Disponible en www.sciencedirect.com

ScienceDirect

Revista Iberoamericana de Automática e Informática industrial 8 (2011) 196–203

www.elsevier.es/RIAI

Supervisión de sistemas lógicos de control utilizando el diagrama de evolución del estado ☆

Daniel Gómez[a,*], Enrique Baeyens[b], Clemente Cárdenas[a], Eduardo J. Moya[b]

[a] *Fundación CARTIF, Parque Tecnológico de Boecillo 205, 47151 Boecillo, Valladolid, España*
[b] *Instituto de las Tecnologías Avanzadas de la Producción, Universidad de Valladolid, Paseo del Cauce 59, 47011 Valladolid, España*

Resumen

Los sistemas de fabricación actuales están controlados y supervisados por controladores lógicos programables. El análisis y mantenimiento de un programa de control es una tarea muy compleja debido a su gran tamaño medido tanto en líneas de código como en número de variables utilizadas. El análisis de los posibles problemas que puede presentar el proceso controlado, como por ejemplo, cuellos de botella y bloqueos, requiere de herramientas formales. Sin embargo, las herramientas existentes presentan importantes limitaciones. En este trabajo se propone el uso del diagrama de evolución del estado para realizar dicho análisis. Este diagrama complementa a otras herramientas formales basadas en redes de Petri o autómatas finitos y permite analizar los sistemas de control lógico a partir de las señales físicas del proceso. Resulta ser una herramienta de gran utilidad en las fases de validación y puesta en marcha, así como para la supervisión de procesos durante la fase de ejecución. *Copyright © 2011 CEA.*

Palabras Clave: Control lógico, Autómatas finitos, Redes de Petri, Grafos orientados, Bloqueos, Cuellos de botella, Diagrama de estado

1. Introducción

La mayoría de las líneas de fabricación actuales, tanto de procesos como de manufactura, están controladas por dispositivos denominados autómatas programables o controladores lógicos programables (PLC). Estos dispositivos de control se programan en alguno de los lenguajes de la norma IEC 61131-3 (John and Tiegelkamp, 2001).

Los programas de los sistemas actuales constan de un número muy elevado de líneas de programación, por lo que el análisis y seguimiento de los mismos resulta ser una tarea sumamente complicada (Sen and Chu, 2004). Además, debido al gran número de entradas provenientes de sensores y de salidas hacia los actuadores que presenta un sistema de fabricación actual, el análisis de los diferentes problemas que aparecen en el proceso controlado, como por ejemplo, bloqueos o cuellos de botella, requiere muchas veces de la comprobación *in situ* del problema.

Las tres propiedades básicas más estudiadas en la validación y verificación de un sistema de control son la ausencia de bloqueos, la viveza del proceso controlado y la ausencia de cuellos de botella. Dependiendo del marco formal utilizado, se han propuesto diferentes planteamientos para poder detectar y analizar dichos problemas.

El presente trabajo revisa el planteamiento y definición de los problemas que se presentan a la hora de analizar el comportamiento de un sistema de fabricación cuando se diseña o se modifica su programa de control. La principal contribución es la utilización del diagrama de evolución del estado como herramienta complementaria para el análisis.

El artículo está organizado de la siguiente manera. En el apartado 2 se explica el problema de análisis de los sistemas de control lógico y las principales situaciones anómalas que podrían ocurrir y que deben ser identificadas y evitadas. En el apartado 3 se realiza una breve revisión de las principales metodologías existentes para la detección de bloqueos y cuellos de botella. En el apartado 4 se presenta la solución propuesta, que hace uso del diagrama de evolución del estado. Se describe el diagrama de evolución del estado, se explica cómo puede ser obtenido a partir de las señales registradas durante la operación del sistema de control lógico y se presentan métodos basados en este diagrama para la detección de bloqueos y cuellos de botella. Los procedimientos desarrollados se aplican a un ejemplo en el apartado 5. El artículo finaliza con la presentación de

☆ Financiado parcialmente por el Plan Nacional de I+D+i. Código de proyecto DPI2008-05795 y ayuda BES-2006-12849. Ministerio de Ciencia e Innovación de España.

* Autor en correspondencia
Correos electrnicos: dangom@cartif.es (Daniel Gómez), enrbae@eis.uva.es (Enrique Baeyens), clecar@cartif.es (Clemente Cárdenas), edumoy@eis.uva.es (Eduardo J. Moya)

doi:10.1016/j.riai.2011.06.007

conclusiones en el apartado 6.

2. Análisis de sistemas de control lógico

El mantenimiento y seguimiento de un programa de control de un sistema de fabricación con miles de señales y variables interrelacionadas entre sí, es una tarea extremadamente complicada. El uso de los lenguajes de la norma IEC 61131-3 se limita a la programación del sistema de control, pero no facilita en modo alguno la evaluación de las prestaciones del mismo (Sen and Chu, 2004). De esta manera, es muy difícil realizar cualquier cambio o modificación en un programa de control, ya que se podría introducir inconsistencias importantes. En la actualidad, la comprobación de vulnerabilidades se realiza mediante prueba y error durante la fase de desarrollo del proyecto, en su primera puesta en marcha, o incluso durante la fase de ejecución, con consecuencias que pueden ser totalmente indeterministas y nunca suficientemente validadas. Además, se pierde la visión global y las técnicas de análisis utilizadas no pueden garantizar la presencia de alguna de las situaciones que se detallan a continuación.

