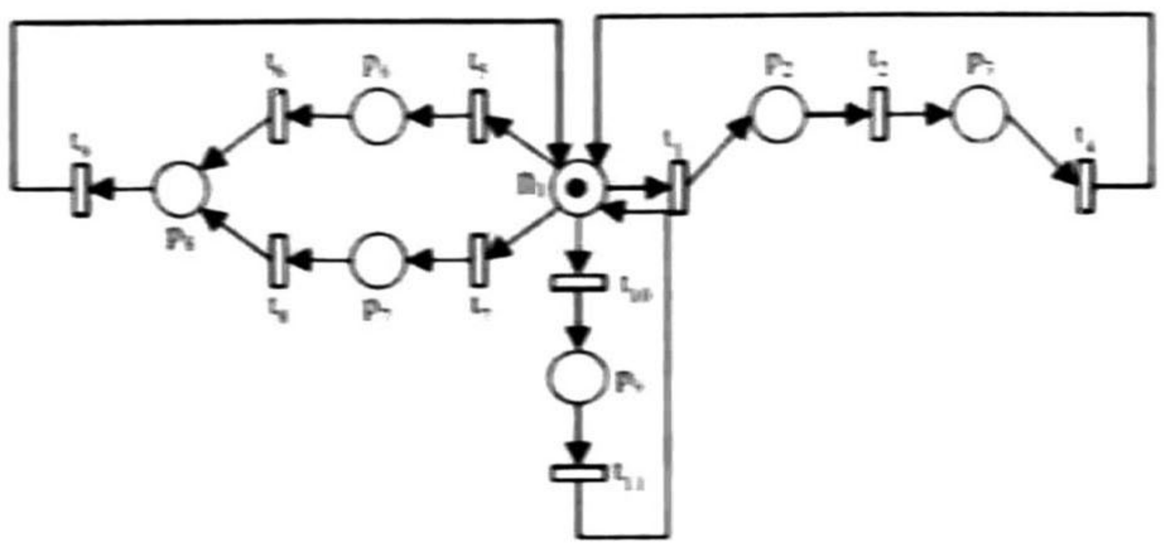


Figura 2.3 Subred generada por el p-invariante Y_1.

Definición 2.10 Un t-invariante de una Red de Petri es un vector no negativo X que cumple $CX = 0$. $\langle X \rangle = \{t_i | X(i) > 0\}$ es llamado el soporte de X. Un t-componente es una subred generada por el soporte de un t-invariante X considerado los lugares de entrada y salida de cada transición en $\langle X \rangle$.

Ejemplo 2.7 El vector $X_1 = [1 \quad 1 \quad 1 \quad 1 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0 \quad 0]^T$ es un t-invariante de la RP G de la Figura 2.2. El soporte de X_1 es $\langle X_1 \rangle = \{t_1, t_2, t_3, t_4\}$. La subred generada por X_1es:

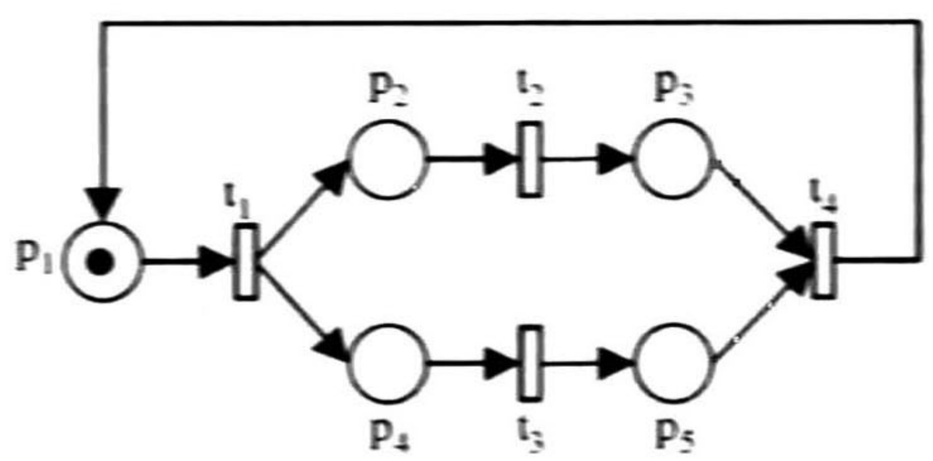


Figura 2.4 Subred generada por el t-invariante X_1.

2.1.5. Redes de Petri vivas y acotadas

Dos propiedades importantes de los SED es la acotamiento y la vivacidad. La vivacidad (*liveness*) es una propiedad que está relacionada con la ausencia de situaciones de bloqueos (*deadlocks*) y/o con el funcionamiento de la totalidad del sistema representado por una RP. El acotamiento (*boundedness*) es el número máximo de marcas que un lugar puede acumular, o bien si existen lugares en los cuales el marcado crece indefinidamente.

Definición 2.11 Un lugar $p_i \in P$ de una RP con un marcado inicial μ_0 esta k-acotado si y solo si $\forall \mu \in R(G, \mu_0)$, $\mu(p_i) \leq k$. Si k es el número de marcas más grande de la red se dice que esta *k-acotada*.

Una red está a salvo si y solo si está *k-acotada* con $k = 1$. En una RP un bloqueo significa que, para un marcado dado, ninguna transición puede ser disparada. Por otro lado, una RP que no se bloquea puede evolucionar sólo en una parte de ella. En cualquiera de estos dos casos se dice que la RP no es viva.

Definición 2.12 Se dice que una transición t_j es una RP es potencialmente disparable en un marcado μ, si existe un marcado $\mu' \in R(\mu, G)$ que la habilita. Una transición está viva si es potencialmente disparble para cada marcado $\mu \in R(\mu_0, G)$.

Definición 2.13 Una RP G está viva respecto a un marcado inicial μ_0 si y solo si todas sus transiciones están vivas.

Otra propiedad deseada de un SED es que presente un comportamiento cíclico. Un SED tiene un comportamiento cíclico si existe una secuencia de eventos que permite llegar al estado inicial desde cualquier estado alcanzable, significa que una tarea puede ser infinitamente realizada. En términos de RP un comportamiento cíclico se define como:

Definición 2.14 Una RP G es cíclica si $\forall \mu \in R(\mu_0, G), \exists \sigma$ tal que $\mu \rightarrow^{\sigma} \mu_0$

2.1.6. Redes de Petri Interpretadas

En el modelado de SED con RP, las transiciones, los lugares y las marcas adquieren cierto significado de acuerdo al contexto del sistema que se desea modelar. Las marcas representan la disponibilidad de recursos, la orden de ejecución de operaciones, la información transmitida, etc.; los lugares pueden representar recursos, estados parciales, etapas del proceso, operaciones. Las transiciones tienen asociados eventos tales como el inicio o fin (o ambos) de actividades, comandos, o información relevante.

Definición 2.15 Una Red de Petri Interpretada (Q, M_0) es una estructura de red $Q = (G, \Sigma, \Phi, \lambda, \varphi)$ con un marcado inicial M_0.

- G es una estructura de RP, $\Sigma = \{\alpha_1, \alpha_2, \ldots, \alpha_r\}$ es el alfabeto de entrada, y $\Phi = \{\phi_1, \phi_2, \ldots, \phi_q\}$ es el alfabeto de salida.
- $\lambda: T \rightarrow \Sigma \cup \{\varepsilon\}$ es la función de etiquetado de las transiciones, donde ε representa un evento interno del sistema que no se puede controlar externamente.
- $\varphi: R(Q, M_0) \rightarrow (\mathbb{Z}^+)^q$ es una función de salida, que asocia a cada marcado en $R(Q, M_0)$ un q-entry vector de salida; $q = |\Phi|$ es el número de salidas. φ está representada por una matriz $q \times |P|$, tal que el símbolo de salida ϕ_i es presentado cada vez que $M(p_j) \geq 1$, entonces $\varphi(i, j) = 1$, en otro caso $\varphi(i, j) = 0$.

Ejemplo 2.8 El sistema mostrado en la Figura 2.5 está compuesto de dos vagones que se desplazan sobre vías independientes por acción de las señales D_i (movimiento a la derecha) e I_i (movimiento a la Izquierda) donde i= 1,2. Las posiciones extremas son detectadas por a, c, d y f. Un botón M sirve para comenzar la operación del sistema desde su posición inicial: cuando es oprimido ambos vagones parten hacia la derecha; al llegar a su extremo derecho el primer vagón invierte su movimiento; al activar el botón b, el vagón parte nuevamente hacia la derecha hasta alcanzar el extremo derecho; ahí cambia su movimiento a la izquierda para regresar a su posición original. El segundo vagón simplemente se desplaza a la derecha; al activar el detector f, se desplaza a la izquierda para regresar a su posición inicial. Sólo se puede iniciar un nuevo ciclo cuando los dos vagones estén detenidos en su posición inicial. Las velocidades de los vagones pueden ser cualesquiera.

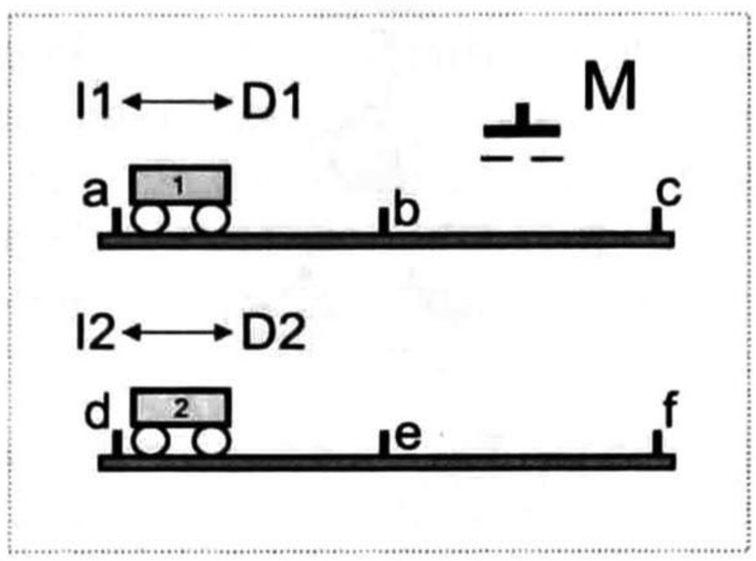


Figura 2.5 Sistema de Eventos Discretos

La RP de la figura 2.6 modela el comportamiento anteriormente descrito en base a la siguiente interpretación :

(ε): Vagón 1 y vagón 2 en reposo.
(D_1): Vagón 1 desplazándose a la derecha.

- I_1: Vagón 1 desplazándose a la izquierda.
- D_2: Vagón 2 desplazándose a la derecha.
- I_2: Vagón 2 desplazándose a la izquierda.

t_1: Se apoya sobre el botón M

- t_2: Detector c oprimido
- t_3: Detector f oprimido
- t_4: Detector a oprimido
- t_5: Detector d oprimido

t_6: Detector b oprimido

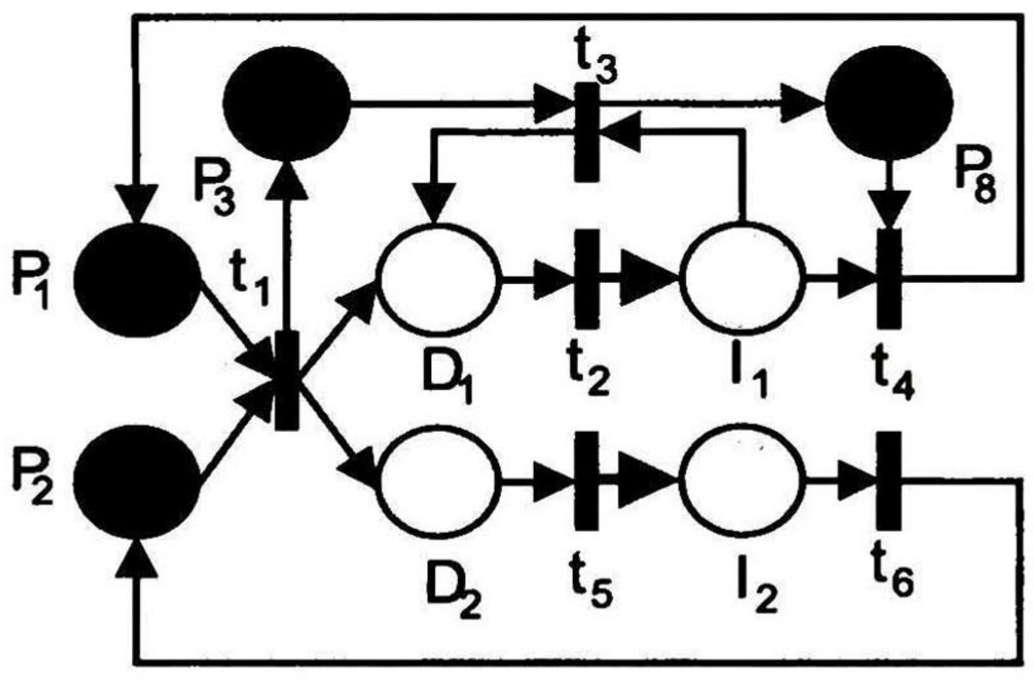


Figura 2.6 Red de Petri que modela el comportamiento de la Figura 2.5

En la figura 2.7 se muestra el grafo de alcanzabilidad del modelo anterior, donde se muestran las entradas y salidas del sistema.

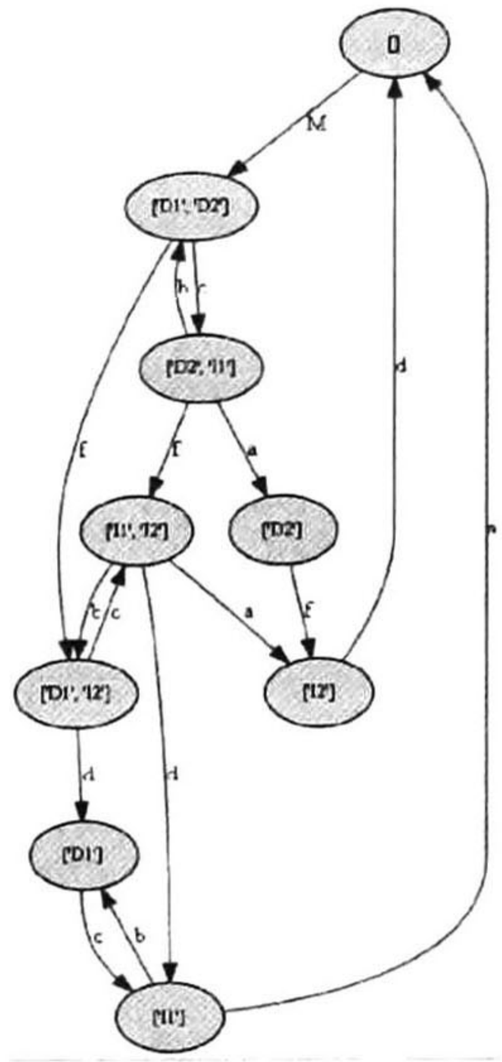


Figura 2.7 Grafo de alcanzabilidad para la RP de la Figura 2.6

2.2. Identificación de procesos de eventos discretos

Para este trabajo se consideran Sistemas de Control de lazo cerrado (Figura 2.8); que consisten en una planta y su controlador industrial (en la mayoría de las veces, un Controlador Lógico Programable: PLC).

Los PLCs son computadoras diseñadas para trabajar en ambientes industriales con la finalidad de controlar una amplia gama de procesos productivos. Los procesos productivos industriales que corresponden a los Sistemas de Eventos Discretos, se implementan usualmente con PLCs.

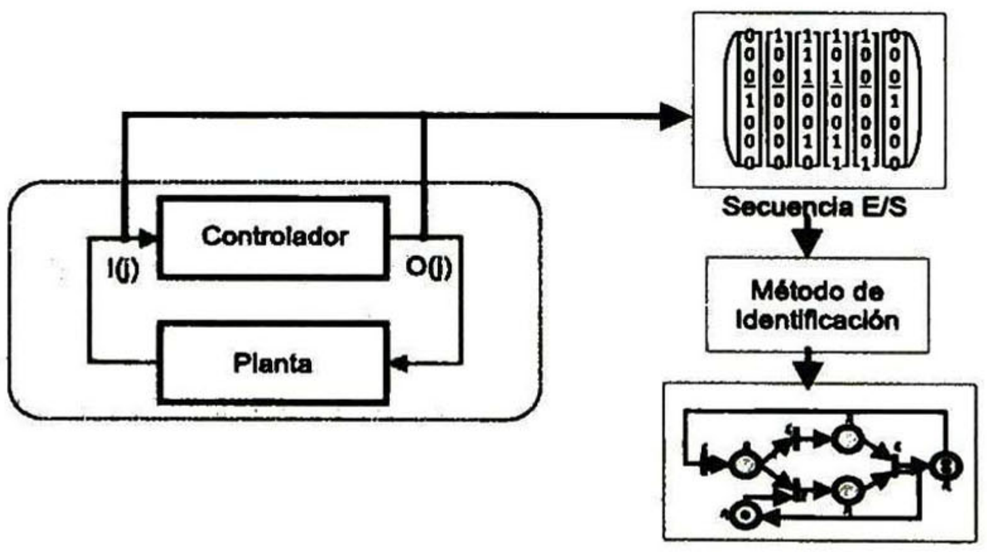


Figura 2.8 Sistema de Control de lazo Cerrado

Aunque existe el interés teórico de la definición de los métodos para la construcción de los modelos de secuencias de símbolos, el objetivo en esta tesis es desarrollar métodos de minería de sistemas automatizados industriales reales establece los retos relacionados con algoritmos escalables y cuestiones tecnológicas: las técnicas deben ser eficientes para enfrentar sistemas grandes y complejos que manejan señales entrada-salida.

Para este trabajo, el objetivo es descubrir, a partir de observaciones de comportamiento del sistema expresado como una sola secuencia w, y cómo los componentes del sistema están relacionados, luego construir un modelo compacto que representa el comportamiento observable, que junto con algoritmos ya desarrollados [Tapia, 2014] para encontrar el modelo no observable, se pueda obtener el modelo final que corresponde al comportamiento observado del sistema.

La minería de procesos ya existentes implica dos aspectos importantes a considerar: el proceso de observación y la operación del sistema. Son cuestiones que tienen que ser consideradas en los algoritmos propuestos con el fin de construir modelos adecuados.

La identificación se realiza desde el punto de vista del PLC. Se presentan varios fenómenos, a causa de la interacción entre la planta y el controlador, que aumentan la complejidad del proceso de identificación. Sin embargo, tales fenómenos deben de tomarse en cuenta, como son:

- El cambio en la entrada (señal emitida por la planta a través de un sensor) no siempre provoca un cambio en la salida (señal emitida por el PLC a un accionador). En el sistema real, algunos de los cambios de entrada provocan que exista un cambio en la salida.
- Al contrario de los supuestos ya establecidos para la teoría de los SED, muchas de las señales de entrada-salida pueden ocurrir simultáneamente; por otra parte, se observa simultáneamente cambios entrada-salida no simultáneos.
- Cuando los cambios en la salida son provocados por los cambios en las entradas, esta relación causal no está necesariamente capturada simultáneamente.

Ahora, se va a explicar estos fenómenos.

2.3. Técnica de Identificación

El trabajo realizado por Estrada [Estrada, 2013] aborda el problema de la identificación de sistemas de eventos discretos grandes y complejos. Uno de los métodos presentados permite construir de manera sistemática un modelo RPI desde una única secuencia de entrada-salida que representa el comportamiento observable. El método consiste de dos etapas; la primera calcula, a partir de la secuencia entrada-salida, el comportamiento reactivo del modelo, la cual obtiene los lugares observables y transiciones del sistema. La segunda etapa construye la parte no observable del modelo incluyendo lugares que aseguran la reproducción total de la secuencia de entrada-salida.

Los procesos que se desean descubrir son sistemas compuestos por un controlador y su planta (PLC) operando en lazo cerrado. La Figura 2.9 muestra el procedimiento a grandes rasgos de la identificación del sistema vista desde este enfoque.

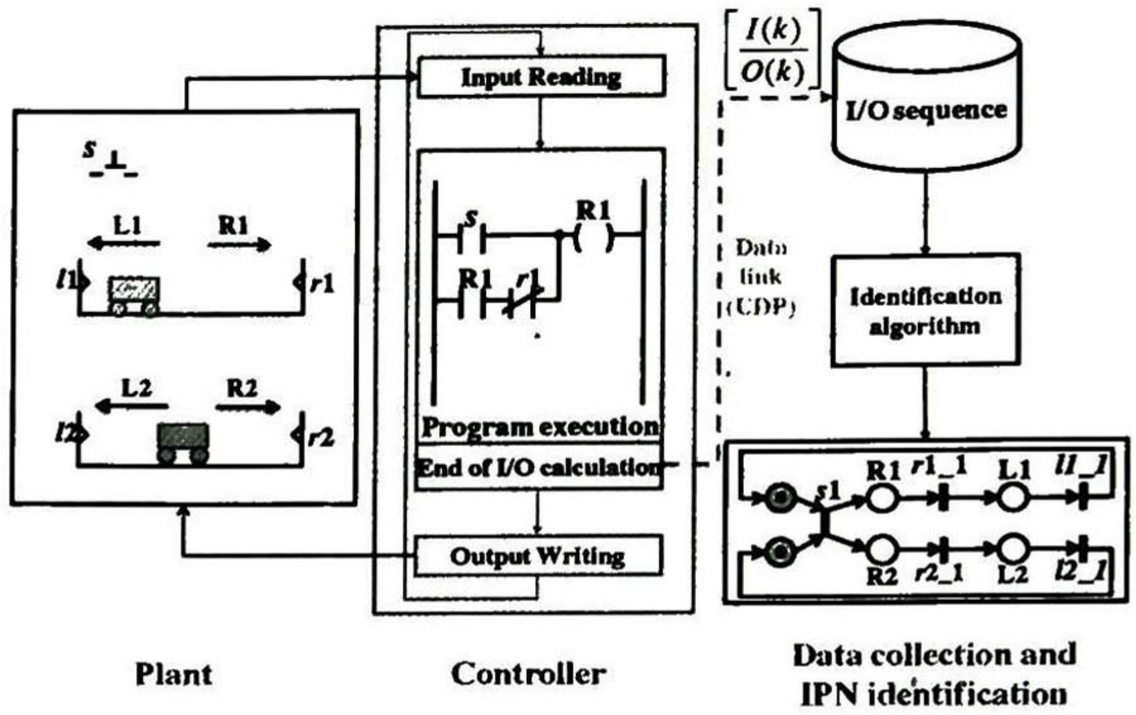


Figura 2.9 PLC + Planta y proceso de identificación [Estrada, 2012]

La Figura 2.9 describe un sistema en el cual encontramos señales de entrada, que son resultado de las salidas de la planta, y señales de salida, que son las siguientes entradas a la planta. Se consideran sistemas con r señales de entrada y q salidas, donde $m, n \in \mathbb{Z}^+$ Los datos de entrada que se utilizan para el proceso de identificación están incluidos en una única secuencia de vectores de E/S w (2.1), donde $I(j)$ y $O(j)$ son respectivamente los valores de las entradas y las salidas en el ciclo del PLC.

Durante el análisis de la evolución de las señales se necesita trabajar con vectores de eventos (2.2), los cuales se definen como el resultado de la diferencia entre dos vectores E/S consecutivos:

$$E(k) = w(k+1) - w(k) \quad (2.1)$$

Los vectores de eventos se pueden descomponer como vectores de eventos de entrada o de salida. (3.3)

$$w = \left[\frac{I(1)}{O(1)}\right], \left[\frac{I(2)}{O(2)}\right], \left[\frac{I(3)}{O(3)}\right], \ldots \quad (2.1) \qquad E(k) = \left[\frac{IE(k)}{OE(k)}\right] \quad (2.2)$$

Tres tipos de situaciones pueden ocurrir durante el cambio observado en los vectores de eventos:

- $IE(j) \neq 0$ y $OE(j) \neq 0$: *Un cambio en la entrada provoca directamente un cambio a la salida;*
- $IE(j) = 0$ and $OE(j) \neq 0$: *El controlador llega en el evento j-1 a un estado en el cual, dados los valores de entrada, una evolución en la salida es observada en el evento j.*
- $IE(j) \neq 0$ and $OE(j) = 0$: *El controlador no es sensible a un cambio en la entrada. O provoca un cambio de estado que no es observado en un cambio a la salida.*

El método propuesto en [Estrada, 2013] opera en dos pasos:

Paso 1: Descubrir el comportamiento reactivo de entradas y salidas, donde la parte observable de la red es construida como pequeños fragmentos de red, los cuales tienen como etiqueta de sus transiciones, expresiones algebraicas que indican las variables de entrada del sistema.

Paso 2: Inferir la parte no observable de la red, para ello la secuencia w es transformada en una secuencia S de disparos de las transiciones observables.

En esta tesis nos enfocamos en especial en el paso 1 y el objetivo es obtener un método alternativo más simple.

Ejemplo 2.9 Considere el sistema de manufactura mostrado en la Figura 2.10 [Estrada, 2012], el objetivo del sistema es ordenar paquetes de acuerdo a su tamaño, este tiene 9 entradas en total(k 1, k2, a0, a1, a2, b0, b1, c0, c1) de las cuales 7 señales son generadas por sensores para detectar la posición de los cilindros (a0, a1, a2, b0, b1, c0, c1) y 2 señales que indican la presencia de los paquetes (k1 ,k2). Cuatro señales de salida que controlan los actuadores de la planta (A+, A-, B, C), la forma en que se presentan los vectores E/S es la siguiente [A+ A- B C k1 k2 a0 a1 a2 b0 b1 c0 c1]T. Se procesa una secuencia de 222 vectores de E/S.

