Ecuación de estado generalizada para redes de Petri no autónomas y con distintos tipos de arcos

Dr. Ing. Orlando Micolini[1], Geol. Marcelo Cebollada y Verdaguer[1], Ing. Maximiliano Eschoyez[1], Ing. Luis Orlando Ventre[1], Ing. Marcelo Ismael Schild[1].

Laboratorio de Arquitectura de Computadoras (LAC) FCEFyN
Universidad Nacional de Córdoba
{orlando.micolini, marcelo.cebollada.y.verdaguer, maximiliano.eschoyez, luis.ventre, marcelo.schild }@unc.edu.ar

Abstract. Este trabajo propone generalizar la ecuación de estado de las redes de Petri, para representar redes de Petri no autónomas con distintos tipos de brazos y semánticas temporales. Esta ecuación generalizada facilita la implementación por hardware de un IP-Core para la ejecución de la red. La solución expresada por esta ecuación de estado da origen a un algoritmo que preserva el modelo original, facilita su ejecución en paralelo y permite abordar problemas de mayor tamaño y complejidad. Además, se aborda un caso donde se pone de manifiesto las ventajas de incluir eventos, distintos tipos de brazos y semánticas temporales.

Abstract. This work proposes to generalize the state equation of a Petri Net, with the objective of represent different types of arcs and time semantics in non-autonomous Petri nets. This generalized equation facilitates the hardware implementation of an IP-Core to execute the network.
The solution that these extended state equation express raises an algorithm that preserves the original model, facilitates parallel execution and allows addressing problems bigger in size and complexity. In addition, exposes a case of application where highlights the advantages of including events, different types of arcs and timed semantics.

Palabras clave: Red de Petri no autónomas, Ecuación de estado, Procesador de Petri, IP-Core.

Keywords: non-autonomous Petri net, state equation, Petri Processor, IP-Core.

1 Introducción

La implementación de hilos en los sistemas informáticos permite explotar los recursos de las arquitecturas multicore [1]. Estos hilos cooperan y se ejecutan concurrentemente. Las aplicaciones diseñadas de esta manera tienen una complejidad intrínseca adicional con respecto a los programas secuenciales. Esta complejidad se manifiesta en el diseño, detección de errores, testing, validación y mantenimiento [2]. Por lo mencionado, es necesario introducir mecanismos de control, como los semáforos, que

penalizan los tiempos de ejecución. Por todo esto es conveniente generar una solución formal que garantice y facilite el desarrollo e implementación del sistema.

Investigaciones recientes mostraron que los modelos obtenidos con redes de Petri (RdP) facilitan la implementación de sistemas de manera directa, utilizando software, procesadores o IP-Core que ejecutan RdP [3, 4]. Procesadores de RdP no-autónomas (PP) se presentaron en [5, 6] como IP-Cores implementados en una FPGA Spartan-6.

La ejecución y simulación de sistemas complejos requieren modelado de RdP con mayor semántica. Para ampliar la capacidad semántica de las redes se incluyen eventos, guardas, distintos tipos de brazos (inhibidores, lectores, etc.) y semánticas temporales [7]. Esto amplía el dominio de los problemas, el tamaño y la complejidad.

Si bien en [8] se presenta una extensión de la ecuación de estado, los autores no consideran eventos, guardas y tiempo. Además, introducen variables en la matriz de incidencia lo que dificulta la implementación como circuito combinacional.

En el presente trabajo se generaliza la ecuación de estado de la RdP, usando el concepto de inhibición de disparos, para obtener las distintas ampliaciones semánticas e implementarlas en un sistema combinacional. De esta forma, se aprovechan más efectivamente los recursos de las FPGA, manteniendo todas las propiedades de las RdP temporales y no autónomas.

2 Fundamentos y Notación

2.1 Ecuación de estado

Las RdP se utilizan para representar gráficamente el comportamiento dinámico de un sistema. Por lo tanto, las características gráficas de las RdP pueden explotarse para inspeccionar la dinámica de un sistema. Este enfoque es adecuado para sistemas pequeños. Sin embargo, la metodología gráfica no es eficiente cuando los sistemas son grandes y complejos. En este artículo, desarrollamos una ecuación de estado para representar modelos de estos sistemas e implementarlos como IP-Core.

