Artículo de Investigación. Revista Ciencia e Ingeniería. Vol. 42-3, No. 3 pp. 297-306, agosto-noviembre, 2021.
ISSN 1316-7081. ISSN Elect. 2244-8780 Universidad de los Andes (ULA)

Consenso y metodología de modelación para una clase de sistemas dinámicos de eventos discretos

Consensus and modeling methodology for a class of discrete event dynamic systems

Barzola-Rizzo, Mónica[1*]; Mata-Diaz, Guelvis [2]; Ruiz-Leal, Bladismir[3]
[1]Instituto de posgrado, Universidad Técnica de Manabí, Ecuador.
[2] Facultad de Ciencias, Departamento de Matemáticas, Universidad de Los Andes, Mérida, Venezuela.
[3]Instituto de Ciencias Básicas, Universidad Técnica de Manabí, Ecuador.
*Monicaalexandrabarzolarizzo@gmail.com

Resumen

Se propone un mecanismo basado en la descomposición de un sistema dinámico de eventos discretos que, una vez conjugadas las acciones comunes entre sus subsistemas, es acoplado a condicionamientos locales mediante una metodología de modelación que considera el concepto de consenso, y que reduce el tamaño del sistema permitiendo análisis global desde un punto de vista local.

Palabras clave: Supervisión, gramática, procesos, categorías, dinámica.

Abstract

A mechanism based on the decomposition of a dynamic system of discrete events is proposed that, once the common actions between its subsystems have been combined, is coupled to local conditions by means of a modeling methodology that considers the concept of consensus, and that reduces the size of the system allowing global analysis from a local point of view

Keywords: discrete event dynamic systems, planning, *k*-interactions, consensus.

1 Introducción

Un Sistema Dinámico de Eventos Discretos (SDED) puede ser descrito como un conjunto de acciones (eventos), junto con un mecanismo que especifica los posibles ordenes en que dichas acciones pueden ocurrir. Los sistemas de manufactura, sistemas de tráfico aéreo, redes de comunicación, sistemas operativos de computación, entre otros, son SDED. Ellos son asincrónicos y secuenciales y eventualmente pueden mostrar características tales como concurrencia, conflicto, exclusión mutual y no determinismo (Aldaniya 2018, Andrade y col., 2018, Arias y col., 2018, Mata y col., 2018). En la literatura podemos encontrar diversos enfoques para la modelación, análisis y control del comportamiento dinámico de SDED tales como las Redes de Petri, los Autómatas y los procesos recursivos finitos (Eilemberg 1974, Zhong y col., 1990, Hopcroft y col., 1979., Cieslak y col., 1988, Willner y col., 1991).

Nosotros nos centramos en el análisis de esta clase de sistemas usando teoría de Autómatas (Caspi 1991, Mata y col., 2018, Murata 1989) para el estudio de la complejidad y la síntesis, garantizando que estos sean funcionalmente correctos, lo que permitirá asegurar el cumplimiento de las tareas u objetivos trazados. Originalmente, la construcción y síntesis comienza por considerar a los sistemas como constituidos por subsistemas; luego, tal como es expuesto en (Mata y col., 2018), conjugar las acciones comunes para la reducción del tamaño del modelo, y finalmente expresar la representación global como una composición paralela. Actualmente, como es mostrado en (Arias y col., 2018, Mata y col., 2016), un elemento adicional es incluido en términos algebraicos para la reducción, que es consistente con la comunicación bidireccional de todos los subsistemas asincrónicos del modelo, orientado desde una estación central.

Nosotros estableceremos que la simplificación en la representación es consecuencia de la aplicación de una

metodología que se traduce en hacer análisis global desde un punto de vista local, bajo consenso. Este concepto es añadido desde una clasificación sobre los eventos: *k*-interacciones; la cual determinará los cambios simultáneos entre las componentes constituyentes de un SDED.

La organización comienza con una sección preliminar que abarca los conceptos básicos de las teorías de Autómatas y Lenguajes, junto con una generalización novedosa y genuina de la composición paralela que establecen los fundamentos de nuestro manuscrito. Seguimos con la descripción de nuestra metodología en un contexto de manufactura y la generación del modelo inducido desde esta, para finalmente acoplar a nuestro diseño las condiciones locales añadidas por consenso.

2 Nociones Preliminares

En esta sección trataremos las nociones básicas de las teorías de lenguajes formales y generadores como estructuras algebraicas, necesarias para el desarrollo de nuestro problema central.

Un lenguaje formal es una representación para un SDED. En efecto, un par (Σ, L) donde Σ es el conjunto de etiquetas de los eventos del sistema y L es su comportamiento dinámico, representa un mecanismo para llevar a cabo el comportamiento lógico. Para precisar un poco más, sea Σ un conjunto cualquiera y consideremos el conjunto Σ^* de todas las combinaciones finitas de elementos de Σ, entonces $s = \alpha_1\alpha_2\cdots\alpha_n \in \Sigma^*$, con $\alpha_i \in \Sigma$. Si $\omega = \sigma_1\sigma_2\cdots\sigma_m$, con $\sigma_j \in \Sigma$, para todo $j > 0$; es otro elemento de Σ^*, entonces el producto $s\omega = \alpha_1\alpha_2\cdots\alpha_n\sigma_1\sigma_2\cdots\sigma_m$ es también un elemento de Σ^*. Esta operación binaria, nombrada concatenación, le da a Σ^* estructura de monoide con unidad θ.

Cualquier subconjunto $\mathcal{L}$ de Σ^* es llamado un lenguaje sobre Σ; en consecuencia, la unión, intersección, diferencia y complemento de lenguajes son lenguajes sobre Σ. Más aún, si $\mathcal{L}$ es un lenguaje sobre un alfabeto Σ entonces

$$\bar{\mathcal{L}} = \{s \in \Sigma^* : \exists\, t \in \Sigma^*, st \in \mathcal{L}\} \qquad (1)$$

es un lenguaje sobre Σ. De hecho, $\bar{\mathcal{L}}$ es el conjunto de todos los prefijos de todas las palabras en $\mathcal{L}$. Por definición, $\mathcal{L} \subseteq \bar{\mathcal{L}}$. Cuando $\mathcal{L} = \bar{\mathcal{L}}$ entonces $\mathcal{L}$ será llamado prefijo-cerrado o simplemente cerrado.