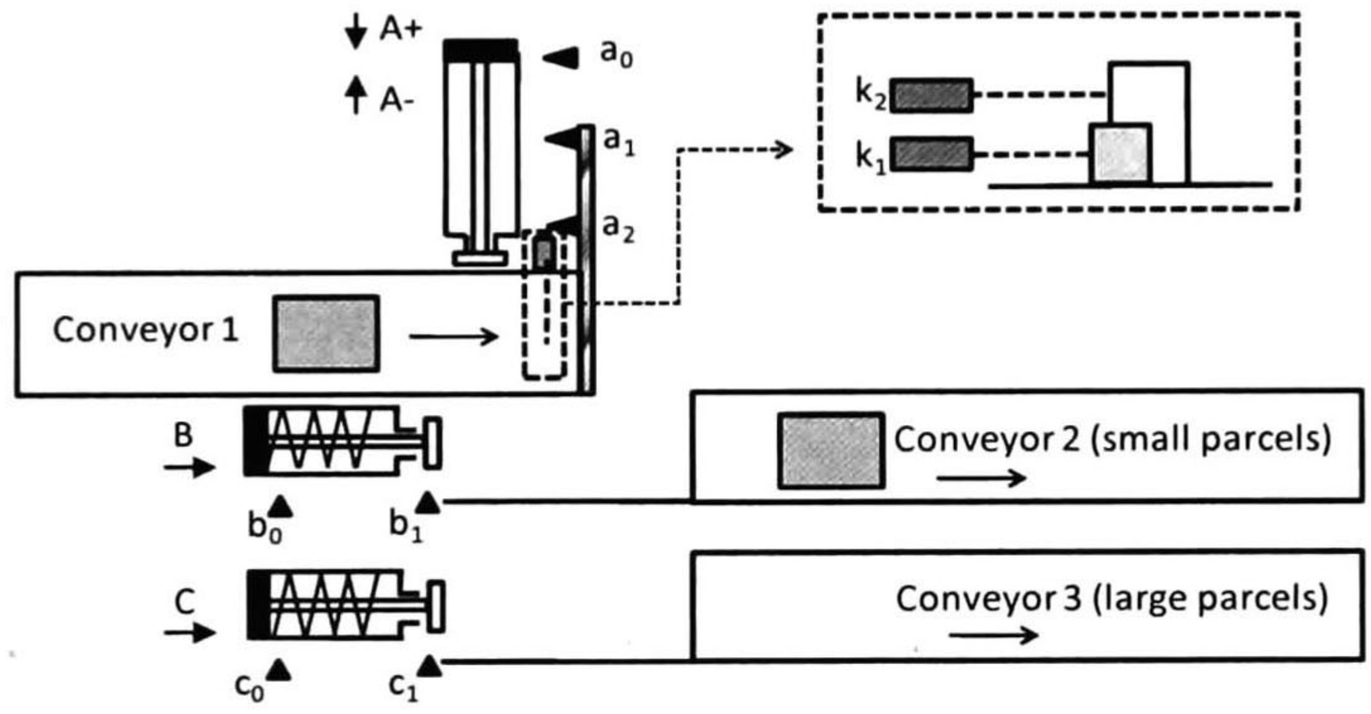


Figura 2.10 Sistema de manufactura Ejemplo 2.1 [Estrada, 2013]

Se presenta el algoritmo para la identificación de eventos discretos:

Algoritmo 2.1 Descripción general del paso 1

Entrada: Secuencia E/S w

Salidas: Matriz de incidencia Observable φC,

---

1. Analizar la secuencia w con el fin de:
   - Procesar la secuencia de vectores evento y los eventos elementales.
   - Procesar la Matriz Directa e Indirecta de Causalidad.
   - Construcción de las Funciones de Disparo desde las matrices.
   - Encontrar los eventos de entrada con influencia diferida.
2. Usar w, Funciones de disparo y los eventos con influencia diferida con el fin de:
   - Procesar las transiciones de RPI y sus λ
   - Procesar la Matriz Observable de incidencia φC
   - Procesar la secuencia de transiciones S

0000001001010	0100100001010	0100010000110	1000010001010	0100010001000	1000000001010
1000101001010	0000101001010	0000011000010	1000000001010	0000011001000	0110000101010
1000100001010	1000101001010	1000011001010	1000000101010	1000011001010	0110000100010
1000000001010	1000100001010	1000010001010	1000000001010	1000010001010	0110000000010
0110000101010	1000000001010	1000000001010	0101000011010	1000000001010	0100000000110
0110000100010	0110000101010	1000000101010	0101000001010	1000000101010	0000101000110
0110000000010	0110000100010	1000000001010	0101000001000	1000000001010	0000101000010
0100000000110	0110010000010	0101000011010	0100000001001	0101000011010	1000101001010
0000101000110	0100010000110	0101000011000	0100000001000	0101000001000	1000100001010
0000101000010	0000011000010	0101000001000	0100000101000	0100000001001	1000000001010
1000101001010	1000011001010	0100000001001	0100000001000	0100000101000	0110000101010
1000100001010	1000010001010	0100000001000	0100010001010	0100000001010	0110000100010
1000000001010	1000000001010	0100000101000	0000011001010	0000001001010	0110000000010
0110000101010	1000000101010	0100000001000	1000011001010	1000011001010	0100000000110
0110000100010	1000000001010	0100000001010	1000010001010	1000010001010	0000101000110
0110000000010	0101100011010	0000001001010	1000000001010	1000000001010	0000101000010
0100000000110	0101100001010	1000011001010	1000000101010	1000000101010	1000101001010
0000101000110	0101100001000	1000010001010	1000000001010	1000000001010	1000100001010
0000101000010	0100100001001	1000000001010	0101000011010	0101100011010	1000000001010
1000101001010	0100100101000	1000000101010	0101000001010	0101100001010	0110000101010
1000100001010	0100100001010	1000000001010	0101000001000	0101100001000	0110000100010
1000000001010	0000101001010	0101100011010	0100000001001	0100100001001	0110000000010
0110000101010	1000101001010	0101100001010	0100000001000	0100100101000	0010001000010
0110000100010	1000100001010	0101100001000	0100000101000	0100100001010	0000001000110
0110010000010	1000000001010	0100100001001	0100000001000	0000101001010	0000001000010
0100010000110	0110000101010	0100100101000	0100010001010	1000101001010	0000001001010
0000011000010	0110000100010	0100100001010	0000011001010	1000100001010	0000101001010
1000011001010	0110000000010	0000101001010	1000011001010	1000000001010	1000101001010
1000010001010	0100000000110	1000101001010	1000010001010	0110000101010	1000100001010
1000000001010	0000101000110	1000100001010	1000000001010	0110000100010	1000000001010
1000000101010	0000101000010	1000000001010	1000000101010	0110000000010	0110000101010
1000000001010	1000101001010	0110000101010	1000000001010	0010001000010	0110000100010
0101100011010	1000100001010	0110000100010	0101000011010	0000001000110	0110000000010
0101100001010	1000000001010	0110010000010	0101010001010	0000001000010	0100000000110
0101100001000	0110000101010	0100010000110	0101010001000	0000001001010	0000101000110
0100100001001	0110000100010	0000011000010	0100010001001	1000101001010	0000101000010
0100100101000	0110010000010	1000011001010	0100010101000	1000100001010	1000101001010

Figura 2.11 Secuencia de vectores de E/S del ejemplo 3.1

	A+_1	*A+_0*	*A-_1*	*A-_0*	*B_1*	*B_0*	*C_1*	*C_0*
k1_1	0.111	0.111	0.111	0.111	0.000	0.200	0.000	0.000
k1_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
k2_1	0.222	0.000	0.000	0.000	0.000	0.000	0.000	0.000
k2_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
a0_1	0.222	0.000	0.000	0.100	0.000	0.000	0.000	0.000
a0_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
a1_1	0.000	0.444	0.444	0.000	1.000	0.000	0.000	0.000
a1_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
a2_1	0.000	0.556	0.556	0.000	0.000	0.000	1.000	0.000
a2_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
b0_1	0.333	0.000	0.000	0.000	0.000	0.000	0.000	0.000
b0_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
b1_1	0.000	0.000	0.000	0.000	0.000	1.000	0.000	0.000
b1_0	0.000	0.000	0.000	0.111	0.000	0.000	0.000	0.000
c0_1	0.111	0.000	0.000	0.000	0.000	0.000	0.000	0.000
c0_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000
c1_1	0.000	0.000	0.000	0.000	0.000	0.000	0.000	1.000
c1_0	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000

Tabla 2.1 Matriz Directa Causal (MDC) para el ejemplo 3.1

	A+_1	*A+_0*	*A-_1*	*A-_0*	*B_1*	*B_0*	*C_1*	*C_0*
k1=1	0.444	0.111	0.111	0.333	0.000	0.250	0.200	0.200
k1=0	0.556	0.889	0.889	0.667	1.00	0.750	0.800	0.800
k2=1	0.556	0.000	0.000	0.333	0.000	0.250	0.000	0.200
k2=0	0.444	1.000	1.000	0.667	1.000	0.750	1.000	0.800
a0=1	1.000	0.000	0.000	1.000	0.000	0.500	0.000	0.000
a0=0	0.000	1.000	1.000	0.000	1.000	0.500	1.000	1.000
a1=1	0.000	0.444	0.444	0.000	1.000	0.000	0.000	0.000
a1=0	1.000	0.556	0.556	1.000	0.000	1.000	1.000	1.000
a2=1	0.000	0.556	0.556	0.000	0.000	0.000	1.000	1.000
a2=0	1.000	0.444	0.444	1.000	1.000	1.000	0.000	1.000
b0=1	1.000	1.000	1.000	0.556	0.000	0.000	1.000	1.000
b0=0	0.000	0.000	0.000	0.444	1.000	1.000	0.000	0.000
b1=1	0.000	0.000	0.000	0.111	1.000	1.000	0.000	0.000
b1=0	1.000	1.000	1.000	0.889	0.000	0.000	1.000	1.000
c0=1	1.000	1.000	1.000	0.889	0.000	1.000	1.000	0.000
c0=0	0.000	0.000	0.000	0.111	1.000	0.000	0.000	1.000
c1=1	0.000	0.000	0.000	0.000	1.000	0.000	0.000	1.000
c1=0	1.000	1.000	1.000	1.000	0.000	1.000	1.000	0.000

Tabla 2.2 Matriz Indirecta Causal (MIC) para el Ejemplo 2.9

Se toma en cuenta la segunda columna de la DCM correspondiente al evento A+_0. Las posibles entradas para estar en relación causal con A+_0 son k1_1, a1_1 y a2_1. Observe que $Prob(A+\_0|a1\_1) + Prob(A+\_0|a2\_1) = 1$. Al considerar la segunda columna de la matriz IMC, podemos verificar que $Prob(A+\_0|a1=1) + Prob(A+\_0|a2=1) = 1$, podemos concluir que el evento A+_0 es causado a veces por a1_1 y otras causadas por a2_1.

Se analiza también la primera columna de la matriz DCM que corresponde al evento de salida A+_1. Los candidatos de entrada para estar en una relación causal con un evento de salida son k1_1, k2_1, a0_1, b0_1 y c0_1. Se observa que $Prob(A+\_1|k1\_1) + Prob(A+\_1|k2\_1) + Prob(A+\_1|a0\_1) + Prob(A+\_1|b0\_1) + Prob(A+\_1|c0\_1) = 1$, pero al observar al mismo tiempo la segunda columna de la IMC, se observa que $Prob(A+\_1|k1=1) + Prob(A+\_1|k2=1) + Prob(A+\_1|a0=1) + Prob(A+\_1|b0=1) + Prob(A+\_1|c0=1) \neq 1$. Sin embargo, al observar $Prob(A+\_1|k1=1) + Prob(A+\_1|k2=1) = 1$ y que $Prob(A+\_1|a0=1) = 1$, $Prob(A+\_1|b0=1) = 1$, y $Prob(A+\_1|c0=1) = 1$.

Por lo tanto, se puede concluir que el evento A+_1 siempre ocurre bajo condiciones a0=1, b0=1, c0=1 y que puede ser a veces ajo la condición de k1=1 y otras veces por k2=1.

Todas las condiciones que se encuentran en las tablas se calcularán formalmente con el método de Estrada [Estrada, 2013], las funciones obtenidas se dan a continuación en la Tabla 2.3.

OE_K	$G(OE_K)$	$F(OE_K)$	$\chi(OE_k)$
A+_1	(ε)	$((k1 \oplus k2) \wedge a0 \wedge b0 \wedge c0)$	$(\varepsilon)\bullet ((k1 \oplus k2) \wedge a0 \wedge b0 \wedge c0)$
A+_0	$(a1\_1 \oplus a2\_1)$	$(=1)$	$(a1\_1 \oplus a2\_1) \bullet (=1)$
A−_1	$((a1\_1 \oplus a2\_1))$	$(=1)$	$((a1\_1 \oplus a2\_1)) \bullet (=1)$
A−_0	$(a0\_1)$	$(=1)$	$(a0\_1) \bullet (=1)$
B_1	$(a1\_1)$	$(=1)$	$(a1\_1) \bullet (=1)$
B_0	$(b1\_1)$	$(=1)$	$(b1\_1) \bullet (=1)$
C_1	$(a2\_1)$	$(=1)$	$(a2\_1) \bullet (=1)$
C_0	$(c1\_1)$	$(=1)$	$(c1\_1) \bullet (=1)$

Tabla 2.3 Funciones de disparo del Ejemplo 2.9

Las funciones de disparo muestran claramente las condiciones en las que las entradas provocan cambios en las salidas. Pueden ser expresadas en términos de RPI como lugares observables marcados o desmarcados. La Figura 2.12 representa gráficamente los fragmentos correspondientes a las funciones anteriores.

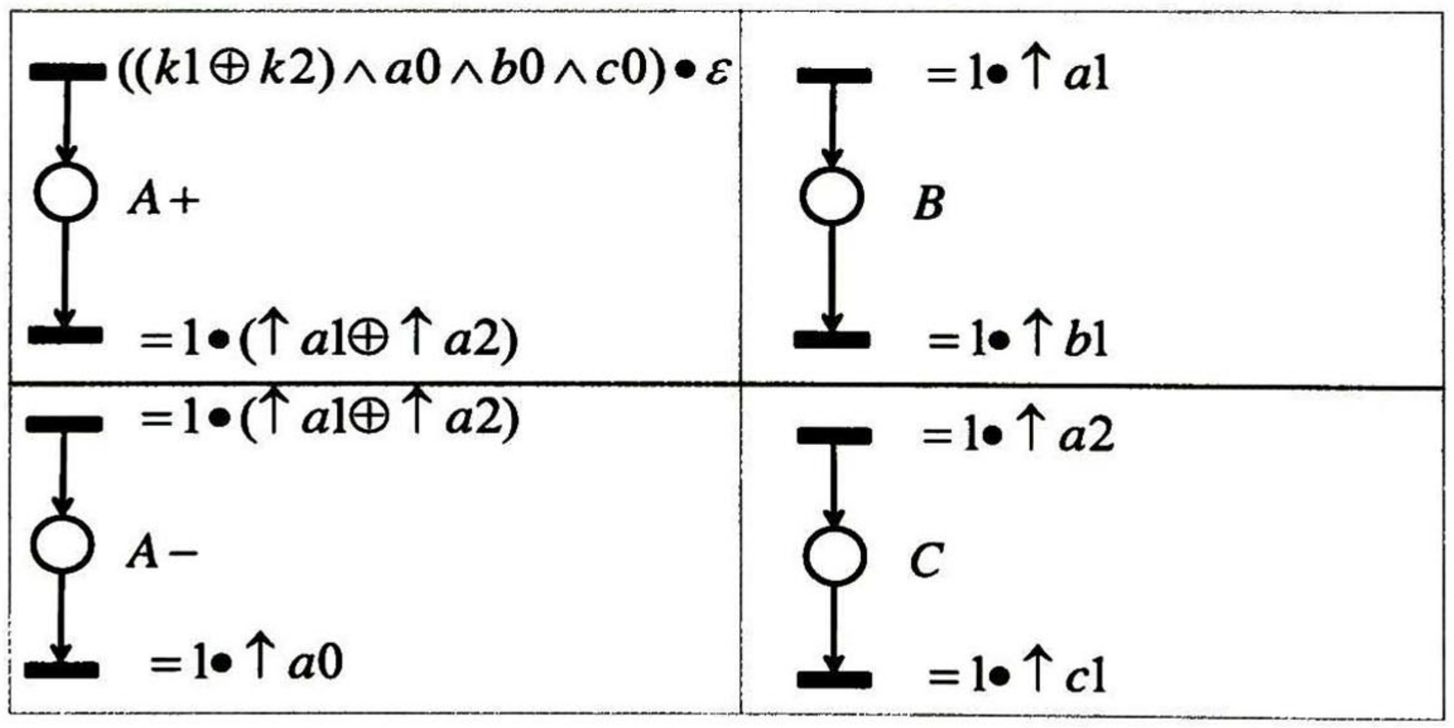


Figura 2.12 Fragmentos de RPI para el Ejemplo 2.9 [Estrada, 2013]

Observe que la condición $DCM_{xj} \neq 0$, $DCM_{yj} \neq 0$,..., $DCM_{zj} \neq 0$ exige que se observen las entradas relacionadas con el cambio de salida al menos una vez, para que se considere el cambio de su valor en el mismo ciclo de PLC. Esta condición puede ser restrictiva si la relación de entrada y salida no se observa en el mismo vector de evento. Por ejemplo, si no

hay un sensor de entrada para indicar que un empujador se ha retraído, puede haber algunas temporizaciones de seguridad que no permiten otro accionador reaccionar en el momento en una condición de entrada se ha llevado a cabo, y se observa el cambio más adelante.

Algoritmo 2.2 Funciones de Disparo

Entrada: *MDC* y *MIC*

Output: $\chi(OE_k)$

$\forall\ OE_k$

1) Realiza $G(OE_k) \leftarrow \varepsilon$
   Procesa $G(OE_k) \leftarrow \Pi DisjE_j$

   Con $DisjE_j \leftarrow (IE_x \oplus IE_y \oplus ... \oplus IE_z)$

   Tal que

   1. $DCM_{xj} \neq 0, DCM_{yj} \neq 0, ... DCM_{zj} \neq 0$
   2. $DCM_{xj} + DCM_{yj} + ... + DCM_{zj} = 1$

2) Realiza $F(OE_k) \leftarrow (=1)$, $\kappa \leftarrow 0$
   Mientras ($G(OE_k) = \varepsilon$ y $F(OE_k) = (=1)$)
   Procesa $F(OE_k) \leftarrow \Pi DisjL_j$

   Con $DisjL_j = (IL_x \oplus IL_y \oplus ... \oplus IL_z)$

   Tal que

   1. $\kappa - DCM_{xj} \neq 0, \kappa - DCM_{yj} \neq 0, ... \kappa - DCM_{zj} \neq 0$
   2. $ICM_{xj} + ICM_{yj} + ... + ICM_{zj} = 1$

   Realiza $\kappa \leftarrow \kappa + 1$

Se analizara la primera parte del método que es la construcción del comportamiento observable del sistema y como los eventos en las entradas producen cambios en las salidas inmediatos o diferidos.

Algoritmo 2.3 Construir el comportamiento observable

Entrada: E/S Secuencia *w*, Secuencia de Eventos *E*, Entradas diferidas *D*, Funciones de Disparo $\chi(OE_k)$

Salida: Matriz de incidencia Observable φC, función de etiquetado λ, secuencia *S*

1) Crear una fila en la matriz de incidencia para cada salida del Sistema.
2) $S \leftarrow \varepsilon$

3) $\forall E(j)$ Considera la secuencia E/S y la secuencia de eventos:
   a) Si $OE(j)=0$ y $IE(j)$ contienen eventos elementarios de $IE_s,\ldots,IE_u$ pertenecientes a D
      - Si no se ha creado antes, crear una nueva transición cero T_j (una columna cero en la matriz de incidencia φC) tal que $\lambda(T_j) \leftarrow IE_s,\ldots,IE_u$
      - $S \leftarrow S \cdot T_j$
   b) Si $OE(j) \neq 0$
      - Considera el evento de salida $OE(j) = OE_{jp} \bullet OE_{jq} \bullet OE_{jr}$ incluido en $E(j)$
      - Calcular una nueva función de disparo $\chi(OE_{jp})$, $\chi(OE_{jq})$, $\chi(OE_{jr})$:

      i) Para cada $\chi(OE_{jk}) = G(OE_{jk}) \bullet F(OE_{jk})$:

      (1) Para cada $DisjE_i = (IE_x \oplus IE_y \oplus \ldots \oplus IE_z)$ en $G(OE_{jk})$ observer en $IE(j)$ el evento de entrada IE_{ik} que se cumpla $DisjE_i = (IE_x \oplus IE_y \oplus \ldots \oplus IE_z)$ y $DisjE_i' \leftarrow IE_{ik}$

      (2) $G'(OE_{jk}) \leftarrow \Pi DisjE_i'$

      (3) $G(OE(j)) \leftarrow G'(OE_{jp}) \bullet G'(OE_{jq}) \bullet \ldots \bullet G'(OE_{jr})$

      (4) Para cada $DisjL_i = (IL_x \oplus IL_y \oplus \ldots \oplus IL_z)$ in $F(OE_{jk})$ look into $w(j+1)$ los niveles de entrada IL_{ik} que cumplan $DisjL_i = (IL_x \oplus IL_y \oplus \ldots \oplus IL_z)$ y $DisjL_i' \leftarrow IL_{ik}$

      (5) $F'(OE_{jk}) \leftarrow \Pi DisjL_i'$

      (6) $F(OE(j)) \leftarrow F'(OE_{jp}) \bullet F'(OE_{jq}) \bullet \ldots \bullet F'(OE_{jr})$

      - Si no se ha creado antes, crear una nueva transición T_j (una nueva columna en la matriz de incidencia φC) tal que $\lambda(T_j) \leftarrow F(OE(j)) \bullet G(OE(j))$ y relacionarla con sus cambios de salida correspondientes.

      i) Para todos los eventos elementarios de salida en $OE(j) = OE_{jp} \bullet OE_{jq} \bullet OE_{jr}$, colocar un -1 en la línea correspondiente a OE_{jk} si este es un evento de bajada, o un 1 si este es un evento de subida; para el resto de las líneas, colocar 0.

      - $S \leftarrow S \cdot T_j$

La secuencia de transiciones S se crea mediante la concatenación progresiva de cada una de las transiciones calculadas. La complejidad del método para la construcción de dicha secuencia es de $O((n' \log m')h)$, donde n y m es el número máximo de entradas y salidas del vector de eventos. Por lo tanto el método implementado por Estrada [Estrada, 2013] se puede ejecutar en tiempo polinomial.

Ejemplo 2.10 Después del procesamiento de la secuencia de vectores de E/S, se pueden calcular las transiciones de la cuarta columna de la Tabla 2.4

Vector E/S	*Eventos elementarios de Entrada*	*Eventos elementarios de Salida*	*Tranisiciones Procesadas*
$w(1) = [00100101000000]^T$	$IE(1) = k1\_1$	$OE(1) = A+\_1$	$\chi(t_1) = (k1 \bullet a0 \bullet b0 \bullet c0) \bullet (\varepsilon)$
$w(2) = [10100101010000]^T$	$IE(2) = a0\_0$	$OE(2) = \varepsilon$	*Sin transición*
$w(3) = [10000101010000]^T$	$IE(3) = k1\_0$	$OE(3) = \varepsilon$	*Sin transición*
$w(4) = [00000101010000]^T$	$IE(4) = a1\_1$	$OE(4) = A+\_0 \bullet A-\_1 \bullet B\_1$	$\chi(t_2) = (=1) \bullet (a1\_1)$
$w(5) = [00010101001100]^T$	$IE(5) = b0\_0$	$OE(5) = \varepsilon$	*Sin transición*
$w(6) = [00010001001100]^T$	$IE(6) = a1\_0$	$OE(6) = \varepsilon$	*Sin transición*
$w(7) = [00000001001100]^T$	$IE(8) = k1\_1 \bullet a0\_1$	$OE(8) = A-\_0$	$\chi(t_3) = (=1) \bullet (a0\_1)$
$w(8) = [10100001000100]^T$	$IE(7) = b1\_1$	$OE(7) = B\_0$	$\chi(t_4) = (=1) \bullet (b1\_1)$
$w(9) = [10100011000000]^T$	$IE(9) = b1\_0$	$OE(9) = \varepsilon$	*Sin transición*
$w(10) = [10100001000000]^T$	$IE(10) = b0\_1$	$OE(10) = A+\_1$	$\chi(t_1) = (k1 \bullet a0 \bullet b0 \bullet c0) \bullet (\varepsilon)$
$w(11) = [10100101010000]^T$	$IE(11) = a0\_0$	$OE(11) = \varepsilon$	*Sin transición*
$w(12) = [10000101010000]^T$	$IE(12) = k1\_0$	$OE(12) = \varepsilon$	*Sin transición*
$w(13) = [00000101010000]^T$	$IE(13) = a1\_1$	$OE(13) = A+\_0 \bullet A-\_1 \bullet B\_1$	$\chi(t_2) = (=1) \bullet (a1\_1)$
$w(14) = [00010101001100]^T$	$IE(14) = b0\_0$	$OE(14) = \varepsilon$	*Sin transición*
$w(15) = [00010001001100]^T$	$IE(15) = a1\_0$	$OE(15) = \varepsilon$	*Sin transición*
$w(16) = [00000001001100]^T$	$IE(16) = b1\_1$	$OE(16) = B\_0$	$\chi(t_4) = (=1) \bullet (b1\_1)$
$w(17) = [00000011001000]^T$	$IE(17) = a0\_1$	$OE(17) = A-\_0$	$\chi(t_3) = (=1) \bullet (a0\_1)$
$w(18) = [00100011000000]^T$	$IE(18) = b1\_0$	$OE(18) = \varepsilon$	*Sin transición*
$w(19) = [00100001000000]^T$	$IE(19) = b0\_1$	$OE(19) = \varepsilon$	*Sin transición*
$w(20) = [00100101000000]^T$	$IE(20) = k2\_1$	$OE(20) = A+\_1$	$\chi(t_5) = (k2 \bullet a0 \bullet b0 \bullet c0) \bullet (\varepsilon)$
$w(21) = [01100101010000]^T$	$IE(21) = a0\_0$	$OE(21) = \varepsilon$	*Sin transición*
$w(22) = [01000101010000]^T$	$IE(22) = k2\_0$	$OE(22) = \varepsilon$	*Sin transición*
$w(23) = [00000101010000]^T$	$IE(23) = a1\_1$	$OE(23) = \varepsilon$	*Sin transición*
$w(24) = [00010101010000]^T$	$IE(24) = a1\_0$	$OE(24) = \varepsilon$	*Sin transición*
$w(25) = [00000101010000]^T$	$IE(25) = a2\_1$	$OE(25) = A+\_0 \bullet A-\_1 \bullet C\_1$	$\chi(t_6) = (=1) \bullet (a2\_1)$
$w(26) = [00001101001010]^T$	$IE(26) = a2\_0$	$OE(26) = \varepsilon$	*Sin transición*
$w(27) = [00000101001010]^T$	$IE(27) = c0\_0$	$OE(27) = \varepsilon$	*Sin transición*
$w(28) = [00000100000101]^T$	$IE(28) = k1\_1 \bullet c1\_1$	$OE(28) = C\_0$	$\chi(t_7) = (=1) \bullet (c1\_1)$
$w(29) = [10000100101000]^T$	$IE(29) = a1\_1$	$OE(29) = \varepsilon$	*Sin transición*
$w(30) = [10010100001000]^T$	$IE(30) = a1\_0 \bullet c0\_1$	$OE(30) = \varepsilon$	*Sin transición*
$w(31) = [10000101001000]^T$	$IE(31) = a0\_1$	$OE(31) = A-\_0$	$\chi(t_3) = (=1) \bullet (a0\_1)$
$w(32) = [10100101000000]^T$	From here, previously observed transitions are found		
...			

Tabla 2.4 Transiciones encontradas para el Ejemplo 2.8

Se procesó una secuencia de 222 vectores de E/S; al final de procedimiento, se obtiene la siguiente matriz de incidencia observable que representa la estructura de la Figura 2.13.

	t_1	t_2	t_3	t_4	t_5	t_6	t_7
A+	1	-1	0	0	1	-1	0
A-	0	1	-1	0	0	1	0
B	0	1	0	-1	0	0	0
C	0	0	0	0	0	1	-1

Por la concatenación de todas las transiciones procesadas en la cuarta columna de la Tabla 2.4 se obtiene la siguiente secuencia de transiciones:

S = t_1 t_2 t_3 t_4 t_1 t_2 t_4 t_3 t_5 t_6 t_7 t_4 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_4 t_1 t_2 t_3 t_4 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_4 t_5 t_6 t_7 t_4 t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_4 t_5 t_6 t_7 t_4 t_5 t_6 t_7 t_4 t_5 t_6 t_7 t_4 t_5 t_6 t_7 t_4 t_1 t_2 t_4 t_3 t_1 t_2 t_3 t_4 t_1 t_2 t_3 t_4 t_1 t_2 t_4 t_3 t_1 t_2 t_3 t_4 t_1

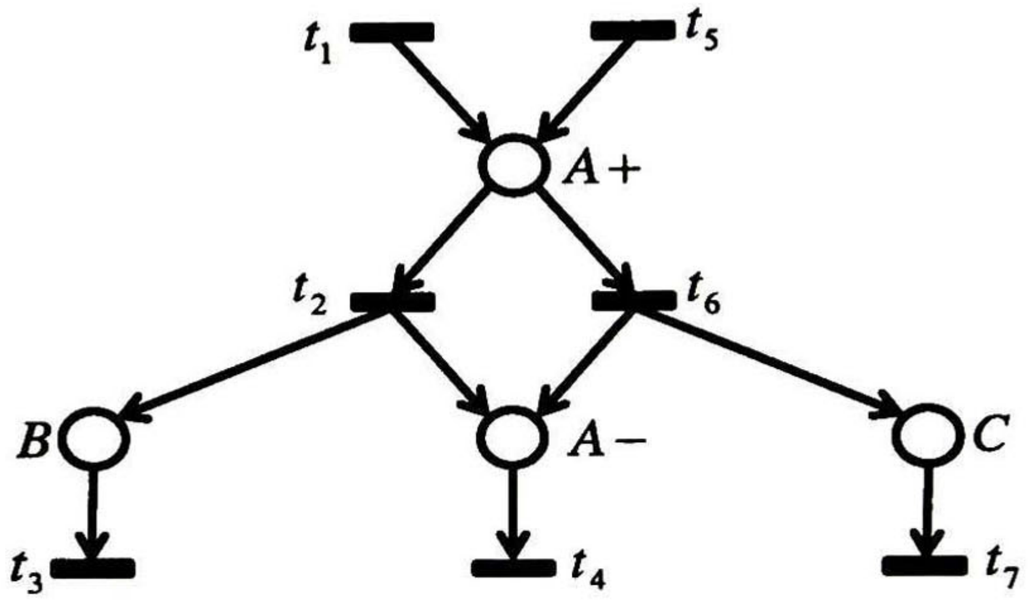


Figura 2.13 Modelo obtenido del

Ejemplo 2.10 [Estrada, 2012]

Después de realizar este proceso, sólo se ha definido la parte observable de la RPI, es decir, el comportamiento reactivo del sistema. Queda inferir las evoluciones de las entradas que provocan un cambio en el comportamiento no observable del controlador. Tal problema puede ser visto como el procesamiento de la secuencia de transiciones que se obtienen hasta este punto del método, para obtener los lugares no observables del modelo final de la RPI, que aseguren la ejecución de la secuencia S de transiciones.