Es importante introducir la ecuación de estado de las RdP para extenderla con distintos tipos de brazos, eventos, guardas y tiempo. Con esta ecuación es posible obtener el siguiente estado del sistema. Esta es una manera más simple que la metodología gráfica para analizar la evolución de los sistemas.

La ecuación de estado de una RdP, con n plazas y m transiciones, con brazos con peso mayor o igual a uno y marca inicial M_0, es:

$$M_{j+1} = M_j + I * \sigma. \quad \textbf{(1)}$$

Siendo: I matriz de Incidencia, con dimensión $n \times m$, σ el vector disparo, con dimensión $m \times 1$, M_0 el vector de marcado inicial, M_j el vector de marcado actual, M_{j+1} el vector de marcado del estado siguiente, todos con dimensión $n \times 1$.

Los elementos i_{ij} de la matriz I se obtienen,

$$i_{ij} = w(t_i, p_j) - w(p_j, t_i). \quad \textbf{(2)}$$

Donde w son los pesos de los arcos (valores enteros con signo):

- Los arcos de la plaza-i a la transición-j, son $-w(p_j, t_i)$.
- Los arcos de la transición-j a la plaza-i, son $w(t_i, p_j)$.

Cuando se dispara una transición sensibilizada, se puede calcular el siguiente estado usando (1).

La ecuación de estado representa matemáticamente el comportamiento dinámico del sistema. La matriz de incidencia caracteriza el comportamiento del sistema. Por lo tanto, la matriz de incidencia representa el sistema en sí.

2.2 Transición sensibilizada

Una transición está sensibilizada si todos los lugares de entrada a la transición tienen una marca igual o mayor al peso del arco que une cada plaza con la transición. Lo que se expresa como:

$$M(p_i) \geq w(p_j, t_i), \forall p_j \in I(t_i).$$

Las transiciones sensibilizadas se expresan como un vector binario E de dimensión $m \times 1$. Cada componente del vector indica con uno la transición sensibilizada y un cero la que no.

$$E = \left(sign(S^0), sign(S^1), \dots, sign(S^{n-1})\right).$$

Donde la relación $sign(S^i)$ de cada vector S^i es:

- S^i es igual a $M_j + I^i$, un vector con el estado que se obtendría de disparar la i-esima transición una vez.
- $I^0, I^1, \dots, I^{n-1}$, son las columnas de la matriz I
- $sign(S^i)$, es binario siendo cero si alguna de las componentes de S^i es negativa, de otra forma es uno.

2.3 Disparo de una transición

Una transición se puede disparar solo si esta sensibilizada. Cuando se dispara se calcula el nuevo estado M_{j+1} y el nuevo vector de transiciones sensibilizadas E.

2.4 Interpretación de la matriz de incidencia

De la semántica de disparo de una RdP, se interpreta la matriz de incidencia como la evaluación conjuntiva entre las columnas (transiciones) y las restricciones que imponen las filas (plazas). Es decir, en una matriz de incidencia de dimensión $m \times n$ se evalúan n combinaciones de m variables lógicas, como se expresa en la siguiente ecuación:

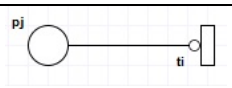
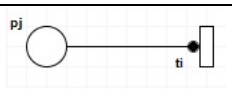
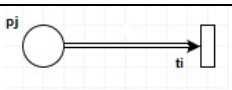
$$e_i = \left(\bigwedge_{h=0}^{m-1} M(p_h) \geq i_{hi} \right), \forall i = 0, \ldots, n-1.$$

Donde i_{hi} son los elementos de la matriz I y $M(p_h)$ es la marca en la plaza h. El resultado son las componentes del vector E.

3 Extensión de la ecuación de estado

3.1 Arcos inhibidores, lectores y reset

Los arcos inhibidores, lectores y reset conectan una plaza con una transición, aumentando la capacidad de expresión de la RdP. Cabe destacar que los arcos inhibidores y lectores posibilitan modelar prioridades.

