Evaluación de Prestaciones de Sistemas Concurrentes Modelados con Redes de Petri

Javier Campos

Dpto. de Informática e Ingeniería de Sistemas, Universidad de Zaragoza
http://www.cps.unizar.es/~jcampos/

1 Introducción

La evaluación del rendimiento de un sistema paralelo y distribuido es tanto una necesidad práctica como un reto científico. El rendimiento, o *prestaciones*, se caracteriza mediante un conjunto de descriptores de eficiencia precisos que ayudan a determinar cuánto de cerca está un sistema de los objetivos previstos para el mismo [1].

El *modelado cuantitativo* para el análisis de las prestaciones intenta describir, predecir y optimizar el comportamiento dinámico y dependiente del tiempo de los sistemas. Es un hecho ampliamente reconocido que la complejidad de los sistemas paralelos y distribuidos es tal que se precisan *herramientas formales* apropiadas durante la etapa de diseño para poder garantizar las especificaciones cualitativas y cuantitativas de partida.

Las *redes de Petri* (RdP) [2] son un conocido formalismo para describir sistemas de eventos discretos con sincronizaciones. En su definición original, las RdP no incluyeron la noción del tiempo pues trataron de modelar únicamente el comportamiento lógico de los sistemas, describiendo las relaciones causales existentes entre eventos. Si se quiere obtener un modelo útil para la evaluación de prestaciones de sistemas distribuidos, entonces la introducción de una especificación temporal es esencial, conduciendo a los modelos denominados *RdP estocásticas* (RdPE) [3].

El objetivo que se persigue al caracterizar el comportamiento de los modelos de RdPE es obtener los *índices de prestaciones* del modelo mediante *técnicas eficientes de cálculo*. Las técnicas de análisis de prestaciones pueden clasificarse de acuerdo con la calidad de los resultados obtenidos en: exactas, aproximadas y acotaciones. En el ámbito del análisis de modelos basados en RdPE pueden encontrarse técnicas para los tres tipos de resultados. Por otra parte, dada la *capacidad de abstracción* de las RdP (para las que un modelo compacto puede describir un enorme espacio de estados y las transiciones entre los mismos), pueden encontrarse *técnicas de análisis enumerativas*, es decir, que requieren el cómputo del espacio de estados del modelo, y *técnicas estructurales*, más eficientes, que pueden realizarse con la información de la estructura del modelo, sin necesidad de conocer (calcular) su espacio de estados.

En los párrafos siguientes describiremos brevemente los índices habituales de prestaciones que se analizan mediante RdPE y las técnicas más utilizadas

actualmente para su cálculo: el *análisis exacto enumerativo*, que es el más frecuentemente disponible en las *herramientas software* desarrolladas para analizar RdPE y el *análisis exacto mediante técnicas tensoriales*, que pretende paliar el problema del coste prohibitivo en espacio del método exacto enumerativo.

Con respecto al resto de técnicas disponibles, el *análisis aproximado* y el *cálculo de cotas*, nos limitaremos, por las restricciones de espacio en este documento, a incluir alguna referencia bibliográfica.

La organización del documento es como sigue. En la sección 2 se incluye la terminología y conceptos básicos sobre evaluación de prestaciones. La sección 3 presenta la interpretación de los modelos autónomos de RdP con aspectos de tiempo (duración de los eventos) y probabilidad (resolución de conflictos). En la sección 4 se describe la técnica de análisis exacto basada en la resolución de la *cadena de Markov en tiempo continuo* (CMTC) isomorfa a la RdPE. La sección 5 incluye una breve descripción de la mejora basada en técnicas tensoriales para el análisis exacto.

2 Evaluación de prestaciones

El análisis cualitativo o validación de propiedades lógicas de un sistema paralelo y distribuido se plantea los problemas de *corrección*, *ausencia de bloqueos*, *vivacidad*... Por contra, la evaluación de prestaciones plantea cuestiones de tipo cuantitativo sobre el funcionamiento del sistema: ¿cómo de *rápido* realiza las tareas?, ¿con qué *aprovechamiento* utiliza los recursos?, o, en el caso en que el sistema no funciona, ¿con qué probabilidad *falla*?, ¿cuánto tiempo *tarda* en fallar?

Las cuestiones de tipo cuantitativo acerca del sistema deben concretarse en el cálculo de ciertos *índices de rendimiento*. Los índices de rendimiento (o índices de prestaciones) son descriptores cuantificables usados para representar algún aspecto del rendimiento del sistema. Pueden representar medidas de eficiencia, de utilización, de fiabilidad, de disponibilidad...

Por ejemplo, si nos centramos en índices de productividad, el *throughput*[1] representa el volumen de información procesado por el sistema por unidad de tiempo. Si, por el contrario, nos preocupa la capacidad de respuesta del sistema, el índice *tiempo de respuesta* mide el tiempo transcurrido desde que se efectúa una entrada de datos al sistema hasta que se produce la respuesta correspondiente. Si es la utilización de una componente concreta del sistema la faceta de interés, el índice *utilización* hace referencia al porcentaje de tiempo en que esa componente del sistema se usa.

Una vez precisados cuáles son los índices de rendimiento que se desea evaluar, una *técnica de evaluación* tiene por objeto el calcular dichos índices partiendo de la observación del sistema real (técnicas de *medida*) o bien utilizando un *modelo* simplificado del sistema y una cantidad de carga del mismo cercana a la prevista (técnicas de *simulación* o técnicas de *modelado analítico*).

[1] Optaremos por usar el término inglés, dada su difícil traducción.

En las técnicas basadas en modelos, se maneja una visión abstracta simplificada del sistema real bajo estudio en la que únicamente quedan reflejadas las características que interesan del mismo. Se manipula el modelo (por ejemplo, si el modelo es un programa de simulación, se ejecuta dicho programa), se observa su comportamiento y se infiere el comportamiento del sistema real en circunstancias análogas.

En el caso de los modelos analíticos, se maneja una descripción matemática del comportamiento del sistema para obtener información sobre la "solución"(índices de rendimiento) por métodos matemáticos. Utilizando modelos analíticos, el dominio de problemas "tratables" es limitado. Esta técnica puede ser inadecuada, por tanto, para el estudio de sistemas con gran detalle. No obstante, un modelo analítico facilita una buena visión del comportamiento global del sistema y, si el modelo ha sido simplificado lo suficiente, puede resolverse analíticamente y el resultado tiene un buen valor predictivo.