- *Bloqueo:* Es una situación alcanzada por un sistema de fabricación, o una parte del mismo, en la que resulta imposible continuar con el programa previsto de control ya que las condiciones lógicas para su continuidad son incompatibles (Lee and Tilbury, 2007).
- *Cuello de botella:* Se caracteriza por la existencia de una parte de un sistema de fabricación o proceso que limita las prestaciones de todo el conjunto debido a que se ejecuta más lentamente o a que puede tener una mayor probabilidad de no estar disponible (Roser et al., 2002).

3. Métodos utilizados para el análisis

Entre las diferentes estrategias empleadas para hacer frente al bloqueo se pueden citar las siguientes (Pia and Zhou, 2004):

- *Prevención de bloqueos:* Son procedimientos que aseguran ciertas condiciones suficientes para evitar un bloqueo. Se incluyen durante la etapa de diseño y pueden llegar a ser muy conservadoras. Una ventaja que presenta esta estrategia es que no se requiere el conocimiento del estado actual del sistema durante su ejecución, proporcionando leyes de control muy simples.
- *Detección y recuperación de un bloqueo:* Son mecanismos de seguimiento que detectan la ocurrencia de una situación preestablecida de bloqueo y conducen al sistema a otra situación libre de bloqueo mediante la finalización de los procesos que la generaron o la liberación de los recursos apropiados. Son menos conservadores que los métodos de prevención y permiten una mejor utilización de los recursos.
- *Eliminación del riesgo de bloqueo:* Son procedimientos que comprueban el estado actual y determinan los estados futuros del sistema para evitar que evolucione a una situación de bloqueo.

Entre las estrategias para detectar y evitar cuellos de botella cabe señalar las siguientes:

- *Análisis del grado de utilización:* Es uno de los métodos tradicionales para la detección de cuellos de botella (Roser et al., 2002). Los elementos del sistema se clasifican por su grado de utilización, siendo el cuello de botella el elemento con mayor grado de utilización.
- *Análisis de colas de espera:* Consiste en analizar la longitud de las colas de espera de los elementos del sistema de fabricación (Roser et al., 2002). Se calcula la longitud de cada cola o el tiempo de espera y aquel elemento que posea el valor más elevado se considera el cuello de botella del proceso.
- *Otros procedimientos:* Existen métodos que analizan la evolución del cuello de botella de un sistema de producción (Roser et al., 2003). Se basan en el cálculo del tiempo o período activo de una máquina de manera ininterrumpida, de tal forma que se puede calcular cuál es la que representa el cuello de botella en un instante dado o bien a lo largo de un determinado período de tiempo.

3.0.1. Soluciones dependiendo del marco formal utilizado

La dificultad y envergadura de los problemas que aparecen en el control de procesos de fabricación motiva la utilización de un marco formal que permita analizar, verificar y validar el programa de control y actuar en consecuencia cuando se detectan dichos problemas. Numerosos procesos de fabricación pueden ser modelados y analizados haciendo uso de la teoría de sistemas dinámicos de eventos discretos (Cassandras and Lafortune, 2006; Hrúz and Zhou, 2007). Existen diversos marcos formales basados en este paradigma, entre los cuales destacan los grafos orientados (Pia and Zhou, 2004), los autómatas de estados finitos y las redes de Petri (Jiménez et al., 2005; Cassandras and Lafortune, 2006). Existen distintas estrategias y algoritmos para la detección de bloqueos y cuellos de botella en la literatura dependiendo de los modelos utilizados, grafos orientados (Pia et al., 2000), autómatas de estados finitos (Fo and Lawley, 2006) o redes de Petri (Huan, 2007; Uzam et al., 2007; Wu et al., 2008; Dashora et al., 2008).

4. El diagrama de evolución del estado

En este trabajo se propone el uso del diagrama de evolución del estado para la detección y evitación de situaciones de bloqueo y de cuellos de botella durante la ejecución de un programa de control.

4.1. Definición del diagrama de evolución del estado

El diagrama de evolución del estado es una representación gráfica del estado y de las transiciones que un sistema realiza a lo largo del tiempo (Gomez et al., 2009). Este diagrama puede ser utilizado directamente para analizar el comportamiento de un sistema o bien como herramienta auxiliar para obtener un modelo de más alto nivel en alguno de los formalismos indicados anteriormente: grafo orientado, autómata de estados finitos, o red de Petri.

Todas las señales con las que interacciona el controlador lógico programable, tanto las señales de entrada procedentes de los sensores como las señales de salida hacia los diferentes actuadores se registran para su análisis posterior. Además, a partir de los SCADA, se obtiene la evolución de las diferentes variables auxiliares utilizadas en la programación. De esta manera, se dispone de un registro histórico de dichas variables o de su representación en gráficos de tendencia que muestran casi en tiempo real la evolución de las variables. El diagrama de evolución del estado se construye a partir de estos registros históricos analizando la evolución de las señales e identificando los diferentes estados, posiciones, lugares, transiciones, etc. que caracterizan el marco formal que estemos utilizando.

En un primer paso, se agrupan las señales que caracterizan el proceso a lo largo del tiempo en una matriz cuyas filas corresponden al valor de cada una de ellas en diferentes instantes de tiempo. Las señales se pueden obtener bien a partir de un modelo del proceso, o bien durante la evolución real del propio proceso. En este último caso, se obtendrán a partir del SCADA y de los controladores lógicos del sistema. Si las señales son todas ellas binarias, los elementos de la matriz sólo pueden tomar valores 0 y 1. En caso contrario, algunos elementos pueden tomar valores numéricos en representación entera o de punto flotante.