2.4. Limitaciones del trabajo previo

A pesar de que la primera parte del método visto en [Estrada, 2013] es muy completa, en esta tesis se propone una manera alternativa más simple de realizar este proceso con el fin de aumentar la eficiencia de, A continuación se presenta la estrategia general del método propuesto en este trabajo; de manera esquemática y utilizando el caso de estudio del Ejemplo 2.9

A partir de una secuencia w de vectores E/S, se obtiene una secuencia reducida w' de vectores de E/S. A cada vector de E/S que pertenece a la secuencia w', se le asigna una

función correspondiente, después se analiza los eventos de salida y se realiza una clasificación de acuerdo a dichos eventos, para poder decidir que funciones pertenecen a determinada transición.

Una vez que se realiza esta clasificación, se obtienen las transiciones correspondientes. Al finalizar el método de identificación se obtiene la matriz de incidencia observable φC y la secuencia de transiciones S. En el siguiente capítulo se explica a detalle cada una de las partes de este método.

Capítulo 3

Método de Identificación de procesos de eventos discretos

Resumen. En este capítulo se presenta el método de construcción del modelo observable como una red de Petri interpretada (RPI). Se introducen los conceptos básicos y se describen los algoritmos que constituyen cada una de las etapas del método. A partir de la secuencia de entrada/salida w, se obtienen los eventos de entrada compuestos que son asociados a las transiciones del modelo y se crea la RPI observable; posteriormente se determina la secuencia de transiciones correspondiente a w.

3.1. Planteamiento del problema

El método de identificación aquí propuesto se sitúa en el mismo contexto que el desarrollado en [Estrada, 2013]; también el enfoque general el cual consta de dos fases que tratan con la obtención de la partes observable y no observable de un modelo en Redes de Petri Interpretadas (RPI). El trabajo aquí presentado trata con la primera fase, la cual obtiene la parte observable de la RPI mediante una técnica nueva técnica igualmente precisa pero más simple.

Se consideran Sistemas de Eventos Discretos que consisten en una planta y su controlador industrial, tal que el comportamiento de dicho sistema se puede observar mediante la recopilación de las señales intercambiadas entre el controlador y la planta. Representada por una secuencia de vectores de entrada y salida (E/S) *w*.

El objetivo es descubrir, a partir de observaciones del comportamiento del sistema, como los componentes del sistema están relacionados entre sí y construir un modelo compacto que puede mostrar explícitamente el comportamiento descubierto. La identificación de los sistemas ya existentes implica dos aspectos importantes a considerar: la operación del sistema y el proceso de observación.

Se presenta un método de identificación basado en el análisis de las relaciones entre las entradas y salidas a lo largo de un comportamiento observado presentado por la secuencia *w*. Estas relaciones se descubren mediante el análisis de funciones generadas para determinar las transiciones que modelan cambios en las salidas del sistema, obteniendo una secuencia S de transiciones, la cual nos servirá de apoyo para la creación del modelo no observable en futuras referencias. Esto permite construir el modelo observable como una Red de Petri Interpretada.

El método de identificación utiliza la secuencia E/S *w* para obtener un modelo parcial de la RPI que contiene los lugares observables del sistema.

Figura 3.1 Planteamiento del problema

3.2. Estrategia general

Para lograr el objetivo anterior se ha diseñado una estrategia que soluciona el problema fácilmente y que realiza la minería de procesos de una manera eficiente. Se trabaja en dos partes; la primera se encarga de descubrir el comportamiento reactivo del sistema y la segunda parte genera un modelo de RPI que contiene los lugares no observables del sistema, lo que permite generar el modelo completo que representa el comportamiento observado en el sistema.

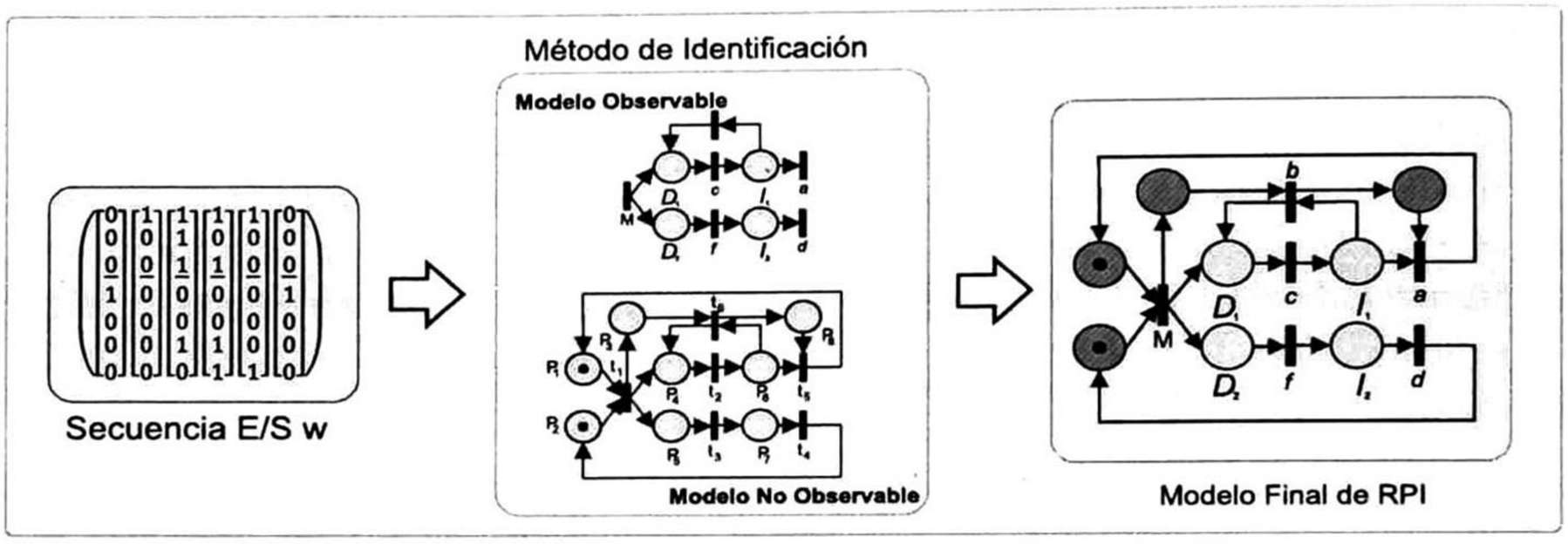


Figura 3.2 Enfoque general del método

En particular, este trabajo se enfoca en la primera parte de la estrategia general. Consiste en procesar una secuencia E/S *w* (figura 3.3 (a)), de la cual se obtienen eventos característicos y una secuencia reducida *w'*. A cada evento característico se le asocia una función; después se analizan los eventos característicos de salida observados para determinar las transiciones finales que están relacionadas a cada una de las funciones. Posteriormente se obtiene una matriz de incidencia y una secuencia de transiciones S correspondiente a *w*. Los pasos del método se describen en el algoritmo 3.1.

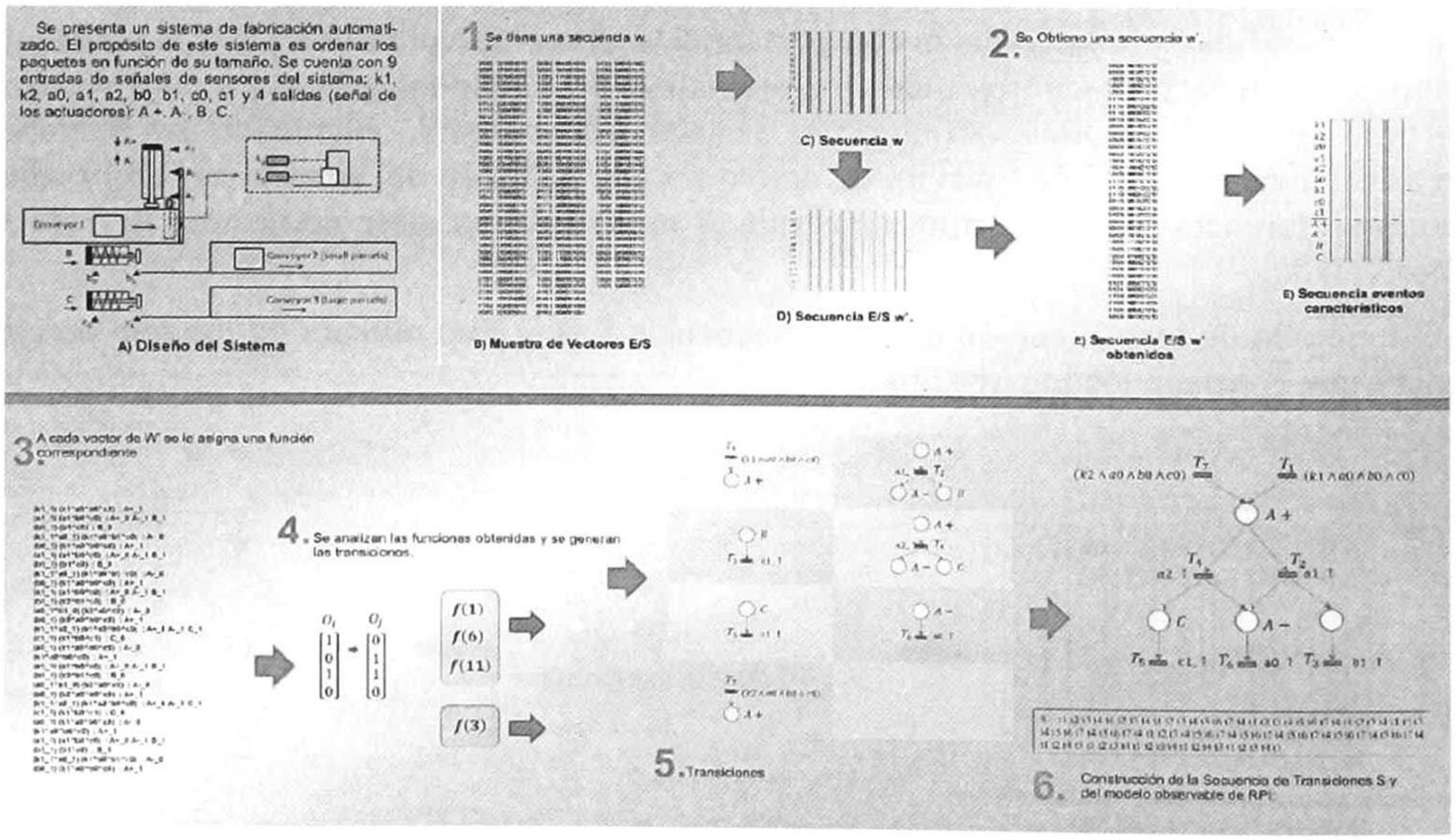


Figura 3.3 Descripción del método de identificación del comportamiento observable del Sistema.

Algoritmo 3.1 Descripción general del descubrimiento del comportamiento reactivo

Entrada: Secuencia E/S *w*.

Salida: Matriz de incidencia observable φC, una función de etiquetado λ, y una secuencia de transiciones *S*.

1. Analizar la secuencia w con la finalidad de que:
   Obtener la secuencia de E/S *w'*.
   Procesar los eventos elementales y los eventos característicos.
   Construir las funciones características.
   Obtener las transiciones relacionadas a un conjunto de funciones características.
2. Usar *w'*, las funciones características y los eventos característicos con la finalidad de que:
   Obtener la función de etiquetado λ.
   Obtener la matriz observable de incidencia φC.
   Obtener la secuencia de transiciones S.

3.3. Conceptos básicos

En esta sección se retoman algunos conceptos básicos previamente definidos en [Estrada, 2013]. A lo largo del desarrollo del método estos conceptos nos serán útiles para poder definir fácilmente los nuevos conceptos que se presentan en este capítulo.

Definición 3.1 La secuencia E/S *w* de un SED con *m* entradas y *n* salidas es una secuencia de vectores de entrada y salida, tal que, $\boldsymbol{w} = \left(\left[\frac{\boldsymbol{I(1)}}{\boldsymbol{O(1)}}\right], \left[\frac{\boldsymbol{I(2)}}{\boldsymbol{O(2)}}\right], \ldots, \left[\frac{\boldsymbol{I(|w|)}}{\boldsymbol{O(|w|)}}\right]\right)$, donde $\left[\frac{\boldsymbol{I(j)}}{\boldsymbol{O(j)}}\right]$ es el j-esímo vector observado E/S de tamaño $m + n$ en la secuencia *w* y $|\boldsymbol{w}|$ es el tamaño de la palabra de E/S *w*.

Con el fin de descubrir la relación entre las entradas y salidas del sistema, se puede calcular en cada vector de evento los cambios específicos que ocurren, es decir las señales de entrada y salida que han cambiado su valor.

Definición 3.2 Un *evento elemental* es la diferencia entre dos vectores consecutivos de E/S $\boldsymbol{E_k = w_{k+1} - w_k}$. Los valores del vector diferentes de cero representan cambios en las variables de entrada o salida. En cada evento se puede distinguir los eventos de entrada $\boldsymbol{IE_k}$ y los eventos de salida $\boldsymbol{OE_k}$, es decir, $\boldsymbol{E_k} = \left[\frac{\boldsymbol{IE_k}}{\boldsymbol{OE_k}}\right]$.

A partir de una secuencia *w* se puede obtener una secuencia de eventos $E = E_1E_2E_3 \ldots E_{|w|-1}$.

También se pueden representar los eventos en una forma algebraica donde se especifican las variables de entrada o salida que cambian en el evento. Cuando para la entrada I_i (salida O_i), ocurre $IE_{ij} = 1$ ($OE_{ij} = 1$) puede representarse por I_i_1 (O_i_1). Para la entrada I_i (salida O_i), ocurre cuando $IE_{ij} = -1$ ($OE_{ij} = -1$) y puede representarse por I_i_0 (O_i_0).

Se puede descomponer cada evento de entrada (salida) como una conjunción de eventos de entrada (salida):

$$ie_j = IE_{j1} \bullet IE_{j2} \bullet \ldots = \prod IE_{ji} \text{ tal que } I_{ji}(j+1) - I_{ji}(j) \neq 0$$

$$oe_j = OE_{j1} \bullet OE_{j2} \bullet \ldots = \prod OE_{ji} \text{ tal que } O_{ji}(j+1) - O_{ji}(j) \neq 0$$

Si no ocurre ningún evento de entrada (salida), se puede representar como $IE_j = \varepsilon$ ($OE_j = \varepsilon$).

Ejemplo 3.1. Considere dos vectores E/S consecutivos pertenecientes a la secuencia w. Se muestran a continuación el evento elemental correspondiente E_1 y los eventos elementales de entrada y salida en su forma vectorial (IE_1, OE_1)y simbólica (ie_1, oe_1).

$$w = \begin{matrix} m \\ a \\ b \\ c \\ d \\ e \\ f \\ D1 \\ I1 \\ D2 \\ I2 \end{matrix} \begin{matrix} \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \\ \end{matrix} \begin{matrix} E_{(1)} \\ \begin{bmatrix}1\\0\\0\\0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} \end{matrix} \begin{matrix} \\ \begin{bmatrix}-1\\0\\1\\0\\0\\1\\0\\1\\0\\1\\0\end{bmatrix} \end{matrix} \begin{matrix} \\ \begin{bmatrix}0\\0\\1\\0\\0\\1\\0\\1\\0\\1\\0\end{bmatrix} \end{matrix}$$

$$IE_1 = [-1,0,1,0,0,1,0]^T$$

$$ie_1 = m\_0 \bullet b\_1 \bullet e\_1$$

$$OE_1 = [1,0,1,0]^T$$

$$oe_1 = D1\_1 \bullet D2\_1$$

Utilizaremos como caso de estudio principal el ejemplo 2.8 del capítulo 2 para la descripción de método diseñado. Se considera una secuencia de 100 vectores de E/S que representan el comportamiento del sistema los cuales se muestran en la tebla 3.1. Se tienen 7 señales de entradas ($\Sigma = \{M, a, b, c, d, e, f\}$) y 4 señales de salida ($\Phi = \{D_1, D_2, I_1, I_2\}$). La disposición de las variables en los vectores E/S es la siguiente: [M a b c d e f D_1 D_2 I_1 I_2]^T.

No. de Vector	O_i	I_j	No. de Vector	O_i	I_j
1	0000	1000000	**51**	0000	1000000
2	1010	0010010	**52**	1010	0010010
3	1010	0001010	**53**	1010	0010001
4	0110	0010010	**54**	1010	0001001
5	1010	0001010	**55**	1001	0001100
6	0110	0010010	**56**	0101	0010100
7	0110	0100010	**57**	1001	0001100
8	0010	0000010	**58**	1000	0001000
9	0010	0000001	**59**	0100	0010000
10	0001	0000100	**60**	0100	0100000
11	0000	1000000	**61**	0000	1000000
12	1010	0010010	**62**	1010	0010010
13	1010	0010001	**63**	1010	0010001
14	1001	0010100	**64**	1010	0001001
15	1001	0001100	**65**	0110	0010001
16	1000	0001000	**66**	1010	0001001
17	0100	0010000	**67**	1001	0001100
18	1000	0001000	**68**	1000	0001000
19	0100	0010000	**69**	0100	0010000
20	0100	0100000	**70**	0100	0100000

21	0000	1000000	**71**	0000	1000000
22	1010	0010010	**72**	1010	0010010
23	1010	0001010	**73**	1010	0001010
24	0110	0010010	**74**	1010	0001001
25	0110	0010001	**75**	1001	0001100
26	0101	0010100	**76**	0101	0010100
27	1001	0001100	**77**	1001	0001100
28	0101	0010100	**78**	0101	0010100
29	0100	0010000	**79**	0100	0010000
30	0100	0100000	**80**	0100	0100000
31	0000	1000000	**81**	0000	1000000
32	1010	0010010	**82**	1010	0010010
33	1010	0001010	**83**	1010	0010001
34	1010	0001001	**84**	1010	0001001
35	0110	0010001	**85**	1001	0001100
36	0101	0010100	**86**	0101	0010100
37	0100	0010000	**87**	1001	0001100
38	1000	0001000	**88**	1000	0001000
39	0100	0010000	**89**	0100	0010000
40	0100	0100000	**90**	0100	0100000
41	0000	1000000	**91**	0000	1000000
42	1010	0010010	**92**	1010	0010010
43	1010	0001010	**93**	1010	0010001
44	0110	0010010	**94**	1001	0010100
45	0110	0010001	**95**	1001	0001100
46	0101	0010100	**96**	1000	0001000
47	1001	0001100	**97**	0100	0010000
48	0101	0010100	**98**	1000	0001000
49	0101	0100100	**99**	0100	0010000
50	0001	0000100	**100**	0100	0100000

Tabla 3.1 Secuencia w de vectores de E/S que representa el comportamiento del sistema.

Ahora veremos un ejemplo de la obtención de eventos elementales de entrada y salida para el ejemplo 2.8.

Ejemplo 3.2. Se considera la siguiente secuencia E/S w:

$$w = \begin{matrix} m \\ a \\ b \\ c \\ d \\ e \\ \underline{f} \\ D1 \\ I1 \\ D2 \\ I2 \end{matrix} \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \underline{0} & \underline{0} & \underline{1} & \underline{1} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{0} & \underline{1} & \underline{0} & \underline{1} & \underline{0} \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}$$

De la secuencia de E/S w se obtiene la secuencia E de eventos observados:

$$E = \begin{matrix} m \\ a \\ b \\ c \\ d \\ e \\ \underline{f} \\ D1 \\ I1 \\ D2 \\ I2 \end{matrix} \begin{bmatrix} -1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ \underline{0} \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ \underline{1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ \underline{-1} \\ 0 \\ 0 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ \underline{0} \\ -1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 1 \\ -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} -1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ \underline{0} \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ \underline{1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ \underline{-1} \\ 0 \\ 0 \\ -1 \\ 1 \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix}$$

Se puede escribir w con los eventos intercalados que se obtienen, para una explicación más clara:

$$w = \begin{matrix} m \\ a \\ b \\ c \\ d \\ e \\ \underline{f} \\ D1 \\ I1 \\ D2 \\ I2 \end{matrix} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_1 \\ \begin{bmatrix} -1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ \underline{1} \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_2 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ \underline{1} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_3 \\ \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_4 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \\ \underline{0} \\ 0 \\ 0 \\ -1 \\ 1 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_5 \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ -1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_6 \\ \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ \underline{0} \\ -1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_7 \\ \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 1 \\ -1 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_8 \\ \begin{bmatrix} 0 \\ 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ \underline{0} \\ -1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_9 \\ \begin{bmatrix} 0 \\ 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_{10} \\ \begin{bmatrix} 1 \\ -1 \\ 0 \\ 0 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ -1 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_{11} \\ \begin{bmatrix} -1 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ \underline{0} \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_{12} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ -1 \\ 1 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \underline{\mathbf{1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \begin{matrix} E_{13} \\ \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ -1 \\ \underline{0} \\ 0 \\ -1 \\ 1 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{0} \\ \underline{\mathbf{0}} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \end{bmatrix} \begin{matrix} E_{14} \\ \begin{bmatrix} 0 \\ 0 \\ -1 \\ 1 \\ 0 \\ 0 \\ \underline{0} \\ 0 \\ 0 \\ 0 \\ -1 \end{bmatrix} \end{matrix} \begin{bmatrix} \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \\ \mathbf{0} \\ \mathbf{0} \\ \underline{\mathbf{1}} \\ \mathbf{0} \\ \mathbf{0} \\ \mathbf{1} \\ \mathbf{1} \end{bmatrix}$$

Los eventos elementales correspondientes a los primeros 14 eventos de la secuencia w son listados en la tabla 3.2 donde se ha utilizado la notación simbólica para eventos de entrada como de salida.

Vector Evento	Eventos Elementales de Entrada	Eventos Elementales de Salida
$\boldsymbol{E(1)}$	$ie_1 = (M\_0 \bullet b\_1 \bullet e\_1)$	$oe_1 = (D_1\_1 \bullet D_2\_1)$
$\boldsymbol{E(2)}$	$ie_2 = (e\_0 \bullet f\_1)$	$oe_2 = \varepsilon$
$\boldsymbol{E(3)}$	$ie_3 = (b\_0 \bullet c\_1)$	$oe_3 = \varepsilon$
$\boldsymbol{E(4)}$	$ie_4 = (d\_1 \bullet f\_0)$	$oe_4 = (D_2\_0 \bullet I_2\_1)$
$\boldsymbol{E(5)}$	$ie_5 = (d\_0)$	$oe_5 = (I_2\_0)$
$\boldsymbol{E(6)}$	$ie_6 = (b\_1 \bullet c\_0)$	$oe_6 = (D_1\_0 \bullet I_1\_1)$
$\boldsymbol{E(7)}$	$ie_7 = (b\_0 \bullet c\_1)$	$oe_7 = (D_1\_1 \bullet I_1\_0)$
$\boldsymbol{E(8)}$	$ie_8 = (b\_1 \bullet c\_0)$	$oe_8 = (D_1\_0 \bullet I_1\_1)$
$\boldsymbol{E(9)}$	$ie_9 = (a\_1 \bullet b\_0)$	$oe_9 = \varepsilon$
$\boldsymbol{E(10)}$	$ie_{10} = (M\_1 \bullet a\_0)$	$oe_{10} = (I_1\_0)$
$\boldsymbol{E(11)}$	$ie_{11} = (M\_0 \bullet b\_1 \bullet e\_1)$	$oe_{11} = (D_1\_1 \bullet D_2\_1)$
$\boldsymbol{E(12)}$	$ie_{12} = (e\_0 \bullet f\_1)$	$oe_{12} = \varepsilon$
$\boldsymbol{E(13)}$	$ie_{13} = (d\_1 \bullet f\_0)$	$oe_{13} = (D_2\_0 \bullet I_2\_1)$
$\boldsymbol{E(14)}$	$ie_{14} = (b\_0 \bullet c\_1)$	$oe_{14} = (D_1\_0 \bullet I_1\_1)$

Tabla 3.2 Lista de eventos elementales para el ¡Error! No se encuentra el origen de la referencia.

3.4. Obtención de eventos compuestos

La cantidad de eventos elementales puede ser grande. EN el peor de los casos el número de eventos distintos es 3^{r+q}, donde r es la cantidad de entradas y q es la cntidad de salidas. Se tratará de reducir la representación de los eventos, por un lado enfocándonos en aquellos que provocan cambios en las salidas y por otro lado, en compactar los eventos de entrada en eventos compuestos. Para ello se definirán nuevos conceptos.

3.4.1. Eventos característicos

Definición 3.3 Un *evento característico* de E/S es el vector donde el evento elemental de salida es no nulo($\boldsymbol{OE_k}' \neq \mathbf{0}$). Denotamos como $\boldsymbol{e'}$ a la secuencia de eventos característicos.

$$\boldsymbol{e'} = \left(\left[\frac{\boldsymbol{IE'_k(1)}}{\boldsymbol{OE'_k(1)}}\right], \left[\frac{\boldsymbol{IE'_k(2)}}{\boldsymbol{OE'_k(2)}}\right], \ldots, \left[\frac{\boldsymbol{IE'_k(|e'|)}}{\boldsymbol{OE'_k(|e'|)}}\right]\right).$$

Se denota como *w'* a la secuencia de vectores de E/S de la secuencia *w* donde $(\boldsymbol{O(i)} \neq \mathbf{0})$. También denotaremos como *OE'* al conjunto de eventos característicos de salida $\boldsymbol{OE'_k}$ encontrados.

Continuando con el Ejemplo 3.1, se calculan los eventos característicos para los primeros 15 vectores de la secuencia w; esto se muestra en la Tabla 3.3 donde se observa que existen eventos de salida que son idénticos para diferentes eventos de entrada. Esto nos ayuda a agrupar eventos de entrada con el fin de encontrar eventos característicos compuestos. En la tabla 3.4 se ha resumido los eventos de entrada que están asociados a algún evento de salida.

Vector Evento	Eventos Característicos de Entrada	Eventos Característicos de Salida
$\boldsymbol{E(1)}$	$ie'_1 = (M\_0 \bullet b\_1 \bullet e\_1)$	$oe'_1 = (D_1\_1 \bullet D_2\_1)$
$\boldsymbol{E(2)}$	$ie'_2 = (d\_1 \bullet f\_0)$	$oe'_2 = (D_2\_0 \bullet I_2\_1)$
$\boldsymbol{E(3)}$	$ie'_3 = (d\_0)$	$oe'_3 = (I_2\_0)$
$\boldsymbol{E(4)}$	$ie'_4 = (b\_1 \bullet c\_0)$	$oe'_4 = (D_1\_0 \bullet I_1\_1)$
$\boldsymbol{E(5)}$	$ie'_5 = (b\_0 \bullet c\_1)$	$oe'_5 = (D_1\_1 \bullet I_1\_0)$
$\boldsymbol{E(6)}$	$ie'_4 = (b\_1 \bullet c\_0)$	$oe'_4 = (D_1\_0 \bullet I_1\_1)$
$\boldsymbol{E(7)}$	$ie'_7 = (M\_1 \bullet a\_0)$	$oe'_6 = (I_1\_0)$
$\boldsymbol{E(8)}$	$ie'_1 = (M\_0 \bullet b\_1 \bullet e\_1)$	$oe'_5 = (D_1\_1 \bullet D_2\_1)$
$\boldsymbol{E(9)}$	$ie'_2 = (d\_1 \bullet f\_0)$	$oe'_2 = (D_2\_0 \bullet I_2\_1)$
$\boldsymbol{E(10)}$	$ie'_5 = (b\_0 \bullet c\_1)$	$oe'_2 = (D_1\_0 \bullet I_1\_1)$

Tabla 3.3 Lista de eventos característicos para el ¡Error! No se encuentra el origen de la referencia.**de los primeros 15 vectores de E/S de la secuencia w'**。

El conjunto *OE'* que se obtiene de la tabla 3.3 es:

$$OE' = \left\{\begin{bmatrix}1\\0\\1\\0\end{bmatrix}, \begin{bmatrix}0\\0\\-1\\1\end{bmatrix}, \begin{bmatrix}0\\0\\0\\-1\end{bmatrix}, \begin{bmatrix}-1\\1\\0\\0\end{bmatrix}, \begin{bmatrix}1\\-1\\0\\0\end{bmatrix}, \begin{bmatrix}0\\-1\\0\\0\end{bmatrix}\right\}$$

3.4.2. Función característica

Una función característica describe las condiciones bajo las cuales se produce un cambio en las salidas, esto es, representa el comportamiento reactivo E/S. Las funciones características tienen dos componentes: la primera es el evento característico de entrada correspondiente, y la segunda es una expresión booleana de variables de entrada que describe el contexto en el que ocurrió el evento característico.