Arco	Representación	Consecuencia sobre la semántica
inhibidor		Si $h(p_j, t_i)$ es un arco inhibidor y hay token en p_j, entonces t_i no está sensibilizada.
lector		Si $l(p_j, t_i)$ es un arco lector y no hay token en p_j, entonces t_i no está sensibilizada. El disparo de t_i no consume los token de p_j.
reset		Si $r(p_j, t_i)$ es un arco reset y se dispara t_i se consumen todos los token de p_j. El disparo de t_i pone a cero la marca de la plaza p_j.

3.2 Guardas

Una guarda es una variable lógica asociada a una transición que aumenta la capacidad de expresión. Gráficamente, la guarda se representa con una etiqueta junto a la transición. Si $g(t_i)$ es una guarda, para disparar t_i se requiere que la transición esté sensibilizada y el valor de la guarda sea verdadero. Las guardas posibilitan comunicar la RdP con el medio haciéndolas no autónoma.

3.3 Eventos

Un evento se almacena en una cola que se asocia a una transición [3], aumentando la capacidad de expresión de la RdP. Gráficamente, la cola se representa como una etiqueta junto a la transición. La cola c_i, es un contador, que se incrementa cuando llega un evento y se decrementa cuando la transición asociada se dispara. Para que la transición esté sensibilizada se requiere que la cola tenga al menos un evento. Los eventos comunican la RdP con el medio para hacerla no autónoma.

3.4 Tiempo

Hay distintas semánticas temporales, como las propuestas en [9-11]. En este trabajo hemos tomado las RdP con tiempo introducidas en [12]. Estas imponen limitaciones de tiempo sobre los disparos, como un intervalo de tiempo asociado a cada transición.

Es decir, cada transición activa tiene asociada un cronómetro implícito que mide el tiempo transcurrido desde la última vez que se sensibilizó.

Una transición habilitada puede dispararse si el valor cronometrado, asociado a la transición, está dentro del intervalo de tiempo predeterminado. Por ejemplo, la etiqueta de intervalo $[\alpha_i, \beta_i]$ asociada a t_i, tiene como instante de inicio α_i y β_i como el instante de fin.

3.5 Política de selección de disparo

Se refiere a la elección de la próxima transición a disparar de entre todas las sensibilizadas. Para una RdP, incluidas las no autónomas o temporales, si esta elección es aleatoria el sistema resulta no determinístico. Para que el sistema sea determinístico hay diferentes soluciones, como la inclusión de: prioridades, probabilidades, arcos inhibidores, arcos lectores, etc.

Con el fin de evitar conflictos en las transiciones usamos la política de servidor único. Es decir, calculamos el nuevo estado considerando solo un disparo de una transición. Si se requieren más de uno, se realiza una secuencia de disparos.

4 Ecuación de estado extendida

4.1 Consideraciones para la generalización de la ecuación

En todas las ampliaciones de la semántica expuesta, destacamos que siempre se trata de determinar cuáles transiciones están sensibilizadas. Una vez determinada la transición a disparar, el disparo se realiza con la ecuación de estado original, excepto para el brazo reset que retira todos los token de la plaza.

Para representar matemáticamente la existencia de los brazos enumerados anteriormente se requiere de una matriz para indicar la conexión plaza transición. Estas matrices son similares a la matriz *I*. Los términos de las matrices son binarios, dado que los pesos de los arcos son uno.

Es decir, para que una transición esté sensibilizada se requiere:

- Si tiene brazo inhibidor que la plaza no tenga token.
- Si tiene brazo lector que la plaza tenga uno o más token
- Si tiene guarda que el valor de la guarda sea verdadero.
- Si tiene evento que tenga uno o más eventos en la cola asociada.
- Si tiene etiqueta con intervalo de tiempo, que el contador se encuentre en el intervalo.

4.2 Nueva ecuación de estado

De las consideraciones anteriores observamos que para disparar una transición se requiere la conjunción lógica entre todas las condiciones enumeradas en el punto anterior y el vector E de transiciones sensibilizadas.