Las señales están muestreadas con un período de muestreo T_s, cuyo valor dependerá de la dinámica del proceso que se esté analizando. Sea $S = \left[s_{ij}\right]$ la matriz de señales, donde s_{ij} representa el valor de la señal de índice $i \in \{1, \ldots, N_s\}$, N_s es el número de señales de que se dispone y $j \in \mathbb{N}$ es un número natural que establece el instante de tiempo especificado en múltiplos del período de muestreo T_s. Es decir, el instante de tiempo del valor s_{ij} corresponde a $t = (j-1) \cdot T_s$.

El valor que puede tomar s_{ij} depende del tipo de señal. En el caso de señales binarias, $s_{ij} \in \{0, 1\}$. En el caso de señales analógicas, su tratamiento es diferente. Dependiendo del proceso, cada señal analógica varía en un intervalo de valores. De esta forma, $s_{ij} \in \left[s_i^L, s_i^H\right]$, donde s_i^L y s_i^H denotan el límite inferior y superior que alcanza la señal ($s_i^L < s_i^H$). El intervalo de variación se subdivide en M subintervalos $\left[s_i^{L_m}, s_i^{H_m}\right]$ donde $m = 1, \ldots, M$, ($s_i^{L_m} < s_i^{H_m}$). El valor que toma s_{ij} será m si la señal analógica i toma un valor comprendido en el correspondiente subintervalo.

A partir de esta representación matricial de las señales, se observa que mientras todas las señales permanecen invariables, el proceso se encuentra en un mismo estado. Cuando alguna señal cambie, el estado del sistema también cambiará. Cada estado se representa mediante una única variable que toma un valor entero no negativo. El estado en el instante de muestreo j, que corresponde a $(j-1) \cdot T_s$ unidades de tiempo, se denota como x_j y toma valores en el conjunto de los enteros no negativos $\mathbb{Z}_+$.

El diagrama de evolución del estado es la representación gráfica del estado x_j del sistema en función del instante de muestreo j.

La obtención de un modelo del sistema dinámico en alguno de los formalismos anteriormente indicados a partir del diagrama de evolución del estado es relativamente simple. Por ejemplo, es posible construir una red de Petri en la que cada estado se representa por un lugar y los cambios de estado son transiciones, que son instantáneas y siempre ocurren en los instantes de muestreo.

4.2. Supervisión utilizando el diagrama de evolución del estado

El diagrama de evolución del estado puede ser utilizado para estudiar los diferentes problemas planteados anteriormente de una manera rigurosa dada su conexión con otros formalismos para el análisis de sistemas dinámicos de eventos discretos.

4.2.1. Cuellos de botella.

Un cuello de botella es detectable en el diagrama de evolución del estado cuando parte de los procesos del sistema que se está analizando pasan forzosamente por una serie de estados o posiciones de una manera repetitiva. Esto representa la existencia de una bifurcación o decisión y genera un cuello de botella que puede desembocar en un bloqueo o bien en que el sistema responda más lentamente.

Para el análisis de la evolución de un sistema se utilizará la siguiente notación. Si k es el número entero no negativo que representa el valor del estado, t_k denota el tiempo inicial de la última ocurrencia de este valor, d_k la duración de esta última ocurrencia, n_k el número de veces que el estado ha tomado valor k y δ_k el tiempo medio entre ocurrencias del valor k.

La detección de un cuello de botella en un diagrama de evolución del estado se puede llevar a cabo de la forma siguiente:

1. Se realiza un seguimiento de todos aquellos estados que forman parte del proceso y que se van sucediendo a lo largo del tiempo. Se van almacenando las siguientes cantidades: el valor del estado, su duración en el tiempo, el instante inicial en el que ocurre, el número de veces que ocurre y el tiempo medio entre ocurrencias.
2. Aquellas posiciones o estados por los que el proceso pasa de una manera continuada y repetitiva serán candidatos a formar parte de un cuello de botella.

Una ventaja del diagrama de evolución del estado es que permite visualizar de forma sencilla posibles problemas que pudieran ocurrir en el proceso. Un posible cuello de botella puede pasar inadvertido en una red de Petri, así como en el resto de marcos formales, debido a la propia definición de transición. Para que no exista un cuello de botella, la ejecución del proceso debe comprender un secuenciamiento de transición única. Traducido a una red de Petri, correspondería a la ejecución secuencial de un lugar y una transición. En el momento en que se

presenta una transición múltiple o multitransición (provocada por ejemplo por una decisión en la ejecución del proceso), el seguimiento se vuelve indeterminista y no es fácil establecer si hay un posible cuello de botella. Por otro lado, el diagrama de evolución del estado permite identificar los estados correspondientes a las transiciones que dan lugar a posibles cuellos de botella de una manera gráfica e intuitiva.

La detección de cuellos de botella mediante simulación de modelos basados en redes de Petri permite el análisis de diferentes métricas de prestaciones. No obstante, los diagramas de evolución del estado siguen siendo una herramienta de gran utilidad, ya que permiten obtener fácilmente la información necesaria para obtener un modelo en red de Petri basado en la evolución real del sistema.

4.2.2. Bloqueos.

Un bloqueo en un sistema de fabricación puede ocurrir al producirse un fallo en alguno de los sensores disponibles o bien cuando alguno de los actuadores no responde a las órdenes recibidas del autómata. Esto se traduce en el diagrama de evolución del estado en que un determinado estado queda activado de forma indefinida.

Otra consecuencia de un bloqueo en un proceso es una secuencia de estados que se repite indefinidamente debido a que algún sensor o actuador ha fallado y no es posible detectar el cumplimiento de la condición de salida (fallo en el sensor) o actuar para continuar la ejecución normal del proceso (fallo en el actuador).