Sean $\Sigma_i,\ i = 1,2,\cdots,n$; alfabetos y pongamos

$$\bigcup_{i=1}^{n} \Sigma_i = \Sigma. \qquad (2)$$

Las proyecciones naturales, $\pi_i : \Sigma^* \longrightarrow \Sigma_i^*$, $i = 1,2,\cdots,n$; son dadas por

$$\begin{aligned} &\pi_i(\theta) = \theta_i \\ &\pi_i(\alpha) = \begin{cases} \alpha, & si\ \alpha \in \Sigma_i \\ \theta_i, & si\ \alpha \notin \Sigma_i \end{cases} \\ &\pi_i(s\alpha) = \pi_i(s)\pi_i(\alpha), \text{ para cualquier } s \in \Sigma^*\ y\ \alpha \in \Sigma \end{aligned} \qquad (3)$$

La teoría de generadores o Autómatas es un enfoque que contiene una estructura de transición de estados que establece operaciones de composición para construir un modelo de SDED. Formalmente,

Sea Σ un alfabeto. Un Generador Finito Determinístico (GFD) $\mathcal{G}$ sobre Σ (Σ-generador finito), es un séxtuple $(Q, \Sigma, \delta, \mathcal{E}, i, T)$, donde Q es un conjunto finito de estados, i y $T \subseteq Q$ llamados respectivamente estado inicial y conjunto de estados finales, δ es un subconjunto de $Q \times \Sigma \times Q$ de eventos y $\mathcal{E}$ es un subconjunto de $Q \times \Sigma$ de eventos activos (o habilitados) sobre los elementos de Q.

Un evento (q, σ, p) será denotado $q \overset{\sigma}{\rightarrow} p$. Así, obtenemos un grafo dirigido ponderado: los estados son los vértices y los eventos determinan los arcos, con marcas en Σ. El conjunto de todas las trazas $c: i \longrightarrow q$, $q \in Q$ es denotado por $\mathcal{L}(\mathcal{G})$, y será llamado lenguaje generado por $\mathcal{G}$. Una traza $c: i \longrightarrow t$ en $\mathcal{G}$, con $t \in T$, será llamada una tarea. El conjunto de las etiquetas de tareas en $\mathcal{G}$ forman un lenguaje $\mathcal{L}_m(\mathcal{G})$ de Σ^* llamado el comportamiento de $\mathcal{G}$; es decir,

$$\mathcal{L}_m(\mathcal{G}) = \{\, s \in \Sigma^* : s \text{ es una etiqueta de la traza } c: i \longrightarrow t\ en\ \mathcal{G}, t \in T\} \qquad (3)$$

Un estado $q \in Q$ será llamado accesible si existe una traza $c: i \longrightarrow q$. Si todos los estados de $\mathcal{G}$ son accesibles, entonces diremos que $\mathcal{G}$ es accesible. Se llama parte accesible de $\mathcal{G}$ al Σ-generador finito $\mathcal{G}^a = (Q^a, \Sigma, \delta^a, \mathcal{E}^a, i, T^a)$, donde Q^a es el conjunto de estados accesibles de $\mathcal{G}$, $T^a = T \cap Q^a$, $\delta^a = \delta / Q^a$, $\mathcal{E}^a \subset \delta^a$. Note que $\mathcal{L}_m(\mathcal{G}) = \mathcal{L}_m(\mathcal{G}^a)$. Por otro lado, δ determina la función parcial $\delta: Q \times \Sigma \longrightarrow Q$. Algebraicamente podemos extender δ a una función parcial $\hat{\delta}: Q \times \Sigma^* \longrightarrow Q$, satisfaciendo las siguientes condiciones

$$\begin{aligned} \hat{\delta}(q, \theta) &= q, \quad \forall\, q \in Q, \\ \hat{\delta}(q, st) &= \delta\big(\hat{\delta}(q, s), t\big), \quad \forall\, q \in Q, s \in \Sigma^*, t \in \Sigma. \end{aligned} \qquad (4)$$

Luego, $\hat{\delta}$ así definida, es efectivamente una extensión de δ. Por lo tanto, en lo que sigue, no se hará distinción entre $\hat{\delta}$ y δ.

Sean $\mathcal{G}_i = (Q_i, \Sigma_i, \delta_i, \mathcal{E}_i, i_i, T_i),\ i = 1,2,\cdots,n;\ n$ Σ-generadores finitos determinístico. Se llama

Composición Paralela (o síncrona) de los $\mathcal{G}_i$; al GFD accesible $| \ |_{i=1}^{n}\mathcal{G}_i = \mathcal{G}^a$, donde

$$\mathcal{G} = \left(\times_{i=1}^{n} Q_i, \bigcup_{i=1}^{n} \Sigma_i , \delta, \mathcal{E}, (i_1, i_2, \cdots, i_n), \times_{i=1}^{n} T_i \right) \quad (5)$$

con $\delta\big((q_1, q_2, \cdots, q_n), \alpha\big) = (P_1, P_2, \cdots, P_n)$,

$P_i = \delta_i(q_i,)$, y $P_j = q_j$, si para $i, j = 1,2, \cdots, n$;

$$\alpha \in \bigcap_i \mathcal{E}_i(q_i) \setminus \bigcup_{j:j\neq i} \Sigma_j \quad (6)$$

es indefinida en otro caso, y

$$\mathcal{E}(q_1, q_2, \cdots, q_n) = \bigcap_i \mathcal{E}_i(q_i) \setminus \bigcup_{j:j\neq i} \Sigma_j , \quad (7)$$
$$i, j = 1,2, \cdots, n.$$

Note que

i) $$\mathcal{L}(| \ |_{i=1}^{n}\mathcal{G}_i) = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}(\mathcal{G}_i)) \quad (8)$$

ii) $$\mathcal{L}_m(| \ |_{i=1}^{n}\mathcal{G}_i) = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_m(\mathcal{G}_i)) \quad (9)$$

En la composición paralela un evento común puede ser ejecutado si los n generadores ejecutan dichos eventos simultáneamente: diremos en este caso que los generadores están sincronizados a través de ese evento. Así, los n generadores están siempre sincronizados sobre los eventos comunes. Los otros eventos no están sujetos a esta condición y por lo tanto siempre pueden ser ejecutados, mientras sea posible.

