

## OPTIMIZACIÓN DE SISTEMAS LOGÍSTICOS MEDIANTE SIMULACIÓN: UNA METODOLOGÍA BASADA EN REDES DE PETRI COLOREADAS

Mercedes E. Narciso Farias\*, Miquel Angel Piera i Eroles\*, Jaume Figueras\*\*

\* *Departamento de Telecomunicación e Ingeniería de Sistemas.  
Universidad Autónoma de Barcelona. Bellaterra 08193 (Barcelona) España.  
Mercedes.Narciso@uab.es, MiquelAngel.Piera@uab.es*

\*\* *Departamento de Ingeniería de Sistemas, Automática e Informática Industrial.  
Universidad Politécnica de Cataluña. Terrassa 08222 (Barcelona), España.  
Jaume.Figueras@upc.es*

Resumen: Los modelos de simulación han demostrado ser útiles para evaluar el rendimiento de diferentes configuraciones y/o procedimientos de operación alternativos para sistemas logísticos complejos y sistemas de producción. Sin embargo, cuando aplicamos las técnicas de simulación para mejorar el rendimiento y competitividad de sistemas surgen varias limitaciones debido a la incapacidad de evaluar más de una fracción del inmenso rango de opciones (escenarios) disponibles. En este artículo se describe un nuevo enfoque para integrar métodos de evaluación (simulación) con métodos de búsqueda (optimización) basados no sólo en resultados de simulación, sino también usando la información que proporciona el modelo de simulación.

Copyright © 2005 CEA-IFAC

Palabras Clave: programación de actividades (*Scheduling*), optimización, planificación de la producción, Redes de Petri Coloreadas, herramientas de apoyo a la toma de decisiones.

### 1. INTRODUCCION

La competitividad en el mercado mundial, los altos requerimientos de calidad de los productos, junto con demandas aleatorias en lugar de demandas fijas, son algunos de los factores que han forzado la aparición del concepto de flexibilidad a dos niveles:

- *Flexibilidad Tecnológica*: de arquitecturas de producción rígidas y/o no automatizadas (tales como *Flow Shop*, *Job Shop*) hacia Sistemas Flexibles de Manufactura (*FMS: Flexible Manufacturing Systems*).
- *Flexibilidad en la Toma de Decisiones*: de metodologías de planificación de la producción convencionales hacia metodologías basadas en heurísticas, con las cuales poder hacer frente a la gran cantidad de variables de decisión inherentes a las actuales arquitecturas de producción.

La solución óptima exacta de un problema de planificación en un sistema flexible de manufactura, distribución o transporte, es bastante compleja y difícil, e incluso imposible de obtener dada la carencia de herramientas que garanticen la convergencia a valores correctos en un tiempo razonable.

Las limitaciones de la simulación surgen de la incapacidad de evaluar más de una fracción del inmenso rango de posibles escenarios. La mayoría de los paquetes comerciales de simulación orientada a eventos discretos están diseñados para ser usados como herramientas de análisis en las que el sistema bajo estudio es modelado, perturbado, parametrizado y simulado para predecir qué cambios causarían en el sistema real las perturbaciones o las diferentes configuraciones de los parámetros. La Figura 1 ilustra este enfoque.

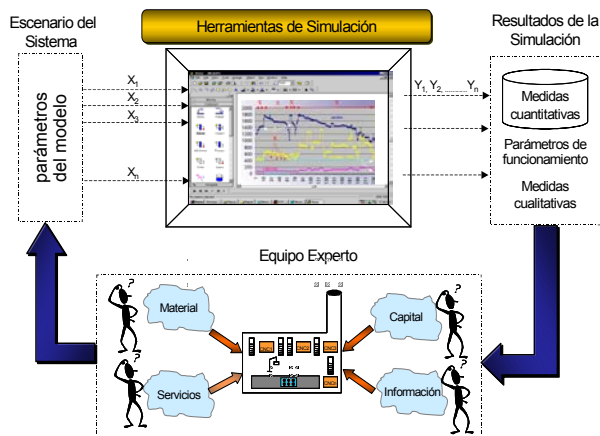


Figura 1. Enfoque de Simulación.

En este sentido, el diseño de experimentos tal vez es una de las técnicas más conocidas que permite analizar de modo automático la respuesta del sistema cuando las variables de decisión (factores cuantitativos) se inicializan a diferentes valores del dominio. El diseño factorial ha sido utilizado extensamente ofreciendo muy buenos resultados para optimizar aspectos relacionados con la planificación estratégica y táctica como pueden ser determinar la cantidad y tipo de recursos necesarios para dar respuesta a los planes de producción.

Ahora bien, cuando los problemas a resolver están relacionados con la toma de decisión operacional, como son la asignación de tareas a máquinas, programación de actividades, enrutamiento de los productos, entre otros, no es posible automatizar de modo eficiente (mantenimiento de los modelos) y eficaz (tiempo de cómputo requerido) la evaluación de los diferentes escenarios.

Los problemas de optimización tradicionales han sido diseñados para buscar las soluciones óptimas cuando el sistema está adecuadamente estructurado en términos de una función objetivo  $F(x)$ , un conjunto de restricciones, requerimientos y límites:

$$\begin{array}{ll}
 \text{Maximizar o Minimizar:} & F(x) \\
 \text{Restricciones:} & Ax \leq b \\
 \text{Requerimientos:} & gl \leq G(x) < gu \\
 \text{Límites:} & l \leq x < u
 \end{array}$$

Los problemas prácticos, a menudo suelen presentar un comportamiento no lineal, relaciones combinatorias y aspectos estocásticos (incertidumbres) que no pueden ser formalizados fácilmente como un conjunto de restricciones y una función objetivo. Además, es de hacer notar que la simplificación de algunas de esas dinámicas complejas en un factor matemático o en una expresión matemática demasiado simplificada conducirá a las técnicas de simulación a resultados muy pobres cuando sean aplicadas al sistema real.

Al integrar en la cadena de producción aspectos relacionados con los servicios que en general las empresas ofrecen o contratan, existe un conjunto de características particulares que conviene formalizar:

- Los servicios se diferencian de los materiales a manipular ya que ellos son: intangibles, heterogéneos (no estandarizados), inseparables (significa la dificultad de separar la producción del servicio del consumo), y perecederos (imposible de almacenar) (Zeithaml, *et al.*, 1985).
- Los servicios logísticos también tienen esas características singulares, sin embargo, también difieren en gran parte de los servicios descritos en la literatura de servicios. Por ejemplo, los servicios logísticos principalmente involucran relaciones negocios-negocios, donde no sólo el comprador es el apostador crítico, sino también su cliente el cual puede ser directamente agraviado debido a un mal servicio. Además, hay en muchos casos una necesidad de interacción estrecha tanto con los procesos de los clientes como con los procesos de sus compradores (Anderson and Norrman, 2002).