Los bloqueos en sistemas modelados mediante redes de Petri se analizan mediante algoritmos que predicen si entre los futuros estados alcanzables del sistema se encuentra una situación de bloqueo. Estos algoritmos explotan la teoría de control supervisor (Ramadge and Wonham, 1987), por lo que no son parte del marco formal de las redes de Petri. El mayor inconveniente es el coste asociado a la generación de los algoritmos de supervisión que puede ser muy superior al del propio diseño y modelado del sistema, pues exige predecir todos y cada uno de los caminos y secuencias de disparo de las transiciones que forman parte del proceso. Además, presenta el inconveniente añadido de la simultaneidad. Esto ocurre en las transiciones que al dispararse se bifurcan en dos ramas diferentes que evolucionan independientemente, lo que complica el análisis de la presencia de bloqueos.

El análisis de bloqueos mediante el diagrama de evolución del estado se basa las siguientes consideraciones:

- Se dispone de una representación de las secuencias factibles de ejecución del proceso. En (Trujillo et al., 2007) se han desarrollado técnicas para la obtención y clasificación de las secuencias factibles de un proceso productivo. El conjunto de todas las secuencias factibles correspondientes al correcto funcionamiento de un sistema se especifica mediante un lenguaje formal que se denota $\mathcal{L}_p$ y que corresponde al cierre de Kleene (Hopcroft et al., 2006) de un número finito de secuencias primitivas denominadas patrones. La especificación de comportamiento de un sistema se completa con los instantes de comienzo y con la duración de cada valor del estado para los sucesivos valores que toma cada secuencia factible $\alpha \in \mathcal{L}_p$.
- Durante la ejecución del proceso, se define una condición lógica para comprobar si el estado activo permanece dentro de su intervalo de tiempo correspondiente a la secuencia factible con una tolerancia asociada que dependerá de la dinámica del sistema. En el momento en que esta condición no se cumpla, el proceso estará evolucionando hacia una posible situación de bloqueo.

Un proceso automatizado bien diseñado satisface la propiedad de viveza y nunca desemboca en una situación de bloqueo. La secuencia de funcionamiento será cíclica, aunque podrá componerse de subsecuencias primitivas de menor longitud que se concatenan para lograr los objetivos de fabricación. La evolución de la secuencia de estados de un sistema productivo puede ser monitorizada de forma continua por un supervisor (Trujillo et al., 2007). En el momento en que el proceso se desvíe del comportamiento factible, y siga un camino que conduzca a una situación de bloqueo, el controlador podrá tomar las acciones oportunas, mientras exista reversibilidad, para que el sistema regrese a un estado o situación conocida desde la cual se puede recuperar su funcionamiento normal. A veces la recuperación del funcionamiento normal exige la parada del sistema en un estado dado y la reparación de los posibles fallos en los componentes que han provocado la situación no especificada.

Un algoritmo para prevenir bloqueos mediante el diagrama de evolución del estado se basa en las siguientes condiciones:

1. Si el estado actual $x_j = k$ tiene una duración $d_k \geq d_{th}$ que supera un valor umbral, establecido por la dinámica del proceso y las secuencias factibles de funcionamiento, entonces se considera que el proceso ha alcanzado una situación de bloqueo.
2. El control supervisor del proceso dispone de las correspondientes secuencias factibles definidas por el lenguaje $\mathcal{L}_p$. Cada secuencia factible $\alpha \in \mathcal{L}_p$ viene caracterizada por los valores del estado que se van sucediendo durante la ejecución del proceso y sus intervalos permitidos de duración $\{(x, d_x^\alpha, \Delta_x^\alpha)\}$ para $x \in \alpha$ y $\alpha \in \mathcal{L}_p$. De esta forma para la terna $(x, d_x^\alpha, \Delta_x^\alpha)$, y suponiendo que la secuencia actual ha venido cumpliendo todos los estados de la secuencia α, la duración del estado x deberá estar comprendida en en intervalo $[d_x^\alpha - \Delta_x^\alpha, d_x^\alpha + \Delta_x^\alpha]$ para que el estado actual siga cumpliendo la secuencia factible. El control supervisor monitorizará el proceso a medida que se ejecute y comprobará el cumplimiento de la secuencia factible.

La primera de las condiciones comprueba que el estado actual es igual al estado anterior o en caso de haber cambiado si corresponde al siguiente estado de una secuencia factible. La segunda condición comprueba si la duración del último estado que fue visitado está dentro del intervalo que establece la secuencia factible. La segunda condición sólo se comprueba cuando hay cambio de estado.

4.3. Algoritmo general del diagrama de evolución del estado

Los métodos explicados en los apartados anteriores para obtener el diagrama de evolución del estado y detectar posibles contingencias se aúnan en un algoritmo que se describe a continuación.