Existen varios formalismos matemáticos orientados hacia la evaluación de prestaciones. Los más conocidos son, por orden histórico de aparición, las *cadenas de Markov* [4], las *redes de colas* [5], las *redes de Petri estocásticas* [6] y las *álgebras de procesos estocásticas* [7]. Las cadenas de Markov son un caso particular de proceso estocástico especialmente adecuado para el modelado y análisis de sistemas con espacio de estados discreto y que evolucionan en el tiempo. Sin embargo, su pobre capacidad de abstracción —cada estado del sistema se representa con un "elemento atómico" en el modelo—, hace que desde un punto de vista práctico resulten inadecuadas, al menos como lenguaje de especificación del sistema, si bien sus técnicas de análisis son útiles también para el estudio de los formalismos posteriores. Las redes de colas han venido siendo utilizadas en la práctica desde los años 60 para el modelado y análisis de sistemas informáticos. Su mayor nivel de abstracción, en comparación con las cadenas de Markov, las hicieron preferibles frente a aquéllas. Además se desarrollaron técnicas de análisis estructural ("soluciones en forma producto") que permiten su estudio sin necesidad de calcular el espacio de estados representados, con el consiguiente ahorro computacional en tiempo y espacio. No obstante, los modelos básicos de redes de colas carecen de una primitiva general para modelar la *sincronización*, y las extensiones que la incluyen carecen de técnicas de análisis eficientes, por lo que dejan de ser adecuadas para el modelado de sistemas concurrentes. Hoy por hoy, son las álgebras de procesos estocásticas y las redes de Petri estocásticas los formalismos que prevalecen en el ámbito del modelado y análisis de sistemas concurrentes, tanto en los aspectos de validación de propiedades lógicas como en los de análisis de prestaciones. En los párrafos siguientes, nos centraremos en el segundo de esos formalismos, y en los aspectos de análisis de prestaciones.

3 Redes de Petri estocásticas

Las redes de Petri (RdP) son un paradigma de modelado de sistemas de eventos discretos concurrentes. Se trata de un modelo formal (matemático) para describir estados y acciones. Disponen de una representación gráfica pues, de hecho,

se definen como un grafo con dos clases de nodos, lugares y transiciones. Guardan cierta similitud con los modelos clásicos de colas puesto que definen una *representación distribuida de un estado*.

Las características fundamentales de las RdP son su *simplicidad* (intervienen muy pocas y simples entidades matemáticas en la definición), su *generalidad* (posibilidad de modelar secuencias, decisiones, concurrencia, sincronizaciones, etc.), su *adecuación* (capacidad de expresar todas las semánticas básicas de la concurrencia: entrelazado, semántica de pasos, semántica de orden parcial), y su *localidad de estados y acciones* (posibilidad de modelado progresivo, por refinamientos sucesivos o por composición modular).

Son valores adicionales de las RdP su posibilidad de *representación gráfica*, la existencia de *técnicas de validación* de propiedades del sistema (como la vivacidad, la limitación, la ausencia de bloqueos...), la posibilidad de *interpretación estocástica* (que las convierte en un formalismo válido para la evaluación del rendimiento), y la existencia de *herramientas software* de diseño y análisis que implementan las técnicas conocidas.

Asumimos que el lector ya está familiarizado con los conceptos básicos sobre RdP [2,8,9]. No obstante, resumimos brevemente las definiciones básicas y la notación habitual.

Una *red de Petri* (RdP) es una 4–tupla $\mathcal{N} = \langle P, T, \mathbf{Pre}, \mathbf{Post} \rangle$, donde P y T son conjuntos disjuntos de *lugares* y *transiciones*, y $\mathbf{Pre}$ y $\mathbf{Post}$ son las funciones de incidencia previa y posterior que representan (en forma vectorial) los arcos de entrada y salida: $\mathbf{Pre}[p,t] \in \mathbb{N}$ y $\mathbf{Post}[p,t] \in \mathbb{N}$. Una RdP puede verse como un grafo dirigido bipartito en el que los lugares y las transiciones son las dos clases de nodos. Los lugares se representan con círculos y las transiciones con barras o cajas. Las RdP *ordinarias* son aquéllas cuyas funciones de incidencia previa y posterior toman valores en $\{0,1\}$.

La función de incidencia de un arco dado en una red no ordinaria se llama *peso* o *multiplicidad*. Los *pre-* y *post-conjunto* de una transición $t \in T$ se definen, respectivamente, como ${}^\bullet t = \{p \mid \mathbf{Pre}[p,t] > 0\}$ y $t^\bullet = \{p \mid \mathbf{Post}[p,t] > 0\}$. Los *pre-* y *post-conjunto* de un lugar $p \in P$ se definen, respectivamente, como ${}^\bullet p = \{t \mid \mathbf{Post}[p,t] > 0\}$ y $p^\bullet = \{t \mid \mathbf{Pre}[p,t] > 0\}$. La transición t es una *sincronización* (respectivamente, *inicio múltiple*) si $|{}^\bullet t| > 1$ (respectivamente, $|t^\bullet| > 1$). La *matriz de incidencia* de la red se define como $\mathbf{C} = \mathbf{Post} - \mathbf{Pre}$.

Se llama *marcado* a una función $\mathbf{m}\colon P \to \mathbb{N}$ (normalmente representada en forma vectorial). Un *sistema*, o *red de Petri marcada*, $\mathcal{S}$, es una red de Petri $\mathcal{N}$ con un *marcado inicial* $\mathbf{m_0}$. Una transición $t \in T$ está *sensibilizada* en un marcado $\mathbf{m}$ si $\forall p \in P$: $\mathbf{m}[p] \geq \mathbf{Pre}[p,t]$. Una transición t sensibilizada en $\mathbf{m}$ puede *dispararse*, obteniéndose un nuevo marcado $\mathbf{m'}$ definido por $\mathbf{m'}[p] = \mathbf{m}[p] - \mathbf{Pre}[p,t] + \mathbf{Post}[p,t]$ (se denota como $\mathbf{m} \xrightarrow{t} \mathbf{m'}$). Una secuencia de transiciones $\sigma = t_1 t_2 \ldots t_n$ es una *secuencia de disparos* en $\mathcal{S}$ si existe una secuencia de marcados tal que $\mathbf{m_0} \xrightarrow{t_1} \mathbf{m}_1 \xrightarrow{t_2} \mathbf{m}_2 \ldots \xrightarrow{t_n} \mathbf{m}_n$. En este caso, el marcado $\mathbf{m}_n$ se llama *alcanzable* desde $\mathbf{m_0}$ disparando σ, y se denota como $\mathbf{m_0} \xrightarrow{\sigma} \mathbf{m}_n$. El *conjunto de alcanzabilidad* $\mathrm{RS}(\mathcal{S})$ de un sistema $\mathcal{S} = \langle \mathcal{N}, \mathbf{m_0} \rangle$ es el conjunto

de todos los marcados alcanzables desde el marcado inicial. $\mathrm{L}(\mathcal{S})$ es el *lenguaje de secuencias de disparo* de un sistema $\mathcal{S} = \langle \mathcal{N}, \mathbf{m_0} \rangle$ ($\mathrm{L}(\mathcal{S}) = \{\sigma \mid \mathbf{m_0} \xrightarrow{\sigma} \mathbf{m}\}$).

