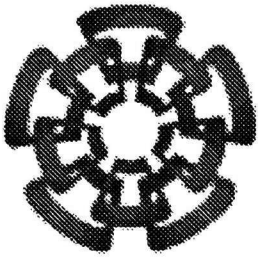
DIB - 15914



CINVESTAV

Centro de Investigación y de Estudios Avanzados del IPN
Unidad Guadalajara de Ingeniería Avanzada

Excelencia en Investigación, Educación y Desarrollo Tecnológico

Laboratorio de Ingeniería Eléctrica y Ciencias de la Computación-Guadalajara

Identificación de Sistemas Dinámicos de Eventos Discretos utilizando Redes de Petri

Tesis que presenta
María Elena Meda Campaña

Para obtener el grado de
Maestro en Ciencias

En la especialidad de
Ingeniería Eléctrica

Guadalajara, Jal., Septiembre de 1998

Identificación de Sistemas Dinámicos de Eventos Discretos

Tesis de Maestría en Ciencias
Ingeniería Eléctrica

Por:

María Elena Meda Campaña

Ingeniero Industrial
Instituto Tecnológico de Culiacán 1991-1995

Becario del CONACyT, expediente no. 112941

Director de Tesis
Dr. Antonio Ramírez Treviño

CINVESTAV del IPN Unidad Guadalajara, Septiembre de 1998

Índice General

Índice General

Capítulo 1

Panorama General

Resumen: Para analizar y diseñar las propiedades de un sistema es importante contar con un modelo del mismo. Sin embargo la elaboración de dicho modelo puede llegar a ser una tarea difícil. Diferentes son las causas que originan esta dificultad, entre ellas se tiene que no se conozca el fenómeno físico y/o los parámetros del sistema. Es en este último punto donde la identificación llega a ser útil, ya que nos permite construir un modelo del sistema a partir de la medición de sus entradas y sus salidas. También la identificación resulta útil cuando el sistema es variante en el tiempo o cuando se desea hacer control adaptable.

En el ámbito de los sistemas de eventos discretos no existen muchos trabajos reportados sobre el problema de identificación, sobre todo cuando se trata de identificación en línea (en tiempo real). Este capítulo introduce la noción de identificación de sistemas de eventos discretos y presenta algunos de los métodos de identificación más importantes tanto paramétricos como estructurales.

En el último apartado se define el problema de identificación que se tratará de resolver a lo largo de la tesis.

Figura 1.1: Caja negra

1.1 Introducción

Imaginemos que existe una caja negra como la de la figura 1.1, que procesa los datos de entrada x y produce cierto resultado y. Supongamos que cuando su entrada es $x = 2$ su salida es $y = 4$. Si nos preguntamos qué es lo que ocurrió dentro de la caja negra para que se produjera ese resultado, algunos diríamos que $y = 2x$, otros que $y = x^2$, algunos más que $y = x^x$ ó que $y = 2^x$. Como vemos con la información que se tiene no se puede daterminar cuál es el proceso que la caja negra representa, para ello necesitamos más pares entrada-salida (x, y). Si ahora la entrada es $x = 0.5$ y la salida es $y = 0.25$, se puede suponer que la caja negra representa al sistema $y = x^2$.

Pongamos ahora el ejemplo de un sistema cuyas entradas y salidas a la caja negra son discretas. Al igual que en el caso anterior con la información de un sólo par entrada-salida no es suficiente para determinar qué relación representa la caja negra. Si se presenta la entrada $x = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$ y se produce la salida $y = 0$, se pensaría que se trata de una compuerta lógica **or** o de una **and**, si se presenta la entrada $x = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$ y se produce la salida $y = 1$, aún no se podría determinar de qué tipo de compuerta lógica se trata, se necesitan más pares entrada salida. Cuando se presentan los pares $x = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$, $y = 0$ y $x = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$, $y = 0$ entonces se puede afirmar con seguridad que la caja negra representa a la compuerta lógica **and**.

Dada la naturaleza de las entradas y salidas, la caja negra del primer ejemplo representa un sistema continuo y la del segundo un sistema discreto.

Con estos ejemplos se introduce intuitivamente el problema de identificación. Identificar un sistema consiste en encontrar un modelo que represente el comportamiento del mismo a partir de las mediciones de sus entradas y salidas que por lo general; es la información que se tiene disponible de un sistema.

Para analizar un sistema se cuentan con varios métodos: a) métodos analíticos, b) métodos funcionales c) simulación [4]. Para analizar sistemas usando métodos analíticos es necesario contar con una descripción formal (modelo) de la estructura del sistema, por ejemplo variables de estado, Armax (autoregresivo, media móvil, exógeno), relación entrada-salida. Para analizar sistemas usando métodos funcionales, es necesario contar con una descripción del comportamiento del sistema (modelo funcional). Por último, los métodos de simulación se basan, en la simulación del sistema, utilizando un modelo de simulación. Como se observa, para realizar cualquier tipo de análisis sobre el sistema, es necesario contar con algún tipo de modelo.

La tarea de modelado puede llegar a ser un problema, ya sea porque se conoce el proceso físico, pero no las constantes que lo afectan, o porque ni siquiera se conoce el fenómeno físico.

Para evitarse la construcción de un modelo se puede utilizar la identificación del sistema o utilizar esquemas libres de modelo como las redes Neuronales. Cuando se identifica un sistema se encuentran los parámetros o la estructura de éste. Si lo que se desea es controlar el sistema, tanto la identificación como los esquemas libres de modelo se pueden utilizar.

En este trabajo se presenta un estudio del problema de identificación en sistemas de eventos discretos, antes de abordar este problema, se presenta una clasificación de los sistemas desde el punto de vista de las variables de estado que los describen [16] y [15].

Definición 1.1 *Sistemas dinámicos de variables continuas. Son sistemas cuyo estado evoluciona de manera continua. Las ecuaciones diferenciales son adecuados para modelarlo. El estudio de identificación de estos sistemas está a cargo de la teoría de estimación (ya sea paramétrica o estructural), utilizando métodos como mínimos cuadrados (recursivos o generalizados) o filtro de Kalman entre otros. Utilizaremos las siglas SVC para nombrarlos.*

Definición 1.2 *Sistemas dinámicos de eventos discretos. Son sistemas cuyo espacio de estados es numerable. Las ecuaciones diferenciales no son adecuadas para modelarlos, por lo que generalmente se utilizan los lenguajes formales, autómatas finitios, lógica temporal o redes de Petri, entre otros. Utilizaremos las siglas SED para nombrarlos.*

En las últimas décadas, los SED´s han despertado un gran interés por su estudio, debido a que muchos de los sistemas físicos pertenecen a esta categoría por lo menos en algún nivel de descripción. En esta clase se pueden encontrar los sistemas de manufactura, los sistemas de computación y comunicaciones, entre otros. Para su análisis también es necesario contar con un modelo, es por ello la importancia de identificar este tipo de sistemas

1.2 Revisión de la literatura

Existen dos tipos de identificación dependiendo de la información que se tenga disponible del sistema: a) la paramétrica, que requiere conocer la estructura del sistema, y b) la estructural, que requiere conocer qué tipo de estructura se va a identificar, ya sea una representación en función de transferencia, en variables de estados, etc. Ambos tipos de identificación requieren información de las entradas y salidas del sistema, y obtienen como resultado un modelo del mismo.

A continuación se presentan algunos de los métodos más importantes de identificación de sistemas dinámicos.

1.2.1 Identificación paramétrica

En este punto, se presentan algunos métodos de estimación de parámetros. Los estimadores son considerados a partir de un punto de vista estadístico. Existe una estrecha relación entre lo que se conoce inicialmente del proceso y el tipo de procedimiento óptimo a utilizar para estimar los parámetros. El tipo de estimador se elige dependiendo de la información que se tenga disponible del sistema. Para el estimador de los mínimos cuadrados, se asume únicamente que la dinámica del proceso puede ser aproximada por el conocimiento de las entradas y salidas. El estimador de Markov requiere también del conocimiento de la matriz de covarianza del ruido. Para el estimador de máxima verosimilitud, se necesita conocer la

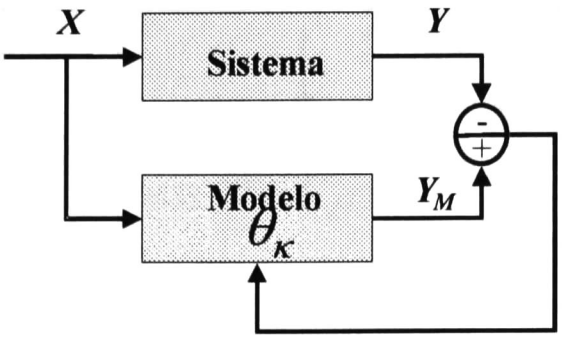


Figura 1.2:

función de densidad de probabilidad del proceso estocástico de donde se toman los valores de las muestras. El procedimiento del mínimo costo o del menor riesgo de Bayes, requiere un conocimiento a priori de las funciones de densidad de los parámetros desconocidos, y los costos de cometer errores.

A continuación se presentan algunos métodos de identificación paramétrica,

Mínimos cuadrados

El algoritmo de los mínimos cuadrados no recursivos requiere conocer la información de las entradas y las salidas del sistema. Las entradas y las salidas se pueden representar en forma vectorial: X representa las entradas, Y las salidas del sistema y Y_M las salidas del modelo, donde $Y_M = X\theta$, θ es el vector de parámetros que se quiere identificar (ver figura 1.2). En este caso, la dimensión de θ es conocida y sólo se obtendrá su valor. Al minimizar el criterio de error $E = e^T e$ (error cuadrado), donde $e = (Y - Y_M)$ se obtiene el esrtimador de mínimos cuadrados

$$\hat{\theta} = (X^T X)^{-1} X^T Y \tag{1.1}$$

Mínimos cuadrados generalizados o estimador de Markov

El algoritmo de mínimos cuadrados generalizados es muy similar al método anterior, sólo que para poder utilizar este método también se necesita tener información del ruido asociado al sistema (ver figura 1.3), esta información se representa en la matriz de covarianza del ruido R:

$$R = E[nn^T] = \begin{bmatrix} E[n(1)n(1)] & \cdots & E[n(1)n(k)] \\ \vdots & & \vdots \\ E[n(k)n(1)] & \cdots & E[n(k)n(k)] \end{bmatrix} \tag{1.2}$$

donde n es el ruido y E es el valor esperado de nn^T

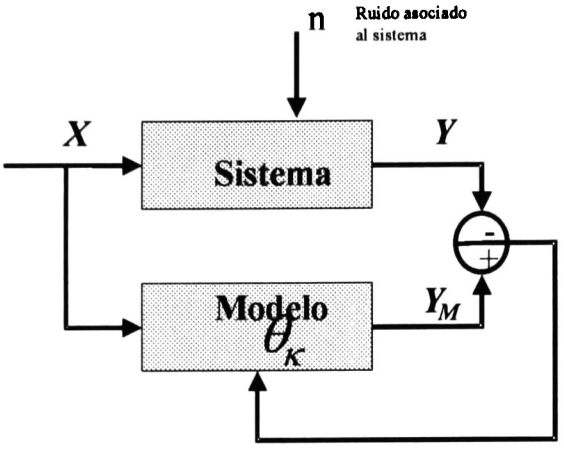


Figura 1.3: Sistema con ruido asociado (n)

En este método el criterio de error a minimizar es $E = e^T R^{-1} e$, donde e es el criterio de error definido en el método anterior y dado que ahora se tiene más información, el estimador que se encuentra es mejor; es decir, que con este estimador se obtiene una mejor aproximación del vector de parámetros del sistema. Después de minimizar el criterio de error el estimador que se obtiene es el siguiente:

$$\hat{\theta} = [X^T N^{-1} X]^{-1} X^T N^{-1} Y \tag{1.3}$$

A este método se le llama mínimos cuadrados generalizados ya que en el caso de que la matriz de covarianza N sea la matriz identidad I, entonces el criterio de error es $E = e^T e$ el cual es el error cuadrado que se minimiza en el método anterior.

Máxima verosimilitud

Este método considera una clase de problemas más generales que los considerados por los mínimos cuadrados. En el caso de que se trate de un modelo lineal con perturbaciones Gaussianas, el estimador de máxima verosimilitud se reduce al estimador de los mínimos cuadrados. Este método asume que el mecanismo generador de las salidas del proceso puede ser representado por un miembro de una familia paramétrica de distribuciones de probabilidad $\rho = \{p(\cdot \mid \theta) : \theta \in \Theta\}$, donde la función $p(\cdot \mid \theta)$ se conoce como la función de máxima verosimilitud, θ es un parámetro y $\Theta \subset R^p$. Dado un vector de salidas Y se dice que la función $p(Y \mid \theta)$ es una función de θ sobre Θ y entonces se puede referir a $p(Y \mid \cdot)$ como la función de máxima verosimilitud sobre Θ.

El estimador de máxima verosimilitud es una función $\hat{\theta}(Y)$ tal que:

$$p(Y \mid \hat{\theta}) = \sup_{\theta \in \Theta} p(Y \mid \theta) \tag{1.4}$$

donde $p(Y \mid \theta)$ es la función de máxima verosimilitud para los parámetros. En sí este método se basa en escoger un miembro de la familia de funciones de distribución de tal forma que

éste haga que el vector dado Y sea el más probable en el sentido de que la función $p(Y \mid \theta)$ se maximice.

1.2.2 Identificación estructural

Mínimos cuadrados repetidos

Este método consiste en incrementar el orden del modelo en corridas sucesivas del proceso de identificación. La correspondencia del modelo con el sistema se basa en el error residual $e = Y - X\theta$. Si al aumentar el orden del modelo no se obtiene una reducción significativa del error residual entonces la estimación se detiene ya que no se minimiza más el error al seguir aumentando el orden del modelo. No existe ninguna prueba formal de la validez de este método.

1.3 Trabajos en sistemas dinámicos de eventos discretos

Se presentan a continuación algunos algoritmos de identificación de SED's. Todos los algortimos que se presentan identifican la estructura de un autómata finito (AF) con ciertas características, por ejemplo, el algoritmo de Dana Angluin [[3]] identifica un AF determinista (AFD) y 0-reversible, mientras que el algoritmo de Hiraishi [[7]] identifica una red de Petri determinista I-reversible. Por su parte el algoritmo de Ullman [[5]] identifica un AF no determinista (AFN) con transiciones-ϵ. Como entrada a los dos primeros algoritmos se requiere un árbol de prefijos que represente a la muestra positiva del lenguaje del sistema a identificar. Mientras que el algoritmo de Ullman requiere que el lenguaje del autómata a identificar se represente con una expresión regular.

Identificación de un AF cero reversible (ZR)

Dana Angluin en [3] presenta un algoritmo de identificación el cual a partir de una muestra positiva[1] del lenguaje del sistema a identificar, genera como salida un autómata determinista 0-reversible [6] El autómata obtenido genera como mínimo la muestra positiva; es decir, que el lenguaje de la muestra es un subconjunto del lenguaje del autómata identificado.

El algoritmo funciona de la siguiente manera: con la muestra positiva del lenguaje se construye un árbol de prefijos. El método de identificación consiste en ir formando bloques con los estados del árbol. El primer bloque que se forma es el que contiene a todos los estados finales. Todos los estados que entren (predecesores) a este bloque con el mismo símbolo formarán un nuevo bloque, lo mismo pasa para los estados que salen (sucesores) del bloque con el mismo símbolo, hasta que se obtenga un AFD ZR. Por la forma de construcción de los bloques, el autómata encontrado es ZR debido a que el estado sólo tiene un predecesor con el mismo símbolo de entrada y es determinista porque un estado sólo tiene un estado sucesor con el mismo símbolo de salida.

[1] Una muestra positiva es un subconjunto de palabras del lenguaje que puede generar un autómata.

Identificación de una red de Petri I-reversible

Al igual que el algoritmo de Dana Angluin, Hiraishi [7] presenta un algoritmo que identifica un autómata a partir de la muestra positiva de un lenguaje, con la diferencia que ahora éste es I-reversible. Hiraishi presenta un algoritmo para representar el autómata identificado en una red de Petri de clase A (subconjunto de las redes de libre elección).

El algoritmo que identifica al autómata también parte del árbol de prefijos de la muestra positiva del lenguaje, este algoritmo difiere del anterior en la manera de formar los bloques. Hiraishi no se basa en los sucesores y predecesores de cada bloque para ir formando nuevos bloques, sino que del árbol de prefijos toma cualquier estado final s_f, calcula los vectores de Parikh Vp_i de las secuencias σ_i asociadas a cada estado final (menos a s_f), se calcula la diferencia de todos los Vp_i con el vector de Parikh asociado a s_f y con los vectores de diferencia que sean linealmente independientes se forma una base de vectores. Después se hace una lista de pares de estados (s1, s2), tal que el vector que resulta de la diferencia de los vectores de Parikh de las secuencias que alcanzan s_1 y s_2 sea linealmente dependiente de los vectores de la base (invariante consistente). Lo último por hacer es formar los bloques. Si el bloque de s1 B(s1) es diferente al bloque de s2 B(s2), entonces B(s1) y B(s2) forman otro bloque, así hasta terminar con la lista de pares; al inicio cada uno de los estados pertenece a un bloque distinto.

El algoritmo para pasar de un autómata a una red de Petri, recibe como entrada a un autómata I-reversible y al conjunto de pares de transiciones (x, y), tal que existe una secuencia σ donde σxy pertenece a los prefijos del lenguaje y σy no pertenece a los prefijos del lenguaje.

Un autómata I-reversible es aquel que sólo tiene un estado inicial y un solo estado final además de ser invariante consistente.

Identificación de un AFN con transiciones-ϵ

En [5] se presenta un algortimo de identificación de autómatas finitos no deterministas (AFN) con transiciones-ϵ con un sólo estado final y sin transiciones saliendo de éste. Este algoritmo requiere que el lenguaje sea regular. El algoritmo posee estructuras definidas de autómatas tanto para las expresiones regulares, como para las operaciones de unión, concatenación y cerradura de Kleene. Lo que hace el algoritmo es descomponer la expresión regular r en expresiones regulares más pequeñas, hasta que cada una de las expresiones se pueda representar por una estructura ya definida. Supóngase que $r = (0(1^*)) + 1$, entonces lo primero que hay que hacer es separar la expresión en expresiones más sencillas, por ejemplo $r = r1 + r2$ donde $r1 = 01^*$ y $r2 = 1$, así pues $r1 = r3r4$, donde $r3 = 0$ y $r4 = 1^*$, la descomposición de r continúa hasta que todos los compontes sean un elemento del alfabeto. Para encontrar el autómata final sólo hay que unir los autómatas que representan a las expresiones regulares mínimas con transiciones ϵ, estas transiciones son mudas y sólo sirven como puente para llegar a otro estado.

1.4 Definición del problema

El problema a resolver en esta tesis es encontrar un algoritmo asintótico de identificación estructural de SED's. El modelo que se quiere obtener es una red de Petri interpretada

binaria.

Como el algoritmo es asintótico, el modelo se va construyendo conforme el SED evoluciona. Este trabajo es una mejora a los algoritmos ya existentes ([[3]], [[7]], [[5]]) que presuponen conocer el lenguaje (o una parte de él) que es aceptado por el SED.