3 Metodología de modelación en manufactura

Un sistema de manufactura es un par de conjuntos (conjunto de actividades y conjunto de recursos), que interactúan para obtener un producto. Las actividades son los procesos de fabricación que son necesarios para la producción. Los recursos son el personal, las máquinas, los materiales, etc.; que son necesarios para la ejecución de las actividades. Un sistema de manufactura incluye un plan de proceso de producción (*programa*) que especifica las actividades y los tipos de recursos, así como también las relaciones de precedencia entre actividades. Sea $\mathcal{G}$ un sistema de manufactura. La metodología incluida en este artículo para llevar a cabo la modelación será establecida mediante la composición paralela, organizada desde las interpretaciones de las condiciones lógicas locales de estados (o estatus) y de los eventos tal como sigue:

1. Identificar las actividades y recursos necesarios para la producción de un artículo (o producto).

2. Ordenar las actividades por las relaciones de precedencia tal como lo establece el programa.

3. Para cada actividad, crear un generador componente que represente el estatus $oci :=\ll ocioso \gg$ ó $eje :=\ll ejecución \gg$, etiquetar un evento $\alpha =\ll comienzo\ de\ ejecución\ de\ la\ actividad \gg$ creando un arco desde *oci* hasta *eje*, y un evento $\beta =\ll completación\ de\ actividad \gg$ para incluir un arco desde *eje* hasta *oci*. En general, el evento de completación para una actividad será el evento de comienzo para la nueva actividad (ver Fig. *1*).

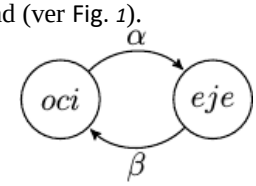

Fig. 1 Un generador componente para una actividad.

4. Para cada actividad, crear un generador componente de cada recurso necesario para el comienzo de la actividad $dis :=\ll disponible \gg$ y $ocu :=\ll ocupado \gg$.

5. etiquetar un evento $\sigma :=\ll comienzo\ de\ utilización\ del\ recurso \gg$ el cual será igual al comienzo de la actividad; creando un arco desde *dis* hasta *ocu*. Finalmente, un evento $\tau :=\ll finalización\ de\ utilización\ del\ recurso \gg$ el cual será igual al de finalización de dicha actividad; para incluir un arco desde *ocu* hasta *dis*. En general, el evento de completación para una actividad con recurso de entrada corresponde a la disponibilidad de dicho recurso de comienzo para la nueva actividad (ver Fig. *2*).

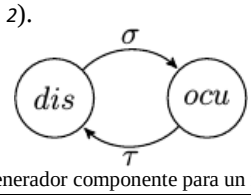

Fig. 2 Un generador componente para un recurso.

6. Establecer el estado inicial: para cada actividad y cada recurso los estatus ocioso y

disponible determinan respectivamente el estado inicial (ver Fig. 3).

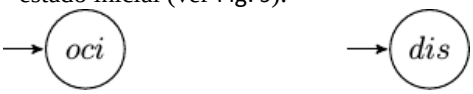

Fig. 3 Estado inicial para cada actividad y cada recurso.

El modelo para el sistema de manufactura $\mathcal{G}$ será establecido por $| \ |_{i=1}^{n}\mathcal{G}_i$, con

$$\mathcal{L} = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_i), \qquad \textbf{(10)}$$

donde los $\mathcal{G}_i$ determinan las dinámicas de las n componentes constituyentes $\mathcal{G}_i, i = 1,2,\cdots,n$; del sistema. Si

$$\bigcap_{i=1}^{n} \Sigma_i = \emptyset,$$

entonces no hay transiciones sincronizadas y en consecuencia $| \ |_{i=1}^{n}\mathcal{G}_i$ es el comportamiento concurrente de los $\mathcal{G}_i$.

Definición 1. Sea $\mathcal{G}$ el generador composición paralela que representa un sistema global. Un evento

$$\alpha \in \bigcup_{i=1}^{n} \Sigma_i \qquad (11)$$

será llamado una *k*-interacción en $\mathcal{G}$ si cada vez que ocurre α éste produce cambios en *k*-componentes, $k \leq n$.

Una *k*-interacción no es más que una expresión simbólica de la ocurrencia simultanea de eventos de ciertos generadores componentes. Estos eventos representan acciones distintas (con etiquetas distintas) pero podríamos darles una interpretación que nos permita conjugarlas mediante un mismo evento. Otro hecho aún más significativo para la construcción del modelo, ocurre cuando una actividad en el sistema requiere de uno o más recursos; donde el comienzo y la completación de dicha actividad tiene una relación directa con el comienzo y la finalización del uso de esos recursos. Igualmente, estos eventos tienen representaciones distintas en el sistema, pero la interpretación de este hecho por el mismo evento nos permitirá simplificar la construcción del modelo. Por lo tanto, diremos que las componentes $\mathcal{G}_i$, $i = 1,2,\cdots,n$; del sistema de manufactura $\mathcal{G}$ estarán determinadas bajo simetría de los eventos que representan los comienzos y finalizaciones de las actividades, así como de la adquisición y liberación de los recursos. Finalmente, en la composición paralela de los $\mathcal{G}_i$, un evento común solo ocurre si ocurre simultáneamente en los $\mathcal{G}_i$ relacionados a través de dicho evento. Por lo tanto, los $\mathcal{G}_i$ están sincronizados sobre los eventos comunes. Los eventos no comunes no están sujetos a esta restricción y pueden ocurrir siempre que sea posible. Por lo tanto, $\mathcal{G} = | \ |_{i=1}^{n}\mathcal{G}_i$.

A continuación daremos un algoritmo para la construcción de la composición paralela $\mathcal{G} = (Q, \Sigma, \delta, \mathcal{E}, q_0, Q_m)$. Para ésta construcción, son fundamentales los conceptos de *k*-interacciones, $k \leq n$; y las componentes

$$\mathcal{G}_i = (Q_i, \Sigma_i, \delta_i, \mathcal{E}_i, q_{0i}, Q_{mi}), \qquad i = 1,2,\dots,n$$

del sistema.

Algoritmo 1 Para la construcción de Generador Composición Paralela

Entrada: $\mathcal{G}_i = (Q_i, \Sigma_i, \delta_i, \mathcal{E}_i, q_{0i}, Q_{mi}), \ i = 1,2,\dots,n$

Salida: $\prod_{i=1}^{n} \mathcal{G}_i$

1. ***Inicio*** $Q \leftarrow \{(q_{01,\ \dots}, q_{0n})\}$, $Q_m \leftarrow \emptyset, \mathcal{E} \leftarrow \emptyset$, $\Sigma \leftarrow \bigcup_{i=1}^{n} \Sigma_i, (q_{01,\ \dots}, q_{0n})$

   no etiquetado

2. ***mientras*** $(q_{1,\ \dots}, q_n) \in Q \wedge (q_{1,\ \dots}, q_n)$ *no etiquetado* ***hacer***