El contexto es una combinación de valores de variables de entrada que no cambian mientras que no ocurra un cambio en las salidas. El formato de estas funciones es el mismo que el de las funciones elementales.

El algoritmo que nos permite construir el contexto es el que se presenta a continuación:

Algoritmo 3.2 Contexto

Entrada: $w\_entradas$, Σ;
Salida: *contexto_simbolico*;

$no\_cambian \leftarrow w\_entradas_0$; $contexto\_simbolico \leftarrow \varepsilon$
Para i=1 hasta $|w\_entradas| - 1$:
$no\_cambian \leftarrow no\_cambian \oplus w\_entradas_i$
Para cada i=0 hasta $|\Sigma| - 1$:
Si $no\_cambian_i = 0$ entonces
Si $w\_entradas_{1i} = 0$ entonces
$contexto\_simbolico \leftarrow contexto\_simbolico \,\&/\Sigma_i$
Si $w\_entradas_{1i} = 1$ entonces
$contexto\_simbolico \leftarrow contexto\_simbolico \,\&\, \Sigma_i$
Regresa $contexto\_simbolico$

A continuación se presenta un procedimiento para obtener las funciones características.

Algoritmo 3.3 Determinación de las funciones características

Entrada: Secuencia w, Σ;
Salida: Secuencia f de funciones características;

---

1) $E \leftarrow \varepsilon$; $e' \leftarrow \varepsilon$; $f \leftarrow \varepsilon$; $funciones\_finales \leftarrow \varepsilon$; $funcion_j(oe_k) \leftarrow null$; $j \leftarrow 0$;
$w\_entradas \leftarrow \varepsilon$
2) Para k=1 hasta $|w|$:
$e_k = w_{k+1} - w_k$
$w\_entradas \leftarrow w\_entradas \cdot I_k$
$E \leftarrow E \cdot e_k$
Si $oe_k \neq \varepsilon$ entonces
Si $ie_k = \varepsilon$ entonces
$indice\_aux = k$
mientras $indice\_aux! = 0$
Si $ie_k = \vec{0}$:
$funcion_l(oe_k) \leftarrow (ie_{indice\_aux}) \bullet Contexto(w\_entradas, \Sigma)$
$indice\_aux \leftarrow indice\_aux - 1$
$e\_aux \leftarrow ie_{indice\_aux} \cdot oe_k$

$e' \leftarrow e' \cdot e\_aux$

Si no:

$funcion_j(oe_k) \leftarrow (ie_k) \bullet Contexto(w\_entradas, \Sigma)$

$e' \leftarrow e' \cdot e_k$

Si $función_j(oe_k) \notin funciones\_finales$ entonces

$funciones\_finales \leftarrow funciones\_finales \cup funcion_j(oe_k)$

$f \leftarrow f \circ "f_j"$

$j \leftarrow j + 1$

Si no:

$p \leftarrow Obtener(funcion_j(oe_k), funciones\_finales)$

$f \leftarrow f \cdot "f_p"$

Fin Para

3) Regresa f

También se presenta a continuación un procedimiento para obtener el índice de la función que ya ha sido calculada; esto complementa el algoritmo 3.2 para su mejor comprensión.

Algoritmo 3.4 Obtener ($funcion \in funciones\_finales$)

Entrada: *función, funciones_finales*;

Salida: p:índice de la función encontrada;

Para i=0 hasta $|funciones\_finales| - 1$:

Si $funciones\_finales_p = funcion$:

Regresa p

Después de aplicar el algoritmo 3.2 en los datos contenidos en la tabla 3.1 se obtienen los resultados registrados en el ejemplo 3.3 los cuales muestran las funciones características obtenidas de cada uno de los eventos en la secuencia e'

Ejemplo 3.3 Para el ejemplo 3.1 se obtienen las funciones características de sus eventos correspondientes mostradas en la table 3.4

	IE'_i	OE'_i	Secuencia de funciones características f
1	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
2	0 0 1 -1 0 0 0	-1 1 0 0	$f_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet /d \bullet e \bullet /f)$
3	0 0 -1 1 0 0 0	1 -1 0 0	$f_3(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /f)$
4	0 0 1 -1 0 0 0	-1 1 0 0	$f_4(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /f)$
5	0 -1 0 0 0 0 0	0 -1 0 0	$f_5(I1\_0) = (a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet e \bullet /f)$
6	0 0 0 0 1 0 -1	0 0 -1 1	$f_6(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /b \bullet /c \bullet /d \bullet /e \bullet f)$
7	1 0 0 0 -1 0 0	0 0 0 -1	$f_7(I2\_0) = (M\_1 \bullet d\_0) \bullet (/M \bullet /a \bullet /b \bullet /c \bullet /e \bullet /f)$
8	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
9	0 0 0 0 1 0 -1	0 0 -1 1	$f_8(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /d \bullet /e \bullet f)$
10	0 0 0 0 -1 0 0	0 0 0 -1	$f_9(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet d \bullet /e \bullet /f)$
11	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$

12	0 0 -1 1 0 0 0	1 -1 0 0	$f_{11}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /e \bullet /f)$
13	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
14	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
15	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
16	0 0 1 -1 0 0 0	-1 1 0 0	$f_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet /d \bullet e \bullet /f)$
17	0 0 0 0 1 0 -1	0 0 -1 1	$f_8(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /d \bullet /e \bullet f)$
18	0 0 -1 1 0 0 0	1 -1 0 0	$f_{13}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
19	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
20	0 0 0 0 -1 0 0	0 0 0 -1	$f_{15}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
21	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
22	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
23	0 0 1 -1 0 0 0	-1 1 0 0	$f_{16}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet f)$
24	0 0 0 0 1 0 -1	0 0 -1 1	$f_{17}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /e)$
25	0 0 0 0 -1 0 0	0 0 0 -1	$f_{15}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
26	0 0 -1 1 0 0 0	1 -1 0 0	$f_{11}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /e \bullet /f)$
27	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
28	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
29	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
30	0 0 1 -1 0 0 0	-1 1 0 0	$f_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet /d \bullet e \bullet /f)$
31	0 0 0 0 1 0 -1	0 0 -1 1	$f_8(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /d \bullet /e \bullet f)$
32	0 0 -1 1 0 0 0	1 -1 0 0	$f_{13}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
33	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
34	0 -1 0 0 0 0 0	0 -1 0 0	$f_{18}(I1\_0) = (a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet d \bullet /e \bullet /f)$
35	1 0 0 0 -1 0 0	0 0 0 -1	$f_7(I2\_0) = (M\_1 \bullet d\_0) \bullet (/M \bullet /a \bullet /b \bullet /c \bullet /e \bullet /f)$
36	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
37	0 0 0 0 1 0 -1	0 0 -1 1	$f_{19}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet c \bullet /d \bullet /e)$
38	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
39	0 0 -1 1 0 0 0	1 -1 0 0	$f_{13}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
40	0 0 0 0 -1 0 0	0 0 0 -1	$f_{20}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
41	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
42	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
43	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
44	0 0 1 -1 0 0 0	-1 1 0 0	$f_{21}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet c \bullet /d \bullet /e)$
45	0 0 -1 1 0 0 0	1 -1 0 0	$f_{22}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /e)$
46	0 0 0 0 1 0 -1	0 0 -1 1	$f_{23}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e)$
47	0 0 0 0 -1 0 0	0 0 0 -1	$f_{20}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
48	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
49	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
50	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
51	0 0 0 0 1 0 -1	0 0 -1 1	$f_{24}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet f)$
52	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$

53	0 0 -1 1 0 0 0	1 -1 0 0	$f_{13}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
54	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
55	0 0 0 0 -1 0 0	0 0 0 -1	$f_{15}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
56	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
57	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
58	0 0 0 0 1 0 -1	0 0 -1 1	$f_{19}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet c \bullet /d \bullet /e)$
59	0 0 1 -1 0 0 0	-1 1 0 0	$f_{14}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
60	0 0 -1 1 0 0 0	1 -1 0 0	$f_{13}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /e \bullet /f)$
61	0 0 0 0 -1 0 0	0 0 0 -1	$f_{20}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet /f)$
62	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
63	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
64	-1 0 1 0 0 1 0	1 0 1 0	$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
65	0 0 0 0 1 0 -1	0 0 -1 1	$f_8(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /d \bullet /e \bullet f)$
66	0 0 0 0 -1 0 0	0 0 0 -1	$f_9(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet d \bullet /e \bullet /f)$
67	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
68	0 0 -1 1 0 0 0	1 -1 0 0	$f_{11}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /c \bullet /d \bullet /e \bullet /f)$
69	0 0 1 -1 0 0 0	-1 1 0 0	$f_{10}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /b \bullet /d \bullet /e \bullet /f)$
70	1 -1 0 0 0 0 0	0 -1 0 0	$f_{12}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/M \bullet a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$

Tabla 3.4 Funciones características $f_j(oe_k)$ asociadas a cada vector de E/S de la secuencia e'.

De este ejemplo se obtienen 21 funciones características para 6 eventos de salida, lo cual es resumido en la siguiente matriz observable:

$$\varphi C = \begin{array}{c} \begin{matrix} f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 & f_8 & f_9 & f_{10} & f_{11} & f_{12} & f_{13} & f_{14} & f_{15} & f_{16} & f_{17} & f_{18} & f_{19} & f_{20} & f_{21} \end{matrix} \\ \begin{bmatrix} 1 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 & 1 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 & 0 & 0 & 1 & -1 & -1 & -1 & 1 & 0 & 1 & 0 & -1 & 0 & 0 & -1 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 0 & 0 & -1 \\ 0 & 0 & 0 & 0 & 1 & -1 & 1 & -1 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 1 & 0 & 1 & -1 & 0 & 1 \end{bmatrix} \end{array}$$

Como se puede observar para un mismo evento de salida, (columnas de φC) existen diferentes funciones características que lo provocan; por lo anterior se analizará la similitud de las funciones características asociadas a un mismo evento de salida, con la finalidad de obtener una función característica más compleja que sintetice las funciones. Llamaremos a esta función *evento compuesto*.

3.4.3. Determinación de eventos compuestos

Para determinar el evento compuesto, se analiza el conjunto de funciones que tienen el mismo evento característico de salida oe'_k en OE'. La estrategia general para agrupar funciones características consiste en considerar, para cada oe'_k, los ie'_k que tienen eventos simbólicos comunes y realizar una reducción boolena de las expresiones en $contexto_k$.

Por ejemplo, considerando las siguientes funciones características para $oe'_k = D1\_0 \bullet I1\_1$

$$f_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d \bullet e \bullet /f)$$

$$f_9(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d \bullet /e \bullet /f)$$

$$f_{13}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet d \bullet /e \bullet /f)$$

$$f_{15}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d)$$

Podemos observar que tienen en común $ie_k = (b\_1 \bullet c\_0)$; entonces se hace la simplificación de las expresiones del $contexto_k$ y se obtiene $(/M \bullet /a)$. El evento compuesto resultante es: $\xi_1 = (b\_1 \bullet c\_0) \bullet (/M \bullet /a)$. A continuación se presenta el algoritmo que obtiene los eventos compuestos.

Primeramente se obtendrán los conjuntos V_k de funciones características que provocan el mismo cambio en la salida.

Algoritmo 3.5 Obtención de V_k

Entrada: *OE', f_final*
Salida: Conjunto $V = \{V_k\}$

$V_k \leftarrow \varnothing$, $V \leftarrow \varnothing$
Para k=1 hasta $|OE'|$:
  Para j=1 hasta n_f:
   Si oek = $\varphi C(j, \bullet)$ entonces:
     $V_k \leftarrow V_k \cup f_j$
  $V \leftarrow V \cup V_k$
Regresa **V**

Del ejemplo 2.8 que se ha desarrollado a lo largo de este trabajo tenemos el siguiente conjunto $V=\{ V_1 \ V_2 , V_3 , V_4 , V_5 , V_6 \}$, donde las funciones agrupadas están detalladas en la tabla 3.7.

$V_1 = \{f_1(D1\_1 \bullet D2\_1)\}$	$V_2 = \{f_5(D2\_0 \bullet I2\_1), f_7(D2\_0 \bullet I2\_1), f_{16}(D2\_0 \bullet I2\_1), f_{18}(D2\_0 \bullet I2\_1), f_{21}(D2\_0 \bullet I2\_1)\}$
$f_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) \bullet (/a \bullet /c \bullet /d \bullet /f)$	$f_5(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /b \bullet /c)$ $f_7(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c)$ $f_{16}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /e)$ $f_{18}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a)$ $f_{21}(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) \bullet (/M \bullet /a \bullet /b \bullet /e \bullet c)$
$V_3 = \{f_6(I2\_0), f_8(I2\_0), f_{14}(I2\_0), f_{19}(I2\_0)\}$	$V_4 = \{f_2(D1\_0 \bullet I1\_1), f_9(D1\_0 \bullet I1\_1), f_{13}(D1\_0 \bullet I1\_1), f_{15}(D1\_0 \bullet I1\_1)\}$
$f_6(I2\_0) = (M\_1 \bullet d\_0) \bullet (/a \bullet /b \bullet /c \bullet /e \bullet /f)$ $f_8(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /e \bullet /f)$ $f_{14}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet b \bullet /c \bullet /e \bullet /f)$ $f_{19}(I2\_0) = (d\_0) \bullet (/M \bullet /a \bullet /b \bullet c \bullet /e \bullet /f)$	$f_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d \bullet e \bullet /f)$ $f_9(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d \bullet /e \bullet /f)$ $f_{13}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet d \bullet /e \bullet /f)$ $f_{15}(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) \bullet (/M \bullet /a \bullet /d)$
$V_5 = \{f_3(D1\_1 \bullet I1\_0), f_{10}(D1\_1 \bullet I1\_0), f_{12}(D1\_1 \bullet I1\_0), f_{20}(D1\_1 \bullet I1\_0)\}$	$V_6 = \{f_4(I1\_0), f_{11}(I1\_0), f_{17}(I1\_0)\}$
$f_3(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /d \bullet e \bullet /f)$ $f_{10}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /d \bullet /e \bullet /f)$ $f_{12}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet d \bullet /e \bullet /f)$ $f_{20}(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) \bullet (/M \bullet /a \bullet /d \bullet /e \bullet f)$	$f_4(I1\_0) = (a\_0) \bullet (/M \bullet /c \bullet /d \bullet e \bullet /f)$ $f_{11}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/c \bullet /d \bullet /e \bullet /f)$ $f_{17}(I1\_0) = (a\_0) \bullet (/M \bullet /c \bullet d \bullet /e \bullet /f)$

Tabla 3.5 Conjunto de funciones características v_t para el ejemplo 3.3

El algoritmo 3.6 procesa cada conjunto V_k para obtener los subconjuntos que tengan un evento característico de entrada en común, con la finalidad de determinar si son candidatos a formar un evento compuesto.

Algoritmo 3.6 Obtención de Eventos Compuestos

Entrada: V
Salida: ξ

$\xi \leftarrow \emptyset$
Para $V_k \in V$:
  $\xi \leftarrow \xi \cdot Compuesto(V_k, \xi)$
Regresa ξ

Ejemplo: Se tienen las funciones características siguientes pertenecientes al conjunto V_k.

$$f_4(I1\_0) = (a\_0) \bullet (/M \bullet /c \bullet /d \bullet e \bullet /f)$$
$$f_{11}(I1\_0) = (M\_1 \bullet a\_0) \bullet (/c \bullet /d \bullet /e \bullet /f)$$
$$f_{17}(I1\_0) = (c\_0) \bullet (/M \bullet /c \bullet d \bullet /e \bullet /f)$$

Del procesamiento del algoritmo 3.4 tenemos:

$$\xi_1(I1\_0) = (a\_0) \bullet (/c \bullet /d \bullet /e \bullet /f)$$
$$\xi_2(I1\_0) = (c\_0) \bullet (/M \bullet /c \bullet d \bullet /e \bullet /f)$$

El algoritmo 3.7 determina como se obtienen estos eventos compuestos.

Algoritmo 3.7 Compuesto

Entrada: V_k, Σ
Salida: funciones compuestas ξ

$\xi \leftarrow \emptyset, v\_similares \leftarrow \emptyset, v\_diferentes \leftarrow \emptyset$
Si $V_k \neq \emptyset$:
  $ie \leftarrow V_1$
  Para $v \in V_k$:
    Si $ie \wedge v_{ie} \neq \emptyset$ entonces:
      $v\_similares \leftarrow v\_similares \cdot v$
    Si no:
      $v\_diferentes \leftarrow v\_diferentes \cdot v$
  $\xi \leftarrow \xi + eventoCompuesto(v\_similares)$
  $\xi \leftarrow \xi + Compuesto(v\_diferentes, \Sigma)$
Regresa ξ

Para el ejemplo que se ha trabajado a lo largo de esta tesis, tenemos como resultado los eventos compuestos siguientes:

$$\xi_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) * (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$$
$$\xi_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) * (/M \bullet /a)$$
$$\xi_3(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) * (/M \bullet /a \bullet /c)$$
$$\xi_4(I1\_0) = (a\_0) * (/M \bullet a \bullet /b \bullet /c \bullet /f)$$
$$\xi_5(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) * (/M \bullet /a \bullet /d)$$
$$\xi_6(I2\_0) = (M\_1 \bullet d\_0) * (/M \bullet /a \bullet /e \bullet /f)$$

Cada uno de estos eventos compuestos le corresponde un evento de salida OE_k. Así la matriz de incidencia $\varphi C'$ reducida queda como sigue

$$\varphi C' = \begin{array}{c} \\ \\ \\ \\ \end{array} \begin{array}{c} \xi_1(oe_1) \quad \xi_2(oe_2) \quad \xi_3(oe_3) \quad \xi_4(oe_4) \quad \xi_5(oe_5) \quad \xi_6(oe_6) \\ \begin{pmatrix} 1 & -1 & 1 & 0 & 0 & 0 \\ 0 & 1 & -1 & -1 & 0 & 0 \\ 1 & 0 & 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 0 & 1 & -1 \end{pmatrix} \end{array}$$

Es posible que un evento de salida OE_k sea provocado por diferentes eventos compuestos, por lo que la matriz de incidencia puede tener columnas iguales. A cada $\xi_t\,(oe_k)$ le corresponde una transición T_t. Con lo cual se determina el conjunto de transiciones T, donde $|T|=|\,\xi\,|$ y $\forall \xi_t \in V,\ \lambda(T_t) \leftarrow \xi_t\,(oe_k)$

En el ejemplo la asignación queda cọmo sigue:

Transición	$\xi_t\,(oe_k)$
T_1	$\xi_1(D1\_1 \bullet D2\_1) = (M\_0 \bullet b\_1 \bullet e\_1) * (/a \bullet /b \bullet /c \bullet /d \bullet /e \bullet /f)$
T_2	$\xi_2(D1\_0 \bullet I1\_1) = (b\_1 \bullet c\_0) * (/M \bullet /a)$
T_3	$\xi_3(D1\_1 \bullet I1\_0) = (b\_0 \bullet c\_1) * (/M \bullet /a \bullet /c \bullet)$
T_4	$\xi_4(I1\_0) = (a\_0) * (/M \bullet a \bullet /b \bullet /c \bullet /f)$
T_5	$\xi_5(D2\_0 \bullet I2\_1) = (d\_1 \bullet f\_0) * (/M \bullet /a \bullet /d)$
T_6	$\xi_6(I2\_0) = (M\_1 \bullet d\_0) * (/M \bullet /a \bullet /e \bullet /f)$

Entonces la matriz $\varphi C'$ se puede reescribir de una manera más simple:

$$\varphi C' = \begin{array}{c} \\ D_1 \\ I_1 \\ D_2 \\ I_2 \end{array} \begin{array}{c} T_1 \quad T_2 \quad T_3 \quad T_4 \quad T_5 \quad T_6 \\ \begin{pmatrix} 1 & 0 & 0 & -1 & 1 & 0 \\ 0 & 0 & 0 & 1 & -1 & -1 \\ 1 & -1 & 0 & 0 & 0 & 0 \\ 0 & 1 & -1 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Ahora para obtener el modelo observable representamos cada OE_k mediante los cambios en el marcado en los lugares observables etiquetados con los símbolos de Φ (D1, I1, D2, I2). Para el ejemplo, los fragmentos de RPI que se generan para todas las transiciones de$\varphi C'$ son mostrados en la figura 3.5.

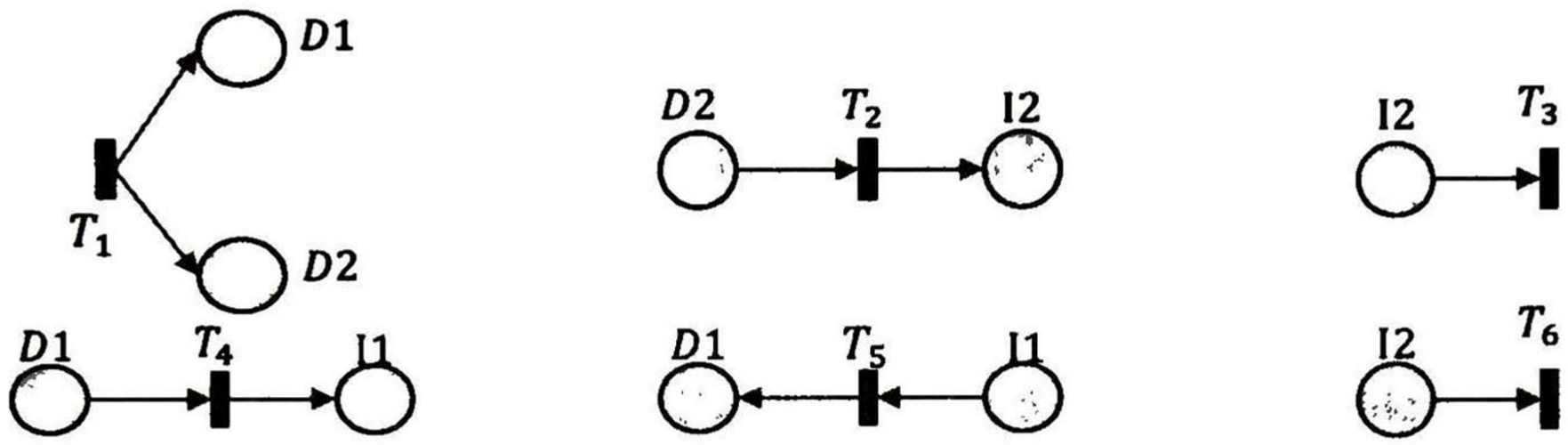


Figura 3.4 Fragmentos de RPI obtenidos de φC'.

Si se fusionan las transiciones y lugares de los fragmentos se obtiene la RPI de la figura 3.6:

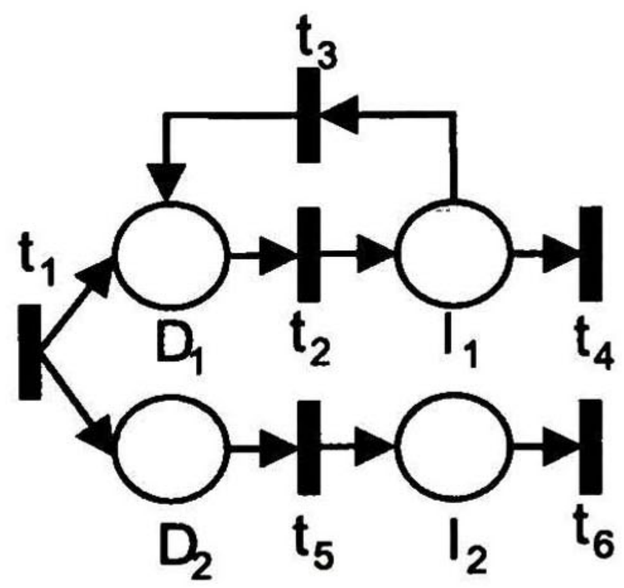


Figura 3.6 Modelo observable del ejemplo 2.8

3.5. Secuencia de transiciones

Además de obtener el modelo observable de la RPI, es necesario determinar la secuencia de transiciones correspondiente a la secuencia de E/S *w*. Esto con el fin de calcular posteriormente el modelo no observable que completa la RPI final.

Para obtener la secuencia de transiciones *S* es suficiente con recorrer el vector de eventos característicos e' y verificar la pertenencia al evento compuesto ξ_t, el cual está asociado a la transición T_t. Este procedimiento se describe a continuación:

Algoritmo 3.8 Obtención de la Secuencia de transiciones

Entrada: e'
Salida: $S \in T\bullet$

$S \leftarrow \emptyset$
Para $\forall f_j \in e'$:
    Siendo ξ_t el evento compuesto que incluye f_j
    $S \leftarrow S \cdot T_t$

Regresa *S*

Así para el ejemplo que se está desarrollando tenemos que:

e'=	f_1	f_2	f_3	f_4	f_5	f_6	f_7	f_1	f_8	f_9	f_{10}	f_{11}	f_{10}	f_{12}	f_1
S=	T1	T2	T3	T2	T4	T5	T6	T1	T5	T6	T2	T3	T2	T4	T1

Donde se observa que T2 tiene asociada varias funciones f_2, f_4, f_{10} en esta subsecuencia, dado que las ξ_2 incluye.

Para el ejemplo completo la secuencia equivalente a e' y en consecuencia a E es:

S = T1 T2 T3 T2 T4 T5 T6 T1 T5 T6 T2 T3 T2 T4 T1 T2 T5 T3 T2 T6 T4 T1 T2 T5 T6 T3 T2 T4 T1 T2 T5 T3 T2 T4 T6 T1 T5 T2 T3 T6 T2 T4 T1 T2 T3 T5 T6 T2 T4 T1 T5 T2 T3 T2 T6 T4 T1 T5 T2 T3 T6 T2 T4 T1 T5 T6 T2 T3 T2 T4

Esta secuencia puede ser procesada por una segunda fase del método que obtiene una RP con lugares no observables la cual va a completar el modelo en RPI.

Capítulo 4
Implementación y Pruebas

Resumen. En este capítulo se presenta la herramienta que implementa el método de identificación propuesto y su aplicación a varios casos de estudio. Primeramente, se describen la arquitectura e interface del software desarrollado. Enseguida se comprueba su funcionamiento procesando diversas secuencias de entrada/salida, algunas obtenidas experimentalmente de procesos de eventos discretos reales.

4.1. Herramienta de software para identificación

Se ha desarrollado una herramienta de software que permite realizar de manera rápida y efectiva el tratamiento de las secuencias de entrada en base al método propuesto en el capítulo 3. A continuación se describe someramente algunos aspectos de la implementación del método. La herramienta procesa para identificación de eventos discretos presentado en el capítulo 3, la entrada del algoritmo es la secuencia de E/S w, la cual proviene de un sistema de eventos discretos cuyo funcionamiento es desconocido. El software desarrollado produce un documento de texto con los resultados del procesamiento: las funciones obtenidas y la matriz de incidencia $\varphi C'$, la cual representa el modelo observable del sistema. Se utiliza interface de Python al lenguaje Dot de Graphviz.

4.1.1. Características

La herramienta de software fue desarrollada en el IDE Eclipse Kepler, en el lenguaje de programación Python. Las principales estructuras de datos son las siguientes:

- Characteristic (Función Característica)
- EventC (Evento Compuesto)

Su método principal recibe como parámetros globales: la secuencia de E/S w, m, n, Σ, y Φ. Se muestra en la figura 4.1 la impresión de pantalla correspondiente al método.