Un lugar $p \in P$ se llama *k–limitado* en $\mathcal{S}$ si $\forall \mathbf{m} \in \mathrm{RS}(\mathcal{S})$, $\mathbf{m}[p] \leq k$. Un sistema se llama k–limitado si todo lugar es k–limitado, y *limitado* si existe algún k para el que es k–limitado. Una red es *estructuralmente limitada* si es limitada para algún $\mathbf{m_0}$. Un sistema es *vivo* si toda transición puede llegar a ser disparada desde cualquier marcado alcanzable. Un estado $\mathbf{m}$ se llama *estado hogar* si es alcanzable desde todos los marcados alcanzables.

Una *red de Petri estocástica* (RdPE) es una RdP con una *interpretación temporal* que reduce el no-determinismo inherente en el modelo autónomo y le hace adecuado para definir y evaluar índices de prestaciones. La interpretación temporal debe incluir la asociación de una *duración temporal* a las actividades (transiciones) y también una *política de resolución de conflictos*. Con respecto a la duración de actividades, se suele asociar una variable aleatoria a cada transición, que modela su *tiempo de servicio*. Denotamos con $\mathbf{s}[t]$ el *tiempo medio de servicio* de la transición t. Con respecto a la resolución de conflictos, se pueden asociar *tasas de encaminamiento* a cada subconjunto de transiciones en conflicto, de manera que quede definida una distribución discreta de probabilidad para elegir la transición que debe dispararse al presentarse el conflicto. Las tasas de encaminamiento pueden definirse explícitamente mediante una anotación asociada a cada *transición inmediata* (transición cuyo tiempo de servicio es nulo y sirve para modelar decisiones), o bien pueden calcularse para el caso de *transiciones temporizadas* (que tienen un tiempo de servicio no nulo) usando una *política de competición* (la primera transición que termina su actividad es la elegida para ser disparada).

Una *red de Petri markoviana* (RdPM) [3] es un caso particular de RdPE en el que los tiempos de servicio son variables aleatorias independientes y con distribución de probabilidad exponencial (con tasa $\lambda_i = 1/\mathbf{s}[t_i]$ para cada transición t_i), y las tasas de encaminamiento se definen mediante la política de competición.

Las *redes de Petri estocásticas generalizadas* (RdPEG) [11,6] son una extensión de las RdPM que incluyen, entre otros aspectos que evitaremos en este documento introductorio, transiciones inmediatas que pueden usarse para modelar conflictos con unas tasas de encaminamiento determinadas (separando así la política de resolución de conflictos de la duración de las actividades).

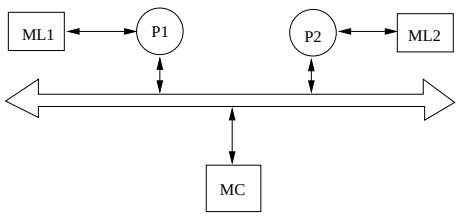

Figura 1. Arquitectura multiprocesador con memoria compartida.

Consideremos, por ejemplo, un sencillo sistema multiprocesador [12] en el que dos procesadores deben acceder en ocasiones a una memoria compartida (véase la figura 1). Se asume que el comportamiento de ambos procesadores es idéntico: una secuencia cíclica de actividad local, seguida de una petición de acceso a la memoria compartida y, por último, del acceso a dicha memoria. Suponiendo que todas estas acciones requieren un tiempo de ejecución distribuido exponencialmente, la RdPM de la figura 2 modela la arquitectura.

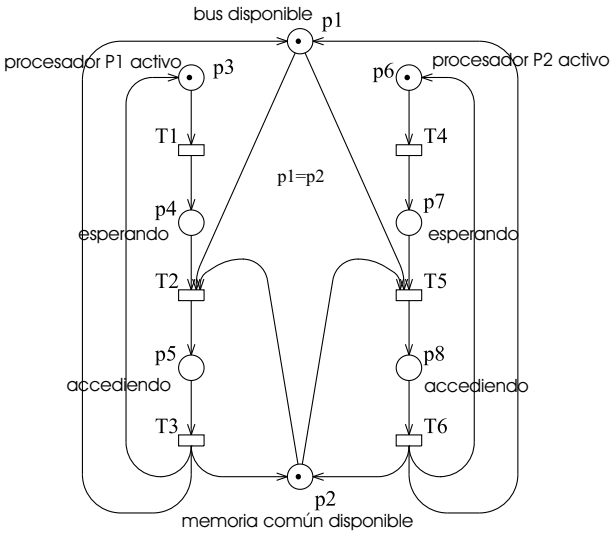

Figura 2. Un ejemplo de RdPM modelando la arquitectura de la figura 1 (los lugares p_1 y p_2 tienen idéntica función de incidencia, por lo que uno de ellos puede eliminarse).

Si las transiciones que modelan la acción de *petición de acceso a la memoria compartida* se consideran instantáneas, entonces el modelo obtenido es una RdPEG como la mostrada en la figura 3. Las transiciones inmediatas se representan con barras (t_2 y t_5) mientras que las temporizadas se representan con cajas (T_1, T_3, T_4 y T_6).

Si nos centramos en técnicas de análisis, se conocen algoritmos eficientes para el cálculo de la distribución del estado estacionario del modelo únicamente para algunas subclases muy particulares de RdPE que poseen la denominada *solución con forma producto* [13,14]. Por tanto, en general, para obtener los *índices exactos* de prestaciones debe emplearse un método numérico para resolver una *cadena de Markov en tiempo continuo* (CMTC) isomorfa a la red [3,6]. Si bien es ésta la técnica más utilizada y más fácilmente implementable en herramientas software (e.g., [15]), el problema de la *explosión de estados* (crecimiento asintótico exponencial del espacio de estados en relación con el número de lugares y transiciones de la red) puede limitar su utilidad para el análisis de modelos grandes, tanto

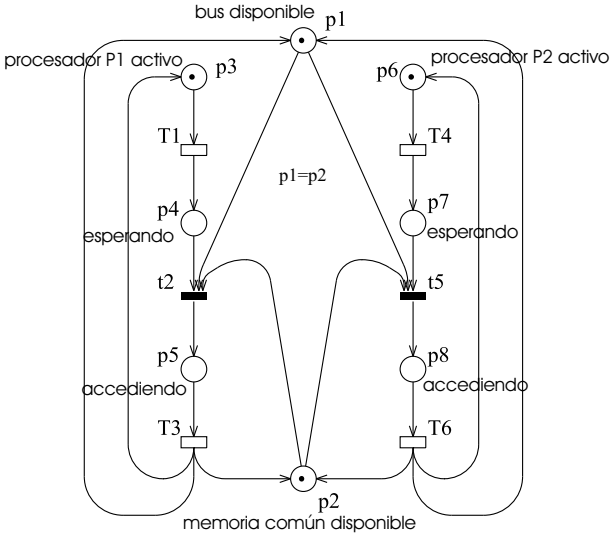

Figura 3. Un ejemplo de RdPEG modelando la arquitectura de la figura 1.

por el coste de almacenar la *matriz generadora infinitesimal* de la CMTC como por la complejidad temporal de los algoritmos de resolución.