3. ***para*** $\alpha \in \Sigma$ ***hacer***

4. ***para*** $k \in \{1, 2, \cdots, n\}$ ***hacer***

5. ***si*** $\alpha \in \mathcal{E}_k(q_k)$ ***entonces***

6. $p_k = \delta_k(q_k, \alpha)$

7. ***fin si***

8. ***si*** $\alpha \notin \Sigma_k$ ***entonces***

9. $p_k = q_k$

10. ***fin si***

11. ***en otro caso***

12. δ *no esta definida*

13. ***fin para***

14. $\mathcal{E}(q_{1,\ \ldots}, q_n) \leftarrow \mathcal{E}(q_{1,\ \ldots}, q_n) \cup \{\alpha\}$

15. ***si*** $\delta\left((q_{1,\ \ldots}, q_n), \alpha\right) = (p_{1,\ \ldots}, p_n) \notin Q$ ***entonces***

16. $Q \leftarrow Q \cup \{(p_{1,\ \ldots}, p_n)\} \wedge (p_{1,\ \ldots}, p_n)$ *no etiquetado*

17. ***fin si***

18. ***si*** $(p_{1,\ \ldots}, p_n) \in \overset{n}{\underset{i=1}{\times}} Q_{mi}$ ***entonces***

19. $Q_m \leftarrow Q_m \cup \{(p_{1,\ \ldots}, p_n)\}$

20. ***fin si***

21. **fin para**

22. ***fin mientras***

23. ***devolver.***

Iniciamos en (1); asignando al conjunto Q el conjunto unitario $\{(q_{01}, \ldots, q_{0n})\}$, a Σ la unión de los Σ_i, al conjunto Q_m el conjunto $\emptyset$ y al estado $(q_{01}, \ldots, q_{0\mathrm{n}})$ le asignamos **no etiquetado**.

Seguidamente, en (2); siempre y cuando el estado $(q_1, \ldots, q_n)$ sea un elemento de Q y no haya sido etiquetado, para cada evento α, debemos identificar las componentes que cambian por su ocurrencia y el estado obtenido (ver 4-13).

Luego, asignamos a $\mathcal{E}(q_1, \ldots, q_n)$ el evento α (ver 14).

En (15-17); si la ocurrencia de α determina un estado $(p_1, \ldots, p_n)$ que no está en Q, entonces $(p_1, \ldots, p_n)$ deberá ser agregado a Q y ser no etiquetado (en otro caso, el estado ha sido visitado previamente).

En (18-20); verificamos si $(p_1, \ldots, p_n)$ es un estado final. Una vez, revisados todos los eventos activos en $(q_1, \ldots, q_n)$; éste es etiquetado (ver 22).

El procedimiento repetido sucesivamente generará los conjuntos de estados Q, Q_m y eventos activos $\mathcal{E}$.

Finalmente, en (24); con la obtención de $\mathcal{G}$ se finaliza el proceso.

Debido a la complejidad de los cálculos para obtener la composición paralela $\mathcal{G}$ frecuentemente es necesario el uso de un software, a saber TCT. Éste es una herramienta diseñada para la síntesis de supervisores de los SDED modelados por generadores finitos. Sin embargo, nosotros utilizaremos en este trabajo esta herramienta solo para el cálculo del generador composición paralela[1].

En la versión actualizada la sintaxis para crear un nuevo SDED es:

$$SDED := Create(SDED1).$$

La entrada es un generador para un SDED, establecido a partir de una lista de transiciones de la forma $[q, \alpha, p]$; donde $\alpha \in \mathbb{Z}^+ - \{0\}$, q y $p \in \mathbb{Z}^+$. El estado inicial es etiquetado siempre por 0 y los estados marcados son establecidos en una lista. Luego, el producto y la composición síncrona de k SDED; $SDES1, SDED2, \ldots, SDEDk$ serán creados por las sentencias:

$$SDED := Meet(SDED1, SDED2, \ldots, SDEDk)$$

$$SDED := Sync(SDES1, SDED2, \ldots, SDEDk)$$

respectivamente.

Sea $\mathcal{G} = (Q, \Sigma, \delta, \mathcal{E}, q_0, Q_m)$ la composición paralela de n componnentes (subsistemas) $\mathcal{G}_i = (Q_i, \Sigma_i, \delta_i, \mathcal{E}_i, q_{0i}, Q_{mi})$, $i = 1, \ldots, n$.

$$\{(q_{01}, \ldots, q_{0n})\},$$

a Σ la unión de los Σ_i, al conjunto Q_m, el conjunto $\emptyset$ y al estado $(q_{01}, \ldots, q_{0\mathrm{n}})$ le asignamos **no etiquetado**.

4 Construcción del Modelo por consenso

Como fue referido en la sección 3, la construcción mediante la composición paralela puede ser una tarea muy laboriosa y de una gran complejidad. En este sentido, se hace necesario introducir un concepto que nos permitirá agrupar los generadores componentes con la finalidad de simplificar la construcción del modelo. Sea $\mathcal{G} = (Q, \Sigma, \delta, \mathcal{E}, q_0, Q_m)$ *un* sistema de manufactura, enfatizamos que la composición paralela de *n* subsistemas

$\mathcal{G}_i = \left(Q_i, \Sigma_i, \delta_i, \mathcal{E}_i, q_{0_i}, Q_{m_i}\right)$, $i = 1, \cdots, n$; es tal que

[1] Se pueden descargar las versiones XPTCT para Windows 95/98/2000/XP y LTCT para Linux en: http://www.control.utoronto.ca/wonham/Research.html

$$\mathcal{L} = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_i) \qquad (12)$$

donde los $\mathcal{L}_i$ determinan las dinámicas de los $\mathcal{G}_i$, sobre los alfabetos Σ_i, $i = 1,2,\cdots,n$.

Los lenguajes $\mathcal{L}_i \subseteq \Sigma_i^*$ y $\mathcal{L}_{m_i} \subseteq \mathcal{L}_i$, $i = 1,2,\cdots,n$; representan los conjuntos de sucesiones finitas de eventos que las componentes pueden generar y el conjunto de sucesiones que determinan las completaciones de tareas locales en dichas componentes, respectivamente. Sabemos que, para cualesquiera $\mathcal{L}_i$ y $\mathcal{L}_{m_i}$, $i = 1,\cdots,n$; existe un generador finito determinístico $\mathcal{G}_i$ tal que $\mathcal{L}_i = \mathcal{L}(\mathcal{G}_i)$ y $\mathcal{L}_{m_i} = \mathcal{L}_m(\mathcal{G}_i)$, $i = 1,2,\cdots,n$. Sea $\mathcal{C}$ una condición global, expresada como un lenguaje, sobre el sistema, y para cada i, con $i = 1,2,\cdots,n$; sea $\mathcal{C}_i = \pi_i(\mathcal{C})$ la proyección *i*-ésima de $\mathcal{C}$. La afectación que produce $\mathcal{C}$ sobre $\mathcal{G}$ puede ser proyectada a las dinámicas de las componentes (ver figura 4).