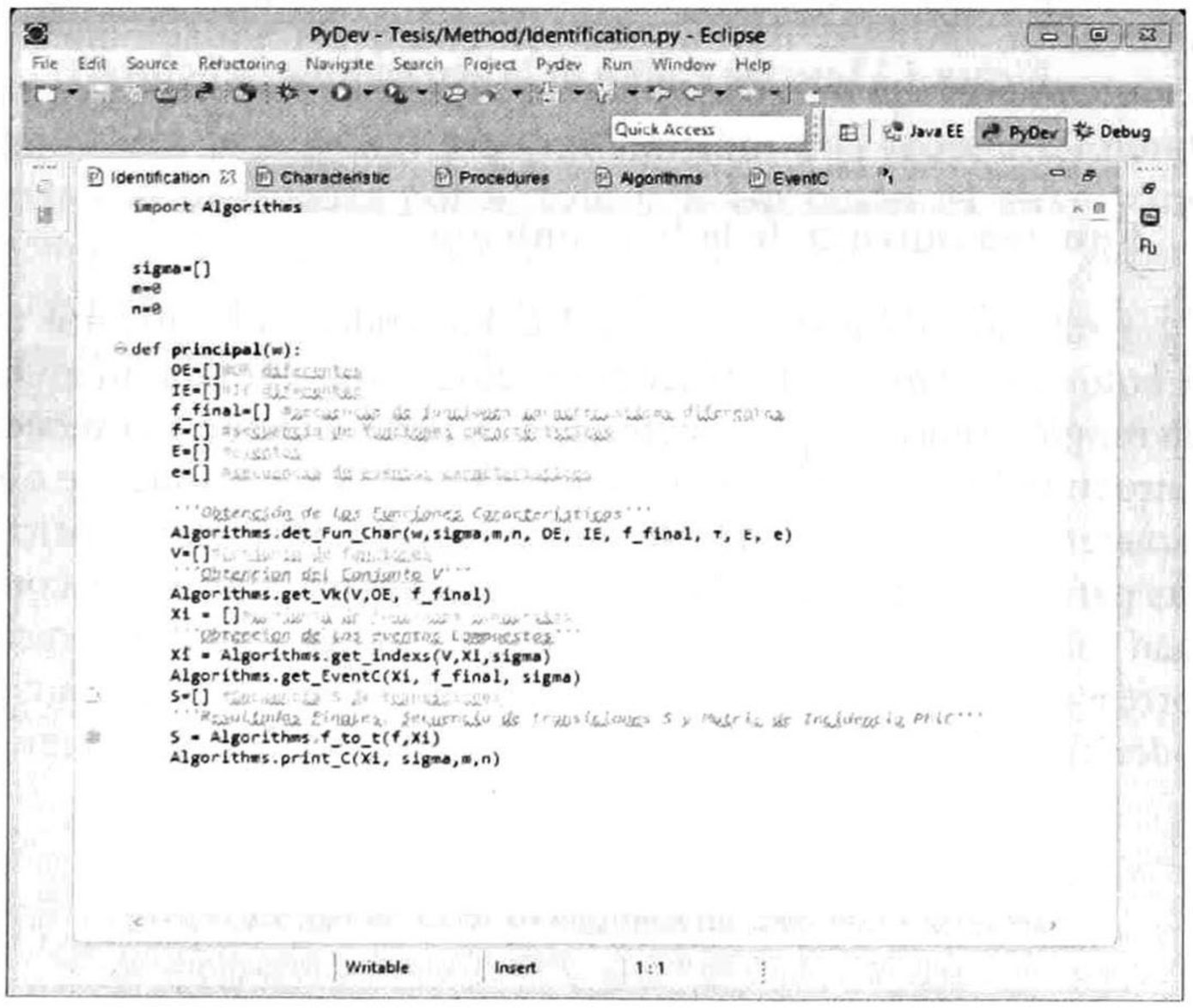


Figura 4.1 Método principal de la herramienta de software.

4.1.2. Interfaz de usuario

La interfaz de usuario se puede observar en la figura 4.2. Para realizar la identificación hay que seleccionar el archivo de texto de entrada, el cual debe contener la secuencia de E/S w a identificar; se debe de introducir también el número de entradas y salidas correspondientes, así como las etiquetas que se le ha dado al sistema, colocando primero el alfabeto de entrada Σ y después el alfabeto de salida Φ, separados por un espacio. La interfaz contiene un *checkbox* para indicar si en el archivo aparecen primero las entradas y luego las salidas o viceversa. El cuadro de texto final mostrará información acerca de la secuencia identificada. El botón "*Start*" inicia el proceso de identificación.

Figura 4.2 Interfaz gráfica de la herramienta de Software.

4.1.3. Funcionamiento de la herramienta

La figura 4.3 muestra de manera general como trabaja el software una vez que se ha presionado el botón de *"Start"*. El software recibe como entrada un archivo de texto (.txt) en el cual cada renglón representa un vector de E/S perteneciente a la secuencia w. El archivo es leído y reestructurado para obtener la secuencia de E/S w, después se obtiene la secuencia e' (eventos característicos), a la cual se le asigna una función característica; estas secuencias son procesadas para la obtención de los eventos finales que llamamos compuestos. Una vez que se obtienen los eventos compuestos, se asigna a cada función característica su evento compuesto correspondiente; con ello se construye la secuencia S de transiciones de T y una matriz de incidencia $\varphi C'$ representando el modelo observable del sistema.

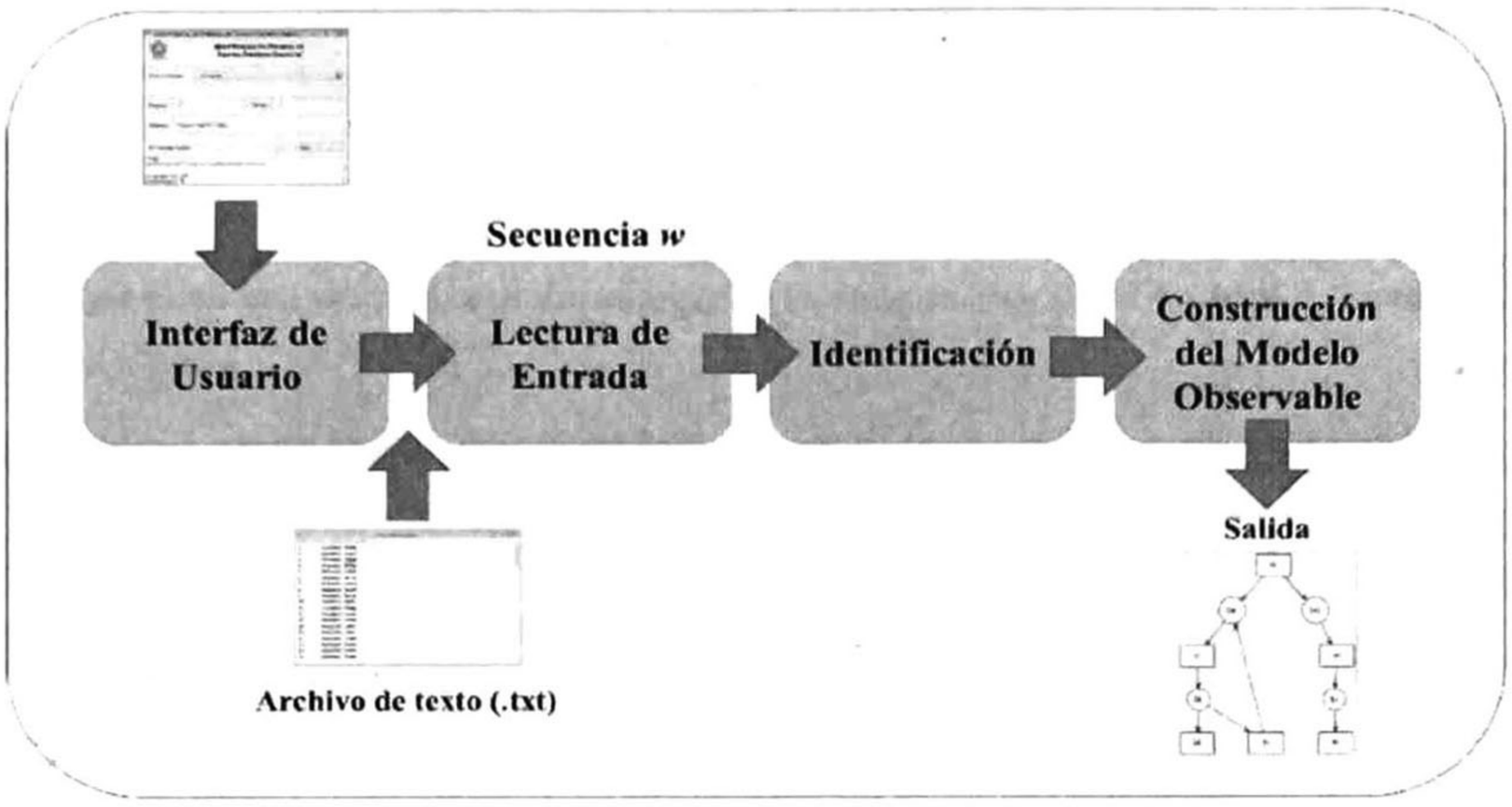


Figura 4.3 Arquitectura del Software

4.2. Aplicación del método a casos de estudio

A continuación se incluyen varios ejemplos de identificación de procesos utilizados para probar la correcta obtención de estructuras diversas en modelos observables del sistema. Las secuencias de E/S *w* utilizadas como entrada al software no se muestran completas por razones de espacio.

4.2.1. Caso Estudio 1: Dos vagones con botón M

El primer caso de estudio que se analiza es el del ejemplo 2.8 que se ha utilizado en el capítulo anterior para ilustrar los pasos del método en capítulo 3. Se trata del sistema compuesto de dos vagones que se desplazan sobre vías independientes por acción de las señales D_i (movimiento a la derecha) e I_i (movimiento a la Izquierda) donde $i= 1,2$. Las posiciones extremas son detectadas por a, c, d y f. Un botón M sirve para comenzar la operación del sistema desde su posición inicial. Las velocidades de los vagones pueden ser cualesquiera. Se reproducen aquí las figuras (4.4 y 4.5) del proceso y del modelo de comportamiento (en principio desconocido) por comodidad.

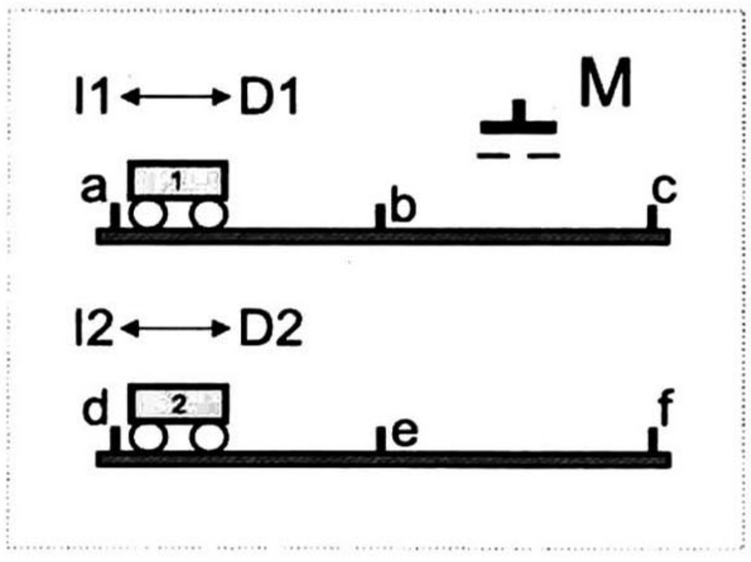


Figura 4.4 Caso de estudio vagones de tren con boton M

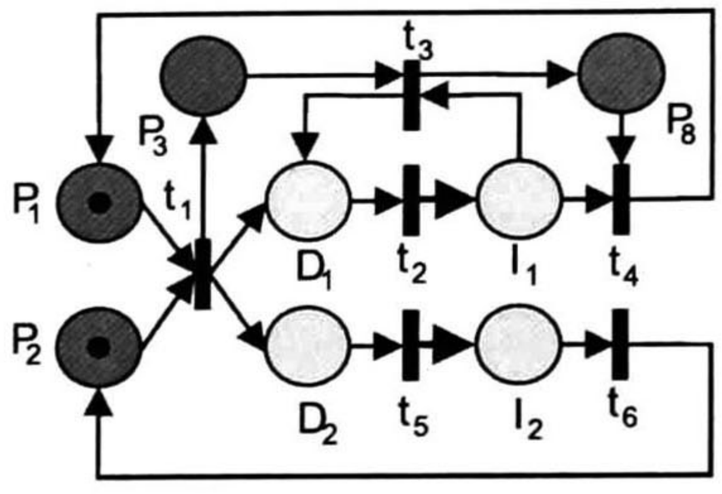


Figura 4.5 Red de Petri que modela el comportamiento del proceso de la figura 4.4

Se consideró el archivo de entrada *w final.txt*, el cual consta de 101 vectores de entrada y salida, con un alfabeto Σ ={M, a, b, c, d, e, f} y un alfabeto Φ ={D1, I1, D2, I2}. En la figura 4.6 solo se muestran los primeros vectores de E/S.

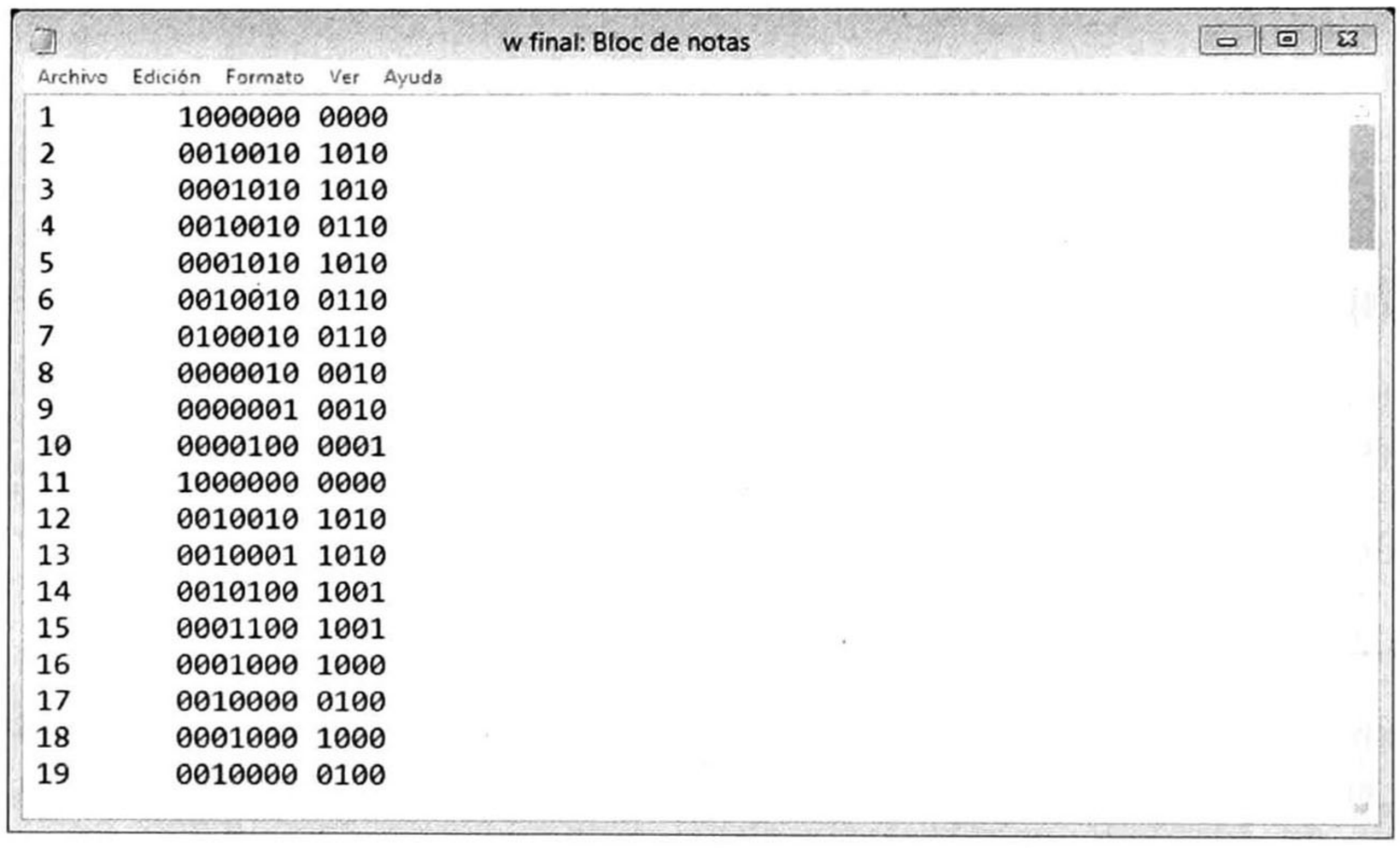


Figura 4.6 Archivo de entrada que contiene la secuencia de E/S w.

Se colocó la información correspondiente en la interfaz gráfica de la herramienta de software como se ilustra en la figura 4.7. Después de presionar el botón "Start", el método de identificación termina satisfactoriamente calculando las funciones características correspondientes a cada evento característico y las funciones finales mostradas en las pantalla de la figura 4.8. Después se obtiene el Conjunto V, y lo tenemos representado en la pantalla figura 4.9.

Figura 4.7 Interfaz gráfica de la herramienta de software que contiene los datos correspondientes al caso de estudio 4.2.1

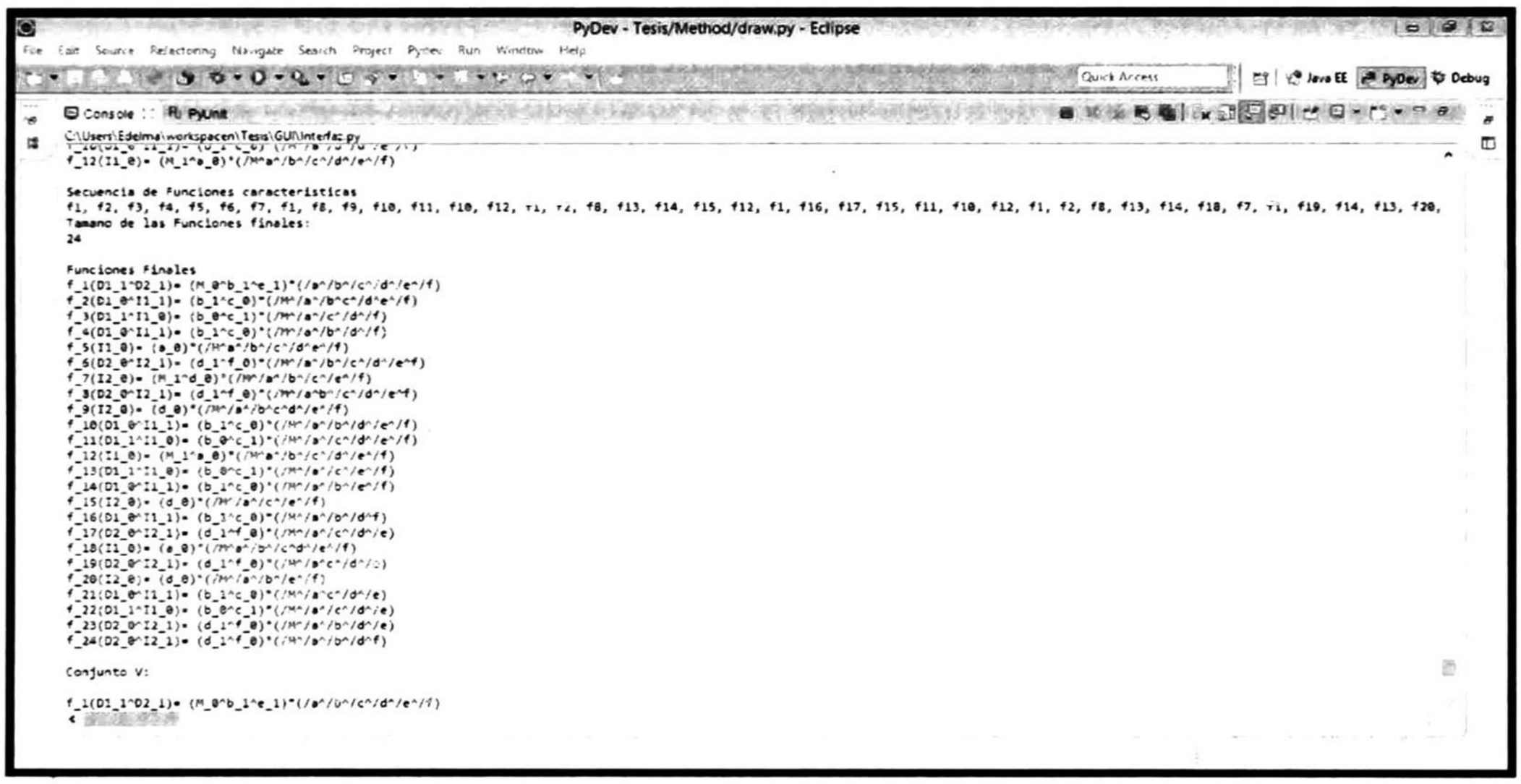


Figura 4.8 Pantalla que contiene las funciones finales que se obtienen del caso de estudio 4.2.1.

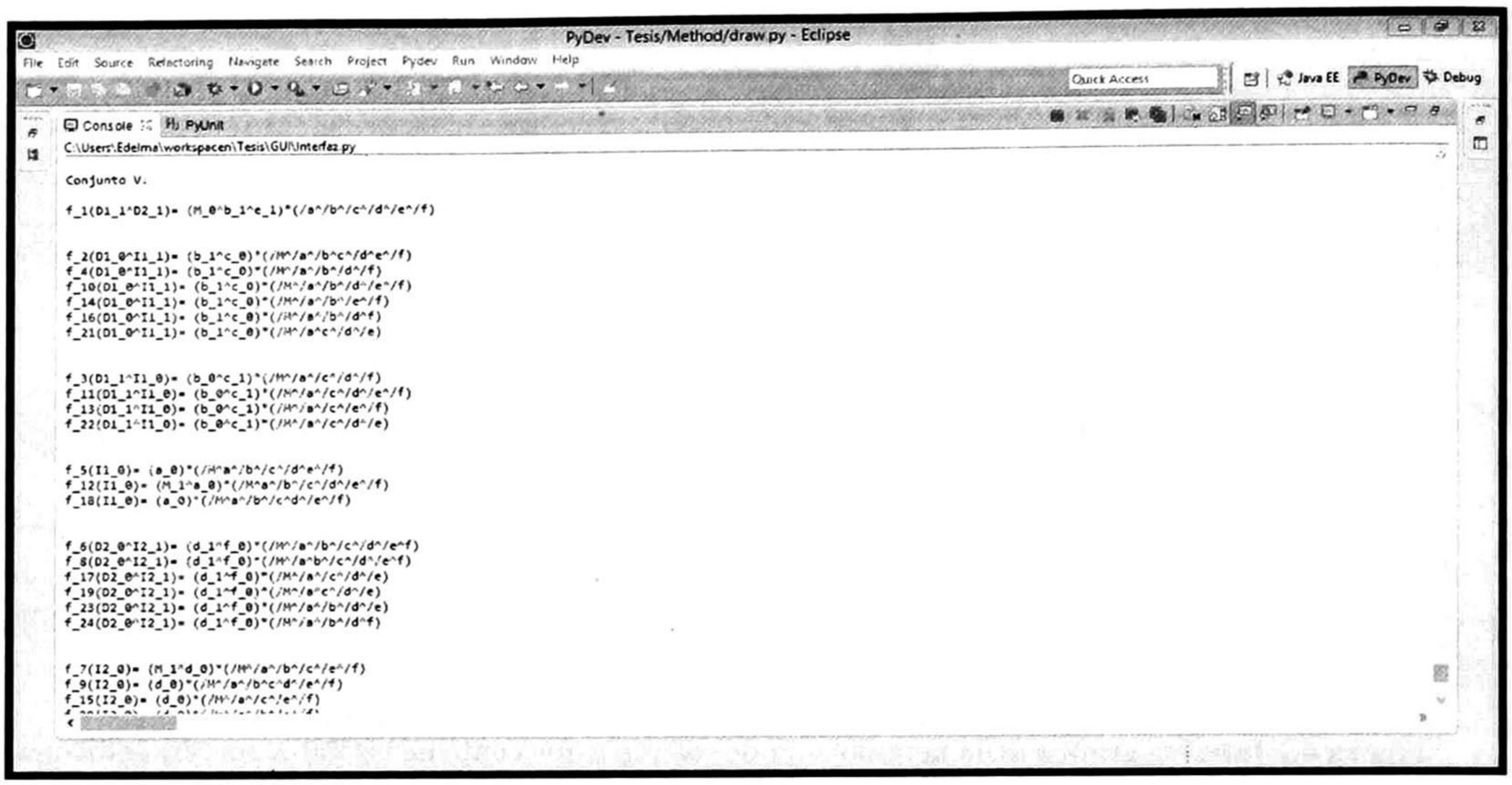


Figura 4.9 Pantalla que contiene el conjunto V que se obtienen del caso de estudio 4.2.1.

Después de obtener el conjunto V, se tiene como resultados finales: los eventos compuestos, la secuencia de transiciones S y la matriz de incidencia $\varphi C'$. Se puede observar en la figura 4.10 los datos que resultan de procesar la secuencia w de E/S mostrada en la figura 4.6.

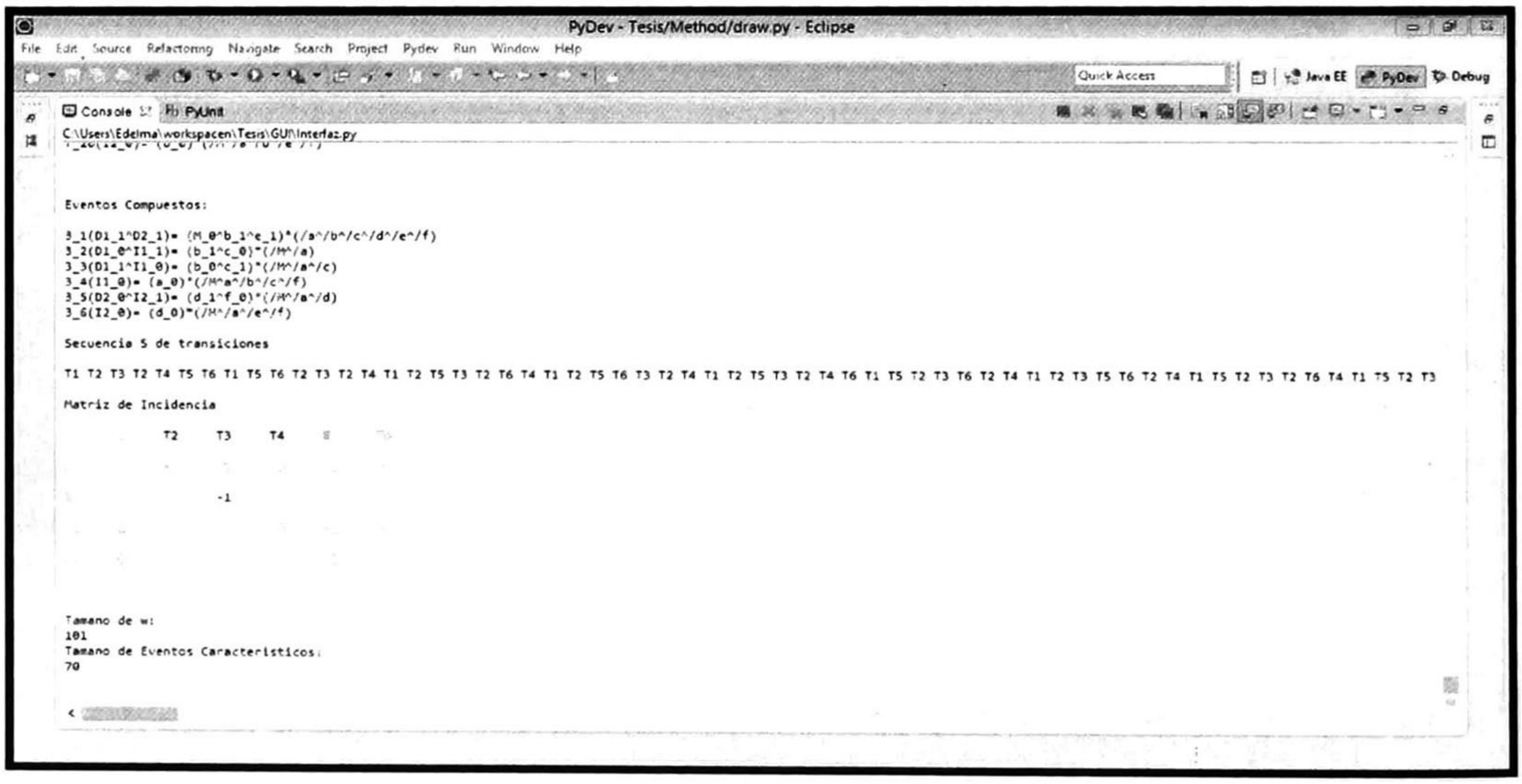


Figura 4.10 Pantalla que contiene los eventos compuestos, la secuencia de transiciones S y la matriz de incidencia φC' que se obtienen del caso de estudio 4.2.1

La RPI correspondiente a la matriz de incidencia, la cual es dibujada por Graphviz se muestra en la figura 4.10. La figura 4.11 muestra la misma RPI redibujada.