Las redes de Petri son utilizadas en este trabajo porque han demostrado ser una herramienta adecuada para representar SED's, ya que capturan la concurrencia, las relaciones causales, decisiones, sincronizaciones, exclusión mutua y otras características, además presentan un marco formal que permite hacer análisis. Una extensión de las RP´s son las redes de Petri Interpretadas (RPI)[2], éstas se utilizan para describir cómo los actuadores afectan el estado del sistema, y el estado provoca que los sensores cambien de valor.

Se tienen las siguientes definiciones:

Definición 1.3 *Sea S_f un SED; y $\mathcal{M} = \{RPI\}$ el conjunto de todas las RPI. El problema de identificación se define como, dada $f : \{S_f\} \times \mathcal{M} \longrightarrow \mathcal{R}^+$ una función que indica qué tanto se parecen los comportamientos de S_f y $RPI_j \in \mathcal{M}$ (entre más pequeño sea el valor de f, más se parecen el modelo y el sistema).*

1. *Encontrar un modelo $k \in \mathcal{M} \mid f(S_f, k\} = \min_{m \in \mathcal{M}} f(S_f, m)$*

En este trabajo se dice que el sistema y el modelo son iguales si ambos generan los mismos lenguajes[3], una forma de medir este parecido es la cardinalidad de la diferencia simétrica del lenguaje del sistema y del modelo, i.e. $f(S_f, m_i) = |L(S_f) - L(m_i) \cup L(m_i) - L(S_f)|$.

Encontrar la función f es un problema complejo computacionalmente hablando [6], por eso en esta tesis se propone ir copiando cada secuencia que se observa del sistema en el modelo, de esta forma, conforme el sistema evoluciona, el algoritmo obtiene más información y por tanto un mejor modelo, esto es lo que se llama *identificación asintótica.*

Formalmente, el problema que se abordará en esta tesis es el siguiente:

Definición 1.4 *Sea S_f un SED; y $\mathcal{M} = \{RPI\}$ el conjunto de todas las RPI. El problema de identificación asintótica se define como, dada $f : \{S_f\} \times \mathcal{M} \longrightarrow \mathcal{R}^+$ una función que indique que tanto se parecen los comportamientos de S_f y $RPI_j \in \mathcal{M}$ (entre más pequeño sea el valor de f, más se parecen el modelo y el sistema).*

1. *Encontrar una secuencia de modelos $m_0, m_1, ..., m_k$, donde $m_i \in \mathcal{M}$ tal que $f(S_f, m_i) > f(S_f, m_{i+1})$*[4]

1.5 Organización de la tesis

En el capítulo 2 se presentan a las redes de Petri (RP) como herramienta de descripción de SED´s, así como algunas de sus propiedades. En el capítulo 3 se presentan tres nuevos resultados: el primero es la definición de los conceptos de observabilidad y de controlabilidad

[2] Las Redes de Petri Interpretadas se presentarán en el capítulo dos de esta tesis.

[3] Para entender los lenguajes que generan los sistemas de eventos discretos, el lector debe consultar el capítulo dos o el apéndice 1.

[4] El problema de identificación asintótica no hace alusión a algún método de identificación (ya sea paramétrico o estructural), así que cualquiera de ellos se puede aplicar.

para RP interpretadas (RPI); el segundo son dos teoremas para determinar la observabilidad y controlabilidad de una RPI; y el tercero es un algoritmo para determinar cuándo una RPI es controlable. En el capítulo 4 se presenta un algoritmo asintótico de identificación para SED's modelados con RPI. Un ejemplo de identificación se presenta en el capítulo 5. Y por último en el capítulo 6 se presentan las conclusiones y el trabajo futuro.

Capítulo 2

Herramientas matemáticas

Resumen: Existen diferentes formalismos para modelar SED's, de entre ellos, las Redes de Petri (RP) se seleccionaron como herramienta de representación. Uno de los motivos para tomar esta decisión es que las RP tienen un poder de representación mayor que los autómatas finitos (permiten modelar lenguajes regulares y algunos sensitivos a contexto). Además de esta razón las RP permiten capturar las principales características de los SED's, y por ser una herramienta formal existe una teoría bien fundada para el análisis. Por último su representación gráfica las hace adecuadas como herramienta de comunicación.

En este capítulo se presentan las RP, sus propiedades y algunas técnicas de análisis así como los lenguajes que generan.

2.1 Introducción

Existen diferentes formalismos para modelar y/o analizar SED's, entre ellos se encuentran los autómatas finitos (AF) que describen en forma de grafo los posibles estados del sistema y cómo éste cambia de un estado a otro. Las tablas de estado que también describen los estados y los cambios de éstos, pero a diferencia de los AF la representación es en forma tabular. Las redes de colas que permiten modelar el arribo de objetos a un servidor y cómo cambia el estado dependiendo si el objeto es atendido o permanece en espera. El PERT-CPM que permite modelar cómo se van realizando las actividades del sistema. Las redes de Petri que permiten modelar las actividades y usos de los recursos del sistema, entre otros.

En este trabajo se utilizan las redes de Petri porque capturan las principales características de los SED's como son asincronismo, concurrencia, exclusión mutua, relaciones causales, decisiones, sincronización, indeterminismo entre otras.

Una red de Petri es una herramienta matemática para describir y analizar SED's. Una característica importante de las redes de Petri es que tienen una representación gráfica y una matemática lo cual hace amigable su utilización.

En este capítulo se presentan los conceptos y propiedades básicas de las redes de Petri, así como una clase de red que asocia el significado físico del sistema a la estructura de la red, estas redes son las redes de Petri interpretadas.

2.2 Redes de Petri (RP)

2.2.1 Estructura de una RP

Definición 2.1 *Una RP generalizada está representada por la cuádrupla* $RP = (L, T, E, S)$ *donde,*

$L = \{l_i, ..., l_n\}$ *es un conjunto finito de lugares,* $n \geq 0$

$T = \{t_1, ..., t_m\}$ *es un conjunto finito de transiciones,* $m \geq 0$

$L \cap T = \emptyset; L \cup T \neq \emptyset$ *L y T son conjuntos disjuntos*

$E : L \times T \rightarrow \mathcal{N}^1 \cup \{0\}$ *Es una función de incidencia previa la cual representa el peso de los arcos de entrada a las transiciones*

$S : L \times T \rightarrow \mathcal{N} \cup \{0\}$ *Es una función de incidencia posterior la cual representa el peso de los arcos de salida a las transiciones*

Una *RP* se puede ver como un grafo dirigido que contiene dos tipos de nodos: los *lugares* y las *transiciones*; los *arcos* unen nodos de diferente tipo (grafo bipartido). Gráficamente los lugares se representan por círculos, las transiciones por barras y los arcos por flechas. Existe un arco que va del lugar l_i a la transición t_j *ssi* $E(l_i, t_j) \neq 0$, de forma similar existe un arco que va de la transición t_k al lugar l_i *ssi* $S(l_i, t_k) \neq 0$. Cada arco se etiqueta con un número entero natural $E(l, t)$ ó $S(l, t)$, que se denomina *peso* del arco. En la figura 2.1 se representa gráficamente la RP siguiente.

- $L = \{l_1, l_2, l_3, l_4, l_5, l_6\}$
- $T = \{t_1, t_2, t_3, t_4, t_5\}$

[1] Es el conjunto de los números naturales, los cuales no incluyen al cero.

Representación gráfica

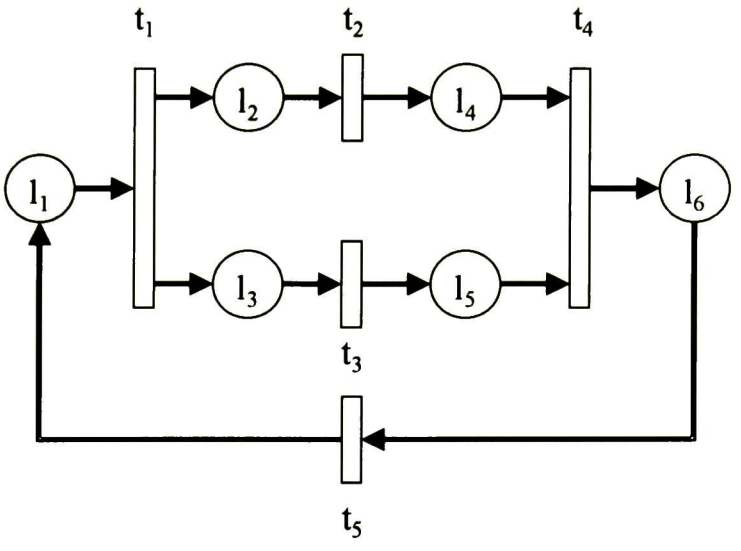


Figura 2.1: Una Red de Petri

- $E(l_1, t_1) = E(l_2, t_2) = E(l_3, t_3) = E(l_4, t_4) = E(l_5, t_4) = E(l_6, t_5) = 1$
- $S(l_2, t_1) = S(l_3, t_1) = S(l_4, t_2) = S(l_5, t_3) = S(l_6, t_4) = S(l_1, t_5) = 1$

El resto de los valores de las funciones de incidencia previa y posterior es cero.

Representación matricial

Las funciones E y S pueden representarse por las matrices $E[e_{ij}]$ y $S[s_{ij}]$, donde se satisface que $e_{ij} = E(l_i, t_j)$ y $s_{ij} = S(l_i, t_j)$. La representación matricial de la figura 2.1 es la siguiente:

$$E = \overset{\text{Matriz de entrada}}{\begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}} \qquad S = \overset{\text{Matriz de salida}}{\begin{bmatrix} 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}} \tag{2.1}$$

Definición 2.2 *Sea una red $RP = (L, T, E, S)$, $l \in L$ y $t \in T$. Se definen los siguientes conjuntos*

- *Conjunto de lugares de entrada a t:*

$$^\bullet t = \{l \in L \mid E(l, t) > 0\}$$

- *Conjunto de lugares de salida de t*

$$t^{\bullet} = \{l \in L \mid S(t,l) > 0\}$$

- *Conjunto de transiciones de entrada a l*

$${}^{\bullet}l = \{t \in T \mid S(t,l) > 0\}$$

- *Conjunto de transiciones de salida de l*

$$l^{\bullet} = \{t \in T \mid E(l,t) > 0\}$$

2.2.2 Marcado y evolución de la RP

Para representar el comportamiento dinámico de un sistema, se utilizan *marcas*, las cuales se representan como puntos dentro de los lugares. El *marcado M* de una *RP* es la distribución de marcas en la red (**o estado de la red**) en un instante, dado que ocurrió un evento y se expresa como una función vectorial del número de lugares, llamado vector de marcado representado por $M(l_i)$.

Definición 2.3 *El marcado de una red RP es una función de L en $\mathcal{N}$, es decir que el marcado asigna un número (marcas) a cada lugar.*

En la figura 2.1 $M(l_1) = 1$ y $M(l_2) = M(l_3) = M(l_4) = M(l_5) = M(l_6) = 0$, el marcado de una RP se puede representar en forma matricial, así el marcado anterior se puede ver como:

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^T \tag{2.2}$$

este marcado se conoce como *marcado inicial* de la RP.

Definición 2.4 *Una RP marcada se define como $RPM = (RP, M_0)$. Donde RP es una RP generalizada y M_0 es su marcado inicial.*

La RP de la figura 2.1 es una RP marcada.

Definición 2.5 *Una transición $t \in T$ está habilitada por el marcado M ssi cada uno de sus lugares de entrada contiene al menos $E(l,t)$ marcas.*

$$\forall l \in {}^{\bullet}t, M(l) \geq E(l,t) \tag{2.3}$$

En la figura 2.1 la transición que se encuentra habilitada es la transición t_1.

Definición 2.6 *El disparo de una transición habilitada t consiste en quitar $E(l,t)$ marcas de cada uno de sus lugares de entrada y agregar $S(t,l)$ marcas a cada uno de sus lugares de salida.*

$M_i \xrightarrow{t} M_j$ significa que t está habilida por el marcado M_i y el disparo de t alcanza el marcado M_j.

En la figura 2.1 si t_1 se dispara, el marcado que se alcanza es $\begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 0 \end{bmatrix}^T$ se quitan las marcas de l_1 y se agregan a l_2 y l_3.

Definición 2.7 *Una secuencia de disparos σ aplicable a partir del marcado M_0 se representa por una secuencia de transiciones tal que el disparo de cada transición conduce a un marcado que habilita la transición que sigue en la secuencia.*

Si

$$M_0 \xrightarrow{t_i} M_1 \xrightarrow{t_j} M_2 \xrightarrow{t_k} \cdots \xrightarrow{t_p} M_q \tag{2.4}$$

entonces la secuencia $\sigma = t_i, t_j, t_k \cdots, t_p$ es aplicable a partir de M_0 y la expresión 2.4 se puede representar como $M_0 \xrightarrow{\sigma} M_q$.

Una secuencia de disparos aplicable a la RP de la figura 2.1 puede ser $\sigma_i = t_1, t_3, t_2, t_4, t_5$

Definición 2.8 *El lenguaje generado por una RP es el conjunto de secuencias disparables a partir de M_0.*

$$\mathcal{L}g(RP, M_0) = \{\sigma \mid M_0 \xrightarrow{\sigma} M\} \tag{2.5}$$

El lenguaje que genera la RP de la figura 2.1 es

$$\overline{(t_1(t_2 + t_3)t_4t_5)^*} \tag{2.6}$$

Definición 2.9 *El vector característico de una secuencia σ o vector de Parikh es un vector $\vec{\sigma} \in \mathcal{N}^m$ cuyo i-ésimo elemento es el número de veces que la transición i se dispara en la secuencia σ.*

El vector característico de la secuencia $\sigma_i = t_1t_2t_3t_4t_5t_1t_2$ es aplicable en la RP de la figura 2.1 es:

$$\vec{\sigma_i} = \begin{bmatrix} 2 & 2 & 1 & 1 & 1 \end{bmatrix}^T \tag{2.7}$$

Definición 2.10 *El conjunto de vectores característicos de las secuencias disparables de una RP es*

$$\vec{\mathcal{L}g}(RP, M_0) = \{\vec{\sigma} \mid M_0 \xrightarrow{\sigma} M\} \tag{2.8}$$

Definición 2.11 *Un marcado M es alcanzable a partir de un marcado inicial M_0 ssi existe una secuencia de disparos aplicable a partir de M_0 y que lleva al marcado M, es decir $M_0 \xrightarrow{\sigma} M$.*

El marcado $\begin{bmatrix} 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix}^T$ no es alcanzable en la RP de la figura 2.1 porque no existe una secuencia σ que lleve a este marcado a partir del marcado inicial. En cambio el marcado $\begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}^T$ si es alcanzable a partir del marcado inicial con la secuencia $\sigma = t_1t_2$.

Definición 2.12 *El conjunto de marcados alcanzables a partir de M_0 se denota como R(RP, M_0) y se define de la siguiente forma::*

$$R(RP, M_0) = \{M \mid (\exists \sigma \in \mathcal{L}g(RP, M_0)) \wedge (M_0 \xrightarrow{\sigma} M)\} \tag{2.9}$$

2.2.3 Ecuación de estado de una RP

El disparo de una transición t origina un cambio en el marcado de la red, este cambio se calcula con la siguiente expresión

$$M_{k+1} = M_k + Cv_k \tag{2.10}$$

A la expresión 2.10 se le conoce como *ecuación de estado* de una RP, donde $C = S - E$ (nótese que C es de dimensión $n \times m$) es llamada *matriz de incidencia* y representa la estructura del sistema. v_k se conoce con el nombre de *vector de disparos*, y se define como $v_k = \vec{\sigma}$ tal que $M_k \xrightarrow{\sigma} M_{k+1}$. En este caso $\sigma = t$. σ se puede extender a una secuencia de transiciones.

La matriz C de la figura 2.1 es $C = \begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}$, si a partir del marcado inicial de la RP se aplica la secuencia $\sigma = t_1 t_2$, utilizando la ecuación de estado se obtiene el marcado M_{k+1} el cual se calcula de la siguiente forma:

$$\overset{M_{k+1}}{\begin{bmatrix} 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix}} = \overset{M_0}{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}} + \overset{C}{\begin{bmatrix} -1 & 0 & 0 & 0 & 1 \\ 1 & -1 & 0 & 0 & 0 \\ 1 & 0 & -1 & 0 & 0 \\ 0 & 1 & 0 & -1 & 0 \\ 0 & 0 & 1 & -1 & 0 \\ 0 & 0 & 0 & 1 & -1 \end{bmatrix}} \overset{v_k}{\begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}}$$

2.3 Propiedades de las RP

Dos de las propiedades características deseables en todos los sistemas son la ausencia de bloqueos y, la finitud del conjunto de estados en que el sistema se puede encontrar. Si se llevan estas propiedades a la terminología de RP, la ausencia de bloqueos se traduce en propiedades de vivacidad y la finitud del conjunto en propiedades de acotamiento. Ver [4].

En esta sección se presentan algunas de las propiedades de las RP y las definiciones de p y t-semiflujos.

2.3.1 P-semiflujos

La letra p de p-semiflujos viene de la palabra en inglés "place"que significa lugar. Los p-semiflujos son todos los vectores $Y^T \geq 0$ (anuladores izquierdos de la matriz C) que cumplen con la siguiente ecuación,

$$Y^T C = 0 \tag{2.11}$$

donde Y^T es un vector n dimensional ($\mid L \mid = n$) diferente de cero. Si la matriz C es de rango pleno por filas, entonces la RP no tiene p-semiflujos. El lugar l_i pertenece al p-semiflujo Y_j^T si $Y_j^T(i) = 1$.

Combinando la ecuación 2.11 en la ecuación 2.10, se obtiene:

$$Y^T M_{k+1} = Y^T M_0$$

Así que un p-semiflujo es un conjunto de lugares en el cual el número de marcas se conserva, por lo que un p-semiflujo posee la propiedad de conservatividad.

2.3.2 T-semiflujos

Un t-semiflujo es cualquier vector $X \geq 0$, anulador derecho de la matriz C, que satisface la siguiente ecuación:

$$CX = 0 \tag{2.12}$$

Un t-semiflujo se dice que tiene la característica de ser repetitivo, debido a que se puede construir una secuencia de transiciones a partir de su valor, que su disparo lleva al marcado inicial, como se observa a continuación. Una transición t_i pertenece al t-semiflujo X_j si $X_j(i) = 1$. X es m dimensional.

Combinando la ecuación 2.12 en la ecuación 2.10 y haciendo $X = v_k$ se obtiene:

$$M_{k+1} = M_0$$

(por lo que la red regresa a su estado inicial).

Estos conceptos son importantes porque nos permiten caracterizar de una manera estructural algunas propiedades de las RP.

2.3.3 Propiedades

Vivacidad

Definición 2.13 *Una transición t es viva para un marcado inicial dado M_0 ssi existe una secuencia de disparos σ a partir de cualquier marcado M, sucesor de M_0, que comprenda a t :*

$$\forall M \in R(RP, M_0)\ \exists\ \sigma : M \xrightarrow{\sigma} M' \mid t \in \sigma$$

Definición 2.14 *Una RP marcada es viva para M_0 ssi todas sus transiciones son vivas para M_0.*

La propiedad de vivacidad se utiliza para caracterizar el bloqueo de un sistema.