Especificar todos estos aspectos cualitativos mediante ecuaciones matemáticas que sean computacionalmente tratables, no siempre será posible debido, entre otros aspectos, a los componentes de incertidumbre inherentes al comportamiento humano, así como otros aspectos que aparecen de modo indirecto como consecuencia de relaciones entre los componentes que integran el sistema logístico.

La simplificación y reformulación del modelo de simulación de acuerdo a unas estructuras de optimización matemática predefinidas ha sido un enfoque ampliamente usado para ciertos sistemas. Sin embargo, cuando aplicamos ese enfoque de simplificación / reformulación a sistemas logísticos y de manufactura, las soluciones resultantes usualmente están drásticamente lejos de ser óptimas o incluso fracasan en satisfacer los requerimientos de factibilidad.

### 1.1 Necesidad de Estructuras Matemáticas Tratables.

Las restricciones lógicas entre los recursos de manufactura (máquinas de procesamiento, unidades de transporte y *stocks* locales), operaciones de producción, junto con sus relaciones temporales y de precedencia son algunos de los elementos claves que usualmente deben ser formalizados en un contexto de simulación como una secuencia de eventos, cada uno con un código computacional asociado que actualice las variables de estado y los contadores estadísticos.

Los actuales paquetes de *software* de simulación ofrecen poderosas herramientas de modelado para describir en el nivel de abstracción deseado todas las relaciones entre operaciones, procesos, recursos, condiciones y tiempos o estados del sistema. Sin embargo, aunque los entornos de simulación han demostrado ofrecer precisión suficiente para representar cualquier comportamiento del sistema, los modelos de simulación carecen de una estructura matemática tratable.

La combinación de métodos de evaluación y métodos de búsqueda es un enfoque atractivo ya que intenta tomar ventaja del potencial combinado de ambos mundos.

### 1.2 Simulación / Optimización.

El uso de algoritmos de búsqueda efectivos para seleccionar un subconjunto (mejores candidatos) de posibles configuraciones es esencial para controlar el crecimiento exponencial en la evaluación de posibles alternativas. Una solución obvia es el uso de rutinas de optimización que puedan guiar una búsqueda intentando mejorar las asignaciones de los parámetros de usuario con respecto a ciertas medidas de rendimiento de interés (Law and McComas, 2002; Pichitlamken and Nelson, 2001).

La Simulación / Optimización puede ser definida (Carson and Maria, 1997) como el uso de métodos de búsqueda para encontrar asignaciones de parámetros de entrada que mejoren los resultados de salida (parámetros de rendimiento) de un sistema simulado (la Figura 2 ilustra este enfoque). Principalmente, en ausencia de estructuras matemáticas tratables este enfoque usa un paquete de "optimización" para organizar la simulación de una secuencia de configuraciones del sistema a fin de que eventualmente pueda obtenerse una configuración del sistema óptima o cercana a la óptima. Conviene hacer notar que cada configuración del modelo corresponde a una asignación particular de las variables de decisión, de tal modo que las rutinas están adaptadas y desarrolladas para alcanzar la solución óptima mediante simulación sólo para un pequeño porcentaje de las configuraciones que serían requeridas por una evaluación exhaustiva.

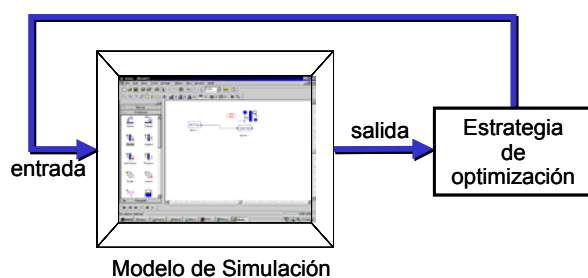


Figura 2. Enfoque Simulación / Optimización.

Avances recientes por parte de la comunidad de optimización han permitido combinar rutinas de optimización matemáticas tradicionales con métodos de Inteligencia Artificial (IA) con el objetivo de guiar satisfactoriamente una serie de evaluaciones y tratar de encontrar valores óptimos para las variables de decisión. Sin embargo, aunque la optimización basada en simulación parece ofrecer un enfoque apropiado para mejorar el rendimiento de ciertos sistemas, aparecen brechas significativas entre la teoría y la práctica en el campo logístico. Cuando se aplica este enfoque a algunos sistemas logísticos complejos, en la práctica no siempre se encuentran buenos resultados, siendo necesario el uso de métodos heurísticos (Glover, *et al.*, 1999).

Por lo general, el enfoque Simulación / Optimización es usado mayoritariamente en aquellos problemas en los que los escenarios a evaluar pueden parametrizarse mediante factores cuantitativos, como es el caso del diseño de capacidades y toma de decisiones estratégicas. Sin embargo, en aquellos problemas relacionados con la toma de decisiones operacional, la evaluación de los distintos escenarios requeriría una inversión de tiempo de CPU superior a la disponible para dar la respuesta industrial requerida, y comportaría una complejidad importante en cuanto a la implementación de un código adicional que garantizara la evaluación automática de los escenarios que pueden aparecer debido a la combinación de circunstancias que no se pueden prever ni evitar debido a la temporalidad estocástica de las actividades. Así pues, esta metodología no ofrece los resultados esperados dado que el elevado número de posibilidades a explorar a través de la simulación es demasiado grande para propósitos prácticos.

Una limitación de la aproximación Simulación / Optimización es que no toma ninguna información del modelo, ni información histórica sobre la evolución de los estados del sistema simplemente. En opinión de los autores, el conocimiento sobre el comportamiento del sistema que se recoge en los modelos es esencial para determinar las alternativas de configuración que conviene evaluar.

Dado que uno de los objetivos claves es automatizar la toma de decisión por medio de rutinas de optimización, se propone una metodología que pueda combinar la información almacenada en el modelo de simulación (identificación de alternativas) junto con los resultados obtenidos mediante diferentes estrategias de evaluación, permitiendo de esta forma cerrar un ciclo completo de un sistema de apoyo a la toma de decisiones (*DSS: Decisión Support System*): definición del problema, identificación de alternativas, análisis, y evaluación de alternativas, seguido de la prescripción de la mejor alternativa.