Las *técnicas dirigidas por la estructura de la red* [16] tratan de reducir la complejidad en espacio y en tiempo de los algoritmos de solución, mediante la utilización de información extraída de la estructura del modelo. Hay múltiples ejemplos de técnicas dirigidas por la estructura, como las de eliminación de transiciones inmediatas [17], las de utilización de simetrías en RdPEG *coloreadas* [18], o las basadas en programación lineal para el cálculo de cotas de índices de rendimiento [19,20,21,22,23,24]. La estructura de la red se ha utilizado también en técnicas aproximadas de *divide y vencerás* para limitar el problema de la explosión de estados [25,26,27]. Igualmente, la estructura de la red resulta fundamental en las técnicas de *álgebra tensorial* [28] que permiten expresar la matriz generadora infinitesimal de la CMTC en términos de matrices más pequeñas obtenidas para las *componentes* (subredes con espacio de estados más pequeño), y así calcular la distribución del estado estacionario sin calcular ni almacenar la matriz completa [29].

4 Análisis exacto enumerativo

Volvamos de nuevo a la RdPM de la figura 2 que modela el sistema de memoria compartida. Partiendo del marcado inicial mostrado en la figura en el que ambos procesadores están activos localmente y no se está accediendo a la memoria común, una evolución posible del sistema es la siguiente:

- el procesador 1 trabaja localmente un tiempo distribuido exponencialmente con media $1/\lambda_1$ (duración asociada a la transición T_1) y a continuación solicita acceder a la memoria común;
- se dispara la transición T_1 y la marca del lugar p_3 ("procesador P1 activo") pasa al lugar p_4 ("esperando");
- como la memoria está disponible (está marcado el lugar p_2) la acción de petición de la memoria empieza inmediatamente y lleva un tiempo de media $1/\lambda_2$ (representado por la transición T_2);
- cuando la transición T_2 se dispara, desaparece la marca del lugar p_4 y se añade al lugar p_5 ("accediendo"), donde permanece un tiempo de media $1/\lambda_3$ requerido por el primer procesador para usar la memoria compartida;
- cuando ese tiempo transcurre, se dispara la transición T_3 y se vuelve al estado inicial.

El ciclo del procesador 2 es similar. Ambos ciclos de actividad (de los procesadores 1 y 2) se *entrelazan* y pueden describirse perfectamente con el grafo de alcanzabilidad de la red.

Si en el grafo de alcanzabilidad se etiquetan los arcos con las tasas de las distribuciones exponenciales en lugar de con los nombres de las transiciones correspondientes, el grafo se denomina *diagrama de transición de estados* de la cadena de Markov en tiempo continuo (CMTC) asociada a la RdPM (véase la figura 4).

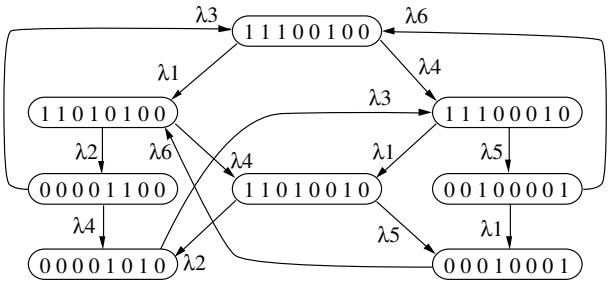

Figura 4. Diagrama de transición de estados de la RdPM de la figura 2.

Nótese que en el estado $(1, 1, 0, 1, 0, 0, 1, 0)$, en el que ambos procesadores quieren acceder a la memoria común (es decir, están sensibilizadas las transiciones T_2 y T_5), existe un *conflicto*. Como ya se ha comentado, el conflicto se resuelve mediante la política de competición, es decir, la transición que termina primero es la que se dispara. La probabilidad de que se dispare T_2 es:

$$\Pr\{T_2\} = \frac{\lambda_2}{\lambda_2 + \lambda_5}$$

Mientras que la probabilidad de que se dispare T_5 es:

$$\Pr\{T_5\} = \frac{\lambda_5}{\lambda_2 + \lambda_5}$$

Nótese, igualmente, que cuando las dos transiciones T_2 y T_5 están sensibilizadas en el marcado $\mathbf{m} = (1, 1, 0, 1, 0, 0, 1, 0)$, la velocidad con la que el sistema sale del estado $\mathbf{m}$ es la suma de las velocidades individuales de ambas transiciones $\lambda_2 + \lambda_5$.

Si la RdPM es limitada y tiene estado hogar, la CMTC asociada a ella es *ergódica*, en cuyo caso es posible calcular la *distribución en estado estacionario* de los marcados, resolviendo el siguiente sistema de ecuaciones lineales [3,6]:

$$\begin{aligned} \boldsymbol{\pi} \cdot \mathbf{Q} &= \mathbf{0} \\ \boldsymbol{\pi} \cdot \mathbf{1} &= 1 \end{aligned} \tag{1}$$

en el que $\boldsymbol{\pi}$ es el vector de la distribución estacionaria (es decir, tiene tantos elementos como marcados alcanzables), $\mathbf{0}$ es un vector de igual tamaño que $\boldsymbol{\pi}$ y con todos los elementos nulos, $\mathbf{1}$ es un vector (de la misma dimensión) con todas sus componentes igual a 1, y la matriz $\mathbf{Q}$, llamada *generador infinitesimal* de la CMTC, es la siguiente:

$$\begin{aligned} q_{ij} &= \textstyle\sum_{\mathbf{m}_i \xrightarrow{t_k} \mathbf{m}_j} \lambda_k, \text{ si } i \neq j \\ q_{ii} &= -\textstyle\sum_{j \neq i} q_{ij} \end{aligned} \tag{2}$$

Es decir, el elemento q_{ij} de la matriz $\mathbf{Q}$ es la suma de las tasas de las transiciones que pueden llevar al modelo desde el estado $\mathbf{m}_i$ al estado $\mathbf{m}_j$ ("velocidad" de salto de $\mathbf{m}_i$ a $\mathbf{m}_j$).