Si una *RP* es viva entonces el sistema que modela no se bloquea debido a que todas sus transiciones pueden llegar a disparse. El caso contrario no es cierto, porque existen redes que no son vivas y no se bloquean, estas redes se denominan *parcialmente vivas,* ejemplo de estas redes son las RP's de las figuras 2.2 (b) y 2.2 (c).

Definición 2.15 *Una RP es estructuralmente viva si existe un marcado M_0 finito para el cual la RP marcada es viva.*

La vivacidad estructural es una condición necesaria pero no suficiente para la vivacidad.

Las RP's de la figuras 2.2 (a) y 2.2 (b) son estructuralmente vivas

Acotamiento

Definición 2.16 *Un lugar l es k-limitado para M_0 ssi existe un número entero k tal que $M(l) \leq k$ para cualquier marcado $M \in R(RP, M_0)$. Al menor entero k que cumple con la desigualdad anterior se le denomina límite del lugar l.*

Definición 2.17 *Una RP marcada es k-acotada para M_0 ssi todos los lugares son k-acotados para M_0:*

$$\forall l \in L \ y \ \forall M \in R(RP, M_0), M(l) \leq k$$

La propiedad de acotamiento determina la finitud del número de estados del sistema representado por una *RP*.

Definición 2.18 *Una RP es estructuralmente acotada si es acotada para cualquier marcado inicial y finito.*

El acotamiento estructural es una condición suficiente para el acotamiento. Las RP's de la figuras 2.2 (a) y 2.2 (b) son estructuralmente acotadas.

Del caso especial cuando una *RP* es 1-acotada para M_0 surge la siguiente definición.

Definición 2.19 *Una RP es binaria si todos sus lugares son 1-acotados para cualquier marcado $M \in R(RP, M_0)$.*

Las RP's de las figuras 2.2 (a) y 2.2 (b) y son binarias.

Ejemplos de las propiedades de vivacidad y acotamiento se encuentran en la figura 2.2.

2.4 Redes de Petri Interpretadas (RPI)

Para que una red de Petri pueda representar un sistema físico es necesario asociarle una interpretación. Interpretar una red de Petri es

- Asociar un significado físico a las condiciones necesarias para el disparo de una transición.
- Establecer las acciones generadas por la evolución del marcado.

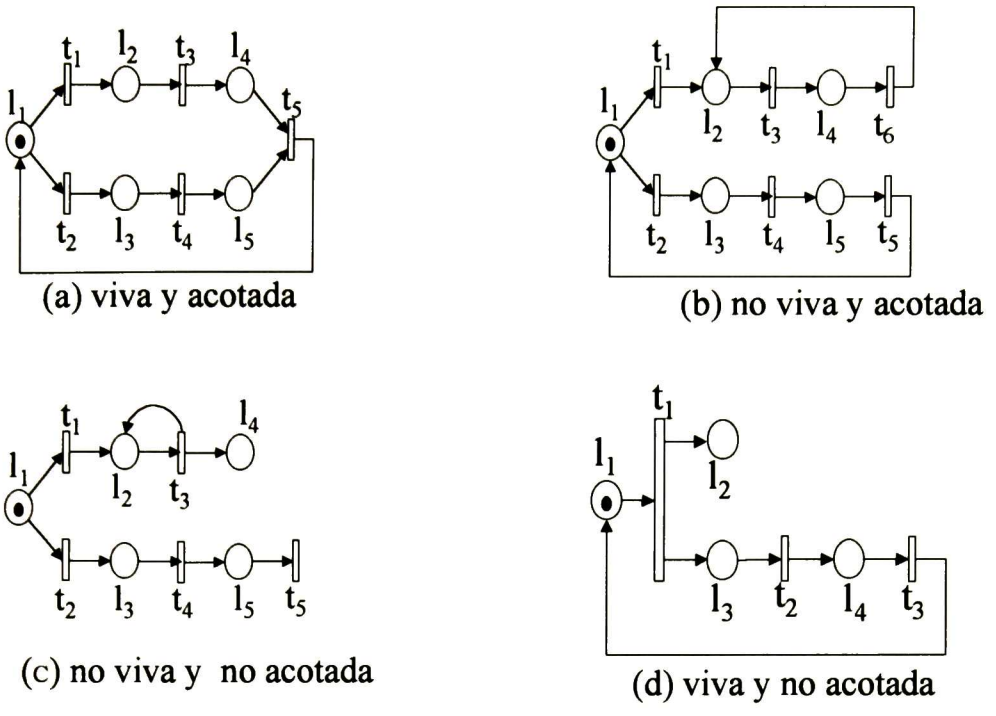


Figura 2.2: Redes de Petri con diferentes propiedades

Aunque las reglas de evolución se modifican debido a la interpretación, el estado interno del sistema se representa siempre por el marcado de la red de Petri. En la interpretación presentada en [4], las señales que hacen que la RPI se comunique con algún sistema se llaman acciones, y pueden generarse cuando se marca un lugar o cuando se dispara una transición, por tanto estas acciones se asocian tanto a las transiciones como a los lugares.

Sea un sistema con

$X = \{x_1, ..., x_e\}$ conjunto de variables entradas o señales que permiten incidir sobre la evolución del sistema. Son variables lógicas que funcionan como permisivas.

$Y = \{y_1, ..., y_s\}$ conjunto de salidas a nivel, es decir señales que mantienen su valor lógico mientras el sistema esté en un estado estable predeterminado apriori.

$Z = \{z_1, ..., z_q\}$ conjunto de salidas impulsionales, es decir señales de salida que sólo se presentan cuando el sistema cambia de un estado estable a otro.

Definición 2.20 *Una condición externa, C_i es un subconjunto de estados de las variables de entrada y puede ser representada por una función combinatoria*[2] *de las variables de entrada.*

Definición 2.21 *Un evento o acontecimiento, E_i, es el cambio de estado lógico de una condición externa o la unión de un conjunto de eventos.*

Los eventos, las condiciones externas y las salidas impulsionales se asocian a las transiciones, y las salidas a nivel a los lugares.

[2]Suma de algunos minitérminos formados con las variables de entrada.

La activación de una salida puede estar ligada a condiciones externas, en este caso se les llamará salidas condicionales.

Definición 2.22 *Una red de Petri interpretada se define por*

- *una RP marcada* (RP, M_0);
- *una función de T en E, que asocia a cada transición un evento,* E_i;
- *una función de T en* C_i*, que asocia a cada transición una condición externa,* C_i;
- *una función de T en Z, que a las transiciones les asocia salidas implusionales;*
- *una función de* $L \times C_i$ *en Y, que a los lugares les asocia salidas a nivel, eventualmente condicionadas.*

2.4.1 Evolución de una RP interpretada

1. El disparo de una transición habilitada sólo se realiza si se verifica $C_i \wedge E_i$. En estas condiciones se dice que t_i es receptiva a la condición y al evento $(C_i \wedge E_i)$.
2. Cuando una transición se dispara, se generan todas las salidas impulsionales asociadas a t_i.
3. Para un marcado dado, las salidas a nivel asociadas a los lugares marcados son generados si se verifican las condiciones externas que eventualmente les pueden estar asociadas. Si Y_j es una salida asociada a un lugar l_i condicionada por C_{ij}, se dirá que la salida es sensible a C_{ij} cuando l_i está marcado.

2.4.2 Ejemplo de modelado con RPI de un sistema físico

Dos carros A y B transportan cierto material desde los puntos de carga C_A y C_B, respectivamente, hasta el punto de decarga D (figura 2.3 *a*).

Los diferentes movimientos, hacia la izquierda o hacia a la derecha, son controlados mediante las acciones i_A, i_B, d_A, d_B.

Si A está en C_A y el pulsador M_A está oprimido, comienza un ciclo $C_B - D - C_B$ con las siguientes características:

> espera eventual en E_A hasta que la zona común a los dos carros esté libre, con el fin de evitar colisiones;
>
> espera obligatoria en D de $T_A = 100$ s de duración.

El carro B tiene un funcionamiento similar (pulsador M_B, ciclo $C_B - D - C_B$ y espera en D de $T_B = 50$ s) pero, en caso de demanda simultánea de la vía común (recurso compartido), el carro B es prioritario (prioridad fija).

El recorrido $E_A - D$ $(E_B - D)$ se establece gracias al posicionamiento de un cambio de agujas controlado por la acción G (respectivamente $\bar{G}$). En lo sucesivo se admitirá que E_A (E_B) proporciona un 1 lógico si el eje delantero de A (B) está en la zona $E_A - D$ $(E_B - D)$. A continuación se presentan una serie de observaciones que ayudan a clarificar el modelo.

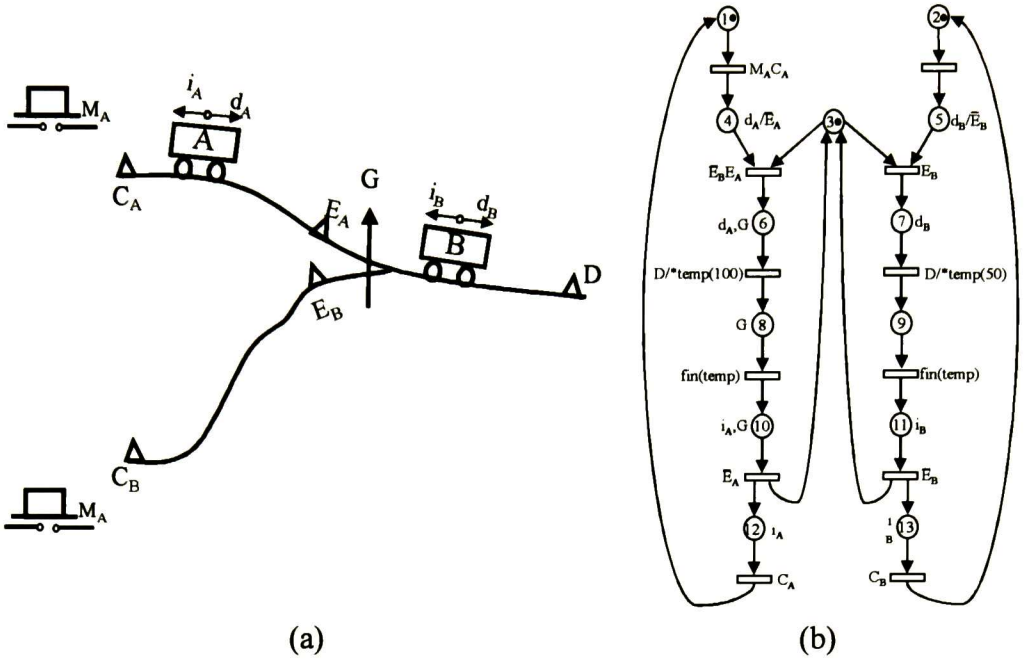


Figura 2.3: Esquema de la instalación y descripción de un sistema con recurso compartido. (a) Esquema de la instalación (b) Descripción funcional con RPI

El carro B tiene un funcionamiento similar (pulsador M_B, ciclo $C_B - D - C_B$ y espera en D de $T_B = 50$ s) pero, en caso de demanda simultánea de la vía común (recurso compartido), el carro B es prioritario (prioridad fija).

El recorrido $E_A - D$ ($E_B - D$) se establece gracias al posicionamiento de un cambio de agujas controlado por la acción G (respectivamente $\bar{G}$). En lo sucesivo se admitirá que E_A (E_B) proporciona un 1 lógico si el eje delantero de A (B) está en la zona $E_A - D$ ($E_B - D$). A continuación se presentan una serie de observaciones que ayudan a clarificar el modelo.

1. En la figura 2.3 *b* se presenta un posible modelo para el sistema.

2. Es importante considerar la construcción que modela el acceso al recurso. El lugar l_3 representa el recurso en un estado de reposo (no utilizado). Los lugares de l_6 a l_{11} representan la ocupación de un recurso común ($\{l_6, l_8, l_{10}\}$ ocupación por el carro A y $\{l_7, l_9, l_{11}\}$ ocupación por el carro B).

3. El conjunto de lugares $\{l_3, l_4, l_5\}$ plantean un conflicto (el acceso al recurso cuando la demanda es simultánea). La realización debe evitar la ambigüedad que se presentaría si tecnológicamente se pudiera tener $M(l_3), M(l_4), M(l_5), \bar{E}_B, E_A, E_B = 1$.

4. La transición t $l_6 \xrightarrow{t} l_8$ es receptiva a D. A ella se le ha asociado la acción impulsional *preseleccionar un temporizador con 100 unidades de tiempo*, $temp(100)$.

5. La acción d_A asociada al lugar l_4 está condicionada por $\bar{E}_A$.

2.5 Aplicación de las RP

Las RP se utilizan para modelar y analizar SED's. Las RP pueden utilizarse en distintos campos de aplicación. Para modelar un SED que pertenece a una determinada aplicación, se tiene que dotar de interpretación, por ello se definieron las RPI. Los principales campos de aplicación de las RPI son: los sistemas operativos y descripción de software en general, la descripción de hardware de computadores y sistemas discretos de control con evoluciones concurrentes, los automatismos lógicos, los lenguajes formales, entre otros. Ver [4].

2.6 Las RP como generadoras de Lenguajes

Una RP etiquetada (o una RP generadora) es una cuádrupla $G = (RP, \ell, M_0, M_f)$ donde:

- $RP = (L, T, E, S)$ es la estructura de una RP, definida anteriormente;
- $\ell : T \longrightarrow \Sigma \cup \{\epsilon\}$ es una función de etiquetación que asigna a cada transición una etiqueta del alfabeto de eventos o asigna el símbolo de vacío como etiqueta;
- M_0 es el marcado inicial
- M_f es un conjunto finito de marcados finales.

La palabra *sistema* se utilizará para hacer referencia a una *RP generadora.*

2.6.1 Etiquetación

Existen tres tipos de funciones de etiquetación:

1. En una RP de *etiquetación-libre* todas las transiciones tienen etiquetas distintas y ninguna se etiqueta con el símbolo de vacío ϵ, es decir, $(\forall t, t' \in T)[t \neq t' \longrightarrow l(t) \neq (t')$ y $\forall t \in T\ [\ell(t) \neq \epsilon]$.
2. En una RP de *etiquetación libre de* ϵ ninguna transición se etiqueta con el símbolo de vacío ϵ.
3. En una RP de *etiquetación arbitraria* no existe restricción sobre ℓ.

2.6.2 Lenguajes generados

Cuatro lenguajes son asociados con G dependiendo de las diferentes nociones de las palabras terminales.

1. El *tipo L* o *lenguaje terminal* (lenguaje marcado en teoría de Control Supervisor) se define como el conjunto de palabras generadas secuencias de disparos que alcanzan un marcado final,

$$\mathcal{L}_L(G) = \{\ell(\sigma) | M_0 \xrightarrow{\sigma} m_f \in M_f\}$$

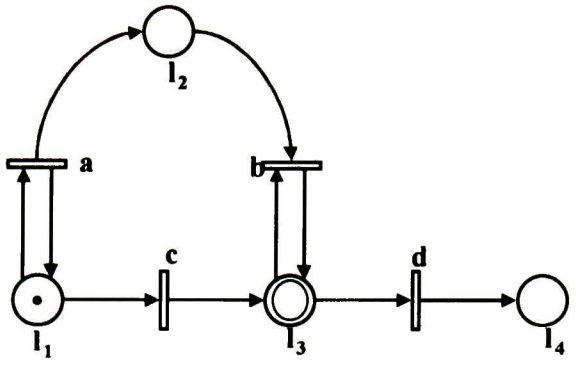


Figura 2.4: Red de Petri generadora con estiquetación libre

2. El *tipo G* o *lenguaje de cobertura* se define como el conjunto de palabras generadas por secuencias de disparos que alcanzan un marcado M que cubre un marcado final,

$$\mathcal{L}_G(G) = \{\ell(\sigma) | M_0 \xrightarrow{\sigma} M \geq mf \in Mf\}$$

3. El *tipo D* o *lenguaje de bloqueo* se define como el conjunto de palabras generadas por secuencias de disparos, que alcanzan un marcado en el cual ninguna transición se habilita,

$$\mathcal{L}_D(G) = \{\ell(\sigma) | M_0 \xrightarrow{\sigma} M \wedge (\forall t \in T) \neg (M \xrightarrow{t})\}$$

4. El *tipo P* o *lenguaje de prefijo* (lenguaje cerrado en teoría de Control Supervisor) se define como el conjunto de palabras generadas por cualquier secuencia de disparos,

$$\mathcal{L}_p(G) = \{\ell(\sigma) | M_0 \xrightarrow{\sigma}\}$$

Ejemplo 2.1 *El sistema G de la figura 2.4 es de etiquetación libre. El marcado inicial también presentado en esa figura, es* $M_0 = \begin{bmatrix} 1 & 0 & 0 & 0 \end{bmatrix}^T$ *El conjunto de marcados finales es* $M_f = \{\begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix}^T\}$. *Los lenguajes del sistema son:*

$$\begin{aligned}
\mathcal{L}_L(G) &= \{a^m c b^m \mid m \geq 0\}; \\
\mathcal{L}_G(G) &= \{a^m c b^n \mid m \geq n \geq 0\}; \\
\mathcal{L}_D(G) &= \{a^m c b^n d \mid m \geq n \geq 0\}; \\
\mathcal{L}_P(G) &= \{a^m \mid m \geq 0\} \cup \{a^m c b^n \mid m \geq n \geq 0\} \cup \{a^m c b^n d \mid m \geq n \geq 0\}.
\end{aligned}$$

2.6.3 Clases de Lenguajes en RP

Las clases de lenguajes de RP se denotan a continuación

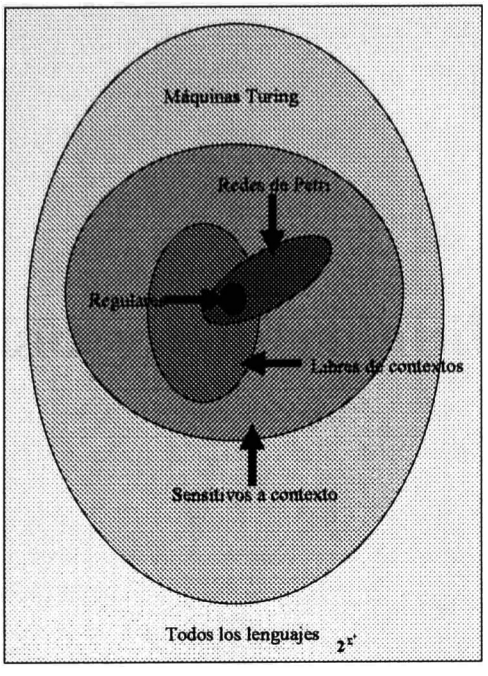


Figura 2.5: Tipos de lenguajes

1. $\mathcal{L}^f$ (respec. $\mathcal{G}^f, \mathcal{D}^f, \mathcal{P}^f$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueo, de prefijo) generados por una *RP generadora* de *etiquetación libre.*

2. $\mathcal{L}$ (respec. $\mathcal{G}, \mathcal{D}, \mathcal{P}$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueo, de prefijo) generados por una *RP generadora* de *etiquetación libre de* ϵ.

3. $\mathcal{L}^\epsilon$ (respec. $\mathcal{G}^\epsilon$, $\mathcal{D}^\epsilon$, $\mathcal{P}^\epsilon$) denota la clase de lenguajes terminales (respec. de cobertura, de bloqueo, de prefijo) generados por una RP generadora de etiquetación arbitraria.