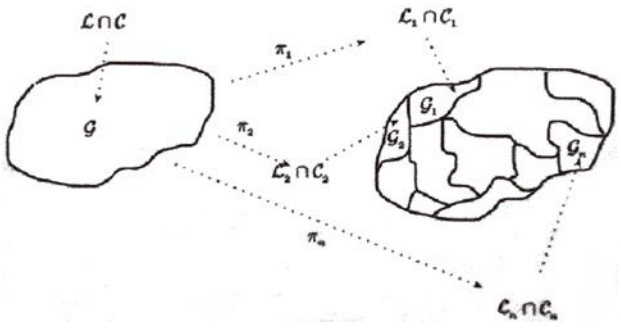

Fig. 4 Las condiciones globales sobre G proyectadas a las dinámicas de las componentes.

Definición 2. Sean $\mathcal{G}$ un sistema global y $\mathcal{C}$ una condición sobre $\mathcal{G}$. Para cualesquiera $q = \delta(q_0, s)$ y $\alpha \in \mathcal{E}(q)$ una *k*-interacción, sean $j_1, j_2, \cdots, j_k$; los índices de las componentes de cambio por la ocurrencia de α en q. Diremos que α es obtenida por consenso si para todo j_l, $l = 1,\cdots,k$ se tiene que $\pi_{j_l}(s)\alpha \in \pi_{j_l}(\mathcal{C} \cap \mathcal{L})$.

La construcción para la planificación está fundamentada en el consenso como sigue: Para cada estado del sistema de manufactura; $q = \delta(i, s)$, donde $s \in (\cup_{i=1}^{n} \Sigma_i)^*$ e $i = (i_1, i_2, \ldots, i_n)$ es el estado inicial, consideramos el conjunto de todos los eventos habilitados en q; $\mathcal{E}(q)$: para cada k-interacción $\alpha \in \mathcal{E}(q)$, donde $j_1, j_2, \ldots, j_k$ son los índices de las componentes de cambio por la ocurrencia en q, α se deberá satisfacer: $\pi_{ij}(s)\alpha \in \pi_{jl}(\mathcal{C} \cap \mathcal{L})$, $\forall j_i$, $i = 1, \ldots, k$, para ser obtenida por consenso, generando de esta manera un árbol que determina el autómata que reconoce la planificación bajo consenso, desde un punto de vista constructivo (ver Fig. 55.).

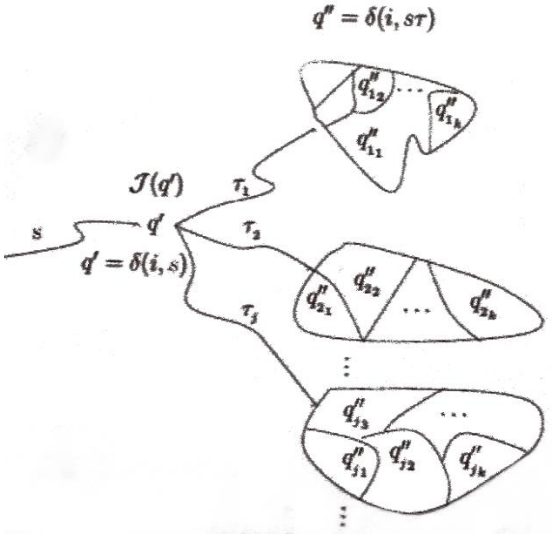

Fig. 5 Una planificación para $\mathcal{G}$ por consenso.

De acuerdo a la definición 2, podemos establecer el conjunto de todos los eventos permitidos en q obtenidos por consenso, el cual denotaremos por $\mathcal{J}(q)$. En efecto, |comenzamos el procedimiento desde el estado inicial *i*, y consideramos el subconjunto de eventos obtenidos por consenso en *i*; $\mathcal{J}(i)$. La ocurrencia de un evento α en $\mathcal{J}(i)$ determinará un nuevo estado *q*, y por ende se generará un arco dirigido desde *i* al nuevo estado *q*, etiquetado por el evento α; $i \xrightarrow{\alpha} q$. Ahora, esto ocurrirá eventualmente para todo evento α en $\mathcal{J}(i)$. Y en el estado actual *q* se repetirá el proceso antes mencionado con respecto a $\mathcal{J}(q)$, generando de esta manera bajo consenso el árbol que determina el generador que reconoce la planificación desde un punto de vista constructivo. El procedimiento repetido sucesivamente generará un conjunto de estados bajo consenso. Finalmente, el proceso resultante será un reconocedor de la planificación y será denotado por $\mathcal{P}(\mathcal{G})$. Formalmente.

Definición 3. Sea $\mathcal{P}(\mathcal{G})$ como antes. El lenguaje generado por $\mathcal{P}(\mathcal{G})$ será llamado lenguaje de planificación y es dada por

- $\theta \in \mathcal{L}(\mathcal{P}(\mathcal{G}))$;
- para cada $s \in \Sigma^*$, y para cada $\alpha \in \Sigma$, se tiene que: $s \in \mathcal{L}(\mathcal{P}(\mathcal{G}))$, $s\alpha \in \mathcal{L}(\mathcal{G}) = \mathcal{L}$, α es obtenido por consenso, si y solo si, $s\alpha \in \mathcal{L}(\mathcal{P}(\mathcal{G}))$.

- El lenguaje de planificación marcado por $\mathcal{P}(\mathcal{G})$ es dado por

$$\mathcal{L}_m\big(\mathcal{P}(\mathcal{G})\big) = \mathcal{L}(\mathcal{P}(\mathcal{G})) \cap \mathcal{L}_m(\mathcal{G}) \qquad \textbf{(13)}$$

Para facilitar la escritura, denotaremos a los lenguajes de planificación y planificación marcada por $\mathcal{L}_p$ y $\mathcal{L}_{p_m}$ respectivamente. El conjunto de todos los estados generados por consenso desde el estado inicial i está dado por $Q_p = \{q \in Q: \exists\, s \in \mathcal{L}_p,\ \delta(i,s) = q\}$, y el generador determinístico accesible $\mathcal{P}(\mathcal{G}) = (Q_p, \Sigma, \delta, \mathcal{J}, i, Q_{p_m})$ es el reconocedor de $\mathcal{L}_p$. La selección de eventos comunes en la composición paralela, lo cual determina el acoplamiento entre las componentes, junto con la reducción en cuanto a la navegación de estados sometidas a las condiciones impuestas localmente permite lidiar eficientemente con la complejidad computacional resultante del modelo completo.