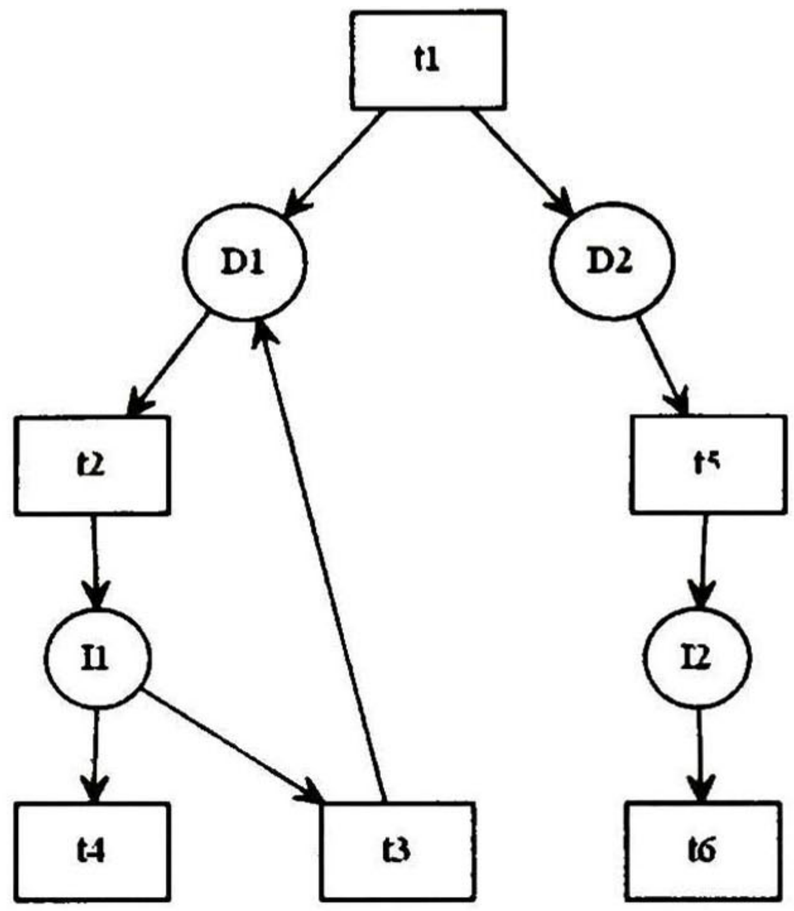


Figura 4.11 Modelo observable para el caso de estudio 4.2.1.

Dibujado de manera que pueda entenderse mejor tenemos la figura 4.12. Que es un fragmento de la RP mostrada en la figura 4.5.

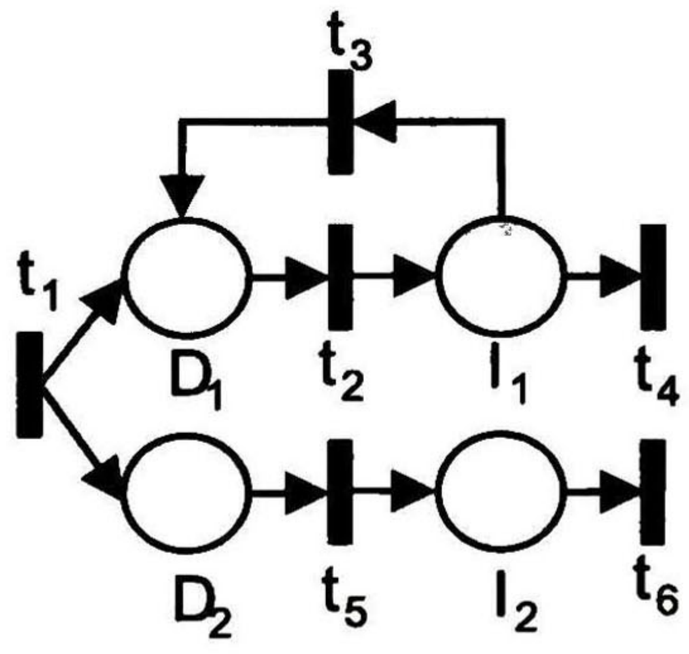


Figura 4.12 Red de Petri que contiene el modelo observable para el caso de estudio 4.2.1.

La secuencia de transiciones que se obtiene para este caso de estudio es la siguiente:

S= T1 T2 T3 T2 T4 T5 T6 T1 T5 T6 T2 T3 T2 T4 T1 T2 T5 T3 T2 T6 T4 T1 T2 T5 T6 T3 T2 T4 T1 T2 T5 T3 T2 T4 T6 T1 T5 T2 T3 T6 T2 T4 T1 T2 T3 T5 T6 T2 T4 T1 T5 T2 T3 T2 T6 T4 T1 T5 T2 T3 T6 T2 T4 T1 T5 T6 T2 T3 T2 T4

Para este caso de estudio procesaremos la secuencia de transiciones S para obtener el modelo no observable para ilustrar el método de identificación completo. Utilizando el software desarrollado para el método de minería de procesos propuesto en [Tapia, 2013], la RP obtenida, la cual reproduce la secuencia *S* se muestra en la figura 4.13. La integración de este modelo con el modelo observable de la figura 4.12 se muestra en la figura 4.14. Posteriormente, al eliminar lugares implícitos no observables, se obtiene el modelo final mostrado en la figura 4.15.

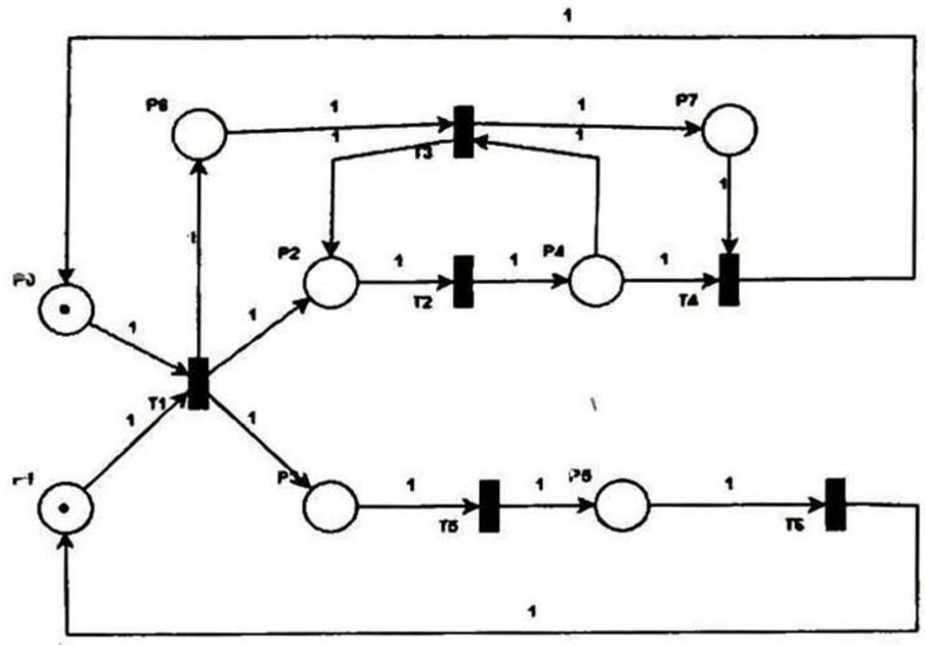


Figura 4.13 Modelo no observable a partir de la Secuencia S.

Si se fusionan ambos modelos se obtiene el modelo de RPI final representado en la figura 4.5

4.2.2. Caso Estudio 2: Vagones de tren sin botón M

Al igual que en 4.2.1 el caso de estudio 2 consiste también en un sistema que está compuesto de dos vagones que se desplazan sobre vías independientes por acción de las señales D_i (movimiento a la derecha) e I_i (movimiento a la Izquierda) donde i= 1,2. Las posiciones extremas son detectadas por a, b, c y d. En este caso no se cuenta con un botón de inicio M. Las velocidades de los vagones se desconocen.

Para este caso de estudio se utilizó una secuencia de E/S w de longitud 1178 con un alfabeto Σ ={a, b, c, d} y un alfabeto Φ={D1, D2, I1, I2}. Se obtiene una secuencia e' de tamaño 588.

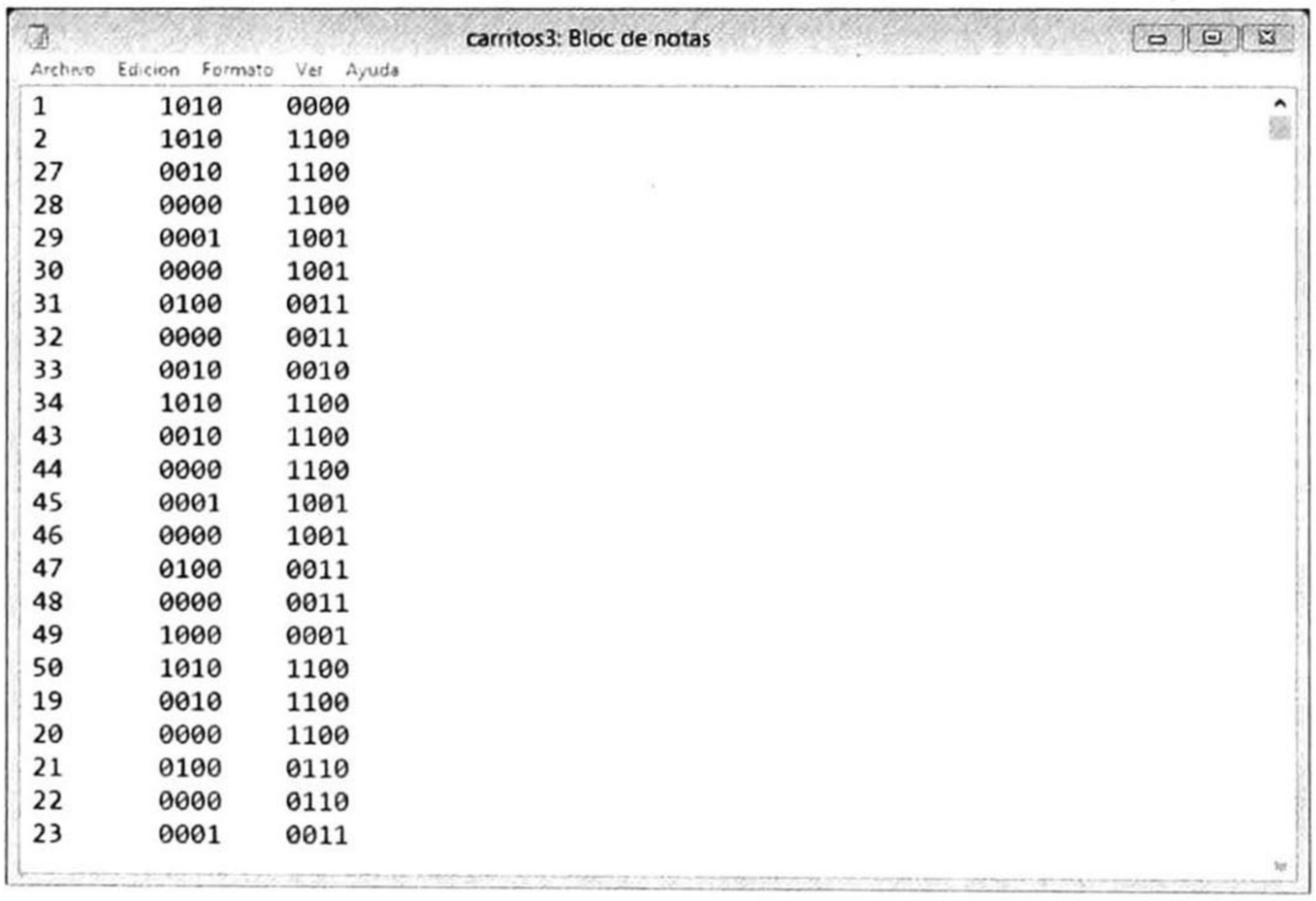


Figura 4.14 Archivo de texto que contiene la secuencia de E/S del caso de estudio 4.2.2

Se muestra la ejecución del software basado en los alforitmos del método a través de capturas de pantalla En la figuras 4.15 y 4.15 se observan las funciones finales y el conjunto V respectivamente. El resultado final es mostrado en la pantalla de la figura 4.17, donde se ve la secuencia de transiciones y la matriz de incidencia observable.

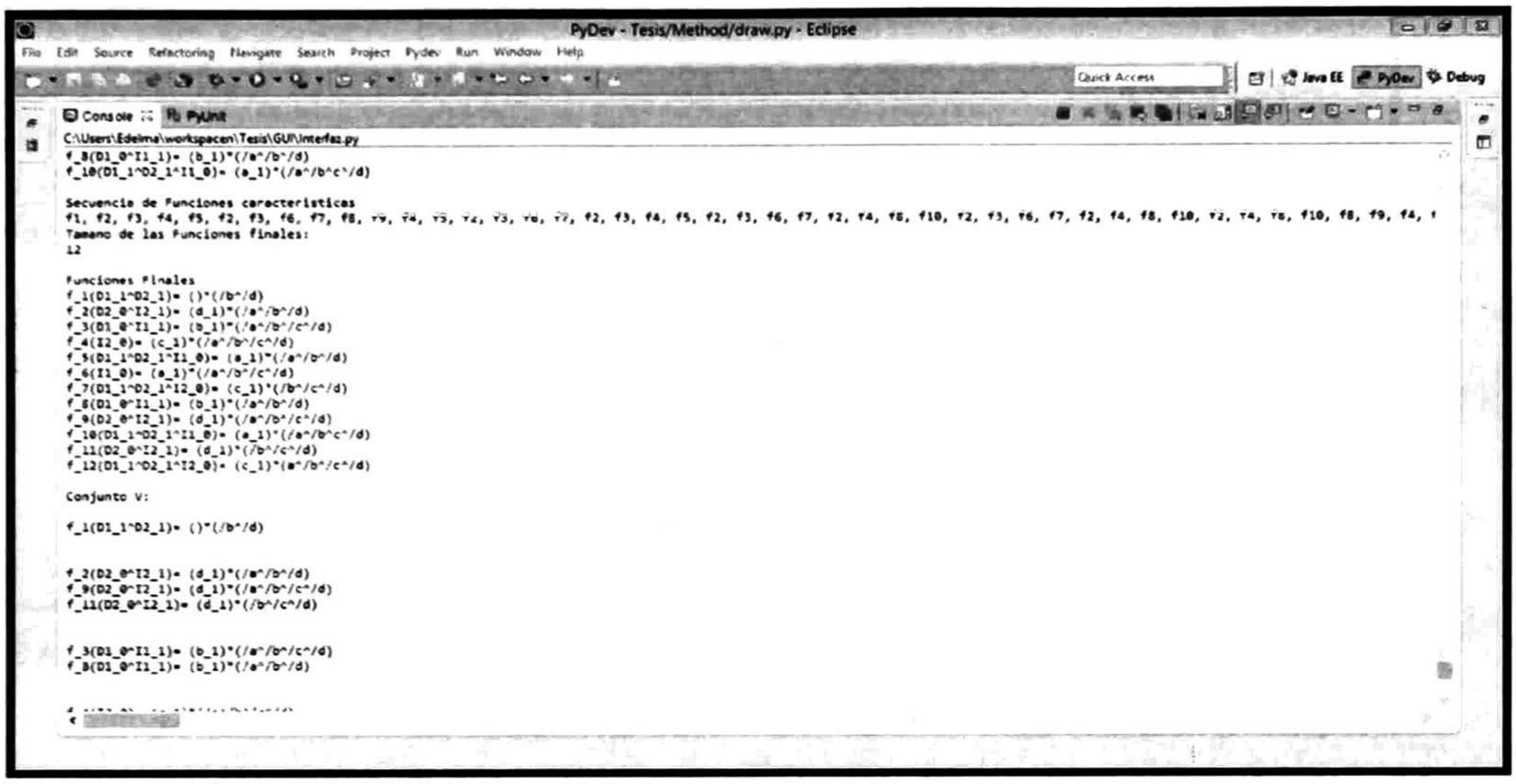


Figura 4.15 Captura de pantalla que muestra las funciones finales del caso de estudio 4.2.2.

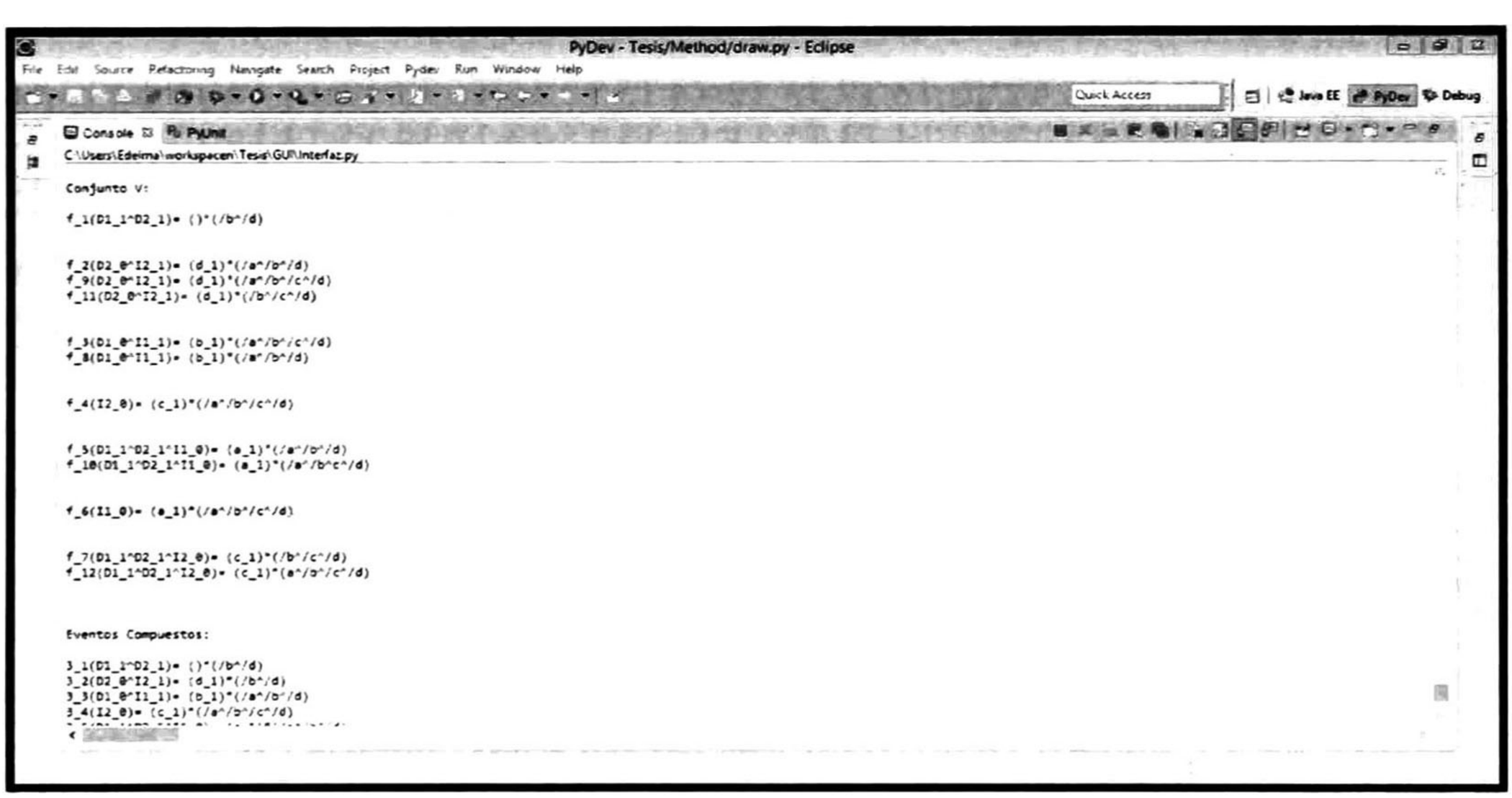


Figura 4.16 Captura de pantalla que muestra el conjunto V del caso de estudio 4.2.2.

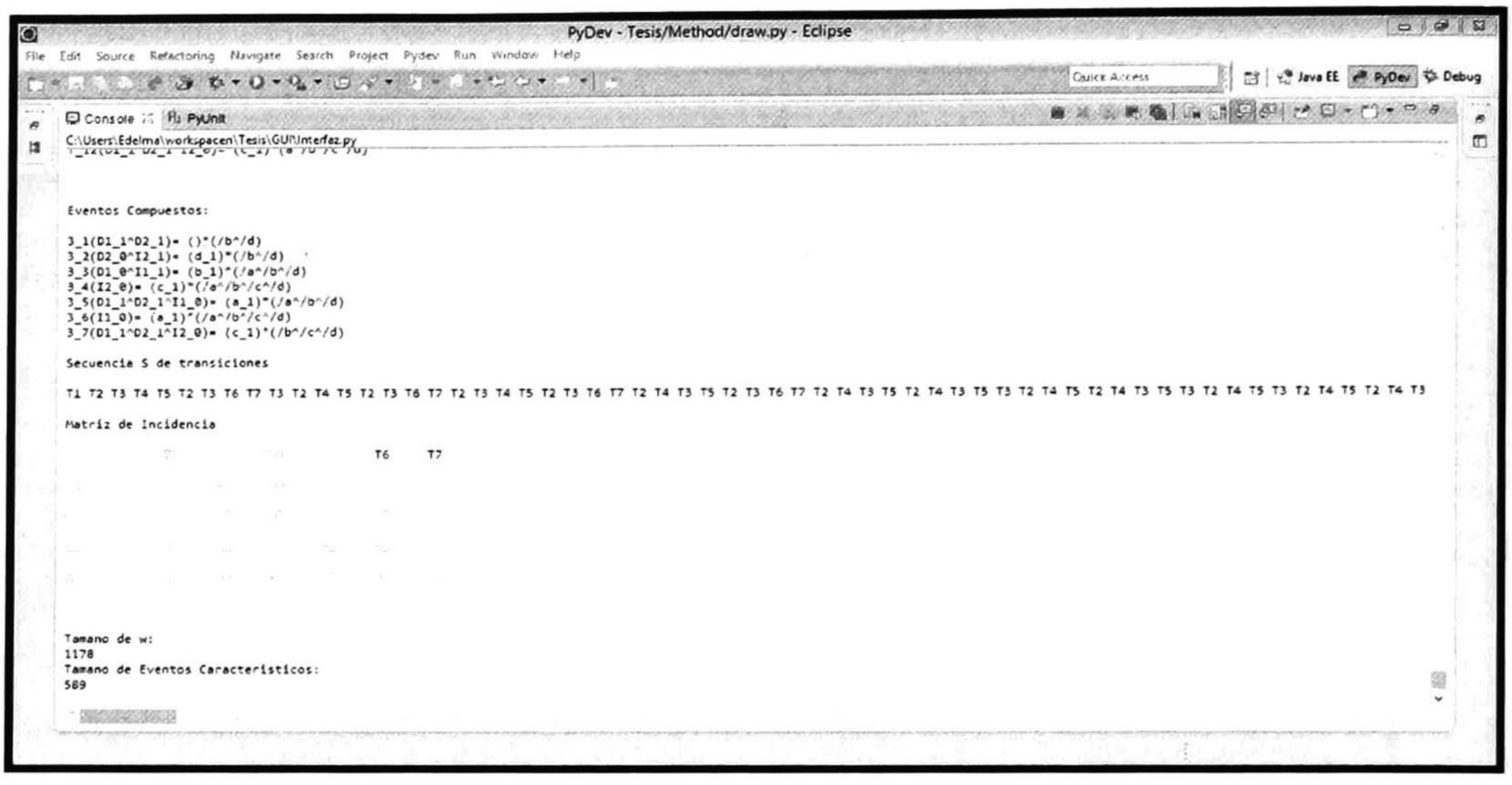


Figura 4.17 Captura de pantalla que muestra los eventos compuestos, la secuencia de transiciones S y la matriz de incidencia φC' del caso de estudio 4.2.2.

Se reproduce aquí una parte de la secuencia de transiciones S la cual tiene una longitud de 589.

S = T1 T2 T3 T4 T5 T2 T3 T6 T7 T3 T2 T4 T5 T2 T3 T6 T7 T2 T3 T4 T5 T2 T3 T6 T7 T2 T4 T3 T5 T2 T3 T6 T7 T2 T4 T3 T5 T2 T4 T3 T5 T3 T2 T4 T5 T2 T4 T3 T5 T3 T2 T4 T5 T3 T2 T4 T5 T2 T4 T3 T5 T3 T2 T6 T7 T3 T2 T4 T5 T2 T4 T3 T5 T3 T6 T2 T7 T2 T3 T6 T7 T2 T4 T3 T5 T2 T3 T6 T7 T3 T2 T4 T5 T2 T3 T6 T7 T3 T6 T2 T7 T2 T3 T6 T7 T3 T2 T6 T7 T2 T3 T6 T7 T2 T4 T3 T5 T2 T3 T4 T5 T2 T3 T4 T5 T2 T4 T3 T5 T3 T2 T6 T7 T3 T2 T6 T7 T3 T6 T2 T7 T3 T2 T4 T5 T2 T3 T6 T7 T3 T2 T4 T5 T2 T3 T6 T7 T3 T6 T2 T7 T2 T3 T6 T7 T2 T4 T3 T5 T2 T3 T6 T7 T3 T6 T2 T7 T2 T3 T6 T7 T2 T4 T3 T5 T2 T3 T6 T7 T2 T4 T3 T5…

La RPI que corresponde a la matriz de incidencia que se muestra en la figura 4.17. La cual se presenta redibujada en la figura 4.18 con propósito de comparación con el modelo obtenido en [Estrada, 2013]. El resultado es que se trata de una red idéntica.

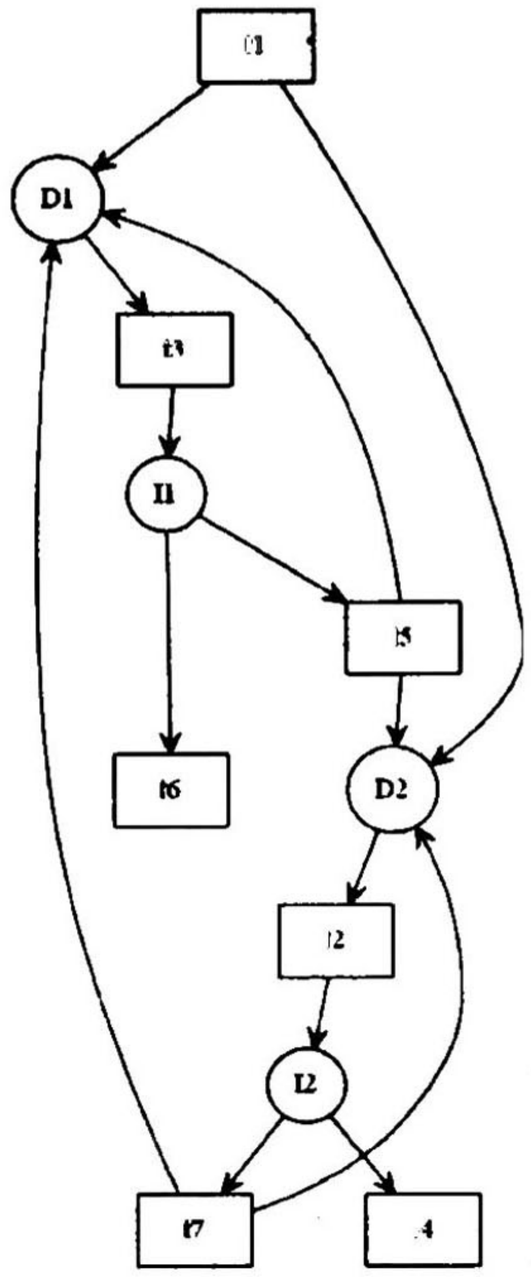


Figura 4.18 Modelo observable para el caso de estudio 4.2.2.