Vector de transiciones des-sensibilizadas por arco inhibidor $\boldsymbol{B}$, es un vector de valores binarios de dimensión $m \times 1$, que indica con un cero cuales transiciones están inhibidas por el arco y uno las que no, y se obtiene:

$$B = H * Q.$$

- Donde H es una matriz de dimensión $m \times n$ y Q un vector binario de dimensión $n \times 1$.
- H es la matriz de incidencia que relaciona las plazas que conectan las transiciones con un brazo inhibidor, los términos h_{ij} de la matriz son uno si hay brazo de la plaza p_i a la transición t_j y cero si no lo hay.
- Las componentes q_i del vector Q se obtienen de aplicar la relación

$$q_i = cero\big(M(p_i)\big).$$

- La relación $cero(M(p_i))$, es cero si la marca en la plaza p_i es distinta de cero, en otro caso un uno.

Vector de transiciones des-sensibilizadas por arco lector $\boldsymbol{L}$, es un vector de valores binarios de dimensión $m \times 1$, que indica con un cero cuales transiciones están inhibidas por el arco y un uno las que no, y se obtiene:

$$L = R * W.$$

- Donde R es una matriz de dimensión $m \times n$ y W un vector binario de dimensión $n \times 1$.
- R es la matriz de incidencia que relaciona las plazas que conectan las transiciones con un brazo lector, los términos r_{ij} de la matriz son un uno si hay brazo de la plaza p_i a la transición t_j y un cero si no lo hay.
- Las componentes w_i del vector W se obtienen de aplicar la relación

$$w_i = uno\big(M(p_i)\big).$$

- La relación $uno(M(p_i))$, es uno si la marca en la plaza p_i es distinta de cero, en otro caso un cero.

Vector de transiciones des-sensibilizadas por guarda G, es un vector de valores binarios de dimensión $m \times 1$, que indica con un cero cuales transiciones están inhibidas por la guarda y un uno las que no, y se obtiene directamente de las guarda.

Vector de transiciones des-sensibilizadas por evento $\boldsymbol{V}$, es un vector de valores binarios de dimensión $m \times 1$, que indica con un cero cuales transiciones están inhibidas porque no hay evento solicitando el disparo de la transición. Las componentes v_i del vector V se obtienen de aplicar la relación:

$$v_i = uno(bufferDeEventos_i).$$

La relación $uno(bufferDeEventos_i)$, es uno si hay uno o más eventos en el buffer asociado con la transición t_i, en otro caso cero.

Vector de transiciones des-sensibilizadas por tiempo Z, es un vector de valores binarios de dimensión $m \times 1$, que indica con un cero cuales transiciones están inhibidas porque no se ha alcanzado o se ha superado el intervalo de tiempo transcurrido desde de que la transición fue sensibilizada.

$$Z = Tim(q(E, B, L, G, clk), intervalos).$$

- E es el vector de sensibilizado, B es el vector de no-sensibilizado por arco inhibidor, L es el vector de no-sensibilizado por arco lector y G es el vector de no-sensibilizado por guarda.
- La relación $Tim(q, intervalos)$ es un vector binario de $m \times 1$, el valor de la componente i es uno si q_i , que es un contador, tiene valor en el intervalo indicado por la componente $intervalos_i = [\alpha_i, \beta_i]$ de otra forma es cero.
- Para que el contador q_i arranque la ecuación $e_i \; and \; b_i \; and \; l_i \; and \; g_i$ debe ser uno de otra forma el contados se pone a cero, este contador se incrementa en una unidad de tiempo por cada pulso de reloj, de la base de tiempo (clk).
- La matriz $intervalos$ es de dimensión $m \times 2$ y contiene el límite inferior el límite superior de la ventana de tiempo para el disparo.

Vector de transiciones reset A, es un vector de valores enteros de dimensión $m \times 1$, que tiene el valor de la marca de la plaza que se quiere poner a cero, mientras que las otras componentes son uno.

La matriz Re expresar matemáticamente los arcos de reset. El arco que une la plaza p_i con la transición t_j se indica con un uno en la componente re_{ij} de la matriz de otro modo es cero.

$$A = Marca(Re * M_j).$$

La multiplicación $Re * M_j$ es un vector con cero sin no hay brazo reset y el valor de la marca si hay brazo reset.