En la sección 2 de este artículo se introducen conceptualmente las Redes de Petri Coloreadas como herramienta para el modelado de sistemas logísticos o

de manufactura. La sección 3 describe un DSS desarrollado bajo un nuevo enfoque de Optimización mediante el uso de Simulación, que integra métodos de evaluación (simulación) con métodos de búsqueda (optimización) mediante la construcción del árbol de alcance (espacio de estados) de un modelo de Redes de Petri Coloreadas. En la sección 4 se propone un método para la reducción del espacio de estados. Finalmente, la sección 5 ilustra el uso de la metodología sobre dos ejemplos (*benchmarking*) tratados en la literatura: un problema clásico de *scheduling* en un sistema de producción Job-Shop, y un ejemplo de optimización para sistemas de manufactura multi-etapas.

## 2. REDES DE PETRI COLOREADAS

Las Redes de Petri Coloreadas (*CPN: Coloured Petri Nets*) han demostrado ser herramientas acertadas para el modelado de sistemas logísticos debido a las diversas ventajas tales como la capacidad de contener tanto la estructura estática como la dinámica del sistema, arquitectura del sistema de producción, y su naturaleza gráfica (Jensen, 1997; Silva and Valette, 1989; Zimmermann, *et al.*, 1996). Además, las CPN son adecuadas para modelar y visualizar patrones de comportamiento que muestran concurrencia, sincronización y recursos compartidos, los cuales son factores claves cuando se trata de mejorar el rendimiento de sistemas logísticos o de manufactura.

Las CPN soportan un cierto nivel de abstracción en la etapa de modelado mediante el uso de colores, los cuales representan los atributos de las entidades que son soportadas por la mayor parte de paquetes de *software* de simulación comercial. Las CPN son una extensión de las Redes de Petri ordinarias (*PN: Petri Nets*) con la misma capacidad descriptiva pero con mucha más capacidad de síntesis en el modelado de cierto tipo de sistemas, entre los que se encuentran muchos ejemplos de sistemas de producción y logísticos. Tanto las PN (Piera and Guasch, 2001) como las CPN (Zimmermann, *et al.*, 1996) han sido usadas extensivamente para la evaluación del rendimiento de sistemas de producción, ya que el modelo contiene todos los eventos y sus interacciones, así como el tiempo consumido por cada evento.

Las principales características de las CPN, como formalismo de especificación de modelos de simulación orientados a eventos discretos para facilitar la identificación de alternativas son:

- Todos los eventos que pueden aparecer a partir de un estado particular del sistema pueden ser fácilmente determinados a partir del espacio de estados de la CPN (árbol de alcance).
- Todos los eventos que pueden originar la ocurrencia de un evento particular pueden ser detectados visualmente.

Una metodología de modelado que pueda soportar ambas características para cualquier tipo de sistema orientado a eventos discretos, es esencial para abordar la mejora en el funcionamiento de sistemas complejos; desde el modelo conceptual que describe todas las relaciones entre eventos, hasta la codificación del modelo de simulación que pueda soportar la tarea de toma de decisiones de las rutinas de optimización en cualquier momento del proceso de evaluación.

Otras características de las CPN que hacen adecuado el uso de este formalismo para especificar sistemas logísticos o de producción:

- Permiten la especificación de un sistema a diferentes niveles de abstracción, de acuerdo a los objetivos de modelado.
- Permiten la especificación de un sistema complejo por medio de técnicas *bottom-up* (Jensen, 1997) o técnicas de ingeniería del *software* más avanzadas, tales como un proceso de desarrollo incremental e iterativo en lugar de un ciclo de cascada.
- Puede ser considerada una herramienta de modelado gráfico con pocas reglas sintácticas.

Las CPN pueden ser extendidas incorporando a los modelos el concepto de tiempo. Para ello es necesario introducir al modelo CPN los siguientes elementos de modelado:

- Un reloj global que representa el tiempo inmediatamente antes de la ocurrencia de un evento o disparo de una transición.
- Un valor de tiempo (*time stamps*) asociado a cada token en un marcado, que describe el menor valor de tiempo en el cual un token puede ser usado para habilitar una transición.
- Un tiempo de retardo asociado a una transición, que indica, por ejemplo que una actividad modelada mediante una transición consume tiempo.
- Un tiempo de llegada a un marcado que representa el tiempo en el cual el disparo de una transición cambia el estado del sistema a un nuevo estado

En este caso, una transición no sólo estará habilitada cuando los tokens satisfacen las correspondientes expresiones de arco de entrada a la transición, sino que además estos tokens deben estar "listos". Esto significa que todos los tiempos (*time stamps*) asociados a los tokens que habilitan la transición deben ser menores o iguales que el valor actual del reloj. En caso contrario, el reloj global debe avanzar hasta el menor valor de *time stamp* de los tokens para los cuales la transición estaría habilitada.

### 3. UN SISTEMA DE APOYO A LA TOMA DE DECISIONES PARA DETERMINAR PLANES OPTIMOS

El espacio de estados de un sistema logístico o de manufactura puede crecer de forma exponencial con respecto al número de actividades que pueden ocurrir en paralelo. Debido a las restricciones computacionales, así como a la necesidad de tener que tomar decisiones en cortos plazos de tiempo, no es posible construir, analizar y mantener el árbol de alcance completo de una CPN.

Un algoritmo ha sido implementado para examinar el árbol de alcance completo almacenando en memoria solamente una estructura estática (objetos) que describe la CPN del sistema, y una estructura dinámica (árbol binario de búsqueda) que almacena en cada elemento (nodo):

- Descripción del marcado: Información asociada a los tokens (estado del sistema).
- Información sobre la transición disparada (ocurrencia de un evento).
- Información de tiempo de los tokens usados por la transición disparada.
- Información de tiempo sobre el marcado.
- Costo de alcanzar un estado.
- Un identificador que indica el marcado a partir de la cual fue generado el marcado actual (nodo padre).

Puesto que el objetivo es evaluar la evolución del sistema en el dominio temporal, el algoritmo ha sido diseñado para soportar la especificación de CPN-temporales. El algoritmo se basa en el tipo de búsqueda exhaustiva conocida como primero en profundidad (*depth-first search*). En este método de búsqueda el árbol de alcance se examina hasta que se alcanza un nodo a partir del cual no es posible examinar más nodos (nodo hoja), entonces la búsqueda retrocede al nivel anterior y explora cualquiera de las alternativas restantes en este nivel, y así sucesivamente hasta haber explorado todo el árbol. Este procedimiento sistemático de *backtracking* o retroceso garantiza que se examinarán exhaustivamente todas las posibilidades.

Nótese que la posibilidad de retroceder el tiempo de simulación a un estado anterior, y poder simular a partir de este punto otra alternativa, aporta una reducción de tiempo de cómputo importante respecto a los simuladores comerciales, puesto que no requiere volver a inicializar el simulador y repetir los mismos cálculos ya realizados.