Algoritmo 1. *El siguiente algoritmo permite obtener la evolución temporal del estado a partir de la matriz S de señales del sistema, detectar y obtener los estados que pueden formar parte de un cuello de botella y detectar la evolución de una secuencia de estados que puede conducir a un bloqueo utilizando el conjunto de secuencias factibles especificadas por el lenguaje $\mathcal{L}_p$. Para cada instante de muestreo $j \in \mathbb{N}$ se comprueba el conjunto de todas las señales s_{ij}, con $i \in \{1, \ldots, N_s\}$ y se determina el estado en el que se encuentra el sistema.*

1. *Inicialización:*
   Si $j = 1$, entonces se considera que la agrupación de todas las señales s_{ij}, para $i \in \{1, \ldots, N_s\}$ constituye el estado $x_1 := 0$.
   Se asignan los siguientes valores:
   - *Valor del estado $x_1 := 0$*
   - *Tiempo inicial $t_0 := 0$*
   - *Duración $d_0 := 0$*
   - *Número de veces que se ejecuta el estado $n_0 := 1$.*
   - *Tiempo medio entre ocurrencias $\delta_0 := 0$.*
   - *Duración $d_0 := 0$*
2. *Para $1 < j \in \mathbb{N}$ el sistema se encuentra en un estado caracterizado por el número entero no negativo k con $k \leq j - 1$ y se realizan las siguientes operaciones*
   - Obtención de los estados de un proceso.
     - *Si $s_{ij} = s_{i\ell}$ para algún $\ell \in \{1, \ldots, j-1\}$ entonces $x_j := x_\ell$.*
     - *En caso contrario $x_j := x_{j-1} + 1$.*
     - *Si el estado actual $x_j = \ell = x_{j-1}$, entonces actualizar*
       - *$d_\ell = d_\ell + 1$.*
     - *Si el estado actual $x_j = k \neq x_{j-1} = \ell$, entonces actualizar*
       - *Valor del estado $x_j := k$*
       - *Tiempo inicial $t_k := j$, donde j es el instante actual.*
       - *Duración $d_\ell := d_\ell + 1$, $d_k = 0$.*
       - *Número de veces que se ejecuta el estado $n_k := n_k + 1$.*
       - *Tiempo medio entre ocurrencias $\delta_k := \delta_k + \frac{1}{n_k}(j - t_k - \delta_k)$*
   - Análisis de cuellos de botella.
     Sea L el número de estados distintos por los que ha ido pasando el proceso durante su evolución hasta el instante de muestreo actual j, si el estado actual $x_j = k$ satisface las siguientes condiciones:
     - *$\frac{n_k}{\sum_{i=1}^{K} n_i} \geq f_{th}$, donde f_{th} es un valor umbral de frecuencia de ocurrencia del estado correspondiente al proceso que se está analizando y,*
     - *$\delta_k \leq \delta_{th}$, donde δ_{th} es un valor umbral del tiempo promedio entre ocurrencias consecutivas del mismo estado k*

     entonces $x_j = k$ es candidato a formar parte de un cuello de botella del sistema.
   - Análisis de bloqueos.
     Para toda secuencia factible $\alpha \in \mathcal{L}_p$ que caracteriza el correcto funcionamiento del sistema de control debe cumplirse las dos siguientes condiciones.
     - *Condición 1: $x_j \in \{x_{j-1}, \sigma^\alpha(x_{j-1})\}$ donde $\sigma^\alpha(x)$ denota el valor del estado siguiente al estado de valor x para la secuencia factible $\alpha \in \mathcal{L}_p$.*
     - *Condición 2: $|d_{x_j} - d^\alpha_{x_j}| < \Delta d^\alpha_{x_j}$ para $x_{j-1} \neq x_j = \sigma^\alpha(x_{j-1})$.*

     Si alguna de ellas no se cumple, entonces el sistema no evoluciona conforme a las especificaciones establecidas por las secuencias factibles y eventualmente finalizará produciendo una situación de bloqueo.

5. Ejemplo

En este apartado se presenta un ejemplo de obtención del diagrama de evolución del estado para un sistema de fabricación y de su utilización para supervisar el comportamiento del sistema.

5.1. Sistema elevador-clasificador de piezas

En la figura 1 se muestra el esquema de un sistema elevador-clasificador de piezas que forma parte de un determinado proceso de fabricación. Los sensores y señales asociados a este proceso se describen en la tabla 1.

El sistema de elevación-clasificación opera de la forma que se describe a continuación. En primer lugar, se mide la longitud de las piezas transportadas por una cadena de rodillos mediante un sensor (DTAM). Para este ejemplo se considera que hay dos tamaños distintos de pieza, pequeño (P) y grande (G). El medidor DTAM emite señal 0 si la pieza es pequeña y señal 1 si es grande. A continuación, las piezas son colocadas en un plano elevador. La secuencia se inicia en el momento en que se detecta la pieza mediante un sensor de proximidad (DPZA). La pieza es elevada mediante un cilindro A accionado por su correspondiente electroválvula. A continuación, se procede a clasificar las piezas. Las piezas pequeñas (P) son colocadas en otra cadena impulsada por el cilindro B y las grandes pasan a una tercera cadena impulsada por acción del cilindro C. Las electroválvulas de los cilindros A y B son monoestables y sólo tienen sentido de avance, mientras que la del cilindro C es biestable con sentido de avance y retorno.

Un modelo en redes de Petri del sistema elevador-clasificador se muestra en la figura 2.