Proposición 1. Dado un lenguaje de planificación $\mathcal{L}_p \subseteq \mathcal{L}$ (respectivamente, un lenguaje de planificación marcado $\mathcal{L}_{p_m} \subseteq \mathcal{L}_m$), entonces $\mathcal{L}_{p_j} \subseteq \mathcal{L}_j$ para cada j, $j = 1,2,\dots,n$ (respectivamente, $\mathcal{L}_{p_{m_j}} \subseteq \mathcal{L}_{m_j}$).

Demostración: $\mathcal{L}_{p_j} = \pi_j(\mathcal{L}_p) \subseteq \pi_j(\mathcal{L}) = \pi_j(\bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_i)) \subseteq \pi_j(\pi_j^{-1}(\mathcal{L}_j)) = \mathcal{L}_j$, pues π_j es sobreyectiva para cada j. Así, $\mathcal{L}_{p_j} \subseteq \mathcal{L}_j$, para todo j, $j = 1,2,\dots,n$. Análogamente se puede demostrar que $\mathcal{L}_{p_{m_j}} \subseteq \mathcal{L}_{m_j}$, para cada j, $j = 1,2,\dots,n$. ■

Teorema 1. Dado un lenguaje de planificación $\mathcal{L}_p \subseteq \mathcal{L}$ para $\mathcal{A}$ (respectivamente, un lenguaje de planificación marcado $\mathcal{L}_{p_m} \subseteq \mathcal{L}_m$ para $\mathcal{A}$). Sea para cada j, $j = 1,2,\dots,n$, $\mathcal{L}_{p_j} = \pi_j(\mathcal{L}_p)$ la planificación local para $\mathcal{A}_j$ (respectivamente, $\mathcal{L}_{p_{m_j}} = \pi_j(\mathcal{L}_{p_m})$ la planificación marcada local para $\mathcal{A}_j$). Entonces, $\mathcal{L}_p = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_{p_i})$ (respectivamente $\mathcal{L}_{p_m} = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_{p_{m_i}})$).

Demostración: Claramente $\mathcal{L}_p \subseteq \bigcap_{i=1}^{n} \pi_i^{-1}\left(\mathcal{L}_{p_i}\right)$. Por otro lado, para cada j, $j = 1,2,\dots,n$, $\pi_j^{-1}\left(\mathcal{L}_{p_j}\right) \subseteq \mathcal{L}_p$; de donde, $\bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_{p_i}) \subseteq \mathcal{L}_p$ .Luego, $\mathcal{L}_p = \bigcap_{i=1}^{n} \pi_i^{-1}\left(\mathcal{L}_{p_i}\right)$. Análogamente, se puede demostrar que $\mathcal{L}_{p_m} = \bigcap_{i=1}^{n} \pi_i^{-1}(\mathcal{L}_{p_{m_i}})$. ■

El Teorema anterior dice que se puede establecer sobre un SDED análisis global desde un punto de vista local.

5 Aplicación

En esta sección, consideramos una aplicación con la finalidad de obtener bajo imposiciones sobre $\mathcal{G}$ la planificación $\mathcal{P}(\mathcal{G})$ y el lenguaje de planificación $\mathcal{L}_p$.

Sea $\mathcal{G}$ un sistema de transferencia que consiste de dos estaciones de trabajo E_1 y E_2 que producen los productos W y V respectivamente, y de un robot R que es utilizado para transferir los productos W y V. Una vez que R comienza a transferir una parte éste no puede ser interrumpido. Más aún, finalizado el traslado de una parte inmediatamente otra parte es hecha disponible y R aleatoriamente transfiere una parte nuevamente.

Las actividades son las transferencias de W y V por R (las actividades de producción de partes por las estaciones de trabajo E_1 y E_2 son operaciones establecidas en el sistema, pero éstas no serán incluidas en el modelo con el fin de simplificar los cálculos), y los recursos son R y las partes.

Así, las actividades a considerar son:

transferir W y transferir V.

Estas actividades no incluyen relaciones de precedencia; cualquiera de ellas puede ocurrir antes que la otra. Para estas actividades requerimos la disponibilidad de los siguientes recursos:

W en E_1 espera ser transferida,

V en E_2 espera ser transferida, y

El robot R esta listo para transferir una parte.

Estas actividades y recursos establecerán las componentes del sistema de transferencia como sigue:

Para una parte, en cada actividad de transferencia el evento σ_1 etiqueta el comienzo de la actividad "transferir una W ", mientras que σ_3 etiqueta la completación de dicha actividad. Así mismo, los eventos σ_2 y σ_4 etiquetan respectivamente el comienzo y completación de "transferir una V". La ocurrencia de la actividad de transferencia de una W determina un estado local; el estado ejecución de la componente representante de la actividad de transferir una W. Denotemos este hecho por $q_{1_{1_3}}$, mientras que la completación de la misma actividad la representamos por $q_{0_{1_3}}$; estado de ocio. Así mismo, los estados $q_{1_{2_3}}$ y $q_{0_{2_3}}$ indican respectivamente ejecución y ocio de la actividad de transferir una V (ver Fig. **8**).

Para una parte, para cada recurso los eventos tienen la misma interpretación que las mencionadas para cada actividad: para R , W en E_1 y V en E_2 (W y V esperando ser transferida por R), los eventos; comienzo y completación de ocupación de dichos

recursos son etiquetados respectivamente por $\sigma_1, \sigma_2, \sigma_3$ y σ_4 según el caso para las actividades. Sin embargo, para cada recurso los estados locales difieren, es decir, para el recurso W en E_1; su disponibilidad es indicada por $q_{0_{1_1}}$ mientras que $q_{1_{1_1}}$ indica la no disponibilidad de éste (ver Fig. 6). Así mismo, los estados $q_{0_{2_1}}$ y $q_{1_{2_1}}$ indican respectivamente la disponibilidad o no del recurso V en E_2, el cual espera ser transferido. El número de ocurrencias de los segmentos $\sigma_1\sigma_3$ y $\sigma_2\sigma_4$ indican las cantidades de W y V transferidas respectivamente. La disponibilidad del robot R para transferir una parte es representada por $q_{1_{1_2}}$ y $q_{1_{2_2}}$, mientras que la no disponibilidad de este recurso para transferir es etiquetado por $q_{0_{1_2}}$ y $q_{0_{2_2}}$, estos indican cual parte ocupa a R; una W y V respectivamente (ver Fig. 7).