Así, a partir del marcado inicial (estado o configuración inicial del sistema), el algoritmo determina todas las transiciones habilitadas para ese

estado. Una transición es seleccionada, un nuevo estado es generado y se chequea su existencia en el árbol binario. Si el nuevo marcado ha sido generado previamente en algún otro nivel del árbol de alcance (nodo *old*), el algoritmo no explora las transiciones habilitadas asociadas con el nuevo estado.

No obstante, si el reloj global del nuevo marcado es menor que el reloj global del nodo *old*, o bien el tiempo de al menos un token del nuevo marcado es menor que el tiempo del mismo token en el nodo *old*, el algoritmo implementado actualiza analíticamente el tiempo de aquellos marcados generados a partir del nodo *old* que pueden conducir a una solución factible del problema. En la siguiente sección se explica este proceso de actualización.

Por otra parte, si el marcado generado corresponde a un nuevo estado no alcanzado previamente, un nuevo nodo es añadido al árbol, y todas las transiciones habilitadas a partir de este nuevo estado son calculadas y procesadas en una forma iterativa similar.

Cuando un marcado generado coincide con el estado final, la rama del árbol es almacenada desde el estado inicial hasta el estado final, con la correspondiente información sobre las transiciones, tiempos y costos de cada marcado en la ruta.

Por cada nodo del árbol binario, la identificación del nodo padre junto con la transición de llegada al marcado es toda la información requerida para retroceder al marcado previo y analizar otro marcado por medio de la selección de otra transición habilitada. De este modo, es posible generar el árbol de alcance de un sistema y determinar la mejor secuencia de acciones que conducen al sistema del estado inicial al estado final deseado.

La Figura 3 muestra los elementos requeridos por el DSS implementado para conseguir la mejor política de *scheduling* (Narciso and Piera, 2001):

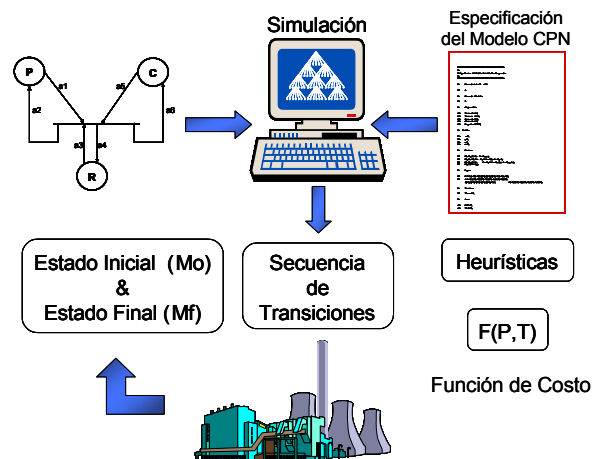


Figura 3. Principales elementos requeridos por el DSS implementado.

- El modelo CPN: se describe sintácticamente en un fichero ASCII. El DSS admite CPN temporales, donde los tiempos asociados a las actividades pueden generarse bien de modo determinista o bien de modo estocástico.
- El Estado Inicial: las condiciones iniciales del sistema se especifican mediante el estado inicial de la CPN.
- El Objetivo de *scheduling*: el objetivo se especifica mediante el estado final deseado de la CPN. Es posible utilizar máscaras (“ \* ”) en la especificación de aquella información no conocida del estado final.
- La Función de Costo: para evaluar el costo de alcanzar ciertos estados.
- Simulador: el sistema de apoyo a la toma de decisiones implementado construye y analiza el árbol de alcance de la CPN descrita en el fichero del modelo, y poda el árbol de acuerdo a la función de costo especificada por el usuario. Como resultado, el simulador proporcionará la secuencia obtenida para alcanzar el estado final a partir del estado inicial.

La ejecución de la herramienta da como resultado, en el caso de abrir todo el árbol de alcance, la mejor secuencia de transiciones (actividades) para conducir el sistema desde el estado inicial hasta el estado final (*scheduling*). En el caso del uso de heurísticas la herramienta suele ofrecer resultados bastante próximos al óptimo.

#### 4. ACTUALIZACIÓN DE NODOS OLD

Al usar CPN no temporales, la existencia de un nodo *old* permite ahorrar tiempo y memoria, puesto que no tiene sentido explorar otra vez el mismo sub-árbol y obtener la misma información sobre cómo el sistema puede alcanzar el estado final a partir de un cierto estado. Sin embargo, al trabajar con CPN temporales, la información de tiempo (*time stamps* y reloj global) asociada a cada estado, debe ser procesada cada vez que un nodo *old* se genera para determinar cual de los dos estados debe ser preservado. Tres situaciones diferentes (cada una con diferente complejidad) pueden aparecer:

a) Dos marcados representan la misma información, sin embargo, el reloj global y *time stamps* de todos los tokens del nuevo marcado generado son mayores o iguales que el reloj global y los correspondientes *time stamps* de los tokens del marcado generado anteriormente (nodo *old*). En este caso, se puede garantizar que cualquier solución que pueda ser obtenida a partir del nuevo marcado será siempre peor o igual solución (en lo respecta a tiempos) que aquellas que puedan ser obtenidas a partir del nodo *old*. Así, el algoritmo realizará un retroceso al nodo

padre y continuará la exploración dinámica del espacio de estados mediante la evaluación de otro hijo del nodo padre.

b) Dos marcados representan la misma información, sin embargo, el reloj global y los *time stamps* de uno o más tokens del nuevo marcado generado son menores que el reloj global y los correspondientes *time stamps* de los mismos tokens en el nodo *old*. En este caso, se puede garantizar que cualquier solución que se podría obtener a partir de del nuevo marcado será siempre una solución igual o mejor (en lo que respecta tiempo) que las que se podrían obtener a partir del nodo *old*. Para evitar explorar el mismo sub-árbol el nuevo marcado no se agrega al árbol, puesto que el sub-árbol que se pueda generar a partir del nuevo marcado será el mismo que el que cuelga del nodo *old*, pero con mejores tiempos.

Sin embargo, es necesario actualizar los *time stamps* y el reloj global de aquellos marcados del sub-árbol que puedan conducir a una solución. Para elegir los marcados cuyos valores de tiempo serán recalculados a partir del nuevo marcado es necesario seleccionar, entre los marcados que se generan a partir del nodo *old*, el que menos incrementará el *time stamp* de los tokens hasta llegar a un nodo hoja o a una solución. Para determinar ésto se calcula lo que hemos denominado “holgura de los tokens” del nuevo marcado respecto a todas las transiciones habilitadas, usando la expresión siguiente:

$$H(\text{token}_x) = \text{Nuevo\_reloj} - \text{Time\_stamp\_token}_x$$

Donde:

$H(\text{token}_x)$ : Holgura del token  $x$  que habilita la transición.