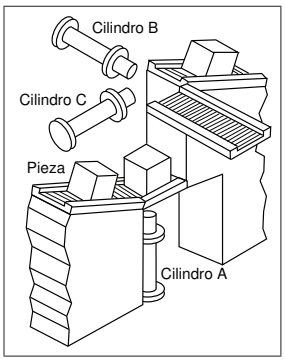

Figura 1: Esquema del proceso del elevador-clasificador

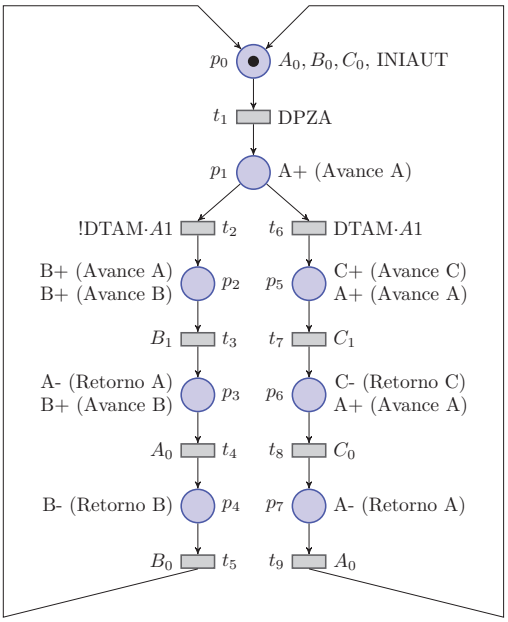

Figura 2: Red de Petri del proceso del elevador-clasificador

Tabla 1: Descripción de los sensores y señales del proceso

SEÑAL	NOMBRE	DESCRIPCIÓN
S1	A0	Sensor (final de carrera) cilindro A atrás
S2	A1	Sensor (final de carrera) cilindro A adelante
S3	B0	Sensor (final de carrera) cilindro B atrás
S4	B1	Sensor (final de carrera) cilindro B adelante
S5	C0	Sensor (final de carrera) cilindro C atrás
S6	C1	Sensor (final de carrera) cilindro C adelante
S7	DPZA	Detector pieza
S8	DTAM	Detector tamaño (0 = pieza P, 1 = pieza G)
S9	EVAVA	Electroválvula Avance A
S10	EVAVB	Electroválvula Avance B
S11	EVAVC	Electroválvula Avance C
S12	EVRTC	Electroválvula Retorno C

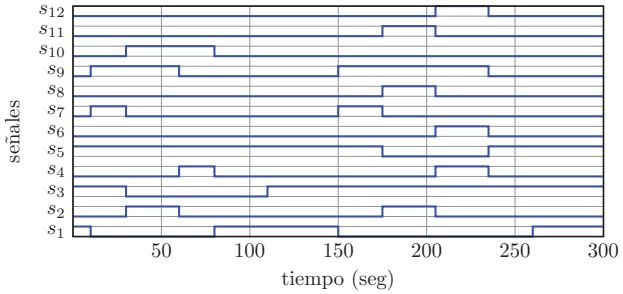

Figura 3: Evolución temporal de las señales reales del proceso obtenidas a partir de un sistema SCADA

El sistema de control del proceso elevador-clasificador se ha simulado en un PLC Siemens de la serie 200 y el programa de supervisión WinCC y sobre él se han estudiado varios ejemplos de operación.

En la figura 3 se representa la evolución de todas las señales que caracterizan al proceso a lo largo de un determinado intervalo de tiempo.

En el eje de abscisas se representa el tiempo en segundos, mientras que en el eje de ordenadas se representan todas las señales físicas del proceso. Las señales son todas digitales binarias por lo que pueden representarse cada una encima de la anterior y variando entre dos niveles, bajo y alto, que representan los valores 0 y 1, respectivamente. Todas estas señales pueden ser almacenadas en una matriz binaria de $N = 12$ filas. La columna $j + 1$ de esta matriz representa el valor de las $N = 12$ señales j segundos después del comienzo del registro de datos.

En la figura 4 se representa la misma información contenida en las señales. En este caso, se representan los intervalos durante los cuales la señal se encuentra activada, es decir cuando toma valor 1. Además en la figura 4 se ha representado mediante elipses el resultado de aplicar el algoritmo 1 de obtención de estados. Cada elipse representa un estado y viene caracterizado porque todas las señales permanecen constantes hasta que se produce un cambio en alguna de ellas, etiquetando los diferentes estados con números naturales consecutivos. Los estados obtenidos tienen una correspondencia directa con los lugares de la red de Petri de la figura 2. Cada vez que una señal cambia se produce una transición de estado. Las transiciones se considera que son instantáneas y que se producen coincidiendo con los instantes de muestreo.

La evolución del estado se representa en la figura 5. En esta figura cada estado se representa por el valor del número entero que lo caracteriza. Para simplificar el análisis en la figura 5 sólo se ha representado en el eje de tiempos los instantes correspondientes a las transiciones, que como ya se ha indicado se consideran instantáneas y cuya ocurrencia coincide con los instantes de muestreo. En la figura 5 se distinguen dos trayectorias de estado esencialmente distintas, la primera desde $t = 10$s hasta $t = 110$s corresponde a la operación sobre una pieza pequeña (P), la segunda desde $t = 150$s hasta $t = 260$s corresponde a la operación sobre una pieza grande (G). Para cada

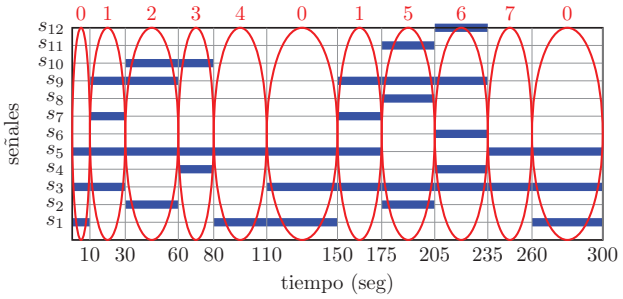

Figura 4: Obtención de los estados por los que evoluciona el sistema

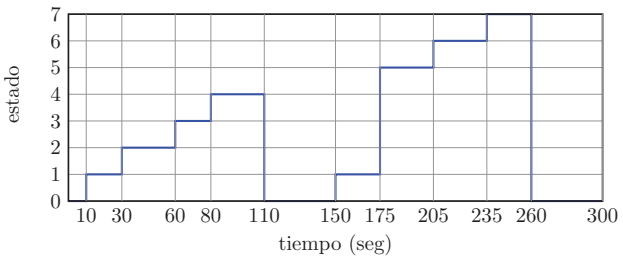

Figura 5: Diagrama de evolución del estado. Clasificación de pieza pequeña y grande sucesivamente