   En la figura 2.5 se presenta en dónde se ubican los lenguajes que genera una *RP.*

Un lector interesado en profundizar en el tema de la Redes de Petri puede consultar [18], [21], [4] y [20].

Capítulo 3

Observabilidad y controlabilidad

Resumen: Debido al problema de identificación presentado en el primer capítulo, se tiene que el sistema debe ser observable para que pueda ser identificable. En el área de SED existen algunas definiciones sobre observabilidad, sin embargo todas ellas hablan sobre un lenguaje observable y proyecciones o restricciones de los lenguajes. Ninguna de estas definiciones hace referencia a inferir en qué estado está el SED. Para solventar esta carencia, en este capítulo se define cuándo una RPI que modela un SED es observable. Además se propone un teorema que da las condiciones necesarias y suficientes para determinar cuándo un SED es observable.

Aunado a la definición de observabilidad se introduce una nueva definición de controlabilidad (basada en la definición introducida por Wonham). Esta definición es más amplia que la ya existente, y ayuda a plantear problemas como seguimiento de modelo y seguimiento de trayectoria (los cuales están fuera del alcance de esta tesis) que con otras técnicas no es posible.

3.1 Introducción

En este capítulo se presentan tres nuevos resultados: el primero es la definición de los conceptos de observabilidad y de controlabilidad para RPI; el segundo son dos teoremas para determinar la observabilidad y controlabilidad de una RPI; y el tercero es un algoritmo para determinar cuándo una RPI es controlable. La propiedad de observabilidad en RPI está estrechamente relacionada con el problema de identificación de SED´s. Mientras que la definición de controlabilidad en RPI viene a complementar el trabajo aquí presentado, ya que la identificación puede ser utilizada en el área de control adaptable.

El concepto de observabilidad para SED's ha sido tratado en diversos trabajos, Wonham en [8] caracteriza la observabilidad en términos de lenguajes. Ramadge define la observabilidad como el problema de determinar el estado actual del sistema, la información disponible son las observaciones parciales; tanto de los estados como de los eventos del sistema. Cüneyt en [1] asume un modelo de observación intermitente; es decir que no hace mediciones directas del estado y sólo puede observar un conjunto específico de posibles eventos, es decir; que si ocurre un evento que no pertenece a ese conjunto, éste no será observado y nunca se sabrá que ha ocurrido un evento. El concepto de observabilidad que se introduce en este trabajo es para SED's cuya representación es una RPI. El estado del sistema o marcado de la RPI se determina con la información del modelo, de las entradas y de las salidas. La diferencia entre los trabajos mencionados y el que aquí se presenta radica en la noción de observabilidad adoptada. La noción de observabilidad adoptada por Cüneyt es que el estado exacto se conoce perfectamente sólo en instantes específicos y fuera de esos instantes los estados son ambiguos, pero esta ambigüedad se resuelve en un número acotado de eventos. Ramadge reconstruye el estado exacto del sistema cada vez que ocurre un evento. En el trabajo que aquí se presenta el estado del sistema se conoce siempre que se presenta una entrada y una salida.

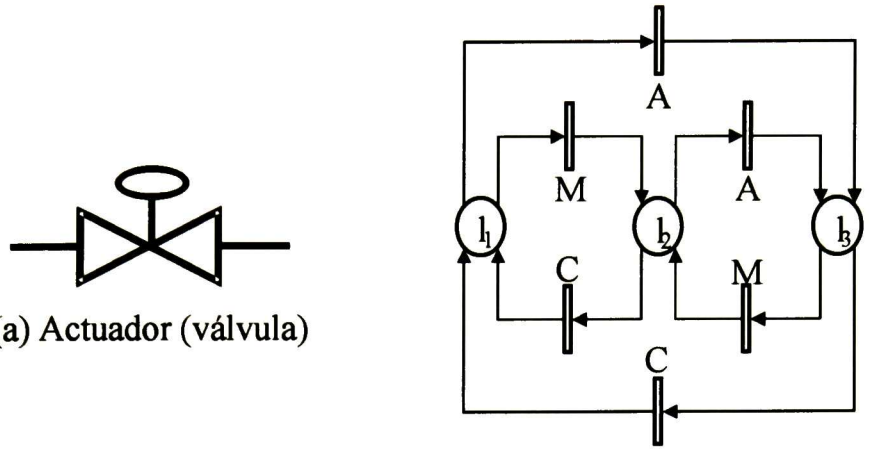
Se introducirá resumidamente una metodología de modelado, la cual está inspirada en una ya existente para máquinas de estado extendidas reportada en [10], un lector interesado puede consultar dicha referencia. Dentro de esta metodología se clasifican las partes del sistema en actuadores, sensores y del sistema en sí. Cada uno de estas partes se modela por separado.

3.1.1 Modelado del actuador

Un actuador se modela con una máquina de estados (ME) de RP [ver anexo de RP's]. Cada señal Sa_i del actuador a_i se representa por una transición t_k, la cual recibe como etiqueta la señal Sa_i. La salida de la transición será el estado estable que alcanza el actuador cuando se le aplica Sa_i.

Ejemplo 3.1 *En la figura 3.1a se representa un actuador que es una válvula cuya posición se ha discretizado en tres niveles: cerrado, medio y abierto. La figura 3.1b es el modelo en RP del actuador (válvula)*

La transición t_1 *representa la señal* A *del actuador, que es* abrir válvula, *de manera análoga las transiciones* t_2 *y* t_3 *representan las señales* medio abrir válvula: M *y* cerrar válvula: C *respectivamente. El lugar* l_1 *representa el estado* válvula carrada. *el lugar* l_2 *el estado* válvula medio abierta *y el lugar* l_3 *representa el estado* válvula abierta *Estos estados*



(a) Actuador (válvula)

(b) Modelo del actuador

Figura 3.1: (a) Un actuador y (b) su modelo en RPI

son los estados estables que alcanza el actuador cuando se aplican las señales; es decir, los estados que se alcanzan cuando se disparan las transiciones que representan dichas señales.

3.1.2 Modelado del sistema

El comportamiento del sistema físico se discretiza en estados. Cada estado e_i es representado por un lugar l_i. Si del estado e_i se puede llegar al estado e_j, entonces se pone una transición t_{ij} de l_i a l_j. A t_{ij} se le asigna el símbolo (no etiqueta) de los estados de los actuadores que hacen que t_{ij} se realice.

Si t_{ij} no depende de ningún estado de los actuadores, entonces t_{ij} no recibe ningún símbolo.

3.1.3 Modelo global

Si una transición t_{ij} del modelo del sistema tiene asignado un símbolo de los estados de los actuadores entonces la transición t_{ij} se une con el estado del actuador que hace que t_{ij} se realice formando un autolazo. Finalmente los lugares que son medibles (que representan un estado que es detectado por un sensor), reciben una etiqueta, que es la etiqueta que representa la señal discreta que envía el sensor cuando está en ese estado.

Ejemplo 3.2 *El sistema a modelar con RPI es un tanque que se presenta en la figura 3.2. El tanque posee un actuador que es la válvula de flujo de entrada y dos sensores que miden el nivel del flujo en el tanque.*

El modelo de la válvula es muy parecido al ejemplo anterior sólo que ahora la posición se discretiza en abrir y cerrar la válvula.

También en la figura 3.2 se presenta el modelo en RPI del sistema. El comportamiento del tanque ha sido discretizado en los estados vacío, nivel bajo *y* nivel alto, *estos estados están representados por los lugares l_1, l_2 y l_3, como del estado nivel bajo se puede llegar al*

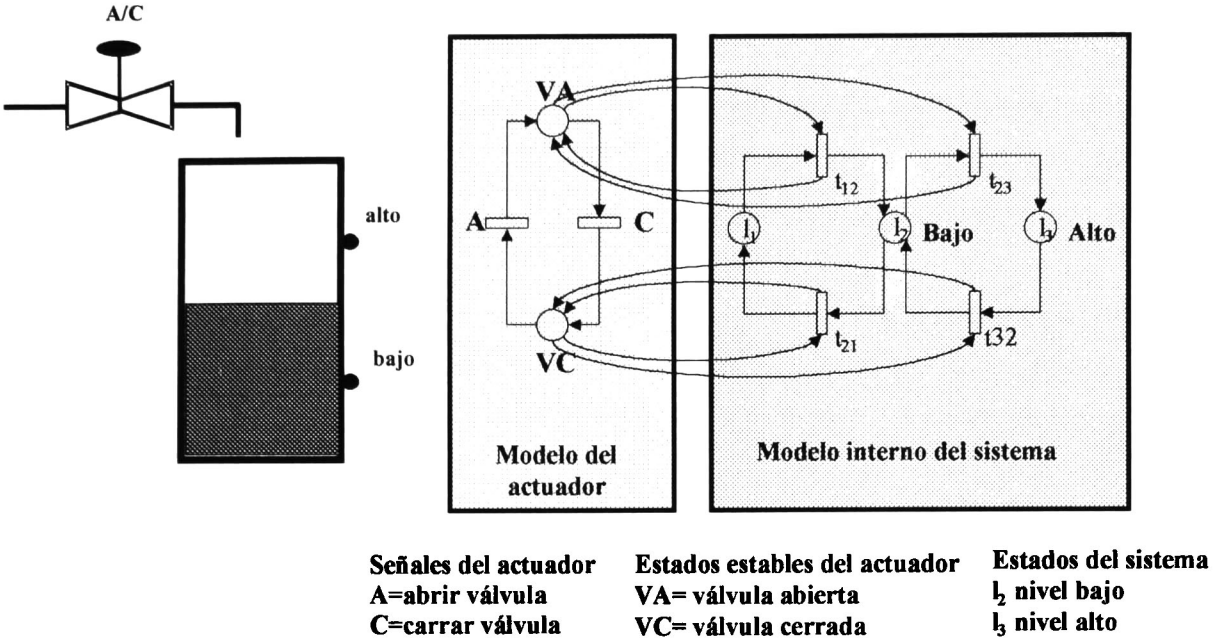


Figura 3.2: Modelo de un tanque utilizando RPI

estado nivel alto cuando la válvula está abierta, existe la transición t_{23} uniendo los lugares que representan esos estados, esta transición tiene asignado el símbolo VA que es un estado estable del actuador. En este ejemplo todas las transiciones tienen asignado un símbolo de los estados de los actuadores.

En el modelo global del tanque se hace un autolazo de la transición t_{23} con el estado estable del actuador VA. Físicamente esto significa que cuando el tanque se encuentra en el nivel bajo, el nivel del tanque puede aumentar hasta llegar a nivel alto si la válvula está abierta. Lo mismo pasa con las demás transiciones del modelo del sistema. Los lugares l_2 y l_3 reciben las etiquetas Bajo y Alto respectivamente, porque representan un estado detectado por un sensor.

A continuación se redefinen las RPI, en este caso son una versión particular de las RPI presentadas anteriormente y son las RPI que se utilizarán a lo largo de esta tesis.

3.2 Redes de Petri interpretadas

Antes de redefinir a las RPI se presentan las siguiente definiciones:

Definición 3.1 *Un actuador es una entidad física sobre la cual se ejecutan acciones determinadas para hacer que el SED o elementos de éste realicen cierta operación u tarea (dependiendo a lo que esté asociado el actuador).*

Definición 3.2 *Un sensor como su nombre lo indica es un dispositivo que es sensible al estado del sistema. Los sensores miden el estado del SED.*

Definición 3.3 *Una entrada o señal de los actuadores, es una señal de un único valor que hace que el actuador cambie de un valor a otro.*

Definición 3.4 *Una salida o una señal de los sensores es el estado del SED indicado por los sensores (Nótese que una señal de salida es un vector).*

Con las definiciones anteriores se pueden representar las entradas y salidas usando símbolos de los alfabetos que a continuación se definen.

Definición 3.5 *Sea* $\Sigma_{entrada} = \{a_i | a_i$ *es una señal de los actuadores*$\}$ *el conjunto de todas las señales de los actuadores*

Definición 3.6 *Sea* $\Sigma_{salida} = \{s_i | s_i$ *es una señal de los sensores*$\}$ *el conjunto de todas las señales de los sensores*

- En sí, a los elementos de un alfabeto se les llama símbolos. Por lo que en este caso en particular es equivalente decir símbolo o señal, ya que los símbolos de los alfabetos son las señales de los actuadores y de los sensores.
- $\Sigma^*_{entrada}$ (Σ^*_{salida}) es el conjunto de todas las palabras de entrada (salida) que se pueden formar con los símbolos de $\Sigma_{entrada}$ (Σ_{salida}).
- w (s) es una palabra de entrada (salida) y es una sucesión finita de símbolos que pertenecen al alfabeto de entrada $\Sigma_{entrada}$ (salida Σ_{salida}).
- $L_{entrada}$ (L_{salida}) es un lenguaje de entrada (salida) formada por palabras w (s) y puede ser cualquier subconjunto de $\Sigma^*_{entrada}$ (Σ^*_{salida}).

Definición 3.7 *Sea* $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$, *donde se permite que* $\Sigma_1 \cap \Sigma_2 \neq \emptyset$. *Sea* $\Sigma = \Sigma_1 \cup \Sigma_2 \neq \emptyset$.

La proyección natural P_i de Σ^* sobre Σ_i^*

$$P_i : \Sigma^* \longrightarrow \Sigma_i^* \qquad (i = 1, 2)$$

se define como

$P_i(\epsilon) = \epsilon$

$P_i(s) = \begin{cases} \epsilon & \text{si} \quad s \notin \Sigma_i \\ s & \text{si} \quad s \in \Sigma_i \end{cases}$

$P_i(\sigma s) = P_i(\sigma)P_i(s) \quad \sigma \in \Sigma^* \ \ s \in \Sigma$

La acción de P_i en una cadena σ es borrar todas las ocurrencias de s en σ tal que $s \notin \Sigma_i$. P_i es la proyección natural de Σ^* sobre Σ_i^*

Definición 3.8 *Si* $P_i : \Sigma^* \longrightarrow \Sigma_i^*$ *es la proyección natural definida anteriormente, entonces* P_i^{-1} *es la proyección inversa de* P_i *y se define como*

$$P_i^{-1} = \{s | P_i(s) \in \Sigma_i^*\}$$

Definición 3.9 *Una RP Interpretada es la cuádrupla* $RPI = (RP, M_0, \lambda, \varphi)$ *donde:*

- $RP = (L, T, E, S)$ *es una RP definida como en el capítulo anterior*
- M_0 *es el marcado inicial*
- $\lambda : T \rightarrow \sum_{entrada} \cup\{\epsilon\}$ *es la función de etiquetación que representa las transiciones que pueden ser habilitadas o deshabilitadas por las señales de los actuadores*
- $\varphi : M \rightarrow \sum_{salida} \cup\{\epsilon\}$ *es una función que representa los sensores que se activan cuando se alcanza cierto marcado.*

Definición 3.10 *Una transición t es controlable ssi $\lambda(t) \neq \epsilon$, es decir que $\lambda(t) = a \in \Sigma_{entrada}$.*

Definición 3.11 *Una transición t es incontrolable ssi $\lambda(t) = \epsilon$.*

Definición 3.12 *Un lugar l es no medible ssi $\varphi(l) = \epsilon$*

Definición 3.13 *Un lugar l es medible ssi $\varphi(l) \neq \epsilon$, es decir que $\varphi(l) = s \in \Sigma_{salida}$*

Evolución de una RPI

1. El disparo de una transición controlable t ($\lambda(t) = a, a \in \Sigma_{entrada}$) se realiza siempre que esté habilitada y que la señal a esté presente.
2. El disparo de un transición incontrolable t ($\lambda(t) = a,$) se puede realizar siempre que esté habilitada.
3. El disparo de una transición t controlable ó incontrolable alcanza un marcado que hace que se generen las señales de los sensores asociados con los lugares medibles que fueron marcados. Como en el presente trabajo se limita al uso de las RPI binarias (salvo se indique lo contrario), la función φ presentada en la definición 3.9 se puede expresar como $\varphi : L \longrightarrow \Sigma_{salida} \cup \{\epsilon\}$. Así, un símbolo de salida $\varphi(L) = [e_i]$ se calcula como:

$$e_i = \begin{cases} \varphi(l_i) & \text{si } M(l_i) = 1 \\ \epsilon & \text{si } M(l_i) = 0 \end{cases}$$

En la figura 3.3 se presenta el esquema de RPI presentado en [4] y el aquí propuesto.

El modelo propuesto es con el que se trabaja por el momento, después se extenderá en trabajos futuros. Como se observa en el modelo de Silva existe prealimentación, es decir, el disparo de una transición puede provocar que se dé un símbolo de salida sin pasar por la dinámica propia del sistema. Este caso no se considera actualmente, pero en trabajos futuros se hará.

3.3 Observabilidad

La propiedad de observabilidad que se propone en este capítulo para los SED´s está definida de la misma forma que en los sistemas continuos, en dónde se establece que si el sistema es observable entonces el estado del sistema se puede conocer a partir de las entradas, de las salidas y de la estructura de éste.

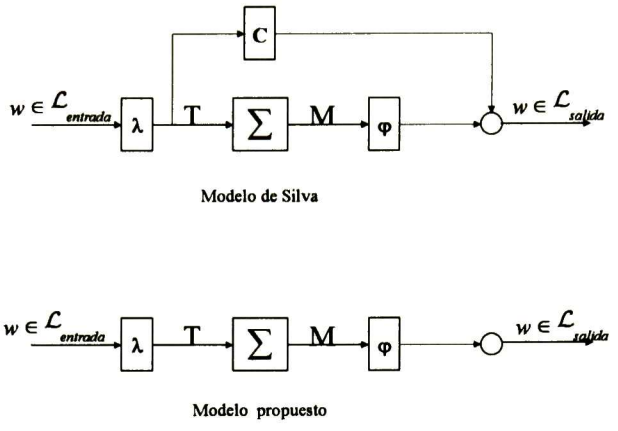


Figura 3.3: Modelos de RPI's

3.3.1 Definición de observabilidad

Definición 3.14 *Una RPI que modela un sistema es observable ssi a partir del conocimiento de sus entradas, salidas y de la estructura del modelo es posible conocer su estado. Es decir, si es posible encontrar una función* $\psi : \sum^*_{entrada} \times \sum^*_{salida} \times \{RPI\} \rightarrow R(RPI, M_0)$[1] *En este caso se dice que el SED es observable.*

Un sistema puede tener lugares que no son medibles, pero en muchos casos la información de las entradas y de las salidas es suficiente para determinar el marcado de esos lugares, debido a ello, los sistemas pueden ser observables y no observables. Dentro de los sistemas observables existen dos clases:

1 los que tienen todos sus lugares medibles, es decir $\forall l_i \in L \longrightarrow \varphi(l_i) \neq \epsilon$

2 los que tienen lugares no medibles $\exists l_i \in L \longrightarrow \varphi(l_i) \neq \epsilon$ pero la información de las entradas y de las salidas es suficiente para conocer su estado en cualquier momento.

En el primer caso se puede considerar que el estado es directamente la salida y el algoritmo de identificación calcula cada transición como la diferencia de marcados consecutivos. Los sistemas que pertenecen al segundo caso no pueden identificarse a partir de la medición del estado, pero sí con la medición de sus entradas y salidas porque éstas contienen información del estado de los lugares no medibles.

En el siguiente apartado se presenta la caracterización de la definición 3.14 en RPI binarias, pero antes se presenta de manera intuitiva el concepto de observabilidad adoptado.