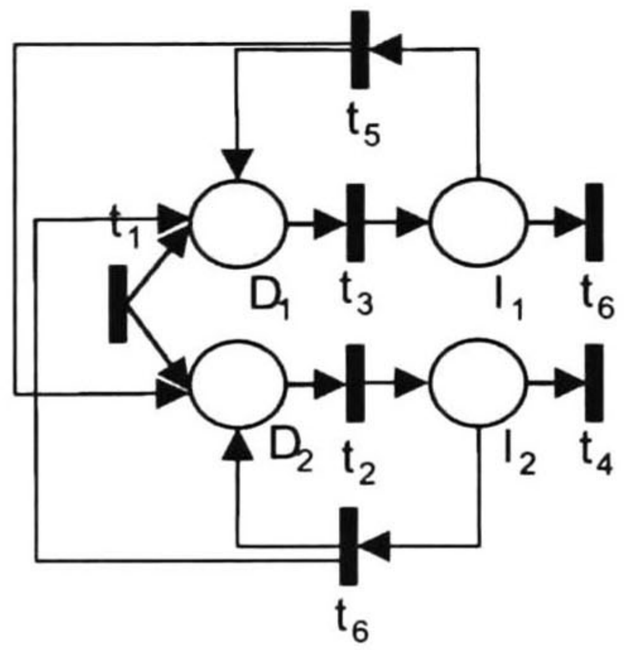


Figura 4.19 RPI para el caso de estudio 4.2.2.

4.2.3. Caso Estudio 3: Ordenar paquetes

Se consideró el caso de estudio tratado en el ejemplo 2.9, del cual reproducimos su figura aquí (Figura 4.19). El objetivo del sistema es ordenar paquetes de acuerdo a su tamaño, este tiene 9 entradas en total (k1, k2, a0, a1, a2, b0, b1, c0, c1) de las cuales 7 señales son generadas por sensores para detectar la posición de los cilindros (a0, a1, a2, b0, b1, c0, c1) y 2 señales que indican la presencia de los paquetes (k1, k2). Cuatro señales de salida que controlan los actuadores de la planta (A+, A-, B, C), la forma en que se presentan los vectores E/S es la siguiente: [A+ A- B C k1 k2 a0 a1 a2 b0 b1 c0 c1]T.

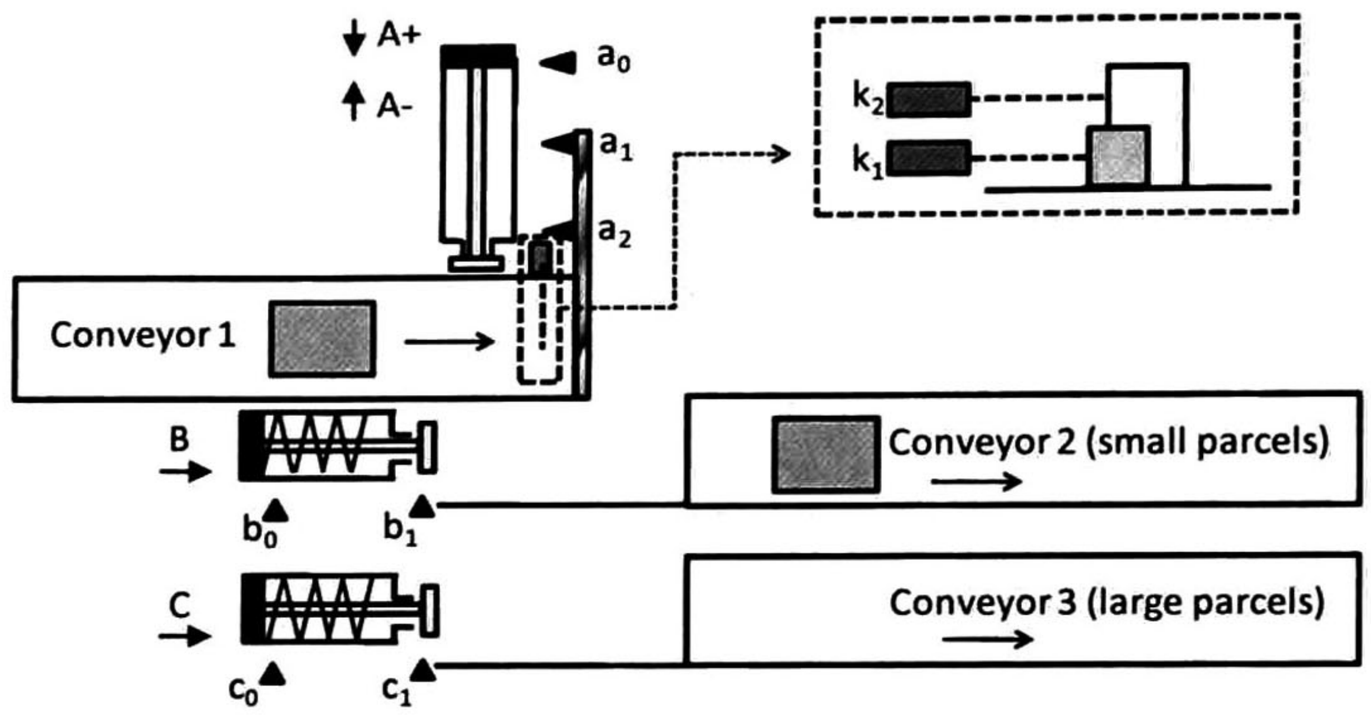
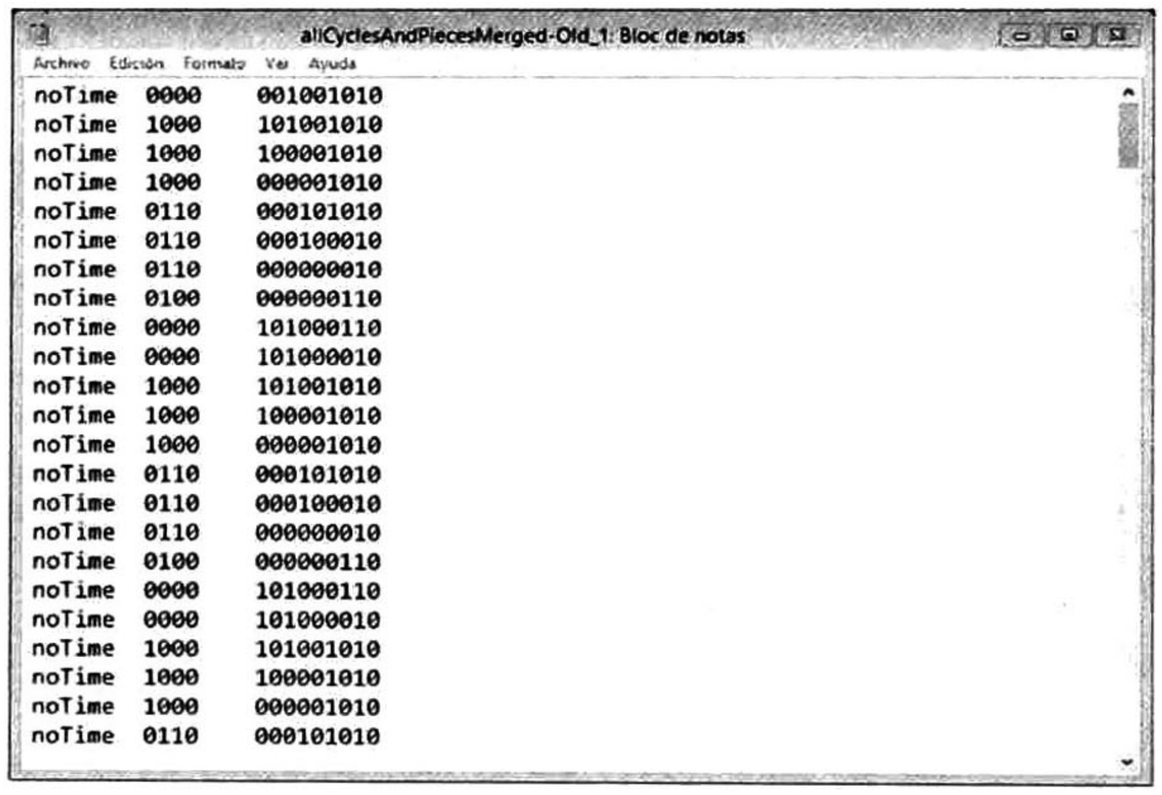


Figura 4.20 Sistema de manufactura Ejemplo 2.1 [Estrada, 2013]

Se procesa una secuencia de E/S de tamaño 222 y el total de eventos característicos obtenidos fue de 85. Con $\Sigma = \{k1, k2, a0, a1, a2, b0, b1, c0, c1\}$ y $\Phi = \{A+, A-, B, C\}$. Parte de la secuencia es mostrada en la pantalla de la figura 4.21


allCyclesAndPiecesMerged-Old_1: Bloc de notas

Archivo Edición Formato Ver Ayuda

```
noTime  0000    001001010
noTime  1000    101001010
noTime  1000    100001010
noTime  1000    000001010
noTime  0110    000101010
noTime  0110    000100010
noTime  0110    000000010
noTime  0100    000000110
noTime  0000    101000110
noTime  0000    101000010
noTime  1000    101001010
noTime  1000    100001010
noTime  1000    000001010
noTime  0110    000101010
noTime  0110    000100010
noTime  0110    000000010
noTime  0100    000000110
noTime  0000    101000110
noTime  0000    101000010
noTime  1000    101001010
noTime  1000    100001010
noTime  1000    000001010
noTime  0110    000101010
```

Figura 4.21 Archivo de texto que contiene la secuencia de E/S del caso de estudio 4.2.3

Resultados

En las figuras 4.212, 4.23 y 4.24 se muestran los resultados que se obtiene de procesar la secuencia de E/S *w* del caso de estudio 3.

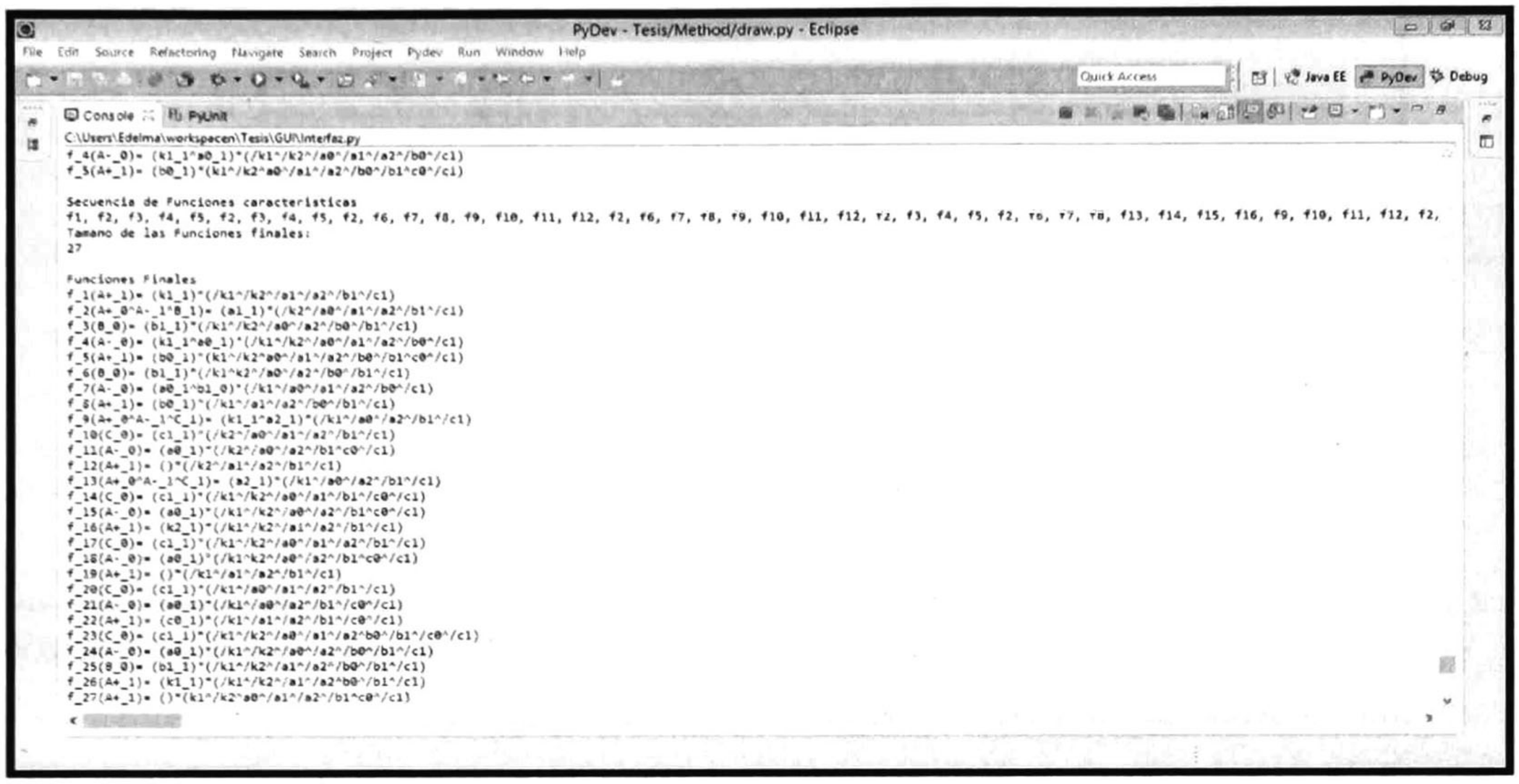


Figura 4.22 Captura de pantalla de las funciones finales para el caso de estudio 4.2.3

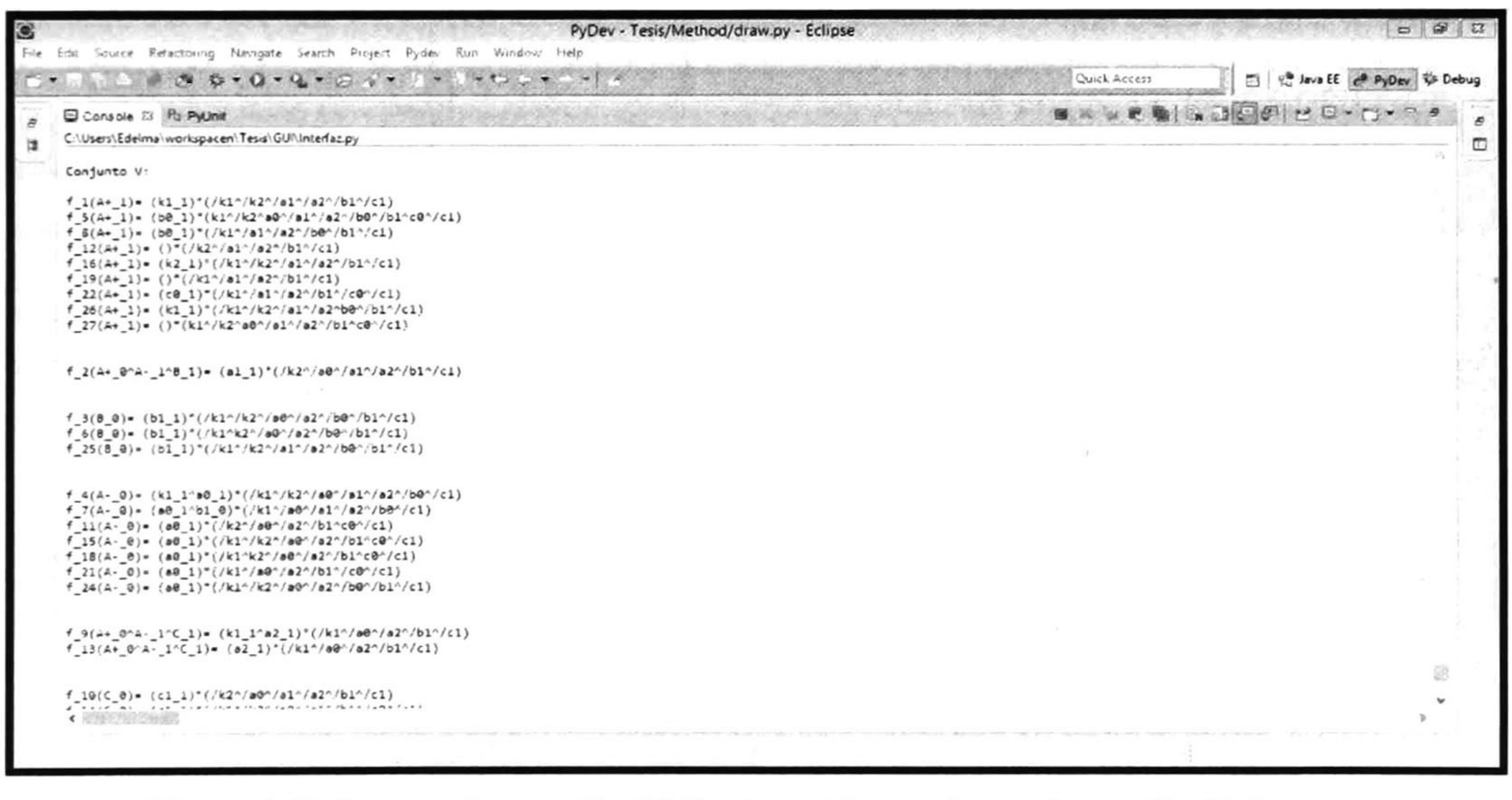


Figura 4.23 Captura de pantalla del Conjunto V para el caso de estudio 4.2.3

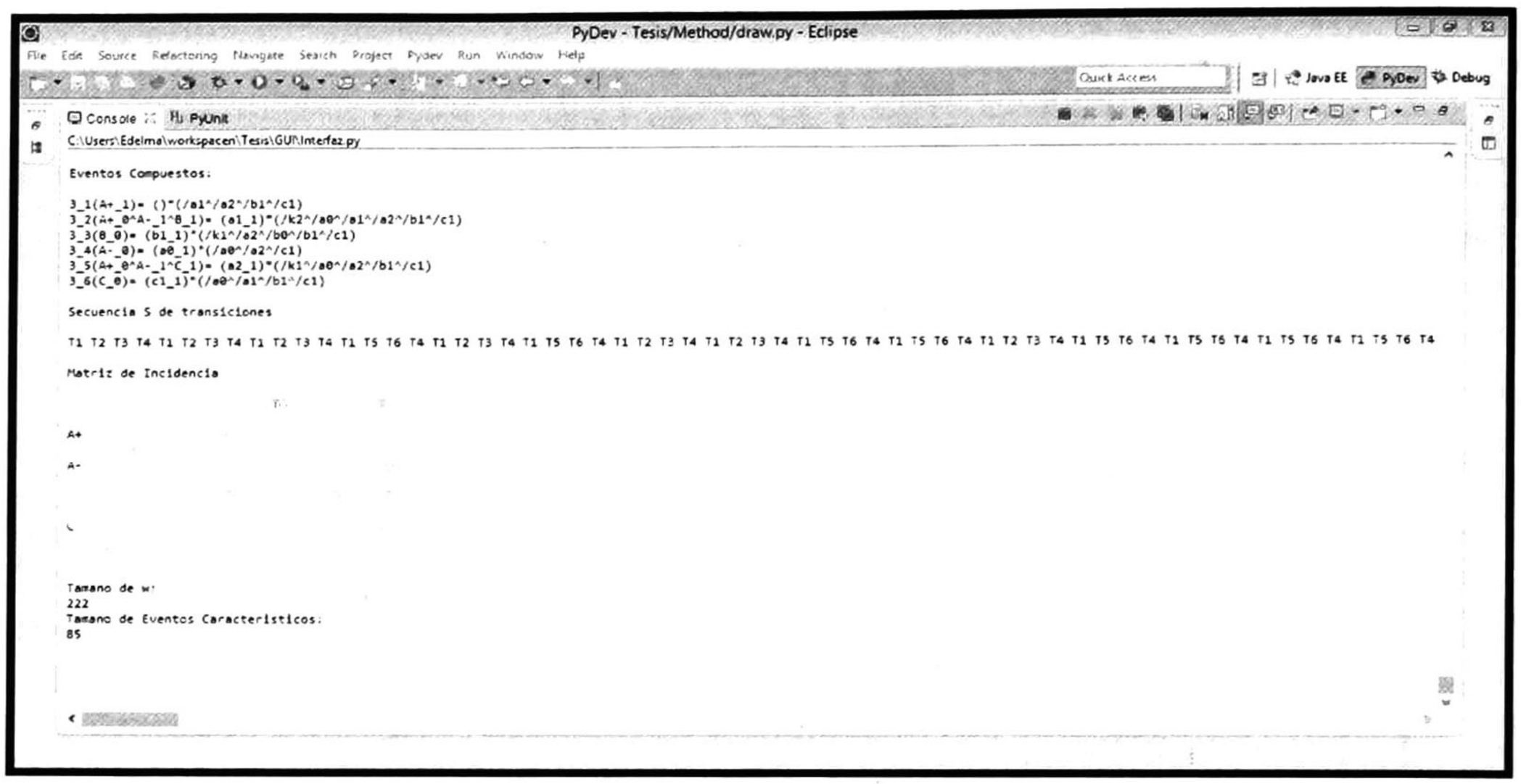


Figura 4.24 Captura de pantalla de eventos compuestos, secuencia de transiciones S y la matriz de incidencia $\varphi C'$ para el caso de studio 4.2.3

La secuencia de transiciones S obtenida es:

S =T1 T2 T3 T4 T1 T2 T3 T4 T1 T2 T3 T4 T1 T5 T6 T4 T1 T2 T3 T4 T1 T5 T6 T4 T1 T2 T3 T4 T1 T2 T3 T4 T1 T5 T6 T4 T1 T5 T6 T4 T1 T2 T3 T4 T1 T5 T6 T4 T1 T5 T6 T4 T1 T5 T6 T4 T1 T2 T4 T3 T1 T2 T3 T4 T1 T2 T3 T4 T1 T2 T4 T3 T1 T2 T3 T4 T1... con |S|= 85

La RPI que corresponde a la matriz de incidencia que se muestra en la impresión de pantalla de la figura 4.24 es mostrada en la figura 4.25:

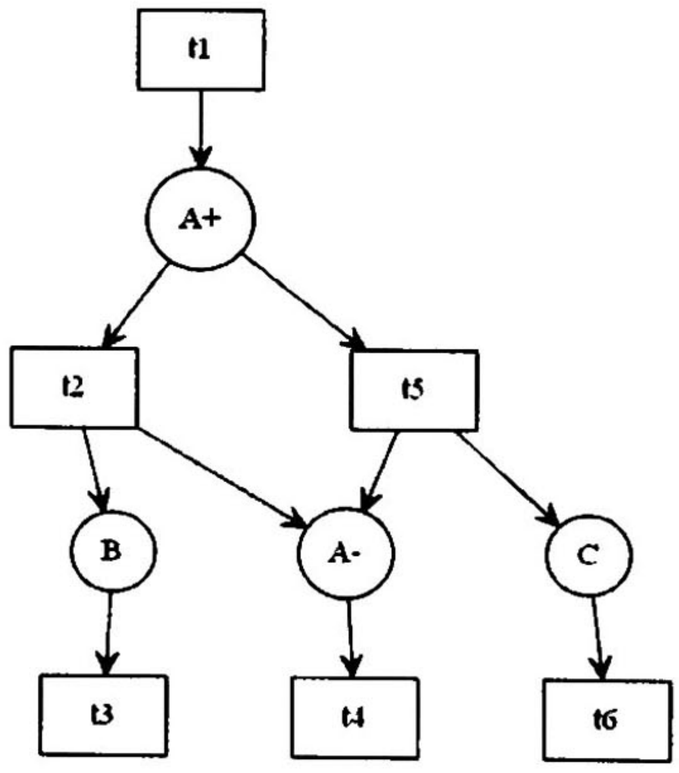


Figura 4.25 Modelo observable para el caso de estudio 4.2.3.

Para una mejor interpretación se ha vuelto a dibujar la red de Petri correspondiente al modelo observable obtenido en el caso de estudio 3, el cual se muestra en la figura 4.26

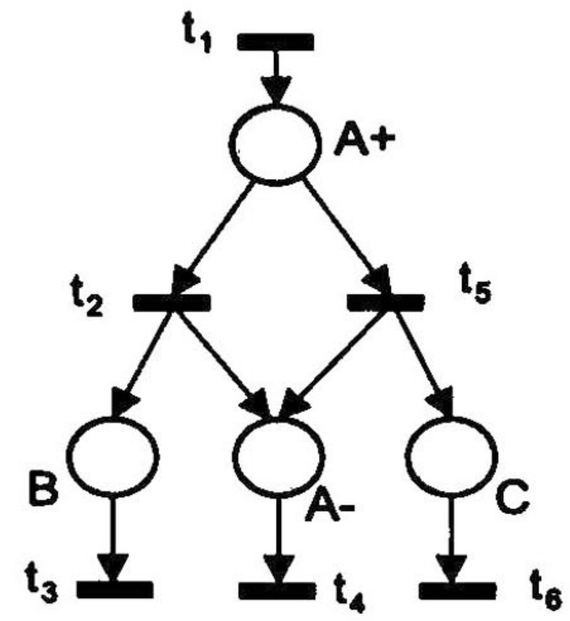


Figura 4.26 RPI para el caso de estudio 4.2.3.

4.2.4. Caso Estudio 4: Sistema de Ordenamiento

Descripción

El sistema de entrenamiento interactivo para PLC (ITS PLC por sus siglas en inglés) edición profesional es una herramienta para programación de PLC que ofrece sistemas virtuales para educación y entrenamiento en programación de PLC. Cada sistema es una simulación visual y de comportamiento de un sistema industrial incluyendo sensores y actuadores, por lo que su estado pueda ser detectado por un PLC real. El objetivo es programar el PLC para controlar cada sistema como si fuera un sistema real. La información de los sensores y actuadores es intercambiada entre el PLC y el sistema a través de una tarjeta de adquisición de datos (DAQ por sus siglas en inglés) con 32 canales E/S aislados y una interface USB.