una de estas trayectorias puede definirse una secuencia factible de funcionamiento. Para la pieza pequeña la secuencia factible de funcionamiento es $\alpha_P = \{0, 1, 2, 3, 4\}$ y para la pieza grande $\alpha_G = \{0, 1, 5, 6, 7\}$. El conjunto de todas las secuencias factibles que especifican las condiciones de buen funcionamiento del sistema corresponden con el cierre de Kleene del lenguaje formado por las secuencias α_P y α_G, $\mathcal{L}_p = \{\alpha_P, \alpha_G\}^*$.

5.1.1. *Determinación de cuellos de botella.*

El estudio del diagrama de evolución del estado del sistema elevador-clasificador de piezas para la fabricación de distintos tipos de piezas permite obtener por simple inspección los estados u operaciones del proceso que van a constituir cuellos de botella.

En la figura 6 puede verse claramente que los valores del estado etiquetados como 0 y 1 ocurren invariablemente en la fabricación de cada uno de los dos tipos de piezas. Por tanto estos estados, que corresponden a la clasificación del tipo de pieza que va a ser procesada definen el cuello de botella del proceso. Los sensores y actuadores que operan mientras el sistema permanece en estos estados son candidatos a fallar con mayor probabilidad y en caso de fallar a producir paradas de graves consecuencias al no poder fabricarse ningún tipo de pieza.

5.1.2. *Determinación de bloqueos.*

En el sistema elevador-clasificador de piezas, se pueden dar diferentes situaciones de bloqueo que pueden ser causadas por fallo en algún sensor o actuador.

En el diagrama de evolución del estado de la figura 7 se muestra una secuencia de operación que finaliza con un bloqueo en el estado 6. En este caso, el ciclo en el que el sistema

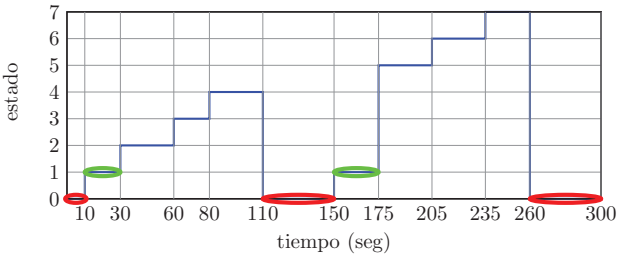

Figura 6: Diagrama de evolución del estado del proceso elevador-clasificador. Identificación de los estados que constituyen cuellos de botella

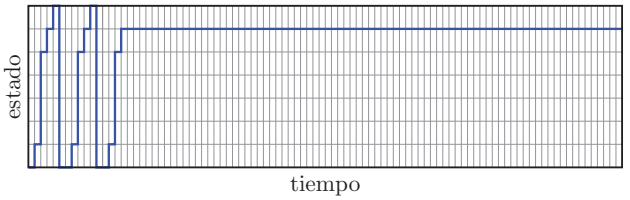

Figura 7: Diagrama de evolución del estado que finaliza en bloqueo en el estado 6

estaba operando corresponde al proceso de clasificación de una pieza grande (G). Las posibles causas de bloqueo son: la electroválvula del cilindro de retorno C no funciona, o bien el sensor correspondiente nunca se activa y no se detecta que dicho cilindro ha alcanzado su posición de reposo.

En el modelo de red de Petri del proceso mostrado en la figura 2, esta situación de bloqueo corresponde a que la ficha se queda en la misma posición indefinidamente a pesar de que la transición de salida esté habilitada.

La forma de evitar que el sistema pueda llegar a la situación de bloqueo consiste en determinar todas las secuencias de procesamiento que permiten obtener los objetivos de fabricación para los diferentes tipos de pieza y que no conducen a situaciones de bloqueo. El lenguaje formal obtenido mediante el cierre de Kleene de todas estas secuencias establece el lenguaje de todas las secuencias factibles. El sistema de control evitará que el sistema evolucione a una situación de bloqueo supervisando que el sistema siempre evoluciona siguiendo una secuencia patrón de buen funcionamiento. Además, si el sistema se desvía de la secuencia de buen funcionamiento, si existiera reversibilidad, el sistema podría ser todavía reconducido por el controlador a un estado determinado desde el cual se pueda restablecer el correcto funcionamiento.

6. Conclusiones

Los sistemas de producción actuales están sometidos a cambios importantes en la demanda y en las especificaciones de producto que obligan a realizar cambios en los sistemas de control de manera rápida y efectiva. Es necesario disponer de herramientas que permitan llevar a cabo el análisis de estos sistemas de control de forma efectiva. Aunque existen algunas

soluciones basadas en marcos formales como los autómatas finitos, las redes de Petri o los grafos orientados, siguen siendo todavía técnicas con poca presencia en la práctica industrial.