*Nuevo\_reloj*: Valor de tiempo del reloj global del nuevo marcado.

*Time\_stamp\_token\_x*: *Time stamp* del token  $x$  en el nuevo marcado.

Una vez que se han calculado las holguras de todos los tokens que habilitan una transición, se selecciona la menor de todas las holguras calculadas. De la misma manera se determina la menor holgura de los tokens que habilitan las diferentes transiciones que se pueden disparar a partir del nuevo marcado, y una vez calculadas todas las menores holguras para todas las transiciones habilitadas, se selecciona como máxima holgura permitida aquella que incrementará menos el tiempo de reloj, es decir, la mayor holgura de todas las menores holguras, seleccionando también aquella transición cuyos tokens que la habilitan originan esa máxima holgura permitida.

Para el marcado que se genera a partir del disparo de la transición seleccionada se calculan los nuevos *time stamps* de los tokens, y será considerado como un nuevo marcado a partir del cual se repite el proceso

de cálculo de las holguras. Este proceso se repite hasta llegar a un nodo hoja o a una solución, y si el valor máximo de entre todos los *time stamps* de los tokens del nuevo estado para esta hoja (o solución) es menor que el valor máximo de entre todos los *time stamps* de los tokens del mismo nodo hoja (o solución) almacenado en el árbol, entonces el padre del nodo *old* se cambiará por el padre del nuevo marcado generado, y los *time stamps* de los tokens de cada marcado en la trayectoria desde el nuevo marcado a la hoja (o solución) serán actualizados.

La actualización de los *time stamps* de los tokens de los marcados generados a partir del nodo *old*, los cuales se encuentran almacenados en el árbol, se realiza, sin disparar las transiciones, a partir del reloj global y los *time stamps* de los tokens del nuevo marcado. Este proceso de actualización no incrementa la necesidad de recursos de memoria, y es muy eficiente en lo que a tiempo de cómputo se refiere.

c) Dos marcados representan la misma información, sin embargo, siendo el reloj global del nuevo marcado menor o mayor (indistintamente) que el reloj global del nodo *old*, los *time stamps* de algunos tokens del nuevo marcado son menores que los correspondientes *time stamps* de los mismos tokens en el nodo *old*, mientras que otros son mayores. En este caso, no se puede decir nada sobre cual de los dos marcados conducirá a una mejor solución.

Considerando que un mismo marcado puede aparecer en el espacio de estados varias veces, el mantenimiento de dos sub-árboles con la misma información de estados podría llegar a ser computacionalmente prohibitivo. Una alternativa posible sería realizar el mismo análisis que en el caso b) pero realizando la actualización sólo sí a partir del nuevo marcado se alcanza un nodo hoja o solución con mejores tiempos, si no es este el caso, el nuevo marcado no se toma en cuenta (caso a)). Este enfoque puede ser considerado eficiente tanto en tiempo de cómputo como en uso de memoria.

Sin embargo, dependiendo de los valores de los *time stamps*, el nuevo marcado podría conducir a otro nodo hoja o solución diferente con menores valores de tiempo que los obtenidos mediante análisis. Esta situación puede ocurrir si un marcado puede conducir a más de un estado final. En este caso la trayectoria que conduce al mejor nodo hoja o solución es almacenada despreciando otras trayectorias.

Cabe notar, que la gestión de los nodos *old* implementada permite un ahorro considerable en la evaluación de marcados que ya habían sido explorados, no actualizando la información de aquellos nodos que no forman parte de las ramas que pueden conducir al óptimo, y garantiza que no se pierde información necesaria para alcanzar soluciones óptimas.

## 5. CASOS DE ESTUDIO

En las siguientes secciones se ilustra el uso de la metodología sobre dos ejemplos tratados en la literatura: un problema clásico de *scheduling* en un sistema de producción Job-Shop, y un ejemplo de optimización para sistemas de manufactura multi-etapas. En ambos ejemplos el tiempo es determinista, a fin de evaluar los resultados sobre problemas *benchmarking*, sin embargo la herramienta soporta la especificación de actividades cuyo tiempo responde a procesos estocásticos (funciones de densidad de probabilidad).

### 5.1 Sistema Benchmarking Job-Shop 6x6.

Este sistema es un problema *benchmarking* bien conocido por la comunidad de optimización (Dauzère and Lasserre, 1994) que consiste en solucionar el problema de programar 6 tareas (*jobs*) en 6 máquinas en un sistema de producción Job-Shop.

La Figura 4 muestra los principales recursos del sistema de producción (6 máquinas de control numérico CNC). La secuencia de máquinas para cada tarea y el tiempo requerido por cada tarea en cada una de las máquinas es conocida (ver tabla 1). El objetivo de optimización consiste en la minimización del *makespan* para finalizar todos los *jobs*. El *makespan* óptimo para este problema es conocido como 55 unidades de tiempo cuando se desecha el tiempo de transporte y se consideran tiempos determinísticos.

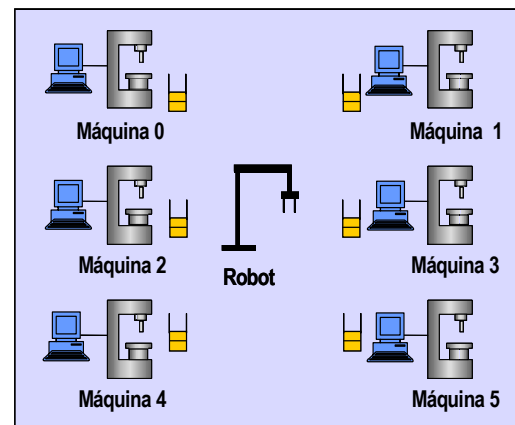


Figura 4. Recursos del problema Job-Shop 6x6.

Tabla 1 Secuencia de operaciones de cada *Job*

Job 1		Job 2		Job 3		Job 4		Job 5		Job 6	
Maq	Tpo	Maq	Tpo	Maq	Tpo	Maq	Tpo	Maq	Tpo	Maq	Tpo
2	1	1	8	2	5	1	5	2	9	1	3
0	3	2	5	3	4	0	5	1	3	3	3
1	6	4	10	5	8	2	5	4	5	5	9
3	7	5	10	0	9	3	3	5	4	0	10
5	3	0	10	1	1	4	8	0	3	4	4
4	6	3	4	4	7	5	9	3	1	2	1

Las tablas 2, 3 y 4 describen los colores, lugares y transiciones del modelo CPN del sistema de producción Job-Shop.