Asignaremos a las actividades y recursos los generadores finitos determinísticos $\mathcal{G}_{i_j}$, sobre los alfabetos Σ_{i_j}, $i = 1,2$; $j = 1,2,3$ donde $\Sigma_{1_j} = \{\sigma_1, \sigma_3\}$; y $\Sigma_{2_j} = \{\sigma_2, \sigma_4\}$; $j = 1,2,3$. Por lo tanto, las representaciones por generadores finitos determinísticos de las actividades y recursos son las siguientes:

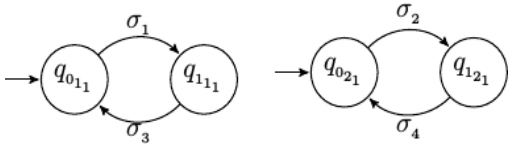

Fig. 6 Los recursos $\boldsymbol{W}$ y $\boldsymbol{V}$ representados por $\boldsymbol{\mathcal{G}_{1_1}}$ y $\boldsymbol{\mathcal{G}_{2_1}}$ respectivamente.

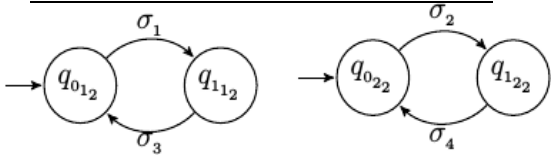

Fig. 7 El recurso $\boldsymbol{R}$ para transferir W y $\boldsymbol{V}$ representados por $\boldsymbol{\mathcal{G}_{1_2}}$ y $\boldsymbol{\mathcal{G}_{2_2}}$ respectivamente.

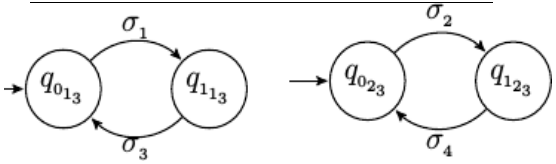

Fig. 8 Las actividades de transferencia de W y V representadas por $\mathcal{G}_{1_3}$ y $\mathcal{G}_{2_3}$ respectivamente.

Si convenimos que los estados para cada generador componente representando la actividad de transferir W y los recursos W en E_1 y R listo para transferirlo, están dados por q_{0_1}, q_{1_1}, donde los q_{i_j}, $i = 0,1$ y $j = 1$ representan los estados locales: los estatus ocioso y ejecución, para la actividad de transferir una W, y los estatus disponibilidad y ocupación, de los recursos W en E_1 y R. Así mismo, los estados q_{0_2}, q_{1_2} conjugan los estatus para la actividad de transferir una V y los recursos necesarios para llevar a cabo dicha actividad. Por lo tanto, podemos agrupar los generadores componentes representando las actividades y los recursos del sistema de transferencia por los siguientes generadores finitos determinísticos (ver Fig. 9)

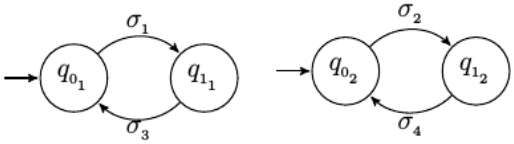

Fig. 9 Representación de las componentes $\boldsymbol{\mathcal{G}_i}$, $\boldsymbol{i} = \boldsymbol{1}, \boldsymbol{2}$; de $\boldsymbol{\mathcal{G}}$.

Luego, la composición paralela de los $\mathcal{G}_i$, $i = 1,2$ representará el sistema de transferencia $\mathcal{G}$. Si convenimos nuevamente, en considerar los estados del sistema como $q_0 = (q_{0_1}, q_{0_2})$, $q_1 = (q_{1_1}, q_{0_2})$ y $q_2 = (q_{0_1}, q_{1_2})$ donde los q_{i_j}, $i = 0,1$, $j = 1,2$ representan los estados locales de las agrupaciones anteriores, entonces podemos representar el sistema de transferencia por el siguiente Σ -generador finito determinístico (ver Fig. 1010.).

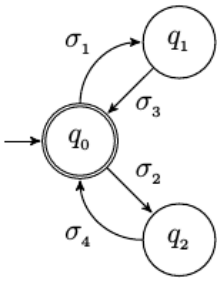

Fig. 10 Representación del sistema global por un $\boldsymbol{\Sigma}$-generador finito determinístico $\boldsymbol{\mathcal{G}}$.

Condición: no ocurre la actividad de transferir una W hasta que una nueva especificación sea establecida. Esto se puede interpretar como el conjunto de todas las sucesiones de eventos que completan tareas y que no contengan el evento σ_2. Más aún, las palabras no contienen el segmento $\sigma_2\sigma_4$ pero determinan completaciones de tareas, es decir, las palabras en $\mathcal{L}_m(\mathcal{G}) \cap \mathcal{K}$, donde

$$\mathcal{K} = \{k \in \Sigma^* : k \neq \mu\sigma_2\sigma_4\omega, \forall\, \mu, \omega \in \Sigma^* \text{ y } \sigma_2, \sigma_4 \in \Sigma\} \qquad \textbf{(14)}$$

Luego, $\mathcal{C} = \mathcal{L}_m(\mathcal{G}) \cap \mathcal{K}$. Ahora, los lenguajes marcados para los $\mathcal{G}_i = (Q_i, \Sigma, \delta_i, \mathcal{E}_i, q_{0_i}, Q_{m_i})$, $i = 1,2$; están dados por $\mathcal{L}_{m_1} = (\sigma_1\sigma_3)^*$ y $\mathcal{L}_{m_2} = (\sigma_2\sigma_4)^*$, respectivamente. Luego, como para $i = 1,2$, $\mathcal{C}_i = \pi_i(\mathcal{L}_m(\mathcal{G}) \cap \mathcal{K})$ es la afectación local sobre la i-ésima componente del sistema, se tiene, $\mathcal{C}_1 = (\sigma_1\sigma_3)^*$ y $\mathcal{C}_2 = \emptyset$. La composición paralela de las componentes $\mathcal{G}_i = (Q_i, \Sigma, \delta_i, \mathcal{E}_i, q_{0_i}, Q_{m_i})$, $i = 1,2$ representará el

sistema de transferencia $\mathcal{G} = (Q, \Sigma, \delta, \mathcal{E}, q_0, Q_{m_i})$, donde $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4\}$, $\delta \subset Q \times \Sigma \times Q$, $Q_{m_i} = \{q_0\}$ y $\mathcal{E}(q_j)$, $j = 0,1,2$ está establecido desde las componentes como sigue en la **¡Error! No se encuentra el origen de la referencia.**:

Tab. 1 *Valores de los* $\mathcal{E}(q_j)$

q_j	$\mathcal{E}(q_j)$
q_0	$\{\sigma_1, \sigma_2\}$
q_1	$\{\sigma_3\}$
q_2	$\{\sigma_4\}$

Luego, la clasificación de los eventos *k*-interacciones) según la sincronización de las componentes $\mathcal{G}_i$, $i = 1,2$; es dada en la **¡Error! No se encuentra el origen de la referencia.**

Tab. 2 *Valores de las k-interacciones*

Σ_i-Orden	$k = 1$
Σ_1	σ_1, σ_3
Σ_2	σ_2, σ_4

Para la construcción de la planificación $\mathcal{P}(\mathcal{G})$ comenzaremos considerando las *k*-interacciones activas en el estado inicial q_0; σ_1 y σ_2. Aquí, $\mathcal{J}(q_0) = \{\sigma_1\}$, ya que,

$$\pi_1(\theta)\sigma_1 = \theta\sigma_1 \in \mathcal{C}_1, \quad \pi_2(\theta)\sigma_2 = \theta\sigma_2 \in \mathcal{C}_2 \qquad (15)$$

Así, σ_1 es obtenida por consenso en q_0. Bajo el supuesto de que ocurre σ_1, se tiene $q_1 = \delta(q_0, \sigma_1)$. Nuevamente, para $\mathcal{E}(q_1)$ se tiene $\pi_1(\sigma_1)\sigma_3 = \sigma_1\sigma_3 \in \mathcal{C}_1$. Luego, σ_3 es obtenida por consenso en q_1 y su ocurrencia en q_1 es $q_0 = \delta(q_1, \sigma_3)$. El proceso finaliza, ya que, todos los estados han sido alcanzados (ver

Fig. 11).

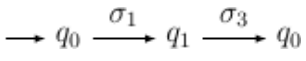

Fig. 11 Construcción parcial de $\mathcal{P}(\mathcal{G})$.

Además, notemos que los estados están siendo generados sucesivamente q_0 y q_1 (son estados duplicados). Esto es ilustrado en la figura12.

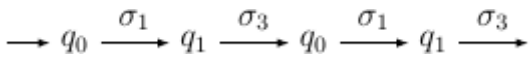

Fig. 12 Construcción parcial de $\mathcal{P}(\mathcal{G})$.

Finalmente, la planificación $\mathcal{P}(\mathcal{G})$ es construida y el lenguaje de planificación está dado por $\mathcal{L}_p = (\sigma_1\sigma_3)^*$

Conclusión

La presentación de la metodología de modelación considerada aquí para sistemas de manufactura expresa por sí misma la construcción de una planificación global obtenida desde planificaciones locales, bajo el concepto de consenso. Esto permite la modelación y análisis global desde un punto de vista local (proyectivo), permitiendo eficiencia y automatización en la reconfiguración de la clase de sistemas de eventos discretos considerada.

Una planificación por consenso implica la consideración de los eventos comunes que denominamos k-interacciones. Cuando ocurre un evento de orden superior, lo cual supone que la planificación dada deja de ser prioritaria, su establecimiento implica el establecimiento de un nuevo consenso. En la implementación, esto implica la salida a estados seguros acordados de antemano. Así, la cooperación entre las componentes constituyentes del sistema expresa la evolución del SDED, una vez se lance el evento de inicio.

Finalmente, dado que nuestra propuesta se desarrolló sobre la base de lenguajes lógicos, y considerando la naturaleza distribuida de las componentes que interactúan, para garantizar la sincronización de las k-interacciones se asume el compartimiento de memoria común en los dispositivos: cuando un evento se presenta, la componente que detecta el evento lo hace común a las otras componentes mediante mensajería.

Referencias

Aldaniya KN, Main factors for the Improvenment of complex systems of strategic production cost managetment, Revista Espacios, 30(11), pp.29, 2018.

Arias L, Agurto W, Chávez Á, Pantoja R, Pinto A, Decisions in Hierarchical production planning: goals, heuristics. Revista Espacios, 30(14), pp 3, 2018.

Caspi P, Model of Discrete event systems in computer science, Proc. ECC 91, European control conference, Grenoble, France, 1991.

Cieslak R, Desclaux C, Fawaz A, Varaiya P, Supervisory control of discrete-event processes with partial observations, IEEE Trans. Autom. Control, Vol. 33, 249-260, 1988.

De Souza H, Loureiro G, Proosta de um modelo de planeamiento estratégico basado em engenharia de sistemas, Revista Espacios, 39(13), pp. 10, 2018.

Eilenberg S, Automata, languages, and machines, Academic Press, New York, 1974.

Hopcroft J, Motwani R, Ullman J, Automata theory, languages, and computation, Addison Wesley, Press

Reading. Massachusetts, 1979.
Mata G, Lugo A, Rojas G, Aplicación de bases de Grobner en el problema de alcanzabilidad de estados de sistemas de eventos discretos modelados por redes de Petri, Lecturas Matemáticas, 37(1), pp 5-23, 2016.
Mata G, Ruiz B, Camacho C, Méndez A, Muñoz S, Zambrano H, A planning algorithm in a class of discrete event system, DYNA. 85(206), 283-293, 2018.
Murata T, Petri Nets: properties, analysis and applications, Proceedings of the IEEE, Vol. 77, No. 4, 1989.
Willner Y, Heymann M, Supervisory control of concurrent discrete-event systems, International Journal of Control, Vol. 54, 1143-1169, 1991.
Zhong H, Wonham W, On the consistency of hierarchical supervision in discrete-event systems IEEE Trans. Autom. Control, Vol. 35, 1125-1134, 1990.

Recibido: *23 de abril de 2021*

Aceptado: *01 de julio de 2021*

Barzola-Rizzo, Mónica A: *Licenciada en Ciencias de la Educación en física y matemática. Instituto de posgrado. Universidad Técnica de Manabí.*

Mata-Diaz, Guelvis E: *Licenciado en Matemáticas. Magister Scientiae en Matemáticas. Doctor en Ciencias Aplicadas. Profesor Titular activo. Facultad de Ciencias ULA. Departamento de Matemáticas. Línea de investigación: análisis y control en sistemas dinámicos de eventos discretos. Correo electrónico: gematad2017@gmail.com*

Ruiz-Leal, Bladismir: *Licenciado en Matemáticas. Magister en Matemáticas. Doctor en Ciencias. Profesor activo de la Universidad Técnica de Manabí. Ciencias Básicas UTM. Línea de investigación: sistemas dinámicos y teoría ergódica.*