La relación $Marca$ es uno si la componente es cero en otro caso el valor de la componente.

El vector A se multiplica elemento a elemento con el vector de disparo σ, operación que indicamos con #. El vector σ tiene un uno en la transición que se quiere disparar por lo que obtenemos la cantidad de disparos necesarios para sacar todas las marcas de la plaza a resetear en otro caso solo un disparo.

4.3 Vector de sensibilizado extendido y ecuación de estado

Este vector de sensibilizado extendido Ex se obtiene de la conjunción lógica de todos los vectores anteriores.

$$Ex = E \; and \; B \; and \; L \; and \; V \; and \; G \; and \; Z.$$

Para introducir el brazo reset hay que multiplicar elemento a elemento (#) a el vector que resulte de la conjunción por A. Por lo que la ecuación de estado extendida resulta:

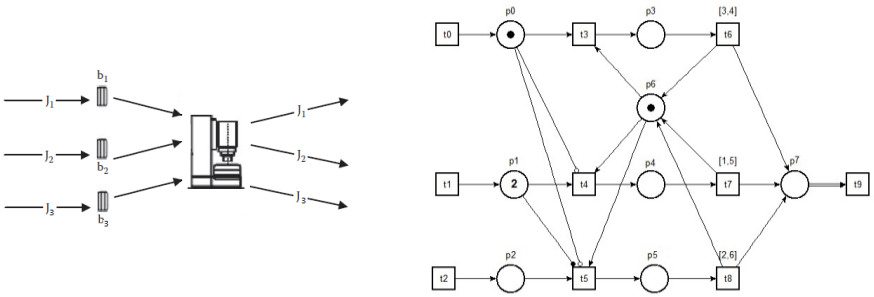
$$M_{j+1} = M_j + I * ((\sigma\ and\ Ex)\#A.$$

La importancia de esta ecuación de estado es que es de simple implementación como circuito combinacional en una FPGA.

5 Caso de Aplicación

5.1 Sistema de producción modelado con RdP extendida

La Figura 1 muestra el uso de arcos inhibidores y lectores para modelar prioridad y secuencias en un sistema de fabricación. Consideremos el proceso de fabricación representado en la Figura 1(a) donde, una máquina procesa tres tipos de trabajos. El trabajo J1 tiene una prioridad más alta que J2 y J3, y debe haber esperando al menos un J2 para procesar J3.



(a) **(b)**

Figura 1: (a) Sistema de producción y (b) la RdP que lo modela

La RdP de la Figura 1(b) muestra el modelo del sistema. En el modelo t0, t1 y t2 son las llegadas de solicitudes de trabajos J1, J2 y J3, los token en p0, p1 y p2 representan estas solicitudes. Un token en alguna de las plazas p3, p4 o p5 representa que se está realizando el trabajo. Las transiciones t6, t7 y t8 tienen asociada la etiqueta de tiempo, que es el requerido por cada trabajo y su disparo corresponde a la finalización de este. La plaza p6 representa la máquina y la plaza p7 totaliza la cantidad de trabajos realizados. El disparo de t9, dado que hay un arco reset entre p7 y t9, pone a cero la plaza p7.

A continuación expresamos las matrices y vectores del modelo de la Figura 1.

Matriz I

-1	0	0	1	0	0	0	0	0	0
0	-1	0	0	0	0	0	0	0	0
0	0	-1	0	1	1	0	0	0	0
0	0	0	-1	0	0	1	0	0	0
0	0	0	0	-1	0	0	1	0	0
0	0	0	0	0	-1	0	0	1	0
0	0	0	1	1	1	-1	-1	-1	0
0	0	0	0	0	0	-1	-1	-1	0

Matriz de brazos inhibidores B

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Matriz de brazos lectores L

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

Matriz de brazos Reset R

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	1

Matriz intervalos temporales $intervalos$		Vector inhibidos por tiempo Z	Vector de Marcado M_j	Vector de transiciones sensibilizadas E	Vector de transiciones inhibidas por arco inhibidor B	Vector de transiciones inhibidas por arco lector L	Vector de transiciones sensibilizadas extendido Ex
0	0	0	1	1	1	1	1
0	0	0	2	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	0	1	0
0	0	0	0	0	0	1	0
3	4	0	1	0	1	1	0
1	5	0	0	0	1	1	0
2	6	0		0	1	1	0
0	0	0		0	1	1	0

La implementación de esta ecuación requiere de 6 matrices. Para obtener matrices de menor dimensión, con el consiguiente ahorro de recursos, es posible aplicar técnicas de división de redes como las presentadas en [13].