3.3.2 Definición intuitiva del concepto de observabilidad en RPI

Este trabajo se restringe a las RP binarias, en un trabajo futuro se extenderá el resultado de observabilidad a otro tipo de redes.

[1] Es decir, si para una palabara de entrada, una de salida y la RPI se determina un único marcado de la RPI.

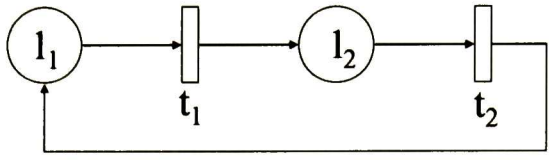


Figura 3.4: Red de Petri binaria

Se puede decir que una RPI binaria es observable si todos sus lugares son medibles, aunque también existen RPI binarias observables que tienen lugares no medibles. En el último caso la información de las entradas, salidas y del modelo son necesarias para determinar el estado en que se encuentra la RPI binaria.

Como un lugar en la RPI binaria es un estado local del SED, el que un lugar sea medible significa que el estado que representa tiene asociado un sensor. Es por ello que cuando la red alcanza un marcado donde los lugares medibles están marcados, se generan los símbolos de salida, los lugares no medibles no generan símbolos de salida.

A continuación se presenta un ejemplo de una RPI binaria donde t_1 y t_2 son no controlables y dependiendo de φ se determina si es observable o no.

Ejemplo 3.3 *Se tiene la RP binaria de la figura 3.4 y los siguientes casos*

- ***caso a**, si l_1 y l_2 no son medibles: $\varphi(l_1) = \varphi(l_2) = \epsilon$, entonces la RPI binaria no es observable.*
- ***caso b**, cuando todos los lugares son medibles: $\varphi(l_1) \neq \epsilon$ y $\varphi(l_2) \neq \epsilon$, automáticamente se conoce el marcado, por lo tanto la RPI es observable.*
- ***caso c**, si uno de los lugares es medible y el otro no: $\varphi(l_1) \neq \epsilon$ y $\varphi(l_2) = \epsilon$, la RPI es observable porque se sabe que si la marca no está en un lugar, entonces está en el otro*

3.3.3 Definición de observabilidad en términos de RPI binarias

La definición 3.14 equivale a decir que un SED modelado con RPI es observable si existe una función $\psi : \sum^*_{entrada} \times \sum^*_{salida} \times \{RPI\} \rightarrow R(RPI, M_0)$.

Debido a que existen transiciones incontrolables, una palabra de entrada puede alcanzar un marcado donde se activen una o varias transiciones incontrolables, alcanzándose, para una misma palabra de entrada varios marcados, es decir, si el SED acepta como palabra de entrada a w, entonces es posible disparar algunas secuencias de transiciones σ_i, donde $P(\sigma_i)|_{transiciones\_controlables} = t_k t_l ... t_h$ y $\lambda(t_k)\lambda(t_l)...\lambda(t_h) = w$ (es una secuencia que preserva el orden de las transiciones controlables tal como la palabra w lo establece). El que sea posible alcanzar diferentes marcados con la misma palabra de entrada se debe a que las transiciones incontrolables pueden dispararse en diferente orden.

Como la existencia de $\psi : \sum^*_{entrada} \times \sum^*_{salida} \times \{RPI\} \rightarrow R(RPI, M_0)$ determina si un SED es observable, entonces es necesario caracterizar el conjunto de marcados alcanzados por la palabra $w \in L_{entrada}$ y después calcular la función ψ. Para esto se tiene la siguiente definición:

Definición 3.15 *Sea $w \in L_{entrada}$, una palabra de entrada a la RPI, entonces el conjunto de marcados alcanzables por la palabra w es $\Omega_w = \{M_k | M_k = M + Cv_k, v_k = \overrightarrow{\sigma_k}, \sigma_k \in P^{-1}(w) \wedge M_k \in R(RP, M_0)\}$.*

El siguiente teorema caracteriza el concepto de observabilidad en términos de los conjuntos Ω_w.

3.3.4 Teorema de observabilidad

Teorema 1 *Sean un SED descrito con RPI binarias y $w \in L_{entrada}$ una palabra de entrada. El SED es observable ssi $\forall M_i, M_j \in \Omega_w$ que sean marcados consecutivos ($M_i \xrightarrow{t_k} M_j, t_k \in T$), $M_i \neq M_j \rightarrow \varphi(M_i) \neq \varphi(M_j)$*

Demostración. ($\rightarrow$) Si un SED es observable, entonces por definición, existe una función $\psi : L_{entrada} \times L_{salida} \times \{RPI\} \longrightarrow R(RP, M_0)$, es decir, $\psi(w, s_s, RPI) = M_i$, donde w es la palabra de entrada y s_s es la palabra de salida, M_i debe pertenecer a Ω_w, es decir $M_i \in \Omega_w \subseteq R(RP, M_0)$. Supóngase por el momento que existe $M_j = M_0 + C\overrightarrow{t_k}$, $M_j \neq M_i$, con $\varphi(M_i) = \varphi(M_j)$, $t_k \in T$ es una transición incontrolable, es decir, $M_j \in \Omega_w$ y es un marcado inmediato sucesor de M_i. Entonces esto implica que $\psi(w, s_s, RPI) = M_j$ porque M_j es alcanzado por la RPI (es también una solución de la ecuación de estados de la RPI para la misma palabra w) y no hay ningún cambio en el símbolo de salida s_s cuando el marcado cambia de M_i a M_j. En este caso ψ no es una función, ya que para dos marcados distintos existe un mismo símbolo de salida, así que el sistema no es observable, lo cual es una contradicción, por lo que no deben existir dos marcados consecutivos $M_j \neq M_i$ con el mismo el símbolo de salida $\varphi(M_i) = \varphi(M_j)$ en el mismo conjunto Ω_w.

($\leftarrow$) (A) Si $\forall w, \forall M_i, M_j \in \Omega_w$, $M_i \neq M_j \rightarrow \varphi(M_i) \neq \varphi(M_j)$ entonces ψ se puede calcular como $\psi(w, s_s\varphi(M_i), RPI) = M_i$. ■

El problema de diseño de observadores y algoritmos para caracterizar cuándo una RPI es observable está fuera del alcance de esta tesis.

3.4 Controlabilidad

El concepto de controlabilidad está relacionado con la interpretación asociada a las transiciones. En el capítulo anterior se definió la función λ que es la que les asigna significado físico a las transiciones dado el lenguaje de entrada al sistema. Aquí se presenta una definición de controlabilidad para SED modelados con RPI.

En el campo de los Sistemas Discretos, Wonham define la Controlabilidad desde el punto de vista de lenguajes, así; para un sistema G el lenguaje de una especificación K (la especificación señala qué es lo que se quiere que haga el sistema) es controlable con respecto al lenguaje del sistema $L(G)$, si todos los prefijos de la especificación seguidos de transiciones incontrolables que aparecen en el lenguaje del sistema también aparecen en los prefijos de la especificación; porque de otra forma, el sistema podría seguir por esas transiciones incontrolables y no hacer lo que indica la especificación. Así $K \subseteq \Sigma^*$ es controlable con respecto a G si

$$\overline{K}\Sigma_u \cap L(G) \subseteq \overline{K}$$

El concepto de controlabilidad introducido por Wonham pide que un conjunto de secuencias (que llevan del estado inicial al estado deseado) dado apriori, sea controlable.

El concepto de controlabilidad que ahora se propone, requiere la existencia de una secuencia de disparo de transiciones controlable en el sentido de Wonham que lleven al sistema del estado inicial al estado deseado.

3.4.1 Cuándo un SED es controlable

Sea una RPI y M_f el conjunto de estados finales deseados. La RPI es controlable ssi

1. $\forall m_f \in M_f \; \exists \sigma = \sigma_l \sigma_j \mid M_0 \xrightarrow{\sigma l} M_x \xrightarrow{\sigma_j} m_f \qquad i,j \in \mathcal{N} \cup \{0\}$ tal que $l + j = k =$ longitud de σ

   y

2. $\forall l \in [0,k]$ se satisface una de las dos condiciones siguientes:

   (a) $\nexists s \in T_i \mid M_x$ no habilita a s ó

   (b) $\forall s \in T_i \mid M_x \xrightarrow{s} M_y \qquad$ y tomando M_y como M_0, M_y cumple con 1.

donde

- T_i es el conjunto de transiciones incontrolables, $t \in T_i \rightarrow \lambda(t) = \epsilon$
- M_x, M_y, etc son marcados alcanzables intermedios entre M_0 y m_f

Esta definición dice que una RPI es controlable si existe una secuencia $\sigma \mid M_0 \xrightarrow{\sigma} m_f \in M_f$ y ya sea que σ esté compueta solamente por transiciones controlables o contenga algunas transiciones incontrolables pero, irremediablemente σ alcanzará el estado deseado.

La definición de controlabilidad propuesta, sirve para los casos en que las condiciones de inicio del sistema no se conocen. Si las condiciones de inicio no se conocen, no se podría hacer el análisis de que una especificación K sea controlable con respecto al sistema porque no se sabría cuál es el lenguaje del sistema. Usando la definición de controlabilidad que aquí se presenta se pueden plantear problemas de seguimiento asintótico de modelo o trayectoría.

Durante el desarrollo de la presente tesis se vió que no era necesaria la controlabilidad para realizar los identificadores asintóticos, sin embargo se presenta aquí por completez y por mostrar el trabajo realizado.

Proposición 3.1 *Un SED es controlable ssi el siguiente problema tiene solución*

1. $\exists \sigma_j \mid m_f = M_0 + C\overrightarrow{\sigma_j} \wedge m_f \in M_f$ y

2. $\nexists \sigma_j' \in pr(\sigma_j) | M_0 + C\overrightarrow{\sigma_j'} > EKi$

3. o si $\exists \sigma_j' = pr(\sigma_j) | M_0 + C\overrightarrow{\sigma_j'} > EKi$

   entonces hacer $M_0 = M_0 + C\overrightarrow{\sigma_j'}$ y regresar a 1

donde

- $pr(\sigma)$ es el conjunto de los prefijos de la secuencia σ
- E es la matriz de entrada
- Ki es un vector del kernel izquierdo de la matriz λ

Demostración. En la línea 1 se pide que exista una secuencia σ_j que lleva del estado inicial al estado final. Además en la línea 2 se pide que los prefijos de dicha secuencia no activen transiciones incontrolables, o que si las activan (línea 3), entonces éstas irremediablemente lleven al estado final. ■

Desafortunadamente se está hablando de un problema de alta complejidad computacional, ya que se tiene que buscar en todas las secuencias del sistema.

La siguiente proposición resulta en un algoritmo polinomial para checar la controlabilidad, aunque éste es sólo una condición suficiente.

Proposición 3.2 *Si el siguiente problema tiene solución, entonces el sistema es controlable.*

Sean
ϕ el conjunto de secuencias que llevan del estado inicial a los estados finales
$\sigma \in \phi$ una secuencia que lleva a estados finales
Realizar
tomar una secuencia $\sigma_i \in pr(\sigma)$
encontrar el vector característico $\overrightarrow{\sigma_i}$ de la secuencia σ_i
$l = |\Sigma_{entrada}|$ número de elementos en el alfabeto de entrada
$sum = 0$
for i=1 *to* l

sum=sum $+\overrightarrow{\sigma_i}(i)$

while $\sigma \neq \emptyset$
for i=1 *to* sum
minimizar $(\overrightarrow{\sigma_j})$
s.a

$C\overrightarrow{\sigma_j} > EKi - M_0$
$-I\overrightarrow{\sigma_j} > \overrightarrow{\sigma_i}$
$\sigma_1 + \sigma_2 + ... + \sigma_l = i$
if $\overrightarrow{\sigma_j} = \emptyset$
tomar otra secuencia de σ
else i++
end
end

3.4.2 Condiciones suficientes para la Controlabilidad en algunas Subclases de Redes de Petri

Afortunadamente para algunas subclases de RP se tienen condiciones necesarias y suficientes para determinar si son o no controlables. Tal es el caso de las RP que se mencionan a continuación.

En lo sucesivo el conjunto de transiciones T se ve particionado en los conjuntos Tc y Ti que son el conjunto de las transiciones controlables y de las incontrolables respectivamente.

Teorema 2 *Sea N una RP fuertemente conexa, donde* $T = Tc \cup Ti$, *si al eliminar el conjunto Ti la RP restante es una MEFC (máquina de estados fuertemente conexa) y* $\forall t_i \in Ti, \forall l_i \in L$ *se cumple que:*

si $\sum M_0(l_j) < |{}^\bullet t_i| \longrightarrow$ *El sistema es controlable*

Demostración. El número de marcas que t_i necesita para dispararse es por lo menos $|{}^\bullet t_i|$ en cualquier marcado.

De $\sum M_0(l_j) < |{}^\bullet t_i|$ se deduce que $|{}^\bullet t_i| = M_0(l_j) + b$, donde b es una constante positiva.

Una MEFC tiene un único p-semiflujo $Y^T = [\ 1\ \ 1\ \ \cdots\ \ 1\ ]$ (Ver [12])

Multiplicando la ecuación de estados $M_{k+1} = M_0 + Cv_k$ por el p-semiflujo anterior se obtiene: $Y^T M_{k+1} = Y^T M_0 \longrightarrow \sum M_{k+1} = \sum M_0$, esto indica que el número de marcas en una MEFC se conserva para todos los marcados alcanzables.

Del punto anterior se deduce que si el marcado inicial no tiene $|{}^\bullet t_i|$ marcas entonces jamás va a tenerlas por lo que nunca habilitará a las transiciones incontrolables

Hasta aquí se cumple que no se habilite ninguna transición incontrolable, lo siguiente es encontrar las secuencias que vayan del marcado inicial a los marcados finales.

Por ser una MEFC todos los marcados son alcanzables a partir de cualquier estado, por lo que existe una secuencia que lleve del marcado inicial al marcado final y entonces esta secuencia cumple con la definción de controlabilidad. ■

Teorema 3 *Sea N una RP fuertemente conexa, donde* $T = Tc \cup Ti$ *y MF el conjunto de marcados finales, tales que el disparo de una* $t_j \in T_i$ *evita que se llegue a un marcado final. Si al eliminar* T_i *de la RP se obtiene un GMFC (grafo marcado fuertemente conexo) y* $\forall t_i \in Ti$, $\forall l_j \in L$ *y* $\exists t \in Tc$ *que necesita dispararse para llegar al estado final, tal que* $|\ t^\bullet\ | > 1$ *se cumple que:*

${}^\bullet t_i \subseteq t^\bullet \longrightarrow$ *El sistema es incontrolable*

Demostración. El disparo de t agrega marcas a todos sus lugares de salida $l_j \in t^\bullet$ entonces $M_k \xrightarrow{t} M_{k+1}$ por lo que $M_{k+1}(l_j) \geqslant 1\ \forall l_j \in t^\bullet$ como ${}^\bullet ti \subseteq t^\bullet$ entonces t_i se ha habilita siempre que t se dispare. Como no se puede evitar que t_i se dispare y no se llegará a un estado final, entonces el sistema es incontrolable ■

Teorema 4 *Sea N una RP fuertemente conexa, donde* $T = Tc \cup Ti$ *y MF el conjunto de marcados finales, tales que el disparo de una* $t_j \in T_i$ *evita que se llegue a un marcado final. Si al eliminar* T_i *de la RP se obtiene un GMFC (grafo marcado fuertemente conexo) y* $\forall t_i \in Ti$, $\forall l_j \in L$ *y* $\exists t \in Tc$ *que necesita dispararse para llegar al estado final, tal que* $|{}^\bullet\ t\ | > 1$ *se cumple que:*

${}^\bullet t_i \subseteq {}^\bullet t \longrightarrow$ *El sistema es incontrolable*

Demostración. Para que la transición t se dispare todos sus lugares de entrada deben estar marcados, esto es $M(l_j) > 1\ \forall(l_j) \in {}^\bullet t$ y como ${}^\bullet ti \subseteq {}^\bullet t$ entonces siempre que t esté habilitada ti se puede disparar llegando a un estado que no es el final. ■

Capítulo 4

Identificación de SED´s

Resumen: Existen varios métodos para la identificación de SED's, los cuales son aplicables dependiendo de la información previa que se tenga del sistema. Así, los mínimos cuadrados son aplicables exclusivamente cuando se conoce el número de transiciones y lugares del sistema, su aplicación es muy fácil. Sin embargo, si no se conoce el número de lugares y transiciones, entonces los esquemas de identificación estructural deben ser empleados.

En este capítulo se presentan tres algoritmos de identificación. El primero utiliza los mínimos cuadrados fuera de línea. El segundo trabaja en línea (asintótico), pero requiere que el sistema sea totalmente medible. Finalmente el tercero trabaja en línea y sólo requiere que el sistema sea observable.

4.1 Introducción

El concepto de identificación en el límite, formulado por [6], ha sido de gran importancia en estudios teóricos de identificación a través de ejemplos. Este concepto se basa en encontrar en el infinito el autómata que representa el comportamiento del sistema a inferir, al ir presentando más y más ejemplos de una determinanda regla. [6] también establece que el problema de encontrar el autómata mínimo que sea compatible con una muestra positiva de un leguaje es NP.

Dada esta complejidad, los trabajos que se han hecho en el área de identificación de SED's están dirigidos a dominios específicos, como son los autómatas finitos y los lenguajes regulares. Algoritmos polinomiales se han encontrado para ciertas clases de lenguajes. Los algoritmos en [3] y [7] identifican ciertas subclases de lenguajes regulares a partir de muestras positivas. El algoritmo que se propone también identifica una subclase de lenguajes regulares sólo que trabaja en línea con el sistema.

En este capítulo se presenta el algoritmo de identificación que se propone y las propiedades de las cuales fue derivado.

A continuación se presentan las características de los algoritmos de identificación existentes, así como las del propuesto.

En [3] se identifica un autómata determinista cero reversible, el algoritmo que presenta se llama ZR (zero reversible). La muestra positiva que utiliza el algortimo para identificar el aceptor es parte del lenguaje de algún aceptor cero-reversible. El algoritmo ZR se puede implementar en un tiempo $O(n\alpha(n))$, donde n es uno más que la suma de las longitudes de las cadenas de entrada y α es una función de crecimiento mínimo. El lenguaje que acepta el autómata que identifica el algoritmo de [7] es un lenguaje I-reversible. Un lenguaje $\mathcal{L}$ es I-reversible si existe un aceptor Θ I-reversible tal que $\mathcal{L} = \mathcal{L}(\Theta)$. El tiempo total de ejecución del algoritmo está acotado por una función polinomial de la muestra positiva dada $\mathcal{Q}$. El tiempo total es $O(n^2m^3)$, donde n uno más que la suma de las longitudes de las secuencias en $\mathcal{Q}$ y m es la cardinalidad del conjunto de transiciones o eventos.

4.2 Algortimo de identificación utilizando mínimos cuadrados

Para poder utilizar el método de los mínimos cuadrados (ver capítulo 1) en el problema de identificación de SED's representados por una RPI, es necesario conocer el número de lugares y transiciones del modelo que se identificará. Un identificador tipo mínimos cuadrados para un sistema cuyos estados son medibles, se presenta en la figura 4.1. Este identificador se puede utilizar bajo algunas condiciones que a continuación se estudian.