Frente a estas soluciones basadas en marcos formales, nuestro trabajo propone la utilización del diagrama de evolución del estado, que se obtiene a partir de las señales físicas de los sensores que monitorizan la evolución de un sistema productivo y que resulta ser una herramienta muy útil por sí misma, así como complemento a otros marcos formales y puede apoyarse en ellos para mejorar el análisis de un sistema. Su principal virtud es que puede ser utilizado durante las fases finales de validación e implantación de los sistemas de control de los procesos de fabricación, ya que puede construirse a partir de las señales reales de los controladores lógicos que gobiernan el proceso. Su empleo permite la detección efectiva de posibles bloqueos y de cuellos de botella de un proceso. Ademas permite identificar situaciones no previstas durante la fase de diseño, ya que pueden ser detectados nuevos estados durante la fase de validación o ejecución que pudieron no ser identificados en la fase de diseño. Finalmente, las trayectorias de estado adecuadas para obtener los objetivos de producción pueden ser especificadas mediante un lenguaje formal que contiene todas las secuencias válidas o patrones. Mediante la utilización de este lenguaje, el sistema productivo puede ser supervisado en tiempo de ejecución evitando su evolución hacia situaciones no deseadas.

English Summary

Analysis of logic control systems using the state diagram.

Abstract

The current manufacturing systems are controlled and monitored by programmable logic controllers. The analysis and maintenance of a control program is a very complex task due to its size measured either by the number of lines of code or number of variables used. The analysis of the potential problems occurring in the controlled process, such as bottlenecks and deadlocks, requires formal tools that can produce such information. However, existing tools have important limitations. This paper proposes the use of the state diagram for such analysis. This diagram complements other formal tools such as those based on Petri nets or finite automata and allows analyzing logic control systems from the physical signals of the process. Therefore, it turns out to be a useful tool in the early stages of validation and implementation, as well as for process monitoring during the implementation phase.

Keywords:

Logic control, Finite state automata, Petri nets, Oriented graphs, Deadlocks, Bottlenecks, State diagram.

Referencias

Cassandras, C., Lafortune, S., 2006. Introduction to Discrete Event Systems. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Dashora, Y., Kumar, S., Tiwari, M., Newman, S., 2008. Deadlock-free scheduling of an automated manufacturing system using an enhanced colored time resource Petri-net model-based evolutionary endosymbiotic learning automata approach. International Journal of Flexible Manufacturing Systems 19, 486–515.

Fo, S., Lawley, M., 2006. Robust supervisory control for production systems with multiple resource failures. IEEE Transactions on Automation Science and Engineering 3 (3), 309–323.

Gomez, D., Trujillo, J., Baeyens, E., Moya, E. J., 2009. Analysis of production systems using the VS-diagram. In: International Symposium On Distributed Computing And Artificial Intelligence 2008. Vol. 50 of Advances In Soft Computing. Springer-Verlag Berlin, pp. 443–451, International Symposium on Distributed Computing and Artificial Intelligence, Salamanca, Spain, Oct 22-24, 2008.

Hopcroft, J., Motwani, R., Ullman, J., 2006. Introduction to Automata Theory, Languages, and Computation (3rd ed.). Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Hrúz, B., Zhou, M., 2007. Modeling and Control of Discrete-event Dynamic Systems: with Petri Nets and Other Tools. Springer Publishing Company, Inc.

Huan, Y., 2007. Design of deadlock prevention supervisors using Petri nets. International Journal of Advanced Manufacturing Technology 35 (3-4), 349–362.

Jiménez, E., Pérez, M., Sanz, F., 2005. Modelado y simulación de sistemas logísticos y de producción mediante redes de petri. Revista Iberoamericana de Automática e Informática Industrial 2 (4), 39–53.

John, K.-H., Tiegelkamp, M., 2001. IEC 61131-3: programming industrial automation systems: concepts and programming languages, decision-making tools. Springer, Berlin.

Lee, S., Tilbury, D., 2007. Deadlock-free resource allocation control for a reconfigurable manufacturing system with serial and parallel configuration. IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews 37 (6), 1373–1381.

Pia, M., Maione, B., Turchiano, B., 2000. Comparing digraph and Petri net approaches to deadlock avoidance in fms. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 30 (5), 783–798.

Pia, M., Zhou, M., 2004. Deadlock control methods in automated manufacturing systems. IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans 34 (1), 5–22.

Ramadge, P., Wonham, W., 1987. Supervisory control of a class of discrete event processes. SIAM Journal of Control and Optimization 25 (1), 206–230.

Roser, C., Nakano, M., Tanaka, M., 2002. Productivity improvement: shifting bottleneck detection. In: WSC'02: Proceedings of the 34th conference on Winter simulation. Winter Simulation Conference, pp. 1079–1086.

Roser, C., Nakano, M., Tanaka, M., 2003. Simulation test bed for manufacturing analysis: comparison of bottleneck detection methods for agv systems. In: WSC'03: Proceedings of the 35th conference on Winter simulation. Winter Simulation Conference, pp. 1192–1198.

Sen, S., Chu, M., 2004. Ladder diagram and Petri-net-based discrete-event control design methods. IEEE Transactions on Systems, Man and Cybernetics - Part C: Applications and Reviews 34 (4), 523–531.

Trujillo, J., Pasek, Z., Baeyens, E., 2007. Analytical method for generating feasible control sequences in controller development. In: ETFA'07: Proceedings of the IEEE Conference on Emerging Technologies and Factory Automation. pp. 673–680.

Uzam, M., Li, Z., Zhou, M., 2007. Identification and elimination of redundant control places in petri net based liveness enforcing supervisors of fms. International Journal of Advanced Manufacturing Technology 35, 150–168.

Wu, N., Zhou, M., Li, Z., 2008. Resource-oriented Petri net for deadlock avoidance in flexible assembly systems. IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans 38 (1), 56–69.