Para nuestro trabajo experimental, hemos seleccionado el "Sistema de Ordenamiento". El sistema transporta cajas de un alimentador a un par de elevadores, ordenándolos por altura. El sistema consiste de 11 entradas (s0, s1, s2, s3, s4, s5, s6, s7, s8, s9, s10) y 7 salidas (A0, A1, A2, A3, A4, A5, A6). El proceso virtual es el mostrado en la figura 4.27

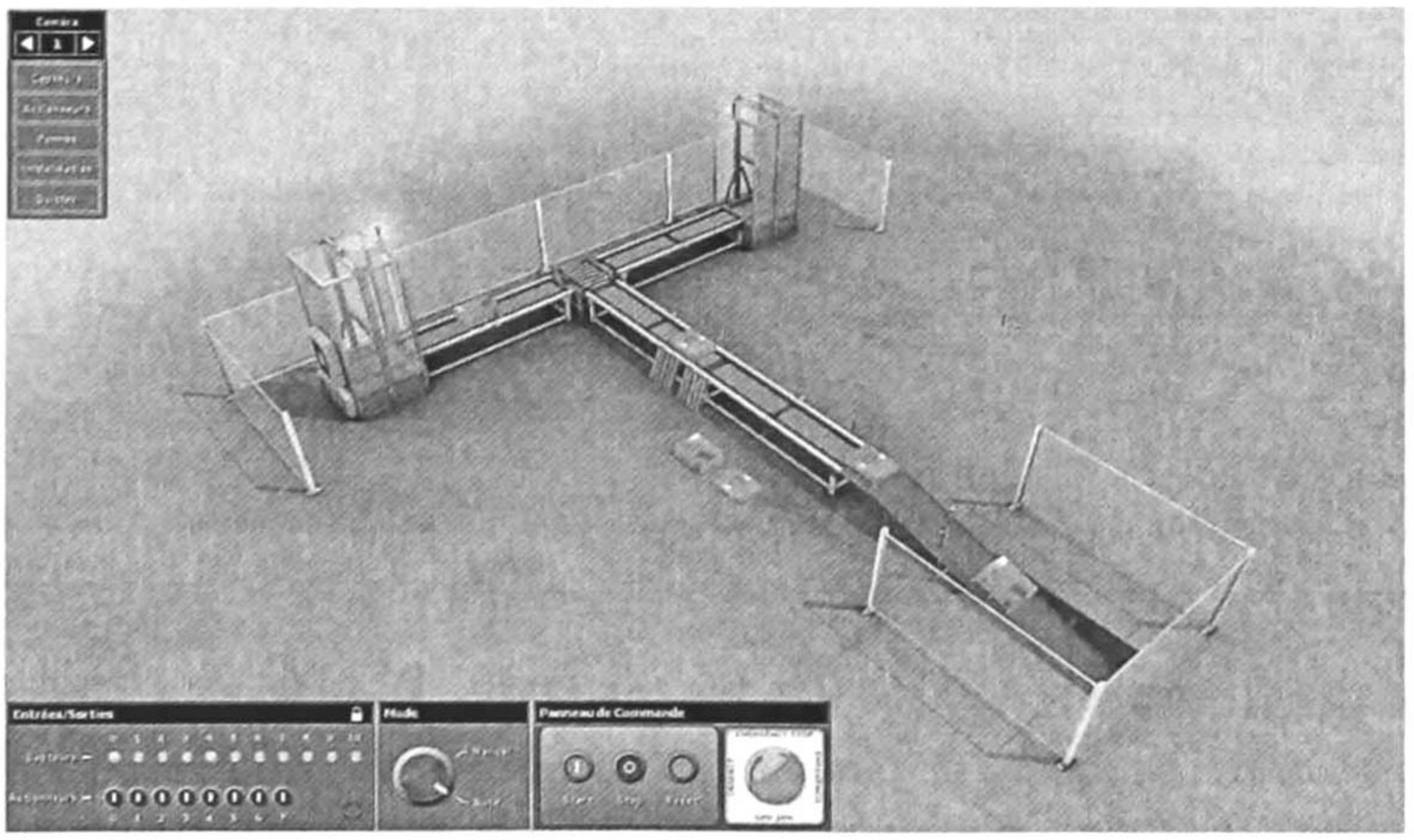


Figura 4.27 El sistema de clasificación de ITS PLC

El comportamiento observado del intercambio de señales entre el proceso y el PLC se registra en un archivo con una secuencia de E/S *w*. Para este caso de estudio la longitud de w es de 4091 vectores, los cuales se muestran parcialmente en la figura 4.27. Durante el procesamiento, la secuencia *e'* tiene una longitud de 2006 eventos característicos

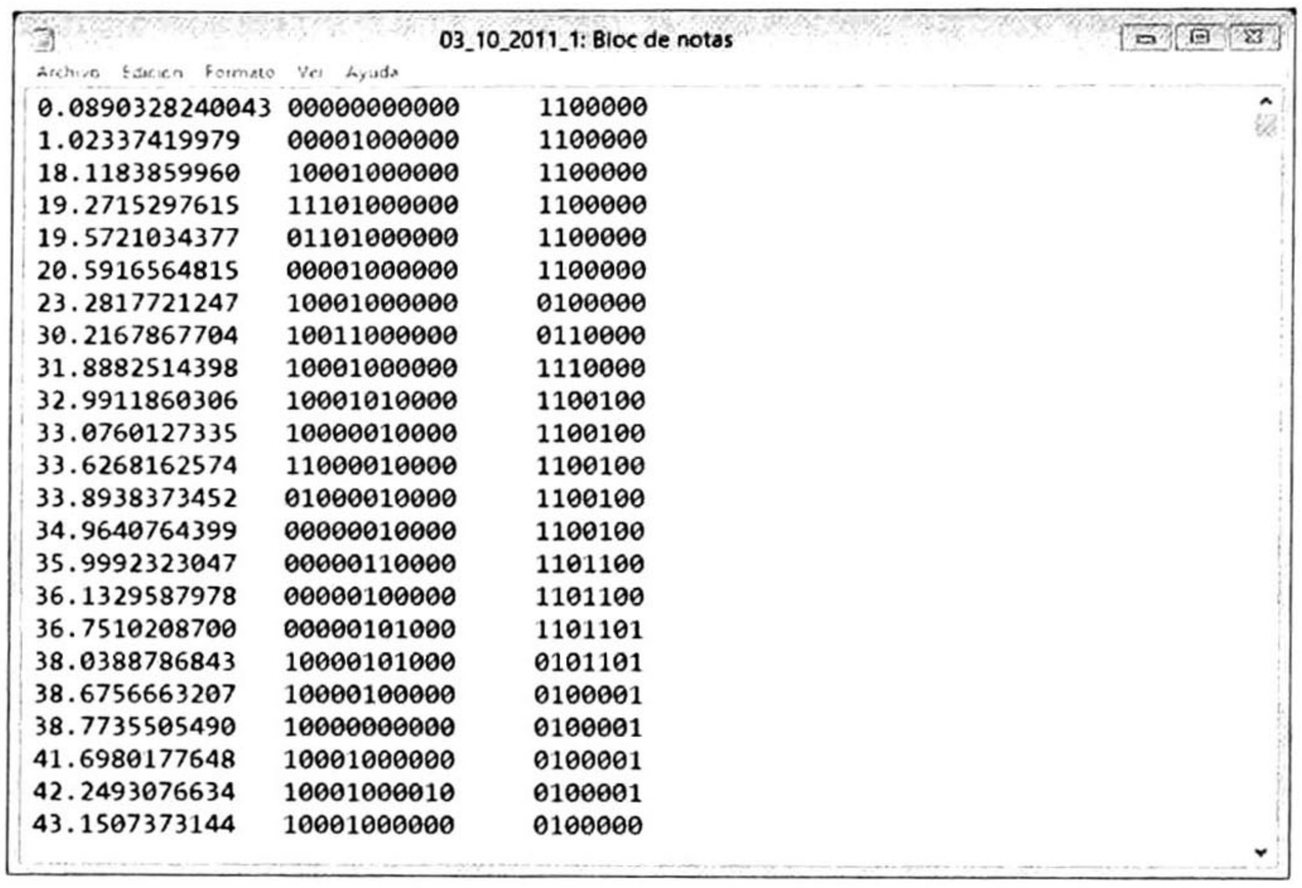


Figura 4.28 Archivo de texto con la secuencia de E/S w del caso 4.2.4.

Resultados

Para este ejemplo se muestran los eventos compuestos, la secuencia de transiciones S y la matriz de incidencia $\varphi C'$. En las pantallas de las figuras 4.29 y 4.30

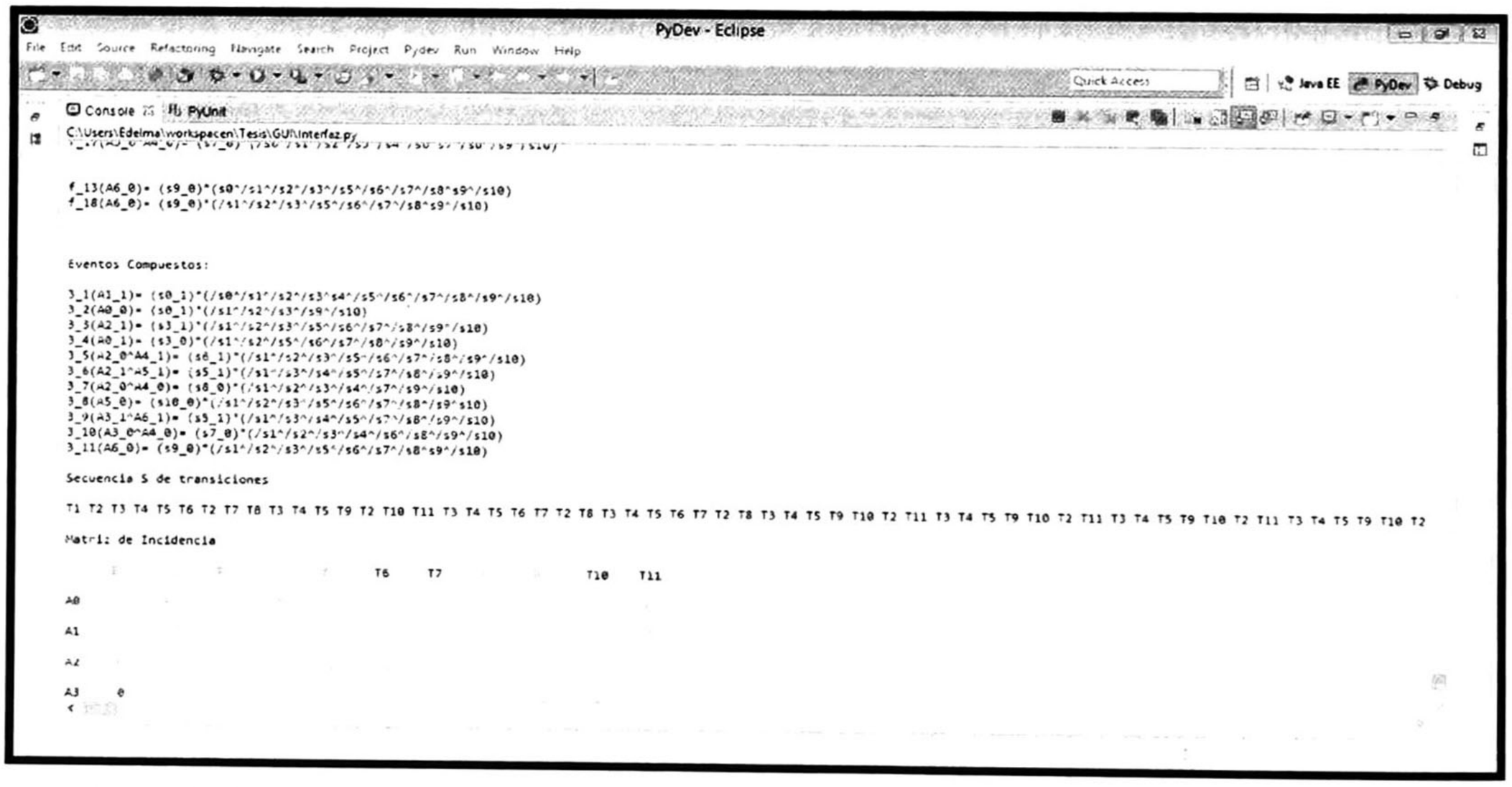


Figura 4.29 Captura de pantalla de los eventos compuesto para el caso de estudio 4.2.4.

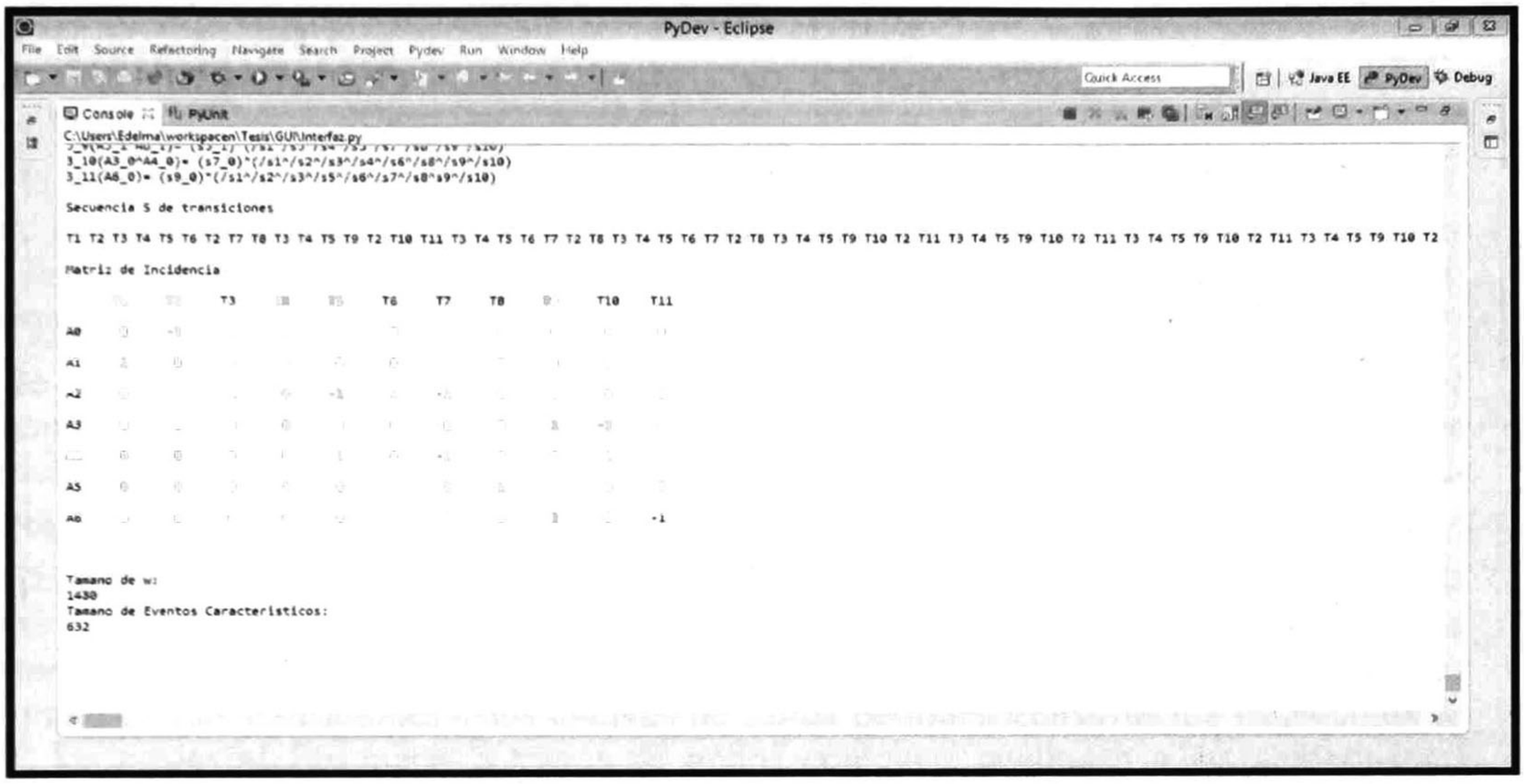


Figura 4.30 Captura de pantalla para el caso de estudio 4.3.4.

La RPI que corresponde a la matriz de incidencia se muestra en la figura 4.30. Esta se ha redibujado para un mejor entendimiento en la figura 4.31

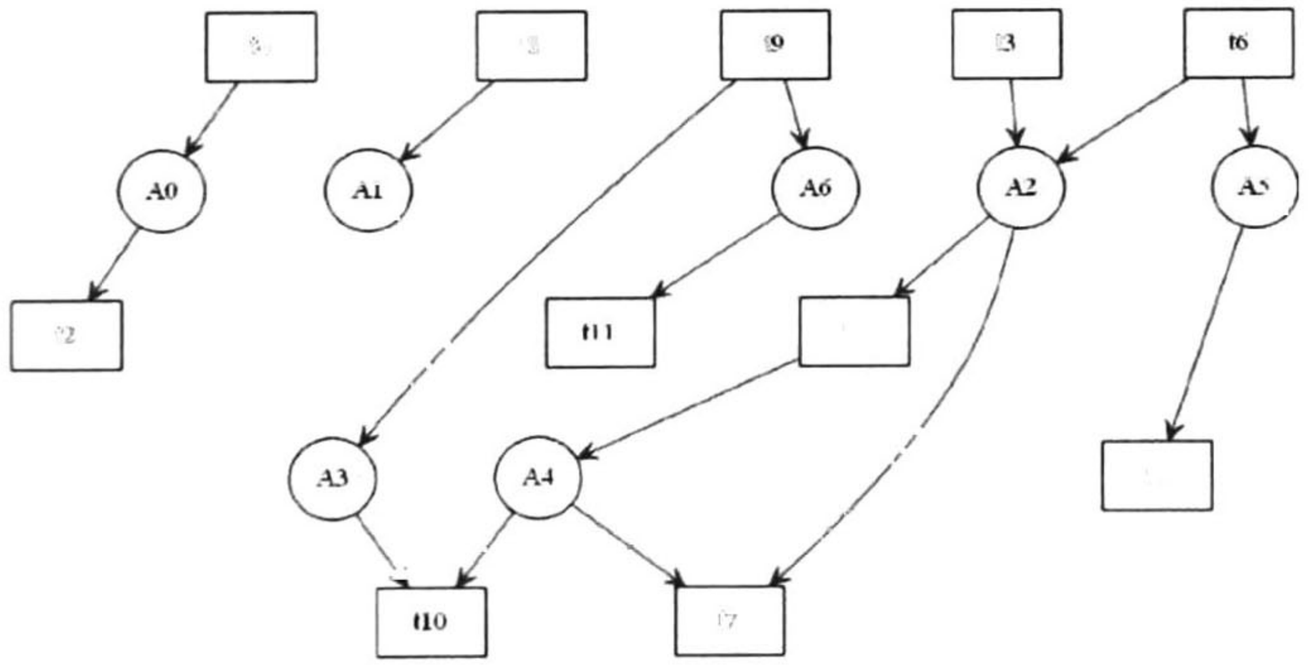


Figura 4.31 Modelo de RPI para el caso de estudio 4.3.4.

4.3. Comentarios sobre las pruebas

La implementación de la herramienta de software para el método de identificación de eventos mostrado en el capítulo 3, ha sido probada para varios casos de estudio. En algunos de los ejemplos se muestra como el algoritmo puede encontrar el comportamiento observable del sistema con menos transiciones que el método implementado en [Estrada, 2013].

Queda por comprobar que la Secuencia de transiciones S, genera un modelo no observable correcto, la cual al fusionarse con el modelo observable genere el modelo final de la RPI que reproduzca exactamente el comportamiento del sistema.

Conclusión

En esta tesis se aborda el problema de identificación de procesos de eventos dicretos, donde a partir de una secuencia w de E/S correspondientes a señales de entrada y salida de un sistema, se obtienen las componentes observables de un modelo en red de Petri interpretada.

Se definió e implementó un método que procesa las secuencias y construye sistemáticamente una RP que representa el comportamiento reactivo del proceso; en particular, el método trata con una sola secuencia *w* de E/S formada las señales de entrada-salida de un sistema.

El método propuesto permite encontrar a partir de *w* una secuencia de eventos reducida llamada *e'*, a la cual se le asocia una función característica; estas funciones son procesadas posteriormente para su agrupación en eventos compuestos, los cuales definen las transiciones del modelo. Las transiciones y los eventos de salida permiten construir los fragmentos de RPI, los cuales al fusionarse a través de nodos comunes, conducen a las componentes observables del modelo.

En relación al trabajo previo presentado en [Estrada, 2013], el presente método es más simple el cual opera sobre una cantidad de eventos reducida; además los procesamientos subsecuentes son de complejidad lineal en la cantidad de elementos en la estructuras de datos utilizadas. En consecuencia el método es más eficiente.

Por otro lado, los eventos compuestos obtenidos expresan con mayor detalle el contexto de los eventos lo cual permite un análisis más detallado del comportamiento representado por el modelo final obtenido.

Con la implementación de los algoritmos derivados del método se facilitaron las pruebas con diversos casos de estudio, reales y simulados, procesando diferentes longitudes de w. La secuencia de transiciones S producida se procesó con el software que implementa el método de obtención de la parte no observable propuesto en [Tapia, 2012] con el fin de completar el modelo final.

El presente trabajo contribuye modestamente a la minería de procesos de eventos discretos reactivos. Sin embargo, durante el desarrollo del presente trabajo, detectamos que existen aspectos que deben ser atendidos en la etapa que calcula la parte no observable a partir de la secuencia de transiciones S, los cuales constituyen temas abiertos de investigación. Estos problemas tiene que ver con comportamientos en los cuales ciertos eventos tienen que ser contabilizados y con aspectos de temporizaciones en eventos internos.

Referencias

[Aalst, 2011] W. van der. Aalst, "Process Mining: Discovery, Conformance and Enhancement of Business Processes", Springer-Verlag, Berlin, 2011.

[Angluin, 1987] D. Angluin, "Learning regular sets from queries and counterexamples", Information and Computation, Volume 75, Issue 2, pp. 87-106, 1987.

[Cabasino, 2009] M. P. Cabasino, "Diagnosis and identification of discrete event systems using Petri nets", Ph. D. Thesis, University of Cagliari, Mar. 2009.

[Dotoli, 2006a] M. Dotoli, M. P. Fanti, A. M. Mangini, "An optimization approach for identification of Petri nets", Proc. of the 8th Int. Workshop on Discrete Event Systems, Ann Arbor, Michigan, USA, Jul. 2006.

[Dotoli, 2006b] M. Dotoli, M. P. Fanti, A. M. Mangini, "On-line identification of discrete event systems: a case study", Proc. of the 2006 IEEE Int. Conf. on Automation Science and Engineering, Shangai, China, Oct. 2006.

[Dotoli, 2007] M. Dotoli, M. P. Fanti, A. M. Mangini, "On-line identification of discrete event systems via Petri nets: an Application to Monitor Specification", Proc. of the 3rd Annual IEEE Conf. on Automation Science and Engineering, Scottsdale, Arizona, USA, Sep. 2007.

[Dotoli, 2008] M. Dotoli, M. P. Fanti, A. M. Mangini, "Real time identification of discrete event systems using Petri nets", Automatica, 44(5), pp. 1209-1219, May 2008.

[Esparza, 1995] J. Esparza y J. Desel. "Free Choice Petri Nets" Cambridge University Press, 1995.

[Estrada, 2009] A. P. Estrada-Vargas, "Identification of Concurrent Discrete Event Systems from Input-Output Sequences", Master Thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Unidad Guadalajara, Aug. 2009.

[Estrada, 2010a] A.P. Estrada-Vargas, E. López-Mellado, J.J. Lesage. "A Comparative Analysis of Recent Identification Approaches for Discrete-Event Systems", Mathematical Problems in Engineering, 2010, Hindawi. Doi:10.1155/2010/453254

[Estrada, 2010a] A.P. Estrada-Vargas, E. López-Mellado, J.-J. Lesage. "A comparative analysis of recent identification approaches for discrete-event systems", Mathematical Problems in Engineering, 2010, Hindawi. Doi:10.1155/2010/453254

[Estrada, 2010b] A.P. Estrada-Vargas, E. Lopez-Mellado, J-J. Lesage. "An Identification Method for PLC-based Automated Discrete Event Systems". Proc. of the IEEE Int. Conf. on Decision and Control, pp.6740-6746, Atlanta, USA, Dec. 2010.

[Estrada, 2011] A. P. Estrada-Vargas, J.-J. Lesage, E. López-Mellado, "Stepwise Identification of Automated Discrete Manufacturing Systems", 16th IEEE Int. Conf. on Emerging Technologies and Factory Automation, Toulouse, France, Sep. 2011.

[Estrada, 2012] A. P. Estrada-Vargas, J.-J. Lesage, E. López-Mellado, "Identification of Industrial Automation Systems: Building Compact and Expressive Petri Net Models from Observable Behavior", Proc. of American Control Conference, pp. 6095 - 6101, Montréal (Canada), Jun. 2012.

[Estrada, 2013] A. P. Estrada-Vargas, J.-J. Lesage, E. López-Mellado, "¬¬¬Identification of Partially Observable Discrete Event Manufacturing Systems", ETFA 2013, Calgliary (Italy), Jun. 2012. [Aguilar, 2007] J. Aguilar, "Identification of discrete-event dynamic systems based on the evolutionary programming", International Journal of Knowledge-based and Intelligent Engineering Systems, pp. 43-53, Amsterdam, The Netherlands, Ene. 2011.

[Gold, 1967] E.M. Gold, "Language identification in the limit", Information and Control, 10, pp. 447-474, 1967.

[Guia, 2005] A. Giua and C. Seatzu, "Identification of free-labeled Petri nets via integer programming", Proc. of the 44th IEEE Conf. on Decision and Control, Seville, Spain, Dec. 2005

[Hiraishi, 1992] K. Hiraishi, "Construction of safe Petri nets by presenting firing sequences", Lectures Notes in Computer Sciences, 616, pp. 244-262, 1992.

[Klein, 2005a] S. Klein, L. Litz,J.-J. Lesage, "Fault detection of Discrete Event Systems using an identification approach", 16th IFAC World Congress, CDROM paper n°02643, 6 pages, Praha(CZ), Jul. 2005.

[Klein, 2005b] S. Klein, "Identification of Discrete Event Systems for Fault Detection Purposes", Ph. D. Thesis, Ecole Normale Supérieure de Chachan, Oct. 2005.

[Meda, 1998] M.E. Meda, "DES identification using Interpreted Petri nets", Int. Symposium on Robotics and Automation, pp. 353-357, Saltillo, Mexico, Dec. 1998.

[Meda, 2000] M. Meda, A. Ramírez, E. López "Asymptotic Identification for DES", IEEE Conf. on Decision and Control, Sydney, Australia, pp. 2266-2271, Dec. 2000.

[Meda, 2000] M. Meda, A. Ramírez, E. López, "Behavioral properties for the control of discrete event systems modeled by interpreted Petri nets", IEEE Int. Conf. on Systems, Man, and Cybernetics, Nashville, USA, pp. 2150-2155, Oct. 2000.

[Meda, 2001] M. Meda, E. López, "A passive method for on-line identification of discrete event systems", IEEE Int. Conf. on Decision and Control, Orlando, Florida, USA pp. 4990-4995, Dec. 2001.

[Meda, 2002a] M. E. Meda-Campaña, "On-line Identification of Discrete Event Systems: Fundamentals and Algorithms for the Synthesis of Petri Net Models", Ph. D. Thesis, Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, Unidad Guadalajara, Nov. 2002.

[Meda, 2002b] M. Meda, E. López, "Incremental Synthesis of Petri Net Models for Identification of Discrete Event Systems", IEEE Conf. on Decision and Control, Las Vegas, USA, pp.805-810, Dec. 2002.

[Meda, 2003] M. Meda, E. López, "Required Transition Sequences for DES identification", IEEE Conf. on Decision and Control, Maui, Hawaii USA pp. 3778-3782, Dec. 2003.

[Meda, 2005] M. Meda, E. López, "Identification of concurrent Discrete Event Systems Using Petri Nets", IMACS 2005 World Congress, Paris, France, pp. 1-7, Jul. 2005.

[Murata, 1989] T. Murata. "Petri Nets: Properties, Analysis and Applications." IEEE, Vol. 77, No. 4, pp. 541-580, 1989.

[Roth, 2009] M. Roth, J.-J. Lesage, L. Litz, "An FDI Method for Manufacturing Systems Based on an Identified Model", Proc. of IFAC Symposium on Information Control Problems in Manufacturing (INCOM 2009), Moscow, Russia, pp. 1389-1394, June 2009

[Roth, 2010a] M. Roth, "Identification and Fault Diagnosis of Industrial Closed-Loop Discrete Event Systems", Ph. D. Thesis, Technische Universität Kaiserslautern, Ecole Normale Supérieur de Cachan, Oct. 2010.

[Roth, 2010b] M. Roth, J.-J. Lesage, L. Litz, "Black-box identification of discrete event systems with optimal partitioning of concurrent subsystems", Proc. of the 2010 American Control Conf., Baltimore, USA, pp. 2601-2606, Jun.-Jul. 2010.

[Roth, 2010c] M. Roth, J.-J. Lesage, L. Litz, "Identification of Discrete Event Systems, implementation issues and model completeness", 7th Int. Conf. on Informatics in Control Automation and Robotics, Funchal, Portugal, pp. 73-80, Jun. 2010

[Roth, 2011] M. Roth, J.-J. Lesage, L. Litz, "The concept of residuals for fault localization in discrete event systems", Control Engineering Practice, 19(9), pp. 978-988.

[Roth, 2012] M. Roth, S. Schneider, J.-J. Lesage, L. Litz, "Fault detection and isolation in manufacturing systems with an identified discrete event model", Int. Journal of Systems Science, 43(10), pp. 1826-1841. Doi:10.1080/00207721.2011.649369.

[Silva, 1985] M.Silva, "Las Redes de Petri: en la Automática y la Informática". Editorial AC. 1985.

[Takada, 1998] Y. Takada, "Grammatical inference for even linear languages based on control sets", Information Processing Letters, 28, pp. 193-199, 1998.

[Tapia,2014] T. Tapia, E. López-Mellado, A.P. Estrada-Vargas, J. J. Lesage, "Petri Net Discovery of Discrete Event Processes by Computing T-invariants", 19th IEEE Int. Conf. on Emerging Technologies and Factory Automation (ETFA'14), Sep 2014, Barcelone, Spain. paper 345, 8 p

[Veelenturf, 1978] L.P.J. Veelenturf, "Inference of sequential machines from sample computations", IEEE Trans. on Computers, 27, pp. 167-170, 1978.

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

MINERIA DE PROCESOS DE EVENTOS DISCRETOS. SÍNTESIS DE LAS COMPONENTES OBSERVABLES DE REDES DE PETRI INTERPRETADAS

del (la) C.

Edelma RODRÍGUEZ PÉREZ

el día 17 de Diciembre de 2014.

Dr. Luis Ernesto López Mellado
Investigador CINVESTAV 3C
CINVESTAV Unidad Guadalajara

Dr. Félix Francisco Ramos Corchado
Investigador CINVESTAV 3B
CINVESTAV Unidad Guadalajara

Dr. Antonio Ramírez Treviño
Investigador CINVESTAV 3A
CINVESTAV Unidad Guadalajara