4.2.1 Uso de los mínimos cuadrados en RP

En el problema de mínimos cuadrados se tiene que el modelo se expresa como $Y_M = X\theta$, y la ecuación de estados de la RP es $M_{k+1} = M_k + Cv_k$; así, si se quiere utilizar este método para identificar RP habrá que modificar la ecuación de estados de la siguiente forma:

$$M_{k+1} - M_k = Cv_k$$

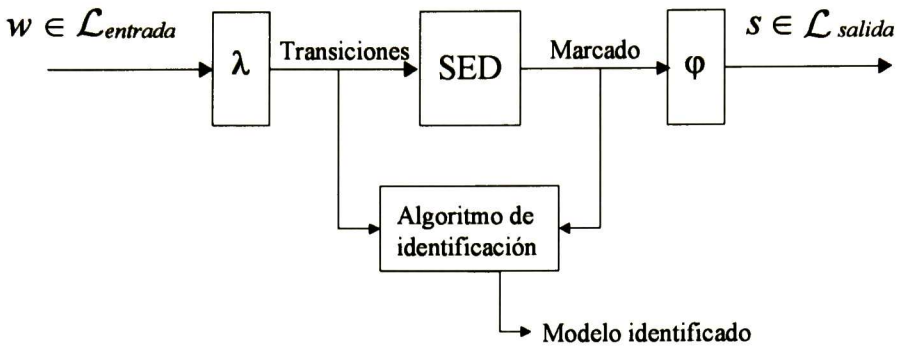


Figura 4.1: Identificador

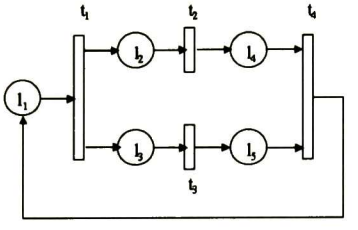


Figura 4.2: Red de Petri totalmente medible y controlable

o, más concretamente como:

$$\Delta M_k = C v_k$$

Recordando que el identificador de mínimos cuadrados es:

$$\theta = (X^T X)^{-1} X Y$$

entonces se tiene que $Y = \Delta {M_k}^T$ y $X = {v_k}^T$; el cálculo de θ será igual a C^T

Ejemplo 4.1 *Sea el sistema descrito por la RPI de la figura 4.2*
Si se tiene la siguiente evolución del sistema:

$$\begin{bmatrix} M_1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow[(t_1)]{\begin{bmatrix} v_1 \\ 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \begin{bmatrix} M_2 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \xrightarrow[(t_2)]{\begin{bmatrix} v_2 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}} \begin{bmatrix} M_3 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix} \xrightarrow[(t_3)]{\begin{bmatrix} v_3 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}} \begin{bmatrix} M_4 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \xrightarrow[(t_4)]{\begin{bmatrix} v_4 \\ 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}} \begin{bmatrix} M_5 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

resolviendo el problema de mínimos cuadrados con estas entradas y salidas se obtiene que:

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

Ejemplo 4.2 *Para el sistema anterior, si t_2 y t_3 se disparan simultáneamente, se tiene la siguiente evolución:*

$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \xrightarrow[(t_1)]{\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}} \begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \\ 0 \end{bmatrix} \xrightarrow[(t_2,t_3)]{\begin{bmatrix} 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}} \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{bmatrix} \xrightarrow[(t_4)]{\begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Ejemplo 4.3 *En este caso el identificador de mínimos cuadrados no funciona porque la matriz que se forma es singular. Para estas situaciones existe una extensión de los mínimos cuadrados que evita la inversa de la matriz singular [13]. Usando este nuevo método se identifica el siguiente modelo:* $C = \begin{bmatrix} -1 & 0 & 0 & 1 \\ 1 & -1/2 & -1/2 & 0 \\ 1 & -1/2 & -1/2 & 0 \\ 0 & 1/2 & 1/2 & -1 \\ 0 & 1/2 & 1/2 & -1 \end{bmatrix}$

Como se observa este modelo no es el real, de hecho esta C no sirve ya que no representa a ninguna RP, los números racionales no son permitidos en la definición de RP. Esto se debe a que el disparo simultáneo de transiciones oculta la información. Para poder identificar se necesita que las secuencias aporten la suficiente información para dicernir la estructura de la RP. La siguiente definición resume este tipo de secuencias.

Definición 4.1 *Una secuencia de entrada σ_i se dice que es de excitación persistente si permite que se pueda identificar la RP, es decir si tiene información suficiente como para decidir la estructura de la RP.*

Hipótesis 4.1 *De ahora en adelante se supone que la entrada a un sistema que se va a identificar es una secuencia de excitación persistente.*

El problema de saber cuándo una sequencia es de excitación persistente está fuera del alcance de esta tesis, pero en algunos casos determinar la secuencia de excitación persistente más corta puede ser útil.

4.2.2 Algoritmo de mínimos cuadrados en RP

De acuerdo con los resultados presentados en la subsección previa se tiene el siguiente resultado.

Proposición 4.1 *Si a un sistema totalmente medible y controlable se le da una secuencia de entrada de excitación persistente, de longitud N, entonces la matriz C del sistema (y por lo tanto la RPI) se puede calcular de la siguiente forma:*

$$C^T = (X^T X)^{-1} X^T Y$$

donde $X^T = \begin{bmatrix} v_0^T \\ v_1^T \\ \vdots \\ v_N^T \end{bmatrix}$ *y* $Y = \begin{bmatrix} \Delta M_0^T \\ \Delta M_1^T \\ \vdots \\ \Delta M_N^T \end{bmatrix}$ *donde N es el horizonte de observación.*

Demostración. Como se observa, sólo se está usando el identificador de mínimos cuadrados y se está pidiendo que la señal sea de excitación persistente. ■

Escrita esta proposición en forma de algoritmo se tiene:

Algorithm identificación_1;

Input: vectores Y y X;

Output: C; /*la matriz de incidencia de la red de Petri */

/* Cálculo de la matriz C por el estimador de mínimos cuadrados */

$C^T = (X^T X)^{-1} X^T Y$

Presentar C como el sistema identificado;

end identificación_1

Este identificador de mínimos cuadrados no se puede utilizar en línea con el sistema, se deben hacer corridas del sistema para obtener los vectores de entrada y de salida (fuera de línea).

Las siguientes propiedades sirven para plantear un algoritmo de identificación asintótica (en línea).

4.3 Algoritmo de identificación asintótico para sistemas totalmente medibles y controlables

Nota 4.1 *Las RP que se utilizarán son las **RP invariantes**, es decir que los arcos que entran y salen de una transición no varían con los eventos de entrada ni con los de salida ni con el tiempo.*

A continuación se menciona una proposición que permite conocer para t_k el valor de su columna en la matriz C. Esta columna se denotará por $C(\bullet, t_k)$. Esta información resultará útil en la construcción de un modelo para el SED.

Proposición 4.2 *Sea una RP y sean σ_i y σ_j dos secuencias que contienen a t_k tal que $M_x^i \xrightarrow{t_k} M_y^i$ $t_k \in \sigma_i$ y $M_x^j \xrightarrow{t_k} M_y^j$ $t_k \in \sigma_j$ en la RP, entonces $C(\cdot, t_k) = M_y^i - M_x^i = M_y^j - M_x^j$*

Demostración. $M_y^i = M_x^i + C\overrightarrow{t_k} \rightarrow M_y^i - M_x^i = C(\cdot, t_k)$ similarmente
$M_y^j = M_x^j + C\overrightarrow{t_k} \rightarrow M_y^j - M_x^j = C(\cdot, t_k)$
Como la matriz C es constante (no varía con los símbolos de entrada, de salida ni con el tiempo), entonces se deduce que $C(\cdot, t_k) = M_y^i - M_x^i = M_y^j - M_x^j$ ■

Con esta información podemos implementar un algoritmo de identificación asintótica, como se observa, se requiere que todos los lugares de la RP sean medibles.

Proposición 4.3 *Si se conoce todo el marcado de la RP y la señal de entrada es de excitación persistente, entonces el siguiente algoritmo*[1] *implementa una identificación asintótica.*

Algorithm identificación_2;

Input: Ninguna;

Output: C; /*la matriz de incidencia de la red de Petri */

/* Inicialización */

$M = M_0$;

$i = 0$;

/* Iteraciones */

loop

while $(Nuevo\_marcado \neq M)$; /* Se obtiene un nuevo marcado */

if $((Nuevo\_marcado - M) \notin C)$; /* La diferencia no es una columna de C * /

then $\{C(\bullet, t_{i+1}) = Nuevo\_marcado - M$;/* aqui se utiliza la proposición anterior para calcular la columna de t_{i+1}*/

$i = i + 1;\}$ /* Se da de alta la nueva transición */

$M = Nuevo\_marcado$;

Presentar C como el sistema identificado;

end loop

end identificación_2.

Demostración. Por la proposición anterior se sabe que una transición t_k genera una columna de la matriz de incidencia $C(\bullet, t^k) = M' - M$, donde $M \xrightarrow{t_k} M'$ La transición debe ser añadida como columna en C si no existe previamente, esto es lo que hace justamente el algortimo. Finalmente, todas las transiciones deben aparecer en la entrada, porque la señal de entrada es de excitación persistente. ■

[1] Propiamente el programa que se describe a continuación no es un algoritmo, ya que nunca se detiene, pero usaremos este término abusando de él y entendiendo que cada iteración es una corrida del algoritmo.

El algoritmo que se especifica en la proposición anterior puede ser utilizado en línea, resultando sencillo de programar y bastante rápido (su complejidad es lineal).

Este algoritmo supone que todos los lugares son medibles, de no ser así se debe utilizar un observador del estado (un método para conocer el estado del sistema). Sin embargo, si no se quiere utilizar dicho esquema, entonces las siguientes propiedades nos permiten obtener un algoritmo que no necesita observador.

4.4 Algoritmo de identificación asintótico para sistemas observables

Esta primera proposición sirve para determinar en qué estado está un actuador conociendo la última entrada que ha recibido. Por tanto, aunque no exista un sensor para saber el estado del actuador, este puede ser inferido de la entrada dada al sistema (suponiendo que el sistema nunca falla).

Proposición 4.4 *Cada vez que se dispara una transición controlable, el actuador asociado con esa transición alcanza un estado que es identificado completamente debido a la transición controlable.*

Demostración. Por construcción, en el modelo de un actuador, cuando se dispara una transición con etiqueta "a" se llega al estado [a]. Ver la metodología de modelado en la sección 3.1.1. ■

Nota 4.2 *La matriz de incidencia C de la RPI se puede descomponer de la siguiente forma: $C = \begin{bmatrix} C_{medible} \\ C_{no\_medible} \end{bmatrix}$, donde $C_{medible}$ contiene la información de todos los lugares que tienen asociado un símbolo de salida y de todos los lugares que representan los estados de los actuadores. Todos los lugares representados en $C_{medible}$ se pueden obtener de la información entrada/salida del SED, porque o el lugar genera un símbolo de salida cuando está marcado o porque el lugar se marca cuando se dispara una transición controlable. Sin embargo la información de $C_{no\_medible}$ se debe inferir a medida que el SED evoluciona.*

Definición 4.2 *Un aceptor de un lenguaje $\mathcal{L}$ se dice que es invariante consistente si $\forall \sigma_i, \sigma_j \in \mathcal{L}$ con $\sigma_i \neq \sigma_j$ y $\vec{\sigma_i} = \vec{\sigma_j}$ entonces ambas secuencias alcanzan el mismo estado del aceptor (ver [7]).*

Hiraishi mostró que una RP binaria que es completamente medible $\forall l_i \in L$, $\varphi(l_i) \neq \epsilon$ y completamente controlable $\forall t_i \in T$, $\lambda(t_i) \neq \epsilon$, es un aceptor invariante consistente de un lenguaje. La demostración de esta aseveración es fácil, ya que tomando dos secuencias diferentes $\sigma_i, \sigma_j \in \mathcal{L}(RP)$ tal que $\vec{\sigma_i} = \vec{\sigma_j}$, como no hay transiciones incontrolables, entonces una secuencia σ sólo puede alcanzar un único marcado, entonces se tiene que $M_0 + C\vec{\sigma_i} = M_0 + C\vec{\sigma_j}$

Sin embargo, si se quiere extender este resultado a RPI binarias que no son completamente medibles y controlables, entonces no se tiene un aceptor invariante consistente, como se explica a continuación.

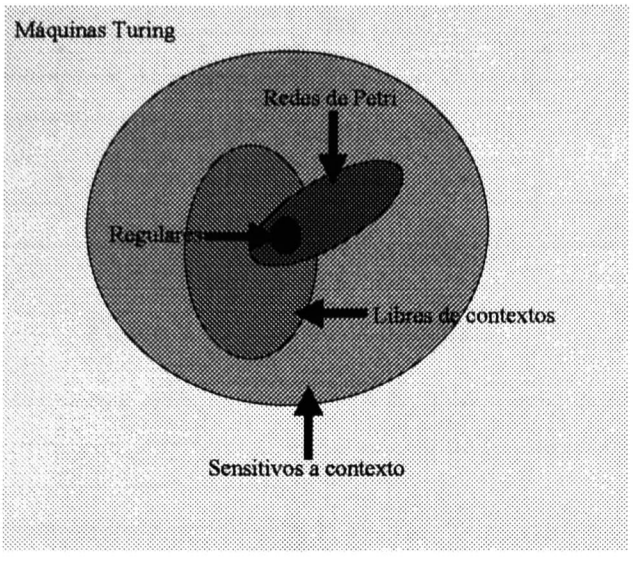


Figura 4.3: Tipos de lenguajes

Si dos secuencias de entrada σ_i y σ_j son diferentes tal que $\vec{\sigma_i} = \vec{\sigma_j}$ no se puede asegurar que ambas lleguen al mismo estado, puesto que hay transiciones no controlables, es decir;

$$M_0 + C \begin{bmatrix} \vec{\sigma_i} \\ \vec{x_i} \end{bmatrix} \neq M_0 + C \begin{bmatrix} \vec{\sigma_j} \\ \vec{x_j} \end{bmatrix} \longrightarrow$$

$$M_0 + \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} \vec{\sigma_i} \\ \vec{x_i} \end{bmatrix} \neq M_0 + \begin{bmatrix} C_1 & C_2 \end{bmatrix} \begin{bmatrix} \vec{\sigma j} \\ \vec{x j} \end{bmatrix}$$

$$M_0 + C_1\vec{\sigma_i} + C_2\vec{x_i} \neq M_0 + C_1\vec{\sigma_j} + C_2\vec{x_j}$$

en general $C_{no\_medible}\vec{x_i} \neq C_{no\_medible}\vec{x_j}$, ya que la parte no controlable no necesariamente es igual. Entonces, para dos secuencias σ_i, σ_j con $\vec{\sigma_i} = \vec{\sigma_j}$, una RPI no necesariemente alcanza el mismo marcado. Por tanto no se pueden utilizar los resultados de Hiraishi, se necesita estudiar más a las RP para poder identificar sistemas que no son totalmente medibles y/o totalmente controlables.

Por otra parte, en [17], [19] y [20] se presentan las clases de lenguajes que pueden aceptar las RPI. La clase de lenguajes qe aceptan las RPI son los lenguajes regulares, algunos libres de contexto y otros sensitivos a contexto, como se muestra en la figura 4.3.

En este trabajo se utilizan las RPI que generan lenguajes regulares. La siguiente proposición demuestra por qué se tienen lenguajes regulares.

Proposición 4.5 *Una RPI binaria genera lenguajes regulares*

Demostración. Sea una RPI binaria. Entonces su grafo de alcanzabilidad es finito; ya que si n es el número de lugares entonces se tendrán a lo más 2^n estados. Claramente, si las transiciones están etiquetadas con diferentes etiquetas, entonces se tendrá un lenguaje regular. En otro caso, si existen transiciones con etiquetas repetidas o etiquetadas con

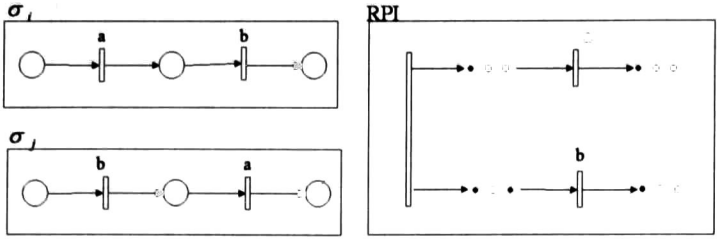


Figura 4.4: Secuencias paralelas

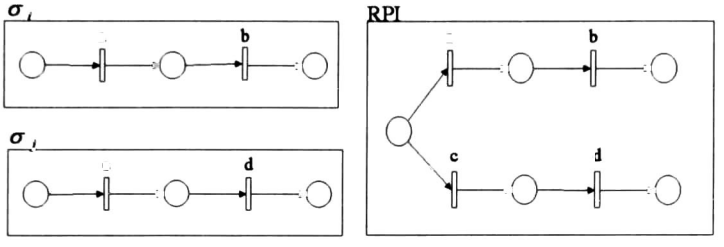


Figura 4.5: Una secuencia o la otra

el símbolo ϵ, entonces se puede tratar este caso como una sustitución (un símbolo se ha sustituido por otro) y de acuerdo con los teoremas presentados en [5], el lenguaje sigue siendo regular. ■

Como se dijo en el capítulo anterior, en este trabajo se utilizan RPI binarias, por tanto se está hablando de lenguajes regulares. A continuación se observa que si se tiene dos secuencias, representadas cada una en una RPI, entonces existe una RPI que acepta a las dos secuencias. Esta característica permite ir incluyendo, paso a paso, en una RPI nuevas secuencias. Será utilizada para identificar una RPI conforme se observen nuevas secuencias.

Proposición 4.6 *Sean σ_i, σ_j secuencias que son aceptadas por las redes de Petri RPI_i y RPI_j respectivamente, entonces existe una RPI_k que acepta ambas secuencias.*

Demostración. a) Si σ_i y σ_j son secuencias que tienen símbolos cambiados, por ejemplo $\sigma_i = ...a...b...$ y $\sigma_j = ...b...a...$, entonces la RPI de la figura 4.4 acepta ambas secuencias.

b) Si σ_i y σ_j, no tienen elementos comunes, por ejemplo $\sigma_i = ab$ y $\sigma_j = cd$ entonces la RPI de la figura 4.5, acepta ambas secuencias.

c) Si σ_i y σ_j tienen elementos comunes y en el mismo orden al inicio, por ejemplo $\sigma_i = abcd$ y $\sigma_j = abef$, entonces la RPI de la figura 4.6 acepta ambas secuencia.

d) Si σ_i y σ_j tienen elementos comunes y en el mismo orden al final, por ejemplo $\sigma_i = abef$ y $\sigma_j = cdef$, entonces la figura 4.7 representa la RPI que acepta ambas secuencias.

Combinando estos esquemas se puede tener cualquier secuencia. ■

La siguiente proposición nos dice cómo representar secuencias que aparentemente son contradictorias, es decir que aparecen símbolos (eventos) en diferente orden.