Tabla 2 Descripción de colores del modelo CPN del sistema Job\_Shop 6x6

Color	Definición	Descripción
J	Int 1..6	Identificador de <i>job</i> .
M	Int 0..5	Máquina en la cual la tarea n° O del <i>job</i> J debe ser procesada.
T	Int 1..10	Unidades de tiempo requeridas para procesar cierto <i>job</i> en cierta máquina.
O	Int 1..7	Posición de secuencia del <i>job</i> O en la máquina M.
Ma	Int 0..5	Identificador de máquina
Jo	Product J*M*T*O	Información que describe las características de cada <i>job</i> .

Tabla 3 Descripción de lugares del modelo CPN del sistema Job-Shop 6x6

Lugar	Color	Descripción
Job	Jo	Los tokens almacenados en el lugar Job representan la información asociada con cada <i>job</i> en el sistema de producción: identificador del <i>job</i> , máquina en la cual la próxima tarea debe ser ejecutada, tiempo requerido por cada operación, y el orden de la tarea en la secuencia de operaciones para finalizar el <i>job</i> .
Maq	Ma	Información asociada con cada máquina.

Tabla 4 Descripción de transiciones del modelo CPN del sistema Job-Shop 6x6

Transición	Descripción
T1	Ejecución de una tarea de un <i>job</i> en una máquina.

La CPN del sistema (Figura 5) consiste tan sólo en dos nodos lugar (máquinas y *jobs*) y una transición que representa el procesamiento de un *job* mediante

una máquina. La única precondition para disparar la transición es la disponibilidad de la máquina (expresión de arco  $1'(m)$ ) requerida por el *job* (color  $m$  de la expresión de arco  $1'(j,m,t,o)$ ). Cada vez que la transición es disparada, se actualiza el reloj de simulación junto con la información de tiempo asociada con cada token en el sistema (tanto los tokens que representan las máquinas, como los tokens que representan los *jobs*). Así, aunque la transición está siempre habilitada, el simulador actualiza los tiempos asociados con los tokens de las máquinas incrementando de esta forma el tiempo asociado con los tokens de los *jobs*.

La expresión de arco F2 es usada para actualizar la información del *job* de acuerdo a su estado actual. Si consideramos por ejemplo que el *job* 3 ha finalizado la segunda operación (M=3, O=2) sus colores serán actualizados a J=3, M=5, T=8, O=3 (Tabla 1). De forma similar, puesto que la próxima operación debe ser ejecutada en la máquina 0, los nuevos valores de color serán J=3, M=0, T=9, O=4.

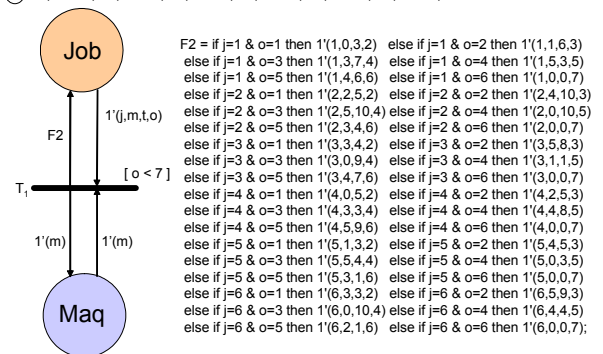
El estado inicial del sistema 6x6 se describe mediante las expresiones subrayadas asociadas con cada nodo lugar, las cuales especifican los colores de los tokens. Así, el lugar Job contiene 6 tokens y cada uno describe un *job* que espera para realizar la primera operación. La raíz del árbol de alcance será el vector:

$$M0=[1'(1,2,1,1)+1'(2,1,8,1)+1'(3,2,5,1)+1'(4,1,5,1)+1'(5,2,9,1)+1'(6,1,3,1),1'(0)+1'(1)+1'(2)+1'(3)+1'(4)+1'(5)].$$

El estado final que se desea alcanzar consiste en que todos los *jobs* hayan finalizado el procesamiento de su última tarea. Este objetivo se especifica mediante el vector:

$$Mf=[1'(1,0,0,7)+1'(2,0,0,7)+1'(3,0,0,7)+1'(4,0,0,7)+1'(5,0,0,7)+1'(6,0,0,7),1'(0)+1'(1)+1'(2)+1'(3)+1'(4)+1'(5)].$$

$$\textcircled{6} \quad 1'(1,2,1,1)+1'(2,1,8,1)+1'(3,2,5,1)+1'(4,1,5,1)+1'(5,2,9,1)+1'(6,1,3,1)$$



$$\textcircled{6} \quad 1'(0)+1'(1)+1'(2)+1'(3)+1'(4)+1'(5)$$

Figura 5. CPN del sistema Job-Shop 6x6.



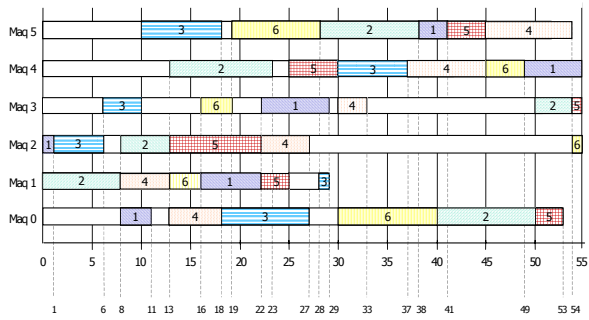


Figura 6. Diagrama de Gantt del *scheduling* solución.

La Figura 6 ilustra mediante un diagrama de Gantt los resultados obtenidos.

### 5.2 Planificación de la Producción para Sistemas de Manufactura Multi-Etapas.

El estudio descrito en (Hennet, 2001) formula un problema cíclico de *scheduling* por lotes para fabricación multi-etapas en un sistema de producción Job-Shop. Este tipo de problema consiste en la organización de la producción en sistemas de fabricación de familias de productos a partir de ciertos productos intermedios comunes, a través de varias etapas de procesamiento y ensamblaje. La naturaleza cíclica del problema se obtiene bajo el supuesto de tasas de demanda constantes para todos los productos finales, y la hipótesis de un ciclo común para todos los productos (primarios, intermedios y finales). La descripción de un producto en el sistema se obtiene mediante un análisis entrada-salida de la estructura de los productos: productos primarios son transformados en productos finales a través de varias etapas que combinan y transforman los productos en las diferentes etapas.

Este problema es considerado NP-Hard, y la búsqueda de un procedimiento óptimo es aún un problema de estudio. La Figura 7 muestra un grafo que describe dos trabajos genéricos: uno asociado con el producto final D, y compuesto de las tareas 2, 4, 6, 9, 10, y otro asociado con el producto final E, y compuesto por las tareas 1, 3, 5, 7, 8.