A partir de la distribución en estado estacionario es posible el cálculo de los índices de rendimiento del modelo. Existe una aproximación general para la definición de índices de rendimiento basada en el concepto de *función de ganancia* [6]. Las funciones de ganancia se definen sobre los marcados alcanzables, de manera que la ganancia media (calculada para la distribución estacionaria de marcado) constituya el índice de rendimiento deseado. Es decir, si se considera la función de ganancia $r(\mathbf{m})$, la ganancia media es la siguiente:

$$R = \sum_{\mathbf{m}_i \in \mathrm{RS}(\mathbf{m_0})} r(\mathbf{m}_i)\pi_i \tag{3}$$

Por ejemplo, si el índice que se desea calcular es el *marcado medio* $\overline{\boldsymbol{\mu}}_i$ de un lugar p_i, (valor medio del número de marcas), que puede aportar información sobre la utilización media de un recurso o el grado medio de ocupación de un almacén, basta con definir la ganancia $r_{\overline{\boldsymbol{\mu}}_i}$ como el número de marcas en ese lugar p_i:

$$r_{\overline{\boldsymbol{\mu}}_i}(\mathbf{m}) = \mathbf{m}[p_i] \tag{4}$$

Si el índice de interés es el *throughput* χ_i de una transición t_i (de tasa λ_i), es decir, el número medio de disparos de la transición por unidad de tiempo (índice de productividad), basta con definir la ganancia r_{χ_i} como:

$$r_{\chi_i}(\mathbf{m}) = \begin{cases} \lambda_i, & \text{si } t_i \text{ está sensibilizada en } \mathbf{m} \\ 0, & \text{en otro caso} \end{cases} \tag{5}$$

En el caso en que se decida separar la decisión de conflictos de la duración de las tareas (por ejemplo, el conflicto existente entre las transiciones T_2 y T_5 en la red de la figura 2), pueden utilizarse las transiciones inmediatas para el modelado del conflicto. Véase, por ejemplo, cómo en la red de la figura 3 se modela ahora el conflicto existente en el marcado $(1, 1, 0, 1, 0, 0, 1, 0)$ con las transiciones inmediatas t_2 y t_5.

El diagrama de transición de estados de la figura 5 representa el grafo de alcanzabilidad de la RdPEG de la figura 3. Nótense los cambios con respecto al

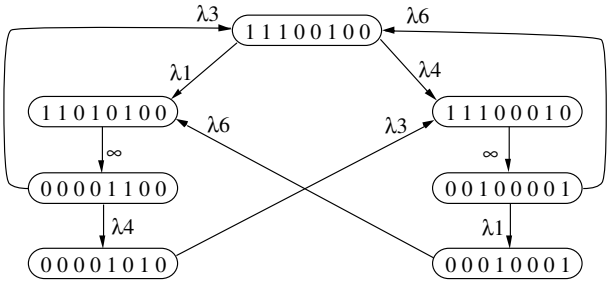

Figura 5. Diagrama de transición de estados de la RdPEG de la figura 3.

diagrama de la figura 4 correspondiente a la RdPM de la figura 2: las tasas λ_2 y λ_5 de las transiciones T_2 y T_5 pasan a tener un valor infinito (las transiciones son ahora inmediatas, es decir, con tiempo de servicio nulo, o lo que es lo mismo, velocidad de disparo infinita). Como consecuencia de éste cambio, el marcado $(1, 1, 0, 1, 0, 0, 1, 0)$ desaparece del espacio de estados alcanzables, y por tanto desaparece el conflicto efectivo entre t_2 y t_5. En otras situaciones, es posible que existan conflictos entre varias transiciones inmediatas y, como ya se ha dicho en la sección 3, tales conflictos se resuelven asociando tasas de encaminamiento a las transiciones (se realiza un *sorteo* entre las transiciones en conflicto para decidir cuál es la que se dispara).

El diagrama de la figura 5 no representa exactamente una CMTC, ya que en ese tipo de procesos estocásticos no se incluye la posibilidad de tasas infinitas. Es posible transformar ese diagrama eliminando los *estados fugaces* del modelo (estados en los que hay una tasa de salida de velocidad infinita), para obtener el *diagrama de transición de estados reducido*, como el de la figura 6, en el que únicamente aparecen los *estados tangibles* del modelo, es decir, aquéllos en los que no hay transiciones inmediatas sensibilizadas y, por tanto, en los que el sistema permanece un tiempo no nulo.

La solución en estado estacionario de este último modelo se obtiene resolviendo el sistema lineal de ecuaciones (1):

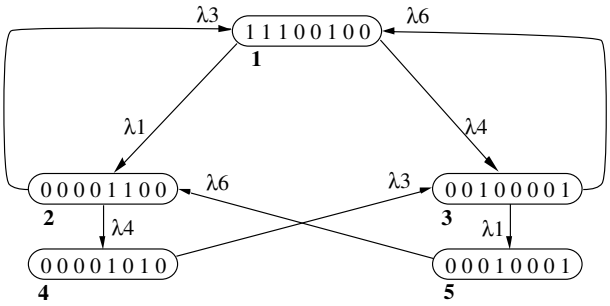

Figura 6. Diagrama de transición de estados reducido de la RdPEG de la figura 3.

$$(\boldsymbol{\pi}_1, \boldsymbol{\pi}_2, \boldsymbol{\pi}_3, \boldsymbol{\pi}_4, \boldsymbol{\pi}_5) \cdot \begin{bmatrix} -(\lambda_1+\lambda_4) & \lambda_1 & \lambda_4 & 0 & 0 \\ \lambda_3 & -(\lambda_3+\lambda_4) & 0 & \lambda_4 & 0 \\ \lambda_6 & 0 & -(\lambda_1+\lambda_6) & 0 & \lambda_1 \\ 0 & 0 & \lambda_3 & -\lambda_3 & 0 \\ 0 & \lambda_6 & 0 & 0 & -\lambda_6 \end{bmatrix} = \mathbf{0}$$

$$\boldsymbol{\pi}_1 + \boldsymbol{\pi}_2 + \boldsymbol{\pi}_3 + \boldsymbol{\pi}_4 + \boldsymbol{\pi}_5 = 1$$

Resolviendo el sistema se puede calcular, por ejemplo, la *tasa de utilización de la memoria común*, que en este caso coincide con la probabilidad en estado estacionario del único estado en el que el lugar p_2 ("memoria común disponible") está marcado:

$$\overline{\boldsymbol{\mu}}[p_2] = \boldsymbol{\pi}_1$$

De forma similar, la *potencia de procesamiento*, P, del sistema, es decir, el número medio de procesadores haciendo un trabajo efectivo (accediendo únicamente a su memoria local correspondiente), se puede calcular definiendo la función de ganancia r_P como:

$$r_P(\mathbf{m}) = \mathbf{m}[p_3] + \mathbf{m}[p_6]$$

Es decir:

$$P = \sum_{\mathbf{m}_i \in \mathrm{RS}(\mathbf{m_0})} r_P(\mathbf{m}_i)\ \pi_i = 2\boldsymbol{\pi}_1 + \boldsymbol{\pi}_2 + \boldsymbol{\pi}_3$$