Proposición 4.7 *Sean $\sigma_i = ...t_i...t_j...$ y $\sigma_j = ...t_j...t_i...$ dos secuencias que pertenencen al mismo lenguaje aceptado por la RPI. Entonces t_i y t_j pertenecen a p-componentes diferentes.*

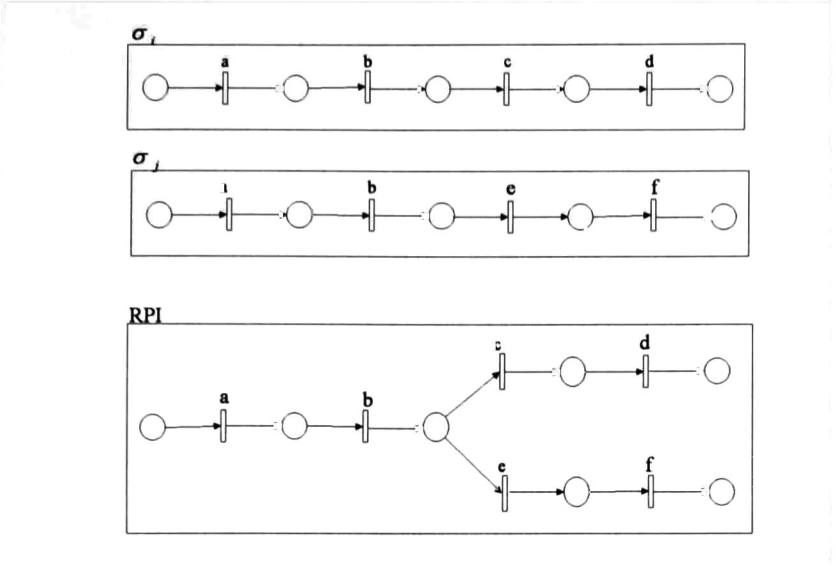


Figura 4.6: Prefijos

Demostración. *Una p-componente es un camino dirigido de t_i a t_j (o de t_j a t_i, pero no ambos). Si ambas t_i y t_j pertenecieran a una única p-componente, entonces sólo habría un camino dirigido y alguna de las secuencias no debería ser aceptada por la RPI.* ■

Corolario 4.1 *Para t_i y t_j de la proposición anterior, se deduce que ambas son independientes. Ver [14].*

Demostración. Es evidente, ya que t_i y t_j pertenecen a diferentes p-semiflujos. ■

Sin embargo, que sean independientes no quiere decir que se ejecuten en paralelo, como se muestra en la figura 4.8.

Viendo los grafos de alcanzabilidad de las RP de la figura 4.8, es claro el por qué los autómatas finitos (el entrelazado) no representan el paralelismo.

De la proposición 4.2 y de la anterior se puede ver que si t_i y t_j son independientes, entonces en la matriz C quedan bloques independientes para t_i y t_j (no hay conexión entre ellas porque aparecen en p-componentes diferentes, o porque se forman autolazos que son cero en la matriz de incidencia).

El siguiente algoritmo forma secuencias aún cuando las transiciones son independientes:

Algorithm Forma_secuencia

Input: Ninguna;

Output: una secuencia

$M = M_0$

$i = 0$

loop

while $((M' = read(marcado)) == M)$;

if $((M' - M) \notin C)$

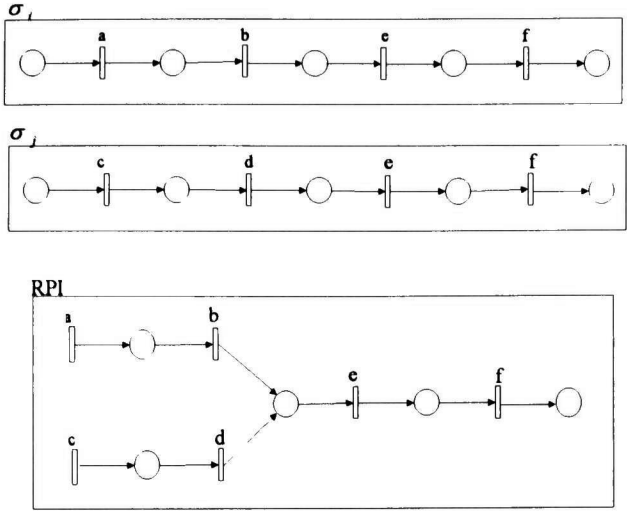


Figura 4.7: Sufijos

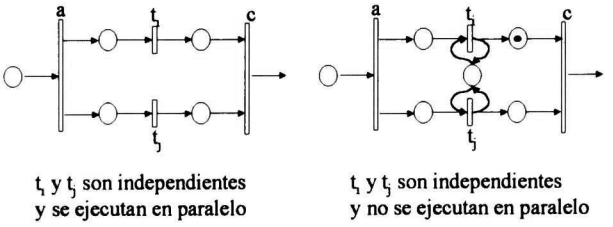


Figura 4.8: Dependencia de transiciones

then { $C(\cdot, t_{i+1}) = M' - M;$

$i = i + 1;$

}

$M = M'$

if (t_i y t_{i-1} forman bloques independientes en C)

then

Agregar un lugar l_k con $\varphi(l_k) = \epsilon$

end loop

end Forma_secuencia

Como se ve, el algoritmo agrega un lugar no medible entre transiciones consecutivas que forman bloques independientes. El siguiente ejemplo muestra cómo funciona el algoritmo. Al final, las secuencias generadas por el algoritmo son confundidas en una sola que acepta ambas secuencias.

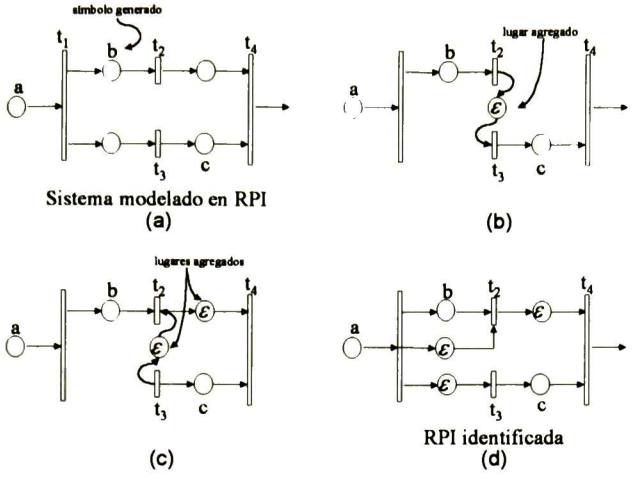


Figura 4.9: Unión de dos secuencias

Ejemplo 4.4 *Suponer el sistema modelado en RPI de la figura 4.9 (a), donde se muestran los lugares medibles a, b y c, si se observa como salida la secuencia abc, se obtiene la RPI de la figura 4.9 (b).*

En cambio si la secuencia que se observase fuese acb, la RPI que se identificaría sería la de la figura4.9 (c).

Al confundir ambos modelos en uno sólo, basta con observar las transiciones t_i, t_j que están en contradicción en ambas secuencias (que en una secuencia t_i preceda a t_j y en otra al es al revés). En este caso t_2 y t_3 están en contradicción. Basta con recorrer lugares agregados a las transiciones predecesoras de ti y tj. Si no tienen transiciones predecesoras, entonces el lugar es eliminado (ver proposición 4.6. En el ejemplo que se ha presentado, el resultado de esta operación es la RPI de la figura 4.9 (d).

Después de estos análisis, el problema que falta resover para poder construir un algoritmo de identificación, es detectar cuándo se debe empezar a identificar otra palabra. Esto se logra viendo cuando se repitió un marcado. La siguiente proposión resume este caso.

Proposición 4.8 *Cada vez que un conjunto de símbolos de salida se repite, un t-semiflujo*[2] *sobre la parte conocida (en un principio sólo es la medible) se realiza en la RPI.*

Demostración. Si se observa la siguiente palabra $\varphi(M_i)\varphi(M_j)....\varphi(M_l)....\varphi(M_i)$, entonces existe una secuencia σ_o tal que $(M^i_{conocida} \xrightarrow{\sigma_o} M^i_{conocida})$ entonces σ_o es un t-semiflujo sobre la parte conocida. La demostración es clara:

$$\begin{bmatrix} M^i_{conocida} \\ M^i_{no\_conocida} \end{bmatrix} \xrightarrow{t-semiflujo_{med}} \begin{bmatrix} M^i_{conocida} \\ M^i_{no\_conocida} \end{bmatrix}; \begin{bmatrix} M^i_{conocida} \\ M^i_{no\_conocida} \end{bmatrix} =$$

$$\begin{bmatrix} M^i_{conocida} \\ M^i_{no\_conocida} \end{bmatrix} + \begin{bmatrix} C_{conocida} & C_{no\_conocida} \end{bmatrix} \begin{bmatrix} \vec{\sigma_o} \\ \vec{\sigma} \end{bmatrix} \longrightarrow$$

[2] No necesariamente es un T-semiflujo de la RP, pero lo llamamos así porque regresa al mismo marcado sobre la parte conocida.

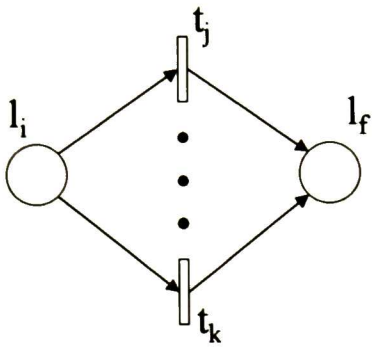


Figura 4.10:

$M^i_{conocida} = M^i_{conocida} + C\vec{\sigma_o} \longrightarrow C\vec{\sigma_o} = 0 \longrightarrow \sigma_o$ es un t-semiflujo sobre la parte conocida

■

Nótese que la parte conocida se incrementa conforme se identifiquen nuevos lugares y marcados..

Antes de seguir conviene hacer una reflexión. Si en la RPI existen transiciones $\{t_i, t_j..., t_k\}$ con el mismo símbolo de entrada $a \in \Sigma_{entrada}$ con las mismas entradas y salidas ver figura 4.10, entonces el sistema no puede ser identificado adecuadamente (aunque el sistema sea observable), ya que no se sabría que transición se dispara cuando la marca pasa de l_i a l_f. De igual forma la RPI no puede tener lugares $\{l_i, l_j..., l_k\}$ con la misma etiqueta asociada y las mismas entradas y salidas, porque nunca se sabría, por medio de las entradas y salidas, cuántos estados tiene el sistema.

Dicho de otra forma, se está especificando la minimalidad del aceptor en RPI de un lenguaje.

Definición 4.3 *Un aceptor en RPI de un lenguaje $\mathcal{L}$ es aquel que genera un mínimo grafo de alcanzabilidad (esta es la minimalidad de [3]) y tiene el menor número de lugares y transiciones (esta parte es para eliminar transiciones by pass y lugares implícitos).*

En base a las proposiciones anteriores surge el siguiente algoritmo de identificación. En forma resumida se realiza lo siguiente.

Identificación_3
Output: Modelo del Sistema
loop
While el estado del sistema no cambie; **end while**;
Calcular la transición que se disparó y lugares que cambiaron /* Algoritmo Identificación_1 */
Formar la secuencia y agregar luagres y transiciones nuevos RPI /*Algoritmo Forma_secuencias
Detectar cuando se ha realizado un t-semiflujo /* Proposición previa */
Fusionar la secuencia y la RPI que ya se había identificado /* Paso a paso se forma la RPI */
end loop;

Más detalladamente, este algoritmo se reescribe como se muestra a continuación.
Algorithm Identificación_3
Input: ninguna
Output: Modelo del sistema
Inicialización
/ Esta etapa pone a condiciones iniciales todas las variables del sistema. El marcado inicial es el medido por los sensores. */*

- Leer los símbolos de salida como el vector v'_0.
- Asociar el lugar l_i al símbolo de salida $v'_0(i) \in \sum_{salida}$, $L_0 = \{li\}$.
- Construir para v'_0, $M_0(l_i) = \begin{cases} 1 \text{ si } v'_0(i) \in \sum_{salida} \\ 0 \text{ otro caso} \end{cases}$
- Asociar el lugar l_{act_i} al actuador act_i. $M_0(l_{act_i}) = 1$. $L_0 = L_0 \cup \{ l_{act_i} \}$.
- $v_0 = \begin{bmatrix} v'_0 \\ \epsilon_1 \\ \dots \\ \epsilon_k \end{bmatrix}$ donde ϵ_i corresponde al estado inicial del actuador i (l principio se supone desconocido).
- $lista = v_0$.
- $RPI_0 = (L_0, T_0, C_0, M_0)$, donde $T_0 = \emptyset$ y $C_0 = \begin{bmatrix} C^0_{medible} \\ C^0_{no\_medible} \end{bmatrix}$ es la matriz de incidencia sin elementos.
- $i = 1$.

/ es prácticamente el algoritmo Identificación_2 */*

1. **while** siguiente_estado= v_{i-1}**y** no se presente algún símbolo de entrada;. */*detectando cuándo cambian las palabras de entrada o de salida*/*

   / Ahora se utiliza la propiedad de que el estado del actuador se conoce viendo el útimo símbolo de entrada */*

2. s^j_i *es el símbolo de entrada asociada al actuador* j *(pudiera ser* ϵ si ningún símbolo de entrada es dado*). Si* $s^j_i \neq \epsilon$ entonces $v_i = \begin{bmatrix} siguiente\_estado \\ v_{i-1}(estado\_Actuador_1) \\ estado\_Actuator_j = s^j_i \\ v_{i-1}(estado\_Actuator_k) \end{bmatrix}$ si no $v_i = \begin{bmatrix} siguiente\_estado \\ v_{i-1}(estado\_Actuator_1) \\ \vdots \\ v_{i-1}(estado\_Actuator_k) \end{bmatrix}$

   / Se utiliza el hecho de que son RPI binarias, si un símbolo nuevo aparece, un nuevo lugar debe ser agregado */*

3. Construir el marcado M_i para v_i. El marcado M_i se construye igual que M_0, pero si $s_i \neq \epsilon$ se debe generar un estado $[s_i]$ y se debe definir el marcado $M_i([s_i]) = 1$ para s_i. */* *buscando los marcados medibles* */*

4. $L_i = L_{i-1} \cup \{l_j | v_i(j) \in \sum_s\} \cup \{[s_i] || \text{ si } s_i \neq \varepsilon\}$, donde l_j son los lugares asociados con los nuevos símbolos $v_i(j)$.

   / *El hecho de que una transición es calculada como una diferencia de marcados es utilizada* */*

5. $t_i = M_i - M_{i-1}$ (agregar ceros a M_{i-1}para hacer compatibles las dimensiones)

   / *Sólo agrega transiciones nuevas cuando éstas no habían sido obtenidas anteriormente* */*

6. **if** t_i no es un columna de $C^{i-1}_{observable}$ */* *construyendo la IPN* */*

   then

$T_i = T_{i-1} \cup \{t_i\}$, agregar a C la columna t_i, donde t_i es la transición asociada con el nuevo símbolo s_i.

/ *Se genera la RPI a cada paso, presentando la matriz calculada y agregando cero a aquellas partes que todaví no se conocen* */*

1. (a) construir la $RPI_i = (L_i, T_i, C_i, M_i)$
   si $C[i, j]$ no está definido, entonces $C[i, j] = 0$.

   else

   t_i es la posición de t_i en la matriz C.

/ *Ahora se busca cuando el marcado se repite, en este caso se dice que un t-semiflujo medible ha sido encontrado y por tanto se toma una nueva palabra de entrada* */*

1. **if** $v_i \in lista$

   then

   (a) hacer $nueva\_lista = v_i, t_k, v_j, ...$ a partir de v_i hasta el final de $lista$
   $lista = lista - lista\_nueva$
   $lista\_nueva = lista\_nueva + ti, vi$
   $restricciones\_de\_secuencia(RPI_i, list\_nueva)$ */* *Se pasa al algoritmo de hacer secuencias, para unir la nueva secuenca observada con las anteriores* */*
   / *En este punto se muestra la RPIi que es la identificada en este paso* */*
   mostrar la RPI^* **agregando los arcos de** t_i **a** v_i **que habían sido borrados en la función** *restricciones_de_secuencia*
   agregar v_i a $lista$
   $i = i + 1$

   else

(a) agregar t_i (la transición disparada) y v_i *lista.*
$i = i + 1$

2. **goto 1**

/ Ahora se presenta la parte de formación de secuencias. cada vez que una nueva secuencia es detectada, ésta se analiza y se representa como RPI. Entonces esta nueva RPI debe ser unida a la RPI global para formar un nuevo modelo */*

restricciones_de_secuencia(RPI_i, *lista_nueva*)

Input: RPI, la RP identificada hasta el momento
lista_nueva es la nueva secuencia observada

Output: RPI, es la RPI que integra la RPI de entrada y la secuencia en lista_nueva

/ Primero quita los arcos del ciclo detectado, esto es para que el algoritmo no se quede ciclando */*

1. construir una $RPI^* = (L^*, T^*, C^*, M_0)$ donde $L^* = L_i, T^* = t_i$,

$$C^*(l_j, t_k) = \begin{cases} C_{RPI}(l_j, t_k) \text{ if } t_k \neq t_i \\ 0 \text{ if } t_k = t_i \text{ and } v_i(l_i) \neq 0 \end{cases}$$

/ Se analizará elemento a elemento de la lista para ver si alguno de ellos aparece en diferente orden en alguna de las secuencias previas */*

2. **while** *nueva_lista* $\neq t_i, v_i$

(a) t_1 es la primera transición que aparece en *lista_nueva.*
borrar de *lista_nueva* todos los elementos hasta t_1 (incluyendo t_1).
t_2 es la primera transición que sigue de t_1 en *lista_nueva*
/ Se verifica si ya existía una secuencia en la RPI donde t2 esté antes que t1 */*
if en la RPI^* t_1 no está en secuencia con t_2 (es decir, que no exista una trayectoria de t_1 a t_2)
then

i. **if** existe una restricción en la RPI^* tal que t_2 esté antes que t_1
then */* si t2 está antes que t1 entonces se borran autolazos creados artificialmente en pasos anteriores */*

A. borrar los arcos que hacen que t_2 ocurra antes que t_1 y reconectar los lugares $l_k \in L_u$ que hacen que t_2 esté antes que t_1 como $l_k^\bullet = t_1$ and ${}^\bullet l_k =^e ({}^\bullet t_2)$ y recalcular C

else
*/*Si t1 no queda como una transición sin salidas y no está t2 antes que t1, entonces se agrega un lugar de t1 a t2 y además un autolazo de las salidas de t1 hacia t2, con el fin de garantizar las secuencias*/*
if $t_1^\bullet \neq \emptyset$
then

A. agregar a la RPI^* un autolazo de $t_1^\bullet$ a t_2 (una vez que un autolazo ha sido borrado, nunca más se vuelve a agregar) y
if t_2 fue una transición fuente
then
agregar un lugar l_{12} entre t_1 y t_2
else
agregar un lugar l_{12} entre t_1 y t_2, donde ${}^\bullet l_{12} = t_1$ y $l_{12}^\bullet = t_2$.
hacer $L_u = L_i \cup l_{12}$, recalcular C

3. **end while**

Aunque los modelos obtenidos con la metodología del capítulo 3 resultan no observables (ver ejemplo 3.2), el algoritmo propuesto si los puede identificar, porque la proposición 4.4 establece como se puede conocer el estado del actuador a partir de su entrada y esto hace el algoritmo.

Salvo estos casos el algoritmo sólo puede identificar sistemas observables, además el sistema debe ser representable en RPI binarias. Estas condiciones se introdujeron en las pruebas de propiedades que hacen que el algoritmo funcione.