6 Conclusiones

Este trabajo presentó la ecuación de estado y el vector de transición sensibilizada de RdP generalizada para múltiples tipos de arcos, eventos, guardas y tiempo. Esta ecuación y vector son de lógica conjuntiva lo que facilita la implementación con circuitos combinacionales en un IP-Core. Se ha tenido la precaución de no introducir variables en las matrices, dado que esto dificulta la implementación y validación de la ecuación de estado en el circuito combinacional.

Esta nueva ecuación de estado facilitó el desarrollo de software - hardware, que implementa y ejecuta RdP temporales y no-autónomas. Este software - hardware se

utilizó para obtener los estados del sistema y la ejecución de la lógica concurrente y paralela del programa.

La ecuación de estado generalizada se obtuvo con la conjunción de vectores. Para esto se ha obtenido un vector para cada tipo de arco, etiquetas con tiempo, eventos y guardas. Esto consiguió la modularidad del software – hardware según sea la red que se desea implementar.

Como trabajo futuro, se está diseñando un nuevo PP con arquitectura pipeline basado en esta ecuación, lo que permitirá ejecutar sistemas de mayor complejidad.

Referencias

1. David R. Martinez, R.A.B., M. Michael Vai, *High Performance Embedded Computing Handbook A Systems Perspective*2008, Massachusetts Institute of Technology, Lincoln Laboratory, Lexington, Massachusetts, U.S.A.: CRC Press.
2. Domeika, M., *Software Development for Embedded Multi-core Systems*2008, 30 Corporate Drive, Suite 400, Burlington, MA 01803, USA Linacre House, Jordan Hill, Oxford OX2 8DP, UK.
3. Micolini, O., *ARQUITECTURA ASIMÉTRICA MULTI CORE CON PROCESADOR DE PETRI*, in *Informatica*2015, UNLaP: La Plata, Argentina.
4. Moutinho, F. and L. Gomes, *Distributed Embedded Controller Development with Petri Nets: Application to Globally-Asynchronous Locally-Synchronous Systems*. Vol. 150. 2015: Springer.
5. Micolini, O., J. Nonino, and C.R. Pisetta. *IP Core Para Redes de Petri con Tiempo*. in *CASIC 2013*. 2013.
6. M. Pereyra, N.G., M. Alasia and O. Micolini, *Heterogeneous Multi-Core System, synchronized by a Petri Processor on FPGA.* IEEE LATIN AMERICA TRANSACTIONS, 2013. **11**: p. 218-223.
7. Diaz, M., *Petri Nets Fundamental Models, Verification and Applications*2009, NJ USA: John Wiley & Sons, Inc.
8. BaşkocagİI, C. and S. Kurtulan, *Generalized state equation for Petri nets.* WSEAS TRANSACTIONS on SYSTEMS, 2011. **10**(9): p. 295-305.
9. Sifakis, J., *Performance evaluation of systems using nets.* Springer-Verlag Berlin Heidelberg, 1979. **84**: p. 307-319.
10. Roussopoulos, J.E.C.y.N., *Timing requirements for time-driven systems using augmented Petri nets.* EEE transactions on Software Engineering, 1983. **9**(5): p. 603-616.
11. Jensen, K. and L.M. Kristensen, *Coloured Petri Nets Modelling and Validation of Concurrent Systems.* Springer 2009.
12. Merlin, P.M., *A Study of the Recoverability of Computing Systems*, 1974, University Microfilms, : University of California.
13. Micolini, O., M. Cebollada, and L.O. Ventre. *Localidad estructural, criterio de división para la ejecución de redes de Petri no autónomas en IP-Core*. in *XXI Congreso Argentino de Ciencias de la Computación (Junín, 2015)*. 2015.