5 Técnicas tensoriales

La técnica de análisis exacto enumerativo de la sección anterior tiene su origen en los primeros años 80 [3] y es seguramente la más utilizada en la práctica junto con la de simulación [6] (no consideraremos esta última en este documento introductorio). No obstante, la técnica plantea algunos problemas que han

provocado que en los últimos diez años se hayan desarrollado otras alternativas. En muchos casos, el problema fundamental se origina en la *explosión de estados* de la cadena de Markov isomorfa. En modelos de sistemas reales, el tamaño del espacio de estados puede ser tal que resulte difícil no ya resolver el sistema lineal de ecuaciones planteado en la sección anterior, sino incluso, el almacenamiento de la matriz generadora infinitesimal.

La estructura de la red juega un papel importante en las *técnicas tensoriales*. En éstas se expresa la matriz generadora infinitesimal, $\mathbf{Q}$, de una RdPEG en términos de matrices más pequeñas, $\mathbf{Q}_i$, calculadas a partir de ciertas *componentes* (subredes con espacio de estados más pequeño). Esto permite calcular la distribución en estado estacionario, $\boldsymbol{\pi}$, resolviendo el sistema $\boldsymbol{\pi} \cdot \mathbf{Q} = \mathbf{0}$ sin siquiera calcular ni almacenar $\mathbf{Q}$ [30,31,32,29]. En esta *técnica estructurada* se construye un producto cartesiano de estados alcanzables de varias componentes, obteniéndose un *espacio producto*, PS, que incluye el espacio de alcanzabilidad real, RS. El problema fundamental ahora es que, en general, el espacio producto puede ser mucho mayor que el real.

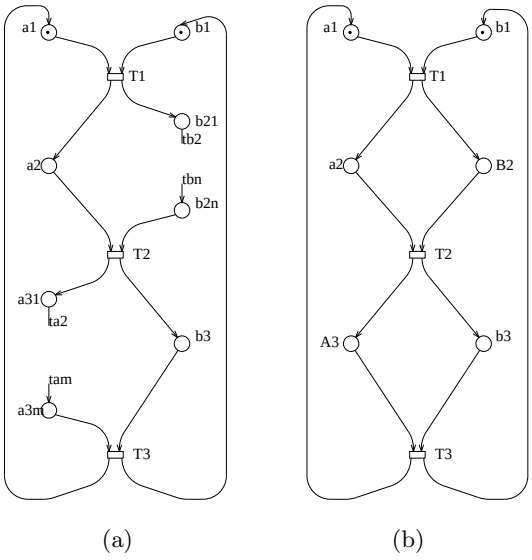

Figura 7. (a) Red compuesta de otras dos sincronizando $T1, T2$ y $T3$ y (b) su representación abstracta.

Para describir brevemente la técnica, consideremos el modelo de la figura 7.a, que muestra una RdPEG, $\mathcal{S}$, que puede considerarse como la composición de otras dos RdPEG's, $\mathcal{S}_1$ y $\mathcal{S}_2$, con tres transiciones comunes, $T1, T2$ y $T3$. Los lugares cuyos nombres empiezan por a definen la componente $\mathcal{S}_1$ y los que empiezan por b definen $\mathcal{S}_2$. Supongamos que hay una secuencia de n lugares y transiciones entre $b21$ y $b2n$ y de m lugares y transiciones entre $a31$ y $a3m$. $\mathcal{S}_1$

tiene, por tanto, $m+2$ estados, mientras que $\mathcal{S}_2$ tiene $n+2$. Se define el espacio producto PS como

$$\mathrm{PS} = \mathrm{RS}_1 \times \mathrm{RS}_2$$

y es evidente que $\mathrm{RS} \subseteq \mathrm{PS}$; de hecho PS tiene $(m+2)\cdot(n+2)$ estados, mientras que el espacio de alcanzabilidad de $\mathcal{S}$ tiene sólo $m+n+1$.

Según la técnica presentada en [31], se puede construir la siguiente matriz $\mathbf{G}$ de tamaño $|\mathrm{PS}| \times |\mathrm{PS}|$:

$$\begin{aligned} \mathbf{G} = \mathbf{Q}'_1 \oplus \mathbf{Q}'_2 \;\; & - \textstyle\sum_{t\in\{T1,T2,T3\}} w(t)[\mathbf{K}_1(t) \otimes \mathbf{K}_2(t)] \\ & + \textstyle\sum_{t\in\{T1,T2,T3\}} w(t)[\mathbf{K}'_1(t) \otimes \mathbf{K}'_2(t)] \end{aligned} \tag{6}$$

donde $\mathbf{Q}'_i$, $\mathbf{K}_i(t)$ y $\mathbf{K}'_i(t)$ (para $i \in \{1,2\}$) son matrices $|\mathrm{RS}_i| \times |\mathrm{RS}_i|$ que pueden obtenerse de los generadores infinitesimales $\mathbf{Q}_i$ de $\mathcal{S}_i$, y los símbolos $\oplus$ y $\otimes$ hacen referencia a operaciones tensoriales (suma y producto de Kronecker) [28].

La idea tras la fórmula anterior es separar el comportamiento de cada componente en: *comportamiento local* (relacionado con las transiciones locales de una única componente) y *comportamiento dependiente* (relacionado con las transiciones de sincronización $T1, T2$ y $T3$). El comportamiento local de cada RdPEG lo representa $\mathbf{Q}'_i$, y, puesto que es independiente, el comportamiento global relacionado con transiciones locales se obtiene mediante una *suma tensorial* de las matrices $\mathbf{Q}'_i$. El comportamiento dependiente requiere, para disparar una transición de sincronización, que ambos sistemas $\mathcal{S}_i$ estén en un estado que sensibilice la transición correspondiente. La matriz de corrección, $\mathbf{K}_i(t)$, de cada transición t, tiene un 1 en cada posición que corresponde a un cambio de estado debido a t en $\mathcal{S}_i$. El *producto tensorial* refleja la condición requerida de que una transición de sincronización se dispara sólo en estados globales que corresponden a estados locales en los $\mathcal{S}_i$ que sensibilizan t. El término con las matrices $\mathbf{K}'_i(t)$ se usa para calcular el fragmento de los elementos de la diagonal que proviene de las transiciones de sincronización.