Este algoritmo identifica (al igual que el de Hiraishi) aceptores I-reversibles, este hecho se introduce en la proposición 4.8, que implica que diferentes secuencias con el mismo vector de Parikh lleven al mismo marcado.

Finalmente, el algoritmo realiza una identificación asintótica, trabaja conforme el sistema evoluciona, a diferencia de los existentes. Otra diferencia es que sólo requiere que el sistema sea observable (el resto requiere que el sistema sea completamente medible).

Paso a paso el algoritmo se parece más al sistema, dado que va incluyendo las cadenas que observay que los anteriores modelos no contemplan. El hecho de que se creen t-semiflujos medibles, que no necesariamente existen el la red hace que el sistema acepte más palabras que las que tiene el sistema en realidad, sin embargo, el modelo obtenido acepta las palabras observadas.

4.5 Resumen de las características de los tres algoritmos de identificación presentados

A continuación se presenta en forma tabular las principales características de los algoritmos presentados. Como se sabe todos ellos requieren que la entrada sea de excitación persistente y trabajan con RP binarias, invariantes y vivas.

Algoritmo 1	Algoritmo 2	Algoritmo 3
Fuera de línea	En línea	En línea
Todos los lugares medibles	Todos los lugares medibles	Puede tener lugares no medibles, pero el sistema debe ser observable
Genera el lenguaje observado	Genera el lenguaje observado	Genera más que el lenguaje observado

Capítulo 5

Ejemplo

Resumen: En este capítulo se presenta el ejemplo de un sistema a identificar con el algoritmo propuesto.

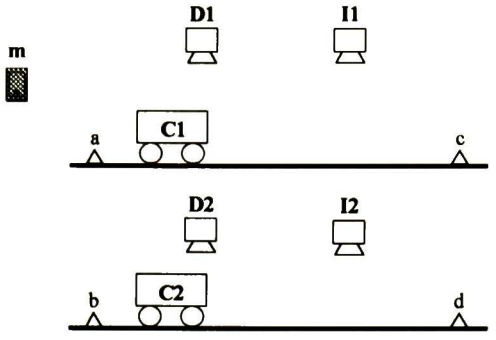


Figura 5.1: Sistema de los carros con todos los movimientos medibles

El sistema a identificar es una variante de los carritos presentados en 2.4.2.

Ejemplo 5.1 *Cuando se pulsa el botón* m, *los carros C1 y C2 parten de los puntos a y b respectivamente, cuando un carro llega a su extremo derecho (c o d), inicia su recorrido a la izquierda. Una vez que los carros se encuentran en sus posiciones iniciales, esperan a que el botón* m *sea pulsado de nuevo.*

Además, el sistema cuenta con 4 sensores, éstos detectan el movimiento a la derecha o izquierda de los carros. El sistema se presenta en la figura 5.1

Cada vez que el algoritmo lee un símbolo nuevo, le asocia un lugar y calcula el marcado, la transición con que cambió y construye la matriz C y RPI. Cuando identifica una palabra, es decir cuando se repite un símbolo de salida, el algoritmo entra a la función *restricción_ de_ secuencia* y agrega arcos y lugares no medibles para preservar el orden de aparición de los símbolos de salida.

- Si la secuencia observada es $\begin{bmatrix} D1 \\ D2 \end{bmatrix}\begin{bmatrix} D2 \\ I1 \end{bmatrix}\begin{bmatrix} I1 \\ I2 \end{bmatrix}[I2]\,[\epsilon]\begin{bmatrix} D1 \\ D2 \end{bmatrix}$ la RPI que se identifica en la primera parte del algoritmo es la de la figura 5.2. Como se ve en esta RPI el orden en que pueden aparecer los símbolos puede no ser el que se observó. Cuando el algoritmo entra a la función *restricción_ de_ secuencia* (la segunda parte del algoritmo) se añaden lugares y arcos para restringir el orden de los símbolos de salida observados. Los arcos y los lugares que se añadieron se agregan a la matriz C. (Ver figura 5.3)

- El algoritmo continua leyendo símbolos y construyendo la RPI, y se encuentra una palabra, por ejemplo $\begin{bmatrix} D1 \\ D2 \end{bmatrix}\begin{bmatrix} D1 \\ I2 \end{bmatrix}\begin{bmatrix} I1 \\ I2 \end{bmatrix}[I2]\,[\epsilon]\begin{bmatrix} D1 \\ D2 \end{bmatrix}$ en este caso no se agregan nuevos lugares porque no hay símbolos nuevos y por lo tanto no hay nuevos lugares medibles ni transiciones. Después de entrar a la función *restricciones_ de_ secuencia*, se obtiene la RPI de la figura 5.4 en donde se han borrado arcos para permitir que ocurran las dos secuencias observadas de símbolos de salida. Esto se presenta cuando existe paralelismo en el sistema.

- El sistema continua operando y ahora se observa la palabra $\begin{bmatrix} D1 \\ D2 \end{bmatrix}\begin{bmatrix} D2 \\ I1 \end{bmatrix}[D2]\,[I2]\,[\epsilon]\begin{bmatrix} D1 \\ D2 \end{bmatrix}$ la RPI que se obtiene es la de la figura 5.5, en este caso sólo se borraron los arcos que no permitían representar el comportamiento observado.

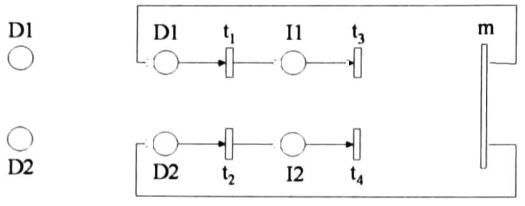


Figura 5.2:

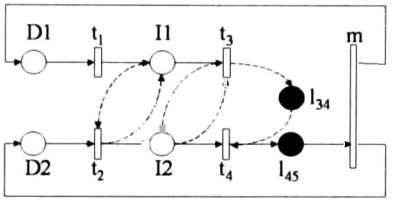


Figura 5.3:

- A continuación se observa la palabra $\begin{bmatrix} D1 \\ D2 \end{bmatrix} \begin{bmatrix} D1 \\ I2 \end{bmatrix} [D1]\,[I1]\,[\epsilon] \begin{bmatrix} D1 \\ D2 \end{bmatrix}$ después que se entra a la función *restricciones_de_secuencia* se obtiene la RPI de la figura 5.6, en la cual se ha agregado un lugar, que se incluye en la matriz C.

- Para las siguientes palabras observadas, la RPI ya no cambia, porque éstas ya no aportan más información del comportamiento del sistema. Por ejemplo las palabras $\begin{bmatrix} D1 \\ D2 \end{bmatrix} \begin{bmatrix} D2 \\ l1 \end{bmatrix} \begin{bmatrix} I1 \\ I2 \end{bmatrix} [I1]\,[\epsilon] \begin{bmatrix} D1 \\ D2 \end{bmatrix}$ y $\begin{bmatrix} D1 \\ D2 \end{bmatrix} \begin{bmatrix} D1 \\ I2 \end{bmatrix} \begin{bmatrix} I1 \\ I2 \end{bmatrix} [I1]\,[\epsilon]$

- En este ejemplo, con las primeras cuatro palabras, el sistema se identifica aunque el algoritmo no para.

Las seis palabras anteriores son todas las que el sistema puede generar. Para el ejemplo, no se repitieron palabras, pero depende del comportamiento real del sistema si las palabras se repiten y el orden en que aparezcan. Si la entrada es de excitación persistente, entonces el sistema se identifica correctamente.

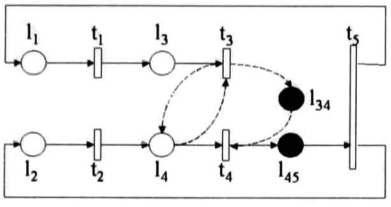


Figura 5.4:

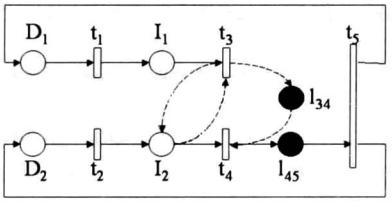


Figura 5.5:

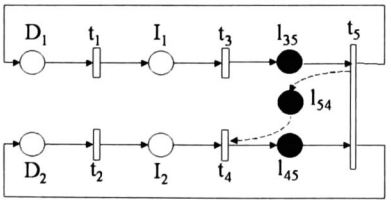


Figura 5.6:

Capítulo 6

Conclusiones

En esta tesis se presentaron tres algoritmos de identificación, cada uno con distintas características.

El primero de ellos es fuera de linea y utiliza el método de los mínimos cuadrados para calcular la matriz de incidencia C de la RPI que representa al sistema. Para utilizar este algoritmo las hipótesis de trabajo son:

1. contar con las muestras de los vectores de disparos y los marcados alcanzados con ellos
2. el orden del modelo es conocido (el número de lugares y transiciones de la RPI)
3. la RPI es totalmente observable (todos sus lugares son medibles) y totalmente controlable (todas las transiciones tienen asociadas una señal de entrada)
4. las entradas al SED son de excitación persistente para que se puedan generar todas las secuencias del sistema.

Con este algoritmo lo que se identifica son los parámetros de la RPI que son los pesos de la matriz C (la relación entre los lugares y las transiciones)

El segundo de los algoritmos es asintótico, es decir que identifica conforme el sistema evoluciona y parte de las siguientes hipótesis

1. el sistema es totalmente observable y controlable
2. la entrada debe ser de excitación persistente

Este algoritmo calcula las columnas de la matriz C al ir restando el marcado observado menos el anterior.

El tercero y último de los algoritmos identifica la estructura de una RPI binaria como modelo del SED. Este algoritmo es más general que el presentado por Hiraishi ya que identifica la estructura de un SED aún cuando éste no sea completamente medible y controlable. Una característica del algortimo es que es asintótico, ya que la estructura del SED en RPI se obtiene conforme éste evoluciona, las hipótesis de las que parte este algoritmo son las siguientes

1. el sistema no es totalmente medible, pero si es observable

2. la entrada es de excitación persistente.

Se presentó una metodología de modelado para SED´s en la que el modelo global del SED es la unión del modelo interno del sistema y de los modelos de los actuadores.

También se estudió la observabilidad y controlabilidad de SED's, en este caso se hizo como se estudian en los sistemas continuos. Desde esta perspectiva, estas nociones resultan más claras y simples que desde el punto de vista de lenguajes. Tal vez se deba a que las RPI representan el comportamiento interno, mientras que el lenguaje el comportamiento entrada/salida, pero esto es sólo una suposición y hay que estudiarlo más a fondo.

Todavía queda mucho por hacer, como trabajo futuro se tiene la extensión de los resultados a redes que no sean binarias e incluir la prealimentación. Este sería el caso más general, abordarlo parece una tarea difícil. Otro trabajo sería que el algoritmo quedara especificado en RP's, así se lograría tener una única herramienta para el modelo y para el algortimo.

Por último la implementación de los algortimos en dispositivos de estado sólido y su utilización como modelos a controlar también resulta necesarios.

Bibliografía

[1] MuraCüneyt M. Özveren and Alan S. Willsky. *Observability of Discrete Event Dynamic Systems.* IEEE Transactions on Automatic Control, Vol. 35, No. 7, July 1990, pp. 797-806.

[2] Yong Li and W.M. Wonham. *Controllability and Observability in the state-feedback Control of Discrete-Event Systems.* Proceedings of the 27th Conference on Decision an Control. Austin, Texas, December 1988, pp. 203-208

[3] Dana Angluin. *Inference of Reversible Languages.* Journal of the Association for Computing Machinery, Vol. 29, No. 3, July 1982, pp. 741-765

[4] Manuel Silva. *Las Redes de Petri: en la Automática y la Informática.* Editorial AC. 1985

[5] John E. Hopcroft and Jefrey D. Ullman. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley publishing company. 1979

[6] E. Mark Gold. *Complexity of Automaton Identification from Given Data.* Information and Control, Vol. 37, pp. 302-320. 1978

[7] Kunihico Hiraishi. *Construction of Safe Petri Nets by Presenting Examples*

[8] W. M. Wonham. *Notes on Control of Discrete-Event Systems.* Systems Control Group, Dept. of Electrical & Computer Engineering. University of Toronto, 1996-97.

[9] Pieter Eykhoff, Systems Identification, editorial John Wiley & sons Inc., 1974

[10] Jonathan S. Ostroff, Temporal Logic for Real -Time Systems, editorial John Wiley & sons Inc., 1989.

[11] M.E.Meda & A. Ramírez, Identification in Discrete Event Systems. IEEE-SMA 1998.

[12] T. Murata, Petri Nets Properties, Analysis and Synthesis of Marked Graphs. Proceedings de la IEEE 77(4), pp. 541-580, (1989).

[13] Goodwin & Payne. Dynamic System, Identification, Academic Press, 1977.

[14] Tadao. Murata, Synthesis of Decision-free Concurrent Systems for Prescribed Resources and Performance, IEEE Trans. on Software Engineering, Vol. SE-6, No. 6, 525/530, 1980.

[15] Yu-Chin Ho, Discrete Event Dynamic Systems, ed. IEEE Press, 1992.

[16] Yu-Chin Ho y Xi Ren Cao, Perturbation Analysis of Discrete Event Dynamic Systems, Engineering and Computer Science, Kluwer Academic Publishers, USA, 1991.

[17] Alessandro Guia, Petri Nets as Discrete Event Models for Supervisory Control, Tesis Doctoral, Rensselaer Polytechnic Institute. Troy, New York, 1992.

[18] Antonio Ramírez Treviño, Scheduling en Redes de Petri, Tesis Doctoral, Dpto de Ingeniería Eléctrica e Informática, Centro Politécnico Superior de Ingenieros, Universidad de Zaragoza, 1993.

[19] M. Jantzen, Language Theory of Petri Nets, Petri Nets: Central Models and their Properties, Advances in Petri Nets 1986, W. Brauer, W Reising and G. Rosenberg (eds), Lecture Notes in Computer Science, Vol 254-I, pp. 397-412, Springer.Verlang, 1987.

[20] J. L. Peterson, Petri Net Theory and the Modeling of Systems, Pretince-Hall, 1981.

[21] Jörg Desel and Javier Esparza, Free Choice Petri Nets, ed. Cambridge University Press, 1995

APENDICE 1.

Autómata finito

Sea $\Theta = (Q, \Sigma, \delta, q_0, F)$ un autómata finito

Q es el conjunto de estados

Σ es el alfabeto de entrada

δ es la función de transición de estados $\delta : Q \times \Sigma \rightarrow E$

$q_o \in Q$ es el estado inicial

F es el conjunto de estados finales $F \subseteq Q$

Un autómata 0-reversible (ZR)

Sea $\Theta = (Q, \Sigma, \delta, q_0, F)$ un autómata finito definido de la forma anterior y Θ^{-1} el autómata inverso de Θ, donde cambia la orientación de los arcos de Θ, un autómata Θ es ZR ssi Θ^{-1} es determinista (ver **??**)

Un autómata es I-reversible si

1. $|F| = 1$ y
2. Θ es consistente invariante

Un lenguaje L es llamado I-reversible si existe un autómata I-reversible tal que $L = L(\Theta)$

Autómata Consistente Invariante

Un autómata Θ definido de la forma anterior, se dice que es consistente invariante si las secuencias que salen de un mismo estado y tienen el mismo vector de Parikh llevan a un mismo estado. Es decir que para que un autómata sea de consistencia invariante debe cumplir con lo siguiente

- las secuencias σ_1 y $\sigma_2 \in \sigma_i$ (secuencias que salen del estado q_i) que tienen el mismo vector de Parikh $\overrightarrow{\sigma_1} = \overrightarrow{\sigma_2}$ deben alcanzar el mismo estado $q_1 = q_2$ donde $q_1 \in \delta(q_i, \sigma_1)$ y $q_2 \in \delta(q_i, \sigma_2)$. Ver [[7]].

Lenguajes regulares

Los lenguajes aceptados por autómatas finitos son fácilmente descritos por simples expresiones llamadas expresiones regulares, por lo que la clase de lenguajes aceptados por autómatas finitos son precisamente la clase de lenguajes que se pueden describir con dichas expresiones. Las operaciones de unión, concatenación y cerradura de Kleene definen expresiones regulares.

Sea Σ un conjunto finito de símbolos y sean L_1 y L_2 conjuntos de cadenas que pertenecen a Σ^*. La concatenación de L_1 y L_2 denotado por L_1L_2, es el conjunto $\{xy \mid x$ está en L_1 y y está en $L_2\}$. Esto es que las cadenas de L_1L_2 están formadas por una cadena de L_1 seguida

de una cadena de L_2, en todas las posibles combinaciones. Se define $L^0 = \{\epsilon\}$ y $L_i = L_{i-1}$ para $i \geq 1$. La cerradura de Kleene de L, denotada por L^*, es el conjunto

$$L^* = \cup_{i=0}^{\alpha} L_i$$

y la cerradura positiva L^+ es el conjunto

$$L^+ = \cup_{i=1}^{\alpha} L_i.$$

L^* denota las palabras construídas al concatenar cualquier número de palabras de L. L^+ es muy similar sólo que excluye el caso de cero palabras cuya concatenación es ϵ.

Sea Σ un alfabeto. Las expresiones regulares sobre Σ y los conjuntos que denotan se definen recursivamente como sigue:

1. $\emptyset$ es una expresión regular y denota el conjunto vacío
2. ϵ es una expresión regular y denota el conjunto $\{\epsilon\}$
3. Para cada a en $\sum$, $\mathbf{a}$[1] es una expresión regular y denota el conjunto $\{a\}$
4. Si r y s son expresiones regulares que representan a los lenguajes R y S respectivamente, entonces $(r+s)$, (rs) y (r^*), son expresiones regulares que representan a los conjuntos $R \cup S$, RS y (R^*); por lo tanto el resultado de las operaciones de unión, concatenación y cerradura de Kleene con expresiones regulares, también son expresiones regulares.

[1] Cuando un símbolo es parte de una expresión regular, éste se escribe en negritas. a y $\mathbf{a}$ son el mismo símbolo, lo que significa que cada símbolo de un alfabeto es una expresión regular.

CENTRO DE INVESTIGACION Y DE ESTUDIOS AVANZADOS DEL IPN
UNIDAD GUADALAJARA

El Jurado designado por el Laboratorio de Ingeniería Eléctrica y Ciencias de la Computación del Centro de Investigación y de Estudios Avanzados del Instituto Politécnico Nacional, aprobó la tesis: **Identificación de Sistemas Dinámicos de Eventos Discretos utilizando Redes de Petri** el día 09 de Octubre de 1998.

El jurado:

Dra. Ofelia Begovich Mendoza
Profesora Titular 3A
CINVESTAV DEL IPN
Guadalajara

Dr. Manuel Edgardo Guzmán Rentería
Profesor Titular 3A
CINVESTAV DEL IPN
Guadalajara

Dr. Luis Ernesto López Mellado
Profesor Titular 3A
CINVESTAV DEL IPN
Guadalajara

Dr. Arturo Veloz Guerrero
Profesor Titular 3A
CINVESTAV DEL IPN
Guadalajara

PROL. LOPEZ MATEOS SUR 590 APARTADO POSTAL 31-438 GUADALAJARA, JAL. 45090 TEL. (3) 684-1580 FAX (3) 684-1708