En el grafo, los arcos corresponden a las restricciones de precedencia, y los nodos representan las restricciones de los recursos compartidos: el recurso 1 es usado para fabricar los productos finales D y E, el recurso 2 es utilizado para fabricar tanto el producto primario B como el producto intermedio C, y el recurso 3 es usado para fabricar el producto primario A. Los pesos en cada arco representan la cantidad de productos primarios e intermedios que se necesitan para fabricar una unidad de producto final D y una unidad de producto final E respectivamente.

El estado final para este problema consiste en minimizar la duración total del plan genérico, es decir, el *makespan* del período genérico.

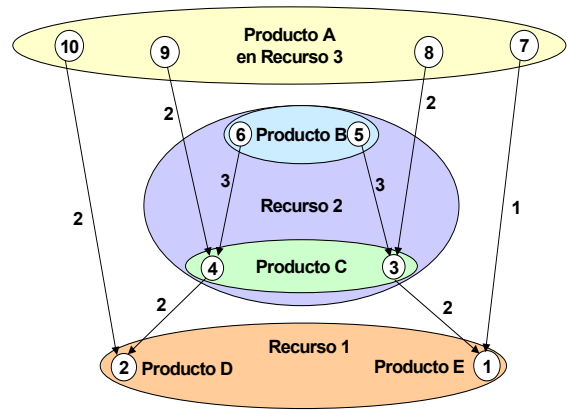


Figura 7. Grafo de una estructura de producción multi-etapas.

Las tablas 5, 6 y 7 describen la semántica del modelo CPN que permite simular este sistema de producción, a fin de encontrar un *scheduling* óptimo utilizando la herramienta desarrollada, que permita fabricar 5 unidades de producto D y 8 unidades de producto E.

Tabla 5 Descripción de lugares del modelo CPN del sistema de producción multi-etapas

Lugar	Color	Descripción
P	Mp	Representa la información asociada a la cantidad de materia prima requerida para fabricar el producto final.
A	Pa	Representa la información asociada al producto A: identificación de la tarea mediante la cual se fabrica (7..10) y producto fabricado (1) o no fabricado (0).
B	Pb	Representa la información asociada al producto B: identificación de la tarea mediante la cual se fabrica (5,6) y producto fabricado (1) o no fabricado (0).
C	Pc	Representa la información asociada al producto C: identificación de la tarea mediante la cual se fabrica (3,4) y producto fabricado (1) o no fabricado (0).
F	Pf	Representa la información asociada al producto final: identificación de la tarea mediante la cual se fabrica (1,2) y producto fabricado (1) o no fabricado (0).
M	Ma	Representa la información asociada a la máquina o recurso de procesamiento: recurso 1 (1), recurso 2 (2), recurso 3 (3).

**Tabla 6 Descripción de colores del modelo CPN del sistema de producción multi-etapas**

Color	Definición	Descripción
MP	Int 8,10,20, 30,32,48	Cantidad de productos primarios (materia prima) necesaria para fabricar 5 unidades de producto final D y 8 unidades de producto final E.
TA	Int 7..10	Identificador de la tarea (7, 8, 9 ó 10) realizada para fabricar el producto A.
FA	Int 0..1	Indica si el producto A ha sido fabricado mediante la tarea cuyo valor corresponde al identificador de tarea TA (0: no ha sido fabricado por la tarea TA, 1: ha sido fabricado por la tarea TA).
TB	Int 5,6	Identificador de la tarea (5 ó 6) realizada para fabricar el producto B.
FB	Int 0..1	Indica si el producto B ha sido fabricado mediante la tarea TB (0: no ha sido fabricado por la tarea TB, 1: ha sido fabricado por la tarea TB).
TC	Int 3,4	Identificador de la tarea (3 ó 4) realizada para fabricar el producto C.
FC	Int 0..1	Indica si el producto C ha sido fabricado mediante la tarea TC (0: no ha sido fabricado por la tarea TC, 1: ha sido fabricado por la tarea TC).
TF	Int 1,2	Identificador de la tarea (1 ó 2) realizada para fabricar el producto final.
FF	Int 0..1	Indica si el producto final ha sido fabricado mediante la tarea TF (0: no ha sido fabricado por la tarea TF, 1: ha sido fabricado por la tarea TF).
Pa	Product TA,FA	Información sobre producto A.
Pb	Product TB,FB	Información sobre producto B.
Pc	Product TC,FC	Información sobre producto C.
Pf	Product TF,FF	Información sobre producto final.
Ma	int 1..3	Identificador de recurso (máquinas).

**Tabla 7 Descripción de transiciones del modelo CPN del sistema de producción multi-etapas**

Transición	Descripción
T1	Fabricación de producto A.
T2	Fabricación de producto B.
T3	Fabricación de producto C.
T4	Fabricación de producto final.

Las Figuras 8, 9, 10 y 11 muestran los diferentes subsistemas que componen el modelo del sistema de producción. El esquema de producción obtenido por el DSS se muestra en la Figura 12. El valor del *makespan* obtenido mediante el método propuesto es de 800 unidades de tiempo, el cual mejora el resultado disponible en la literatura (Hennet, 2001) de 830.

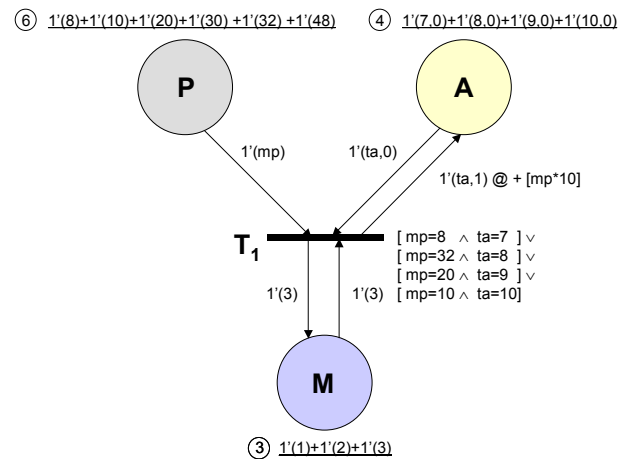


Figura 8. Subsistema de Fabricación de Producto A.

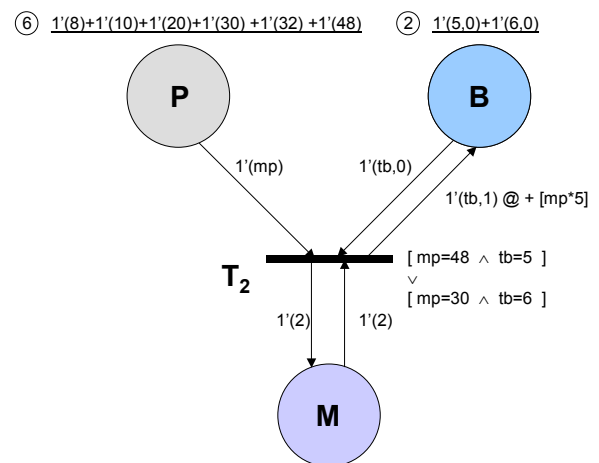


Figura 9. Subsistema de Fabricación de Producto B.