Por definición de suma y producto tensorial, $\mathbf{G}$ es una matriz $|\mathrm{PS}| \times |\mathrm{PS}|$, y se demuestra en [31,33] que los elementos no nulos del vector $\boldsymbol{\pi}$, solución de la ecuación $\boldsymbol{\pi} \cdot \mathbf{G} = \mathbf{0}$, son la solución en estado estacionario de $\mathcal{S}$. Más aún, se puede implementar un algoritmo de resolución [28,33] que no precisa el cálculo y almacenamiento explícito de $\mathbf{G}$; por tanto, la exigencia mayor de memoria es la precisa para almacenar el vector $\boldsymbol{\pi}$.

La técnica anterior produce un ahorro sustancial de memoria cuando el tamaño de PS, y por tanto de $\boldsymbol{\pi}$, es menor que el número de elementos no nulos de $\mathbf{Q}$. La utilidad práctica de la técnica queda limitada si la diferencia entre RS y PS es grande. Una forma de limitar la creación de *estados espúreos* (que están en PS pero no en RS) es generar una *representación abstracta* del modelo completo que actúe como "filtro" de las soluciones espúreas. Ese modelo abstracto, denominado *esqueleto básico*, limita el espacio producto PS, permitiendo obtener en su lugar un *espacio producto restringido* RPS que se expresa como la unión de productos cartesianos de subconjuntos de los espacios de alcanzabilidad de las componentes. La matriz $\mathbf{Q}$ adquiere una estructura de bloques determinada por

la representación abstracta, y se puede obtener una expresión tensorial de cada bloque de $\mathbf{Q}$ en términos de los bloques de las matrices $\mathbf{Q}_i$.

Por ejemplo, consideremos la red $\mathcal{S}_a$ de la figura 7.b como una representación abstracta de la de la figura 7, con el lugar $B2$ "resumiendo" la estructura de lugares $b21, \ldots, b2n$ y $A3$ "resumiendo" la estructura de lugares $a31, \ldots, a3m$. $\mathcal{S}_a$ tiene tres estados alcanzables: $\mathbf{z}_1 = (\mathsf{a1}, \mathsf{b1})$, $\mathbf{z}_2 = (\mathsf{a2}, \mathsf{B2})$ y $\mathbf{z}_3 = (\mathsf{A3}, \mathsf{b3})$. Los estados de $\mathcal{S}_1$ y $\mathcal{S}_2$ pueden clasificarse, de acuerdo con los estados de $\mathcal{S}_a$. Los $m+2$ estados de $\mathcal{S}_1$ se dividen en tres clases de equivalencia: $\mathrm{RS}_{\mathbf{z}_1}(\mathcal{S}_1) = \{\mathsf{a1}\}$, $\mathrm{RS}_{\mathbf{z}_2}(\mathcal{S}_1) = \{\mathsf{a2}\}$ y $\mathrm{RS}_{\mathbf{z}_3}(\mathcal{S}_1) = \{\mathsf{a31}, \ldots, \mathsf{a3m}\}$. De manera similar, para $\mathcal{S}_2$ se obtiene: $\mathrm{RS}_{\mathbf{z}_1}(\mathcal{S}_2) = \{\mathsf{b1}\}$, $\mathrm{RS}_{\mathbf{z}_2}(\mathcal{S}_2) = \{\mathsf{b21}, \ldots, \mathsf{b2n}\}$ y $\mathrm{RS}_{\mathbf{z}_3}(\mathcal{S}_2) = \{\mathsf{b3}\}$. El espacio producto restringido RPS se obtiene entonces como:

$$\mathrm{RPS}(\mathcal{S}) = \biguplus_{\mathbf{z} \in \mathrm{RS}(\mathcal{S}_a)} \mathrm{RS}_{\mathbf{z}}(\mathcal{S}_1) \times \mathrm{RS}_{\mathbf{z}}(\mathcal{S}_2) =$$
$$= \{\mathsf{a1}\} \times \{\mathsf{b1}\} \cup \{\mathsf{a2}\} \times \{\mathsf{b21}, \ldots, \mathsf{b2n}\} \quad \cup \quad \{\mathsf{a31}, \ldots, \mathsf{a3m}\} \times \{\mathsf{b3}\}$$

donde $\biguplus$ es la unión *disjunta* de conjuntos. Nótese que, para este ejemplo, se obtiene una caracterización precisa del espacio de estados. Sin embargo, en general, la unión de los productos cartesianos puede generar un superconjunto del espacio de estados alcanzables, dependiendo de la "precisión" de la representación abstracta.

Como el espacio de estados ya no es ahora el producto cartesiano de conjuntos de estados locales, sino la unión de productos cartesianos, la expresión de la matriz generadora infinitesimal resulta algo más complicada: se construye una matriz $\mathbf{G}$, de tamaño $|\mathrm{RPS}| \times |\mathrm{RPS}|$, y se considera su estructura de bloques de acuerdo con los estados de $\mathcal{S}_a$. Cada bloque se refiere al conjunto de estados obtenidos mediante un único producto cartesiano, y se puede obtener una expresión tensorial para cada bloque. Los bloques diagonales $\mathbf{G}(\mathbf{z}, \mathbf{z})$ se pueden expresar como

$$\mathbf{G}(\mathbf{z}, \mathbf{z}) = \mathbf{Q}_1(\mathbf{z}, \mathbf{z}) \oplus \mathbf{Q}_2(\mathbf{z}, \mathbf{z})$$

donde $\mathbf{Q}_i(\mathbf{z}, \mathbf{z})$ son las submatrices del generador infinitesimal de $\mathcal{S}_i$ determinadas por los estados cuya representación abstracta es $\mathbf{z}$ ($\mathbf{z} \in \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\}$). Cada bloque $\mathbf{G}(\mathbf{z}, \mathbf{z}')$, con $\mathbf{z} \neq \mathbf{z}'$, $(\mathbf{z}, \mathbf{z}' \in \{\mathbf{z}_1, \mathbf{z}_2, \mathbf{z}_3\})$, describe en cambio el comportamiento que produce un cambio de estado de alto nivel; cada bloque $\mathbf{Q}(\mathbf{z}, \mathbf{z}')$ puede escribirse como

$$\mathbf{G}(\mathbf{z}, \mathbf{z}') = \sum_{t: \mathbf{z} \xrightarrow{t} \mathbf{z}'} w(t)[\mathbf{K}'_1(t)(\mathbf{z}, \mathbf{z}') \otimes \mathbf{K}'_2(t)(\mathbf{z}, \mathbf{z}')]$$

donde $\mathbf{K}'_i(t)(\mathbf{z}, \mathbf{z}')$ son las submatrices del generador infinitesimal de $\mathcal{S}_i$ cuyas filas (columnas) son representadas en forma abstracta por $\mathbf{z}$ ($\mathbf{z}'$), proyectadas para incluir exclusivamente la contribución debida a t. Obsérvese que, en nuestro ejemplo, hay una sola t para cada par $(\mathbf{z}, \mathbf{z}')$.

En [29] se desarrolla la técnica anterior, presentada aquí para un ejemplo, para RdPE arbitrarias (acotadas).

Referencias

1. R. Jain. *The Art of Computer Systems Performance Analysis. Techniques for Experimental Design, Measurement, Simulation, and Modeling.* John Wiley & Sons, 1991.
2. M. Silva. *Las Redes de Petri en la Automática y la Informática.* Editorial AC, Madrid, 1985. In Spanish.
3. M. K. Molloy. Performance analysis using stochastic Petri nets. *IEEE Transactions on Computers*, 31(9):913–917, September 1982.
4. E. Cinlar. *Introduction to Stochastic Processes.* Prentice-Hall, Englewood Cliffs, NJ, 1975.
5. K. Kant. *Introduction to Computer System Performance Evaluation.* Mc Graw-Hill, 1992.
6. M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* J. Wiley, 1995.
7. H. Hermanns, U. Herzog, and V. Mertsiotakis. Stochastic process algebras as a tool for performance and dependability modelling. In *Proceedings of the IEEE International Computer Performance and Dependability Symposium*, pages 102–113, Erlangen, Germany, April 1995. IEEE-Computer Society Press.
8. T. Murata. Petri nets: Properties, analysis, and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.
9. M. Silva. *Introducing Petri Nets*, chapter 1. In [10], 1993.
10. F. DiCesare, G. Harhalakis, J. M. Proth, M. Silva, and F.B. Vernadat. *Practice of Petri Nets in Manufacturing.* Chapman & Hall, London, 1993.
11. G. Chiola, M. Ajmone Marsan, G. Balbo, and G. Conte. Generalized stochastic Petri nets: A definition at the net level and its implications. *IEEE Transactions on Software Engineering*, 19(2):89–107, February 1993.
12. M. Ajmone Marsan, G. Balbo, and G. Conte. *Performance Models of Multiprocessor Systems.* MIT Press, Cambridge, Massachussetts, 1986.
13. M. Sereno and G. Balbo. Mean value analysis of stochastic Petri nets. *Performance Evaluation*, 29(1):35–62, 1997.
14. W. Henderson and P.G. Taylor. Aggregation methods in exact performance analysis of stochastic Petri nets. In *Proc. 3^{rd} Intern. Workshop on Petri Nets and Performance Models*, pages 12–18, Kyoto, Japan, December 1989. IEEE-CS Press.
15. G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo. GreatSPN 1.7: GRaphical Editor and Analyzer for Timed and Stochastic Petri Nets. *Performance Evaluation*, 24:47–68, 1995.
16. M. Silva and J. Campos. Performance evaluation of DEDS with conflicts and synchronizations: Net-driven decomposition techniques. In *Proceedings of the 4th International Workshop on Discrete Event Systems*, pages 398–413, Cagliari, Italy, August 1998. IEE Control.
17. G. Chiola, S. Donatelli, and G. Franceschinis. GSPNs versus SPNs: What is the actual role of immediate transitions? In *Proceedings of the 4^{th} International Workshop on Petri Nets and Performance Models*, pages 20–31, Melbourne, Australia, December 1991. IEEE Computer Society Press.
18. G. Chiola, C. Dutheillet, G. Franceschinis, and S. Haddad. Stochastic well-formed coloured nets for symmetric modelling applications. *IEEE Transactions on Computers*, 42(11), November 1993.
19. J. Campos, G. Chiola, and M. Silva. Ergodicity and throughput bounds of Petri nets with unique consistent firing count vector. *IEEE Transactions on Software Engineering*, 17(2):117–125, February 1991.

20. J. Campos, G. Chiola, and M. Silva. Properties and performance bounds for closed free choice synchronized monoclass queueing networks. *IEEE Transactions on Automatic Control*, 36(12):1368–1382, December 1991.
21. J. Campos, G. Chiola, J. M. Colom, and M. Silva. Properties and performance bounds for timed marked graphs. *IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications*, 39(5):386–401, May 1992.
22. J. Campos and M. Silva. Structural techniques and performance bounds of stochastic Petri net models. In G. Rozenberg, editor, *Advances in Petri Nets 1992*, volume 609 of *Lecture Notes in Computer Science*, pages 352–391. Springer-Verlag, Berlin, 1992.
23. J. Campos and M. Silva. Embedded product-form queueing networks and the improvement of performance bounds for Petri net systems. *Performance Evaluation*, 18(1):3–19, July 1993.
24. Z. Liu. Performance bounds for stochastic timed Petri nets. In G. De Michelis and M. Diaz, editors, *Application and Theory of Petri Nets 1995*, volume 935 of *Lecture Notes in Computer Science*, pages 316–334. Springer-Verlag, Berlin, 1995.
25. J. Campos, J. M. Colom, H. Jungnitz, and M. Silva. Approximate throughput computation of stochastic marked graphs. *IEEE Transactions on Software Engineering*, 20(7):526–535, July 1994.
26. Y. Li and C. M. Woodside. Complete decomposition of stochastic Petri nets representing generalized service networks. *IEEE Transactions on Computers*, 44(8):1031–1046, August 1995.
27. C. J. Pérez-Jiménez and J. Campos. On state space decomposition for the numerical analysis of stochastic Petri nets. In *Proceedings of the 8th International Workshop on Petri Nets and Performance Models*, pages 32–41, Zaragoza, Spain, September 1999. IEEE Computer Society Press.
28. M. Davio. Kronecker products and shuffle algebra. *IEEE Transactions on Computers*, 30(2):116–125, 1981.
29. J. Campos, S. Donatelli, and M. Silva. Structured solution of asynchronously communicating stochastic modules. *IEEE Transactions on Software Engineering*, 25(2):147–165, March 1999.
30. P. Buchholz. A hierarchical view of GCSPN's and its impact on qualitative and quantitative analysis. *Journal of Parallel and Distributed Computing*, 15(3):207–224, July 1992.
31. S. Donatelli. Superposed generalized stochastic Petri nets: Definition and efficient solution. In R. Valette, editor, *Proceedings of the* 15^{th} *International Conference on Applications and Theory of Petri Nets*, volume 815 of *Lecture Notes in Computer Science*, pages 258–277. Springer-Verlag, Berlin Heidelberg, 1994.
32. P. Kemper. Numerical analyisis of superposed GSPN. *IEEE Transactions on Software Engineering*, 22(4):615–628, September 1996.
33. B. Plateau. On the stochastic structure of parallelism and synchronization models for distributed algorithms. In *Proceedings of the 1985 SIGMETRICS Conference*, pages 147–154, Austin, TX, USA, August 1985. ACM.