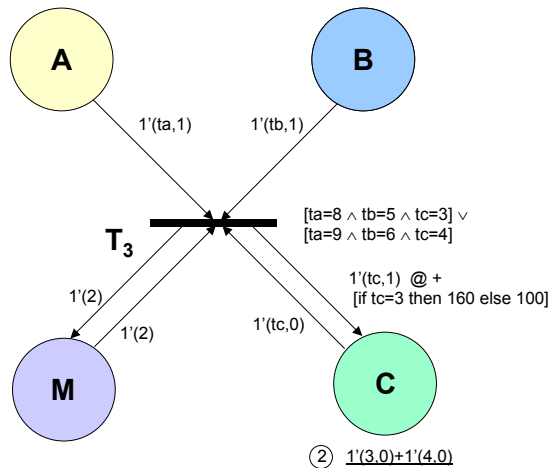


Figura 10. Subsistema de Fabricación de Producto C.

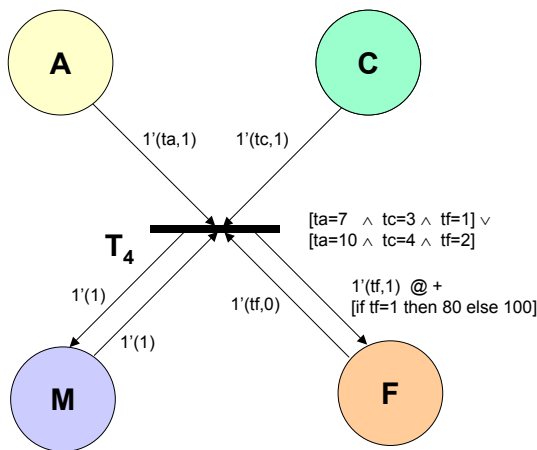


Figura 11. Subsistema de Fabricación de Producto Final.

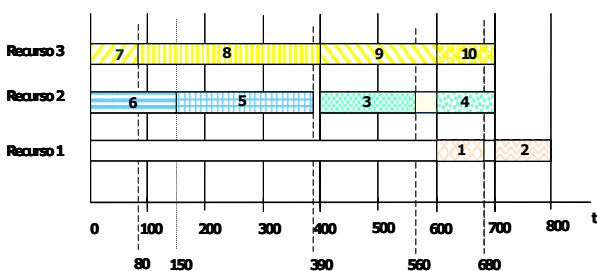


Figura 12. Diagrama de Gantt del esquema de producción obtenido.

6. CONCLUSIONES

Un Sistema de Apoyo a la Toma de Decisiones basado en un nuevo enfoque de Optimización mediante el uso de Simulación ha sido desarrollado. El enfoque de generación de un plan a partir de la

exploración del espacio de estados, implementado mediante el árbol de alcance de una Red de Petri Coloreada, permite generar políticas de *scheduling* a fin de satisfacer ciertos objetivos de funcionamiento de sistemas logísticos complejos.

El elevado número de variables de decisión en los actuales sistemas logísticos, suele dar origen a un espacio de estados de tamaño considerable, lo cual hace prácticamente imposible su tratamiento computacional. El DSS ha sido desarrollado de forma tal que diferentes algoritmos heurísticos pueden ser implementados para guiar al simulador de CPN desde un estado inicial del sistema modelado hasta un estado final deseado.

AGRADECIMIENTOS

Los autores quieren agradecer el apoyo financiero recibido del programa Español CICYT DPI2004-08056-C03-01.

REFERENCIAS

Andersson, D. and A. Norrman (2002). Procurement of logistics services – a minutes work or a multi-year project?. *European Journal of Purchasing & Supply Management*, Vol. 8(1), 3 - 14.

Carson, Y. and A. Maria (1997). Simulation Optimization: Methods and Applications. *Proceedings of the 29th conference on Winter simulation, WSC'97*, 118 -126.

Dauzère-Péres, S. and J-B. Lasserre (1994). An Integrated Approach in Production Planning and Scheduling, En: *Lecture Notes and Mathematical Systems*. Springer-Verlag, Berlin.

Glover, F., J. P. Kelly and M. Laguna (1999). New Advances for wedding optimization and Simulation. *Proceedings of the 31st conference on Winter simulation: Simulation---a bridge to the future, WSC'99*, Vol. 1, 255 - 260.

Hennet, J-C. (2001). A Common Cycle Approach to Lot-Scheduling in Multistage Manufacturing Systems. *Production Planning & Control*, Vol. 12 (4), 362 - 371.

Jensen, K. (1997). *Coloured Petri Nets: Basics Concepts, Analysis Methods and Practical Use, Vol. 1,2,3*. Springer-Verlag, Berlin.

Law, M. and M. G. McComas, (2002). Simulation Based optimisation. *Proceedings of the 2002 Winter Simulation Conference, WSC'02*, 41- 44.

Narciso, M.E. and M.A. Piera (2001). Coloured Petri Net Simulator: A Generic Tool for Production Planning. *Proceedings of the 2001 8th IEEE International Conference on Emerging Technologies and Factory Automation, ETFA'2001*, Vol. 2, 139 - 147.

Pichitlamken, J. and B. L. Nelson (2201). Selection of the best procedures for optimization via simulation. *Proceedings of the 2001 Winter Simulation Conference, WSC'01*, 401 - 407.

- Piera, M.A. and T. Guasch (2001). Coloured Petri Net for Simulation FMS Model Maintenance. *Proceedings of the 4th International Eurosim 2001 Congress: Shaping Future with Simulation*.
- Silva, M. and R. Valette (1989). Petri Nets and Flexible Manufacturing. En: *Lecture Notes in Computer Science, Advances in Petri Nets*, **Vol. 424**, 374 - 417.
- Zeithaml, V.A., A. Parasuraman and L.L. Berry (1985). Problems and Strategies in Service Marketing. *Journal of Marketing*, **Vol. 49**. 33 - 46.
- Zimmermann, A., K. Dalkowski and G. Hommel (1996). A Case Study In Modelling And Performance Evaluation Of Manufacturing Systems Using Colored Petri Nets. *Proceedings of the 8th European Simulation Symposium, ESS '96*, 282 - 286.